iyatı .....: 225.000
SATIŞ FİYATI .....: 300.000

TOPTAN SATIŞ
İstanbul Devlet Kitapları Müdürlüğü, Adana, Ankara, Elazığ, Erzurum, İzmir, Samsun, Trabzon ve Van Bölge Şeflikleri.

PERAKENDE SATIŞ
İlli Eğitim Yayınevleri ve Bakanlık yayınları satıcısı kitapçılar.

7

MİKROİŞLEMCİLER VE SAYISAL SİSTEMLER

1994

MİKROİŞLEMCİLER VE SAYISAL SİSTEMLER

Yoğun İşlemciler
Sayısal Devreler

5

30 01 97
F.B

2726 — 87

ENDÜSTRİYEL OKULLAR İÇİN

MİKROİŞLEMCİLER VE SAYISAL SİSTEMLER

Douglas V. HALL
Instructor, Portland Community College,
Portland, Oregon, Design Engineer, Hall
Electronics Consultants

ETAM A.Ş. Matbaa Tesisleri, ESKİŞEHİR - 1994

ÖNSÖZ

Varlıklarını sürdürmek isteyen toplumlar, kalkınmanın gerektirdiği sayıda nitelikli insan gücünü yetiştirmek için eğitime değer vermek ve ona bilimsel ve teknolojik bir nitelik kazandırmak mecburiyetindedirler.

Eğitim, Cumhuriyetin kuruluşundan beri ülkemizde yenileşme aracı olarak görülmüştür. Bugün Eğitim Sistemimiz, bilim çağına girilen dünyamızda, toplumumuzun büyüyen ve çeşitlenen ihtiyaçlarına cevap vermede bir takım problemlerle karşı karşıyadır.

Eğitimle ilgili problemlerin çözümünde, yeni yöntemler, teknikler ve araçlar geliştirmek için araştırmalar yapmak, ayrıca daha önce yapılmış araştırmalar sonucu geliştirilen bilgi ve teknolojiyi ülkemize getirmek zorundayız.

Eğitime ayrılacak finansman kaynaklarının sınırlı olması, ülkemizi, genel bütçe dışındaki imkanlardan faydalanmaya zorlamaktadır. Devletimiz bu imkanları araştırmış, mesleki ve teknik öğretim kurumlarımızın bilim ve teknolojide meydana gelen gelişmelere paralel olarak modernleştirilmesi için Uluslararası İmar ve Kalkınma Bankası (Dünya Bankası - IBRD) ile yapılan İkraz Anlaşmasıyla Endüstriyel Okullar Projesi uygulamaya konulmuştur.

Bu projenin amaçları; Endüstriyel Okulların yeni teknoloji ürünü makina ve teçhizatla donatılarak yenilenmesi, çeşitli meslek alanlarında müfredat programlarının geliştirilmesi, burslar ve yurt dışından danışman temin edilmesi yoluyla öğretmenlerimizin eğitilmesi ve çeşitli meslek alanlarında ders kitaplarının tercüme ve yayın haklarının satın alınarak Eğitim Sistemimize kazandırılmasıdır.

Proje ile belirlenen hedeflere büyük ölçüde ulaşılmıştır. Projenin amaçlarından biri olan çeşitli meslek alanlarında (Hidrolik-Pnömatik, Soğutma ve İklimlendirme, CNC, Döküm, Elektronik, Bilgisayar, PLC ve Metal İşleri) teknik ders kitapları, uzmanlardan kurulu komisyonlarca seçilmiş ve tercüme edilerek yayımlanmıştır.

Büyük kaynak ve emek harcayarak Eğitim Sistemimize kazandırdığı mız kitapların öğretmen ve öğrencilerimize faydalı olmasını dilerim.

Salih ÇELİK
Projeler Koordinasyon
Kurulu Başkanı

İÇİNDEKİLER

BÖLÜM 1 TEMEL ÖRNEK GELİŞTİRME VE TEST DEVRELERİNİN KULLANIMI 1
Temel Örnek Geliştirme 1
Sayısal Voltmetreler ve Mantık Probları 11
Osiloskoplar 12
Diğer Test Devreleri 28

BÖLÜM 2 SAYISAL MANTIK KAPISI KARAKTERİSTİKLERİ VE BAĞDAŞTIRMA 33
Analoga Karşın Sayısal 34
İkilik Kod : İki Tabanlı Sayı Sistemi 34
Temel Mantık Kapıları 38
Mantık Ailelerinin Karakteristikleri 42
Giriş Sinyal Uyumlanması ve Sinyal Kaynakları 81
İletim Hatları 90
Mantık Ailelerini Birbirlerine Bağdaştırma 96
Mantık Kapılarını Basit Gösterge ve Rölelerle Bağdaştırma 98

BÖLÜM 3 BİRLEŞİK MANTIK, VERİ SEÇİCİLER, KODLAR VE ROMLAR 108
Mantık Kapılarını Birleştirme 108
Boole Cebiri 110
Negatif Mantık 118
Karma Mantık 121
Karnaugh Haritalaması 127
Basit Mantık Kapı Devrelerinin Sentezi 134
Anahtarlar, Veri Seçiciler ve Veri Dağıtıcılar 140
İkilik Tabanlı Kodlar 151
Kod Çeviriciler 164
Yalnız - Okunur Bellekler 172
PLA lar, FPLA lar, FPGA lar ve PAL lar 180

BÖLÜM 4 FLİP-FLOPLAR, SAYICILAR VE KAYDEDİCİLER 192
Mandallar ve Flip-Floplar 195
Asenkron ve Dalgacık Sayıcılar 208
Senkron Sayıcılar 220
Frekans Sayıcılar 224
Çoğullamalı Göstergeler 229
Bir LSI Frekans Sayıcı 233
Kaydediciler ve Kaydırmalı Kaydedicileri 236

BÖLÜM 5 OKU - YAZ BELLEKLER 263
Rastgele Erişilebilir Bellek 263
Zamanlama 269
Uçucu Olmayan RAM lar 273
Dinamik MOS RAM lar 274
Josephson Kavşağı Bellekleri 282
Kaydırma Kaydedici Bellekler 282
Manyetik Balon Bellekler 284
Manyetik Çekirdekli Bellek 287
Manyetik Bantta Saklama 289
Manyetik Diskte Veri Saklama 292
Elektronik Işınla Erişilen Bellek 299
Yarıiletken Belleklerin Test Edilmesi 300

BÖLÜM 6 D/A VE A/D ÇEVİRİCİLER 311
Sayısaldan - Analoga Çeviriciler 312
D/A Çeviricinin Karakteristikleri, Hata Kaynakları ve Özelliklikleri 319
Analogdan Sayısala Çevirme 321
Örnekle - ve - Tut Devreleri 332
Sayısal Göstergeli Voltmetreler ve Multimetreler 333
Sayısal Ses İletimi ve CODEC ler 338

BÖLÜM 7 SAYISAL ARİTMETİK 342
Toplama 342
Çıkarma 348
Aritmetik Mantık Birimi 356
İkilik Çarpma ve Bölme 361
Sabit - Virgüllü Sayılar 366
Kayan - Virgüllü Sayılar 368

BÖLÜM 8 MİKROİŞLEMCİ YAPISI VE PROGRAMLAMA 375
Bir Sayısal Bilgisayarda Genel Yapı ve Program Akışı 376
Mikroişlemci CPU ları : 8080A ve 8085A Mikroişlemcileri 380
8080A/8085A Komut Kümesi 384
Çeviri Dilinde Program Yazma 403
Çeviri Dili Programlama Örnekleri 405
Program Yürütme Zamanı ve Geciktirme Sabitleri 413
Mikroişlemcili Geliştirme Sistemleri 417
Çevirici Yöndendiricileri veya Sözde Komutlar 419
Çeviri Dili Programlarında Arıza Bulma 422

BÖLÜM 9 8080A /8085A SİSTEM DONANIMI VE ZAMANLAMASI 428
Bir 8080A Mikroişlemci Sistemi 429
Bir 8085A Mikroişlemci Sistemi 438
Mikrobilgisayar Geliştirme Kartları : SDK - 85 441
Adres Çözme 449
Port Kodu Çözme 452
Mikrobilgisayar Zamanlama Parametreleri 457
Bekleme, Tutma ve Durma Durumları 460
Kesmeler 463
Maskeleme ve Yetkilendirme 465
Bir 8080A veya 8085A Mikrobilgisayar Sisteminin Arızasını Bulma 470
Zilog Z80 476
Z80A Komut Kümesi 477
Z80A Zamanlaması ve Sistem Bağlantıları 479
Tek - yonga Mikroişlemciler 480
İntel 8048 Ailesi 480

BÖLÜM 10 MC6800 MİKROİŞLEMCİSİ VE MİKROİŞLEMCİ GELİŞİMİ 489
MC6800 Mikroişlemcisi 490
MC6800 Komut Kümesi 492
Program Örnekleri 503
Basit bir MC6800 Sistemi 511
6800 Sistem Zamanlaması 519
6800 ün Kesmeleri 521
Bir 6800 Tabanlı Mikrobilgisayar Sisteminde Arıza Bulma 524
MC6800 ün Soyundan Gelenler 525
MOS Technology 6500 Mikroişlemci Ailesi 528
Süper İşlemciler 537

BÖLÜM 11 ARABİRİMLER ÜZERİNDEN GİRİŞ VE ÇIKIŞLARIN BAĞLANTILARINI KURMA 544
Basit, Taramalı ve Kesmeli I/O 545
Bir Mikrobilgisayar ile Onaylı Giriş ve Çıkış 553
Onay Sinyallerinin Üretilmesi 554
Klavyeleri Mikrobilgisayarlara Arabirimlerle Bağlama 564
Yedi - Parçalı Göstergelerin Mikrobilgisayarlara Arabirimlerle Bağlanması 571
8279 un İlk Kullanıma Hazırlanması 581
Likid Kristal Göstergelere Arabirimlerle Bağlanma 585
Güç Devreleri 586

D/A Çeviricileri Arabirimlerle Bağlama ve Uygulamaları 589
A/D Çeviricileri Arabirimlerle Bağlama ve Uygulamaları 590
Ortak Yol ve Veri Haberleşmesi Standartları 594
MODEM ler ve RS-232C 606
Programlanabilir Seri Port Entegre Devreleri 610
Yazılım Kullanarak Seri Giriş ve Çıkış 622
Bir CRT Monitörünü Bir Mikroişlemciye Bağlama 624

BÖLÜM 12 PARÇALARI BİR ARAYA KOYMA 639
Çalışma Teorisi : Donanım 640
Çalışma Teorisi : Yazılım 645
Temel Altyordamlar 662
MC6800 Tabanlı Bir EPROM Programlayıcı 669
Endüstriel Süreç Kontrolunun Gözden Geçirilmesi 687
SDK - 85 - Tabanlı bir Süreç Kontrol Sistemi 690
Kontrolör Programı 696

BÖLÜM 13 MİKROİŞLEMCİ TABANLI SİSTEMLERİN TEMEL ÖRNEKLERİNİN GELİŞTİRİLMESİ VE ARIZA GİDERİLMESİ 709
Bir Temel Örnek Yapılması ve Hata Ayıklanması 709
Program Modüllerinin Test Edilmesi, Hatalarının Ayıklanması ve Bağlanmaları 716
Bir Temel Örneği Geliştirme Kartı ve Monitör Programı Olmadan Geliştirme, Test Etme ve Hatalarını Ayıklama 723
Devre-içi bir Emülatör ya da Simülatör ve bir Geliştirme Sistemi Kullanarak Donanım ve Yazılım Geliştirmesi 728
Üretim Test Yöntemleri 733
Mikroişlemci Tabanlı Aletler İçin Üretim Testi ve Arazi Servisi Aletleri ve Teknikleri 734
Tasarım İçin Kendi Kendine Test ve İşaret Analizi 737

BÖLÜM 14 YÜKSEK - SEVİYELİ DİLLER 742
Yüksek-Seviyeli Dil Programlarını Yazmak ve Yürütmek İçin Kullanılan Sistemler ve Programlar 743
Yaygın Yüksek-Seviyeli Diller 746
BASIC ile Programlama 749
Program Örnekleri 756
Karakter Dizileri 762
FORTRAN ile Programlama 766
Program Şekli ve Deyimler 770
Bir FORTRAN Programının Derlenmesi ve Çalıştırılması 780
PASCAL ile Programlama 781
Operatörler 786
Deyimler 787
Mikrobilgisayar-Tabanlı Üretimin Geliştirilmesindeki Eğilimler 794

EKLER :
A. DURUM DİYAGRAMLARI VE ALGORİTMİK DURUM MAKİNE ŞEMALARI 802

B. YENİ MANTIK SEMBOLLERİNİN AÇIKLAMASI 805
C. VERİ SAYFALARI 836
D. VOTRAX SC - 01 KONUŞMA SENTEZLEYİCİ FONEM TABLOSU 863

İNDEKS 865
TERİMLER SÖZLÜĞÜ 884

ÖNSÖZ

Elektronik Teknolojisinin ve elektronik endüstrisinin patlayan bir biçimde gelişmesi tek bir ders kitabının ve hatta bir dizi dersin bile elektronik teknisyenine mesleği süresince gerekli bütün bilgileri vermesini çok zor hale getirmiştir. Bu nedenle, sahasında başarılı olmak ve onunla büyümek isteyen teknisyen için iki yetenek özellikle önemlidir. Birincisi, üreticinin veri kitaplarını, uygulama notlarını ve bakım el kitaplarını okumak ve anlamak yeteneğidir. İkincisi ise, ki bu endüstriden gelen bilgilerle desteklenmektedir, elektronik teksinyenlerinin test cihazlarını kullanmada akıcı olmaları ve arıza bulmadaki sistemli yaklaşımları anlamalarıdır.

Mikroişlemciler ve Sayısal Sistemlerin bu ikinci baskısı ve Mikroişlemciler ve Sayısal Sistemlerle Deneyler isimli ilgili laboratuvar kılavuzu sayısal ve mikroişlemci sistemleri alanında bu yetenekleri geliştirmenize yardımcı olacaktır.

Bu kitap spiral şeklinde bir yaklaşım kullanır. Yani, bir konu temelde anlaşılabilmesi için gerekli miktardaki ayrıntılarıyla tanıtılmakta ve sonra bu konuyla tekrar karşılaşıldığında konuya daha fazla derinlik eklenmektedir.

Test cihazlarının kullanılmalarındaki yetenekleri geliştirmek üzere, kitap, sayısal devrelerin arızalarının bulunmasında sık kullanılan mantık probu, sinval üreteci ve osiloskop gibi test cihazlarının kullanılması üzerine bir bölümle başlamaktadır. Mantık analizorlerinin kullanımı 5. ve 13. Bölümlerde anlatılmıştır. Bölüm 13 ayrıca geliştirme sistemlerinin, devre içi emulatörlerin ve imza analizörlerinin kullanımını da anlatmaktadır.

Sistematik arıza bulma yeteneklerini geliştirmek için, bu kitap başından sonuna kadar bir sistemler yaklaşımı kullanmaktadır. Bir elektronikçi sadece parçaların görevlerini değil, aynı zamanda parçaların nasıl birleştirildiğini ve bir sistem olarak nasıl çalıştığını bilmek zorundadır. Bir sayısal saat, bir frekans sayıcısı ve art arda yuvarlama metoduyla çalışan bir A/D çevirici sayısal sistemlerin örnekleri olarak ayrıntılarıyla açıklanmaktadır. Bir mikroişlemci tabanlı terazi, EPROM programlayıcısı ve endüstriyel kontrol cihazı ise mikroişlemci tabanlı sistemlerin örnekleri olarak ayrıntılarıyla açıklanmaktadır. Bu sistemler için sistematik temel örnek geliştirme ve/veya arıza bulma yöntemleri de anlatılmaktadır.

Üretici firmaların kataloglarını yorumlama yeteneğini geliştirmek için bu kitap en yeni veri sayfalarını kullanmakta ve bunlarda bulunan parametreleri açıklamaktadır. Son zamanlarda sayısal veri kitaplarına eklenmiş olan yeni IEEE/IEC mantık sembolleri tanımlanıp açıklanmaktadır. Texas Instruments firmasının TTL Data Book for Desing Engineers (Tasarım Mühendisleri için TTL Veri Kitabı), National Semiconductor firmasının Logic Databook (Mantık Veri Kitabı), veya Signetics firmasının Logic-TTL Manual (Mantık-TTL Kılavuzu) gibi standart TTL veri kitapları, bu veri kitaplarıyla daha fazla yakınlık kazanmak için sayısal sistemler hakkındaki bölümlerle birlikte kullanılmalıdır. Intel firmasının MCS 80/85 Family User's Manual (MCS 80/85 Ailesinin Kullanıcı Kılavuzu), Intel firmasının Components Catalog (Parça Kataloğu) ve Motorola firmasının Microprocessors Data Manual (Mikroişlemciler Veri Kılavuzu) nun son baskıları bu kitaptaki mikroişlemciler hakkındaki bölümlerle birlikte kullanılmalıdır.

Bir teknisyen için devreyi kurmayı ve test etmeyi bilmeden devre teorisini bilmenin fazla bir yararı olmayacağı kabulüne dayalı olarak, bu kitap test cihazlarının doğru ve yanlış kullanımını ayrıntılı bir şekilde açıklayarak başlamaktadır. Bunu takiben, Bölüm 2 ikilik sayıları, temel kapıları, bilinen mantık ailelerinin ayrıntılı karşılaştırmalarını, bağdaştırma devrelerini, sinyal uyumlaştırmasını ve sayısal mantıkta sık karşılaşılan problemleri tanıtmaktadır.

3 ten 7 ye kadar olan bölümler basit birleşik mantıktan başlayıp kod çevrimlerine, ROM lara, sayıcılara, kaydedicilere, RAM lara, A/D çeviricilere ve aritmetik mantık birimlerine ilerlemektedir. Bu bölümlerin tamamlanmasıyla, mikroişlemcilerde kullanılan bütün parçalar tamamen kapsanmış olacaktır.

Bir sayısal bilgisayarın veya mikroişlemci sisteminin genel yapısıyla başlayan Bölüm 8, hızlı bir şekilde 8080A/8085A mikroişlemcisinin yapısına ve çeviri dili programlamasına geçmektedir. Deneyimler göstermektedir ki teknisyenlere bir veya iki özel mikroişlemciyi ayrıntılarıyla öğretmek, özel olmayan veya gerçekte var olmayan aygıtları kullanarak genel bir yaklaşımla öğretmeye çalışmaktan çok daha etkilidir. Bir defa bir veya iki mikroişlemci ailesinin yapısı, programlaması ve problemlerini iyice tanıdıktan sonra gerektikçe diğer aileleri öğrenmek oldukça kolaydır. Spiral çalışma yaklaşımı Bölüm 9 da 8080A/8085A tabanlı sistemlerin donanımının ayrıntılı açıklamalarıyla devam etmektedir. Bölüm 8 ve 9, birlikte 8080A/8085A tabanlı sistemlerin iyi bir şekilde anlaşılmasını sağlar. Ayrıca Bölüm 9, Z80 ve 8048 mikroişlemcilerinin yapılarını ve komut kümelerini kısaca tanıtmaktadır.

Sizleri ikinci önemli mikroişlemci ailesiyle tanıştırmak üzere Bölüm 10, 6800 tabanlı mikroişlemci sisteminin yapısını, komut kümesini ve donanımını anlatmakta ve bunları 8080A/8085A sistemininkilerle karşılaştırmaktadır. Yazılımı öğrenme spirali, indislenmiş adresleme modlarını tanıtmakla devam etmektedir. Bölüm 10, 6502 mikroişlemcisinin yapısı ve komut kümesini anlatmaktadır ve sonra tanınan üç 16-bitlik mikroişlemciyi karşılaştırmaktadır.

Mikroişlemcilerin birçok uygulaması dış dünya ile bağdaşmasını gerektirdiğinden, Bölüm 11, klavyeler, LED göstergeler, katot ışınlı tüpler ve A/D çeviriciler dahil olmak üzere çeşitli giriş ve çıkış aygıtlarının ve yöntemlerinin analizini yapmaktadır. Bir çok programlanabilir çevre aygıtının hazırlanması ve kullanımları tanıtılmaktadır. Intel ve Motorola gibi her üreticinin mevcut programlanabilir çevre aygıtları hakkında kalın kitapları vardır. Bütün bunları burada anlatmanın imkanı yoktur, fakat birkaç özel aygıtın hazırlanmasını öğrenirseniz, diğerlerininkine ait ayrıntıları da gerektikçe üreticilerin veri kitaplarından öğrenebilirsiniz.

Bölüm 12, daha önceki bölümlerde tartışılan bu giriş/çıkış aygıtlarını ve mikroişlemcileri birleştirerek üç mikroişlemci tabanlı sistemi eksiksiz anlatmaktadır: Bir 8086A tabanlı "akıllı" terazi, bir 6800 tabanlı EPROM programlayıcısı ve bir 8085A tabanlı endüstriyel kontrol cihazı. Bu sistemlerin devreleri, akış şemaları ve çeviri dilindeki programlarının analizleri yapılmaktadır. Daha sonra bunlar Bölüm 13 te LSI sistemlerinin arızalarının bulunmasını öğretmekle kullanılmaktadır. Hem yazılım hem de donanım arızalarının bulunmasına ait teknikler geliştirilmiştir. Intel SDK-85, Motorola MEK 6800 D2 ve Motorola 6800D5 gibi nisbeten ucuz mikroişlemci geliştirme kartlarıyla kitapta verilen bir programı adım adım çalıştırabilmeniz ve adres, veri ve kontrol hattı sinyallerini gözleyebilmeniz için yeterli özel bilgi verilmektedir. Elinizde bir mantık analizörü varsa, basit bir test programı halkası çalıştırılabilir ve gözleyebilirsiniz.

Bölüm 14 yüksek-seviyeli programlama dillerini tanıtmaktadır. Belleklerin bit başına maliyetlerinin düşmesi ve çeviri dilinde geliştirilmiş programların yüksek maliyeti, zaman yönünden daha ekonomik olan yüksek-seviyeli dillerin programlamada daha fazla kullanılmasına neden olmuştur. Bir sistem programının büyük bir kısmı yüksek-seviyeli bir dilde yazılabilir ve programlanabilir aygıtları hazırlamada, zamana bağlı bölümlerde ve kesme servis altprogramlarında ihtiyaç duyulan çeviri dili moduller programa eklenebilir. Bu bölümdeki örneklerin çoğunda, çok geniş kullanımı olduğundan ve çoğu ucuz ev bilgisayar sistemlerinde bulunduğundan BASIC dili kullanılmıştır. Ayrıca FORTRAN ve PASCAL dilleri de tanıtılmaktadır.

Yazdıklarımın tenkidini yapan, sayısız parçaları birleştirerek bu kitabı oluşturan ve daha birçok sayısız katkısı olan eşim Marybelle B. Hall'a, mantık sembol gösterilişlerine olan katkılarından dolayı Texas Instruments, Inc. firmasından Fred Mann'a, kitabımı baştan sona inceleyen DeVry Institute of Technology, Irving, Texas'ta çalışan Profesor John L. Morgan'a; Portland Community College'daki meslekdaşlarıma: Birçok önerilerde bulunan ve endüstriden bilgi akışını sağlayan Okutman Davit M. Hata'ya, 6 ve 14 üncü

bölümleri inceleyen Dr. Richard Morris'e, 1-3 bölümlerini inceleyen William O'Dell'e; Bölüm 12 deki endüstriyel kontrol cihazını kurma ve test etmede yardımcı olan arkadaşım Micheal Hudson'a, kitabı daktilo eden ve bu arada bilgisayarı öğrenen arkadaşlarım Rosemary Griffeth ve Melody Reasoner'a en derin teşekkürlerimi sunmak isterim.

Son olarak, bana ihtiyaçlarını bildiren birçok yeni öğrencime ve elektronik endüstrisinde çalışmakta olan eski öğrencilerime teşekkür etmek isterim.

Bu kitabı kullananlardan, çok faydalı olacağına inandığım yorum, eleştiri ve önerilerini duymaktan çok memnun olacağım.

Douglas V. Hall

1. BÖLÜM

TEMEL ÖRNEK GELİŞTİRME VE TEST DEVRELERİNİN KULLANIMI

Elektronik teknisyeni veya öğrencisi olarak bir gün bir sayısal devreyi kurmak ve test etmek zorunda kalacaksınız. Bu, bir temel örnek geliştirmek veya var olan bir devrede değişiklik yapmak için olabilir. Devrenin başarılı olması, sadece onun tasarımına değil ayrıca elemanları yerleştirirken ve devreyi kurarken göstereceğiniz dikkate bağlıdır. Kuş yuvası gibi parçaların aceleyle bir yere yığılması genellikle çalışma ihtimali olmayan, arıza bulması zor ve kullanımı güvenilmez bir devre ortaya çıkarır. Bu nedenle, bir devreyi kurarken dikkate alınması gereken ilk konu temel örnek devrenin veya değişiklik devresinin uygun şekilde inşa edilmesidir.

Bu bölümün sonunda;

1. Üzerinde temel örneği kurabileceğiniz malzemeyi seçebileceksiniz.

2. Renk kodlu teller kullanarak ve etkili yangeçitlerle bir devreyi düzgün bir şekilde yerleştirip kurabileceksiniz.

3. Bir osiloskop, bir sayısal voltmetre veya bir mantık probu kullanarak temel mantık durumlarını test edebileceksiniz.

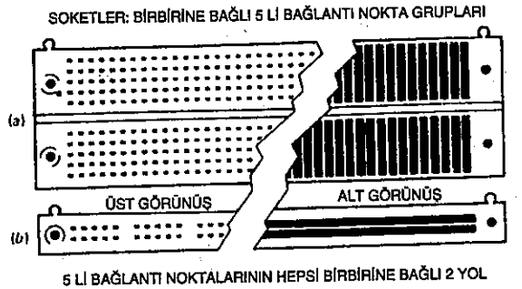
4. Sinyal kaynağı olarak darbe üretecini veya zaman-işaret üretecini kullanabileceksiniz.

TEMEL ÖRNEK GELİŞTİRME

KART SEÇİMİ

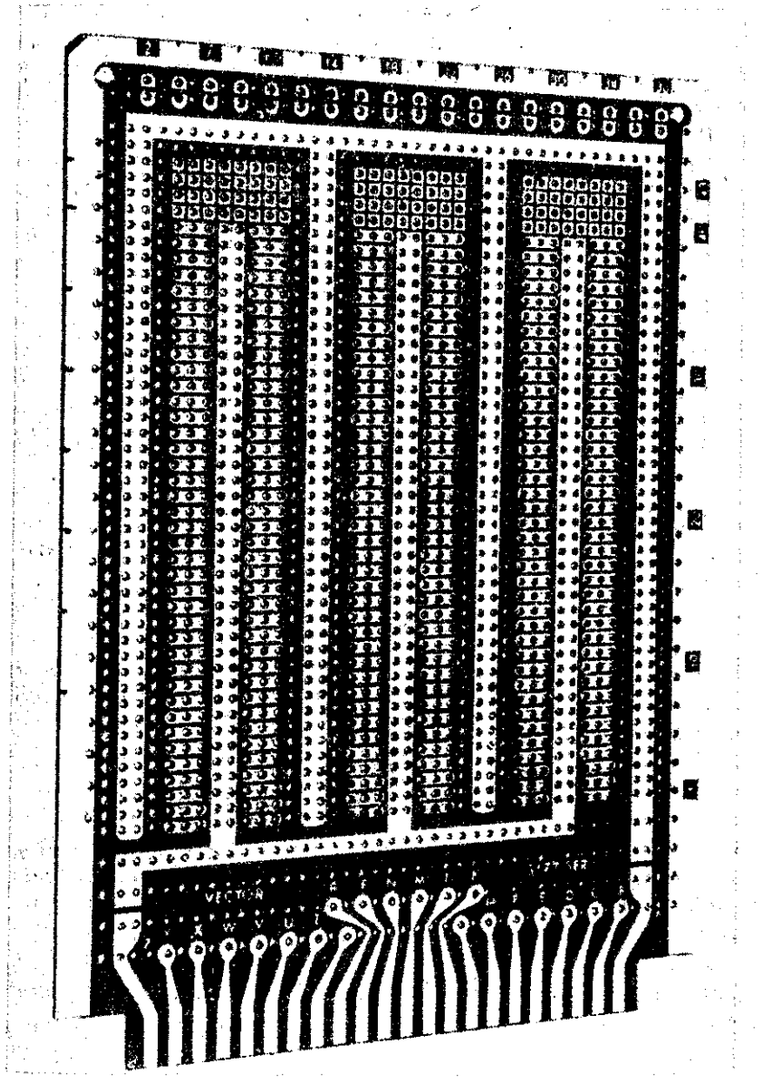
Temel örnek veya değişiklik devrelerini üzerinde kurabileceğiniz birçok malzeme vardır. Seçilecek malzeme devrenin istenen ömrüne ve kullanılışına bağlıdır. Sabit bir devre gerektirmeyen ve çalışma frekansı 10 MHz nin altında olan laboratuvar kullanımı veya diğer kısa süreli uygulamalar için Global Specialties Corporation (Şekil 1-1 e bakınız), E & L Instruments, Inc. ve benzeri firmalardan sağlanabilecek lehimsiz temel örnek kartlarının kullanılması uygundur. Parçalar diğerleriyle birleştirilerek istenilen genişlikte bir alan oluşturabilmektedir. Şekil 1-4 te görüldüğü gibi entegre devreleri ve diğer parçaları takmada soketler kullanılmaktadır. Şekil 1-1 a da görüldüğü gibi, düşey yönde beşli gruplar halinde bulunan delikler elektriksel olarak birbirlerine bağlıdır. Böylece entegre devrenin her bacağına dört tel veya eleman ucunun bağlanmasına imkan verilmiştir. Güç ve toprak hattı için kullanılan taşıt parçaları çoğunlukla ayrı parçalar halindedir (Şekil 1-1b ye bak). Yol par-

çalarında, iki bağımsız iletken, parça boyunca uzanır. Yol parçaları, soket parçaları arasına gerektiği gibi konulabilir. Örneğin, 2 yol parçası toprak, +5, -12 ve +12V hatları için kullanılabilir. Bu kartlar tekrar tekrar kullanılabildiklerinden uzun süreli maliyetleri düşüktür.

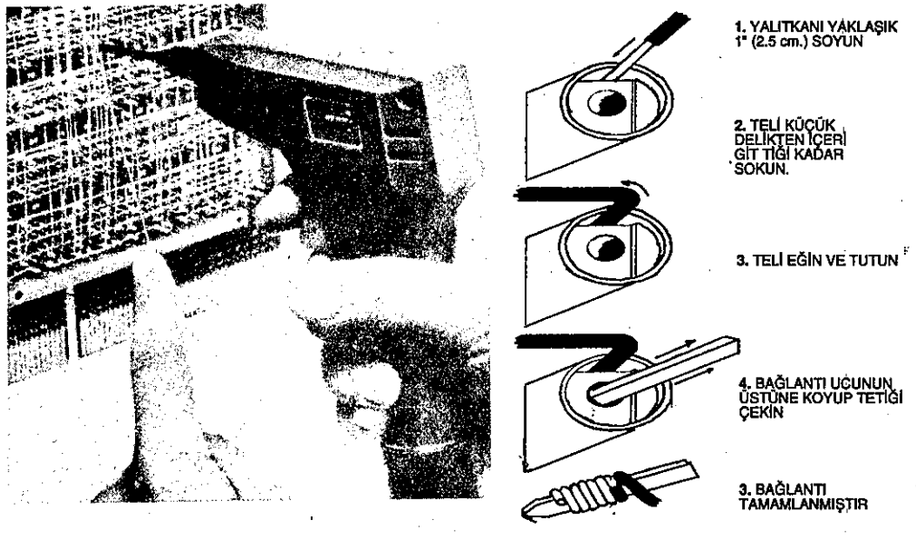


Şekil 1.1 *(a)* Lehimsiz temel örnek kartı entegre devre soket parçası ve iç yapısı. *(b)* Lehimsiz temel örnek kartı yol parçası ve iç yapısı. (Global Specialties Corporation'ın müsadeleriyle.)

Sistem temel örneğinin bir parçası olan veya bir devrede yerinde yapılacak bir değişiklik gibi sabit işlerde, Douglas Electronics veya Vector Electronic Company, Inc. gibi firmalardan sağlanabilecek üniversal baskılı devre kartları (Şekil 1-2) daha kullanışlıdır. Bu kartlar birçok boy ve türde gelmektedir. Bunlar standart elemanlara (entegre devreler ve transistorler gibi) ve soketlere uyacak şekilde önceden delinmişlerdir ve çoğu standart kenar konnektörlerine sahiptir. Güç ve toprak yolları kart üzerindedir. Bağlantılar, ya bağlantı tellerini lehimleyerek veya tel sargı soketleri takıp tel sargı bağlantıları kullanarak yapılır. Tel sargı bağlantıları, telin çıplak ucunu tel sargı soketinin üzerindeki bacağa bir tel sargı aleti veya tabancasıyla sıkıca sararak yapılan bağlantılardır. (Şekil 1.3 *a* ya bakınız). Eğer Şekil 1.3*b* de görüldüğü üzere gereği gibi yapılmışsa, bu bağlantılar lehimlenmiş bağlantılar kadar güvenilirdir.



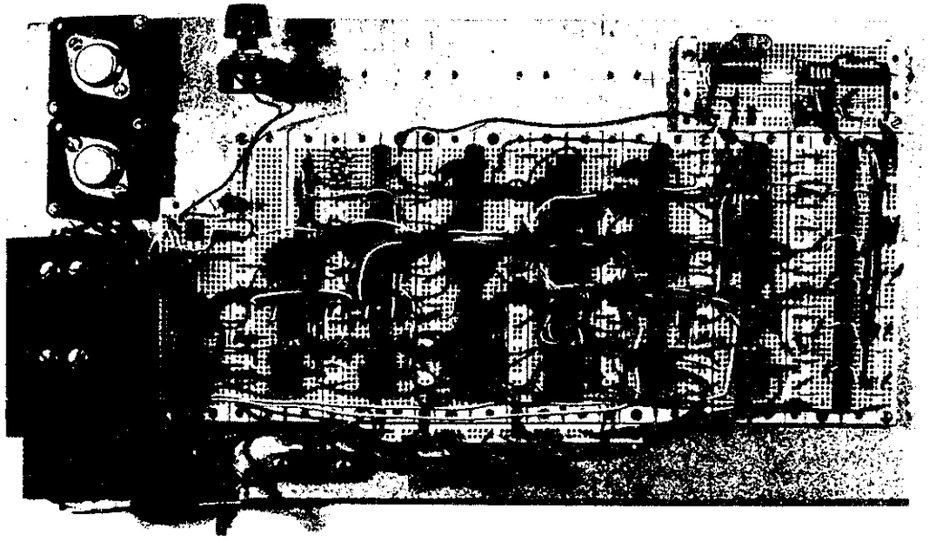
Şekil 1.2 Bir üniversal baskılı devre kartı (Vector Electronics Company, Inc. in müsadeleriyle)



Şekil 1.3 *(a)* Atlamalı bağlantıları yapmak için kullanılan bir tel sargı tabancası. *(b)* Tel sargı bağlantılarının yapılışı. (OK Machine and Tool Corporation.)

Bir defa üzerine devre kurulacak malzemeyi seçtikten sonra, devre şemasını inceleyerek entegre devre ve diğer parçaların sayısını belirleyin. Buradan gerek duyulacak kart alanını belirleyebilirsiniz. Entegre devreler arasında tel bağlantıları için 1/8 ile 1/2 inç (3,2 ile 12,7 mm) açıklık bırakın, böylece gerektiğinde entegre devreler kolayca çıkarılabilir. Bir standart 14 veya 16 bacaklı entegre devre yaklaşık 1 inç^2 (6,5 cm^2) kart alanı gerektirir. Ayrıca, devrede sonradan bir değişiklik yapmak gerekirse, bunu bütün kartı yeniden kurmadan yapabilmek için bir iki entegre devre ekleyecek kadar ilave bir alan bırakmayı unutmayın. Şimdi uygun bir üniversal baskılı devre kartı temin edebilir veya gerekli temel örnek kartı montajla oluşturabilirsiniz. Lehimsiz

temel örnek kartlarını monte ederken kart parçalarını bir sunta veya sert bir plastik parçası üzerine vidalamak iyi bir fikirdir. Bu, karta yapısal sağlamlık verir ve parçaların kilitleme tırnaklarının kırılmasını önler. Güç ve toprak bağlantıları için karta uç bağlantı direkleri eklenebilir (Şekil 1.4 e bakınız).



Şekil 1.4 Bir MSI frekans sayıcının lehimsiz temel örnek kartı üzerindeki yapısı.

DEVRE ELEMANLARININ YERLEŞİMİ

Devre elemanlarının yerleşiminin dikkatli planlanması bağlantıları basitleştirir, hataları azaltır ve arıza bulmayı kolaylaştırır. Devrenin şemasının düzenli olduğu kabul ediliyorsa, genelde devre elemanlarının yerleşimi mümkün olduğunca şemadaki yerleşime paralel olmalıdır. Eğer bu olmuyorsa, devreyi mantıksal sinyal akışına göre düzenlemeye çalışın. Bu, karta bakan herkesin devrenin kısımlarını kolayca bulmasına ve üzerinde sinyalleri kolayca izlemesine yardımcı olacaktır. Ayrıca, daha sonra devrenin bir baskılı devre kartı hazırlanacaksa, ön çalışmaların çoğu yapılmış olur.

Bütün entegre devreler, ters konma ihtimalini azaltmak için mümkün olduğunca aynı yönde konmalıdır. Bu aynı zamanda bağlantıları yaparken ve arıza bulunurken bacak numaralarını izlemeyi kolaylaştıracaktır.

DEVRENİN BAĞLANTILARINI YAPMA VE ETKİLİ YANGEÇİT

RENK-KODLU TELLERLE BAĞLANTI

Tel bağlantı hatalarını azaltmaya ve bir devrede arıza bulmaya yardımcı olması için bütün bağlantıları renk kodlu yapın. Telefon şirketlerinin çok çiftli iletişim kablolarının dış kılıfları sıyrılarak çok çeşitli renk birleşimleri olan teller elde edilebilir. Ancak, lehimsiz temel örnek kartı için çok telli olmamasına ve 22 veya 24 numara olmasına dikkat edin. Daha geniş tel veya uçlar iç kontakların açılmasına neden olur, daha ince teller ise her zaman güvenilir temas yapmayabilir. Lehimsiz temel örnek kart bağlantıları için izolasyonun 1/4 inç (6,4 mm) kadarını soyun. Tel sargı bağlantıları için özel bir 30 numara tel kullanılmaktadır.

Mümkünse arıza bulmayı kolaylaştırmak için devreyi parça parça kurup test edin. Direnç ve diğer elemanların uçlarının boylarını, kaza ile olabilecek kısa devreleri önlemek için mümkün olduğunca kısa tutun. Bağlantı tellerinin boylarını kısa tutun fakat arızalı bir entegre devreyi değiştirmek gerektiğinde telleri çıkarıp, çoğu zaman da hatalı olarak, yeniden takmaya gerek kalmayacak şekilde telleri entegre devreler ve soketler etrafından dolaştırın. Kapasitor ve diğer elemanlar değerleri kolayca okunabilecek şekilde monte edilmelidir. Dirençler, hataları kolayca görebilmek için (örneğin 15 Ohmluk direnç yerine 1 MOhmluk direnç takmak gibi) renk kodları soldan sağa veya yukarıdan aşağıda doğru okunacak şekilde monte edilmelidir.

Her bağlantı teli veya eleman devreye eklendikçe, devre şeması üzerinde karşılık gelen kısmı saydam bir sarı veya başka bir renkte keçeli kalem ile çizin, böylece hangi kısımlarının bitirilmiş olduğunu kolayca izleyebilirsiniz. Şema hâlâ görülebilir ve bir hafta sonu tatilinden geri döndüğünüzde size nereden başlayacağınızı gösterecektir. Devrenin bir kısmının bağlantıları tamamen bitirildiğinde, bütün entegre devreleri, birinci bacağından son bacağına ve son bacağından birinci bacağına kadar, kontrol edip her bacağın doğru nokta veya noktalara bağlanmış olduğundan emin olun. Bu her ne kadar fazla bir iş gibi görünüyorsa da, dikkatli inşa ve kontrol etmek için harcanan zamanın ödülü daha fazla ilk defasında çalışan devreler ve daha az yanan entegre devreler olacaktır.

AĞAÇ ŞEKLİNDE BAĞLANTI

Lehimsiz temel örnek kartı ile çalışırken, gücün karta girdiği uç bağlantı direğinden herbir güç yolu şeridine ayrı bir bağlantı yapın. Ayrıca, her toprak yol şeridini toprak bağlantı direğine ayrı ayrı bağlayın. (Şekil 1-4 e bakınız). Uygun yangeçit kapasitörlerinin de eklenmesiyle, bu düzen güç kaynağı yollarını birbirinden ayırır. Böylece devrenin bir tarafında üretilen geçici gerilim değişiklikleri veya gürültülerin diğerlerini etkilemesi engellenmiş olur. Ayrıca bu yapı bütün kısımların aynı kaynak gerilimini almasını sağlar. Bir yol şeridinden diğer yol şeridine atlayarak giden bir tek tel (papatya zinciri denir), hattın sonundaki kısımlara daha düşük gerilim sağlayacaktır. Küçük devreler için bu önemsiz olabilir, fakat büyük devreler için önemlidir.

DEKUPLAJ VE YANGEÇİT

Dekuplaj bağlantısı kullanmak geliştirmeniz gereken iyi bir alışkanlıktır. 25 ile 100 µF arasında bir elektrolitik kapasitörle 0,1 µF lik bir kapasitör gerilimdeki gürültüleri filitrelemek için güç bağlantı direği ile toprak bağlantı direği arasına, veya baskılı devre kartının güç ve toprak girişleri arasına bağlanmalıdır. Burada niçin 0,1 µF lık bir kapasitörün kendisinden 250 veya 1000 defa büyük bir kapasitörle paralel bağlandığı sorusu akla gelebilecektir. Her kapasitör uçları ve iç yapısı dolayısıyla küçük bir seri endüktans gösterir. Bu seri endüktans kapasitörle birlikte bir frekansta seri rezonans devresi oluşturur. Temel alternatif akım teorisi bu rezonans frekansının altında net reaktansın kapasitif olduğunu ve rezonansda minimuma ulaştığını söyler. Rezonans frekansının üstünde net reaktans endüktiftir ve frekans arttıkça yükselir. Bu artan endüktif reaktanstan dolayı, kapasitör, mantık kapılarının durum değiştirirken oluşturduğu yüksek frekanslı gerilim gürültülerini filitrelemekte veya kısa-devre etmekte etkisiz kalmaya başlar. Her iki ucu 1 inç (25 mm) olan 100 µF lik bir kapasitörün rezonans frekansı, telin inç başına olan endüktansı $20x10^{-9}$H ve $f_r = 1/(2\pi\sqrt{LC})$ alınırsa, yaklaşık 80 KHz dir. Bu frekansların çok yukarısında, 100µF lik bir kapasitör etkili bir yangeçit filitresi olmamakla beraber, uçları 1 inç (25mm) uzunluğunda olan paralel bir 0,1µF lik kapasitör yangeçiti 2,5 MHz ye uzatır. Böylece iki kapasitörün birleşimi geniş bir frekans aralığında filitreleme sağlar.

Yangeçit kapasitörlerinin endüktanslarını azaltmak ve böylece performanslarını geliştirmek için, uçları mümkün olduğunca kısa tutulmalıdır. Örneğin, 0,1µF lik bir kapasitörün her iki ucunu 1/4 ince (6,4mm) azaltmak etkili yan geçit aralığını 2,5 tan 5 MHz ye uzatır. TTL ve CMOS devrelerinde, her entegre devrenin (en azından bir atlayarak olabilir) yanına güç ve toprak uçları arasına ucuz bir 0,1 veya 0,01 µF lik yangeçit kapasitörü konmalıdır. Daha yüksek frekanslarda yangeçişi sağlamak için, ECL (Emitör kuplajlı mantık) ve

Schottky TTL devrelerinde 0,1 ve 0,01 µF lik kapasitörlere ek olarak 100 ile 1000 pF arasında kapasitörlere gerek duyulur.

İLK TEST

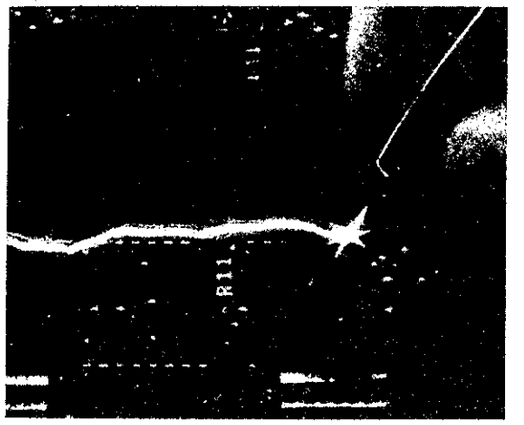
Devrenin bağlantılarını dikkatle yaptıktan ve elemanların yönlerini kontrol ettikten sonra güç kaynağı ile toprak arasında kısa devre olmadığını kontrol etmek için bir ohmmetre kullanın. Sonra güç uygulayın ve devreden gelecek ses, duman, patlama, sızıldama gibi anormal işaretleri gözleyip dinleyin. Eğer bunlardan herhangi birisini görür veya duyarsanız derhal gücü kesin ve parmağınızla hafifçe dokunarak çabucak aşırı ısınmış elemanları arayın. Burada dikkatli olmak gerekir, çünkü birçok entegre devre ve elektrolitik kapasitör ters bağlandığında kötü bir yanık oluşturacak kadar çabucak ısınır. Ayrıca genellikle gürültü ve eleman ısınmasına sebep olan hatalı tel bağlantıları, yanlış direnç değerleri ve lehim köprüleri olup olmadığını araştırın.

Sıcak bir elektrolitik kapasitör ya içinden kısa devre olmuştur, ya da daha büyük bir olasılıkla devreye ters yönde takılmıştır. Bazı "epoksi yuvarlağı" tantalum elektrolitik kapasitörlerin yön işaretlerini görebilmek için çok yakın muayene gerekmektedir. Ayrıca, nadiren işaretler hatalı olabilmektedir. Bir elektrolitik kapasitör bir defa ters yönde yüklenmişse, genellikle bozulmuştur ve değiştirilmelidir. Büyük elektrolitikler yanlış yönde yüklendiklerinde havai fişek gibi patlayabilir. Büyük elektrolitikler içeren bir devreye ilk defa güç uyguluyorsanız 120 V luk bir ototransformatör veya varyak kullanarak devreye uygulanan gerilimi yavaş yavaş olması gereken değere ulaştırmalısınız. Bu, kapasitörlerde dielektiriğin artmasına müsade eder ve patlatacak yeterli iç basınç oluşturmadan sıcak kapasitörleri bulmanız için size süre tanır.

Entegre devrelerin ısınmasının en önemli sebepleri, soketlere ters yerleştirilmeleri veya bağlantı hatası yüzünden kaynak yönünün ters olmasıdır. Diğer bir sebep de çıkışın V_{cc} ye veya toprağa kısa devre olmuş olmasıdır. Bazı mantık ailelerinde, iki entegre devrenin çıkışlarını birbirine bağlamak, çıkışlardan biri yükseğe gitmek isterken diğeri alçağa gitmek isterse, kapıların kızmasına sebep olur. Ters yerleştirilmiş entegre devreler yenileriyle değiştirilmelidir.

Eğer gözle muayenede belirgin bir problem gözlenmezse, bir ohmmetre mümkün olan kısa devreleri veya elemanları bulmak için kullanılabilir. Bu tür bariz problemler giderildikten sonra devreye yeniden güç verin ve sayısal bir voltmetre ile bütün güç kaynağı gerilimlerinin varlığını doğrulayın. Güç kaynağını yükleyerek aşağı çekilmesine neden olan kısa devreler Hewlett Packard 547A (Şekil 1.5 e bakınız) veya General Radio Bughound gibi akım izleme problarıyla çabucak bulunabilir. Akım probları sadece değişen akıma tepki

gösterdiklerinden, güç kaynağı bağlantısı ayrılır ve devrenin güç kaynağı girişine Hewlett Packard 546 A mantık darbe üreteci gibi bir düşük empedanslı darbe kaynağı bağlanır. 547 A akım izleme probunun ucundaki ışığın parlaklığı akan akımın miktarını gösterir. Probun ucunu güç kaynağı teli boyunca veya en fazla akımı gösteren iz boyunca hareket ettirin. Kısa devre bulunan nokta geçildiğinde ışık sönecektir.



Şekil 1.5 HP 547A Akım izleme probu (Hewlett-Packard Company.)

Herhangi bir bağlantı hatası, kısa devre veya hatalı kaynak gerilimi olmadığı halde hâlâ kızgın bir entegre devre varsa, gücü kesin ve entegre devreyi değiştirin. Entegre devreleri soketlerinden veya temel örnek kartlarından çıkarırken entegre devre yerinden fırlayıp çıkıncaya kadar önce bir ucunu sonra diğer ucunu küçük bir tornavida (veya entegre devre çıkarıcı) kullanarak kaldırın. Entegre devreleri parmaklarınızla çekip çıkarmaya çalışırsanız sonuçta parmaklarınızı delip entegre devrenin bacaklarını yamultabilirsiniz. Güç uygulandığında sürekli aşırı ısınma, farkedilmeyen bir bağlantı hatası veya devreyi kurmada kullanılan şemada bulunan bir hata anlamına gelebilir. İlgili entegre devrenin veri sayfasına bakıp şemadaki bacak numaralarının doğruluğunu kontrol edin. TTL ile çalışırken 7 veya 8 inci bacağı toprağa ve 14 veya 16 ıncı bacağı V_{cc} ye bağlamaya kolayca alışabilirsiniz. Ancak, çok kullanılan bir TTL olan 7475 in 5 inci bacağı V_{cc} ve 12 inci bacağı topraktır. Genelde toprak ve kaynak bacakları olan 8 ve 16 ıncı bacakların her ikisi de çıkıştır. Bu

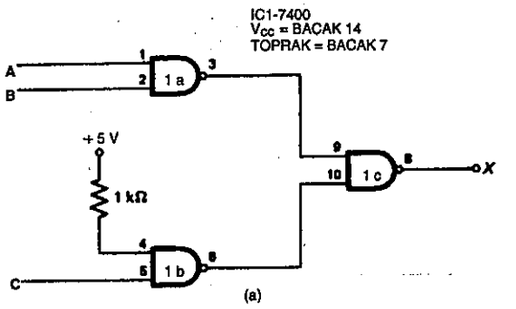
tür hatalara dikkat edin ve bazen tasarımcıların bir tasarımda bir entegre devreyi değiştirdiklerini fakat şemadaki bacak numaralarını değiştirmeyi unutmuş olabileceklerini aklınızdan çıkarmayın.

FONKSİYON TESTİ VE ARIZA BULMA

Bir devre duman ve ısı testlerini geçtikten sonra, devrenin doğru çalışıp çalışmadığını anlamak için hangi testleri uygulamanız gerektiğini düşünün. Hangi sinyaller uygulanmalıdır? Hangi çıkışlar beklenmektedir? Bu girişler nasıl üretilebilir? Bu giriş ve çıkışları ölçmek için hangi cihazlar gereklidir? Dikkatli hazırlanmış testler olmadan ve cihazların kullanımını iyi bilmeden çok büyük bir zamanınızı gelişigüzel denemelerle boşa harcayabilirsiniz. Problemlere sistemli olarak yaklaşmaya çalışın. Bu bölümün gerisi bilinen test cihazlarının çalışmasını tanıtacaktır. Daha sonraki bölümlerde daha karmaşık test cihazları, bunlara gerek duyan devreler tanıtıldıkça, anlatılacaktır.

Başlangıç olarak, Şekil 1.6 a daki örnek devreyi test ettiğinizi kabul edelim. Şekilde görülen kapıların VE DEĞİL kapıları olduklarını ve pozitif mantıkta girişlerden herhangi birisi alçak olduğunda, bir VE DEĞİL kapısının çıkışının yüksek olduğunu hatırlarsanız, bu devre için Şekil 1.6 *b* de görüldüğü gibi bir doğruluk tablosu hazırlayabilirsiniz. (Bu devre için çıkış ifadesinin $X = A.B + C$ olduğunu farkedebilirsiniz). Bu ayrık bir devre ise, giriş koşullarını oluşturmak için girişten toprağa veya V_{cc} ye atlama telleri koyabilirsiniz. Devre daha büyük bir devrenin bir parçası ise giriş koşullarını bir osiloskop veya mantık probu kullanarak belirlemek zorundasınız. Bundan sonra gerçek çıkış, doğruluk tablosunda öngörülen çıkışla karşılaştırılabilir.

Mantık gerilim seviyeleri doğrudan entegre devre bacakları üzerinde kontrol edilmelidir. Doğruluk tablosundan birden fazla giriş birleşimini oluşturmaya çalışın ve sonra da hem sinyal çıkış bacağını hem de sinyal giriş bacağını uygun mantık seviyeleri için kontrol edin. Çıkış öngürüldüğü gibi değilse otomatik olarak entegre devrenin bozuk olduğunu kabul etmeyin. Örneğin, Şekil 1.6 *a* da gösterilen devrede, entegre devrenin 3 ve 6 numaralı bacaklarında aynı sinyal seviyesinin olup olmadığını kontrol edin. Benzer şekilde, 6 ve 10. bacakları kontrol edin. Soğuk lehim eklemlerini, lehim köprülerini ve hatalı bağlantıları yok ettikten sonra devre hâlâ öngörüldüğü gibi çalışmıyorsa entegre devre yongasını değiştirmeyi deneyin ve seviyeleri yeniden kontrol edin.



(a)

A	B	C	X
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$X = A \cdot B + C$

(b)

Şekil 1.6 *(a)* Sayısal devre. *(b)* Doğruluk Tablosu

SAYISAL VOLTMETRELER VE MANTIK PROBLARI

SAYISAL VOLTMETRELER (DIGITAL VOLTMETERS, DVM)

Şekil 1.6 *a* daki örnek devrenin giriş ve çıkış sinyalleri sadece doğru akım (d.a.) seviyeleri olduğundan dolayı, sayısal voltmetreler, mantık seviye probları ve seviye test cihazları gibi bunları kontrol etmekte kullanabileceğiniz birçok cihaz vardır.

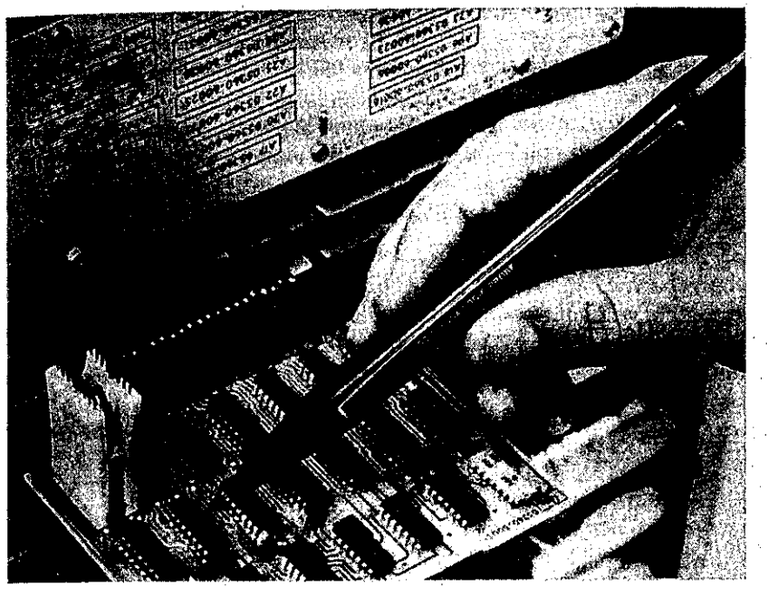
Bir sayısal voltmetre bir çıkışın yüksek mi yoksa alçak mı olduğunu çok hassas bir şekilde söyleyecektir. (Bölüm 2 de gösterildiği üzere gerçek sayısal değerler mantık ailesine göre değişmektedir.) Ancak birçok sayısal voltmetre probu entegre devreler üzerinde gezdirmek için fiziksel olarak uygun değildir. Siz bir değer okumaya çalışırken, prob kayabilir ve yandaki entegre devre bacaklarını kısa devre yapabilir. Ayrıca devre ile gösterge arasında sürekli ileri geri bakmak yorucu olmaktadır. Bazı sayısal voltmetre üreticileri, göstergeleri prob üzerinde bulunan, küçük, elde tutulabilen cihazlar yaparak bu problemi çözmüşlerdir.

MANTIK PROBLARI

Sayısal voltmetrenin hassasiyetinin gerekmediği mantık seviyesi belirlemelerinde HP545A veya Continental Specialties Corporation LP-1 (Şekil 1.7 ye bakın) gibi bir mantık probu kullanabilirsiniz. Bu problarin, giriş yüksekse yanan bir LED i (ışık-yayan-diyot) ve giriş alçaksa yanan diğer bir LED'i vardır. Probdaki gerilim karşılaştırma seviyeleri bilinen mantık ailelerinkilere karşılık gelecek şekilde ayarlanabilir ve çoğu mantık probları tek bir darbe veya bir darbe treninin varlığını gösterebilecek şekilde yapılmışlardır. Mantık probu kavramı çok bacaklı, her bacak için bir LED i olan kıskaçlı seviye test cihazlarına uzanmıştır. Bunlar, dışarıdan güç sağlanırsa iyi çalışır. Test edilen entegre devreden güç alırlarsa, 15 civarında LED i yakmak için gerekli 200 mA lık akım çekilmesi devrenin çalışmasını bozabilir. Ayrıca 74121 (tek kararlı multivibratör) gibi yongalardaki bazı bacaklar bu test cihazlarıyla aşırı yüklenmektedirler. Bu bacaklar için bir osiloskop kullanılmalıdır.

OSİLOSKOPLAR

Bir osiloskop yukarıda gördüğümüze benzer devrelerde mantık seviyelerini kontrol etmek için kullanılabilir. Ancak, osiloskopun gerçek gücü darbe genişliğini ölçmeniz, frekansları hesaplamanız, kısa süreli istenmiyen durumları bulmanız veya iki veya daha fazla sinyalin zamanlamasını karşılaştırmanız gibi dinamik mantık devrelerinin derinlemesine analizini yapabilmenizi sağlamasında yatar. Osiloskop çoğunlukla sizin ana test cihazınız olduğundan, kullanacağınız osiloskopun üzerindeki bütün kontrolların fonksiyonlarını öğrenmek için zaman ayırmanızda yarar vardır. Bu size cihaza sağlanmış bütün güçten ve özelliklerden yararlanma imkânı verecektir. Örneğin bazı zamanlama problemleri sadece gecikmeli taramayı kullanarak görülebilir hale getirilebilir. Bu bölümde ve bütün kitap boyunca osiloskopun bu ve birçok diğer fonksiyonlarının genel kullanışı üzerinde durulmaktadır. Kendi özel osiloskopunuzun çalışma ayrıntıları için üreticinin verdiği kılavuza başvurunuz.



Şekil 1.7 HP545 A Mantık Probu. IC lerden biri üzerindeki glomper kıskacına dikkat edin. (Hawlett-Packard Company.)

TETİKLEMELİ TARAMA OSİLOSKOPUNDA BİR İZ ELDE EDİLİŞİ

Tetiklemeli tarama osiloskopunu ilk açtığınızda, ekranda hiç bir iz belirmemesi mümkündür. Görünen bir iz elde etmek için, ışın şiddetini artırın, tetiklemeyi otomatiğe getirin ve yatay ve düşey konum kontrollarını sahalarının ortasına ayarlayın. Eğer osiloskopun çeşitli tarama modları (gecikmeli, gecikmeli tetiklemeli, v.b gibi) varsa, ana taramayı seçin ve sonra tetikleme seviyesi veya tetikleme kararlılık kontrolunu bir görüntü elde edilinceye kadar ayarlayın. Eğer hâlâ bir iz görülmüyorsa, ışın bulma düğmesine basın. Bu kontrol yatay ve düşey konum kontrollarının sahasını sıkıştırarak sahaları içinde herhangi bir yerdeki izi görünür yapar. İzi bir defa bulduktan sonra, yatay ve düşey konum kontrolları onu merkezlemek için kullanılabilir. Eğer osiloskopunuzun bir ışın bulma düğmesi yoksa, yatay ve düşey konum kontrollarını yavaş yavaş çevirerek ışını bulmak zorundasınız.

Ekranda en iyi görüntü için, iz açıkça görülecek fakat ışın ekranın fosfor kaplamasında bir delik açmayacak şekilde ışık şiddetini azaltın. Alçak şiddetli ışınlar, görüntüyü daha ince ve hassas olarak netleştirebilmenizi sağlar ve katot ışınlı tübün ömrünü uzatır. Eğer varsa astigmatizm kontrolu, bir darbe görüntüsünün yatay ve düşey kısımları netlik kontrolu ile, aynı netliğe getirilinceye kadar ayarlanır. Eğer yatay iz ekranın yatay eksenine paralel değil ise, iz döndürme kontrolunu bulup problemi düzeltin. İz döndürme kontrolu osiloskopun arkasında olabilir.

DEĞİŞEN TARAMA VEYA BÖLÜMLENMİŞ TARAMA

Sabit bir iz elde edildikten sonra, gösterilecek sinyale uygun yatay zaman/bölme ve düşey volt/bölme değerleri seçilebilir. Değişken volt/bölme ve zaman/bölme kontrollarının kalibrasyonlu konumda olmasına mutlaka dikkat edin. Eğer ekranda tek ışınla iki sinyali göstermek istiyorsanız, ya bölümlenmiş veya değişen görüntü modunu seçmeniz gerekir.

Değişen modunda, ışının ekranı birinci taramasında kanal 1 deki sinyal görünür, ikinci taramasında kanal 2 deki sinyal görünür. Düşey konum kontrollarını ayarlayarak her görüntüye eklenen bir d.a. gerilim ekranda dalga şekillerini birbirinden ayırır. Tarama hızının 1 ms/bölme den yukarıda olduğu yüksek frekans sinyallerine bakarken gözünüz izlerdeki sıra değişmesini görmeyecek, her iki görüntü de aynı anda varmış gibi görülecektir. Düşük tarama hızları gerektiren alçak frekanslı sinyalleri gözlerken, değişen mod analizi çok zor olan bir yanıp sönen görüntü oluşturacaktır.

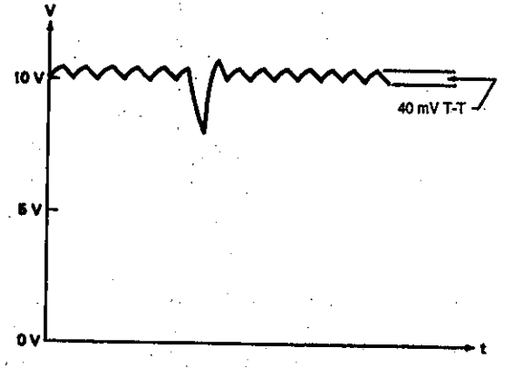
İki alçak frekanslı sinyalin kararlı görüntüsü için, ışının ekranı her tarayışı sırasında iki kanal arasında hızlı bir şekilde değişmesi, yani sinyalleri bölümlemesi durumu olan bölümlenmiş tarama modunun kullanılması gerekir. Işın önce kanal 1 deki dalga şeklinin bir parçasını, sonra kanal 2 deki dalga şeklinin bir parçasını, sonra kanal 1 den bir başka parçayı ve yine kanal 2 den bir başka parçayı çizerek ekran boyunca hareket eder. Bölümleme gözünüzün fark edebileceğinden çok daha hızlı (50 ile 100 KHz arası) olduğundan görüntüler sürekliymiş ve aynı anda varmış gibi görülür. Bölümlenmiş mod yüksek frekanslı sinyallerde kullanılamaz, çünkü dalga şeklinin bölümlenmesi gözlenen dalga şekillerinin görünen anahtarlama zamanını hatalı olacak şekilde etkileyebilir. Örneğin, ışın kanal 2 yi gösterirken, kanal 1 deki sinyal seviye değiştirebilir. Ancak görüntü, ışın kanal 1 i göstermeye başlayıncaya kadar bu değişiklik görüntülenmeyecektir. Böylece tam olarak seviye değişikliğinin ne zaman olduğunu bilmenize imkan olmayacaktır.

Eğer osiloskopunuz çok bölmeli tiplerden ise, sağ ve sol yükselteç bölmeleri arasında bölümlenmiş ve değişen görüntü seçimi için zaman/bölme anahtarı yanında bir kontrol bulanacaktır. Bölümlenmiş ve değişen görüntü seçimi ile ilgili yukarıdaki açıklamalar aynı zamanda bu seçim için de geçerlidir.

D. A. VEYA A. A. KUPLAJ

Doğru akım kuplaj, girişi doğrudan düşey yükseltece bağlar ve böylece giriş gerilimi yükseltilip görüntü sapmaları üretilir. Eğer osiloskopun düşey ölçeği 2V/bölme ye ayarlanmışsa ve girişe 5 V d.a. uygulanmışsa, iz sinyal olmadığı durumdaki konumundan 2,5 bölme yukarı kayacaktır. (5V x 1 Bölme / (2V) = 2,5 Bölme). Girişin yönü ters çevrilirse, iz, sinyal olmadığı durumdaki konumundan 2,5 bölme aşağıya kayacaktır. Böylece bu kuplaj modunda osiloskop kaba bir d.a. voltmetre (% 3-5 hassas) olarak kullanılabilir. Gerilim kaynaklarının varlığını ve bir sinyal hattının yüksek veya alçak seviyede olduğunu belirlemede bu yeterli olacaktır. Örneğin, çok fazla yüklenmiş bir kapı, çıkışında hala bir darbe gösteriyor olabilir, halbuki darbe nizami alçak veya yüksek seviyeye ulaşmamış olabilir. d.a. kuplaj sayesinde, osiloskop bu problemi kolayca gösterecektir. Kuplaj seçim anahtarında kalibrasyonda yardımcı olması için genellikle bir de toprak konumu bulunur.

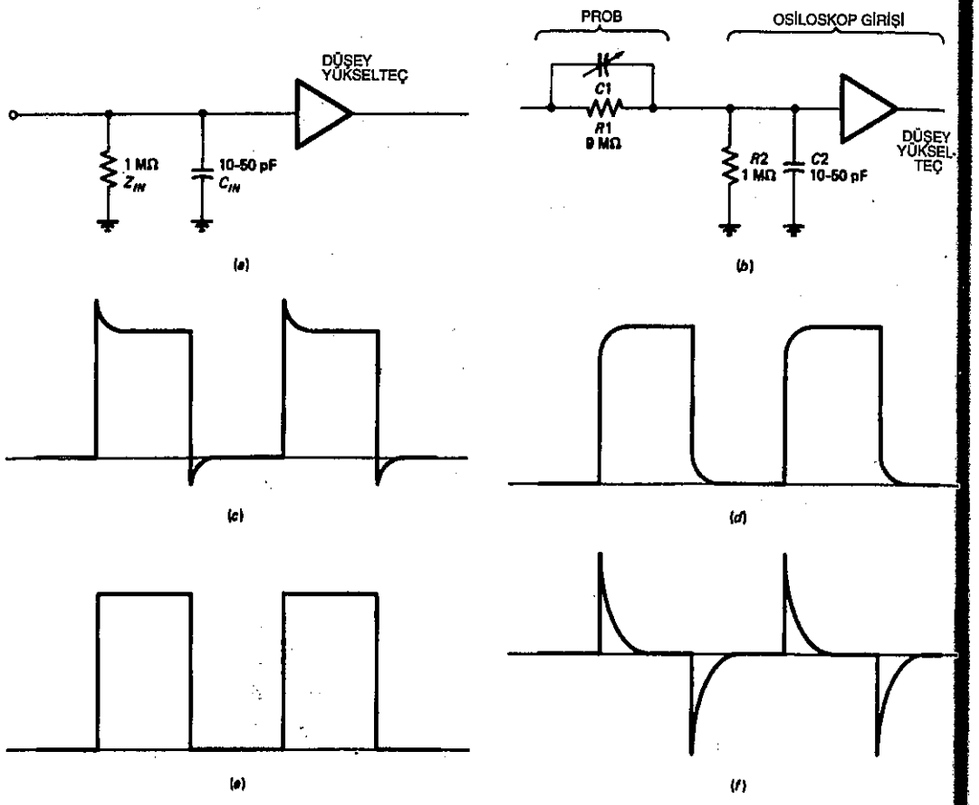
Bir osiloskopun girişi a.a. kuplaja çevrildiğinde, sinyal girişi ucu ile seri bir kapasitör bağlanmış olur. Bu kapasitor gelen sinyalin d.a. bileşenini bloke eder ve sadece sinyalin a.a. yani değişen kısmının görülmesine müsade eder. a.a. kuplajı kullanmanın, Şekil. 1.8 de görüldüğü gibi, bir güç kaynağındaki dalgacıkları veya gürültüleri ararken faydalı olacağı açıktır. Güç kaynağı, 10 V d.a. üzerine eklenmiş 40 mV a. a. dalgacık sinyalleri olan kaynaktır. D.a. kuplaj kullanılırsa, 40 mV luk a.a. sinyali inceleyebilmek için görüntüyü yeterince genişletemezsiniz, çünkü 10 V d. a. görüntünün ekran dışına çıkmasına sebep olacaktır. a. a. kuplaj d. a. bileşeni bloke eder ve böylece 40 mV lik sinyal ekranda sıfır ekseninde merkezlenmiş olarak görülür. Bundan sonra da 40 mV luk sinyalin ayrıntılı analizini yapabilmek için görüntü ekranı dolduracak şekilde genişletilebilir.



Şekil 1.8 Güç kaynağındaki dalgacıklar ve gürültü

PROBLAR VE PROB DENGELENMESİ

Osiloskopa en iyi sinyali ulaştırmak ve test edilen devreyi en az yüklemek için genellikle problar kullanılır. En çok kullanılanları X1, X10 ve X100 pasif gerilim problarıdır. Örneğin X10, osiloskopun girişinde görülen sinyalin, prob ucunda görülen sinyalin onda biri büyüklüğünde olduğunu gösterir. Osiloskop probları güzel sapı olan tel parçaları değildir ve en iyi sonuçları almak için kullanıldıkları özel girişe uymak üzere ayarlanmaları gerekir.



Şekil 1.9 *(a)* Orta frekans osiloskopunun denk giriş empedansı. *(b)* X10 osiloskop probunun ve osiloskop girişinin denk devresi. *(c)* Prob ayarında C1 çok yüksekse görülecek osiloskop dalgaşekli. *(d)* Prob ayarında C1 çok düşükse görülecek osiloskop dalgaşekli. *(e)* Doğru prob ayarında görülecek dalgaşekli. *(f)* Osiloskop probunun orta iletkeni kopmuş veya iç direnci açık devre iken görülebilecek osiloskop dalgaşekli.

Tipik bir orta-frekans osiloskopunun giriş empedansı 1 MΩ luk bir dirençle 10-50 pF lik bir kapasitörün paralel birleşiminin toprağa bağlanışı şeklinde gözükür (Şekil 1.9 *a* ya bakın). Osiloskop girişini doğrudan bir mantık devresine bağlarsanız 1 MΩ luk direnç büyük bir ihtimalle bir problem yaratmayacaktır, fakat 10-50 pF lik ilave kapasitans devrenin zamanlamasını değiştirecek ve hatalı bir çalışmaya sebep olacak kadar test altındaki devreyi yükleyebilir. Bu problemin bir çözümü *X10* veya *X100* lük bir prob kullanmaktır. (Şekil 1. 9 *b* ye bakın). 9 MΩ luk direnç ile 1 MΩ luk osiloskopun giriş empedansı, d.a. da 9:1 lik veya *X10* luk bir gerilim bölücü oluşturmaktadır. Ancak frekans arttıkça, osiloskopun giriş kapasitansı 1 MΩ luk direnci şöntler ve girişteki sinyalin genliğini azaltır. *C1* olmadan probun bant genişliği sadece birkaç kilohertzdir. Eğer iki devrenin zaman sabitleri eşit olacak şekilde *(R1C1 = R2C2) C1* ayarlanırsa, bütün frekanslar için osiloskopun girişi, prob ucunun girişinin onda biri olacaktır. Frekans arttıkça, probun toplam giriş empedansı doğru akım değeri olan 10 MΩ dan aşağı doğru azalacaktır, fakat prob tarafından devreye eklenen maksimum kapasitif yük her zaman için 10 pF den, ve hatta çoğunlukla 5 pF den, az olacaktır.

Çoğu osiloskop üreticileri en iyi frekans tepkisini elde etmeniz için probu dengelemenizi, diğer bir deyişle *R1C1 = R2C2* olacak şekilde ayar yapmanızı kolaylaştırmıştır. Prob çıkışını osiloskobun kullanılacak kanalına, prob ucununu da osiloskobun kalibratör çıkışına bağlayın. Düşey volt/bölme ve yatay zaman/bölme kontrollarını kalibratör dalga şeklinin bir periyodunun ekranı doldurmasını sağlayacak şekilde ayarlayın. Sonra ya düzeltici kapasitörü çevirerek veya bazı Tekronix problarında prob üzerindeki kilitleyici somunu gevşetip prob namlusunu döndürerek, kalibratör dalga şeklinde en iyi kare köşelerini görünceye kadar ayar yapın. Çok küçük *C1*, dalga şekli köşelerini yuvarlar; çok büyük *C1*, sivrilmiş veya az sönümlü köşeler verir (Şekil 1.9 *c* ve *d* ye bakın). Prob dengelenmesinin, probu kullanılacak olan osiloskop girişine bağlayarak yapılması gerektiğini unutmayın. Eğer Şekil 1.9 *f* deki gibi kalibratör dalgaşekli çok küçük bir genlik gösterirse, büyük bir olasılıkla R1 açıktır veya prob kablosunun orta iletkeni kopmuştur. Türev alınmış görüntü ya sinyalin *C1* üzerinden kapasitif kuplajından veya kırık orta iletkenin uçları arasında oluşan kapasitansdan dolayı oluşmaktadır.

Probun ve düşey yükseltecin genliğinin doğruluğu da kalibratör çıkışı kullanılarak kontrol edilmelidir. Bunu yaparken, değişken volt/bölme kontrolunun kalibreli konumunda olduğuna dikkat edin. Eğer görüntünün genliği ciddi olarak hatalı ise, uygun ayar işlemleri için osiloskopun kılavuzuna bakın.

Osiloskop problarını entegre devre yongalarına yandaki bacaklarla kısa devre yapmadan bağlamak çok zor olduğu için, bir entegre devre klipi veya "glomper" kilipi (Şekil 1.7 ye bakın) kullanılması çok yardımcı olabilir. Bunlar entegre devrenin üzerine bir çamaşır mandalı gibi takılır ve her entegre ba-

cağından uzanan, osiloskop probunun bağlanabileceği kısa uçları vardır. Test klipinin her ardışıl bacağını yaklaşık 45° aşağı doğru kıvırarak test klip bacaklarının kısa devre olma şansını daha da azaltabilirsiniz.

TETİKLEME KONTROLLARI

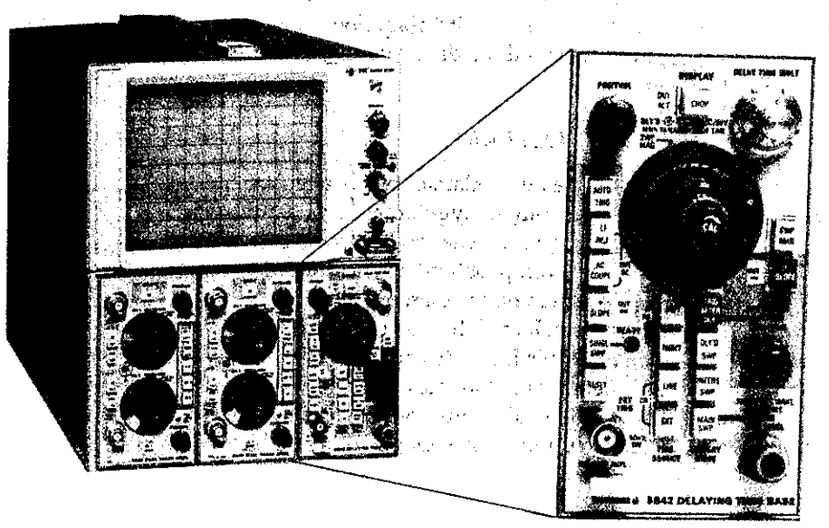
Osiloskop kullanımına ait açıklamalarımızda tetikleme ve tarama kontrollarına kısaca değinmiştik. Birçok uygulamalarda bu kontrolların sadece bir kaçının kullanılması gerekebilir. Ancak, bu kontrolların bilinçli ve doğru kullanılması birçok gözlenemeyen problemleri görülür hale getirebilir. Sistemli sinyal tarama genellikle kazara bozuk olan bir tanesini buluncaya kadar, rastgele entegre devre değiştirmekten daha etkilidir. Şimdi üzerinde duracağımız özel örnek, çok kullanılan bir laboratuvar takılabilir osiloskop birimi olan Tektronix 5B42 gecikmeli zaman tabanıdır (Şekil 1.10 a bakınız). Eğer bu birimlerden birine sahip değilseniz, sahip olduğunuz takılabilir zaman tabanı biriminde veya osiloskopunuzda karşılık gelen kontrolları bulmanız zor olmayacaktır.

TETİKLEME KAYNAĞI

Öncelikle tetikleme kaynağını ele alalım. Birçok uygulamada iç tetikleme kaynağını kullanacaksınız. Bunun için düşey yükselteçten sinyalin küçük bir kısmı alınır, yükseltilir ve katot ışınlı tüp boyunca olacak ışın taramasını başlatacak tetiklemeyi üretmek üzere şekillendirilir. 5B42 de bunu yapmak için tetikleme hangi düşey yükselteç bölmesinden sağlanacaksa, tetikleme kaynağı sütunundaki SOL veya SAĞ düğmeye basılır. Sayıcı çıkışları gibi farklı frekansları olan sinyalleri görüntülemeye çalışıyorsanız, çoğunlukla en düşük frekanslı sinyali tetikleme kaynağı olarak seçmek en iyisidir. Bu kararlı bir görüntü verir ve görüntülenen dalgaşeklinin faz ilişkisinin doğru olmasını garantiler.

A. a. güç hattı frekansı ile ilgili frekansı olan küçük bir sinyale (d. a. güç kaynağındaki dalgacıklar gibi) bakmak istediğinizde, hat kaynağı genellikle kararlı bir görüntü verecektir. Hat kaynağı, osiloskopun güç transformatöründe 60 Hz (veya 50 Hz) şebeke frekansını alır ve osiloskop tetiklemesini üretmek için bunu kullanır.

Dış tetikleme kaynağı, görüntülenmekte olan sinyallerin geçişleri yerine, taramanın başlatılmasının dış koşullar tarafından belirlenmesi istendiğinde kullanılır. Örneğin, üç çıkışlı bir devreniz varsa ve sadece *C* çıkışı alçaktan yükseğe geçiş yaptığında *A* ve *B* çıkışlarını karşılaştırmak istiyorsanız, *A* ve *B* sinyallerini düşey yükselteçlere, *C* sinyalini de dış tetikleme girişine bağlayın. Dış tetiklemeyi seçin ve tetikleme eğimi seçicisini + ya getirin. C



(a) (b)

Şekil 1.10 *(a)* Tekronix 5440 osiloskopu. *(b)* Tekronix 5B42 gecikmeli zaman tabanı (Tekronix, Inc. in müsadeleriyle).

çıkışı alçaktan yükseğe geçiş yaptığında görüntüyü başlatmak üzere ışın tetiklenecektir ve katot ışınlı tüp *C* deki geçişten sonraki *A* ve *B* nin durumlarını görüntüleyecektir. Genellikle, dış tetikleme girişi için en az 100 mV luk bir sinyal gerekir. Genliğin büyük olduğu mantık sinyallerinde, bir *X10* probu bu girişe bağlandığında kapı çıkışının kapasitif yüklenmesi en aza indirilecektir. Daha geniş devreler için dış tetiklemeyi kelime tanıyıcı denen cihazlardan elde edebilirsiniz. Anahtarla alçak veya yüksek durum seçimi yapılabilen 32 girişli bu cihaz, girişleri verilenle uyuştuğunda tetikleme sinyali üretir.

TETİKLEME GİRİŞ KUPLAJI

Tetikleme giriş kuplajının bilinen tipleri d.a., a.a., a.a. alçak frekans reddi ve a.a. yüksek frekans reddidir. İsimlerden anlamlarının ne olduğunu hemen hemen bellidir. d.a. kuplaj çok alçak frekanslarda veya rampa türü çok yavaş değişen sinyallerde tetikleme için kullanılır. İsmine rağmen d.a. kuplaj osilos-

kopun bütün band genişliği içinde kullanılabilir. a.a. kuplajı ise 50 MHz nin üstündeki sinyallerden tetiklemede veya tetiklemede d.a. bileşeninin etkisini yok etmek amacıyla (örneğin, güç kaynağındaki dalgacıklara bakıyorken) kullanılır. Diğer taraftan, eğer tetikleme sinyali üzerinde istenmeyen 60 Hz (veya 50 Hz) gürültüsü varsa, a.a alçak frekans reddi kuplajı kullanın. Bu kuplaj birkaç kilohertz altındaki sinyal bileşenlerini zayıflatır. Eğer yüksek frekans bileşenleri tetiklemede zorluk çıkarıyorsa, a.a. yüksek frekans reddi kuplajı kullanarak yaklaşık 50 kHz in üstündeki sinyal bileşenlerini zayıflatabilirsiniz. Bu filitreler sadece tetikleme kuplajını etkiler, katot ışınlı tüpteki görüntüyü etkilemez.

TETİKLEME EĞİM VE SEVİYE KONTROLLARI

Tetikleme eğim ve tetikleme seviye kontrolları çoğunlukla karıştırılır, fakat oldukça basittirler. Eğim kontrolü olarak + seçmek, alçaktan yükseğe geçişte ışının tetikleneceğini ve - seçmek taramanın yüksekten alçağa geçişte başlayacağını belirler. Seviye kontrolu ise, geçiş sırasında hangi gerilimde tetiklemenin olacağını belirler. İki kontrol birbirinden bağımsızdır. Pozitif eğimde ve negatif seviyede veya tersine tetiklemek mümkündür. Bunu göstermek için 1 KHz lik bir sinüs dalgasını düşey olarak ekranın dörtte üçünü dolduracak şekilde görüntüleyin. Seviye kontrolunu ortalayın, eğimi + seçin ve iç tetikleme kaynağını seçin. Yaklaşık olarak sıfır geçme noktasından başlamak üzere, ekranın sol kenarında sinus dalgasının potizife giden kısmını göreceksiniz. Seviye kontrolu döndürüldükçe, görüntünün başlama noktasının (veya tetikleme noktasının) sinüs dalgasının negatif tepesinden pozitif tepesine kadar kaydırabildiğini göreceksiniz. (-) eğim seçerseniz, görüntünün sinüs dalgasının negatife giden kısmıyla başladığını görürsünüz ve seviye kontrolunun etkisini yine gözleyebilirsiniz. Dış tetikleme kullanıyorken eğim anahtarı yine açıklandığı gibi çalışır ve seviye kontrolu tetikleme noktasını en az -1,5 dan +1,5 V a kadar olan bir aralıkta ayarlayabilecektir. Birçok iyi kalite osiloskopta her düşey giriş yükselteci ile seri bir iç geciktirme hattı vardır. Bu, sinyali yeteri kadar geciktirerek taramanın önce başlamasını ve tetiklemeyi yapan darbenin ön kenarını ekranda görebilmenizi sağlar.

TETİKLEME MODLARI

Otomatik tetikleme modunda, tarama üreteci serbest çalışır ve bundan dolayı da, osiloskopa bir giriş sinyali olmasa bile, ekranda her zaman bir iz görülür. Bir sinyal uygulandığında, bu sinyal seviye ve kararlılık kontrollarıyla birlikte tetiklemeyi oluşturur. Otomatik tetikleme d.a. seviyelerini ve basit sinyalleri kontrol etmeye çok uygundur, çünkü toprak referans izini gösterir ve tetikleme seviyesini ayarlamanızı gerektirmez. Ancak kompleks dal-

gaşekilleri için ve daha yüksek frekanslarda, normal yani otomatik olmayan tetikleme en iyi sonuçları verir. Eğer varsa, yüksek frekans senkronize modu, normal tetikleme devreleri için çok küçük veya çok yüksek frekanslı olan sinyallerle tetikleme için kullanılır. Bu modda, serbest çalışan tarama osilatörünün frekansını seviye kontrolu ile ayarlayarak görüntülenen frekansın tam bölenleri elde edilir ve kararlı bir görüntü sağlanır. Eski, serbest çalışan osilaskoplar da bu moda benzer şekilde senkronize edilirdi.

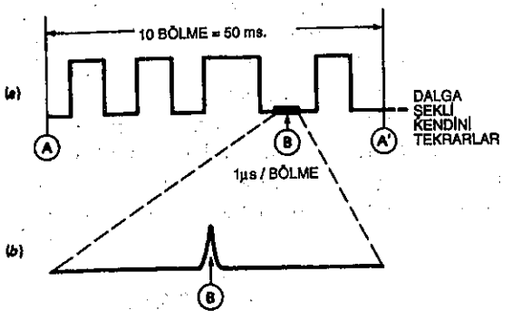
Tek-tarama modu, bir defa olan veya çok düşük oranlarda tekrarlayan olayların fotoğrafını çekmek için kullanılır. Tek-tarama seçildiğinde, bir 'hazır' ışığı yanar. Bir sonraki tetikleme darbesi ışının bir tek tarama yapmasını sağlar, hazır ışığı söner, ve tarama yeniden kuruluncaya kadar hiçbir darbe dikkate alınmaz.

Bazı osiloskopların *tetikleme bekletme* kontrolları vardır. Bu, iz ekranın sağ tarafına ulaştığında dalgaşekli simetrik olarak kendini tekrarlamıyorsa, bu dalgaşekillerinin kararlı bir görüntüsünü elde etmek için kullanılır. Normal olarak, osiloskop ekran boyunca olan taramasını tamamladıktan sonra, bir sonraki taramaya başlamak üzere giriş sinyalinin bir sonraki aktif kenarında tetikleme yapacaktır. Tetikleme bekletme kontrolu, tetikleme tanıma devresini sinyalin istenen aktif kenarı görülünceye kadar beklemek üzere ayarlanabilir. Böylece ışının her taraması simetrik olmayan giriş dalgaşeklinin hep aynı kısmını gösterecektir.

TARAMA MODLARI

Çoğu osiloskop uygulamalarında, ana yani normal tarama yeterlidir. Ancak bazı özel örnekler, büyütülmüş, geciktirilmiş ve tetiklemeli geciktirilmiş tarama modlarının önemini göstermektedir. Şekil 1-11 *a* bir sayısal cihazın basit bir kontrol saatinin dalgaşeklini göstermektedir. (Bunun gibi düzensiz bir dalgaşeklinin kararlı bir görüntüsünü elde etmek için bazan tetikleme bekletme kontrolunu veya değişken zaman/bölme ayarını ayarlamanız gerekir). Yatay ölçek 5 ms/bölme dir ve *A* ile *A'* arasındaki kısım bir kontrol periyodunu göstermektedir (desen *A'* de kendini tekrar etmeye başlar).

Yüzeysel olarak, görünen dalgaşekli cihazın bakım kılavuzunda gösterilen dalgaşekline tam olarak benzemektedir. Halbuki *B* noktasından sonraki alçaktan yükseğe geçişle başlaması gereken bir işlem gerçekte *B* anında başlamıştır. Bu *B* noktasında istenmiyen çok dar bir darbe veya gürültünün varlığından şüphelenmeye yol açmaktadır. Fakat eğer bu darbe veya gürültü sadece 10-15 ns genişliğinde ise, 5ms/bölme tarama konumunda görülmeyecektir. 2 ms/bölme gibi daha hızlı bir tarama kullanılırsa *B* noktası ekran dışına taşacaktır ve görülmesi mümkün olmayacaktır.



Şekil 1.11 *(a)* Kontrol saatinin osiloskoptaki dalgaşekli. *(b)* Gecikmeli tarama kullanılarak görülen *B* noktasındaki gürültü.

BÜYÜTÜLMÜŞ TARAMA

İlgilenilen bölgeyi genişletmenin diğer bir yolu da tarama büyülteci kullanmaktır. Bu, ışın izini yatay yönde 5 veya 10 defa genişletir. Daha sonra, yatay konum kontrolu, ilgilenilen bölge ekran üzerinde görülünceye kadar ayarlanır. Ancak yukarıdaki problem için bu büyütme şüphelenilen küçük darbeyi görmeye pek yeterli olmayacaktır. Belki geciktirilmiş tarama problemi açığa çıkarılabilecektir.

GECİKTİRİLMİŞ TARAMA

Öncelikle iki tane zaman/bölme kontrolu, iki tane eğim kontrolu ve iki tane seviye kontrolu olduğuna dikkat edin. Kontrolların bir takımı ana tarama, diğer takımı da geciktirilmiş tarama içindir. (Eğer uygun bir osiloskobunuz yoksa Şekil 1.10 a bakın.) Şekil 1.11 a daki gibi kararlı bir görüntü ana tarama kontrolları kullanılarak elde edilmiştir. Geciktirme zaman çarpanı alanının ortasına getirilmiştir ve geciktirilmiş zaman/bölme seçicisi ana taramadan daha hızlı bir taramaya (Örneğin 1 µs/bölme ye) ayarlanmıştır. Şiddetlendirilmiş tarama çalıştırılınca, görüntülenen izin bir parçası daha parlak veya şiddetlendirilmiş gözükecektir. Bu parlak bölgenin genişliği geciktirilmiş taramanın zaman/bölme ayarına karşılık gelir. Geciktirilmiş taramanın zaman/bölme kontrolunu döndürerek parlak bölgenin genişliğinin değiştiğini gözleyebilirsiniz.

Parlak kısmın iz boyunca olan yeri geciktirme zaman çarpanı kontrolu tarafından belirlenmektedir. Geciktirilmiş tarama düğmesine basıldığında, parlak kısım bütün ekranı dolduracak şekilde genişler. Bazı osiloskoplar hem ana tarama görüntüsünün hem de genişletilmiş veya geciktirilmiş tarama görüntüsünün aynı anda görülmesine imkân verirler. İki zaman tabanı birbirinden bağımsız olduklarından, geciktirilmiş zaman tabanını daha yüksek hızlara ayarlayarak ilgilenilen bölgenin daha fazla büyütülmesini sağlayabilirsiniz. Parlak kısım, şüphelenilen darbenin varlığı veya yokluğu açıklığa kavuşuncaya kadar geciktirme zamanı çarpanı aracılığı ile B bölgesi boyunca hareket ettirilir (Şekil 1.11 *b* ye bakınız).

TETİKLEMELİ GECİKTİRİLMİŞ TARAMA

Geciktirilmiş taramanın daha geliştirilmişi, tetiklemeli geciktirilmiş taramadır. Bu modda, geciktirilmiş tarama, ana tarama görüntüsünde seçilen geçiş seviyesinde tetiklenir. Sadece geciktirilmiş taramada, geciktirme zaman çarpanı, parlak kısmı bütün ekran boyunca düzgün olarak kaydırır. Tetiklemeli geciktirilmiş taramada ise geciktirilmiş parça sadece geciktirilmiş tarama zaman tabanının eğim seçme anahtarıyla seçilen + veya - kenar, yani geçişle, başlayabilir. Eğer pozitif eğim seçilmişse, geciktirilmiş zaman çarpanı döndürüldükçe, parlak kısım görüntülenen dalgaşeklinin bir pozitif kenarından bir sonraki pozitif kenarına sıçrar.

Şekil 1.12 de gösteriline benzer bir dalgaşeklini incelerken, fazla büyütme gerektiğinde, tetiklemeli geciktirilmiş tarama görüntüye daha az titreme verir, çünkü tarama sadece bir geciktirme aralığının sonunda tetiklenmek yerine belirli bir geçişle tetiklenmektedir.



Şekil 1.12 Video dalgaşekli

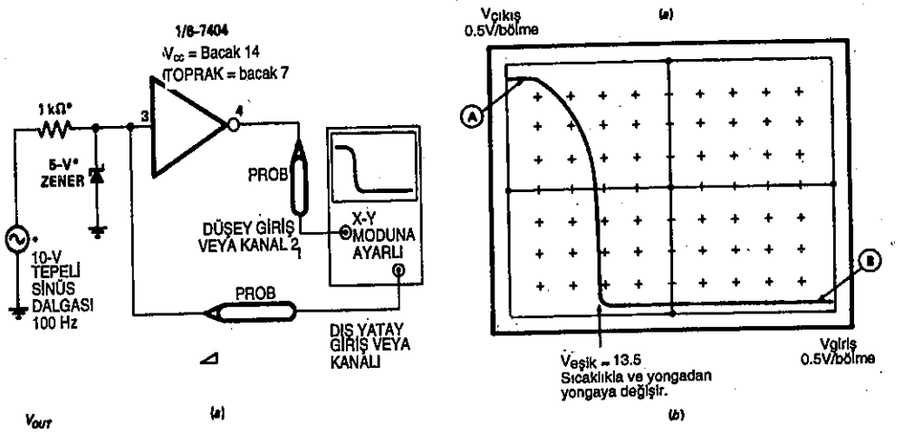
DIŞ YATAY YÜKSELTEÇ

X-Y TARAMA MODU

Bir mantık kapısının transfer fonksiyonunu belirlemek gibi bazı uygulamalarda, osiloskoptaki görüntünün yatay ekseninin zaman yerine gerilimi temsil etmesine gerek duyabilirsiniz. Bu, zaman/bölme kontrolunu yükselteç konumuna çevirerek veya başka bir şekilde dış yatay yükselteç girişini seçerek yapılır. Bazan buna *X-Y* modu da denir, çünkü düşey eksen standart grafik

terminolojisindeki y eksenine ve yatay eksende x eksenine karşılık gelir. Ekran üzerindeki görüntünün yatay konumu dış yatay girişe uygulanan d.a. veya a.a. sinyallerinin genliğine orantılıdır (düşey girişe uygulanan girişin düşey görüntüyü etkilediği gibi). Yatay yükseltece uygulanan bir pozitif giriş gerilimi, ekranın yüzüne baktığınızda, sağa doğru bir sapma oluşturur. Bazı osiloskoplarda yatay kazanç değişkendir, fakat bir çoğunda 50 mV/bölme lik sabit bir yatay sapma vardır.

Alternatif akım teorisi laboratuvarında sinüs dalgasının faz ilişkilerini gösterirken Lissajous şekilleri yapmak için *X-Y* modunu kullanmış olabilirsiniz. Burada bir mantık kapısının transfer eğrisinin (Bölüm 2 de üzerinde daha fazla durulacaktır) analizini yapmak için bu modu nasıl kullanacağımızı göstereceğiz. Şekil 1.13 *a*, 7404 gibi bir TTL DEĞİL kapısının çıkış gerilimini, uygulanan giriş gerilimine göre görüntülemek için gerekli yapıyı göstermektedir. Kapının çıkışı düşey girişe bağlanmış ve düşey sapma 0,5 veya 1 V/bölme konumuna getirilmiştir. Kapının giriş tarama sinyali osiloskopun dış yatay girişine bir *X10* probla bağlanmıştır. Eğer osiloskopun 50mV/bölmelik sabit yatay sapması varsa, bu takdirde prob ve osiloskop birlikte 500 mV/bölme lik bir denk yatay sapma verir. İlgilenilen giriş gerilimi aralığı 0 ile 5V (5000mV) arası olduğuna göre, 500mV/bölme lik 10 yatay bölme, girişi görüntülemek için iyi bir ölçek seçimi olacaktır.



Şekil 1.13 *(a)* Bir mantık kapısının transfer eğrisini göstermek için düzenek. *(b)* Bir TTL 7404 ters çeviricisinin transfer eğrisi.

Eğer güç ve sinyal kapıya uygulanırsa, osiloskoptaki görüntü Şekil 1.13 b dekine benzeyecektir. Buna kapının girişten-çıkışa transfer eğrisi denir. Girişteki sinüs dalgası sıfırdayken, kapının çıkışı yüksektir ve böylece görüntü A noktasındadır. Sinüs dalgası pozitif yönde artarken, iz sağa doğru hareket eder. Giriş gerilimi yaklaşık 1,3 V luk eşik gerilimini geçince, kapının çıkışı düşmeye başlar ve iz *B* noktasına kayarken minimum değerine ulaşır. Sinüs dalgası pozitif tepeden tekrar sıfıra doğru dönerken, aynı eğrinin üzerinden geçilir. Zener diyodu, kapının girişinin zarar görmesini önler. Eğer üretecin çıkışı kaza ile çok yükseğe ayarlanmışsa, zener diyodu kapının girişindeki gerilimi 5 V ile sınırlar. Sinüs dalgasının negatif yarıperiyotlarının hiç bir etkisi yoktur, çünkü zener diyodu kapı girişinin 0,7 V tan daha fazla negatif olmasını engeller, fakat pozitif yarıperiyotlar aynı eğriyi çizmeye devam ederek kesintisiz bir görüntü verir. Eğer görüntü kalibre edilmişse ve ekranın sol kenarından başlamışsa, **geçiş** geriliminin eşiği ekranda tayin edilebilir. Dış yatay girişi kullanırken hatırlamanız gereken bir nokta bu girişin bant genişliğinin genellikle sadece 1-2 MHz olduğudur. İki düşey giriş kanalı olan birçok osiloskop, kanal 2 düşey girişinin *X-Y* görüntüsünün yatay bileşeninin girişi olarak kullanılmasına imkân vermektir. Böylece görüntünün hem düşeyi hem de yatayı kalibre edildiğinden ve volt/bölme kontroluyla kontrol edildiğinden bu avantajlıdır.

Z EKSENİ

Z ekseni, ışının şiddetini module etmekle ilgilidir. Dış ışık şiddet (intensity) veya z ekseni girişi genellikle osiloskopun arka panosu üzerindedir. Tipik bir Tekronix osiloskopunda, eğer ön panodaki şiddet kontrolu iz gösterecek şekilde ayarlanmışsa, z ekseni girişine uygulanacak -5 V luk bir sinyal ışını kapatacaktır (kesecektir). Eğer şiddet kontrolu iz görünemeyecek bir değere ayarlanmışsa ve z girişine uygulanacak +5 V luk bir sinyal ışını açacaktır (yani iz görülecektir). Bu özellik, şiddetlendirilmiş tarama modunda izin içeriden parlaklaştırılmasına benzer şekilde, izin vurgulanmak istenen kısımlarını şiddetlendirmek için kullanılabilir. Diğer bir uygulama düşey ve yatay girişlere senkronize edilmiş düşey ve yatay tarama uygulayarak bir televizyonda olduğu gibi bir raster görüntüsü üretmeyi içerir. Z ekseni girişine uygulanan video bilgisi, bir resim veya grafik görüntü üretmek üzere, ışın ekranı tararken şiddetini module eder.

SAKLAYICI OSİLOSKOPLAR

Osiloskopları incelerken, standart çift-izli, tetiklemeli osiloskoplardan bahsettik. Birçok uygulama için bu tip yeterlidir. Ancak, bu tip bir osiloskopla alçak frekanslı veya tek-atışlı sinyalleri gözlerken, yüksek frekanslarda ol-

duğu gibi bütün dalgaşeklini aynı anda görmek yerine, ekran boyunca dalgaşeklini izlemek üzere yavaşça hareket eden parlak bir nokta göreceksiniz. Fotoğrafik bir belleğiniz yoksa, hareketli bir noktadan oluşan bir görüntüyü analiz etmek zordur. Problemin bir çözümü, daha önce anlatılan tek-tarama moduyla birlikte bir fotoğraf makinası kullanmaktır. Diğer bir çözüm ise ışının taradığı yerlerdeki izleri görünür tutan özel bir katot ışın tüplü bir osiloskop kullanmaktır. Böylece görüntü saklanmış olur. Kullanılan saklayıcı osiloskopun tipine bağlı olarak görüntü saniyelerce veya günlerce saklanabilir. Bir silme kontrolu, istendiği anda görüntüyü silmeye imkân verir. Alçak bantgenişlikli, yavaş taramalı görüntüler için saklama doğrudan katot ışınlı tüpün forforu üzerine yapılır. Yüksek bandgenişlikli, hızlı taramalı görüntüler için saklama katot ışınlı tüpün fosforunun arkasındaki ağ üzerine yapılır. Bazı saklayıcı osiloskoplar ekranı ayrı ayrı yazılıp silinebilen üst ve alt yarıya bölerler. Böylece bir devrenin bir dalgaşeklini görüntüleyip saklar, devrede değişiklik yapar ve sonra elde edilecek yeni dalgaşeklini eskisiyle karşılaştırabilirsiniz. Tekronix 7834 gibi çok yüksek tarama oranı olan saklayıcı osiloskoplar, sadece bir defa olan veya tekrarlama oranı çok düşük olan, nanosaniye genişlikli darbeleri yakalamak ve ölçmek için kullanılabilir.

Bazı saklayıcı osiloskoplar önce analog giriş sinyalini bir seri sayısal (ikilik kodlanmış) değerlere çevirir. Sayısal değerler, yarıiletken bellekte saklanabilir ve sonra katot ışınlı tüpte görüntü oluşturmak üzere tekrar tekrar okunurlar. Ayrıca sayısal değerler yükselme zamanları, periyotları, frekansları ve tepe gerilimlerini hesaplamak üzere bir mikrobilgisayarda kolayca işlenebilirler. Ayrıca bu değerler ekranda gösterilebilir.

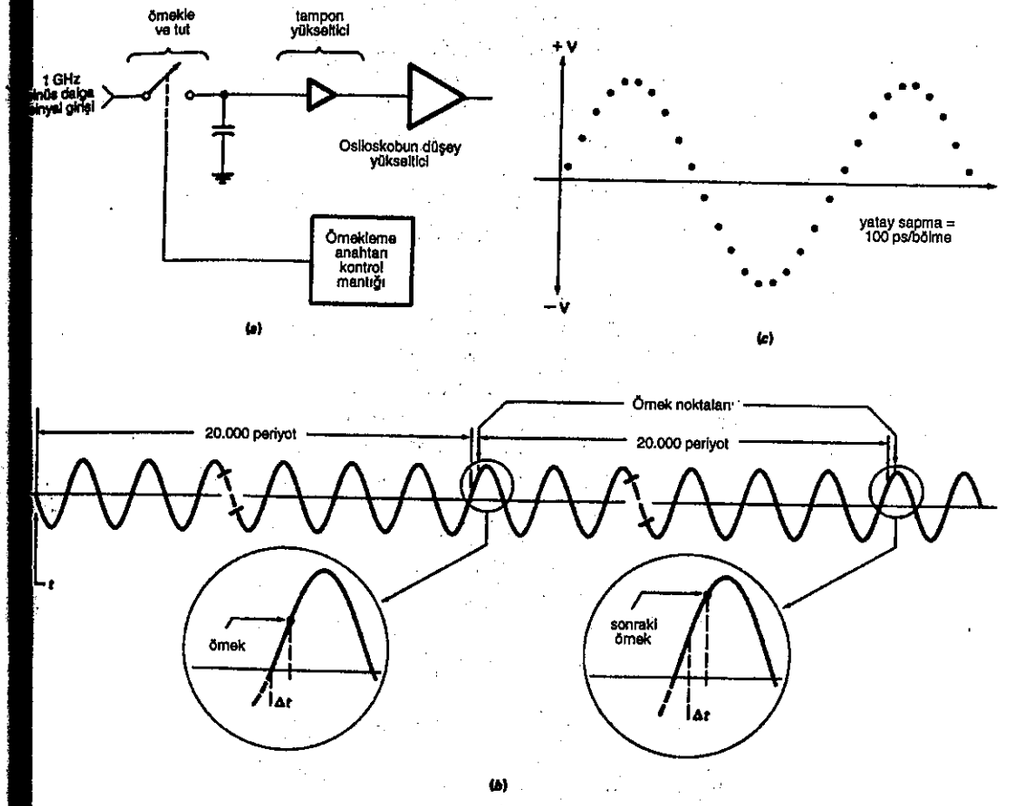
Bir saklayıcı osiloskopun düşey yükselteci, zaman tabanı, tarama modları ve tetiklemesi daha önce tanıtılan standart tetiklemeli tarama osiloskopununki ile aynıdır. Ancak çok değişik tipte saklayıcı osiloskoplar olduğundan, yazma, saklama ve silme yordamları için elinizdeki osiloskopun kendi kılavuzuna başvurmanız gerekmektedir. Saklayıcı ekranı korumak için çok önemli şu noktaları unutmayın: Işık şiddetini aşağıda tutun, ekranın aynı alanını tekrar tekrar kullanmaktan kaçının, saklama süresini en azda tutun ve gerekmedikçe saklama modunu kullanmayın.

ÖRNEKLEYEN OSİLOSKOPLAR

Standart osiloskopun kullanılamadığı diğer bir alan da 350 MHz in üstündeki yüksek-frekans sinyallerinin veya bunların 1 ns nin altındaki darbe yükselme zamanlarının görüntülenmesidir. Eğer 8 cm lik düşey ve 20 cm lik yatay boyutları olan bir ekranı 1 GHz lik bir sinüs dalgası ile doldurmak isterseniz, 20.000 cm/μs lik bir ışın tarama hızı gerekmektedir. Bu hızda, ışın ekran üzerinde o kadar hızlı hareket etmektedir ki görülen bir iz bırakmaz. Herhangi bir osiloskopta tarama hızını artırdığınızda, izi görülür tutabilmek için ışık

şiddetini artırmanız gerektiğini daha önce belki farketmişsinizdir. Bu yüksek-frekans sinyallerini standart veya gerçek-zaman osiloskoplarında görüntülemedeki diğer bir problem de düşey yükselteç için aşırı büyük bir bandgenişliğinin gerekmesidir.

Düzgün şekilde kendini tekrar eden sinyaller için bu problemler örnekleyen osiloskoplar kullanılarak çözülür. Görüntünün doğrudan giriş sinyali tarafından çizildiği bilinen gerçek-zamanlı görüntü yerine bir örnekleyici değişik dalgalardan derlediği bir dizi nokta değerleri ile görüntüyü oluşturur. Bu, aşağıdaki şekilde gerçekleştirilmektedir.



Şekil 1.14 ***(a)*** **Bir örnekleyen osiloskop için örnekle ve tut devresi.** ***(b)*** **Bir 1 GHz lik sinüs dalgası örneklenen noktaları göstermektedir.** ***(c)*** **Örnekleyen osiloskoptaki görüntü.**

Örnekleyen osiloskobun girişi bir anahtar ve kapasitörle temsil edilebilen (Şekil 1.14 a) bir *örnekle-ve -tut devresidir*. Anahtar kapanınca kapasitör giriş değerini takip etmek üzere şarj eder. Anahtar açıldığında kapasitör o anda üzerinde olan gerilimi tutar. Bu gerilim yüksek empedanslı tampon yükselteç ve düşey yükselteç tarafından görüntüde bir nokta elde etmek için yükseltilir. Anahtarın her kapanıp açılışıyla yeni bir nokta elde edilir. Tipik olarak anahtar 50 KHz hızında kapanıp açılır, yani 1 GHz lik bir sinüs dalgası görüntüleniyorsa yaklaşık 20.000 giriş periyodunda bir kapanıp açılır. Her 20.000 giriş periyodundan yalnız birisinden bir örnek alınmaktadır, fakat periyotlar aynı olduğu için bunun sakıncası yoktur. Şimdi, eğer bir periyodun hep aynı anında bir örnek alınsaydı, osiloskop her örnek için aynı değeri elde edecekti ve görüntü sadece düz bir yatay çizgi olacaktı. Eğer 1 inci örnek t anında alınırsa (Şekil 1-14 b ye bakın), 2 inci örnek 20.000 periyot artı küçük bir Δt zamanı sonra alınır ve 3 üncü örnek de bundan 20.000 periyot artı Δt kadar sonra alınır. Böylece örnek noktaları giriş dalgaşekli boyunca Δt aralıklarla alınmış gibi görülür ve dalgaşeklinin tamamı görüntülenmiş olur (Şekil 1.14 c ye bakın). Örnekleyen osiloskobun yatay zaman tabanı pikosaniye/bölme veya nanosaniye/bölme-ye denk gelecek şekilde kalibre edilmiştir. Böylece, yükselme zamanları ve frekanslar ekrandaki görüntüden kolayca belirlenebilir.

SAYISAL OKUMALI OSİLOSKOPLAR

HP 1725 A gibi bazı osiloskopların içinde a.a./d.a. gerilim, direnç ve akım ölçmek için bir sayısal multimetre vardır. Multimetre ayrıca ekran üzerindeki görüntünün şiddetlendirilmiş (parlak) iki noktası arasındaki zamanı nanosaniye veya mikrosaniye olarak sayısal okuyabilmektedir. Bu özellik görüntülerdeki yükselme zamanlarını ve gecikmeleri kolayca ölçmenizi sağlar.

DİĞER TEST CİHAZLARI

DARBE ÜRETEÇLERİ VE ZAMAN-İŞARET ÜRETEÇLERİ

Bir darbe üretecinin sinyal kaynağı olarak kullanılması oldukça basittir, ancak bazı tedbirleri almanızda yarar vardır. Birincisi bir darbe üretecinin bir devreye bağlanmadan önce, darbe üreteci üzerindeki kaydırma ve genlik kontrollarının kullanmakta olduğunuz mantık ailesinin nizami seviyelerine ayarlanmış olduğunu, bir osiloskopu d.a. kuplaj modunda kullanarak, kontrol edin. (Bilinen mantık ailelerinin nizami seviyeleri için Bölüm 2 ye başvurun). Darbe üreteci gibi alçak-empedanslı bir kaynaktan kazara bir aşırı gerilim verilmesi mantık kapısının girişini kolayca tahrip edebilir. İkincisi, eğer darbe üretecinin çıkış empedansı 50 Ω ise, sinyali üreteçten devreye taşımak

için RG-58 gibi bir 50 Ω luk kablo kullanın ve sonra kablonun devreye bağlandığı yerde orta iletkenle kablo ekranı arasına 50-Ω luk bir direnç bağlayın. Bunun sebepleri Bölüm 2 de *"Iletim Hatları"* başlığı altında açıklanmaktadır. Üçüncüsü, gücü kapatırken aynı anda veya daha önce darbe üreticini de kapatmalısınız, çünkü aksi halde, mantık yongalarında ters yönde akan akımlar oluşacaktır.

Zaman-işaret üreteçleri hassas zaman aralıklarında ince darbeler üreten yüksek doğruluklu darbe üreteçleridir. Zamanlamanın doğruluğunu bir kristal osilator sağlar. Bu zaman işaretleri osiloskop ve diğer cihazların zaman tabanlarının kalibrasyonunda kullanılır.

MANTIK-DURUM ANALİZÖRLERİ

Rastgele ulaşılabilir bellek gibi birçok çıkış hattı olan sayısal devrelerin analizini yaparken, sekiz veya daha fazla sinyali aynı anda görüntüleyebilmek ve karşılaştırabilmek avantajlıdır. Bu mantık-durum analizlerinin işlevlerinden biridir. En basit şekliyle, bir mantık-durum analizörü sekiz veya daha fazla giriş kanalını bölümleyerek ve aralarında değişerek her biri için bir osiloskop görüntüsü veren bir cihazdan ibarettir. Bir kelime tanıyıcı, giriş hatlarında belirli bir kelime yani 1 ler ve 0 lar deseni olduğunda tetikleme imkânı verir. Mantık analizörleri ayrıca giriş kanallarından gelen bilgiyi 1 ler ve 0 lar dizisi olarak veya noktaların çizimi olarak veya onaltı-tabanlı kelimeler olarak gösterebilir. Mantık analizörlerinin çoğu, sürekli kaydeden belleklere sahiptir ve bu tetiklemeden önceki verilerin görüntülünmesine imkân verir. Bölüm 5 te mantık analizörlerinin bazı özel devrelerin analizde kullanışlarını anlatacağız.

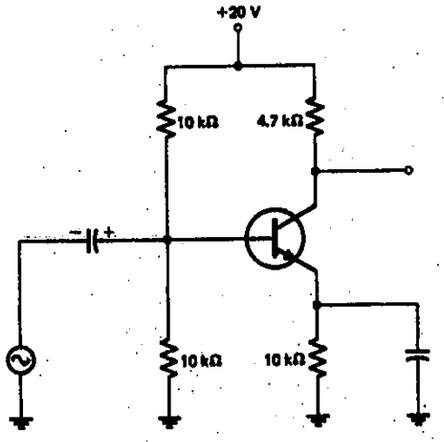
GÖZDEN GEÇİRME SORULARI VE PROBLEMLER

1. Lehimsiz temel örnek kartlarının avantaj ve dezavantajları nelerdir?
2. Üzerinde devre kurmak üzere üniversal baskılı devre kartını ne zaman tercih edersiniz?
3. Baskılı devre kartına giren güç kaynağı uçlarını yangeçitlemek için niçin 0,1 µF lık bir kapasitörle paralel 25 µF lık kapasitör konur?
4. Papatya zinciri bağlantı ne gibi problemlere sebep olur?
5. Niçin devre elemanlarının yerleşimi mümkün olduğunca şemadaki yerleşimi izlemelidir?
6. Lehimsiz temel örnek kartlarında hangi tip ve kalınlıkta tel kullanılmalıdır?

7. Bir şemadan devre bağlantılarını yaparken nerede kalmış olduğunuzu nasıl hatırlayabilirsiniz?
8. Renk-kodlu bağlantı yapma niçin önemlidir?
9. Yeni kurulmuş bir devrenin ilk test yordamını açıklayın.
10. Mantık seviyelerini kontrol ederken mantık probunun çoğu sayısal voltmetreye göre avantajı nedir?
11. *a.* Bir elektrolitik kapasitörün kızmasının muhtemel sebepleri nelerdir?

    b. Bir entegre devrenin kızmasının muhtemel sebebleri nelerdir?
12. Mantık probunun kullanılışını anlatın. Bu problar hangi devrelerde kullanılmamalıdır?
13. Bölümlenmiş ve değişen tarama modları arasındaki fark nedir? İki yüksek-frekans sinyalini görmek için hangi mod kullanılır?
14. Mantık sinyallerini görmek için a.a. kuplaj mı, yoksa d.a. kuplaj mı daha iyidir?
15. Şekil 1.15 basit bir transistör yükselteç katının şemasını göstermektedir.

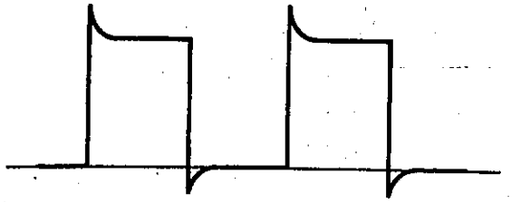
    a. Bazdaki öngerilimi,
    b. Bazda görülecek giriş sinyalini,
    c. Kollektördeki çıkış sinyalini, görüntülemek için hangi tür kuplaj kullanırsınız?



Şekil 1.15

16. Osilokop girişlerinde problar niçin kullanılır?
17. *a.* En iyi frekans tepkisi için osiloskop probları nasıl ayarlanır?

    b. Bir X10 prob, kalibratör karedalga çıkışına bağlıyken, bir osiloskop Şekil 1-16 daki dalgaşeklini görüntülemektedir. Buradaki problem nedir ve bunu nasıl düzeltirsiniz?



Şekil 1.16

18. Tetikleme seviye kontrolu ile tetikleme eğim kontrolu arasındaki fark nedir?
19. Tek taramanın faydası nedir?
20. Şiddetlendirilmiş ve geciktirmeli taramanın kullanılışını anlatın.
21. X-Y görüntü modunda, osiloskoptaki görüntünün iki ekseni ile hangi miktarlar temsil edilmektedir?
22. *a.* Saklayıcı osiloskopla hangi tip sinyaller en iyi görüntülenir?

    b. Işının şiddeti niçin düşük tutulmalıdır?
23. *a.* Örnekleyen osiloskoplar hangi tip ölçümler için gereklidir?

    b. Örnekleyen osiloskopların çalışma prensibi nedir?
24. Sinyal üreteci bir devreye bağlanmadan önce niçin sinyal üretecinin çıkışındaki genlik ve kayma bir osiloskopla kontrol edilmelidir?
25. Toplam 1/2 inç (12,7 mm) uçları olan 0,001 µF lik bir kapasitörün en iyi yangeçitlediği frekans nedir?

REFERANSLAR/BİBLİYOGRAFYA

Using Your Oscilloscope Probe, Uygulama notu, Tekronix, Inc. / A-3051-1.

Allison, Gordon (ed.): Teknique-Understanding Oscilloscope Triggering Controls" Tekscope, Cilt 6, No.5, (Kasım/Aralık 1974), sayfa: 11-13.

Global Specialties Corporation Breadboard and Test Equipment for the Professional and Hobbyist (Katalog), Global Specialties Corporation, 1982.

Bechtold, Jim (ed.): "Troubleshooters in Action", Hewlett-Packard'dan Bench Briefs-Service Information, Eylül/Ekim 1977, Sayfa: 1-5.

Tektronix Products Catalog, Tektronix, Inc. 1982.

Tektronix 5440 Oscilloscope Instruction Manual, Tektronix, Inc., Nisan 1976.

Tektronix 5B42 Delaying Time Base Instruction Manual, Tektronix, Inc., Haziran 1976.

Tektronix 5111 Storage Oscilloscope Instruction Manual, Tektronix, Inc., Ağustos 1976.

2. BÖLÜM

SAYISAL MANTIK KAPISI KARAKTERİSTİKLERİ VE BAĞDAŞTIRMA

Birinci bölümde devre yapım tekniklerini ve bilinen test cihazlarının kullanımını görmüştük. Bu İkinci Bölümde bir ön tanıtımdan sonra, temel mantık kapıları ve bunların giriş-çıkış karakteristikleri üzerinde duracağız. Birinci Bölümde öğrendiğimiz test cihazları hakkındaki bilgileri kullanarak bu karakteristikleri ölçüp karşılaştırabilirsiniz. Böylece bu bilgiler kafanızda yerleşecek ve sizlere test cihazları ile daha fazla deneyim sağlayacaktır.

Bu bölümü bitirdikten sonra şunları yapabileceksiniz:

1- Analog ve sayısal miktarlar arasındaki farkı açıklayabileceksiniz.

2- Onluk sayıları ikilik sayılara ve ikilik sayıları onluk sayılara çevirebileceksiniz.

3- VE, VEYA, VE DEĞİL, VEYA DEĞİL, ters çevirici ve özel VEYA mantık kapılarının doğruluk tablolarını yazabileceksiniz.

4- Bilinen mantık kapılarının giriş akım ve gerilimlerini, çıkış akım ve gerilimlerini ve yayılma gecikme zamanlarını belirlemek için üretici firmanın veri katologlarını kullanabileceksiniz.

5- Bilinen bir mantık ailesine ait bir entegre devre kapının iç yapısının şeması verildiğinde, alçak ve yüksek girişler için çıkışın durumunu öngörebileceksiniz.

6- Bilinen mantık aileleri için giriş sinyallerini üretecek veya uygun şekle getirecek devreleri çizip kurabileceksiniz.

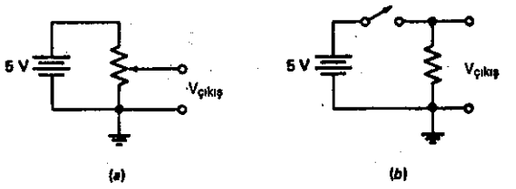
7- Bilinen mantık ailelerini birbirleriyle veya gösterge ve röle gibi başka birimlere bağdaştırmak için devreler çizip kurabileceksiniz.

ANALOGA KARŞIN SAYISAL

Elektronik, iki temel miktar türü içerir: analog ve sayısal. Analog miktarlar herhangi bir değeri alabilir. Örneğin, Şekil 2-1a daki devrenin çıkış gerilimi 0 ile 5 V arasında sonsuz sayıdaki değerlerden herhangi birine ayarlanabilir. Sıcaklık, basınç gibi birçok doğal miktarlar analogtur. Bir sinüs dalgası bir analog sinyal örneğidir. Hareketli sargılı voltmetre bir analog cihazdır, çünkü voltmetrenin iğnesi sonsuz sayıda farklı noktayı gösterebilir.

Bir sayısal miktar veya sinyal sadece belirli sabit değerler alabilir. Örneğin Şekil 2-1b deki devrenin çıkış geriliminin sadece iki mümkün değeri vardır: Eğer anahtar açıksa 0 V ve anahtar kapalıyla 5V. Eğer bir kare dalganın yükselme ve düşmesinin ani olduğu kabul edilirse, kare dalga da sayısal sinyale örnek olur. Adından da anlaşılacağı üzere, bir sayısal voltmetre iyi bir sayısal devre örneğidir. Gösterge sadece 1,95 V, 1, 96 V gibi bazı sabit değerleri gösterebilir. Ayrıca, bu sayısal voltmetre, test edilmekte olan gerilimi ölçmek ve hesaplamak için sayısal devreler kullanır.

Elektronik hesap makinaları ve bilgisayarların çoğu, 0 ve 5 V gibi sadece iki değeri olan sayısal sinyallerle çalışır. Bu arada akla gelen ilk soru, bir hesap makinasının sadece iki gerilim seviyesini kullanarak nasıl 327 gibi bir sayıyı belleğinde saklayabildiğidir. Bunun cevabı ise hesap makinası ve bilgisayarların sayıları ikilik tabanda kodlanmış şekilde saklayıp işlediği şeklinde olacaktır.



Şekil 2-1 Analog ve sayısalı gösteren devreler. *(a)* Analog çıkışlı devre. *(b)* Sayısal çıkışlı devre.

İKİLİK KOD: İKİ TABANLI SAYI SİSTEMİ

ONLUK SİSTEMİN İNCELENMESİ

İkilik sayı sisteminin yapısını öğrenmek için, ilk adım bilinen desimal, yani 10-tabanlı sayı sistemini incelemektir. Şekil 2-2 *a* her basamağı 10 un kuvvetleriyle ifade edilen bir onluk sayıyı göstermektedir. Onluk sayı 5346,72 nin basamakları, bu sayıda 5 tane binlik, 3 tane yüzlük, 4 tane onluk, 6 tane birlik, 7 tane onda birlik ve 2 tane yüzde birlik olduğunu söylemektedir. Herhangi bir tabanlı sayı sisteminde, gerekli olan sembol sayısı tabanın değerine eşittir. Nitekim, onluk sayı sisteminde 0 dan 9 *a* kadar 10 tane sembol vardır. Herhangi bir basamaktaki sayı en yüksek sembolün değerini geçerse bir sonraki basamağa taşınan 1 elde oluşur ve basamak sıfıra döner. Bir arabanın kilometre sayacı bunun iyi bir örneğidir.

Basamaklar için herhangi bir sayının kuvvetlerini kullanarak bir sayı sistemi kurulabilir, ancak bazı tabanlar diğerlerinden daha yararlıdır. 10 farklı gerilim seviyesini saklayan ve işleyen elektronik devreler kurmak zordur, halbuki iki seviye ile çalışan devreler kurmak daha kolaydır. Bu nedenle, ikilik yani 2-tabanlı sayı sistemi kullanılmaktadır.

5	3	4	6 ,	7	2
10^3	10^2	10^1	10^0	10^{-1}	10^{-2}

(a)

			1	0	1	1	0 ,	1	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}
128	64	32	16	8	4	2	1	$\frac{1}{2}$	$\frac{1}{4}$

Şekil 2-2 Sayı sistemleri ve basamakların değerleri *(a)* Onluk *(b)* İkilik.

İKİLİK SİSTEM

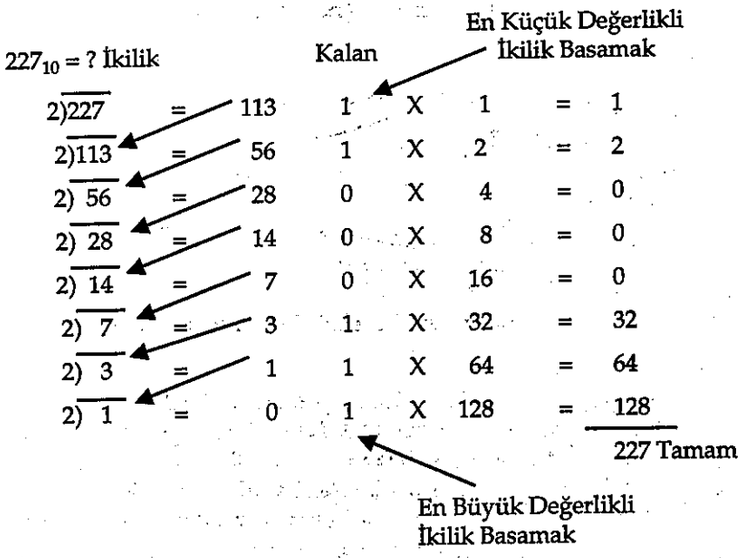
Şekil 2-2*b* bir ikilik sayının her basamağının değerini göstermektedir. Her basamak 2 nin bir kuvvetine karşılık gelmektedir. Bir ikilik sayının her basamağına bit (binary digit) denir. Bir ikilik sayıdaki virgülün sağındaki basamaklar sayıların 1 den küçük kesir kısımlarını temsil etmektedir. İkilik sistem sadece iki sembol kullanır, 0 ve 1. Bu nedenle ikilik sayı sisteminde 0, 1, 10, 11, 100, 101, 111, 1000, ... şeklinde sayarız.

Bir ikilik sayıyı, karşılığı olan onluk sayıya çevirmek için, her basamağı onluk değeri ile çarpıp bunları toplayın. Örneğin 101 ikilik sayısı $1x2^2+0x2^1+1x2^0$ ı yani 4+0+1 i göstermektedir. Elbette, bu da onluk sistemde 5'e eşittir. İkilik sayı 10110,11 için ise

N = 1x16+0x8+1x4+1x2+0x1+1x0, 5+1x0, 25=22, 75

onluk karşılığı elde edilir.

Onluk bir sayıyı ikilik sayıya çevirmek için bilinen iki yöntem vardır. Birincisi yukarıda anlatılan ikilikten onluğa çevirme yönteminin tersidir. Örneğin, 21 onluk sayısını ikilik sayıya çevirmek için, bu sayıya sığan 2 'nin en büyük kuvvetini bu sayıdan çıkarın. 21_{10} un içine sığan 2 nin en büyük kuvveti 16, yani 2^4'tür. 21 den 16 çıkarılınca 5 kalır. 2^4 basamağına bir 1 koyun ve kalana bir azaltılmış 2 nin kuvvetinin sığıp sığmadığına bakın. 2^3=8 olduğuna göre ve 8 sayısı 5 kalanına sığmayacağına göre 2^3 basamağına bir 0 koyun. Sonra bir sonraki 2 nin kuvvetini arayın. Bu durumda, 2^2=4 bir sonraki 2 nin kuvvetidir ve 5 e sığar. Böylece 2^2 basamağına 1 konur. Eski kalan 5 ten 2^2=4 çıkarılınca yeni kalan 1 olarak bulunur. Bu kalana 2^1=2 sığmayacağından 2^1 basamağına 0 konur. 2^0 basamağına da bir 1 konur, çünkü 2^0=1 dir ve bu da kalana tam olarak sığmaktadır. Sonuçta 21_{10} sayısının 10101 ikilik sayısına eşit olduğu görülür. Çevrim işleminin anlatılması biraz karışık fakat yapılması kolaydır. 46_{10} sayısını çevirmeyi deneyin. 101110 elde etmeniz gerekir.

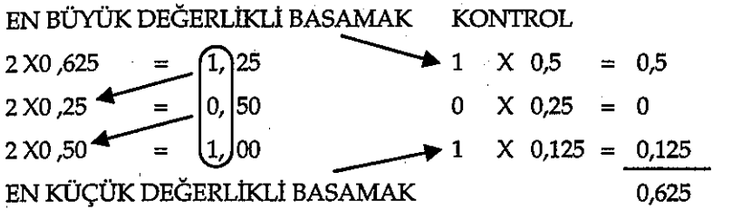


$227_{10} = 11100011_2$

Şekil 2-3 Onluk bir sayının ikilik bir sayıya ve ikilik bir sayının onluk bir sayıya çevrilmesi.

Bir onluk sayıyı ikilik sayıya çevirmenin ikinci yöntemi ise Şekil 2-3 te gösterilmiştir. Onluk sayıyı 2 ye bölün ve bölüm ve kalanı şekilde gösterildiği gibi yazın. Bu bölümü ve bundan sonraki bölümleri bölüm sıfır oluncaya kadar 2 ye bölmeye devam edin. Kalanlar sütunu, verilen onluk sayının ikilik karşılığı olacaktır. En büyük değerlikli basamağın sütunun en altında, en küçük değerlikli basamağın sütunun en üstünde olduğuna dikkat edin. İkilik sayının doğru olduğunu Şekil 2-3 ün sağ tarafında görüldüğü gibi ikilikten onluğa tekrar çevirerek gösterebilirsiniz.

1 den küçük onluk sayıları ikilik sayılara çevirmeyi, Şekil 2-4'te görüldüğü gibi, noktanın sağındaki miktar sıfır oluncaya kadar iki ile çarparak ve her seferinde tamsayı kısmını ayırıp bir kenara not ederek yapabilirsiniz. Tamsayılar, en büyük değerlikli basamak, yani bit, sütununun en üstünde olmak üzere onluk sayının ikilik karşılığını gösterir. Onluk 0,625 sayısı ikilikte 0,101 dir. Bu örnektekinden farklı olarak, tam olarak dönüşmeyen onluk değerler için (virgülün sağındaki kısım hiç bir zaman sıfıra ulaşmazsa), çevrim işlemine istenilen basamak sayısına ulaşıncaya kadar devam edebilirsiniz.



Şekil 2-4 den az bir onluk sayının ikilik bir sayıya çevrilmesi.

Bu arada, sayıları onluk ifade ederken gerekli basamak sayısı ile ikilik sistemde ifade etmek için gerekli basamak sayısını karşılaştırmakta yarar vardır. Onluk sistemde bir basamak 0 dan 9 a kadar 10 sayıyı gösterebilir. İki basamak 0 dan 99 a kadar 10^2, yani 100 sayıyı, üç basamak 0 dan 999 kadar 10^3, yani 1000 sayıyı gösterebilir. İkilik sistemde de benzer bir desen vardır. Bir ikilik basamak 0 ve 1 olmak üzere iki sayıyı, iki ikilik basamak 0 dan 11 e kadar 2^2, yani 4 sayıyı ve üç ikilik basamak 0 dan 111 e kadar 2^3yani 8 sayıyı gösterebilir. O halde N onluk basamak 10^N sayıyı, N ikilik basamak ise 2^N sayıyı gösterebilir. Örneğin, sekiz ikilik basamak 0 dan 255 e kadar 2^8, yani 256 sayıyı gösterebilir.

TEMEL MANTIK KAPILARI

Daha önce çoğu bilgisayar ve sayısal devrelerin bütün sayılar ve miktarları göstermek için ikilik kodu kullandığını belirtmiştik. Bunun nedeni her basamağın yalnız 0 ve 1 gibi iki sembolle gösterilebilmesidir. Sayısal devrelerde bu iki sembole "durum" adı verilir. Şekil 2-5 bu iki durumu tanımlamakta çoğunlukla kullanılan koşul ve terimlerin bazılarını göstermektedir.

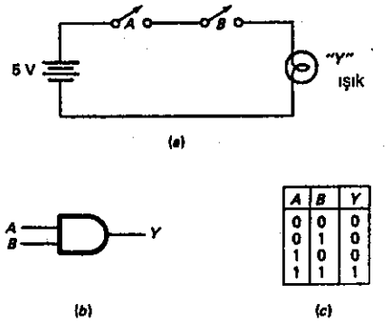
Sayısal devrelerin temel yapı taşları mantık kapılarıdır. Bunlara bu isim ikilik verileri mantık olarak işledikleri için verilmiştir. Bazı örneklerle bunu açıklayalım:

YÜKSEK	ALÇAK
1	0
DOĞRU	YANLIŞ
+5V	0V
AÇIK	KAPALI
AKTİF	AKTİF DEĞİL

Şekil 2-5 Pozitif mantık sayısal durumların bazı terim veya isimleri

VE KAPISI

Şekil 2-6 seri bağlı iki anahtardan oluşan basit bir devreyi göstermektedir. Işık sadece *A* anahtarı ve *B* anahtarı kapalıysa yanacaktır. Buna mantıksal VE fonksiyonu denir ve Şekil 2-6*b* deki mantık kapısı sembolüyle gösterilir.



Şekil 2-6 VE fonksiyonunun gösteriliş şekilleri. *(a)* Denk anahtar devresi. *(b)* Mantık kapısı sembolü. *(c)* Doğruluk tablosu

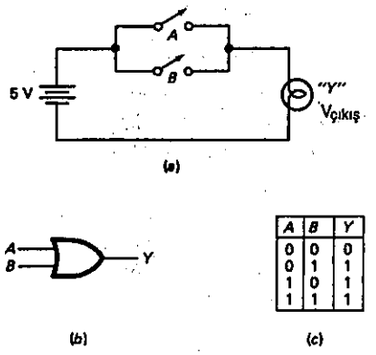
DOĞRULUK TABLOSU Her mümkün giriş durumu birleşimi için elde edilecek çıkış durumunu göstermenin basit bir şekli **doğruluk tablosu**yladır.

Şekil 2-6*c*, iki girişli bir VE kapısının doğruluk tablosunu göstermektedir. *A* ve *B* gibi iki tane ikilik giriş değişkeni olduğuna göre, 4 tane mümkün giriş birleşimi vardır. Bir mantık kapısı veya sayısal devrenin doğruluk tablosunu yazarken bütün mümkün giriş birleşimleri için çıkışları göstermek önemlidir. Bunu yapmanın kolay bir yolu Şekil 2-6*c* de görüldüğü gibi giriş durumlarını ikilik sayma sırasında yazmaktır. Üç girişli bir kapı için sekiz mümkün giriş bileşimi vardır, Bunlar ikilik sayılar 000, 001, 010, 011, 100, 101, 110 ve 111 den ibarettir. Bir VE kapısı herhangi bir sayıda girişe sahip olabilir.

BOOLE İFADELERİ Doğruluk tabloları mantık kapılarının tepkilerini göstermek için kullanılan geçerli bir yoldur, fakat biraz fazla yer ve zaman alırlar. Bir kapının mantık fonksiyonunu ifade eden daha derli toplu bir şekil Boole Cebiri ifadeleridir. Boole cebiri bildiğimiz cebire benzer fakat aynı değildir. Şekil 2.6 daki VE kapısının boole cebiri ifadesi $Y=A.B$ dir. *A* ve *B* arasındaki nokta VE fonksiyonunu gösterir. Çoğunlukla nokta ihmal edilir ve ifade basitçe $Y=AB$ şeklinde yazılır. Bir VE kapısı herhangi bir sayıda girişe sahip olabilir. Üç girişli bir VE kapısının ifadesi $Y=A.B.C$ veya $Y=ABC$ şeklindedir.

VEYA KAPISI

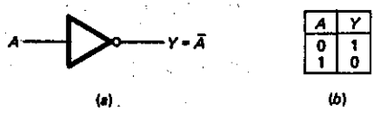
Çok karşılaşılan diğer bir mantık kapısı VEYA kapısıdır. Şekil 2-7*a* bir VEYA kapısının anahtarlı denk devresini göstermektedir. Anahtar *A* **veya** anahtar *B* kapalıysa ışık yanacaktır. Şekil 2-7*b* iki girişli bir VEYA kapısının şematik sembolünü ve Şekil 2-7*c* de doğruluk tablosunu göstermektedir. VEYA kapısı için boole cebiri ifadesi $Y=A+B$ dir. Artı işareti VEYA fonksiyonunu gösterir. $A=1$ ve $B=1$ olması durumunda doğruluk tablosundan bu işaretin bildiğimiz cebirle aynı anlam taşımadığına dikkat edin. Bir VEYA kapısının da herhangi bir sayıda girişi olabilir. Üç girişli VEYA kapısının boole ifadesi $Y=A+B+C$ şeklindedir.



Şekil 2-7 VEYA fonksiyonunun gösteriliş şekilleri. *(a)* Denk anahtar devresi. *(b)* Şematik sembol. *(c)* Doğruluk tablosu.

DEĞİL KAPISI (TERS ÇEVİRİCİ, INVERTER)

Diğer bir faydalı mantık kapısı ters çevirici yani DEĞİL kapısıdır. Şekil 2-8*a* bir ters çeviricinin şematik sembolünü, Şekil 2-8b de doğruluk toblosunu göstermektedir. Çıkış kısaca girişin tersi, zıddı, yani tümleyenidir. Bu kapı için boole ifadesi $Y=\overline{A}$ dır. *A* nın üzerindeki çizgi *A* nın tersini gösterir ve "*A* değil" şeklinde okunur.

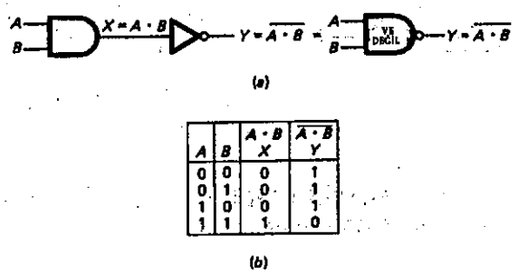


Şekil 2-8 Ters çevirici veya DEĞİL kapısı. *(a)* Şematik sembol. *(b)* Doğruluk tablosu.

VE DEĞİL KAPISI

VE kapıları ve VEYA kapıları diğer bazı temel kapıları oluşturmak üzere DEĞİL kapılarıyla birleştirilebilir. Bir VE kapısı DEĞİL kapısıyla birleştirilerek VE DEĞİL kapısı oluşturulabilir. Şekil 2-9*a* şematik sembolleri göstermekte, Şekil 2-9*b* de VE ve VE DEĞİL kapılarının doğruluk tablolarını göster-

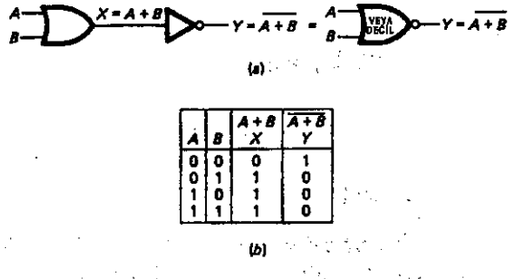
mektedir. VE DEĞİL kapısının çıkışları VE kapısının çıkışlarının tersidir. Şematik semboldeki çıkışta görülen küçük yuvarlak bunu, yani tersini alma işlemini göstermektedir. VE DEĞİL kapısının boole ifadesi $Y=\overline{A.B}$ veya $Y=\overline{AB}$ şeklindedir. VE DEĞİL kapılarını içeren devrelerde arıza ararken hatırlanması gereken önemli bir nokta VE DEĞİL kapısının girişlerinden herhangi birisi alçak (0) olduğunda çıkışının yüksek (1) olduğudur.



Şekil 2-9 VE DEĞİL fonksiyonunun türetilmesi. *(a)* Şematik semboller. *(b)* Doğruluk tabloları.

VEYA DEĞİL KAPISI

Çıkışında bir DEĞİL kapısı olan bir VEYA kapısı VEYA DEĞİL kapısını oluşturur. Şekil 2-10*a* VEYA DEĞİL kapısının şematik sembolünü göstermektedir. Şekil 2-10*b* de isé VEYA ve VEYA DEĞİL kapılarının doğruluk tabloları görülmektedir. VEYA DEĞİL kapısının boole ifadesi $Y=\overline{A+B}$ şeklindedir. Bu

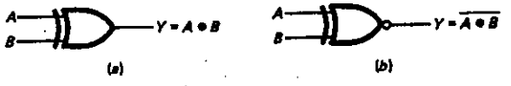


Şekil 2-10 VEYA DEĞİL fonksiyonunun türetilmesi. *(a)* Şematik semboller. *(b)* Doğruluk tabloları

kapı için girişlerden herhangi birisi yüksek (1) ise çıkışın alçak (0) olduğuna dikkat edin.

"ÖZEL VEYA" VE "ÖZEL VEYA DEĞİL" KAPILARI

Temel kapıların sonuncuları ÖZEL VEYA ve ÖZEL VEYA DEĞİL kapılarıdır. Şekil 2-11a ÖZEL VEYA kapısının şematik sembolünü, Şekil 2-11b de ÖZEL VEYA DEĞİL kapısının sembolünü göstermektedir. Şekil 2-11c deki doğruluk tablosundan da görüleceği gibi, bir ÖZEL VEYA kapısının çıkışı sadece iki giriş birbirinin zıddı olduğunda yüksek (1) olmaktadır. ÖZEL VEYA kapısının boole ifadesi $Y = A.\overline{B}+\overline{A}.B = A \oplus B$ şeklinde yazılabilir. Diğer taraftan bir ÖZEL VEYA DEĞİL kapısının çıkışı sadece her iki girişin aynı olduğu hallerde yüksek (1) olmaktadır. ÖZEL VEYA DEĞİL kapısının boole ifadesi $Y=\overline{A}\,\overline{B}+A\overline{B} = \overline{A \oplus B}$ şeklindedir.



A	B	A ⊕ B	Ā ⊕ B
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

(c)

Şekil 2-11 ÖZEL VEYA ve ÖZEL VEYA DEĞİL fonksiyonları. *(a)* ÖZEL VEYA nın şematik sembolü. *(b)* ÖZEL VEYA DEĞİL in şematik sembolü. (c) Doğruluk tabloları.

MANTIK AİLELERİNİN KARAKTERİSTİKLERİ

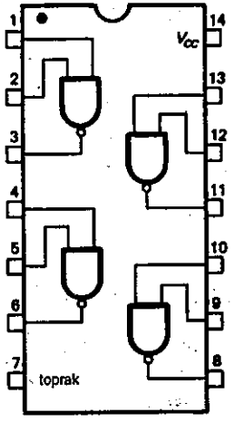
Bu bölümün önceki kısımlarında temel mantık kapılarını tanıttık. Bölüm 3'de ve sonraki bölümlerde bu kapıların nasıl kullanıldığını göreceğiz. Bu bölümün bundan sonraki kısımlarında, piyasada bulunan mantık kapı ailelerinin özelliklerini inceleyeceğiz.

TTL MANTIK KAPISI ÖZELLİKLERİ

Üzerinde duracağımız ilk mantık ailesi TTL *(Transistör-Transistör Lojik)* dir. Bu ailenin standart 7400 serisi ve onun çeşitlemeleri 1960 lı yılların sonlarında ve 1970 li yıllarda sayısal entegre devre pazarında çok büyük bir paya sahipti. TTL ailesi olgunlaşmış bir ailedir, yani birçok farklı fonksiyonlar piyasada bulunmaktadır ve üreticiler yüksek verimle bu yongaları üretmeyi bilmektedirler. Bu nedenle maliyetleri düşüktür. Endüstri entegrasyonun üst düzeylerine eriştikçe (tek bir yongaya daha fazla fonksiyon ve kapılar koyarak), TTL kullanılmaz hale gelmiştir, çünkü her kapı için çok büyük yonga alanı gerekmektedir ve kapı başına çok fazla güç harcanmakta, ısı oluşmaktadır. Bu problemleri yenmek için büyük ölçekli entegre devreler, MOS (metal-oksit yarı iletkeni), CMOS veya I^2L teknolojilerinden yararlanmaktadırlar. Buna rağmen, birçok TTL mantık elemanı ile halen karşılaşılmaktadır ve bu nedenle de TTL mantık ailesini iyice anlamalısınız.

TTL ENTEGRE DEVRE PAKETLERİ VE NUMARALANDIRMA

Genellikle TTL kapıları "dual in-line" (DIP) paketler halinde gelir. Bağlantı bacakları Şekil 2-12 de görüldüğü gibi yanyana iki sıra halindedir. Paketin içerdiği fonksiyonlara göre 14 ile 24 arasında bacağı olabilir. Şekilde 4 tane iki-girişli pozitif mantık VE DEĞİL kapısı içeren 14 bacaklı bir tek DM5400/7400 paketinin bacak diyagramı görülmektedir.



Şekil 2-12 DM5400/7400 dörtlü iki-girişli pozitif mantık VE DEĞİL kapısının açık şeması

TTL ailesinin elemanlarının çoğu AA74XXX şeklindeki bir sayı ile tanınır. AA, elemanı üreten firmanın adına ait kodu gösteren harf veya harflerdir. Örneğin DM National Semiconductor, SN Texas Instruments ve S Signetics firmalarını temsil etmektedir. XXX ise parçanın fonksiyonunu tanıtan iki veya üç basamaklı bir sayıdır. Örneğin, 74502, 4 tane iki-girişli VEYA DEĞİL kapısı içerir. 7430 bir tane 8-girişli VE DEĞİL kapısı, 74193 4-bitlik bir ikilik sayıcı içerir. Var olan fonksiyonların bir listesi için TTL veri kitabına başvurunuz.

Entegre paketinin üzerinde ayrıca üretim tarihini ve üretim grup numarasını belirten sayılar bulunabilir.

BİR TTL VERİ SAYFASI: Tablo 2-1 bir 5400/7400 kapısı için tipik bir veri sayfasını göstermektedir. V_{cc}, entegre devreyi veya paketin tamamını çalıştırmak için gerekli besleme gerilimidir. Giriş ve çıkış gerilimleri paketteki her kapı için mantık seviye sinyalleridir. Herhangi bir veri sayfasında bakılacak ilk parametreler maksimum değerler ve çalışma şartlarıdır. V_{cc} nin mutlak maksimumu olan 7 V, değişken bir güç kaynağı kullanıyorsanız, kaynak gerilimini TTL elemanına bağlamadan önce çalışma gerilimi sahasının ortası olan +5 V a ayarlamanız gerektiği konusunda uyarmaktadır. Kazara uygulanacak 7 Volttan fazla bir gerilim elemanı bozabilecektir.

Çalışma şartları altında, üreticinin, devrenin performansını garanti ettiği besleme gerilimi ve ortam sıcaklık aralıklarını görmekteyiz. 5400 serisi fonksiyon olarak 7400 serisi ile aynıdır, fakat görüldüğü gibi 5400 serisinin daha geniş bir besleme gerilimi ve ortam sıcaklık aralığında çalışması garanti edilmiştir.

TABLO 2-1

DM5400/DM7400, DM5410/DM7410, DM5420/DM7420, DM5400 VERİ SAYFASI

Mutlak Maksimum Değerler		Çalışma Koşulları			
			Min	Mak	Birim
Vcc	7,0V	Besleme gerilimi Vcc			
Giriş Gerilimi	5,5 V	DM54XX	4,5	5,5	V
Saklama Sıcaklık Aralığı	-65 ile + 150°C	DM74XX	4,75	5,25	V
Çıkış yelpazesi	10	Sıcaklık T_A			
Uç Sıcaklığı		DM54XX	-55	+125	°C
(Lehimleme, 10 sn)	300°C	DM74XX	0	70	°C

ELEKTRİKSEL ÖZELLİKLER*

PARAMETRE	KOŞULLAR		MIN	TİPİK	MAK	BİRİM
Giriş diyodu kelepçeleme gerilimi	V_{cc} = 5.0V, T_A=25°C $I_{giriş}$= -12mA				-1,5	V
Mantık 1 giriş gerilimi	V_{cc}=Min		2,0			V
Mantık 0 giriş gerilimi	V_{cc}=Min				0,8	V
Mantık 1 çıkış gerilimi	V_{cc}=Min, $V_{giriş}$=0,8V, $I_{çıkış}$= -400 µA		2,4			V
Mantık 0 çıkış gerilimi	V_{cc}=Min, $V_{giriş}$=2,0V, $I_{çıkış}$= -16 µA				0,4	V
Mantık 1 giriş akımı	V_{cc}=Mak, $V_{giriş}$=2,4 V				40	µA
Mantık 1 giriş akımı	V_{cc}=Mak, $V_{giriş}$=5,5V				1	mA
Mantık 0 giriş akımı	V_{cc}=Mak, $V_{giriş}$=0,4 V				-1,6	mA
Çıkış kısa-devre akımı *	V_{cc}=Mak, $V_{giriş}$=0 V, V_0=0 V	DM74XX DM54XX	-20 -18		-55 -57	mA mA
Besleme akımı mantık 0‡	V_{cc}=Mak, $V_{giriş}$=5,0 V			3	5,1	mA
Besleme akımı mantık 1‡	V_{cc}=Mak, $V_{giriş}$=0 V			1	1,8	mA
Mantık 0a yayılma gecikme zamanı t_{PDO}	V_{cc} = 5,0V, T_A=25°C C = 50 pF			8	15	ns
Mantık 1'e yayılma gecikme zamanı t_{PD1}	V_{cc} = 5,0V, T_A=25°C C =50 pF			13	25	ns

* Tersi belirtilmedikçe, DM54XX için min/mak sınırları -55 ile +125°C sıcaklık aralığında ve DM74XX için 0 ile 70°C aralığında geçerlidir. Bütün tipik değerler Vcc = 5,0 V T_A = 25°C için verilmiştir.

* Hiçbir zaman birden fazla çıkış kısa-devre edilmemelidir

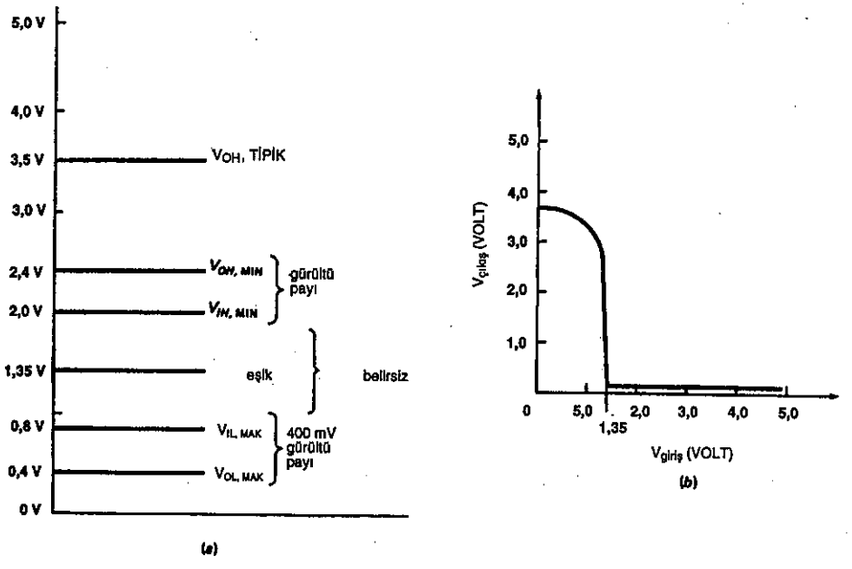
‡ Her kapı için

KAYNAK: National Semiconductor.

Veri sayfasına göre, pozitif yöndeki maksimum sinyal giriş gerilimi + 5,5 volttur. Negatif yöndeki maksimum giriş gerilimi (giriş diyodu kelepçeleme gerilimi) -1,5 volttur. Bu değerleri aşmamayı garanti etmek için, darbe üreteçlerinin ve diğer sinyal kaynaklarının sinyal seviyelerinin 0 ile +5 V arasında olduğunu kontrol ederken mutlaka osiloskopu d.a. kuplaj durumunda kullanınız.

Bundan sonra, veri sayfasının, elektriksel özellikler kısmındaki TTL aygıtının hangi gerilimi mantık 1, yani yüksek ve hangi gerilimi mantık 0, yani alçak olarak kabul ettiğini inceleyin. 2,0 voltluk minimum mantık 1 giriş gerilimi $V_{1H,MIN}$, üreticinin 2,0 volt veya üstünü bu kapının, yüksek olarak tanıyacağını ve kapının çıkış durumunu buna göre belirleyeceğini garanti ettiğini belirtmektedir. 0,8 voltluk maksimum mantık 0 giriş gerilimi $V_{IL,MAK}$, 0,8 volta eşit veya daha az bir giriş geriliminin kapı tarafından alçak olarak tanınacağını göstermektedir. Giriş sinyal gerilimi 0,8 ile 2,0 volt arasında ise çıkış mantık seviyesi belirsizdir, bu nedenle de giriş sinyalleri bu bölgede olmamalıdır veya bu bölgeden hızlı bir şekilde geçmelidir.

Kapının çıkışında verilen 2,4 voltluk minimum mantık çıkış gerilimi $V_{OH,MIN}$, yüksek durumda çıkışın 2,4 volta eşit veya daha büyük olacağını garanti eder. Minimum yüksek çıkış geriliminin, bir sonra gelecek olan kapının gerekli yüksek giriş geriliminden 0,4 V fazla olduğuna dikkat edin. (Şekil 2-13*a* ya bakın). Bu 0,4 voltluk fark gürültü payı olarak adlandırılır. Bağlantı hattındaki küçük bir geçici gürültünün bir sonraki kapının durumunu değiştirmemesini sağlar. Bir sonraki kapının girişini, 2,0 V yüksek giriş minimumunun altına çekebilmek için geçici gürültünün tepe geriliminin 400 mV dan büyük olması gerekir. 400 mV luk benzer bir gürültü payı, 0,4 voltluk maksimum mantık 0 çıkış gerilimi $V_{OL,MAK}$, ile 0,8 voltluk maksimum mantık 0 giriş gerilimi $V_{IL,MAK}$, arasında da vardır. (Şekil 2-13*a* ya bakınız.) Daha sonra göstereceğimiz üzere, aşırı yükleme veya hatalı giriş sinyalleri, bu gürültü paylarının birini veya ikisini de yok edebilir ve yanlış çalışmaya neden olabilir.

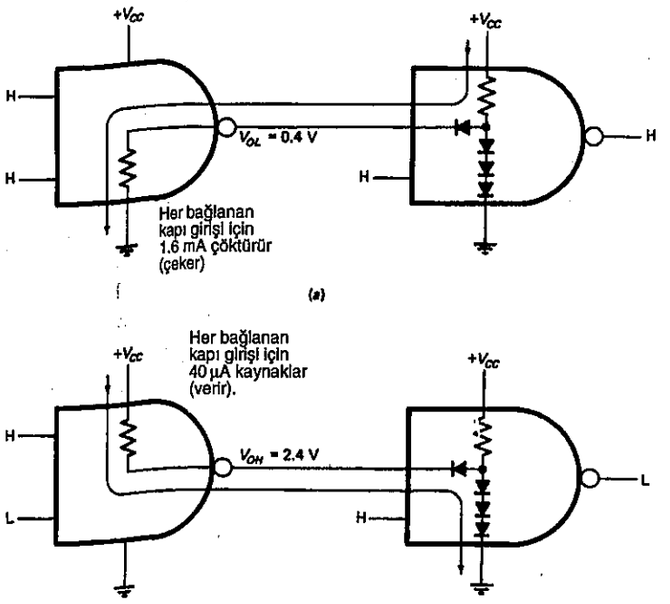


Şekil 2-13 *(a)* TTL gerilim seviyeleri ve gürültü payları. *(b)* Bir TTL ters çeviricinin transfer eğrisi.

Bir mantık kapısının giriş ve çıkış gerilimi seviyelerini göstermenin faydalı bir diğer yolu da transfer eğrisidir. Bu eğri çıkış geriliminin giriş gerilimine göre çizildiği basit bir grafikten ibarettir. Şekil 2-13*b*. bir TTL DEĞİL kapısı için transfer eğrisini göstermektedir. Bölüm 1 deki Şekil 1-13 ise bir TTL kapısı için bu grafiğin osiloskopta nasıl görülebileceğini göstermektedir.

Şekil 2-13*b* de, giriş gerilimi 0,8 voltluk $V_{IL,MAK}$, değerinin üzerine yükseldiğinde, çıkış geriliminin gerekli $V_{OH,MIN}$ olan 2,4 V un altına düştüğünü gözleyin. $V_{çıkış}$ın en hızlı değiştiği grafiğin $V_{giriş}$ bölgesi, eşik gerilimi adını alır ve yaklaşık 1,35 volttur. 2,0 volttan daha az bir $V_{giriş}$ için $V_{çıkış}$ istenilen 0,4 voltluk $V_{OL,MAK}$, değerinin üstüne çıkar. Bu grafikten 0,8 den 2,0 volta kadar olan giriş gerilim aralığının niçin belirsiz olarak nitelendirildiği açıkça görülmektedir. Bu giriş aralığında çıkış seçilmiş olan yüksek veya alçak değerlerinde değildir.

Bir TTL kapısının giriş ve çıkış akımı özelliklerini tartışmadan önce kaynak akımını ve çöktürme akımını tanımlamalıyız. Şekil 2.14, bir TTL kapısının çıkışını onu takibeden bir kapının girişine bağlanmış olarak göstermektedir. Şekilde görülen direnç ve diyotlar tipik TTL kapılarının giriş ve çıkışlarının denk iç devrelerini temsil etmektedir. Eğer birinci kapının çıkışı Şekil 2-14*a* da görüldüğü gibi alçak ise, bu takdirde +5 voltluk V_{cc} kaynağından ikinci kapı üzerinden birinci kapıya doğru bir pozitif akım akacaktır. Bu akım birinci kapıdan da toprağa doğru akacaktır. Bir çıkış, bir sonraki kapının girişinden toprağa veya bir negatif kaynağa bir akım yolu oluşturuyorsa, buna akımı çöktürüyor (çekiyor) denir. O halde Şekil 2-14*a* daki birinci kapının çıkışı ikinci kapının girişini aşağı çekmekle akımı çöktürmektedir. Tablo 2-1, bir TTL kapısı için maksimum mantık 0 giriş akımını -1,6 mA olarak göstermektedir. Bunun anlamı, bir TTL girişinin aşağı çekilmesi için ona bağlı çıkışın 1,6 mA e kadar bir akımı çöktürüp toprağa vermesi demektir. Veri sayfalarında bir akıma ait eksi işareti, kabul edilen akım yönünün belirtilen aygıt bacağından dışarı doğru olduğunu gösterir. Akımın önündeki artı işareti ise akımın aygıt bacağından içeri doğru olduğunu gösterir.



Şekil 2-14 Çıkış çökme ve kaynak akımı. *(a)* Alçak çıkış bir sonraki kapı girişinden maksimum 1,6 mA çöktürür. (çeker). *(b)* Yüksek çıkış bir sonraki kapı girişine maksimum 40 µA kaynaklar (verir).

Şekil 2-14*b*, çıkışı yüksek olan bir kapıyı bir sonraki kapının girişine bağlanmış olarak göstermektedir. Akımın yönü +5 V kaynağından, birinci kapı üzerinden, ikinci kapının girişine doğrudur. Bir çıkış, pozitif kaynaktan bir sonraki kapının girişine doğru bir akım yolu oluşturuyorsa, buna akım kaynaklandırıyor denir. Birinci kapının çıkışı yüksek durumda iken, bir sonraki kapının girişini yükseğe çekmek için gerekli akımı kaynaklandırmaktadır. Tablo 2-1, bir TTL kapısının maksimum mantık 1 giriş akımının ($I_{IH,MAK}$) 40 µA olduğunu göstermektedir. Bu nedenle birinci kapının çıkışı, ikinci kapının girişini yükseğe çekerken 40 µA e kadarlık bir akımı kaynaklandırmak zorunda kalabilir. Daha sonraki bir kısımda, alçak giriş akımının yüksek giriş akımından niçin çok daha fazla olduğunu açıklayacağız.

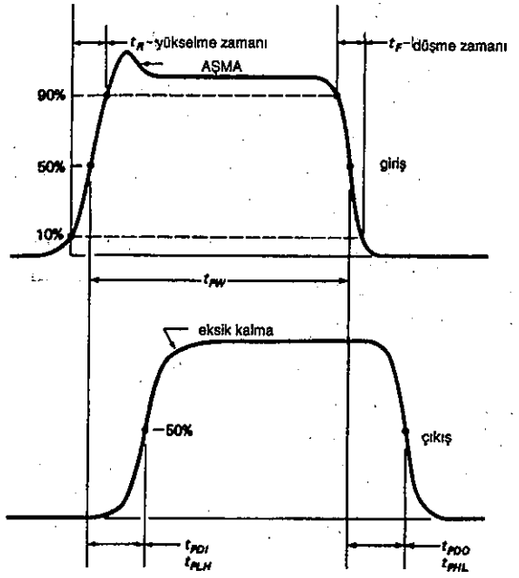
Tablo 2-1 de mantık 1 çıkış gerilimi ve mantık 0 çıkış gerilimini izleyen koşullara bakıldığında, çıkışların yüksek durumda 400 µA ($I_{OH,MIN}$) kaynaklandırdığı, alçak durumda 16 mA ($I_{OL,MIN}$) çöktürdüğü görülecektir.

Çıkış yelpazesi (Fan-out), normal yüksek ve alçak gerilim seviyelerine ulaşmasını engellememek üzere tek bir kapının çıkışına bağlanabilecek maksimum kapı girişleri sayısıdır. Çıkış yelpazesi, I_{OL}/I_{IL} veya I_{OH}/I_{IH} oranlarından hangisi daha küçük değer veriyorsa ona eşittir. Bu örnek kapı için I_{OL}/I_{IL} = 16 ma/1,6ma = 10 ve I_{OH}/I_{IH} = 400 µa/40 µa =10 dur. Bu nedenle standart bir TTL çıkışı 10 taneye kadar TTL girişi sürmek üzere yeterince akım kaynaklandırabilir veya çöktürebilir. Bir çıkışa 10 dan fazla kapı bağlarsanız, kapının çıkış akımını çöktürebilen transistörün doyum gerilimi 10 kapılık yük için üreticinin belirttiği 0,4 voltluk maksimumu aşacaktır. Bu gerilimi artırmak, bütün bağlı girişler için gürültü payını azaltır ve onları geçici gürültülere daha hassas hale getirir. Bir girişin bir direnç üzerinden toprağa bağlandığı devrelerde (Schmitt tetiklemesi veya daha sonra bu bölümde göreceğimiz osilatör gibi), direncin değeri giriş gerilimini 0,4 V civarında veya altında tutacak kadar küçük olmalıdır. 0,4 Volt, 1,6 mA e bölünürse bu direnç için 250 Ω luk bir değer bulunur. Ancak en kötü çalışma şartları ile karşılaşması beklenmeyen devrelerde çoğunlukla kullanılan değer 470Ω dur.

Dikkate alınması gereken, bir sonraki parametre besleme akımı yani her kapıyı çalıştırmak için güç kaynağından çekilecek toplam akımdır. Mantıksal alçak çıkışı üreten bir kapının 3,0 ile 5,1 mA arasında bir akım gerektirdiğine, mantıksal yüksek çıkışı üreten bir kapının 1,0 ile 1,8 mA arasında bir akım gerektirdiğine dikkat edin. Kapı durum değiştirdiğinde gerekli besleme akımındaki değişiklik, V_{cc} hattındaki geçici gürültülerin bir nedenidir. Bir devrenin güç gereksinimini ilk hesaplayışınızda, her entegre için maksimum besleme akım değerini ikiye katlayarak kullanın. Böylece bir güvenlik payı bırakmış olursunuz ve ihtiyaç duyabileceğiniz ek devreler için güç gereksinimine hazır olursunuz.

Son parametre, yayılma gecikmesi, bir aygıtın girişindeki bir değişikliğin çıkışında bir değişiklik oluşturuncaya kadar geçen zamandır. Şekil 2-15, yayılma gecikmesi ve çeşitli önemli darbe parametrelerinin nasıl tanımlandığını göstermektedir. Yayılma gecikmesi zamanı, girişteki geçişin %50 noktasından, buna karşılık gelecek çıkıştaki geçişin %50 noktasına kadar geçen zaman olarak ölçülür. Eğer çıkış alçaktan yükseğe doğru bir geçiş yapıyorsa yayılma gecikmesi zamanı T_{PLH} veya T_{PDI} olarak tanımlanır. Eğer çıkış yüksek durumdan alçağa doğru geçiyorsa, yayılma gecikmesi zamanı T_{PHL} veya T_{PDO} olarak tanımlanır. Bir TTL kapısı için Tablo 2-1 de T_{PDI} ve T_{PDO} zamanlarının farklı olduğuna dikkat edin. Bunun nedeni, bu bölümün ileriki bir kısmında açıklanacaktır.

Şekil 2-15'de diğer gözlenmesi gereken darbe parametreleri yükselme zamanı, düşme zamanı ve darbe genişliğidir. Yükselme zamanı T_R genelde bir darbenin genliğinin %10 undan %90 ına kadar ulaşması için geçen zaman ola-



Şekil 2-15 Darbe parametreleri

rak tanımlanır. Düşme zamanı T_F ise bir darbenin %90 ından %10 una ulaşıncaya kadar geçen zamandır. Nadiren %20 ve %80 noktaları da T_R ve T_F için kullanılmaktadır. Yükselme zamanına alçaktan yükseğe geçiş zamanı T_{TLH} ve düşme zamanına da yüksekten alçağa geçiş zamanı T_{THL} de denir. Bir darbenin genişliği, T_{PW}, genellikle darbe geçişlerinin %50 noktaları arasındaki zaman olarak belirtilir.

TTL KAPILARIN ÖZELLİKLERİNİN ÖZETİ Standart 7400 serisi TTL kapısı 4,75 ile 5,25 V arasında bir besleme gerilimi gerektirir ve bir kaç miliamperlik kaynak akımına gereksinim duyar.

Bir TTL girişini aşağıya çekmek için, 0,0 ile +0,8 V arasında bir gerilim uygulamanız gerekir. Girişi aşağı çekmek için kullanılan aygıt 1,6 mA e kadar bir akımı çöktürebilmelidir.

Bir TTL girişini yukarı çekmek için, +2,0 ile +5 V arasında bir gerilim uygulamanız gerekir. Süren aygıt veya devre 40 µA e kadar bir akımı kaynaklandırabilmelidir.

Standart bir TTL kapısının çıkışı doğru gerilim seviyelerinde 10 standart TTL kapısının girişini sürebilecek kadar akım kaynaklandırabilir veya çöktürebilir. Bunu belirtmek için kapının çıkış yelpazesi 10 dur denir.

Garanti edilen 2,4 V luk yüksek çıkış gerilimi 2,0 V luk yüksek giriş geriliminden 0,4 V büyüktür. Bu, sürücü bir kapının çıkışı ile alıcı bir kapının girişi arasında 0,4 V luk bir gürültü payı oluşturur. Aynı gürültü payı 0,4 V luk alçak çıkış maksimumu ile 0,8 V luk maksimum alçak giriş gerilimi arasında da vardır.

Bir TTL kapısının girişindeki bir değişikliğin çıkışta bir değişikliğe neden olması 10 ile 25 ns alır. Bu süre, yayılma gecikmesi adını alır.

TTL İÇ DEVRESİ:

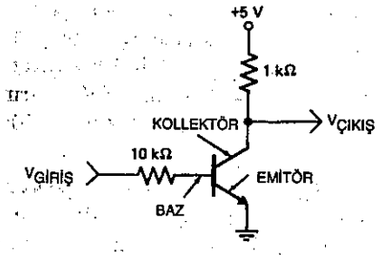
ÇALIŞMA TEORİSİ

Önceki kısımda bir standart TTL kapısının devresini dikkate almadan özelikleri üzerinde durmuştuk. Bu kısımda bir standart TTL kapısının çalışmasını tartışacağız ve daha önce tanımladığımız parametrelerin kaynağını açıklayacağız.

BİR ANAHTAR VE TERS ÇEVİRİCİ OLARAK BİPOLAR TRANSİSTOR

Mantık kapıları aslında elektrik anahtarlarıdır. Entegre devre mantık kapıları bu anahtarları gerçekleştirmek için bipolar veya *MOS* transistörleri kullanır. Şekil 2-16 bir ters çevirici olarak çalışan bir *NPN* bipolar transistor anahtar devresini göstermektedir. Girişe +5 voltluk yüksek mantık gerilimi uygulanırsa, transistörün baz-emitör kavşağı ileri kutuplanacaktır ve transistör iletime geçecektir. Baz yaklaşık 0,7 V ta olacaktır. 10 KΩ luk baz direnci üzerinde 5-0,7 = 4,3 V luk gerilim kalır. Böylece bazdan içeri 4,3 V/10KΩ = 0,43 mA lik bir akım akar. Bu baz akımı, transistörün kollektöründen emitörüne çok daha büyük bir akım akmasına sebep olur. Kollektör akımının baz akımına oranına beta *(β)* denir ve tipik olarak 100 civarındadır. Bunun anlamı kollektör akımının baz akımının 100 katı olacağıdır. Gösterilen devre için, 0,43 mA lik baz akımı teorik olarak 43 mA lik kollektör akımı oluşturacaktır. Ancak 43 mA lik bir kollektör akımının 1 KΩ luk kollektör direnci üzerinde 43 voltluk bir gerilim düşümü oluşturması gerekir (43 mA x 1 KΩ = 43 V). 5 V luk kaynakla bunun mümkün olmadığı gayet açıktır. Gerçekte olan olay, kollektör direnci üzerindeki gerilim düşümü, kollektör gerilimini emitör geriliminin onda birkaç volt üstüne düşürünceye kadar kollektör akımının artması şeklindedir. Verilen örnekte kollektör gerilimi 0,1 V kadar (mantıksal düşük) olacaktır. Bu

durumda transistöre doymuş denir. Transistörün baz bölgesi yüklerle doymuş durumdadır, çünkü kollektörde var olan bütün yükleri çekecek kadar gerilim yoktur. Doymuş terimi buradan kaynaklanmaktadır. Buradaki önemli nokta doymuş bir bipolar transistörün kollektörü ile emitörü arasında onda birkaç voltluk bir gerilim olmasıdır. O halde, Şekil 2-16 daki devrede girişteki mantık yüksek, çıkışta mantık alçak oluşturur.



Şekil 2-16 *NPN* bipolar transistör anahtar/ters çevirici

Eğer Şekil 2-16 daki devrenin girişi toprağa bağlanırsa (mantık düşük), baz-emitör kavşağında onu ileri kutuplamak için gerekli 0,7 V olmayacaktır. Böylece transistör kesimde olacaktır. Baz akımı olmayınca kollektör direnci üzerinde gerilim düşümü olmayacaktır. Kesimdeki bir transistörün kollektörü, hangi gerilim uygulanmışsa ona kayar. O halde kollektör 5 voltta (mantık yüksek) olacaktır. Girişteki mantık alçak, transistörü kesime götürür ve çıkışta mantık yüksek oluşturur. Devre bir ters çevirici gibi davranmaktadır. Bu devredeki transistör ya kesimdedir ya da doyumdadır.

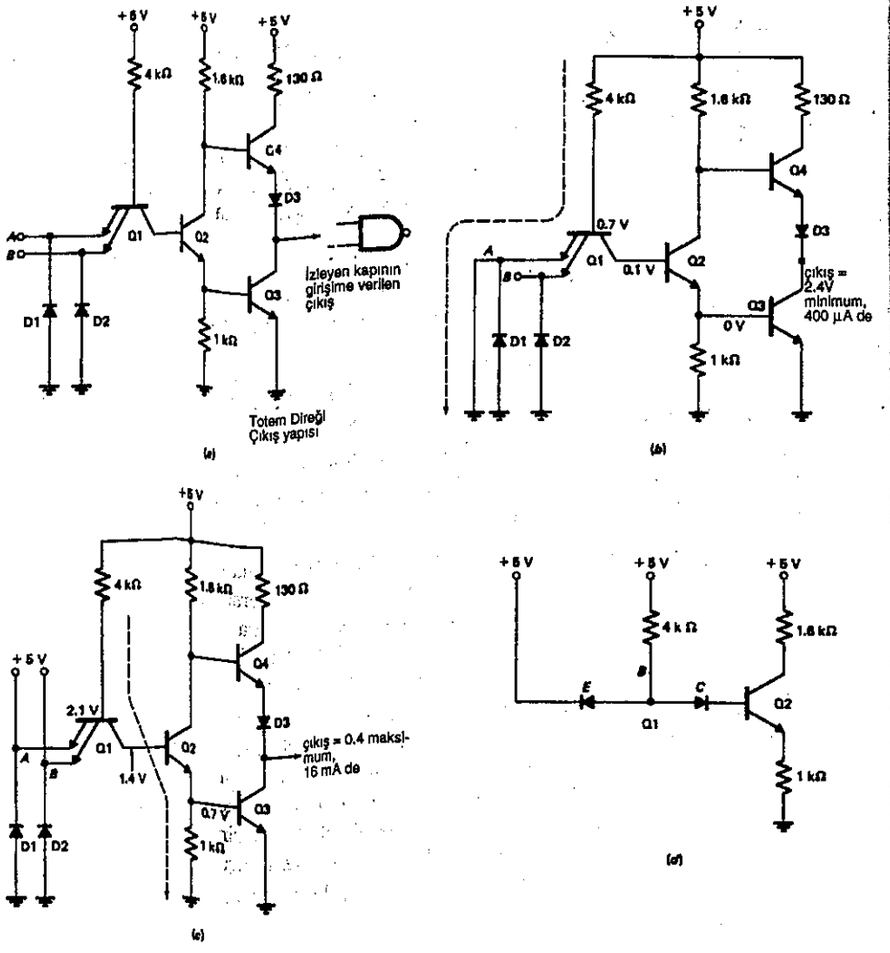
TTL VE DEĞİL KAPISININ DEVRESİ Şekil 2-17 bir standart *TTL* VE DEĞİL kapısının iç devresini göstermektedir. Bu devrenin çalışmasının analizini yapabilmek üzere B girişinin hiç bir etkisinin olmaması için +5 V a bağlandığını kabul edin. Ayrıca Şekil 2-17b de görüldüğü gibi *A* girişinin topraklandığını yani mantık sıfır durumunda olduğunu kabul edin. A girişini topraklamak baz akımının *Q1* den akması için biryol sağlar ve *Q1* iletime geçer. *Q1* ın bazı toprağın V_{BE} kadar (yaklaşık +0,7 V) üzerindedir. Bu 4 KΩ luk direnç üzerinde 4,3 V kalmasını sağlar. Ohm kanununu kullanarak baz akımını yaklaşık 1,1 mA olarak hesaplayabilirsiniz. *Q1* in kollektör akımı için kaynağı olmadığından 1,1 mA lik baz akımı *Q1* i doyuma çeker ve kollektör gerilimi emitör geriliminin yaklaşık 0,1 V üstünde olur. Emitör topraklanmış olduğundan *Q1* in kollektörü ve *Q2* nin bazı yaklaşık 0,1 V tadır. Bu *Q2* yi iletime geçirmek için gerekli 0,7 V tan çok daha azdır, bu nedenle *Q2* kesimdedir.

Q2 den akım akmayınca 1 KΩ luk direnç üzerinde gerilim düşümü olmaz ve *Q3* ün bazı toprak gerilimindedir. Bu nedenle *Q3* de kesimdedir ve çıkış üzerinde etkisi yoktur, çünkü kesimdeki bir transistörün kollektörü bir açık anahtar gibi davranır.

Q2 nin emitöründen çıkışa kadar olan yolu izlemiş olduk, şimdi de *Q2* nin kollektöründen çıkışa giden yolu izleyelim. *Q2* kesimde olduğuna göre kollektörünün *Q4* ün bazı üzerinde etkisi yoktur. +5 V a bağlı 1,6 KΩ luk direnç üzerinden Q4 ün sağlamak zorunda olduğu maksimum çıkış yük akımı yaklaşık 10 kapı girişini sürmek için gerekli 0,4 ma yani 400μA dir. Bu çıkışı üretmek için gerekli çok küçük baz akımı 1,6 KΩ luk direnç üzerinde çok küçük bir gerilim düşmesi oluşturur. *Q4* ün bazı 5V civarındadır. Böylece çıkış bu gerilimin V_{BE} (0,7 V) ile diyot üzerindeki 0,7 V luk düşme toplamı kadar altındadır, yani yaklaşık 3,6 V tur. Standart *TTL* için 2,4 V tan büyük bir çıkış gerilimi nizami mantık yüksek seviyesidir. Böylece görüyoruz ki *A* girişindeki bir alçak giriş veya topraklama, çıkışı yüksek yapar. Fonksiyon olarak *B* girişinin *A* ile aynı olduğunu devredeki akım ve gerilimleri izleyerek gösterebilirsiniz. O halde, eğer *A* alçak ise, veya eğer *B* alçak ise veya hem *A* hem de *B* alçak ise bu TTL VE DEĞİL kapısının çıkışı yüksektir.

Şimdi de *A* girişini yüksek yapmanın etkisini görelim (Şekil 2-17c ye bakınız). Yine *B* girişinin yüksek +5 V a bağlandığını kabul edin. *A* girişi de pozitif mantık yüksekte ise, yani +5 V a bağlandığında, *Q1* kesimdedir. Ancak NPN transistörüne bir ohmmetrenin gördüğü gibi bakarsak (Şekil 2-17*d*) *Q2* nin bazına, 4 KΩ luk direnç ve *Q1* in kollektör-baz diyodu üzerinden bir akım yolu olduğunu görürüz. Bu yol *Q2* ye onu doyuracak kadar baz akımı sağlar. *Q2* de *Q3* ü sürerek doyuma götürür. Çıkıştaki gerilim, doymuş bir transistörün uçları arasındaki, onda birkaç volttan ibarettir. Bu da mantık alçaktır.

Şimdi geri gidip diğer yoldan *A* nın *Q4* ü nasıl etkilediğini görelim. *Q2* nin emitörü *Q3* ün V_{BE} si tarafından belirlenen 0,7 V tadır. *Q2* doymuş olduğundan kollektör gerilimi bunun 0,1 V üzerindedir yani yaklaşık olarak toprağa göre 0,8 V tur. Bu, *Q4* baz-emitör kavşağı ve *D3* diyodu seri birleşimini iletime geçirmek için gerekli 1,4 V tan oldukça azdır. Bu nedenle *Q4* kesimdedir ve alçak çıkış üzerinde etkisi olmaz. Gerçekte *D3* ün amacı *Q4* ün Q3 le aynı zamanda iletimde olmasını mümkün olduğunca engellemektedir. Eğer *Q3* ve *Q4* uzun süre birlikte iletimde olurlarsa, V_{cc} den doğrudan toprağa, yongayı ısıtıp güç harcayan, büyük bir akım akacaktır. *D3* ün kullanıldığı bu totem-direği çıkış yapısı ile bile, çıkışın her durum değiştirişinde her iki transistörün de iletimde olduğu kısa bir an vardır. Bu, iletimde olan transistörler üzerinden toprağa kısa bir aşırı akım geçmesine müsade etmiş olur. Bu da V_{cc} hattı üzerindeki gerilim gürültülerini doğurur. Bölüm 1 de açıklandığı gibi yeterli yangeçit kapasitörünüz yoksa, bu gürültüler diğer devreleri rahatsız edebilir.



Şekil 2-17 *(a)* **Totem-direği çıkış yapısına sahip standart TTL VE DEĞİL kapısının devresi.** *(b)* **A girişi topraklandığında gerilimler.** *(c)* **A girişi yüksekken oluşan gerilimler.** *(d)* ***Q1*** **in ohmmetre tarafından görülüş şekli.**

Şekil 2-17 de görülen *D1* ve *D2* giriş kelepçe diyotları bu bölümün daha sonraki giriş sinyal uyumlaması kısmında açıklanmaktadır.

TTL VE DEĞİL KAPISI DEVRESİNİN ÇALIŞMASININ ÖZETİ

Eğer girişlerden herhangi biri alçaksa *Q1* transistör olarak iletimdedir, *Q2* ve *Q3* kesimdedir ve çıkış *Q4* tarafından yükseğe çekilir. Bütün girişler yüksekse *Q1* in kollektör-baz kavşağı bir diyot gibi işlev görür. Bu, çıkışı mantık alçağa çekmek üzere *Q2* ve *Q3* ü iletime geçirir. Sonuç olarak bu olayların VE DEĞİL kapısının boole ifadesi $Y=\overline{A.B}$ eşitliğini sağladığını görürüz.

TTL KULLANARAK DEVRE KURARKEN HATIRLANMASI GEREKEN ÖNEMLİ NOKTALAR

Açık bırakılan bir *TTL* girişi yüksek gibi davranır, çünkü *Q1* için bir baz-emitör akım yolu yoktur. Kullanılmayan *TTL* VE ve VE DEĞİL kapılarının girişleri 1KΩ luk bir direnç üzerinden V_{cc} ye bağlanmalıdır. 1KΩ luk direnç, kapı girişini V_{cc} hattı üzerindeki yüksek gerilim sıçramalarından korur. Kullanılmayan VEYA ve VEYA DEĞİL kapılarının girişleri doğrudan toprağa bağlanabilir.

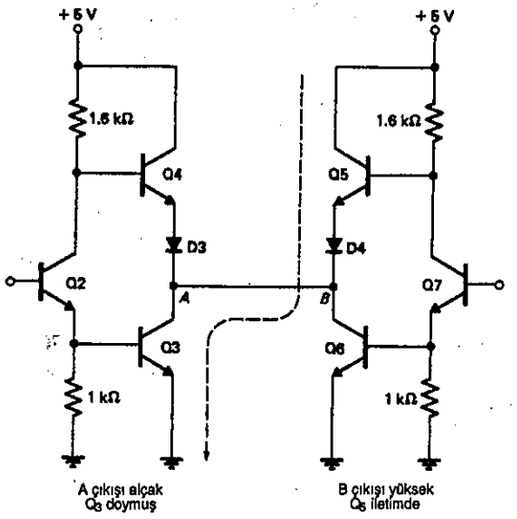
Standart *TTL* çıkışları hiçbir zaman beraber bağlanmamalıdır. İki standart totem-direği çıkış yapısını birbirine bağlamayı denerseniz (Şekil 2-18), eğer bir çıkış yükseğe gitmeye çalışırsa ve diğer çıkış alçağa gitmeye çalışırsa çıkışlar arasında ölümcül bir savaş başlayacaktır. Bunun *Q3*, *D4* ve *Q5* üzerinden V_{cc} den toprağa bir alçak empedans yolu yaratacağını şekil 2-18 de görebilirsiniz. Bazı üreticiler kısa devre akımını sınırlamak için *Q4* ve *Q5* in kollektörleriyle seri 130 Ω luk dirençler koymaktadırlar. Bu dirençler, eğer yongadaki kapılardan sadece bir tanesinin çıkışı kaza ile toprağa kısa devre edilmişse yongayı koruyabilir. Dirençlerin orada olduğuna her zaman güvenemezsiniz, bu nedenle de totem-direği çıkışları asla birbirine bağlamayın. Kapıların durum değiştirmeleri sırasında oluşan V_{cc} deki gürültüleri filitrelemek için 0,01-0,1 µF lık yangeçit kapasitörleri V_{cc} ile toprak arasında bağlanmalıdır.

Şimdi bir *TTL* kapısının iç devresinin çalışma teorileriyle tanışık olduğumuza göre T_{PLH} ve T_{PHL} yayılma gecikme süreleri arasında daha önce bahsedilen fark artık açıklanabilir. Bir alçak çıkış sırasında *Q2* ve *Q3* doyumdadır (Şekil 2-17c ye bakınız). Bu transistörler kesime götürülüp çıkışın durumunu değiştirmeden önce doyum sırasında baz bölgesinde birikmiş olan yüklerin boşaltılması gerekir. Bunun için gerekli zamana **saklama zamanı** denir. Transistör iletime geçerken saklama zamanı problemi olmadığı için *Q3* ve

Q4 kesimine geçmeye göre daha hızlı iletime geçer. Böylece yayılma gecikme zamanları arasında farklılık olur. Bu farklılık *TTL* ailesinin çoğu üyesi için tipiktir ve hızlı bir sayısal-analog çeviricinin girişlerinde olduğu gibi, devrelerde zamanlama problemlerine sebep olabilir.

Eğer bir çıkışa 10 kapı girişinden fazla bağlarsanız, daha önce bahsedildiği gibi sadece gürültü payını azaltmazsınız, aynı zamanda çıkış değişiklikleri sırasında şarj ve deşarj edilmesi gereken kapasitansı artırarak yayılma gecikme zamanlarını artırırsınız. Bu bir *TTL* kapısının çıkış yelpazesi sınırını niçin aşmamanız gerektiğini açıklamaktadır.

AÇIK-KOLLEKTÖRLÜ *TTL* Bazı standart *TTL* devreleri gördüğümüz totem-direği çıkışı yerine açık-kollektörlü çıkış yapısına sahiptir. Açık-kol-



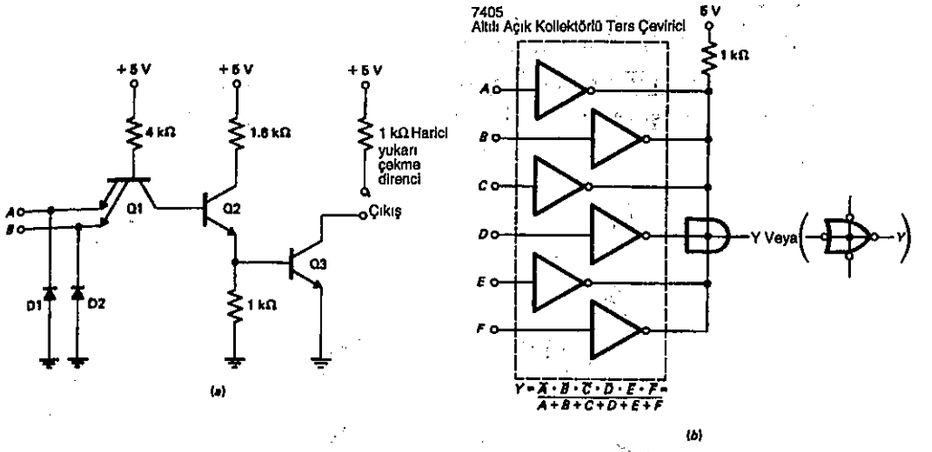
Şekil 2-18 İki totem-direği çıkışı birbirine bağlamakla oluşturulan alçak empedans yolu.

lektör devresinde üst transistör ve diyot yoktur, kollektör bir harici dirençle V_{cc} ye bağlanır (Şekil 2-19 ya bakınız). Alttaki transistör *(Q3)* kesimde iken bu harici yukarı-çekme direnci çıkışı yüksek yapar.

Açık-kollektör devresinin avantajı, başka bir kapı kullanmaya gerek kalmadan, Şekil 2-19b de görüldüğü gibi birçok çıkış birlikte bağlanarak

bağlanmış VE fonksiyonunun oluşturulabilmesidir. Bu devre için yalnız bütün girişler alçak ise çıkış yüksek olacaktır. Bu bağlanmış fonksiyonu gösteren iki alternatif sembol olduğuna dikkat edin.

Açık-kollektör mantığının dezavantajları ise standart totem-direği çıkışına göre daha gürültülü ve yavaş olması ve arıza bulmanın zor olmasıdır.



Şekil 2-19 (a) Açık-kollektör çıkışlı standart TTL VE DEĞİL kapısının devresi. (b) 7405 açık kolllektörlü ters çeviricilerden yapılmış altı-girişli VEYA DEĞİL kapısı.

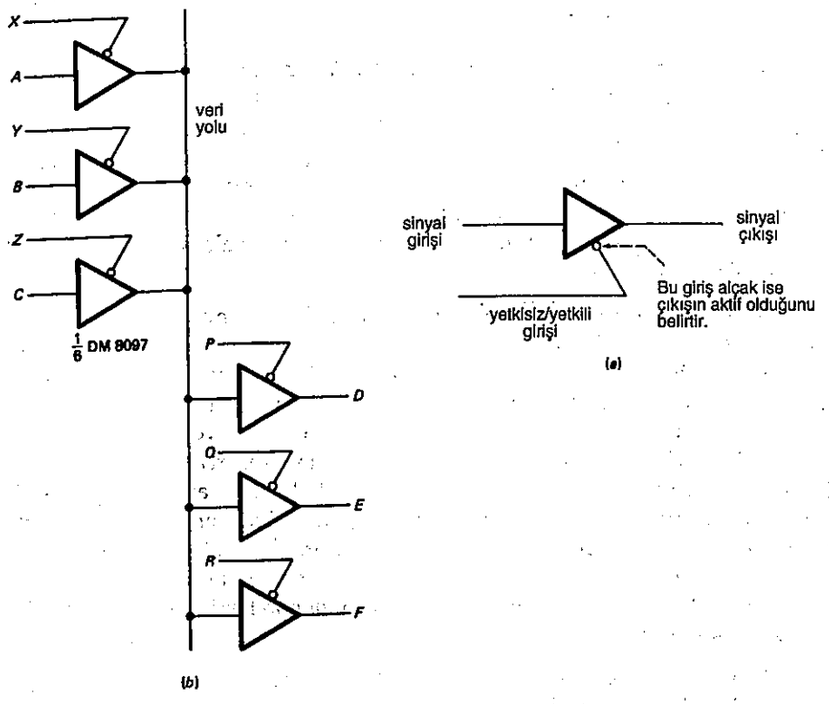
Eğer çıkışlardan herhangi birisi içten alçağa kısa devre olmuşsa, suçluyu bulmanın yolları ya yongaları tek tek değiştirmek ya da bir akım izleme probu ile akımı izleyerek çıkışa ulaşmaktır.

ÜÇ-DURUMLU ÇIKIŞ MANTIĞI Üç-durumlu çıkış mantığı çoğunlukla açık-kollektör mantığının yerini almıştır. Üç durumlu mantık, normal *TTL* alçak ve normal *TTL* yüksek çıkış durumlarına, bir de her iki çıkış transistörünün ayrı bir yetki girişi ile kesime çekildiği bir yüksek-empedans, kayan-çıkış durumuna sahiptir (Şekil 2-20*a* ya bakınız). Yetki girişine uygulanan bir dış sinyal çıkışın aktif veya kayan (farketmez)durumunda olmasını sağlar. Birçok üç-durumlu çıkış bir ortak taşıt hattına bağlanabilir, fakat her-

hangi bir anda yalnız bir çıkış yetkilendirilebilir. Hangi çıkışı öğrenmek istiyorsanız, o çıkışa ait kapıyı yetkilendirirsiniz. Şekil 2-20*b* de görüldüğü gibi üç-durumlu kapılarla, veriler istenilen şekilde seçilerek yola yüklenir veya yoldan alınır. Eğer *A* kapısı yetkilendirilmişse, yol üzerindeki veriyi *A* kapısının çıkışı belirleyecektir ve veri *D,E* veya *F* kapılarından hangisi yetkilendirilirse onun çıkışına verilecektir.

TTL ALT AİLELERİ

Halen bu mantık ailesinin önemli parametrelerini görmüş ve standart 5400/7400 TTL ailesinin analizini yapmış olduğumuza göre, şimdi diğer kullanılan mantık ailelerini tanıtıp karşılaştıralım. Tablo 2-2 bu parametreleri ve karşılaştırmaları özetlemektedir.



Şekil 2-20 ***(a)*** **Üç-durum çıkışlı tampon.** ***(b)*** **Seçime bağlı olarak bir yola veri koyan veya yoldan veri yollayan üç-durum çıkışlı tamponlar.**

Mantık ailelerinin verimlerini karşılaştırmada sık kullanılan bir terim **hız-güç çarpımıdır.** Hız-güç çarpımı ne kadar düşükse ailenin enerji verimi o kadar yüksektir. Üreticiler hız-güç çarpımı için çoğunlukla pikojul birimini kullanırlarsa da siz Tablo 2-2 deki verilerden yararlanarak ve basit olarak kapı gecikmesi ile kapı başına olan güç miktarını çarparak çeşitli mantık ailelerinin hız-güç çarpımlarını karşılaştırabilirsiniz.

İzleyen kısımlarda, ilk üzerinde durulan aileler standart *TTL* devresinin değişikliğe uğratılmasıyla türetilenlerdir.

YÜKSEK-GÜÇLÜ VE ALÇAK-GÜÇLÜ TTL Eğer Şekil 2-17 deki temel devrede bütün direnç değerleri azaltılırsa, daha fazla akım geçer ve devre iki misli daha hızlı anahtarlanır. Bu değişiklik yüksek-hızlı veya yüksek-güçlü *TTL* diye bilinen 74H00/54H00 ailesinin temelini oluşturur. Tablo 2-2 den görebileceğiniz gibi bu ailedeki aygıtlar normal *TTL* den daha kısa yayılma gecikmesine sahiptir, fakat kapı başına daha fazla güç kullanırlar. 74H00 ailesinin genelde modası geçmiştir ve çoğunlukla yerlerini daha sonra göreceğiniz 74S00 Schottky *TTL* ailesi almıştır.

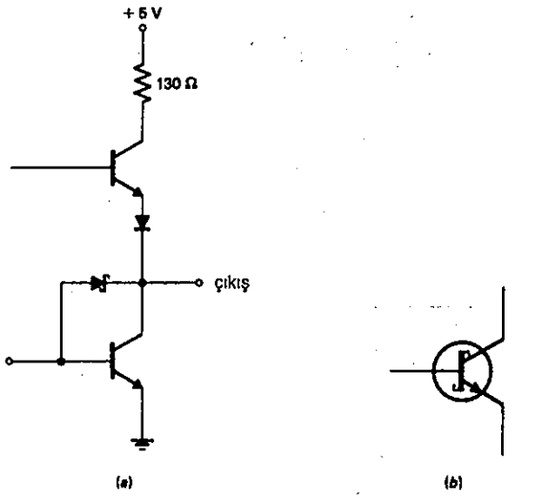
Kısa yayılma gecikmesi veya standart *TTL* nin yüksek hızı gerekmeyen ve daha az güç harcaması istenen uygulamalar için 74L00/54L00 alçak-güçlü *TTL* ailesi geliştirilmiştir. Şekil 2-17 deki temel devrenin bütün dirençlerinin değerlerini artırmakla güç 10 katı, hız da 3 katı küçülür ve böylece alçak-güçlü *TTL* ailesi elde edilir. Tablo 2-2 den belirleyebileceğimiz gibi, her 74L00 alçak-güçlü *TTL* çıkışı en fazla iki tane standart *TTL* girişi veya 10 tane alçak güçlü girişi sürebilir. Çoğu kullanımlarda *CMOS* mantık veya alçak güçlü Schottky *TTL* kapıları, alçak-güçlü *TTL* in yerini almıştır.

SCHOTTKY *TTL* Entegre devre tasarımcıları, süregelen hız arayışlarında, standart *TTL* kapısındaki doymuş transistörlerin saklama zamanı probleminin üstünden gelmenin bir yolunu aradılar. Çözüm metal- yarıiletken ekleminden yapılmış özel bir diyot şeklinde geldi. Schottky diyodu adı verilen bu diyodun 0,3 V luk bir ileri gerilim düşümü vardır ve saklama zamanı problemi olmadığından çok hızlı bir şekilde iletim ve kesim arasında değişebilir. Şekil 2-21*a* ya bakın. Bir transistörün baz ve kollektörü arasına bağlanmış bir Schottky diyodu, transistörün kollektör gerilimi baz geriliminin yaklaşık 0,3 V altına düşünce iletime geçer. Bu, kollektör gerilimini 0,4 V ta sabitler ve transistörün doyuma gitmesini engeller. Transistör doyumda olmadığından ve Schottky diyodunun saklama zamanı olmadığından her ikisi de çok hızlı bir şekilde kesime çekilebilir.

Entegre devrelerde Schottky diyodu transistörün bir parçası olarak yapılır. Şekil 2-21*b* Schottky diyot-kelepçeli transistörün şematik sembolünü göstermektedir. Entegre devrelerdeki transistörlerin çoğunu Schottky diyot-kelepçeli yaparak ve bir parça farklı bir devre kullanarak 74S00/54S00

TABLO 2-2
KULLANILAN MANTIK AİLELERİNİN KARAKTERİSTİKLERİNİN KARŞILAŞTIRILMASI

MANTIK AİLESİ	ÇALIŞMA BESLEME GERİLİMİ MIN	ÇALIŞMA BESLEME GERİLİMİ MAK	MİNİMUM MANTIK 1 GİRİŞ GERİLİMİ	MAKSİMUM MANTIK 0 GİRİŞ GERİLİMİ	MİNİMUM MANTIK 1 ÇIKIŞ GERİLİMİ	MAKSİMUM MANTIK 0 ÇIKIŞ GERİLİMİ	MAKSİMUM MANTIK 0 GİRİŞ AKIMI	MAKSİMUM MANTIK 1 GİRİŞ AKIMI	MİNİMUM MANTIK 0 ÇIKIŞ AKIMI	MİNİMUM MANTIK 1 ÇIKIŞ AKIMI	TİPİK YAYILMA GECİKMESİ C_L = 50 pF T_{PHL}	TİPİK YAYILMA GECİKMESİ C_L = 50 pF T_{PLH}	MAKSİMUM SAYICI FREKANSI	YELPAZE ÇIKIŞI	HER KAPI İÇİN TİPİK GÜÇ HARCAMASI	TİPİK t_{PD} @ 10–50%	KULLANILAN AYGIT
7400	4,75	5,25	2,0	0,8	2,4	0,4	1,6 mA	40 µA	16 mA	400 µA	8 ns	13 ns	35 MHz	10	10 mW	15 ns	7400
5400	4,5	5,5															5400
74H	4,75	5,25	2,0	0,8	2,4	0,4	2,0 mA	50 µA	20 mA	500 µA	6,2 ns	5,9 ns	50 MHz	10-74H	22 mW	7 ns	74H00
74L	4,75	5,25	2,0	0,8	2,4	0,4	0,18 mA	10 µA	3,6 mA	200 µA	30 ns	60 ns	3 MHz	10-74L	1 mW	30 ns	74L00
74S	4,75	5,25	2,0	0,8	2,7	0,5	2,0 mA	50 µA	20 mA	1 mA	5 ns	5 ns	125 MHz	10-74S	19 mW	3 ns	74S00
74LS	4,75	5,25	2,0	0,8	2,7	0,4	0,4 mA	20 µA	8 mA	400 µA	8 ns	8 ns	45 MHz	20-74LS	2 mW	15 ns	74LS00
74ALS	4,5	5,5	2,0	0,8	2,5	0,4	0,2 mA	20 µA	10 mA	400 µA	5 ns	4 ns	32 MHz 74ALS192	6-74 20-74ALS	1 mW		74ALS00
PECL III veya MECL III	–4,7	–5,7		–1,485	–0,960	–1,600	MIN 350 µA	350 µA	40 mA	40 mA	1 ns	1 ns	1000 MHz	DC → 63 MECL III	60 mW	1 ns	MC1692
MECL 10,000	–4,2	–5,7	–1,105	–1,475	–0,960	–1,630	MIN 0,5 µA	265 µA	50 mA	50 mA	2 ns	2 ns	200 MHz	DC → 82 MECL	25 mW	3,5 ns	MC10101
FL	1	15				0,4	enjektör akımı	enjektör akımı	20 mA	V_{CC}'ye direnç bağlı değilken sıfır	ayarlanabilir 25–250 ns			Enjektör akımına bağlıdır	5 mW–70 µW	–	–
ALÇAK EŞİTLİ PMOS	V_{CC} = +5 V ± 5% V_{DD} = –9 V ± 5% V_{SS} = V_{GG}		3,0	0,65	3,5	Vcc–6V MOS veya 0,45	1 µA	1 µA	1,6 mA @ 0,45 V	0,2 mA @ 3,5 V	—	—	1 MHz	—	—	–	1702A ROM
+12, +5, –5 NMOS	V_{CC} = +5 V_{DD} = +12 V_{GG} = –5		3,3	0,8	3,7	0,45	±10 µA	±10 µA	1,9 mA	150 µA	—	—	—	—	—	–	8080A
NMOS yalnız +5	4,5	5,5	2,0	0,8	2,4	0,4	10 µA	10 µA	2,1 mA	100 µA	—	—	—	1-74 5-74LS	—	–	2102 RAM
4000 A +5 V CMOS	3	15	⅓V_{CC}	⅓V_{CC}	V_{CC} – .01	0,01	10 pA	10 pA	.12 mA @ 4,5 V	.12 mA @ 4,5 V	50 pF 180 ns	50 pF 125 ns	1 MHz CD4040	AC = 3-10 CMOS	Çalışma frekansına bağlıdır.	175 ns	CD4011
4000B +5 V CMOS	3	18	⅓V_{CC}	⅓V_{CC}	V_{CC} – .01	0,01	±1 µA	±1 µA	0,4 mA @ 0,4 V	1,6 mA @ 2,5 V	80 pF 180–320 ns	50 pF 210–420 ns	2,5 MHz 4520	DC = = AC = 3-10 CMOS		100 ns	CD4081B
4000B +15 V CMOS	3	18	⅓V_{CC}	⅓V_{CC}	V_{CC} – .01	0,01	±1 µA	±1 µA	3 mA @ 1,5 V	3 mA @ 13,5 V	50 ns	65 ns	6 MHz 10 V 4520	3-10 CMOS		40 ns	CD4081B
74C00 +5 V CMOS	3	15	3,5	1,5	4,5	0,5	5 nA	5 nA	0,4 mA @ 0,4 V	0,36 mA @ 2,4 V	50 pF 80 ns	50 pF 80 ns	2 MHz 74C90	3-10 CMOS 2-74L00		100 ns	74C00



Şekil 2-21 ***(a)*** **TTL çıkış transistör kollektörünün doyuma ulaşmasını engelleyen Schottky kelepçe diyodu.** ***(b)*** **Schottky diyot-kelepçeli transistörün şematik sembolü.**

Schottky *TTL* ailesi yaratılmıştır. Bu ailenin üyeleri, yüksek-güçlü *TTL* den en az 2 defa ve normal *TTL* den 4 defa daha hızlıdırlar. Ancak, Schottky *TTL* nin hızlı anahtarlaması ve kenarların kısa yükselme zamanları, kapılar arasındaki bağlantılar kısa tutulmazsa ve güç kaynağına iyi bir yangeçit yapılmazsa, problemlere sebep olabilir. Bunu, daha sonraki iletim hatları kısmında daha derinlemesine görüşüceğiz.

ALÇAK-GÜÇLÜ SCHOTTKY Schottky *TTL* kapı devrelerindeki dirençlerin değerlerinin artırılması onları alçak-güçlü Schottky *TTL* ye dönüştürür. Bu aile en az yayılma gecikmesi, hızlı anahtarlama yapma ve alçak güç için optimize edilmiştir. 74S00/54LS00 ailesi standart *TTL* den çok az daha hızlıdır, fakat onların gücünün sadece beşte birini gerektirir. Böylece hız-güç çarpımı standart *TTL* den 5 defa daha iyidir (düşük). Texas Instruments'ın 74ALS serisi gibi alçak-güçlü Schottky'lerin geliştirilmiş türleri tipik t_{PD} yi 8 ns den 4 ns ye ve güç harcamasını 2 mW dan 1mW a düşürmüştür. Bu daha da iyi bir hız-güç çarpımı vermektedir. Hız ve güç avantajları dolayısıyla 74LS ve 74ALS aileleri standart 7400 serisi *TTL* lerin hızla yerini almaktadır.

Alçak-güçlü Schottky'lerin diğer bir avantajı alçaktan-yükseğe ve yüksekten-alçağa geçişlerdeki yayılma zamanlarının birbirlerine çok daha

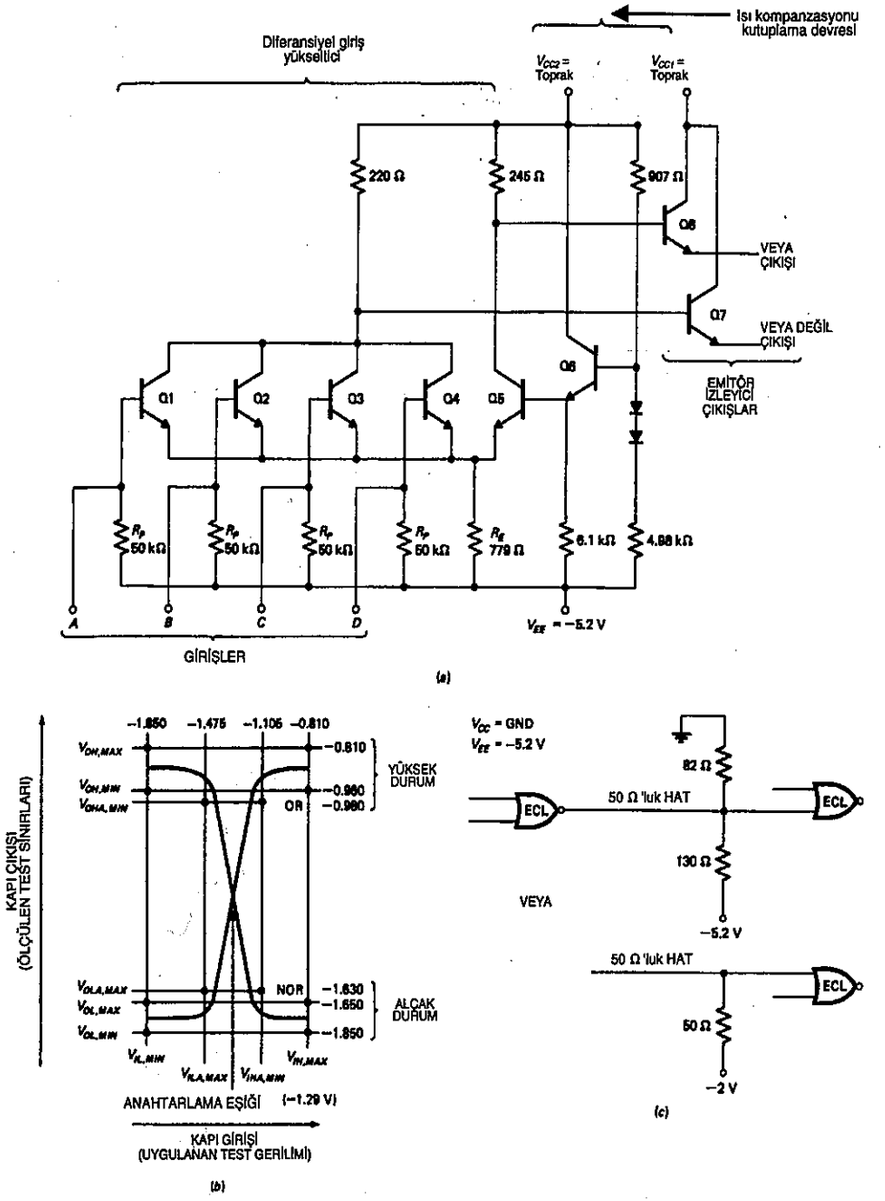
yakın olmasıdır. Ayrıca alçak-güçlü Schottky, *CMOS* ile standart *TTL* arasında uygun bir bağdaştırma ailesi oluşturmaktadır. Bu bölümde *CMOS*, başlığı altında daha sonra açıklanacağı gibi, çoğu *CMOS*, alçak-güçlü Schottky'yi sürebilmesine rağmen, standart *TTL* yi doğrudan sürememektedir. Bu halde alçak-güçlü Schottky ile iki standart *TTL* girişi sürülebilir. O da çoğunlukla yeterlidir. Standart *TTL* de bulunmayan bir çok mantık fonksiyonu şimdi alçak-güçlü Schottky *TTL* de bulunmaktadır.

TTL KAPILARIYLA ÇALIŞIRKEN HATIRLANMASI GEREKEN NOKTALAR:

1. Kullanılmayan girişleri ya doğrudan toprağa veya 1 KΩ luk bir direnç üzerinden V_{cc} ye bağlayın.
2. *TTL* kapılarının çıkışları birbirlerine bağlanmamalıdır. Açık-kollektörlü ve üç-durumlu *TTL* bu kuralın istisnalarıdır.
3. Maksimum V_{cc} değeri 7 V tur.
4. Maksimum sinyal girişi -1,5 V ile +5,5 V arasındadır.
5. Her *TTL* ailesi içinde maksimum çıkış yelpazesi 10 dur. 74SXX ve 74ALSXX ailelerinde çıkış yelpazesi 20 dir.
6. Her entegrenin yanına veya mümkün olduğunca sık olacak şekilde V_{cc} ile toprak arasında 0,1 veya 0,01 µF lik yangeçit kapasitörleri koyun.
7. Bağlantı uçları standart *TTL* için 12-14 inç (30,5-35,6 cm) den ve Schottky *TTL* için 3-4 inç (7,6-10,2 cm) den uzun olmamalıdır.

DAHA DA FAZLA HIZ: EMİTÖR KUPLAJLI MANTIK

Emitör-kuplajlı mantık, yani *ECL*, çok kısa yayılma gecikmelerine ulaşmak için *TTL* ailelerinden çok farklı devrelerden yararlanan mantık ailelerinin başka bir grubudur. Şekil 2-22*a* MECL 10.000 ailesinden tipik bir *ECL* kapısının devresini göstermektedir. Kaynak gerilimlerinin toprak ve -5,2 V oluşuna dikkat edin. Bu ailelerin en basit kapısı, *TTL* ailelerindeki VE DEĞİL kapısı yerine, (VEYA)-(VEYA DEĞİL) kapısıdır. Şekil 2-22a ya bakın. *Q2, Q3* ve *Q4* ü yok kabul edin. Basit bir fark yükseltici oluşturmak üzere *Q1* ve *Q5* in emitörlerinin birlikte bağlandığını göreceksiniz. *Q7* ve *Q8* transistörleri, fark yükselticinin kollektörlerinden alçak-empedanslı emitör-izleyici çıkışları sağlar. *Q6*, fark yükselticinin bir tarafındaki *Q5* e -1,3 V luk kararlı bir V_{BB} referans gerilimi sağlar. Eğer *Q1* ve *Q5* in bazları aynı gerilimde (-1,3 V) olsaydı, bu takdirde iki transistör, emitör akımını eşit olarak paylaşırdı. *Q1* in baz gerilimi



Şekil 2-22 ***(a)*** **Motorola MECL 10.000 ailesinin tipik emitör-kuplajlı mantık kapısı.** ***(b)*** **ECL VEYA-VEYA DEĞİL transfer eğrileri. (Motorola'nın müsadeleriyle).** ***(c)*** **ECL 50 Ω luk iletim hattının sonlandırılmaları.**

V_{EE} (-5,2 V) ye çekildiğinde, *Q1* den hiç akım akmaz ve kollektör gerilimi VEYA DEĞİL çıkışında mantık yüksek vermek üzere yükselir. *Q5* ten geçen akımın artması, VEYA çıkışında mantık alçak vermek üzere kollektör gerilimini düşürür. Tablo 2-2 den görebileceğiniz gibi minimum yüksek çıkış seviyesi -0,980 V ve maksimum alçak çıkış seviyesi -1,630 V tur. R_P iç dirençleri, kullanılmayan girişlerin bazlarını -5,2 V luk V_{EE} ye (mantık alçak) bağlar ve böylece bunlar çıkışı etkilemez. Şimdi, A girişi, alçak durumdan yüksek duruma (0,980 V) getirilirse, akım akışı *Q1* den *Q5* e geçecektir, çünkü şimdi *Q1* in bazı *Q5* in bazından daha yüksek gerilimdedir. Kesime giden *Q5* VEYA çıkışını yüksek seviyeye yükseltir, iletime geçen *Q1* VEYA DEĞİL çıkışını alçak seviyeye düşürür. Girişlerin herhangi birini veya hepsini yüksek seviyeye çıkarmak VEYA çıkışının yükseğe çıkmasına ve VEYA DEĞİL çıkışının alçağa düşmesine sebep olur. Tipik olarak 510 Ω olan emitör aşağı-çekme dirençleri, çıkış yükselme zamanları ve düşme zamanlarını eşitlemek için çıkışla -5,2 V arasına bağlanmışlardır.

TTL için göstermiş olduğumuz gibi, *ECL* için de giriş ve çıkış gerilim karakteristiklerini ve gürültü paylarını grafik olarak faydalı bir gösteriş şekli, transfer eğrisi iledir. Şekil 2-22*b*, "*ECL* MC10102 dörtlü iki-girişli (VEYA)-(VEYA-DEĞİL) kapısı" entegre devresinin transfer eğrilerini göstermektedir. İlk bakışta, farklı gerilim parametrelerinin sayısı şaşırtıcı görülebilir, ancak birer birer ele alınırlarsa anlaşılmaları kolaylaşacaktır.

GERİLİM PARAMETRELERİ -0,810 V luk $V_{IH, MAK}$ (25°C da) ın anlamı, bundan daha pozitif bir giriş geriliminde, üreticinin, aygıtın çıkış geriliminin bir sonraki aygıtın uygun giriş alanı içinde olacağını garantilemediği şeklindedir. -1,850 V luk $V_{IL, MIN}$, bir sonraki kapıya uygun çıkış seviyesi vermek için en negatif alçak giriş gerilimini belirtir. $V_{IHA-MIN}$ ve $V_{ILA-MAK}$ değerleri, bir sonraki kapıya nizami olmayan çıkış gerilimi vermeden, giriş geriliminin anahtarlama eşiğine ne kadar yaklaşabileceğini gösterir. $V_{OHA-MIN}$ ve $V_{OLA-MAK}$ değerleri, giriş anahtarlama eşiğine en yakın olan en-kötü-durum gerilimlerinde ($V_{IHA, MIN}$ *VE* $V_{ILA, MAK}$) iken en-kötü-durum çıkış gerilimlerini gösterir.

GÜRÜLTÜ PAYLARI Alçak durum için gürültü payı $V_{ILA, MAK} - V_{OLA, MAK}$ olarak tanımlanır. Ya Şekil 2-22*b* den ya da Tablo 2-2 den MECL 10000 serisi için alçak-seviye gürültü payı

-1,475 V- (-1,630 V) = 0,155 V

olarak bulunur. Yüksek durum için gürültü payı $V_{OHA, MIN} - V_{IHA, MIN}$ olarak tanımlanır. Bu takdirde yüksek-seviye gürültü payı

-0,980 V- (-1.095 V) = 0,115 V

olarak hesaplanır. *ECL* gürültü paylarını *TTL* ninkilerle (0,400 V) karşılaştırın. *TTL* ye göre *ECL* daha düşük gürültü paylarıyla tasarlanmıştır, çünkü ECL nin daha küçük çıkış empedansı onu gürültüye karşı daha az hassas yapar.

ECL nin çok kısa yayılma gecikme zamanı birçok tasarım hilelerinin sonucudur. Birincisi kaçak ve devre kapasitansının çabuk şarj ve deşarj olabilmesi için akımlar yüksek ve çıkış empedansı alçak tutulmuştur. Alçak empedanslı çıkış ayrıca kapılara büyük bir çıkış yelpazesi, uzun iletim hatlarını sürme yeteneği ve sinyal hatalarında daha az gürültü kapma şansı verir. İkincisi, mantık salınımı küçük (tipik olarak 1 V dan az) tutulmuştur, böylece kapasitans diğer mantık ailelerine göre daha az şarj ve deşarj olacaktır. Üçüncüsü, çıkış gerilim seviyeleri ve fark yükselteci akımı transistörlerin hiçbiri doyuma gitmeyecek şekilde ayarlanarak saklama zamanı problemlerinden kaçınılmıştır.

ECL nin dezavantajları, nispeten yüksek güç harcaması, *TTL* ile uyuşmayan gerilim seviyelerinin olması ve birkaç inçten uzun tel bağlantılarına iletim hattı gibi davranılmasını gerektiren hızlı yükselme zamanlı geçişleridir. İletim hatları kısmında açıklanan gerilim yansımasından kaçınmak için bu hatlar, hattın karakteristik empedans değerine eşit dirençlerle sonlandırılmalıdır. Şekil 2-22*c*, 50 Ω luk ECL iletim hattının ya 50 Ω luk bir direnci -2 V *a* bağlayarak ya da bunun Thevenin dengiyle; 82 Ω luk bir direnci toprağa ve 130 Ω luk bir direnci -5,2 V *a* bağlayarak uygun bir şekilde sonlandırılabileceğini göstermektedir. -5,2 V luk kaynak bir a.a. sinyali için toprak olduğuna göre paralel gözüken 82 ve 130 Ω luk dirençler 50 Ω luk bir denk direnç verir. İkinci yol -2 V luk ayrı bir kaynak ihtiyacından kaçınır, fakat daha fazla güç kullanır. Eğer sonlandırma yapılmışsa 510 Ω luk çıkış aşağı-çekme dirençlerine gerek yoktur.

Motorola Emitör-kuplajlı Mantık III (MECL III) ailesi çok güçlüdür çünkü çok yüksek frekanslarda çalışma yeteneği vardır, ancak 1 inç (25 mm) den uzun bağlantı tellerinin sonlandırılması gerekliliği bu aileyle çalışmayı zorlaştırmaktadır. 1968 de Motorola MECL 10.000 serisini üretti. Bu serinin kısa kapı yayılma gecikmeleri vardı, ancak çıkış yükselme zamanları, MECL III ün 1 ns sine göre daha uzun, 3,5 ns idi. Böylece hız düşmedi, fakat sonlandırılmamış maksimum tel uzunluğu 8 inçe (20,3 cm) yükseldi. MECL 10 K serisinin geliştirilmiş şekli olan MECL 10 KH serisi t_{PD} yi 1 ns ye azalttı. MECL 10 K çoğunlukla yüksek hızlı bilgisayarlarda kullanılır. Ayrıca bir MECL 10 K mikroişlemci ailesi ve 10800 gurubu vardır.

V_{EE} (-5.2 V) ye çekildiğinde, *Q1* den hiç akım akmaz ve kollektör gerilimi VEYA DEĞİL çıkışında mantık yüksek vermek üzere yükselir. *Q5* ten geçen akımın artması, VEYA çıkışında mantık alçak vermek üzere kollektör gerilimini düşürür. Tablo 2-2 den görebileceğiniz gibi minimum yüksek çıkış seviyesi -0,980 V ve maksimum alçak çıkış seviyesi -1,630 V tur. R_P iç dirençleri, kullanılmayan girişlerin bazlarını -5,2 V luk V_{EE} ye (mantık alçak) bağlar ve böylece bunlar çıkışı etkilemez. Şimdi, A girişi, alçak durumdan yüksek duruma (0,980 V) getirilirse, akım akışı *Q1* den *Q5* e geçecektir, çünkü şimdi *Q1* in bazı *Q5* in bazından daha yüksek gerilimdedir. Kesime giden *Q5* VEYA çıkışını yüksek seviyeye yükseltir, iletime geçen *Q1* VEYA DEĞİL çıkışını alçak seviyeye düşürür. Girişlerin herhangi birini veya hepsini yüksek seviyeye çıkarmak VEYA çıkışının yükseğe çıkmasına ve VEYA DEĞİL çıkışının alçağa düşmesine sebep olur. Tipik olarak 510 Ω olan emitör aşağı-çekme dirençleri, çıkış yükselme zamanları ve düşme zamanlarını eşitlemek için çıkışla -5,2 V arasına bağlanmışlardır.

TTL için göstermiş olduğumuz gibi, *ECL* için de giriş ve çıkış gerilim karakteristiklerini ve gürültü paylarını grafik olarak faydalı bir gösteriş şekli, transfer eğrisi iledir. Şekil 2-22*b*, "*ECL* MC10102 dörtlü iki-girişli (VEYA)-(VEYA-DEĞİL) kapısı" entegre devresinin transfer eğrilerini göstermektedir. İlk bakışta, farklı gerilim parametrelerinin sayısı şaşırtıcı görülebilir, ancak birer birer ele alınırlarsa anlaşılmaları kolaylaşacaktır.

GERİLİM PARAMETRELERİ -0,810 V luk $V_{IH,MAK}$ (25°C da) ın anlamı, bundan daha pozitif bir giriş geriliminde, üreticinin, aygıtın çıkış geriliminin bir sonraki aygıtın uygun giriş alanı içinde olacağını garantilemediği şeklindedir. -1,850 V luk $V_{IL,MIN}$, bir sonraki kapıya uygun çıkış seviyesi vermek için en negatif alçak giriş gerilimini belirtir. $V_{IHA-MIN}$ ve $V_{ILA-MAK}$ değerleri, bir sonraki kapıya nizami olmayan çıkış gerilimi vermeden, giriş geriliminin anahtarlama eşiğine ne kadar yaklaşabileceğini gösterir. $V_{OHA-MIN}$ ve $V_{OLA-MAK}$ değerleri, giriş anahtarlama eşiğine en yakın olan en-kötü-durum gerilimlerinde ($V_{IHA,MIN}$ *VE* $V_{ILA,MAK}$) iken en-kötü-durum çıkış gerilimlerini gösterir.

GÜRÜLTÜ PAYLARI Alçak durum için gürültü payı $V_{ILA,MAK} - V_{OLA,MAK}$ olarak tanımlanır. Ya Şekil 2-22*b* den ya da Tablo 2-2 den MECL 10000 serisi için alçak-seviye gürültü payı

-1,475 V- (-1,630 V) = 0,155 V

olarak bulunur. Yüksek durum için gürültü payı $V_{OHA,MIN} - V_{IHA,MIN}$ olarak tanımlanır. Bu takdirde yüksek-seviye gürültü payı

-0,980 V- (-1.095 V) = 0,115 V

olarak hesaplanır. *ECL* gürültü paylarını *TTL* ninkilerle (0,400 V) karşılaştırın. *TTL* ye göre *ECL* daha düşük gürültü paylarıyla tasarlanmıştır, çünkü ECL nin daha küçük çıkış empedansı onu gürültüye karşı daha az hassas yapar.

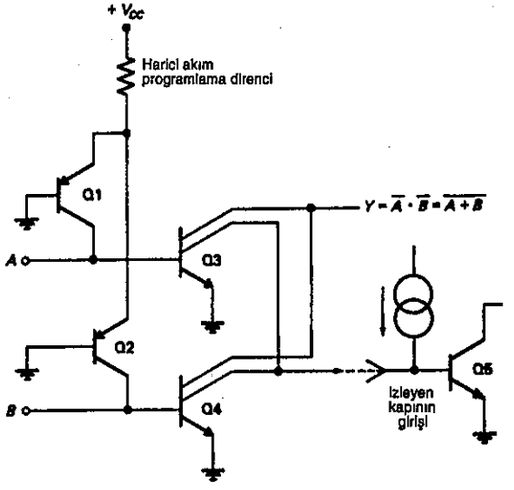
ECL nin çok kısa yayılma gecikme zamanı birçok tasarım hilelerinin sonucudur. Birincisi kaçak ve devre kapasitansının çabuk şarj ve deşarj olabilmesi için akımlar yüksek ve çıkış empedansı alçak tutulmuştur. Alçak empedanslı çıkış ayrıca kapılara büyük bir çıkış yelpazesi, uzun iletim hatlarını sürme yeteneği ve sinyal hatalarında daha az gürültü kapma şansı verir. İkincisi, mantık salınımı küçük (tipik olarak 1 V dan az) tutulmuştur, böylece kapasitans diğer mantık ailelerine göre daha az şarj ve deşarj olacaktır. Üçüncüsü, çıkış gerilim seviyeleri ve fark yükselteci akımı transistörlerin hiçbiri doyuma gitmeyecek şekilde ayarlanarak saklama zamanı problemlerinden kaçınılmıştır.

ECL nin dezavantajları, nispeten yüksek güç harcaması, *TTL* ile uyuşmayan gerilim seviyelerinin olması ve birkaç inçten uzun tel bağlantılarına iletim hattı gibi davranılmasını gerektiren hızlı yükselme zamanlı geçişleridir. İletim hatları kısmında açıklanan gerilim yansımasından kaçınmak için bu hatlar, hattın karakteristik empedans değerine eşit dirençlerle sonlandırılmalıdır. Şekil 2-22*c*, 50 Ω luk ECL iletim hattının ya 50 Ω luk bir direnci -2 V *a* bağlayarak ya da bunun Thevenin dengiyle; 82 Ω luk bir direnci toprağa ve 130 Ω luk bir direnci -5,2 V *a* bağlayarak uygun bir şekilde sonlandırılabileceğini göstermektedir. -5,2 V luk kaynak bir a.a. sinyali için toprak olduğuna göre paralel gözüken 82 ve 130 Ω luk dirençler 50 Ω luk bir denk direnç verir. İkinci yol -2 V luk ayrı bir kaynak ihtiyacından kaçınır, fakat daha fazla güç kullanır. Eğer sonlandırma yapılmışsa 510 Ω luk çıkış aşağı-çekme dirençlerine gerek yoktur.

Motorola Emitör-kuplajlı Mantık III (MECL III) ailesi çok güçlüdür çünkü çok yüksek frekanslarda çalışma yeteneği vardır, ancak 1 inç (25 mm) den uzun bağlantı tellerinin sonlandırılması gerekliliği bu aileyle çalışmayı zorlaştırmaktadır. 1968 de Motorola MECL 10.000 serisini üretti. Bu serinin kısa kapı yayılma gecikmeleri vardı, ancak çıkış yükselme zamanları, MECL III ün 1 ns sine göre daha uzun, 3,5 ns idi. Böylece hız düşmedi, fakat sonlandırılmamış maksimum tel uzunluğu 8 inçe (20,3 cm) yükseldi. MECL 10 K serisinin geliştirilmiş şekli olan MECL 10 KH serisi t_{PD} yi 1 ns ye azaltır. MECL 10 K çoğunlukla yüksek hızlı bilgisayarlarda kullanılır. Ayrıca bir MECL 10 K mikroişlemci ailesi ve 10800 gurubu vardır.

I^2L

Diğer bir bipolar transistor mantık yaklaşımı, akım enjeksiyon mantığı yani *entegre enjeksiyon mantığıdır (Integrated Injection Logic; I^2L).* Bu tip mantık, önceki ailelerde olduğu gibi ayrık kapı entegreleri için kullanılmaz fakat içinde binlerce kapı içeren sayısal kol saati devreleri veya Texas Instruments'ın SBP9989 bir çeşidi olan komple bir mikroişlemci gibi entegrelerde kullanılır. Temel I^2L kapısının devre yapısı Şekil 2-23 de gösterilmiştir. *Q1* ve *Q2* değerleri V_{cc} ve dış direnç tarafından programlanan akım kaynaklarıdır.



Şekil 2-23 (a) Temel I^2L kapısı.

Bu enjekte edilen akım (verilen adı budur), 10 na dan 100 μA e kadar herhangi bir değere ayarlanabilir. *A* girişi açıksa, yani yüksekse, *Q1* enjektör akımı *Q3* baz-emitör kavşağından geçer. Bu akım *Q3* ü doymaya götürür ve kollektöründe yaklaşık 50 mV luk bir alçak çıkış gerilimi oluşturur. Eğer A ya 50 mV luk bir alçak veya toprak girişi uygulanırsa, enjektör akımı Q3 ün bazından uzaklaştırılacak, *Q3* kesime gidecek ve kollektör çıkışı ona bağlanan yüksek gerilime kayacaktır. Şekil 2-23 teki gibi çıkış bir başka I^2L kapısının girişine bağlandığında, yüksek seviye çıkış gerilimi, *Q5* in V_{BE} si

olan 0,7 V olacaktır. *Q3* ve *Q4* paralel olduğuna göre eğer *A* veya *B* girişi yüksek ise, bunların kollektör kavşağının çıkışı alçaktır. Bu pozitif mantık VEYA fonksiyonunu gösterir.

I^2L mantığının bir çok avantajları vardır. Birincisi, kapıların basit olmaları ve az güç harcamaları, çok küçük bir yonga alanı üzerine birçok kapı koyma imkanı vermektedir. İkincisi, akımlar sabit olduğu için, *TTL* ve *CMOS*'ta olduğu gibi V_{cc} hattında anahtarlama gürültüleri oluşmaz. Üçüncüsü, enjektör akımını programlayarak, yayılma gecikmesini ve güç harcamasını geniş bir alan içinde değiştirebilirsiniz. 10 nA lik bir enjektör akımı 100μs lik bir yayılma gecikmesi verir ve 100 mA lik bir enjektör akımı yayılma gecikmesini yaklaşık 25 ns ye düşürür. Sayısal kol saati devresi gibi yavaş uygulamalar için enjektör akımı çok aşağılara çekilerek çok az güç harcanması sağlanır. I^2L nin diğer bir avantajı da işlemsel kuvvetlendirici gibi bipolar analog devrelerde aynı yonga üzerinde imal edilebilmesidir.

I^2L nin bir dezavantajı ise, üretim sırasında *LSI (büyük ölçekli entegrasyon)* pazarında ana rakibi *MOS* mantığından bir adım daha fazla işlem gerektirmesidir.

MOS MANTIK

Şimdiye kadar üzerinde durduğumuz mantık aileleri, devrelerinde bipolar transistörler kullanırlar. Bundan sonra inceleyeceğimiz aileler artma-modlu *MOS (metal-oksit yariletken)* transistörleri kullanır.

MOS TRANSİSTÖRLERİNİN GÖZDEN GEÇİRİLMESİ

MOS transistörleri ile ilgili olarak hafızanızı tazelemek üzere, Şekil 2-24 te bilinen MOSFET tiplerinin yapım, iletim eğrileri ve uygun gerilim bağlantılarıyla şematik sembolleri gösterilmiştir. Bu tiplerden ilki olan artma-modu (enhancement mode) tipinin yapısı Şekil 2-24a da gösterilmişitir. Bu tiplerden *kaynak ve akaca (drain)* bağlı N_+ bölgeler, taşıyıcıların çoğu elektron olan fazlaca katkılanmış N tipi silikondur. Bir *P*-tipi altlık, bir kat silikon dioksit ve silikon dioksiti kirlenmeden koruyan bir kat silikon nitrid ile kontrol kapısından elektriksel olarak yalıtılmıştır. Elemanın gösterilişine göre kaynaktan akaca iletken bir yol yoktur; çünkü altlık, kaynak ve akaç sırt sırta iki *PN* kavşağı oluşturmaktadır. Oda sıcaklığında, *P* altlık ısıl çalkantıdan oluşan bazı serbest elektronların yanısıra katkılanmadan oluşan delikleri içerir. Eğer altlık kaynağa bağlanırsa ve kapıya pozitif gerilim uygulanırsa, bu

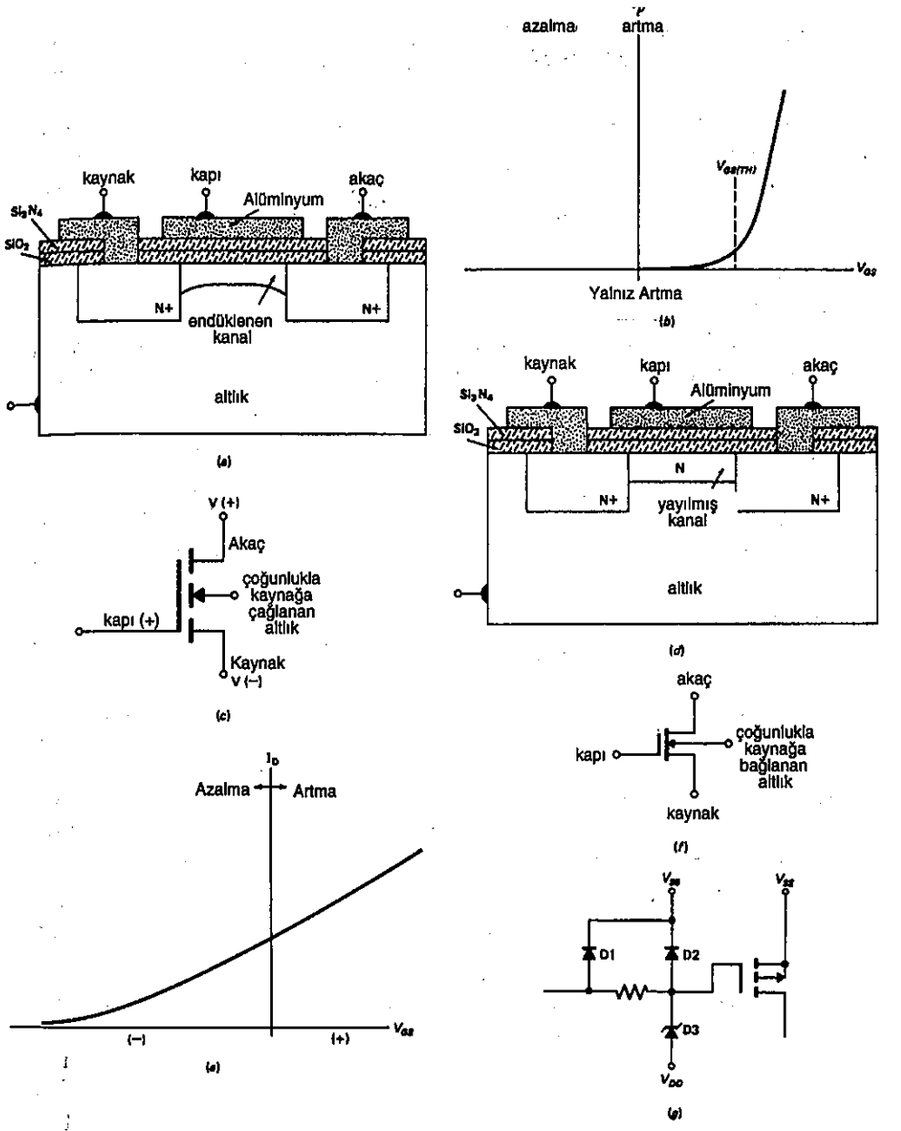
serbest elektronlar kapının yanındaki altlık kenarına doğru çekilecektir. Çekilen elektronlar kaynakla akaç arasında bir elektron iletimi yani *N* kanalı (Şekil 2-24a da indüklenen kanal diye isimlendirilmiştir) oluşturur. Şekil 2-24b, kapı ne kadar fazla pozitif yapılırsa aygıttan da o kadar fazla akaç akımının akacağını göstermektedir. Kapı ile kaynak arasındaki gerilim tipik olarak 2-3 V olan eşik gerilimi $V_{GS,TH}$ ye ulaşıncaya kadar çok az bir akaç akımı akar. Şekil 2-24c deki şematik sembolün kanalı kesikli çizgiyle göstermesi, iletimi artırmak için kapı pozitif yapılmadıkça bir kanalın var olmadığını size hatırlatmak içindir. Bu nedenle de *yalnız-artma (Enhancement-Only, E-only)* MOSFET ismi verilmiştir.

Şekil 2-24d de gösterilen azaltma-artma (depletion-enhancement, D-E) MOSFET inin yapısı yalnız-artma MOSFET ininkine benzer, ancak *N* tipi kanal *N* + kaynağı ile akaç bölgeleri arasına yayılmıştır. Bunun anlamı, kapı ile kaynak altlıkları (substrate) arasına sıfır volt bile uygulansa, kaynakla akaç arasında bir iletken kanal vardır ve akım akacaktır. Eğer kapı kaynağa göre daha pozitif yapılırsa, serbest elektronlar altlıktan çekilir ve kanal "artmıştır", böylece daha fazla akaç akımı akar. Eğer kapı daha negatif yapılırsa, elektronlar kanal bölgesinden uzaklaştırılır. Bu kanalı "azaltır" ve direncini artırır, böylece daha az akım akar. Şekil 2-24e bu D-E MOSFET te, kapı-kaynak geriliminin akaç akımı üzerindeki etkisini göstermektedir. Şekil 2-24*f* deki şematik sembol, kanalın herhangi bir gerilim uygulamadan önce var olduğunu hatırlatmak üzere, kanalı kesiksiz bir çizgiyle göstermektedir.

Yalnız-E MOSFET ler ve D-E MOSFET ler, *N* bölgelerini *P* ye ve *P* bölgelerini *N* ye çevirerek kolayca *P*-kanal aygıtları olarak yapılabilir. Uygulanan gerilimlerin yönleri de değişir. Bir P-kanal yalnız-E MOSFET için kaynak pozitif bir beslemeye ve akaç daha negatif bir beslemeye bağlanır. *P*-kanal yalnız-*E* elemanı iletime geçirmek için kapı kaynaktan daha negatif yapılır.

Bir MOS transistörün girişi elektriksel olarak ayrılmış olduğundan hemen hemen hiç giriş akımı gerekmemektedir ve giriş bir önceki kapının çıkışına sadece bir kapasitif yükmüş gibi davranır.

MOS-tabanlı mantığın aşırı yüksek giriş empedansı ciddi bazı problemler oluşturmaktadır, çünkü kapıyı ayıran ince oksit tabakası statik elektrik tarafından kolayca delinebilmektedir. Şekil 2-24g de görüldüğü gibi her girişe koruyucu devre koyarak aygıtın korunmasına çalışılmıştır. Giriş gerilimi, V_{SS} nin 0,7 V üstüne çıkarsa *D1* ve *D2* diyotları iletecekitir. *D3* bozulma gerilimi 35 V olan keskin-dizli bir zenerdir ve bir giriş gerilimi V_{DD} nin 35 volt üzerine çıkarsa iletime geçecektir. Kapıdaki oksit tabakasının bozulma gerilimi genellikle 100 ile 200 V arasında olduğundan, bu devre oksit tabakasını korumalıdır ve bir noktaya kadar da korur. Ancak, az nemli bir günde bir halı üzerinde yürürken oluşan statik elektrik koruma devresinin kapasitesinden 10 defa da-



Şekil 2-24 ***(a)*** **Yalnız-E MOSFET yapısı.** ***(b)*** **Yalnız-E MOSFET iletkenlik eğrisi.** ***(c)*** **Yalnız-E MOSFET in şematik sembolü.** ***(d)*** **D-E MOSFET yapısı.** ***(e)*** **D-E MOSFET iletkenlik eğrisi.** ***(f)*** **D-E MOSFET in şematik sembolü.** ***(g)*** **Tipik MOS girişini statikten koruma devresi.**

ha fazla olabilir. Kapının çalışmasını daha kötüleştirmeden koruma devresinin geliştirilmesi mümkün değildir, bu nedenle bu aygıtları ellerken ve kullanırken çok dikkatli olmak gerekir.

MOS AYGITLARINI KULLANIŞI İÇİN KURALLAR

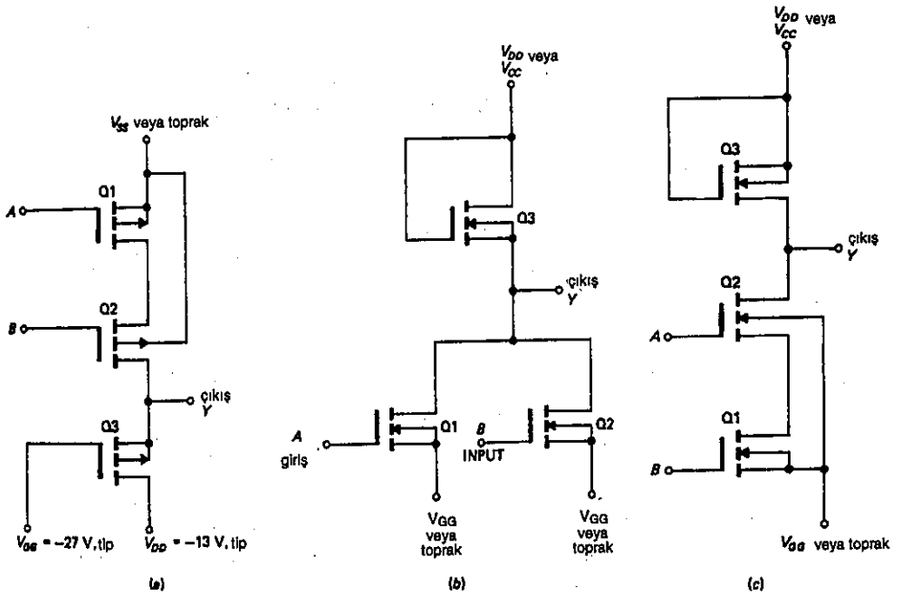
MOS la çalışırken dikkatsiz olmaya meyledebilirsiniz. Sonra yün bir gömlek üzerine naylon laboratuvar önlüğü giydiğiniz az nemli bir gün, dokunduğunuz her MOS devresi veya yongasının bozuk veya sınırda olduğunu görürsünüz. Naylon önlüğünüzün yün gömleğinize sürtünmesiyle oluşan statik gerilim sizi "MOS katili" ne dönüştürmüştür. Kötü şöhret sahibi olmaktansa, aşağıdaki MOS kullanma kurallarını takip edin:

1. Entegre devreler iletken köpük (genellikle donuk siyah veya pembe) veya alüminyum folyo içinde saklanmalıdır. Yongaları beyaz suni köpük (styrofoam) içine koymayın, çünkü bu köpük mükemmel bir statik üreticidir.
2. Havyanızın ucunu, metal çalışma masanızı ve kullandığınız her test cihazının şasesini gerçek toprağa bağlayın.
3. Az-nemli, yüksek statikli durumlarda, bileğimizi bir boy kablo ile seri bağlı 1 MΩ luk dirençle gerçek toprağa bağlayın. Direnç, statiğin akıp gitmesini sağlar.
4. Bütün entegre devrelerde olduğu gibi, güç devreye bağlı iken MOS entegre devrelerini takıp çıkarmayın.
5. Güç kesikken herhangi bir girişe sinyal uygulamayın. V_{cc} yokken, bir alçak-çıkış empedanslı sinyal üreteci, koruma diyotlarını ileri-kutuplayabilir ve onları bozmaya yetecek kadar akım geçmesine neden olabilir.
6. Kullanılmayan bütün girişleri, V_{cc} veya V_{DD} den hangisi istenilen mantığa uygunsa, ona bağlayın. Uçları asla açık bırakmayın, çünkü açık girişler statik toplar ve beklenmedik mantık seviyelerine kayabilir.
7. Baskılı-devre kartının dışında başka bir yere bağlanan MOS girişlerine seri olarak dirençler koyun.

MOS MANTIK AİLELERİ

MOS transistörleri kullanan kapılar daha az güç harcadıkları ve daha küçük yonga alanı gerektirdikleri için MOS mantığı, kaydırma kaydedicileri, bellekler ve mikroişlemciler gibi bir çok LSI devrede kullanılmaktadır. MOS mantığının bilinen değişik türleri PMOS, NMOS, CMOS, DMOS, HMOS, VMOS ve SOSMOS dur. Bu ailede TTL deki gibi ayrık yongalar üzerinde basit kapı fonksiyonları olan tek grup CMOS dır.

PMOS Bu ailenin toplu üretilen ilk gurubu PMOS, artma-modu P-kanal MOS transistörleri kullanarak kapıları oluşturur. Şekil 2-25*a* temel bir yüksek-eşikli P-kanal kapısının devresini göstermektedir. Tipik kaynak gerilimlerinin toprak, -13 ve -27 V olduğuna dikkat edin. Bazı cinslerinde, *Q3* ün akacı V_{GG} ye bağlanarak kaynaklardan biri gereksiz hale getirilmiştir. Bu halde *negatif kaynağa* V_{GG} denir. Bu devrede *Q3*, *Q1* ve *Q2* için sadece bir direnç veya akım kaynağı yükü görevini görür. Diğer bir deyişle, *Q3*, iletimindeki direnci, *Q1*



Şekil 2-25 *(a)* **Temel PMOS kapısı.** *(b)* **NMOS VEYA DEĞİL kapısı.** *(c)* **NMOS VE DEĞİL kapısı.**

ve Q2 nin her ikisi de iletimdeyken oluşturdukları denk dirençten daha büyük olacak şekilde tasarlanmıştır. Eğer *A* veya *B* girişi V_{ss} ye yakın bir yüksek gerilimde ise, *Q1* veya *Q2* kesimdedir ve çıkış gerilimi *Q3* tarafından V_{DD} civarında alçak bir gerilime çekilir. A ve B girişlerinin her ikisi de V_{DD} civarında alçak bir gerilime çekilirse, *Q1* ve *Q2* nin her ikisi de iletir. Bu, çıkışı, iki transistörün küçük iletim direnci üzerinden pozitif kaynağa bağlar ve V_{ss} civarında bir yüksek çıkış elde edilmiş olur. Pozitif mantıkta bu kapı bir VEYA DEĞİL kapısıdır: $A.B=Y=A+B$. Daha yüksek eşikli PMOS için gerekli giriş mantık seviyeleri 0 V ile V_{DD} civarında bir gerilimdir.

ALÇAK-EŞİKLİ PMOS PMOS un ilk yüksek-eşikli tiplerini TTL sinyalleri ile bağdaştırmak için yüksek gerilim bölücüleri gerekmekteydi. Daha sonra iç transistörler için metal kapılar yerine silikon kapılar kullanan alçak-eşikli PMOS tipleri çıktı. Bu hızı artırmakta ve TTL den daha iyi bir giriş-çıkış bağdaşması vermektedir. Alçak eşikli PMOS için tipik güç kaynağı gerilimleri +5 V luk V_{cc}, -5 V luk V_{DD} ve -12 V luk V_{GG}; veya +5 V luk V_{cc} ile -12 V luk V_{DD} ve V_{GG} dir. Bu gerilim guruplarından herhangi biriyle çalışan silikon-kapılı PMOS lar bir standart TTL yükünü sürebilir. Çoğu alçak-eşikli PMOS aygıtları +5 V ve 0 V luk veya +5 V ve -12 luk giriş mantık seviyelerini kabul edebilir.

NMOS *P*-kanal tipi, üretiminin *N*- kanal tipininkinden daha az kirlenme problemleri olması nedeniyle PMOS ailesinin ilk gurubuydu. *N*-kanal aygıtların daha hızlı olması ve transistör başına daha küçük yonga alanı gerektirmesi, üretim teknolojisi gelişir gelişmez, üreticileri NMOS aygıtları geliştirmeye yöneltmiştir. MOS belleklerin ve mikroişlemcilerin çoğu *N*-kanal MOS un bir türüdür. Şekil 2-25b ve c temel NMOS VEYA DEĞİL kapısı ile VE DEĞİL kapısının devresini göstermektedir. VEYA DEĞİL kapısında (Şekil 2-25b), N-kanal artma-modlu FET *Q3*, sadece bir yük direnci veya akım kaynağıdır. A ve *B* girişleri V_{GG} civarında bir gerilimde ise, Q1 ve Q2 kesimdedir ve çıkış Q3 tarafından yukarı çekilir. Eğer *A* veya *B* girişinden biri V_{DD} gerilimi civarına çekilirse, *Q1* veya *Q2* iletime geçer ve çıkışı V_{GG} civarında aşağı çeker. Bu pozitif mantık VEYA DEĞİL kapısının ifadesi $Y=A+B$ i sağlar. Şekil 2-25c deki devrenin analizini yaparak bir pozitif VE DEĞİL kapısı olarak nasıl çalıştığını görebilirsiniz.

8080A mikroişlemcisi gibi ilk LSI NMOS devreleri V_{cc} olarak +5 V ve V_{BB} önkutuplama gerilimi olarak -5V kullanmaktadır. Bunlar, iç devrelerin hızını artırmak için ayrıca +12 V luk V_{DD} kaynağı kullanırlar. Çıkışların gerilimleri TTL ile uyuşur ve bir TTL girişini alçağa çekebilecek kadar akım çöktürebilirler. 2102 RAM (rastgele erişimli bellek) gibi daha sonraki bazı NMOS devreler,

silikon-kapı teknolojisi ile birlikte it-çek çıkış yapısını kullanarak tek bir TTL yükünü tek bir +5 V luk kaynak kullanarak sürmeye yeterli hale getirilmiştir.

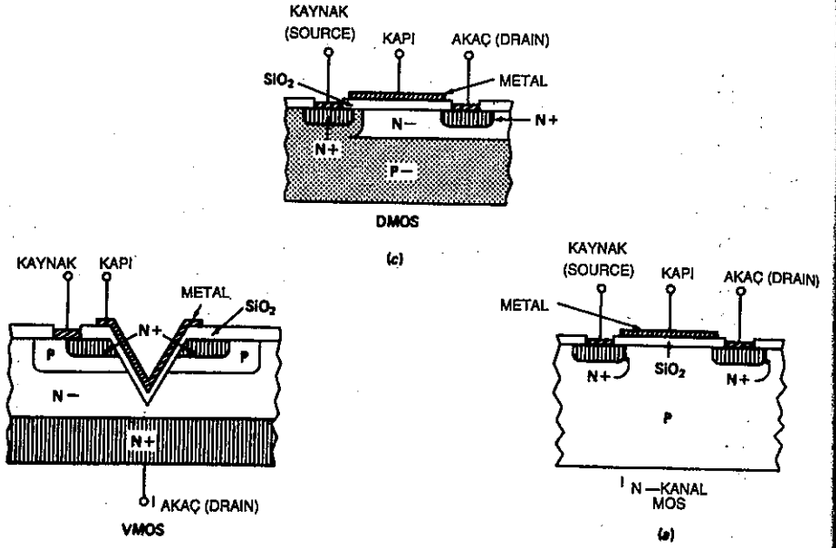
PMOS ve ilk NMOS devrelerini içeren sistemlerde arıza ararken, büyük bir olasılıkla çok çeşitli gerilim kaynakları ile karşılaşacaksınız. Burada bütün mümkün gerilim ve akım seviyelerinin ayrıntılarını veremeyeceğiz. Tablo 2-2, +5 ve -12 V ta çalışan alçak-eşikli *P*-kanal silikon-kapı MOS ile +5, -5 ve +12 V ta çalışan *N*-kanal silikon-kapı MOS un giriş-çıkış seviyelerini ve diğer karakteristik değerlerini vermektedir. Ayrıca +5 V ve toprak ile çalışan silikon-kapı NMOS a ait çalışma değerleri de gösterilmiştir.

VMOS, DMOS VE HMOS VMOS, DMOS ve HMOS daha az yayılma gecikmesi için NMOS un değişik şekillerde üretilmesi ile elde edilen devrelerdir. Şekil 2-26*a*, *b* ve *c*, standart NMOS, VMOS ve DMOS yapılarını karşılaştırmaktadır. VMOS ismini *V* şeklindeki yapısından ve akımın kaynaktan akaca, standart NMOS taki yatay yerine, *düşey (vertical)* olarak akmasın-dan dolayı almıştır. Küçük kapasitansları ve yüksek hızları dolayısıyle VMOS transistörleri LSI mantık kapılarının yanısıra radyo-frekans güç yükselticilerinde de gelecek vadetmektedir.

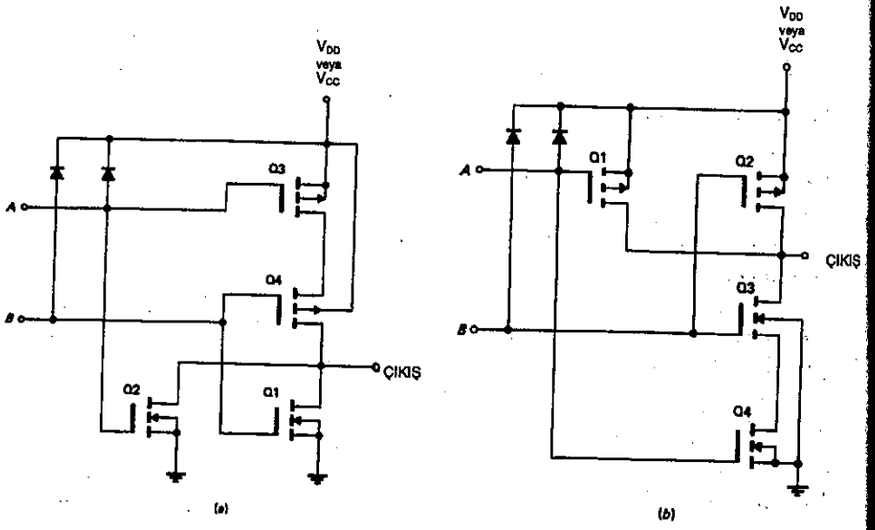
DMOS, kapı bölgesinde çift-dağılmış katkılanma kullanarak yayılma gecikmesini kısaltmak üzere etkin kanal uzunluğunu azaltır. Şekil 2-26c de bu, standart NMOS da görülmeyen, kapı altındaki *N*- bölgesiyle temsil edilmiştir. HMOS, yani yüksek-başarımlı MOS, yonga üzerindeki *N*-kanal MOS transistörün bütün boyutlarını orantılı olarak küçülterek kısa yayılma gecikmelerine ulaşır. HMOS ile ilgili tek sıkıntı en iyi çalışmayı elde etmek için gerekli kaynak geriliminin artık yerleşmiş olan standart +5 V dan daha az olmasıdır. Bütün bu tipler ECL den daha az güç ve yonga alanı kullanmakla birlikte teorik olarak ECL hızlarına ulaşabilirler.

CMOS

LSI devreleri için *P*-kanal teknolojisi geliştirilirken, yaklaşık aynı zamanlarda TTL ile karşılaştırılabilir mantık fonksiyonları bulunan fakat MOS aygıtlarının tipik alçak-güç harcama özelliğini taşıyan bir mantık ailesi üretilmek üzere tümleyici (complementary) MOS (CMOS) kullanılmıştır. CMOS devreleri, PMOS dan çok daha fazla yonga alanı kullanır, fakat daha hızlıdırlar, giriş-çıkış özellikleri bilinen mantık aileleriyle daha iyi uyuşur ve 3 ile 5 V arasındaki tek bir gerilim kaynağı ile çalışırlar. Şekil 2-27*a* ve *b* temel CMOS VEYA DEĞİL ve VE DEĞİL kapılarının devrelerini göstermektedir. Bu aileye adını veren, tümleyici *N*- ve *P*-kanal artma-modu MOS transistörlerin kullanılmasıdır.



Şekil 2-26 *(a)* **NMOS yapısı.** *(b)* **VMOS yapısı.** *(c)* **DMOS yapısı. (Electronic Desing).**



Şekil 2-27 *(a)* **CMOS VEYA DEĞİL kapısının yapısı.** *(b)* **CMOS VE DEĞİL kapısının yapısı.**

Şekil 2-27a da bir temel CMOS VEYA DEĞİL kapısı görülmektedir. Eğer *A* girişi V_{DD} (V_{cc}) yakınında yüksek bir seviyede ise, bu takdirde *P*-kanal *Q3* kesimde olacak ve *N*-kanal *Q2* iletimde olacaktır. Böylece *Q2*, çıkışı toprak civarında alçak seviyeye çekecektir. *B* girişindeki bir yüksek *Q4* ü kesime götürüp, Q1 i iletime geçirecek ve böylece çıkışı alçağa çekecektir. Eğer *A* ve *B* girişlerinin her ikisi de toprak civarında alçak seviyede ise *Q1* ve *Q2* kesimde Q3 ve Q4 iletimde olacaktır. Bu, çıkışı V_{DD} (V_{cc}) civarında yüksek bir seviyeye çeker. *A*, *B*, veya hem *A* hem de *B* yüksekse çıkış alçak olduğuna göre bu devre VEYA DEĞİL fonksiyonunu gerçekleştirir. Benzeri analizle Şekil 2-27b deki CMOS devrenin VE DEĞİL görevini gördüğünü gösterebilirsiniz. CMOS un dört bilinen serisi vardır: Orijinal 4000A serisi; bunun gelişmişi 4000B serisi; Fairchild 4500 serisi ve 4000B serisiyle benzer özelliklere sahip fakat TTL yongalarındaki aynı mantık fonksiyon ve bacak numaralarına sahip National'ın 47C00 serisi.

GİRİŞ/ÇIKIŞ KARAKTERİSTİKLERİ VE GÜRÜLTÜ PAYI

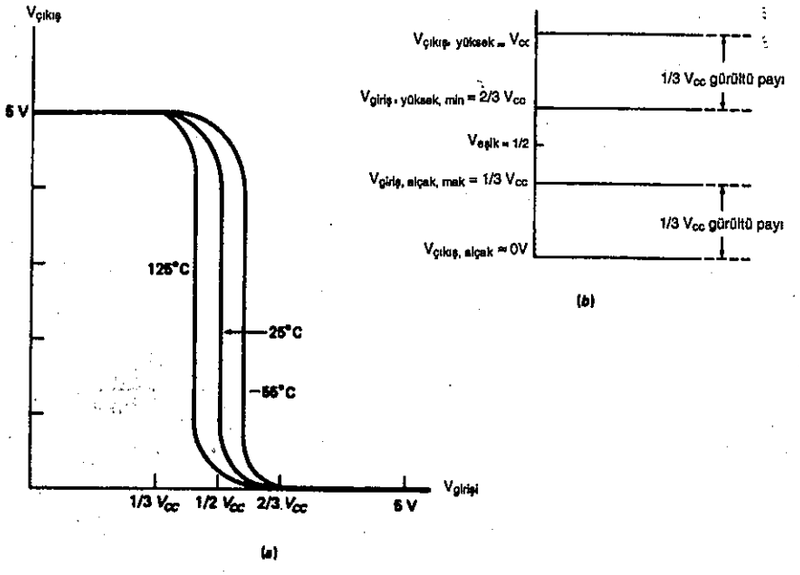
Artık CMOS kapılarının temel yapılarını görmüş olduğumuza göre, bunların giriş ve çıkış karakteristikleri üzerinde durabiliriz. Tablo 2-2, 5 V luk V_{cc} ile 4000A serisinin, 5 V ve 15 V kaynaklarla 4000B serisinin, 5 V kaynakla 74C00 serisinin ve yeni 74HC00 serisinin parametrelerini göstermektedir. (Not: MOS aygıtlarının besleme gerilimlerinin indislerinde görülen karmaşayı önlemek üzere 74C ve 74HC serilerinin kullanıldığı +5 V luk pozitif besleme geriliminin genellikle V_{cc} olarak adlandırıldığına dikkat edin. Diğer CMOS veya NMOS aygıtlarda genellikle pozitif besleme V_{DD} ve negatif besleme veya toprak V_{SS} olarak adlandırılır. +5 V luk besleme akaç (drain), kaynak (source) veya kollektöre bağlanmış olduğuna bakılmadan endüstride V_{cc} olarak bilinir hale gelmiştir.) Bir CMOS kapısının girişi bir MOS transistör olduğuna göre, giriş empedansı çok yüksektir ve alçak veya yüksek halde iken sadece yaklaşık 10 pA lık bir giriş akımı gereklidir. Gerçek entegre devrelere, Şekil 2-24g de gösterildiği gibi statik elektrikten oluşacak bozulmaları engellemek amacıyla diyot devreleri eklenmişitir. Toplam giriş kapasitansı 5-7 pF dır.

Eğer elinizde bir CMOS kapısı ve bir çift-izli osiloskop varsa, osiloskopu Bölüm 1 de anlatıldığı gibi kullanarak kapının giriş-çıkış transfer eğrisini elde edebilirsiniz (Şekil 2-28*a* 'ya bakınız). En-kötü durum giriş gerilim seviyeleri; alçak giriş geriliminin maksimum değeri 1/3 V_{cc} ve yüksek giriş geriliminin minimumu 2/3 V_{cc} dir. Eşik gerilimi yaklaşık 1/2 V_{cc} dir. En-kötü durum çıkış gerilim seviyeleri ise yüksek seviye için yaklaşık V_{cc} den 10 mV daha az ve alçak seviye için V_{SS} veya topraktan 10 mV daha fazladır. Şekil 2-28b deki gibi

bir diyagram çizerek giriş ve çıkış seviyelerinin, yüksek seviye için 1/3 V_{cc} ve alçak seviye için 1/3 V_{cc} lik etkin gürültü payları verdiğini gösterebilirsiniz. 5 V luk besleme ile çalışan bir CMOS için gürültü payı yaklaşık 1,7 V tur. Bu da TTL nin gürültü payının 4 katından fazladır. Ancak, MOS devrelerinin giriş ve çıkış empedansları daha yüksek olduğundan onların bağlantı telleri daha kolay gürültü kapar ve gerçekte 5 V luk CMOS ile TTL nin gürültü red oranları kabaca aynıdır. Daha yüksek çalışma gerilimlerinde CMOS daha iyi gürültü reddetme oranına sahiptir.

CMOS, çıkış geçişleri sırasında V_{cc} hattı üzerinde gürültü oluşturur çünkü her iki çıkış transistörünün iletimde olduğu kısa anda V_{cc} den toprağa bir aşırı akım çekilir. Yangeçit kapasitörleri bu problemi gidermeye yardımcı olur.

Sık sık CMOS un TTL ile uyuştuğu söylenir, ancak bu biraz yanıltıcı olabilir. Bir problem, bazı CMOS ların TTL leri sürecek yeterli çıkış akımları olmamasından kaynaklanmaktadır.



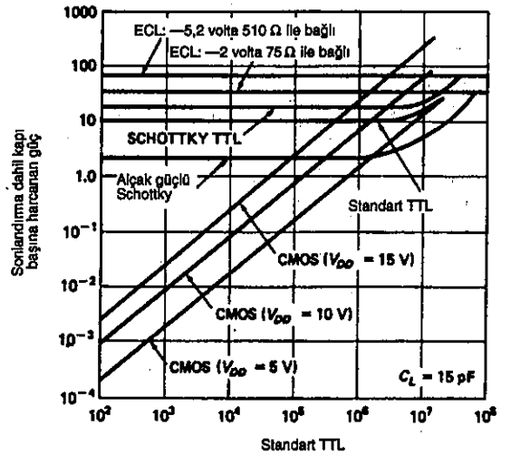
Şekil 2-28 *(a)* **CMOS transfer eğrileri.** *(b)* **CMOS seviyeleri ve gürültü payları.**

5 V ta çalışan bir CMOS un çıkış akımı 0,4 V luk alçak çıkış geriliminde 0,4 mA dir. Bu, iki alçak-güçlü TTL girişini veya bir alçak-güçlü Schottky girişini sürmeye yeterlidir, fakat standart TTL yi sürmek için, 4050 gibi bir tampon kullanmalısınız. CMOS-TTL uyuşmasındaki diğer bir problem de mümkün yetersiz gerilim seviyeleridir. 5 V besleme ile çalışan bir CMOS kapısının girişi, yüksek giriş için minimum 3,6 V olmalıdır. TTL üreticileri standart totem-direği çıkışı için sadece 2,4 V luk yüksek seviye gerilimini garanti etmektedir. 3,5 V luk tipik TTL çıkışının, dışarıdan bir 10 KΩ luk dirençle TTL çıkışı V_{cc} ye bağlanmadığı sürece, CMOS girişini yüksek seviyeye çekeceğine güvenilemez. Daha sonra bu bölümde farklı mantık ailelerinin bağdaştırılması ayrıntılarıyla incelenecektir, ancak yukarıdaki örnek sizlere ne gibi etkenlerin içerildiğini göstermektedir.

YAYILMA GECİKME ZAMANLARI 5V luk beslemeyle çalışan CMOS'un yüzlerce nanosaniyelik yayılma gecikmeleri, bunların kullanım alanlarını birkaç megahertzle sınırlar. Besleme gerilimini 15 V a çıkarmak yayılma gecikmesini 100 ns nin altına düşürür ve CMOS un biraz daha yüksek frekanslarda kullanılmasına imkan verir. CMOS un yayılma süresi ve yükselme zamanı kapasitif yüklemeyle oldukça uzar. 15 pF lik, yani üç kapı girişinin kapasitif yükü ile, tipik bir B-serisi kapının 50 ile 75 ns arasında bir maksimum t_{PD} si *(yayılma gecikme zamanı)* vardır. Eğer yük 50 pF a, yani 10 kapı girişine artırılırsa, t_{PD} maksimum 360 ns ye artar. CMOS un uzun ve değişken t_{PD} si, bazı devrelerde zamanlama problemine sebep olabilir.

GÜÇ HARCAMASI CMOS un avantajlarından biri, alçak frekanslarda çalıştırılırken çok az güç harcamasıdır. Tipik bir CMOS kapısı 5 V luk kaynakla 1 KHz de çalışırken yaklaşık 10 µW harcar. Çalışma frekansının niçin güç harcaması ile birlikte bahsedildiğini merak edebilirsiniz. Bunun sebebi çalışma frekansı artarken CMOS taki güç harcamasının her iki ana kaynağı da artar. Birincisi, anahtarlama sırasında her iki çıkış transistörü kısa bir an için iletimde iken V_{cc} den toprağa akan akımdır. Daha yüksek anahtarlama frekanslarında, bu transistörler daha sık iletime geçerler ve böylece ortalama güç harcaması artar. Güç harcamasının diğer ana kaynağı kapı girişindeki kapasitansı şarj ve deşarj etmek için akan akımdır. Giriş alçaktan yüksek duruma değiştirildiğinde, giriş kapasitansını V_{cc} yakınında bir gerilime şarj etmek için yukarıdaki P-kanal çıkış transistörü üzerinden akım sağlanmalıdır. Çıkış alçağa gidince bu giriş kapasitansı alttaki *N*-kanal çıkış transistörü üzerinden toprağa deşarj olur. Yüksek frekansta çalışma, giriş kapasitansının daha sık şarj ve deşarj olması demektir. Bu da toprağa akan ortalama akımın ve ortalama güç harcamasının artması demektir. Birçok mantık ailesinin giriş frekansına göre kapı başına olan tipik güç harcamasının gösterildiği Şekil 2-29 dan görebileceğiniz gibi, 1 MHz in üstünde 5 V CMOS un güç harcaması alçak-güçlü Schottky TTL den daha büyüktür. 5 MHz in üstünde 15 V luk CMOS, ECL den bile daha fazla

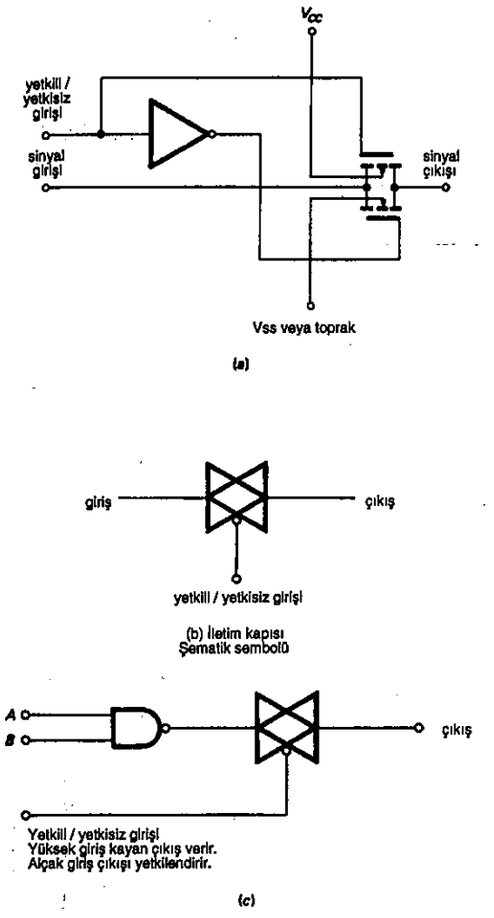
güç kullanır. Çoğunlukla CMOS alçak-frekanslı, alçak-güçlü, pille-çalışan elemanlarda kullanılır.



Şekil 2-29 Mantık ailelerinin giriş frekansına göre tipik güç harcamaları. ***(Fairchild Low Power Schottky Data Book.)***

CMOS İLETİM KAPILARI VE ÜÇ-DURUMLU ÇIKIŞLAR

Standart CMOS çıkışları, totem-direği TTL çıkışlarında olduğu gibi aynı nedenlerle birbirlerine bağlanamazlar. Çıkışlardan biri yükseğe gitmek isterken diğeri alçağa gitmek isterse toprağa bir doğru akım yolu oluşur. 10 V veya daha fazla bir gerilimle beslenen CMOS için bu genellikle yongalardan biri veya her ikisi için ölümcüldür (yongalar bozulacaktır). Bir çok kapı çıkışının ortak bir veri taşıt hattına bağlanması gereken ve herhangi bir anda sadece bir çıkışın seçildiği durumlarda üç-durumlu çıkış CMOS aygıtları kullanılır. TTL de olduğu gibi üç çıkış durumu; yüksek, alçak ve kayan veya farketmez durumlarıdır. Bazı CMOS lar üç-durumlu çıkışı oluşturmak üzere **iletim kapısı** denilen özel bir devre kullanır (Şekil 2-30 *a* bakın). Bu devredeki *N*- ve *P*-kanal transistörlerin her ikisi de iletime geçirilirse, bu takdirde çıkış, paralel transistörlerin alçak kanal direnci ile girişe bağlanmış olur. Çıkış, yüksek veya alçak girişi izler. Her iki transis-tör de kesimde ise açık devre veya açık anahtar olarak görünürler ve böylece çıkışlar kayar durumda olur. İletim kapısındaki ters çevirici *N*- ve *P*-kanal transistörleri kontrol için gerekli ters yönlü sinyalleri verir. Şekil 2-30b iletim kapısının şematik sembolünü göstermektedir. Yetkili/yetkisiz girişindeki bir alçak sinyal, çıkışı açar (yetkilendirir) ve yetkili/yetkisiz girişindeki bir yüksek sinyal, çıkışı kayan hale getirir.



Şekil 2-30 ***(a)*** **Basit CMOS iletim kapısı.** ***(b).*** **Şematik sembolü.** ***(c)*** **Standart kapıda üç-durumlu çıkış oluşturmak için kullanılan iletim kapısı.**

Şekil 2-30c bir iletim kapısının standart CMOS yapısına üç-durumlu çıkış vermek üzere nasıl bağlandığını göstermektedir. İletim kapıları gerçekte mantık-kontrollu anahtarlardır. Bu kapılar analog ve sayısal sinyallerin bir ortak hat üzerine seçilip konulmasında Bölüm 3 de anlatıldığı gibi kullanılır. Bir

pakette 4 veya 8 tane bulunur. Örneğin 4016 CMOS dörtlü çift yönlü anahtardır.

YALITKAN-ÜZERİNDE-SİLİKON CMOS

CMOS transistörlerini yalıtkan bir altlık (subtsrate) üzerine koyarak yüksek performanslı bir türü elde edilmiştir. Yalıtkan altlık, kapasitansı azaltır ve 50 MHz ve üstüne kadar çalışmaya imkan verir. Bazı elemanlarda altlık olarak safir kullanılır ve bunlara SOS (Silikon-üzerinde-Safir) denir. Diğer elemanlar altlık olarak silikon dioksit kullanır ve bunlara SOI (Yalıtkan-üzerinde-Silikon) denir. SOI CMOS çoğunlukla bellek ve mikroişlemciler gibi LSI devreler için kullanılır.

CMOS İLE ÇALIŞIRKEN HATIRLANACAK NOKTALAR

1. Herhangi bir MOS mantığı için olduğu gibi, statik elektrik tedbirlerine uyun.
2. Bütün kullanılmayan girişlerin V_{CC} veya toprağa bağlanmış olmasına dikkat edin.
3. Alçak empedanslı sinyal kaynaklarını CMOS devrelerin güç kaynakları ile aynı anda veya daha önce kapatın.
4. Standart CMOS çıkışı ile standart TTL sürmeye kalkmayın.
5. Üç-durumlu değilse, iki CMOS çıkışını birlikte bağlamaya kalkmayın ve üç-durumlu çıkışların da aynı anda ikisini birden yetkilendirmemeye dikkat edin.
6. t_{PD} yi mümkün olduğunca kısa tutmak için çıkışlardaki kapasitif yüklemeyi en aza indirin.
7. Nisbeten yüksek-frekanslı devrelerde CMOS un nisbeten uzun t_{PD} süresinin sebep olacağı zamanlama problemlerine dikkat edin.

GELİŞMEKTE OLAN YENİ MANTIK TEKNOLOJİLERİ

Yarı iletken üretim yöntemleri geliştikçe yongalar üzerindeki devre yoğunluğunu artırmak ve yüksek frekanslarda çalışan mantık devreleri yapmak için yeni teknolojiler aranmaktadır. Bazı firmalar, geleneksel mantıkta

kullanılan 2 mantık çıkış gerilim seviyesi yerine 10 taneye kadar seviye kullanarak kapı başına daha fazla bilgi taşıyan çok seviyeli mantık üzerinde çalışmaktadırlar. Bunun anlamı, örneğin bir hesap makinası yongasında çok daha az sayıda kapı kullanılması gerekeceğidir.

Hewlett-Packard yayılma gecikmesi 100 ps in altında olan GaAs (Galyum Arsenayd) FET leri ve Schottky diyotlarından kapılar üretmiştir. Bu, hızlı ECL den bile 5-10 defa daha hızlıdır. 10 ps yayılma gecikmesi olan *Superiletken Josephson* kavşaklı kapılar üretilmiştir. Bu gelişmeler birkaç gigahertz frekanslarında çalışabilecek ultra-yüksek hız ailelerine doğru bir gidişe yol açmaktadır.

GİRİŞ SİNYAL UYUMLANMASI VE SİNYAL KAYNAKLARI

GİRİŞ SİNYALİ GEREKSİNİMLERİ

Bir sayısal mantık devresini test ederken veya arızasını ararken en sık karşılaşılan problem uygun olmayan giriş sinyalleridir. Sayısal mantık için kullanılan sinyal kaynaklarının aşağıdaki özellikleri taşıması gerekir.

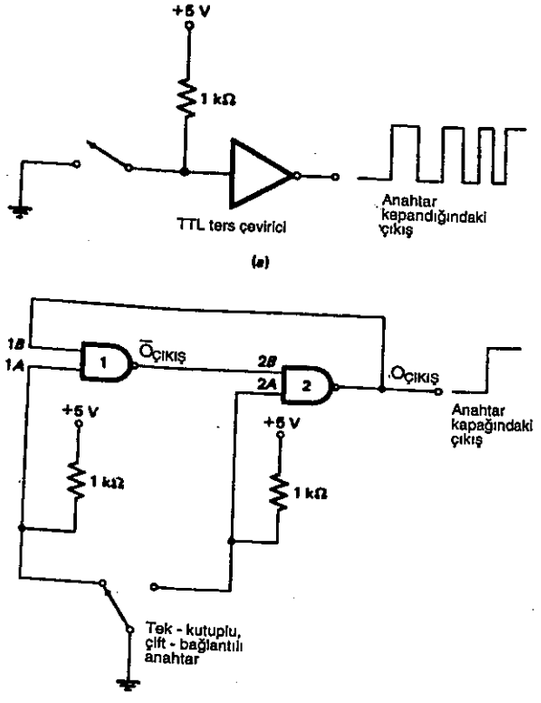
1. Çıkış geriliminin yüksek seviyesi kesinlikle minimum yüksek giriş geriliminin üstünde olmalı fakat girişi yakacak $V_{GIRIŞ,MAK}$ dan da büyük olmamalıdır.
2. Çıkış geriliminin alçak seviyesi maksimum alçak giriş geriliminin çok altında olmalı, fakat $V_{IN,MIN}$ den büyük olmalıdır. ($V_{GIRIŞ,MIN}$ den daha küçük bir gerilim girişi bozabilir).
3. Sinyal kaynağının çıkışı yük altında uygun mantık seviyelerini koruyabilmek için yeterli akım kaynaklandırılabilmeli veya çöktürebilmelidir.
4. Eğer sinyal girişi bir anahtar ise, girişe anahtar kontaklarının her zıplamasıyla oluşan darbeler değil, tek bir geçiş uygulanmalıdır.
5. Eğer sinyal kaynağı bir saat veya darbe kaynağı ise, darbelerin yükselme zamanı, sürülen kapının çıkışı salınmayacak veya iki çıkış transistörünün iletimde olduğu aktif bölgede yongayı aşırı ısıtacak kadar uzun kalmayacak şekilde hızlı olmalıdır.

İlk üç gereksinim için Tablo 2-2 deki özel değerleri kullanın. Problemlerin bir kaynağı, çıkışı sıfır üzerinde merkezlenmiş sinyal üreteçleridir. Darbe çıkış genliği, kapı girişini mantık yüksek seviyeye sürmek için yeterince büyük bir değere ayarlandığında, darbenin negatif salınımı yonganın maksimum giriş gerilim değerini aşacaktır. TTL için yaklaşık 0,7 V dan daha negatif

...lerin giriş kelepçe diyotlarını ileri kutuplayacaktır. Bu diyotlardan geçen ...mA lik veya daha fazla bir akım büyük bir ihtimalle bunları bozacaktır. ...Q" ve CMOS devrelerinin statik-koruma diyotları da giriş sinyali salınımı ...çok negatif ise veya sinyal üretici açıkken yonganın güç kaynağı kapatılırsa aynı akıbete uğrayacaktır. Bu bölümün daha sonraki bir kısmında bir mantık ailesinden diğer birine uygun seviyeli sinyalleri nasıl ulaştıracağımızı göreceğiz.

ANAHTARLAR VE ANAHTAR ZIPLAMALARININ GİDERİLMESİ

Sinyal kaynakları olarak anahtarları kullandığınızda, anahtar kontaklarının zıplaması birkaç milisaniye sürebilir. Şekil 2-31*a* da gördüğünüz gibi

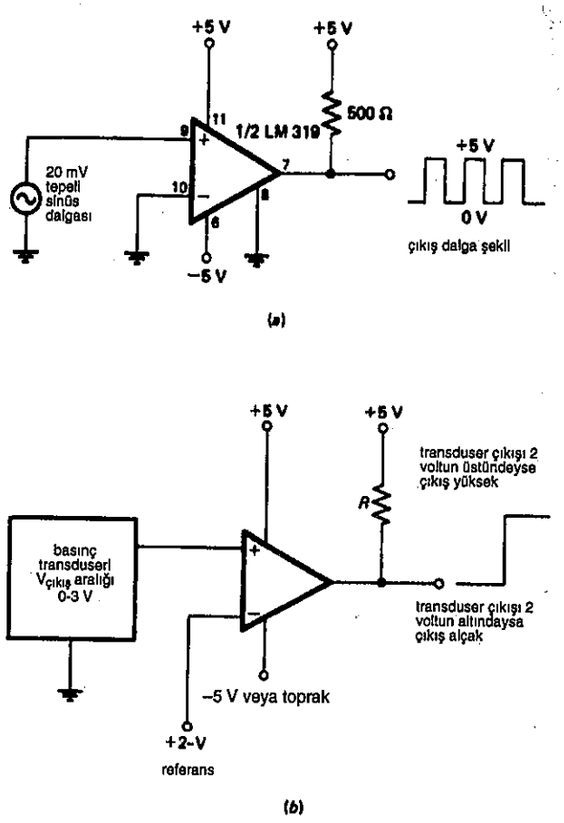


Şekil 2-31 *(a)* **Anahtar zıplamasının etkisini gösteren TTL ters çevirici çıkışı.** *(b)* **Anahtar zıplamasını önleyen basit bir devre.**

zıplama, istenilen tek bir alçaktan yükseğe geçişi sağlamak yerine, yarım düzine civarında çıkış darbesi üretir. TTL girişi her kontak zıplaması arasında yükseğe çekilir. Bir saklayıcı osiloskop bu zıplamayı göstermek için kullanılabilir. Şekil 2-31*b* zıplaması giderilmiş çıkış geçişleri üreten bir devreyi göstermektedir. Eğer anahtar başta *1A* girişini toprağa bağlıyorsa, bu takdirde birinci VE DEĞİL kapısının *Q* çıkışı ve ikinci kapının 2B girişi yüksektir. *2A* girişi +5 V a bağlı bir dirençle yükseğe çekilmiş olduğuna göre ikinci kapının her iki girişi de yüksek olur ve böylece *Q* çıkışı alçak olur. Anahtar diğer konuma çevrildiğinde, *1A* girişi yükselir ve *2A* girişi düşer. *2A* daki alçak giriş *Q* çıkışını ve *1B* girişini yükseğe çeker. Şimdi birinci kapının her iki girişi de yüksek olduğundan, *Q* çıkışı alçak olur. Birinci kapının *2B* girişine uygulanan bu alçak seviye, *2A* girişindeki zıplamalar ne olursa olsun *Q* çıkışını yüksekte tutar. Bu çıkışın tersi de birinci kapının $\overline{Q}$ çıkışından alınabilir. Çok sayıda anahtar kullanılan uygulamalarda, 74279 gibi üzerinde bir çok anahtar zıplamalarını giderici devrelerin bulunduğu entegre devre paketleri kullanılabilir. Mikroişlemcilerin veri girişlerindeki anahtarların zıplamaları, zıplama giderici yordamlarla giderilebilir.

KARŞILAŞTIRICILAR

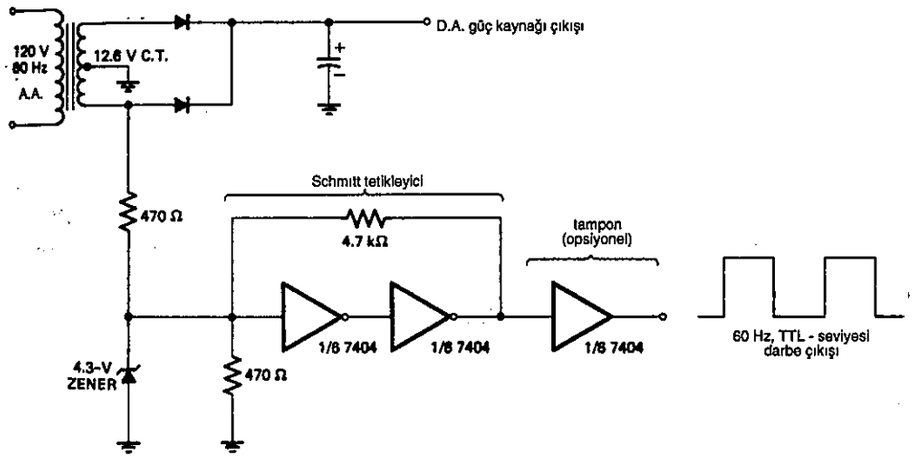
Bir mantık kapısını sürmek için çok küçük olan sinyaller, TTL ve CMOS la uyuşan çıkışları olan National'in LM311 veya LM219 gibi karşılaştırıcılarla mantık seviyelerine yükseltilebilir. 300 MHz e kadar olan yüksek frekans uygulamalarında, ECL çıkışlarına ve 6 ns yayılma gecikmesine sahip *Advanced Micro Devices AM 685* karşılaştırıcısı kullanılabilir. Karşılaştırıcılar aslında açık-döngü modunda çalışan işlemsel kuvvetlendiricilerdir ve bu nedenle çok yüksek kazançları vardır. Karşılaştırıcıların negatif sinyal girişi referans gerilimi olarak toprağa bağlandığında ve pozitif sinyal girişine küçük bir a.a. sinyal uygulandığında (Şekil 2-32a), girişin toprağın üzerine her birkaç milivolt çıkışında, çıkış yüksek mantık seviyesine gidecektir. Giriş sinyalinin referans geriliminin altına her düşüşünde de çıkış alçak mantık seviyesine düşer. 8 numaralı bacak toprağa bağlandığında, LM 319 tipik açık döngü işlemsel kuvvetlendiricinin +V ve -V (+5 V ve -5 V) arasındaki gerilim salınımını vermek yerine standart TTL mantıktaki +5 ve 0 V arasındaki gerilim salınımını verir. O halde, bir karşılaştırıcı, giriş sinyalinin bir referans giriş geriliminin altında veya üstünde oluşunu hisseder ve ona göre çıkış verir. Referans gerilimi sadece sıfır olmak zorunda değildir, ilgili aygıtın çalışma alanının içinde herhangi bir gerilim olabilir. Bu nedenle, Şekil 232*b* de görüldüğü gibi, karşılaştırıcılar bir basınç veya sıcaklık transdüserinin çıkış gerilimi, bir kazanın güvenli çalışma şartları altındaki maksimum değeri olan referans gerilimini aştığında, gerekli uyarıcı mantık seviyelerini üretmekte kullanabilir.



Şekil 2.32 ***(a)*** **Küçük bir sinüs dalgası girişinden mantık seviye sinyalleri üretmek için kullanılan karşılaştırıcı.** ***(b)*** **Bir kazanın basıncının emniyet sınırını aştığını algılamakta kullanılan karşılaştırıcı.**

SCHMITT TETİKLEYİCİ

Diğer bir problem, giriş sinyali uygun gerilim seviyelerine sahipken fakat giriş sinyalinin yükselme ve düşme zamanları TTL için 1 µs den daha uzunsa oluşmaktadır. Eğer giriş gerilimi eşik bölgesinde çok uzun kalırsa, giriş sinyalindeki gürültü çıkışın yüksek ile alçak arasında salınmasına sebep olabilir. Bazı flip-floplarda olduğu gibi a.a. kuplajlı girişlerde yavaş bir geçiş, istenilen iç geçişleri sağlayacak kadar enerji veremiyebilir. Yavaş geçiş sürelerinin tedavisi Schmitt tetikleyici, çıkış yükselme ve düşme zamanlarını pozitif geri besleme kullanarak hızlandıran bir devre, kullanmakla yapılabilir. Schmitt tetikleyici devreleri içinde, hazır kurulmuş olan bir histeresis vardır. Yani giriş sinyalinin pozitife giden geçişlerindeki eşik, negatife giden geçişlerindeki eşikten farklıdır. TTL için girişin pozitife giden geçişindeki eşik tipik olarak 1,7 V tur, fakat negatif geçişte, giriş 0,9 V a ulaşıncaya kadar çıkış değişmeyecektir. Bu nedenle, girişteki küçük bir gürültü gerilimi, girişin yavaş geçişlerinde çıkışın salınmasına sebep olmayacaktır. Bu Schmitt tetikleyici devreleri TTL de 7413 ve 7414 olarak veya CMOS da 4584 ve 4093 olarak numaralandırılmışlardır. 50 Hz lik güç hattından bir mantık seviye saati elde edilişi gibi basit uygulamalarda, Şekil 2-33 de gösterilen ev yapısı Schmitt tetikleyici kullanılabilir. Doğrultulmuş 50 Hz lik sinüs dalgasının yavaş geçişlerini hızlandırmak için kullanılan pozitif geri besleme, çıkışla giriş arasına bağlanan 4,7 KΩ luk dirençle sağlanmıştır. Eğer devrede CMOS ters çeviriciler kullanırsanız, iki Schmitt tetikleyicinin dirençlerini 10 la çarpınız.



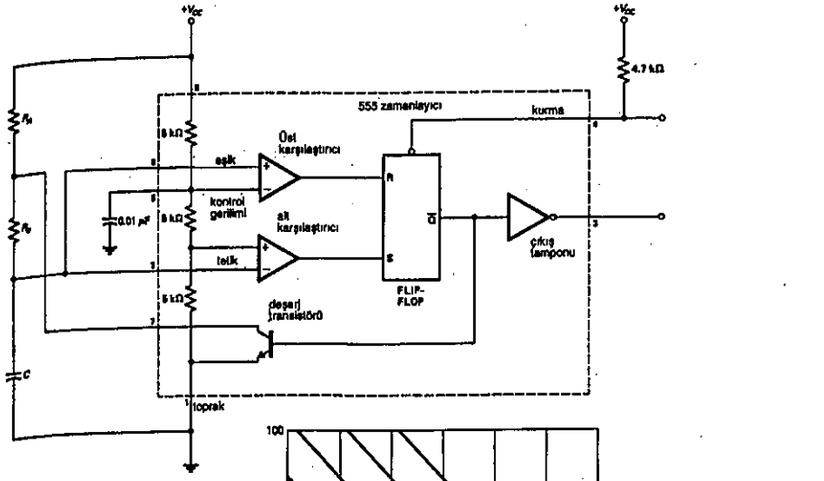
Şekil 2-33 A.a. hattından 50 Hz lik saat üretmek için kullanılan basit Schmitt tetikleyici devresi.

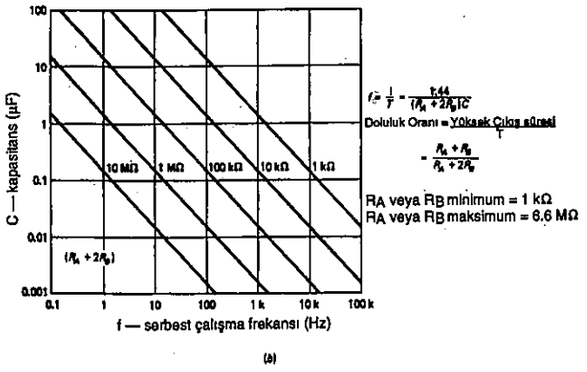
BASİT MANTIK VEYA DARBE KAYNAKLARI

Sayısal devreleri test ederken veya sürerken, temel örnek üzerinde kurulabilecek ufak bir darbe kaynağının bulunması çoğunlukla oldukça faydalıdır.

KARARSIZ MULTİVİBRATÖR DARBE KAYNAĞI OLARAK 555 ZAMANLAYICI

Şekil 2-34*a* daki devredeki *555* zamanlayıcı entegre devre, +5 ve toprak arasına bağlandığında, doğrudan TTL ve CMOS uyuşumlu sürekli bir darbe treni üretecektir.





Şekil 2-34 *(a)* TTL veya CMOS darbe kaynağı olarak kullanılan *LM 555* zamanlayıcı IC. *(b) 555* kararsız multivibratörün *R* ve *C* değerlerini belirlemekte kullanılan grafik *(National Semiconductor).*

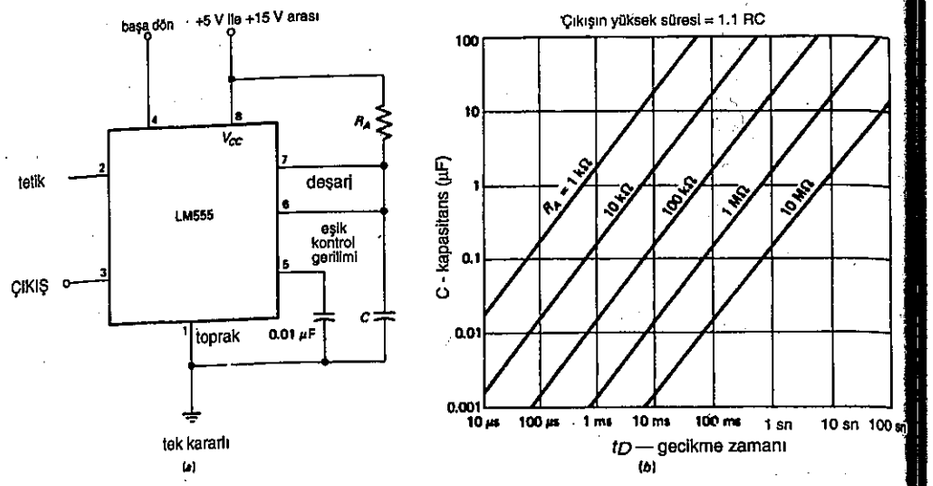
Devrenin çalışması şu şekildedir. Güç uygulandığında 3. bacaktaki çıkış yüksektir. Toprakla 2. bacak arasındaki zamanlama kapasitörü, üzerindeki gerilim 2/3 V_{cc} ye ulaşıncaya kadar, R_A ve R_B seri dirençleri üzerinden şarj olur. İçerideki bir karşılaştırıcı kapasitör üzerindeki gerilimin bu değere ulaştığını hisseder ve daha önceki yüksek çıkışı alçak seviyeye getirir. Böylece, bir transistör iletime geçirilerek kapasitörün sadece R_A üzerinden toprağa boşaltılması sağlanır. İçerideki başka bir karşılaştırıcı kapasitörün ne zaman 1/3 V_{cc} ye düştüğünü hisseder ve çıkışı yeniden yüksek duruma çevirir. Böylece deşarj transistörünü kesime götürerek yeniden şarj çevrimini başlatır. Çıkış darbelerinin frekansı

$$f = \frac{1{,}44}{(R_A + 2R_B)C}$$

formülünden veya Şekil 2-34*b* deki grafikten bulunabilir. Grafiği kullanmak için, istediğimiz frekansı yatay eksen üzerinde seçin. Sonra, var olan bir kapasitör değerini kesinceye kadar bu frekanstan yukarı çıkın ve en yakın köşegenden R_A+2R_B değerini okuyun. Örneğin 100 Hz lik bir frekans ve 0,1µF lık bir kapasitör için R_A+2R_B yaklaşık olarak 100 KΩ a eşittir. Eğer yaklaşık olarak bir kare dalga istiyorsanız, R_A yı R_B den çok küçük seçin. Yukarıdaki örnek için R_A yı minimum değeri 1 KΩ ve R_B yi 47 KΩ seçebilirsiniz. Doluluk oranı, yani çıkışın yüksek olduğu sürenin bir çevriminin toplam süresine bölümü, $(R_A+R_B)/(R_A+2R_B)$ ye eşit olduğuna göre seçilen değerler için çıkıştaki hata sadece %0,5 tir. (R_A+R_B) nin maksimum değeri 6,6 MΩ, ve R_A veya R_B nin minimum değeri 1 K Ω civarındadır. Değişken frekanslı bir çıkış için R_B direnci 1 KΩ ile seri bağlı bir değişken dirençle değiştirilebilir. Zamanlama kapasitörünün değeri birkaç yüz pikofaraddan 1000µF ve üstüne kadar olabilir. Yüksek değerli kapasitörler için alçak kaçak akımı olanları seçin. V_{cc} den toprağa ve 5. bacaktan toprağa olan yangeçit kapasitörleri her zaman gereklidir.

TEKKARARLI MULTİVİBRATÖR DARBE KAYNAĞI OLARAK 555 ZAMANLAYICI

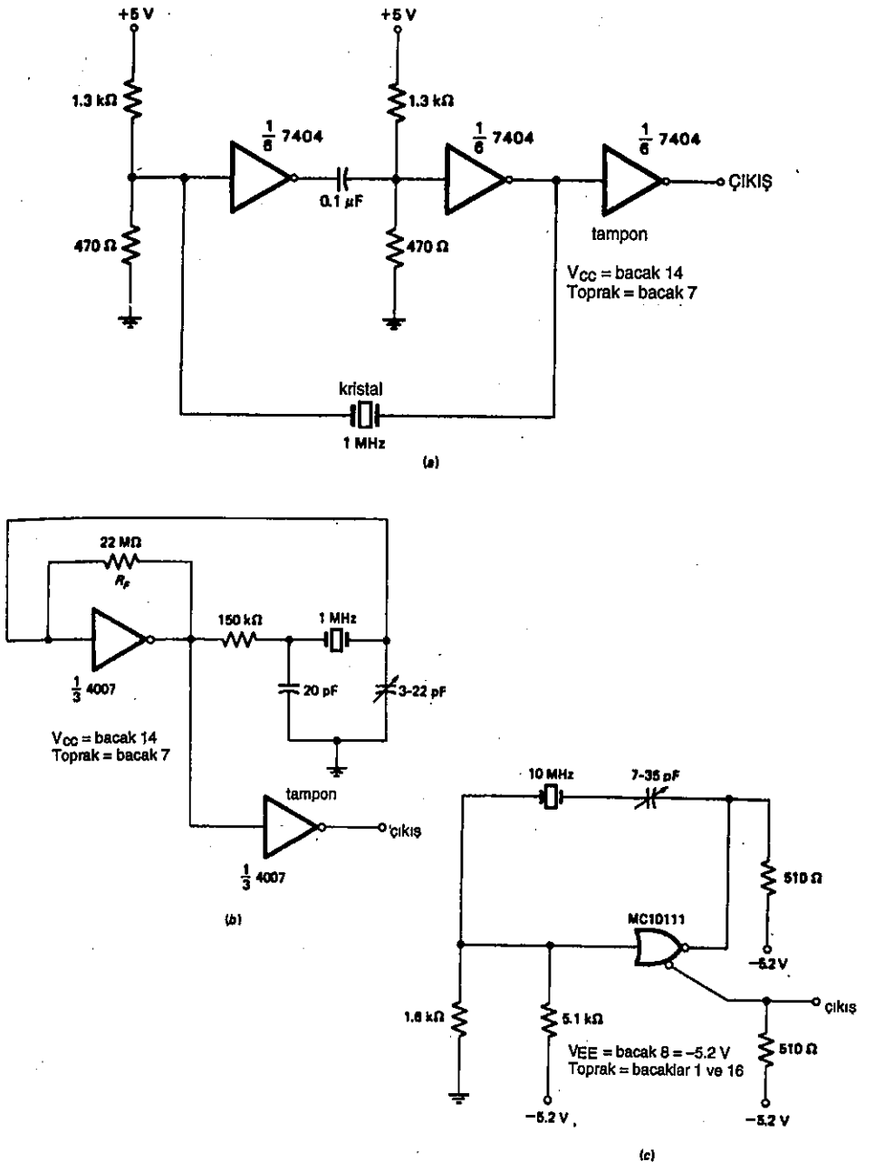
Şekil 2-35*a* da görüldüğü gibi 555 zamanlayıcı entegre devresi, 2. bacaktaki girişin sadece yüksekten alçağa geçişi ile tetiklendiğinde, tek bir çıkış darbesi veren tekkararlı bir multivibratör olarak da kulanılabilir.



Şekil 2-35 *(a)* **Darbe uzatıcı veya tekkararlı multivibratör olarak kullanılan LM 555 zamanlayıcı.** *(b)* **Tekkararlı *555* in *R* ve *C* değerleri için grafik**

Tetikleme sinyalinin genişliği istenilen çıkış-darbesinin genişliğinden daha az olmalıdır. Çıkışın yüksek kalma süresi 1,1 RC ye eşittir. Şekil 2-35, 555 in tekkararlı olarak kullanılması ile ilgili tasarım tablosunu vermektedir. Eğer darbenin yüksek olduğu sürenin 2-3μs den az olmasını istiyorsanız, bu takdirde 74121 veya 74122 TTL tekkararlı multivibratörlerini kullanın. Tekkararlı multivibratörlerin tetikleme giriş uçları, gürültü darbelerince endüklenecek hatalı tetiklemelerden kaçınmak için, kısa tutulmalıdır.

KRİSTAL OSİLATÖRLER *555* zamanlayıcısının üretebileceğinden daha yüksek frekanslara veya daha kararlı frekans darbelerine gerek duyulan uygulamalarda genellikle kristal tabanlı osilatörleri kullanırız. Bunlar standart IC soketlere takılabilen komple paketlenmiş birimler olarak bulunabilmektedir. Fakat elinizde bunlardan biri yoksa, Şekil 2-36 *a,b* veya c deki devrelerden birini kullanabilirsiniz.



Şekil 2-36 *(a)* **TTL kristal osilatör.** *(b)* **CMOS kristal osilatör.** *(c)* **ECL kristal osilatör.** ***(Motorola'nın müsadesiyle)***

Şekil 2.36 *a* 'daki devrenin TTL elemanlarla gerçekleştirildiği durum için, direnç gerilim bölücüleri, ters çeviricileri kazançlarının en yüksek olduğu aktif bölgelerine, yaklaşık 1,35 V, kutuplamaktadır. Her ters çevirici 180° lik bir faz kayması verdiğinden, ikinci ters çeviricinin çıkışı birincinin girişi ile aynı fazdadır. Seri rezonans frekansında, kristal bu frekansta salınımları oluşturmak üzere pozitif geri besleme sağlar. Bu frekans şekildeki devre için 1 MHz dir. Ters çeviricilerin aktif bölgelerinin ortasına kutuplanmış olmalarını sağlamak için her bölücüdeki dirençleri biraz ayarlamak gerekebilir. Alçak empedanslı kristaller en kullanışlı olanlarıdır. Çıkıştaki yüklemelerin osilatör frekansını etkilememesi için üçüncü bir ters çeviricinin tampon olarak kullanılması da iyi bir fikirdir. Daha alçak frekanslar, Bölüm 4 te anlatıldığı gibi 1 MHz lik kararlı kaynaktan sayısal bölücülerle elde edilebilir.

Şekil 2-36*b* deki CMOS osilatör devresi tek bir ters çevirici ile paralel rezonans modunda çalışan kristal kullanmaktadır, çünkü +5 V luk kaynakla çalıştırıldığında seri bağlı iki CMOS ters çeviricinin yayılma gecikmesi 1 MHz lik bir osilatör yapmak için çok uzundur. R_F, kapı çıkışını aktif bölgeye kutuplar. Osilatör çıkışına bir tampon kapı konulması yine önerilir.

Şekil 2.36*c* deki osilatör devresinde standart MECL 10.000 serisi VEYA DEĞİL kapısı kullanılarak ECL çıkış seviyelerinde çalışan yüksek-frekans osilatörü oluşturulmuştur. Aktif bölgeye kutuplama 1,6 ve 5,1 kΩ luk dirençlerle gerçeklenmiştir. Daha önce olduğu gibi, kristal, salınımları üretmek için geribesleme sağlar. 510 Ω luk dirençler MECL emitör izleyici çıkışlarını aşağı-çekme dirençleridir.

ÇIKIŞ SİNYALİNİN KULLANILIŞI

Bir kapının çıkışından diğer bir kapının girişine olan kısa mesafeli bağlantılarda, basit bir bağlantı teli veya baskılı devre üzerindeki yollar hiçbir problem olmadan sinyali yerine ulaştıracaktır. Ancak süren ve sürülen kapılar arasındaki mesafe belli bir uzunluğu aştığında (bu değer her mantık ailesi için farklıdır), bağlantı teli, koaksiyel kablo veya baskılı devre yolu bir iletim hattı gibi görülmeli ve ona göre dikkate alınmalıdır. Bu bağlantılar iletim hattı gibi ele alınmazsa yansıma problemleri fazladan giriş sinyali geçişlerine sebep olabilir.

İLETİM HATLARI

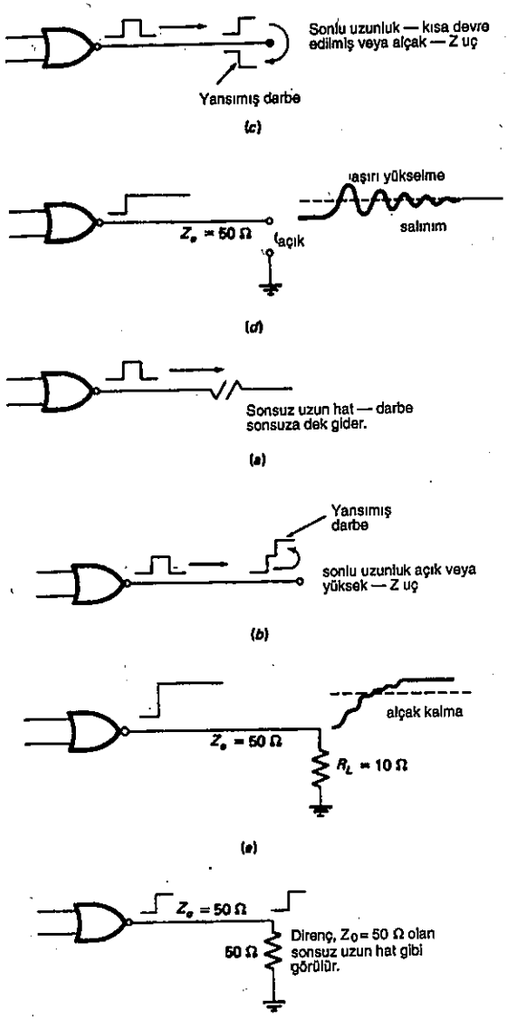
Bir iletim hattı onu süren kapıya, hattın karakteristik empedansı olan, Z_0 lık bir yük olarak görülür. Hattın birim uzunluğunun gösterdiği endüstans L_0, birim uzunluktaki iki iletken arasıdaki kapasitans C_0 olmak üzere, karakteristik empedans Z_0 ın değeri $\sqrt{L_0/C_0}$ eşittir. L_0 ve C_0 birim uzunluk için ölçüldüğünden, hattın uzunluğu 5m de olsa 5000 m de olsa Z_0 ın değeri değişmez.

Toprak üzerindeki bir tel için veya iki açık tel için Z_0 30 ile 600 Ω arasında olacaktır. Koaksiyel kablolar 50, 62, 75, 93 Ω gibi sabit empedans değerleri ile üretilirler. Sık kullanılan *RG-58AU* koaksiyel kablosunun Z_0'ı 50 Ω dur. Baskılı devre kartları üzerinde homojen alçak-empedans iletim hatları, kartın bir yüzünde mümkün olan yerlerde bakır iletkeni toprak düzlemi oluşturacak şekilde bırakarak ve kartın diğer tarafındaki sinyal hattının genişliğini ayarlayarak elde edilebilir. İki iletken arasındaki fiberglass epoksi malzeme dielektrik görevini görür. Örneğin 0,062 inç (0,16 cm) kalınlığındaki *fiberglass, epoksi* kartı üzerinde 50 Ω luk bir iletim hattı oluşturur. Çok katlı baskılı devre kartlarında, kartın üstünde ve altında iletim hatlarının oluşmasını sağlamak üzere toprak düzlemi orta kata konulur. Bir sinyalin iletim hattı üzerindeki hızı dielektrik malzemeye bağlıdır ve 6 ile 8 inç/ns (15,2 ile 20,3 cm/ns) arasındadır. (Karşılaştırma için ışık hızının 11,8 inç/ns (30,0 cm/ns) olduğunu hatırlayın).

Sonsuz uzun bir iletim hattının bir ucundan Şekil 2.37 *a* daki gibi bir darbe konulursa, darbe hat boyunca ilelebet ilerleyecektir. Eğer hat sonsuz uzunlukta değilse ve kaynaktan uzakta olan ucu açıksa, darbe hattın yüksek empedans gösteren açık ucunda yük biriktirecektir ve kaynağa aynı fazda bir darbe yansıtacaktır (Şekil 2.37 *b*). Bunun oluşumu şöyledir. Şekil 2.37 *b* de görüldüğü üzere, açık uçta, yansıyan darbe ile gelen darbe, gelen darbe süresince üst üste binecek gelen darbenin genliğinin iki misli bir gerilim oluşturacaktır. (Sanki yansıyan darbe gelen darbenin üstüne tırmanıp sırtından yuvarlanıp inmekte ve sonra da iletim hattı boyunca hareket ederek yola ilk çıktığı andaki faz ve genlikle kaynağa dönmektedir).

Eğer kaynaktan uzakta olan hattın ucu kısa devre edilirse, gönderilen darbe ile faz farkı olan bir yansımış darbe üretilir (Şekil 2.37 *c*). Yansımış darbenin fazı 180° kaymış olmak zorundadır, çünkü kısa devre noktasında, gelen ve giden gerilimlerin toplamı sıfır olmalıdır. Yansıtılmış bir darbe kaynağa geri dönecektir. Eğer kapının çıkış empedansı hattın karakteristik empedansına eşit değil ise darbe tekrar hattın sonuna yansıyacaktır. Şekil 2.37 *d* bu ileri geri zıplamanın, açık uçlu bir kablonun alma ucundaki darbe dalgaşekli üzerindeki etkisini göstermektedir. Gördüğünüz gibi dalgaşekli önce aşırı yükselir sonra istenilen tek bir alçaktan yükseğe geçiş yerine mantık eşik gerilimini defalarca geçecek şekilde salınır. Şekil 2.37*e* ucuna alçak bir empedans bağlanmış hattın sonunda görülen dalga şeklini göstermektedir. Gördüğünüz gibi alçaktan-yükseğe geçişin ilk kısmı aşırı alçak kalır ve eşik geriliminin üstüne çıkabilir veya çıkamayabilir.

Bu yansıma problemlerini önlemenin kolay bir yolu giden darbeyi son-

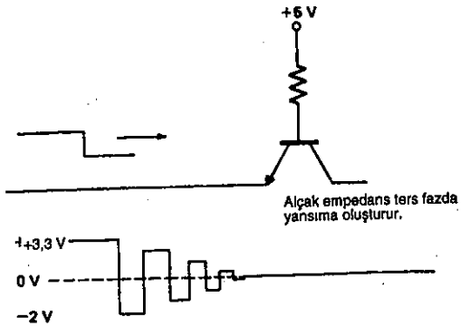


Şekil 2.37 *(a)* İletim hattına uygulanan darbe. *(b)* İletim hattının açık-devre ucundan aynı fazda yansıyan darbe. *(c)* İletim hattının kısa-devre edilmiş ucundan zıt fazlı yansıma. *(d)* Birçok yansımadan sonra aşırı yükselme ve salınımları gösteren iletim hattının açık-devre ucundaki sinyal *(e)* Z_0 dan küçük bir *R* ile sonlandırılmış iletim hattının ucunda görülen sinyal. *(f)* Z_0 a eşit *R* ile sonlandırılan iletim hattının ucunda görülen sinyal.

suz bir hat üzerine gittiğine inandırmaktır. Bunu, hattın karakteristik empedansına eşit değerde bir direnci hattın sonuna bağlayarak yapabilirsiniz (Şekil 2.37 *f*). Direnç darbenin enerjisini dağıtarak yansımaları önler. Bu yönteme **paralel sonlandırma** denir ve mümkün olduğu yerlerde en iyi çözümdür. Örneğin, 50 Ω çıkışlı bir darbe üreteci kullanıyorsanız, ona 50 Ω luk bir koaksiyel kablo bağlamalısınız ve kablonun sonunu 50Ω luk bir dirençle sonlandır-malısınız. Özet olarak: Z_0 dan büyük bir dirençle sonlandırılmış bir hattın çıkışı aşırı yükselme, Z_0 dan küçük bir dirençle sonlandırılmış bir hattın çıkışı aşırı alçalma gösterir; fakat Z_0 a eşit bir dirençle sonlandırılmış bir hat, bir darbeyi yansıma problemleri olmadan iletir.

Şimdi iletim hatları hakkında bir şeyler bildiğinize göre, herhalde bir sonraki sorunuz, "Bütün bunlara üzülmeme gerek kalmadan kullanabileceğim bağlantı telinin uzunluğu nedir?" olacaktır. Bunun cevabı şöyledir: Kapının çıkış yükselme zamanı hat boyunca olan yayılma gecikmesinin 3-4 katı ise hiç bir probleminiz olmaz. 15 ns lik yükselme zamanı olan TTL için, 4-5 ns yayılma gecikmesi olan bir hat kullanabilirsiniz. Sinyal 7 inç/ns (17,8 cm/ns) hızla hareket ettiğine göre toprak düzleminin üzerinde olan 28-35 inç (0,7-0,9 m) uzunluğunda bir teli problemsiz kullanabilirsiniz. Toprak düzlemi olmazsa açık-tel uzunluğu 12-14 inç (30,5-35,6 cm) den fazla olmamalıdır. Birkaç yüz nanosaniyelik yükselme zamanı olan 5V luk CMOS lar için uzaklık 20-30 ft (6,1-9 m) yi bulmaktadır. Fakat 1 ns lik yükselme zamanı olan MECL II için, problemler başlamadan önceki maksimum açık-tel uzunluğu yaklaşık 1 inç (2,5cm) ye düşmektedir. Neyse ki, ECL çıkışları paralel sonlandırma dirençlerini sürebilmektedir. Daha önce görüldüğü gibi, -5,2 V ta çalışan ECL ler, ya hattan -2 V a 50 Ω luk bir direnç bağlayarak veya toprağa 82Ω ve -5,2 V a 130Ω bağlayarak yapılacak gerilim bölücüyle 50 luk hatlara paralel sonlandırılabilir. Voltaj bölücünün, kaynağın yani Thevenin empedansının 82 Ω ile 130 Ω un paralel dengi olan yaklaşık 50 Ω ile yaklaşık -2 V üretmesinden dolayı 2. method çalışır.

Şimdi, Şekil 2-17 *a* daki bir TTL kapısının girişlerinde bulunan *D1* ve *D2* kelepçe diyotlarının amacı açıklanabilir. Kelepçe diyotları olmayan standart bir TTL kapısının girişine gelen yüksekten alçağa geçen bir darbe, alçak bir empedans görür ve Şekil 2-38 deki dalgaşeklini vermek üzere ileri geri yansır. Gördüğünüz gibi dalgaşekli, sonunda alçak seviyeye ulaşmadan önce tekrar eşiği geçerek yüksek seviyeye ulaşabilir. Şekil 2-17 *a* daki *D1* ve *D2* kelepçe diyotları TTL girişlerinin yaklaşık 1 V tan daha fazla negatife gitmesini engeller. Bu gerilimin altında kelepçe diyotları darbenin enerjisini harcamak üzere iletir ve yansımanın genliğini dönüşte eşik geriliminin yakınına gelmeyecek şekilde azaltır.

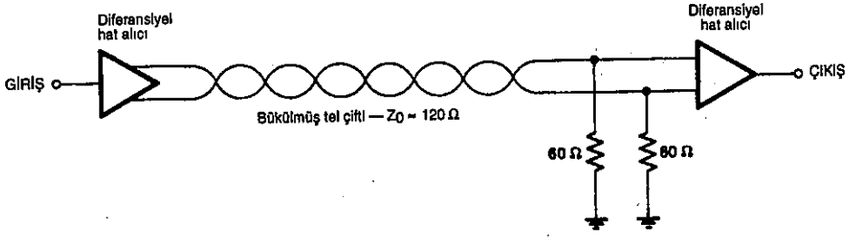


Şekil 2.38 Kelepçe diyotları olmadan TTL girişinde alınan sinyal

HAT SÜRÜCÜLERİ VE ALICILARI

CMOS ve TTL gibi çok kullanılan mantık ailelerinin standart çıkışları bir hattın ucundaki paralel sonlandırmanın rezistif yükünü süremez. Bu ailelerde uzun mesafelere sinyal göndermek için Signetics N8T13 gibi özel tasarlanmış çıkış sürücüleri kullanmak en iyisidir. Bu sürücüler TTL giriş seviyelerini kabul ederler ve sonunda sonlandırma direnci olan 50 Ω veya daha yüksek empedanslı hatları sürmeye yeterli akım verirler. Karşılık gelen hat alıcısı Signetics N8T14 tür. Hat alıcısının sonlandırmayı yüklememek için yüksek giriş empedansı vardır.

Sinyaller çok gürültülü ortamlarla uzun mesafelere iletilmek istendiğinde, 75110 gibi tümleyici çıkışlı diferansiyel hat sürücüleri, bir hattı veya bir bükülü tel çiftini sürmek için kullanılır. Sinyal, diferansiyel hat alıcısıyla Şekil 2.39 da görüldüğü gibi alınır. Bağlantı sağlayan bükülü tel çiftinin her iki teli

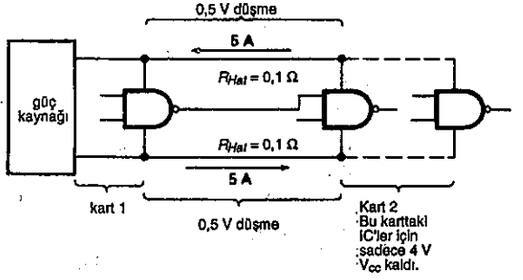


Şekil 2.39 Veri iletimi için bir diferansiyel hat sürücü ve alıcının kullanımı

de aynı gürültülü ortamdan geçtiğine göre, her birinde eşit miktarda gürültü indüklenir. Bu ortak mod gürültü, alıcının diferansiyel girişi tarafından reddedilir. İki sonlandırma direncinin, bükülü tel çifti arasında seri olduğuna ve toplamlarının hattın Z_0 ına eşit olduğuna dikkat edin.

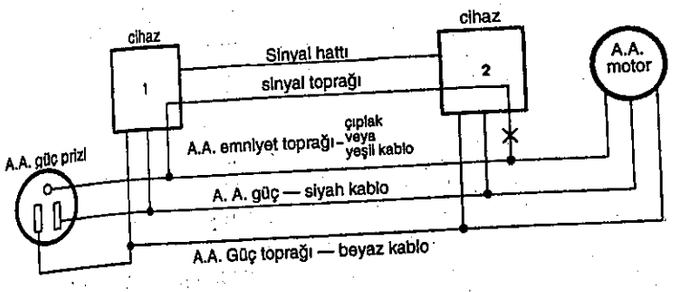
GÜÇ VE TOPRAKLAMA PROBLEMLERİ

Karşılaşılan iki ana topraklama problemi uzun tellerdeki IR gerilim düşmeleri ve mantık toprağı ile a.a. güç toprağının birleştirilmesiyle hatlarda oluşan gürültülerdir. Bunlardan ilki Şekil 2.40 ta gösterilmiştir. IR gerilim düşmeleri ikinci karttaki IC ler üzerinde sadece 4 V bırakmaktadır ki, eğer bu IC ler TTL ise, onları çalıştırmak için bu gerilim yeterli olmayacaktır. Bu, Bölüm 1 de uyardığımız papatya-zinciri bağlantının örneklerinden biridir. Problem, dallandırılmış bağlantı ile veya her kart üzerinde bir gerilim regülatörü ile +5 V üreterek ve her karttan tek bir ortak toprak noktasına ayrı ayrı kalın toprak hatları bağlamakla çözülebilir (Şekil 1. 4).



Şekil 2.40 IR gerilim düşmeleri mantık devrelerinde nasıl hatalı çalışmaya sebep olmaktadır.

Sinyal toprak hatlarında a.a. gürültü akımları tarafından oluşturulan gürültü problemleri ise sistemin tamamının tek bir noktada a.a. güvenlik toprağına bağlanması sağlanarak yok edilebilir. Şekil 2.41 deki devreye bakın. İkinci cihazın sinyal toprağı ile a.a. güvenlik toprağı arasındaki bağlantı X noktasında kesilmediği sürece, güvenlik toprağı üzerindeki gürültü gerilimleri sinyal toprağı üzerinde de görülecektir. Sinyal toprağı üzerindeki aşırı gürültü ikinci cihazdaki mantık kapılarının çalışmasını engelleyebilir.



Şekil 2.41 Toprak-halkası problemi

MANTIK AİLELERİNİ BİRBİRLERİNE BAĞDAŞTIRMA

Farklı mantık ailelerini bağdaştırırken her ailenin hem gerilim seviyelerini hem de akım gereksinimlerini dikkate almalısınız. Tablo 2.2. bu konuda size yardımcı olacaktır.

TTL ALTAİLE ARABAĞLANTILARI

Farklı TTL altaileleri arasındaki arabağlantılar kolaydır, çünkü çıkış gerilim seviyeleri uyuşmaktadır. Çıkış yelpazesi sınırlarının aşılmamasına dikkat edilmelidir. 74LS00 ve 74L00 ailelerinin sadece iki standart TTL girişi sürebildiklerini unutmayın. Farklı ailelerin kapı girişlerinden oluşan bir yükünüz varsa, maksimum alçak giriş akımlarını toplayın ve toplamın, sürücü kapının alçak çıkış akım çökme sınırını aşmadığından emin olun. Örneğin bir 74LS00 çıkışının bir 7400 kapısı ile beş 74LS00 kapısının girişini sürüp süremeyeceğini görmek için Tablo 2.2 yi kullanarak maksimum mantık 0 giriş akımlarını bulun. Bu akım bir 7400 için 1,6 mA ve beş 74LS00 kapısı için 5x0,36 mA=1,8 mA dir. Toplam giriş yük akımı 3,4 mAdir. Bu da 74LS00 çıkışının kaldırabileceği 4 mA dan daha az olduğundan 74LS00 bu kapıları sürebilecektir.

TTL DEN CMOS A VE CMOS TAN TTL YE

TTL ailelerini 5V ta çalışan CMOS a bağladığınızda, TTL nin çıkışının CMOS için gerekli yüksek seviyeye ulaşmasını sağlamak için genellikle 10 K Ω luk bir yukarı-çekme direnci TTL nin çıkışına bağlanır. CMOS çıkışlarının TTL ailesinin girişlerine bağdaştırılması için TTL tarafından alçak çıkış sürme kapasitesinin dikkatle incelenmesi gerekir. Örneğin 5 V kaynakla çalışan

CD4000A VEYA DEĞİL kapısı, alçak çıkış durumunda 0,4 mA lık minimum akımı (en-kötü-hal akımı) çöktürebilir. Bu iki alçak-güçlü TTL girişini veya bir alçak-güçlü Schottky TTL girişini sürmeye yeterli olacaktır. Ancak 5 V luk kaynakla çalışan CD4011AE VE DEĞİL kapıları 0,12 mA lık minimum alçak çıkış akımı çöktürebilir ve bu bir tek alçak-güçlü TTL girişini bile alçağa sürmeye yeterli olmayacaktır. Bunun gibi küçük problemler temel örnekte gözden kaçırılabilir ve cihazın üretimine geçilinceye kadar ortaya çıkmayabilir, çünkü sadece biraz daha fazla sürme gücü olan entegre devreler çalışmakta, diğerleri çalışmamaktadır ve temel örnekte tesadüfen güçlü entegrelerden biri kullanılmıştır. Bu, devre tasarımcısının kuralını ispatlar: Bir yonganın arasıra bir işi başarması, bütün yongaların her zaman o işi başarmayacağını gösterir.

4000B, 74C00 ve 4500 serisi CMOS ların hepsi bir alçak güçlü Schottky TTL veya iki alçak-güçlü TTL yükü sürmek üzere standart hale getirilmişlerdir. Eğer şüphedeyseniz veya daha fazla sürmeye ihtiyacınız varsa, pakette 6 tane olarak gelen ve her biri iki standart TTL girişi sürebilen CD4049 ters çeviren veya CD4050 ters çevirmeyen tamponlar kullanın. +10 V veya +15 V kaynakla çalışan CMOS arasında her iki yönde bağdaştırma sağlamak için 74C901 ters çeviren veya 74C902 ters çevirmeyen tamponlar (pakette 6 kapı vardır) kullanılabilir.

TTL DEN PMOS VE PMOS DAN TTL YE

Yüksek-eşikli PMOS çıkış gerilimi tipik olarak toprak ile -13 V civarında bir negatif gerilim arasında salınır. Daha yeni alçak-eşikli PMOS çıkışları +5 V luk yüksek ile -12 V luk alçak arasında salınır. TTL veya 5 V luk CMOS seviyelerini yüksek-eşikli PMOS seviyelerine bağlamak için National Semiconductor DS7800 gibi bir seviye çevirici/tampon kullanabilirsiniz. Bazı eski alçak-eşikli PMOS devreleri TTL seviyelerini kabul eden veri girişlerine sahiptir, ancak saat girişleri +5 ile -12 V arasında salınım gösteren dış sürücüler gerektirir.

Daha sonraki alçak-eşikli PMOS devrelerinin girişleri bir yukarı-çekme direnciyle TTL tarafından, veya +5 V luk CMOS ile sürülebilmektedir ve çıkışları bir standart TTL girişi sürebilir. Bir PMOS, çıkışı -12 V a çekmeğe çalışacaktır, fakat TTL nin kelepçe diyotları veya CMOS un statik diyotları, çıkışın -1 V un altına düşmesini engeller. PMOS çıkışı diyotlara zarar verecek kadar akım çöktüremediği için bu kabul edilebilir.

NMOS DAN TTL YE DE TTL DEN NMOS A

Çoğu NMOS, ister +5 V ve toprakla ister +5 ve -12 V ile beslensin, standart TTL veya 5 V luk CMOS seviyelerini kabul edecek şekilde tasarlanmış girişlere sahiptir. Eğer bir TTL diğer bir TTL ile bir NMOS girişini sürüyorsa,

yüksek girişin iyi bir gürültü payı için yeterince yüksek olmasını sağlamak üzere, bir yukarı-çekme direnci +5 V a bağlanmalıdır. Çoğu NMOS un çıkışları bir standart TTL girişini, dört alçak-güçlü Schottky TTL girişini veya sekiz alçak-güçlü TTL girişini sürebilir.

TTL DEN ECL YE VE ECL DEN TTL YE

TTL, CMOS veya NMOS tan ECL seviyesine çevirme 10124 dörtlü TTL-den-ECL-ye çevirici ile kolayca yapılır. TTL çıkışları çeviricinin girişine doğrudan bağlanabilir, ancak CMOS ve NMOS çıkışları ile çeviricinin girişi arasına bir tampon gerekir, çünkü çeviricinin girişlerinden 3,2 mA e kadar akım çökme gereksinimi vardır.

ECL den TTL, CMOS veya NMOS seviyelerine çevirim için 10125 dörtlü ECL-den-TTL-ye çevirici kullanılabilir. Eğer bir TTL ve NMOS karışımı sürüyorsanız, çeviricinin çıkışı ile +5 V arasına bağlanacak 10KΩluk bir yukarı-çekme direnci uygun MOS yüksek seviyesine ulaşılmasını garantiler.

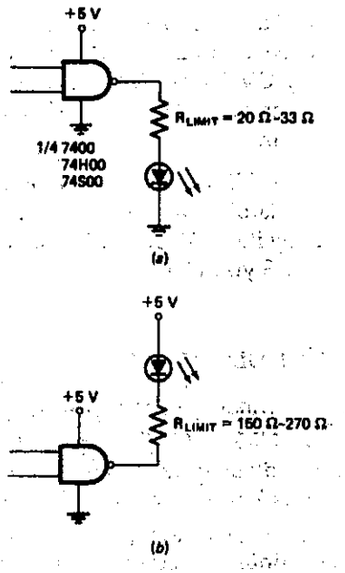
TTL DEN I^2L YE VE I^2L DEN TTL YE

I^2L nin çıkış yapısı açık kollektörlü TTL ye benzediğinden çıkışla +5 V arasına bir direnç bağlanarak NMOS, CMOS veya TTL çıkış seviyeleri elde edilebilir. Eğer enjekte edilen akım alçak ise, çıkış transistörü doğrudan bir TTL'yi sürecek kadar akım çöktüremeyecektir. Bir CMOS tampon iyi bir bağlantı aracıdır. TTL, NMOS veya CMOS çıkışları I^2L girişlerini doğrudan sürebilir, fakat giriş transistörünün baz-emitör ekleminden aşırı akım geçmesini önlemek için bir seri akım-sınırlayıcı direnç kullanılmalıdır. Çoğu I^2L devrelerinin giriş ve çıkışları TTL ve CMOS ile doğrudan uyuşmaktadır.

MANTIK KAPILARINI BASİT GÖSTERGE VE RÖLELERLE BAĞDAŞTIRMA

LED LER Çoğu zaman bir mantık kapısının çıkış durumunu bir osiloskopa veya DVM ye bağlanmaya gerek kalmadan görmek isteyebilirsiniz. Eğer uygun bir şekilde bağlanmışsa basit bir ışık-yayan-diyot *(Light-emitting-diode, LED)* bunu yapabilir. Bir LED, genellikle galyum arsenat fosfattan yapılmış, ileri yönünde akım geçirildiğinde görülür ışık yayan bir diyottur. Verilen ışığın rengi kullanılan malzemeye bağlıdır ve kırmızı, turuncu, sarı veya yeşil olabilir. Diyot üzerinde ileri yöndeki gerilim düşümü 20 mA de, bir kırmızı LED için 1,5 V tan bir yeşil LED için 3 V a kadar değişir.

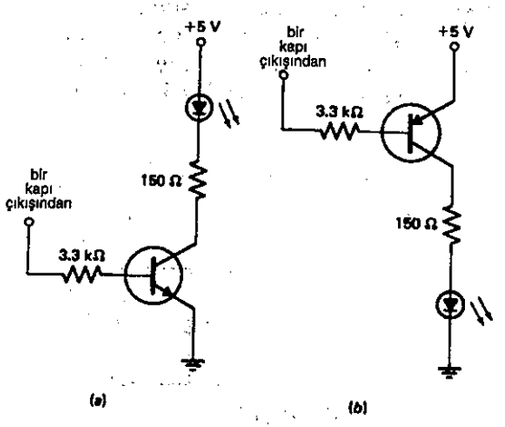
Standart TTL, Schottky TTL veya yüksek-güçlü TTL, bir LED i yakacak kadar akım verebilir. Eğer bir kapının çıkışının ne zaman yüksek olduğunu göstermek isterseniz, Şekil 2.42a daki devre bunu yapar. Bir alçak çıkışı göstermek için, Şekil 2.42b deki devre kullanılabilir. Akım sınırlayıcı dirençler kapı çıkışının veya LED'in aşırı ısınmasını engellemek için konmuştur.



Şekil 2.42 *(a)* Çıkışı yüksekken bir *LED* i yakacak şekilde bağlanmış standart TTL kapısı. *(b)* Çıkışı alçakken bir *LED* i yakacak şekilde bağlanmış standart TTL kapısı.

Bir LED i alçak-güçlü Schottky TTL, alçak-güçlü TTL, CMOS veya NMOS la sürmek için en kolay yol Şekil 2.43 *a* veya Şekil 2.43*b* gösterildiği gibi bir sürücü transistör kullanmaktır. Transistörün bazındaki akım sınırlayıcı dirençlere dikkat edin. Kollektör direncinin değeri LED üzerinde gerilim düşümünü (20 mA lik kırmızı LED için yaklaşık 0,5 V) ve transistörün doyma gerilimini (20 mA de yaklaşık 0,5 V) 5V luk V_{cc} değerinden çıkartılarak ve bu gerilim farkı düşünülen LED akımına bölünerek bulunur. Yukarıdaki 20 mA lik akım ve 3 V luk dirençteki gerilim düşümü 150Ω luk bir direnç değeri verir.

Bazdaki akım sınırlayıcı direncin değeri kritik değildir: 1 ile 10 KΩ arası uygundur. Doğrudan bir LED i süren bir kapının çıkışının (Şekil 2.42) genellikle normal mantık seviyelerine ulaşmayacağını unutmayın. Transistörle sürülen LED gösterge ise kapı devresinin çıkışını yüklemeyecektir.



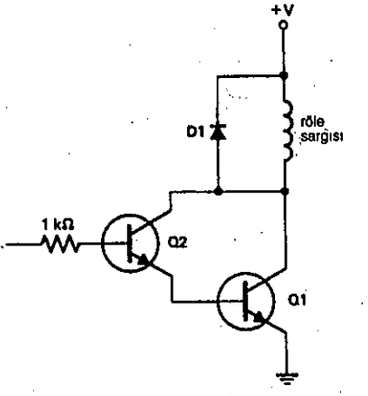
Şekil 2.43 ***(a)*** **Yüksek çıkışı algılamak için transistör LED sürücüsü.** ***(b)*** **Alçak çıkışı algılamak için transistör LED sürücüsü.**

AMPÜLLER

Küçük alçak-gerilimli akkor filamanlı ampüller Şekil 2.43*a* ve *b* deki devrelerle sürülebilirler. 6 veya 4,5 V luk ampüllerde kollektör akım sınırlayıcı direnci kullanılmayabilir. Şekil 2.43 *a* dakine benzer bir devre bir neon ampülü sürmekte kullanılabilir. Bu ampüller yanmak için 55 ile 150 V arasında gerilim gerektirirler. Bu nedenle sürücü transistörün kollektör-emitör ve kollektör-baz bozulma gerilimleri bu uygulanan yüksek gerilimlerden büyük olmalıdır. Akım-sınırlayıcı kollektör direnci çok daha yüksek bir değere, genellikle 100 KΩ civarına, artırılmalıdır.

MEKANİK RÖLELER

Mantık seviye çıkışlarıyla birkaç ampere kadar olan doğru akımları kontrol etmek isterseniz, Şekil 2.43 *a* ve *b* deki transistörler Darlington transistörleriyle değiştirilmelidir. Alternatif akımları veya daha büyük doğru akımları kontrol etmek için ya mekanik veya katı-hal rölesi kullanın. Bir mekanik rölenin sargısından bir kontrol akımı geçtiğinde açmak veya kapatmak üzere çeken bir elektromanyet oluşur. Şekil 2.44 teki devre TTL, CMOS veya NMOS mantık seviyelerinden 200 mA e kadar sargı akımı gerektiren alçak-gerilimli röle sargısını sürmek için kullanılabilir. Röle sargısının uçları arasındaki D1 diyodu, sargıdaki akım kesildiğinde oluşan "tepme" yani ters (elektromotif kuvvet) geriliminin transistörleri yakmasını engellemek için mutlaka gereklidir. Röle kontakları mekanik anahtarlar olduklarından, basit transistör devreleriyle kontrol edilemeyen 120 V veya 240 V a.a da çalışan ışıkları, ısıtıcıları, motorları, v.b. ni çalıştırmak için kullanılır.



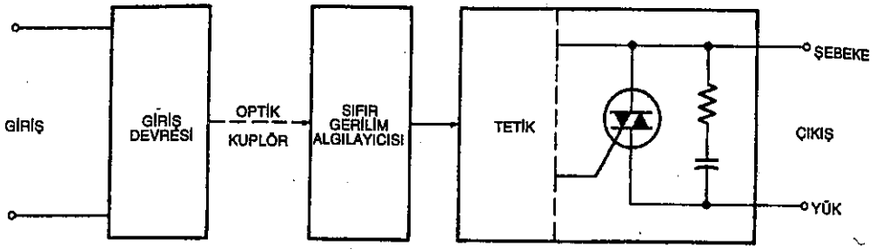
Şekil 2.44 Röle sargısı için transistör sürücü devre.

YARI İLETKEN RÖLELER

Mekanik rölelerin kontak uçlarının ark yapma ve sonunda yanıp iletim sağlamama problemleri vardır. Bu ve diğer röle problemleri, büyük akımları anahtarlamada kontak uçları yerine Triyak anahtarlar kullanan *katı-hal* röleler kullanılarak çözülebilir. Anahtarlama bir yarı iletken aygıtın içinde yapıldığından ark oluşmamaktadır. Ayrıca birçok *katı-hal* rölelerin, triyak anahtarının a.a. güç girişinin sıfır-geçiş noktalarında tetiklenmesini sağlayan iç sıfır-gerilim detektörleri vardır. Sıfır-geçiş noktalarında anahtarlama büyük akımların hızlı bir şekilde kesilip verilmeleri sırasında oluşan *radyo-frekans girişimlerini (Radio-frequency Interference: RFI)* en aza indirger. Çoğu katı-hal rölelerinin TTL ile bağdaşan girişleri vardır ve bazıları, triyak kontrol devresine optik kuplaj sağlayan giriş devresine bile sahiptir. Bu optik kupleli devre-

lerde, giriş akımı bir iç LED i yakar. LED in ışığı, triyak anahtarını tetikleme devresine bağlı fototransistörün bazını sürer. Sinyali ışık ilettiği için, optik kuplaj giriş ve kontrol devresi gerilimlerini tamamen birbirinden ayırır. Şekil 2.45, optik kupleli ve sıfır-geçiş noktası anahtarlamalı bir yarı-iletken rölesinin blok şemasını göstermektedir.

Optik kuplörler Monsanto MCT277 gibi ayrı birimler halinde de bulunabilir. Bunlar bir mantık sinyalini iletmek, fakat gerilim kaynaklarını ayrı tutmak istediğinizde kullanılabilir.



Şekil 2.45 Optik kupleli katı-hal rölesi (DEC).

ÖNEMLİ BİR NOKTA

Bu bölümde çok kullanılan birçok mantık ailesinin temel çalışma şeklini, özelliklerini ve kullanışlarını tanıdınız. Verilen bilginin çok geniş kapsamlı olması ve Tablo 2.2 de karşılaştırma amacıyla verilen bazı parametre değerlerinin aile elemanları arasında çok fazla değişiklikler göstermeleri dolayısıyla bu bölümün sadece bir başlangıç olduğunu söyleyebiliriz. Çalışmayı düşündüğünüz mantık ailesine ait üreticinin veri kitapları ve uygulama notlarını elde etmelisiniz. Bu veri kitaplarındaki ve uygulama notlarındaki bilgilerin çoğunu artık anlayabilecek durumdasınız.

GÖZDEN GEÇİRME SORULARI VE PROBLEMLER

1. Aşağıdaki miktar veya aygıtların her birinin analog mu yoksa sayısal mı olduğunu belirtin:

   a. Bir arabanın hızı
   b. Bir atomun etrafındaki elektronların enerji seviyeleri
   c. Zaman
   d. Okyanustaki su moleküllerinin sayısı
   e. Soket
   f. Transformatör çıkış gerilimi

2. Aşağıdaki onluk sayıları ikilik sayılara çevirin.

   a. 10 b. 72 c. 4096 d. 0,704 e. 0,35

3. Aşağıdaki ikilik sayıları onluk sayılara çevirin.

   a. 110
   b. 1010101
   c. 1111
   d. 0,1101
   e. 0,011

4. Aşağıdaki kapıların doğruluk tablosunu kurunuz ve boole ifadelerini yazınız.

   a. Ters çevirici
   b. İki-girişli VE
   c. Üç-girişli VEYA
   d. İki-girişli VE DEĞİL
   e. Dört-girişli VEYA DEĞİL
   f. İki-girişli özel VEYA
   e. İki-girişli özel VEYA DEĞİL

5. Aşağıdaki cümleleri tamamlayın.

   a. Bir girişi alçağa bağlanmış bir VE kapısının çıkışı mantık olarak ..... .

   b. Her iki girişi alçağa bağlanmış bir VEYA DEĞİL kapısının çıkışı mantık olarak ..... .

c. Bir girişi alçağa bir girişi yükseğe bağlanmış olan bir özel VEYA kapısının çıkışı mantık olarak ..... .

6. Tipik bir 16 bacaklı çift sıralı IC paketinin üstten görünüşünü çizin. Bacakları numaralandırın. Bir DIP aygıtta 1 numaralı bacağı nasıl tanırsınız?

7. a. Standart *TTL* için maksimum V_{cc} gerilimini ve V_{cc} gerilimi 5 V iken maksimum giriş gerilimini söyleyiniz.

   b. 7400 serisi *TTL* ile 5400 serisi *TTL* arasındaki fark nedir?

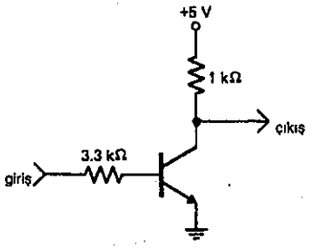
8. Bir standart *TTL* kapısı için en-kötü-durum alçak ve yüksek giriş gerilimlerini, alçak ve yüksek çıkış gerilimlerini ve tipik yayılma gecikmesi zamanını söyleyin.

9. Gürültü payını tanımlayın. Bir *TTL* kapısı için gürültü payları nelerdir?

10. Bir standart *TTL* girişinde alçak giriş oluşturacak gerekli maksimum akım nedir? Yüksek giriş için ne kadar akım gerekir?

11. Standart bir *TTL* kapısının çıkış yelpazesi nedir? Bu çıkış yelpazesini aşmanın etkileri nelerdir?

12. Şekil 2.46 da görülen devreye 3,6 V luk bir mantık yüksek uygulanmıştır. Çıkışta görülecek mantık seviyesi nedir?



Şekil 2.46

13. *TTL*'nin totem-direği şeklindeki çıkış yapısındaki diyodun amacı nedir?

14. İki totem-direği çıkışı niçin birbirlerine bağlanmamalıdır?

15. *TTL* devreleri yakınlarında, niçin V_{cc} ile toprak arasında yangeçit kapasitörleri gereklidir. Hangi yangeçit kapasitör değerleri kullanılmalıdır?

16. Açık-kollektörlü çıkışların avantaj ve dezavantajları nelerdir?

17. Üç-durum çıkışlı *TTL* lerin üç durumu nelerdir? Üç-durumlu çıkışlar ne için kullanılır?

18. Niçin Schottky *TTL* standart *TTL* den çok daha hızlıdır?

19. Aşağıdaki soruları cevaplamak için Tablo 2.2 yi kullanın.

   a. Bir 74LS serisi çıkışı kaç tane standart *TTL* girişini güvenli bir şekilde sürebilir?

   b. Bir 74ALS çıkışı kaç tane 74ALS girişini güvenli bir şekilde sürebilir?

   c. Bir 74C00 çıkışı kaç tane 74ALS girişini güvenli bir şekilde sürebilir?

20. ECL için tipik besleme gerilimleri nelerdir? Niçin *ECL* kapılarının o kadar kısa yayılma gecikmesi vardır? *ECL* bağlantılarındaki ana problemlerden birini anlatın.

21. Bir I^2L kapısının çalışmasını anlatın. I^2L nin bazı avantajları nelerdir?

22. *MOS* transistörleri veya entegre devrelerine dokunurken alınması gereken önlemleri anlatın.

23. Bir *CMOS* ters çeviricinin bir devresini çizin ve çalışmasını anlatın.

24. Alçak frekanslarda *CMOS* un önemli bir avantajı nedir? Bu avantaj yüksek frekanslarda neden kaybolmaktadır?

25. Kullanılmayan *TTL* girişleri ile ne yapılmalıdır? Kullanılmayan *CMOS* girişleri ile ne yapılmalıdır?

26. Bir sinyal üretecinin genliği, niçin üreteç bir mantık kapısına bağlanmadan önce bir osiloskopla kontrol edilmelidir?

27. Zıplamasız bir anahtar sinyali oluşturmak için kullanılabilecek basit bir devre gösterin.

28. Bir karşılaştırıcının çalışmasını anlatın.

29. Bazı giriş sinyallerini iyileştirmek için niçin Schmitt tetikleyici gerekir?

30. Bir kristal osilatör sinyal kaynağı ne zaman 555 zamanlama devresi gibi daha ucuz bir kaynağın yerine kullanılmalıdır?

31. Hattın karakteristik empedansından daha büyük bir dirençle sonlandırılmış bir iletim hattının sonunda, girişteki bir darbe tarafından oluşturulacak dalga şeklini gösteriniz.

32. 75 Ω luk bir iletim hattını sonlandırmak için kullanılacak direncin değeri ne olmalıdır?

33. -5,2 V kaynakla çalışan ECL çıkışındaki 50 Ω luk iletim hattının uygun sonlandırılması nasıl yapılmalıdır?

34. Güç kaynağı ile sinyal toprağı niçin sadece bir noktada birleştirilmelidir?

35. TTL yi CMOS a bağdaştırırken, TTL nin çıkışına bir yukarı-çekme direncinin konulması niçin iyi bir fikirdir?

36. Niçin çoğu CMOS standart TTL yi doğrudan süremez? CMOS tan TTL ye uygun bağdaştırma sağlayan iki metot nedir?

37. ECL yi TTL ye bağdaştırmak için ne gerekir?

38. Standart bir TTL kapısının alçak çıkış durumunu göstermek için bir LED in nasıl kulanılabileceğini bir devre şeması ile açıklayınız.

39. Bir CMOS veya TTL kapısından 100 mA lık bir röle sargısını sürmekte kullanılabilecek bir devre gösterin. Niçin her zaman ters kutuplandırılmış bir diyot röle sargısının uçları arasına bağlanmalıdır?

40. Katı-hal rölelerinin mekanik rölelere göre üstünlükleri nelerdir?

REFERANSLAR/KAYNAKÇA

Signetics Analog Manual-Applications/Specifications. Signetics Corporation. 1976, Kısım VI, "Timers." Sayfa 105-127.

Logic Data Book, National Semiconductor, 1981.

Linear Integrated Circuits (elkitabı), National Semiconductor, Şubat 1975.

CMOS/MOS Digital Integrated Circuits (Selection Guide/Data Application Notes). 1975 RCA Solid State DATA BOOK Series.

Interface Integrated Circuits (elkitabı), National Semiconductor, Ekim 1975.

Signetics Data Manual: Logic, Memories, Interface, Analog, Microprocessor, Military, Signetics Corporation, 1976.

CMOS Data Book, National Semiconductor, 1978.

MECL System Design Handbook. 1. baskı., Motorola Semiconductor Products, Inc., Computer Applications Engineering Department: William R. Blood, Jr. (yazar); Jon M. DeLaune and Jerry E. Prioste (katkıda bulunanlar); Edmund C.Tynan, Jr. (ed.) tarafından derlenmiştir. Ekim 1971.

Solid State Relays, RS Series (veri sayfası), Catalog no ISC-109, IDEC Systems and Controls Corporation, Santa Clara, CA.

Morris, Robert L., and John R. Miller (ed.): Designing with TTL Integrated Circuits (Texas Instruments Incorp. ın uygulama grubu tarafından hazırlanmıştır.), Texas Instruments Electronics Series, McGraw-Hill, New York, 1971.

Signetics Digital Linear MOS Applications (elkitabı), kısım. 7, "MOS Applications" Signetics Corporation, 1974.

Low Power Schottky and Macrologic TTL, Fairchild Semiconductor, Components Group, Fairchild Camera and Instrument Corp., 1975.

McMOS Handbook-Products, Characteristics, Applications, Motorola Semiconductor Products, Inc., 1974.

Blakeslee, Thomas R.: Digital Design with Standard MSI&LSI, Bölüm 10: Nasty Realities II, "Noise and Reflection," Wiley, New York, 1975, sayfa 256-278.

Intel Data Catalog 1982, Intel Corporation, 1982.

Allan, Alan: "Noise Immunity Comparison of CMOS versus Popular Bipolar Logic Families," Application Note AN-707, Motorola Semiconductor Products, Inc., 1973.

"Field Effect Transistors in Theory and Practice" AN-211 A Application Note, Motorola Semiconductor Products, Inc. in uygulama mühendisleri tarafından hazırlanmıştır.

BÖLÜM 3

BİRLEŞİK MANTIK, VERİ SEÇİCİLER, KODLAR VE ROMLAR

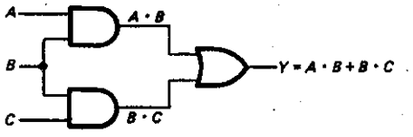
Bölüm 2 de temel mantık kapılarını tanıttık ve kullanılan mantık ailelerinin giriş ve çıkış parametrelerini tartıştık. Bu bölümde, bir fırın kontrol birimi, veri seçici veya kod dönüştürücü gibi sayısal sistemleri oluşturmak üzere mantık kapılarının nasıl birleştirilebileceğini göreceğiz.

Bu bölümü bitirdikten sonra şunları yapabileceksiniz.

1. Pozitif mantık, *negatif mantık* ve *karışık mantıklı* temel sayısal devrelerin çıkışlarını öngörebileceksiniz.
2. Mantık ifadelerini basitleştirmek için *boole cebirini* kullanabileceksiniz.
3. Dört değişkenliye kadar olan bir mantık ifadesini basitleştirmek için bir *Karnaugh* haritası kullanabileceksiniz.
4. Anahtar olarak kullanılan basit kapıların devrelerini çizebileceksiniz.
5. Tek bir hat üzerinden bir çok sinyali iletmek için kullanılan *veri seçici* ve *veri dağıtıcılarının* devrelerini çizebileceksiniz.
6. Uygun tablolar verildiğinde bir koddaki bir sayıyı başka bir koddaki karşılığı olan sayıya *dönüştürebileceksiniz.*
7. Pariteyi üretmek veya algılamak için *özel VEYA* kapılarının nasıl kullanılacağını gösterebileceksiniz.
8. Belli bir *ROM (yalnız-okunur-bellek)* verildiğinde gerekli adres ve yetki hatlarının sayısını belirleyebileceksiniz.
9. Bir *ROM* veri sayfasında bulunan zamanlama ve diğer parametreleri tanımlayabileceksiniz.
10. Tipik bir *EPROM* veya *EEPROM* un programlanmasını ve silinmesini anlatabileceksiniz.
11. Bir ROM ile bir *programlanabilir mantık dizisi (Programmable Logic Array: PLA)* arasındaki farkları söyleyebileceksiniz.

MANTIK KAPILARINI BİRLEŞTİRME

Sayısal mantık devreleri mantık şemalarıyla, doğruluk tablolarıyla veya boole cebiri ifadeleriyle gösterilebilirler. Her üç şekilde de çalışmayı öğrenmeniz önemlidir. Şekil 3-1, bir kapı devresi için üç şekli de göstermektedir. Şekil 3-1a' daki mantık şeması verildiğinde, doğruluk tablosu, her mümkün giriş birleşiminin etkisi devre üzerinde izlenerek kurulur. Örneğin A=0, B=1 ve C=1 şeklindeki giriş birleşimi yukarıdaki VE kapısının çıkışında bir 0, aşağıdaki VE kapısının çıkışında bir 1 oluşturur. Yukarıdaki VE kapısından gelen 0 ile aşağıdaki VE kapısından gelen 1, VEYA kapısının Y çıkışında bir 1 elde edilmesini sağlar. Bir VE kapısında girişlerden birisi alçak yani 0 ise, çıkışın da alçak yani 0 olacağını hatırlarsanız böyle bir devreyi izlemek kolaydır. Devrenin boole ifadesi Şekil 3-1*a* da gösterildiği gibi, mantık şemasında her kısım için çıkış ifadesini yazıp bunları birleştirerek doğrudan elde edilebilir. Yukarıdaki VE kapısının çıkışı $A.B$ dir. Aşağıdaki VE kapısının çıkışı $B.C$ dir. Bu terimler VEYA lanmıştır ve böylece Y çıkışı $A.B+B.C$ ye eşit olur.



(a)

A	B	C	Y	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$Y = \bar{A} \cdot B \cdot C$
1	0	0	0	
1	0	1	0	
1	1	0	1	$Y = A \cdot B \cdot \bar{C}$
1	1	1	1	$Y = A \cdot B \cdot C$

$Y = \bar{A} \cdot B \cdot C + A \cdot B \cdot \bar{C} + A \cdot B \cdot C$

(b)

$Y = A \cdot B + B \cdot C = B(A + C)$

(c)

Şekil 3-1 Birleşik mantık devresi. *(a)* Şema. *(b)* Doğruluk tablosu. *(c)* Çıkış ifadesi.

Doğruluk tablosundan boole cebiri ifadesini elde etmek için önce doğruluk tablosundaki her 1, yani **doğru**, çıkışı için bir terim yazın. Örneğin, Şekil 3-1 deki devre, A=0, B=1 ve C=1 girişleri için 1 çıkışı üretmektedir. Boole cebirinde bu terim $A.B.C$ olarak ifade edilecektir. Her üç değişkenin de terimde içerildiğine dikkat edin. Boole ifadesinin tamamını oluşturmak için her bir terim VEYA lanır (toplanır). Şekil 3-1 deki devre için sonuç, doğruluk tablosunun altında gösterildiği gibi

$$Y = \overline{A}.B.C + A.\overline{B}.C + A.B.\overline{C}$$

olur. Bu ifadeye çarpımlar-toplamı şeklindedir denir. Bu sonuç doğrudur; fakat mantık şemasından doğrudan elde edilen ifadeyle aynı değildir ve devrenin işlevini tanımlayan en basit boole ifadesi de değildir. Daha sonra iki ifadenin denk olduğunu göstereceğiz. Mantık şemasından veya doğruluk tablosundan elde edilen boole ifadesi nadiren en basit şeklinde olur. Mantık fonksiyonlarını basitleştirmede çok kullanılan iki yöntem boole cebiri ve Karnaugh haritalarıdır.

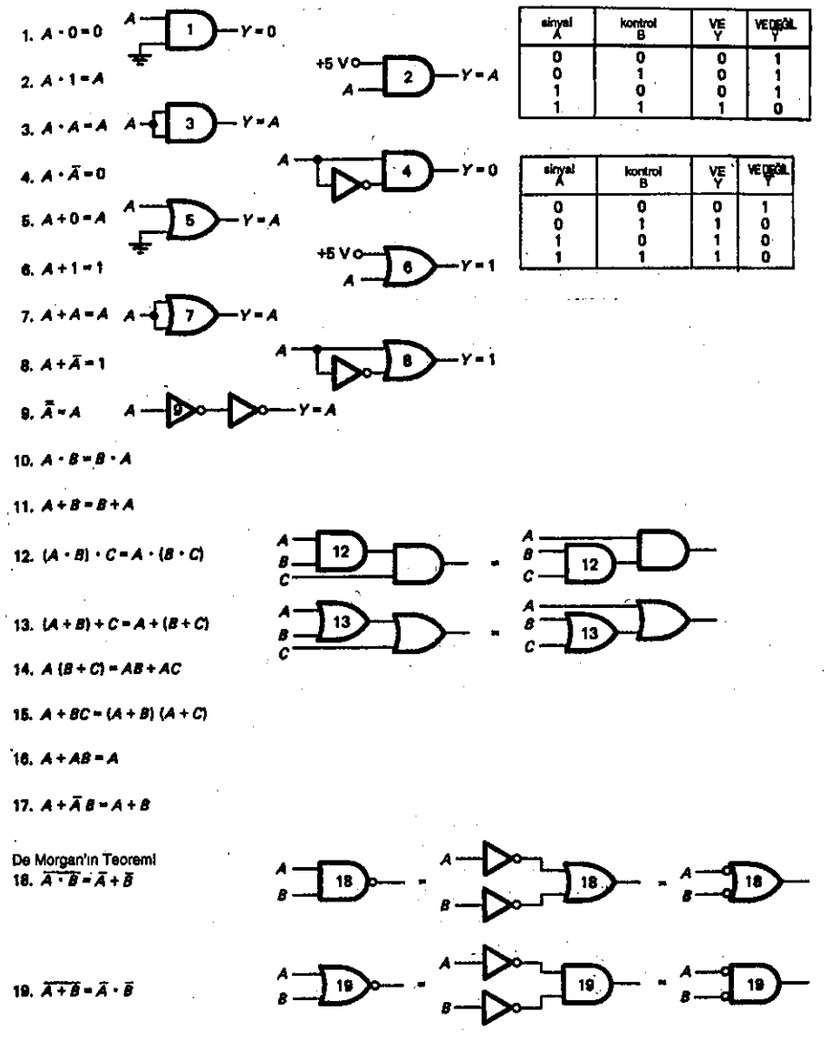
BOOLE CEBİRİ

Şekil 3-2 boole cebirinin kurallarını gösterir. Bunların bazıları ilk bakışta garip görünebilir, fakat bir doğruluk tablosundan veya mantık şemasından her birinin geçerli olduğunu görebilirsiniz. Bu kuralların bazılarının bildiğimiz cebirdeki kurallarla aynı olmadığına dikkat edin. Kurallar herhangi bir sayıda değişken için geçerlidir, ancak basit olması için gerekli olan minimum sayıda değişken kullanılmıştır.

Kural 1, A.0=0, basit olarak çok girişli bir VE kapısının girişlerinden herhangi birisi alçağa bağlanırsa, diğer girişlerin mantık durumu ne olursa olsun, VE kapısının çıkışının alçak olacağını belirtir. Buna kapıyı *yetkisiz kılmak (disable etmek)* denir veya bazan maskeleme (masking) denir, çünkü A girişindeki bir sinyal çıkışa ulaşmayacaktır. Bunu Şekil 3-2 deki VE/VE DEĞİL doğruluk tablosunda görebilirsiniz.

VE DEĞİL fonksiyonu da çok kullanıldığı birinci kuralın doğruluk tablosunda gösterilmiştir. Bir VE DEĞİL kapısı için eğer girişlerden herhangi birisi alçaksa çıkış yüksek olacaktır.

Kural 2, iki-girişli bir VE kapısının bir girişi veya çok-girişli bir VE kapısının biri dışında bütün girişleri yükseğe bağlanırsa, çıkışın kalan diğer girişle aynı olacağını belirtmektedir. Bu, Şekil 3-2 deki doğruluk tablosundan kolayca görülebilir. O halde, biri dışında bütün girişleri yüksek olan bir VE kapısı, A girişindeki sinyali çıkışına geçirmek üzere yetkili kılınmış olur (enable edilir).



Şekil 3-2 Boole cebiri denkleri

A sinyal girişi dışında bütün girişleri yükseğe bağlanan bir VE DEĞİL kapısı bir ters çevirici gibi çalışacaktır.

Kural 3, bir VE kapısının girişleri birbirine bağlanırsa çıkışın girişi izleyeceğini göstermektedir. Bu yapı bazan tampon olarak isimlendirilir ve doğruluk tablosunda A=0 ve B=0 durumu ile A=1 ve B=1 durumu olarak gösterilmiştir.

Ayrıca kural 3 ün doğruluk tablosundan, girişleri birbirine bağlanmış bir VE DEĞİL kapısının bir ters çevirici gibi çalışacağını söyleyebilirsiniz. Ancak unutmayın ki pratikte girişlerin birbirine bağlanması pek istenmez, çünkü bu, sürücü kapıya fazladan kapasitif yük ekler.

Kural 4, bir değişken tümleyeni ile VE lenirse, çıkışın her zaman alçak olduğunu gösterir. Her zaman girişlerden birisi alçak olacağından, çıkış alçaktır.

Kurallar 5, 6, 7 ve 8, VEYA /VEYA DEĞİL kapıları için karşılık gelen sonuçları göstermektedir. İki-girişli VEYA/VEYA DEĞİL kapılarının bir girişteki alçakla yetkili kılındığına, ve bir girişteki bir yüksekle yetkisiz kılındığına dikkat edin. Bir VEYA kapısının girişleri birbirine bağlanırsa, çıkış girişi izler. Girişleri birbirine bağlanmış bir VEYA DEĞİL kapısı bir ters çevirici gibi çalışır. Bir değişken tümleyeni ile VEYA lanırsa, her zaman girişlerden biri yüksek olacağından, çıkış her zaman yüksektir. Bir VEYA kapısının herhangi bir girişindeki bir yüksek, çıkışında yüksek oluşturur.

Kural 9, iki ters çevirme işleminin birbirlerini kısalttığını söyler. Bir ters çeviriciden geçen bir 0, bir 1 üretir. Bu 1, bir diğer ters çeviriciden geçince orijinal 0'ı tekrar üretir.

Kurallar 10 ve 11, bir kapının girişlerinin denk olduğunu ve aralarında yer değiştirebileceklerini göstermektedir.

Kurallar 12 ve 13, değişkenlerin hangi sırada VE lendiğinin veya VEYA landığının önemli olmadığını gösterir. Örneğin, eğer A ile B VEYA lanırsa ve sonuç, C ile VEYA lanırsa, bu, B ile C yi VEYA lamak ve sonucunu da A ile VEYA lamakla aynıdır.

Kurallar 14 ve 15, boole ifadelerinin dağılma özelliklerini gösterir. Kural 14 normal cebirdeki paranteze alma işlemine denktir. Kural 15 çok açık değildir, fakat şu şekilde düşünebilirsiniz. İlk ifade, eğer A=0 ise, bir 1 çıkışı oluşturmak için, B ve C nin ikisinin de 1 olması gerektiğini söyler. Eğer A=1 ise, bu taktirde B ve C "farketmez" (don't care), çünkü bir 1 ne ile VEYA lanırsa VEYA lansın bir 1 çıkışı verir. İkinci ifade de ise, eğer A=0 ise, 0 ile B VEYA lanınca B, ve 0 ile C VEYA lanınca C elde edilir. Böylece, daha önce olduğu gibi yalnız B ve C nin her ikisi de 1 ise çıkış 1 olur. Eğer A=1 ise, $A+B$ terimi 1 e eşit ve $A+C$ terimi 1 e eşit (Kural 6) olacağına göre B ve C "farketmez". 1.1=1 ol-

duğuna göre çıkış 1 olacaktır. Bu tartışma boole cebiri kurallarının mantık ifadelerinin şekillerini değiştirmek ve onları azaltmakta nasıl kullanılabileceğini göstermiştir.

Kurallar 16 ve 17 çok kullanılan kısaltmalardır. Kural 16 yı ispatlamak için önce iki terimi A parantezine alarak $A(1+B)$ yi elde edin.sonra, $1+B=1$ olduğuna göre (kural 6), buradan $A.1$ elde ederiz ki bu da A ya eşittir (kural 2) Kural 17 yi ispatlamak için kural 15 i kullanarak denklemin sol tarafını genişletin.

$$A+\bar{A}B=(A+\bar{A}).(A+B)$$

$A+\bar{A}=1$ olduğuna göre (kural 8), bu ifade $A+B$ ye indirgenir.

Son iki kurala geçmeden önce bu kuralların mantık ifadelerinin basitleştirilmesinde nasıl kullanıldığına dair bir örnek daha gösterelim. Şekil 3-1, bir mantık devresinin mantık şemasını ve doğruluk tablosunu göstermektedir. Daha önce açıklandığı gibi, mantık şemasından doğrudan elde edilen boole ifadesi

$$Y=A.B+B.C$$

dir. Doğruluk tablosundan elde edilen boole ifadesi ise

$$Y=A\bar{B}C+AB\bar{C}+ABC$$

dir. Boole kurallarını kullanarak bunların denkliğini şu şekilde gösterebilirsiniz:

$$Y=A\bar{B}.C+A.B.\bar{C}+A.B.C$$

ile başlayalım. Son iki terimi $A.B$ parantezine alın.

$$Y=A\bar{B}.C.+A.B.(\bar{C}+C)$$

bulunur. $\bar{C}+C=1$ (kural 8) olduğuna göre ifade

$$Y=A\bar{B}.C+A.B$$

ye indirgenir. Buradaki iki terim B parantezine alınırsa

$$Y=B.(A+A\bar{.}C)$$

elde edilir. Kural 17 kullanılarak

$$Y=B\ (A+C)$$

elde edilir. Kural 14'e göre çarparak

$$Y=A.B+B.C$$

bulunur. Bu da mantık şemasından doğrudan elde edilen ifadenin aynısıdır.

Kurallar 18 ve 19 a genellikle *De Morgan'ın teoremleri* denir. Mantık dev-

relerinin analizinde ve basitleştirilmelerinde çok faydalıdırlar. Kural 18 bir VEYA kapısının girişleri ters çevrilirse bir VE DEĞİL kapısının elde edileceğini söylemektedir. Diğer bir deyişle, bir VE DEĞİL kapısı, girişleri ters çevrilmiş bir VEYA kapısı gibi düşünülebilir. Bunu Şekil 3-3 teki $A.B$ ve $A+B$ sütunlarını karşılaştırarak ispat edebilirsiniz.

$$\overline{A+B} = \overline{A}.\overline{B}$$

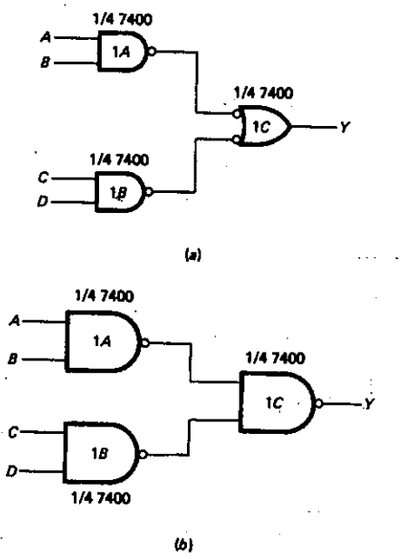
A	B	$A.B$	$\overline{A.B}$	$A+B$	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$	$\overline{A}+\overline{B}$
0	0	0	1	0	1	1	1	1	1
0	1	0	1	1	0	1	0	0	1
1	0	0	1	1	0	0	1	0	1
1	1	1	0	1	0	0	0	0	0

Şekil 3-3 De Morgan'ın teoremini gösteren doğruluk tablosu

Şekil 3-2 ayrıca, bu teoremi mantık sembolü şeklinde göstermektedir. Ters çevrilmiş girişli VEYA sembolü, mantık şemalarında VE DEĞİL kapılarını temsil etmek için sıklıkla kullanılır. Şekil 3-4a, 7400 dörtlü iki-girişli VE DEĞİL kapı entegresinin (IC nin) dörtte birini temsil eden sembolün kullandığı bir şema göstermektedir. Şemayı bu şekilde çizmek çıkış ifadesini ve doğruluk tablosunu belirlemeyi kolaylaştırır. IC 1*a* ve IC 1*b* nin çıkışlarındaki ters çevirici yuvarlaklar, IC 1*c* nin girişindeki yuvarlaklarla kısalacağından, çıkışın $A.B$ ile $C.D$ nin VEYA lanmasından ibaret olduğunu görebilirsiniz. Çıkış ifadesi

$$Y = A.B+C.D$$

dir. Bu, sadece pozitif mantık VE DEĞİL kapıları kullanarak VEYA fonksiyonları oluşturmanın çok kullanılan bir yöntemidir. Şekil 3-4*b* deki devre, Şekil 3-4*a* daki devre ile aynı parçaları kullanmaktadır ve aynı doğruluk tablosuna sahiptir, fakat bunun çıkış ifadesi, IC 1*c* nin ters çevrilmiş girişli VEYA ile temsil edildiği devreninki kadar hemen açık bir şekilde ifade edilememektedir.

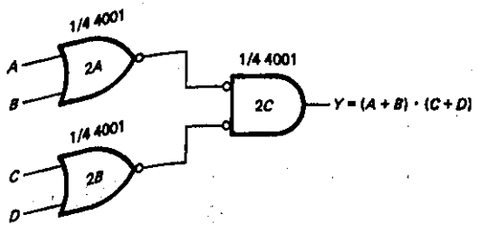


Şekil 3-4 Pozitif mantık VE DEĞİL kapıları kullanan mantık devresi. *(a)* VE DEĞİL'in, girişleri ters çevrilmiş VEYA ile temsil edilmesi *(b)* Sadece standart VE DEĞİL kapısı kullanan şema.

Kural 19, bir VEYA DEĞİL kapısının, girişleri ters çevrilmiş bir VE kapısına denk olduğunu göstermektedir. Bunu Şekil 3-3 teki $A+B$ ve $\overline{A}.\overline{B}$ sütunlarını karşılaştırarak ispatlayabilirsiniz. Şekil 3-2*b* bu teoremi ayrıca mantık sembolü şeklinde göstermektedir. Şekil 3-5 te CMOS 4001 VEYA DEĞİL kapısı IC 2*c* nin ters çevrilmiş VEYA gösterilişi, çıkış ifadesinin

$$Y=(A+B).(C+D)$$

olduğunu kolayca görmemizi sağlamaktadır. IC 2*a* ve IC 2*b* nin çıkışlarındaki ters çevirici yuvarlaklar IC 2*c* nin girişindeki ters çevirici yuvarlakları kısaltmaktadır. Bu örnekte gösterilen yol, sadece pozitif mantık VEYA DEĞİL kapıları kullanarak VE fonksiyonu elde etmekte çok kullanılır.



Şekil 3-5 VEYA DEĞİL kapısı devresini izlemeyi kolaylaştırmak için ters çevrilmiş girişli VE sembolünün kullanılışı.

Boole ifadelerini basitleştirmede De Morgan'ın teoremlerini kullanırken faydalı bir düşünce şekli de şöyledir. Bir boole ifadesindeki terimlerin üzerindeki çizgiyi bölmek veya birleştirmek, terimler arasındaki işareti değiştirir. Örneğin, $A.\overline{B}$ üzerindeki çizgiyi bölmek aradaki (.) işaretini değiştirerek dengi olan $\overline{A}+B$ yi verir (kural 18). $\overline{A.B}$ üzerindeki çizgiyi birleştirmek aradaki (.) işaretini değiştirerek dengi olan $\overline{A}+\overline{B}$ yi verir (kural 18). Bu oyunun boole ifadelerinin basitleştirilmesinde kullanılışı şöyledir. Şekil 3-4*b* deki üç VE DEĞİL kapılı devreden doğrudan elde edilen ifade

$$Y=\overline{(A.B)}\,.\,\overline{(C.D)}$$

dir. Üstteki çizgiyi bölerek ve terimler arasındaki işareti değiştirerek

$$Y=\overline{\overline{(A.B)}}+\overline{\overline{(C.D)}}$$

bulunur. Herhangi bir terim üzerindeki çift çizgi kısalacağından, bu ifade Şekil 3-4*a* da doğrudan bulunan sonuçta aynı olan

$$Y=A.B+C.D$$

ye eşittir.

Diğer bir örnek olarak,

$$Y=\overline{\overline{(A+B)}+\overline{(C+D)}}$$

boole ifadesi üstteki çizgiyi bölerek

$$Y=\overline{\overline{(A+B)}}.\overline{\overline{(C+D)}}$$

ve çift çizgileri kısaltarak

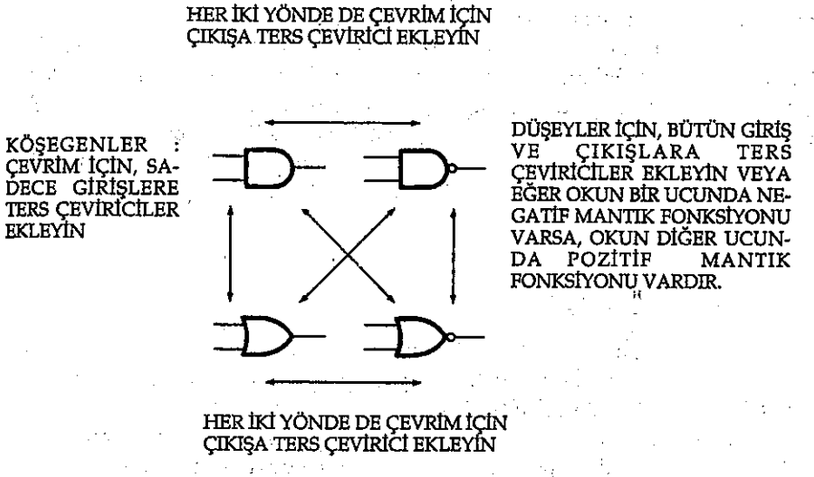
$$Y=(A+B)\,.\,(C+D)$$

şeklinde basitleştirilebilir.

O halde, sonuç olarak, bir VE DEĞİL kapısı girişleri ters çevrilmiş bir VEYA kapısına $(\overline{A.B}=\overline{A}+\overline{B})$ ve bir VEYA DEĞİL kapısı girişleri ters çevrilmiş bir VE kapısına $(\overline{A+B}=\overline{A}.\overline{B})$ denktir.

MANTIK KAPI DÖNÜŞÜMLERİNİN ÖZETİ

Unutmamanızı sağlamak için, Şekil 3-6, bir kapının çıkışlarına, girişlerine veya hem çıkışlarına hem de girişlerine ters çeviriciler ekleyerek yapılabilecek kapı dönüşümlerini özetlemektedir. Sadece bir çıkışa bir ters çevirici eklenmesi, bir VE den bir VE DEĞİL e veya bir VEYA DEĞİL den VEYA ya geçişte olduğu gibi, şemada yatay yöndeki dönüşümü gerçekleştirir.



Şekil 3-6 Mantık kapı dönüşümlerinin özeti

Sadece girişlere ters çeviriciler eklenmesi şemada çapraz yöndeki dönüşümleri yapar. Ters çevrilmiş girişli bir VEYA kapısı bir VE DEĞİL gibi çalışır $(A+B=\overline{A.B})$, ve ters çevrilmiş girişli bir VE kapısı bir VEYA DEĞİL kapısı gibi çalışır $(\overline{A.B}=A+B)$. Okların çift yönlü olduğunu unutmayınız. Bunun anlamı dönüşümlerin her iki yönde de geçerli olduğudur. Bu çapraz oklar De Morgan'ın teoremini özetler.

Şekil 3-6 daki dik oklar, bir kapının bütün girişlerine ve bütün çıkışlarına ters çeviriciler ekleyerek gerçekleştiren dönüşümleri temsil eder. Ters çevrilmiş giriş ve ters çevrilmiş çıkışlı bir VE DEĞİL kapısının boole ifadesi

$$Y=\overline{\overline{A}.\overline{B}}$$

dır. Çift çizgi kısaltılırsa

$$Y=\overline{A.B}$$

bulunur ki bu da

$$Y=\overline{A+B}$$ *(De Morgan'ın teoremi)*

şeklinde ifade edilebilir. Bu dik oklar pozitif mantık fonksiyonları ile negatif mantık fonksiyonları arasındaki karşılıkları gösterir. Bu, önümüzdeki kısmın konusudur. Şekil 3-6 daki şemada önemli diğer bir nokta, şemanın, herhangi bir mantık fonksiyonunun sadece bir çeşit kapı, örneğin VE DEĞİL kapıları, ve ters çeviriciler kullanılarak nasıl gerçekleştirilebileceğini göstermesidir. Bir tasarımda kullanılan farklı IC sayısını azaltmak ve kapıları boş yere harcamaktan kaçınmak için devre tasarımcıları bu tür hileleri sık sık kullanırlar. Örneğin, sadece bir VE DEĞİl kapısı gerekiyorsa, dört pozitif mantık VEYA DEĞİL kapısı içeren bir 7402 dörtlü iki-girişli VEYA DEĞİL kapı entegresi kullanmak yerine, bir tasarımcı, devrede kullanılmayan arta kalmış bir VE DEĞİL ve üç ters çeviriciyi, bir VEYA DEĞİL fonksiyonu oluşturmak için kullanabilir.

NEGATİF MANTIK

Bu kitapta şimdiye kadar basit olması için sadece, bir kapının daha pozitif çıkış geriliminin mantık 1 ve daha negatif çıkış gerilimin mantık 0 olduğu, pozitif mantık gösteriliş şekilleriyle uğraştık. ECL, eski PMOS ve bazı bilgisayarlar bu gösterilis şekillerinin ters döndüğü negatif mantığı sık kullanır. Negatif mantıkta bir 0 daha pozitif gerilimdir ve bir 1 de daha negatif gerilimdir. O halde yüksek-eşikli PMOS için negatif mantık 1 in karşılığı -13 V, 0 ın karşılığı toprak olabilir. TTL için bir negatif mantık 1 veya yüksek 0,4 V tur ve bir negative mantık 0 veya alçak 2, 4 V tur.

Pozitif Mantık VE DEĞİL

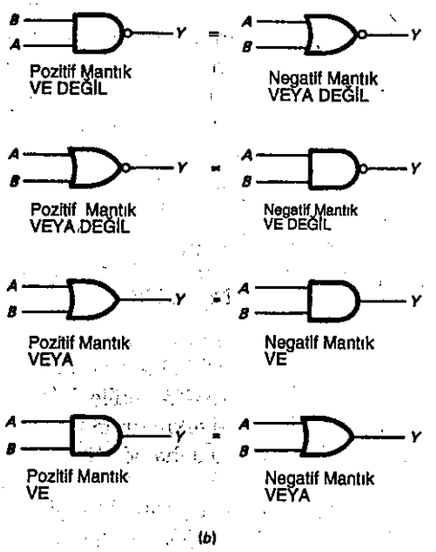
A	B	Y
0,4 V = L	0,4 V =L	2,4 V =H
0,4 V = L	2,4 V =H	2,4 V =H
2,4 V = H	0,4 V =L	2,4 V =H
2,4 V = H	2,4 V =H	0,4 V =L

$Y=\overline{A.B}$

Negatif Mantık VEYA DEĞİL

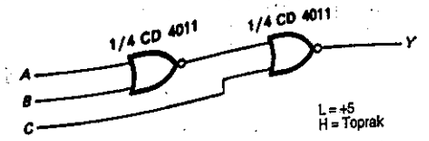
A	B	Y
0,4 V = H	0,4 V =H	2,4 V =L
0,4 V = H	2,4 V =L	2,4 V =L
2,4 V = L	0,4 V =H	2,4 V =L
2,4 V = L	2,4 V =L	0,4 V =H

$Y=\overline{A+B}$



Şekil 3-7 Negatif ve pozitif mantık denkleri. *(a)* Doğruluk tabloları. *(b)* Şematik sembolleri.

7400 pozitif mantık VE DEĞİL kapılarından biri gibi bir kapı, giriş gerilimlerine yine aynı şekilde tepki gösterir. Fakat mantık seviye tanımları ters döndürüldüğü için negatif mantık gösteriliş şekillerinin kullanıldığı bir sistemde, bu kapı VEYA DEĞİL fonksiyonunu gerçekleştirecektir. Bunu göstermek için Şekil 3-7*a* daki doğruluk tablolarını kullanabilirsiniz. Doğruluk tablosunun üst kısmı bildiğimiz pozitif mantık VE DEĞİL kapısının gerilim seviyelerini ve denk mantık seviyelerini göstermektedir.



(a)

A	B	C	Y
L	L	L	L
L	L	H	L
L	H	L	H
L	H	H	L
H	L	L	H
H	L	H	L
H	H	L	H
H	H	H	L

$Y = \overline{A} \cdot B \cdot \overline{C} + A \cdot \overline{B} \cdot \overline{C} + A \cdot B \cdot \overline{C}$
$Y = \overline{C}(A + B)$

(b)

Şekil 3-8 Negatif mantıkta VEYA DEĞİL kapısı olarak kullanılan CD4011 pozitif mantık VE DEĞİL kapısı. *(a)* Şema. *(b)* Doğruluk tablosu.

Tablonun alt kısmı aynı giriş gerilimlerini negatif mantık seviyeleri ile birlikte göstermektedir. Tablonun alt kısmındaki mantık durumlarını dikkate alırsak, eğer girişlerden herhangi birisi yüksekse çıkışın alçak olduğunu söyleyebiliriz. Bunu VEYA DEĞİL kapısının çalışması olarak tanımanız gerekir. Bir pozitif mantık VE DEĞİL kapısı negatif mantıkta VEYA DEĞİL fonksiyonunu gerçekleştirir. Benzer analiz gösterecektir ki bir pozitif mantık VEYA DEĞİL kapısı negatif mantıkta VE DEĞİL fonksiyonunu gerçekleştirmektedir. Benzer şekilde, Şekil 3-7*b* de gösterildiği gibi, bir pozitif mantık VE kapısı bir negatif mantık VEYA kapısına denktir ve bir pozitif mantık VEYA kapısı bir negatif mantık VE kapısına denktir. Şekil 3-6 daki dik oklar pozitif ve negatif

mantıktaki denk kapıları gösterir. Bu dik oklar, aynı zamanda, örneğin bir VE DEĞİL kapısının, bütün giriş ve çıkışlarını ters çevirerek bir VEYA kapısına dönüştürülebileceğini göstermektedir. Tamamen negatif mantık kullanan sistem şemalarında kullanılan şematik semboller kapıların fonksiyonlarını gösterir. Bir kapı gerçekte bir 7400 ın dörtte biri olabilir, fakat negatif mantık şemasında bir VEYA DEĞİL kapısı olarak gösterilecektir. Şekil 3-8a ve b negatif mantık kullanan bir devrenin şemasını ve doğruluk tablosunu göstermektedir. Bir defa kapılar uygun negatif mantık fonksiyonları ile gösterildikten sonra, analiz, pozitif mantık devrelerininki ile aynıdır. Saf negatif mantık sistemleri ile nadiren karşılaşılır, fakat birinde arıza bulmanız gerekirse, mantık yüksek ve mantık alçakla uygun gerilim seviyelerini eşleştirmeyi unutmayın.

KARMA MANTIK

Pozitif mantık gösteriliş şeklini kullanan bir sistemde sinyallerin *"aktif"* veya *"doğru"* seviyesi yüksek gerilimdir. Veri kitapları genellikle aygıtları pozitif mantık fonksiyonlarına göre isimlendirirler. Örneğin, bildiğimiz 74LS00, her biri pozitif mantık VE DEĞİL fonksiyonunu gerçekleştiren dört tane kapı içerir. Şekil 3-7 bu kapılardan biri için gerilim doğruluk tablosunu göstermektedir.

Girişlerin ve çıkışın aktif seviyesi 2,4 V ise, çıkış fonksiyonunun

$$Y = \overline{A \cdot B}$$

olduğunu görebilirsiniz.

Negatif mantık kullanan bir sistemde sinyallerin *"aktif "* veya *"doğru"* seviyesi bir alçak gerilimdir. Şekil 3-7 deki gerilim doğruluk tablosundan 74LS00 aygıtındaki bir kapı

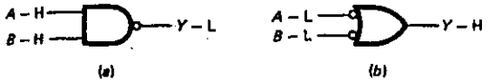
$$Y = \overline{A+B}$$

fonksiyonunu gerçekleştirir.

Karma mantık kullanan sistem şemalarında yüksek gerilimli "aktif" veya "doğru" seviyeli sinyallerle alçak gerilimli "aktif" veya "doğru" sinyallerin ikisi de bulunmaktadır. Şemadaki her sinyalin isminin arkasında bulunan bir harf veya sembol o sinyal için aktif seviyenin bir yüksek gerilim mi yoksa bir alçak gerilim mi olduğunu belirtir. Bir aktif yüksek sinyal -*H*, .*H*, -1 sembolleriyle veya hiç sembolsüz gösterilir. *KEY* isimli aktif yüksek bir sinyalin gösterilişi KEY-H, KEY.H, KEY-1 veya sadece KEY şeklinde olabilir. Bir aktif alçak sinyal -L, .L, -0, /, veya * ile gösterilir. Örneğin BELT isimli aktif alçak bir sinyalin gösterilişi BELT-L, BELT.L, BELT-0, BELT/, ve BELT* şeklinde olabilir. Verilen bir şemada, sadece bir sembol grubu bir sinyalin aktif yüksek veya aktif alçak olduğunu göstermek için kullanılır. Çoğu mantık tasarımları ve sistem

şemaları karma mantık gösteriliş şekillerini kullanmaktadır. Göreceğiniz gibi, karma mantık gösterimin kullanımı bir devrenin mantık fonksiyonlarını doğrudan şemadan belirlenmesini çok kolaylaştırır. Aslına bakarsanız, bu gösteriliş şekilleri De Morgan'ın teoreminin tartışılması sırasında gösterilen sembol denkliklerinin bir uzantısıdır.

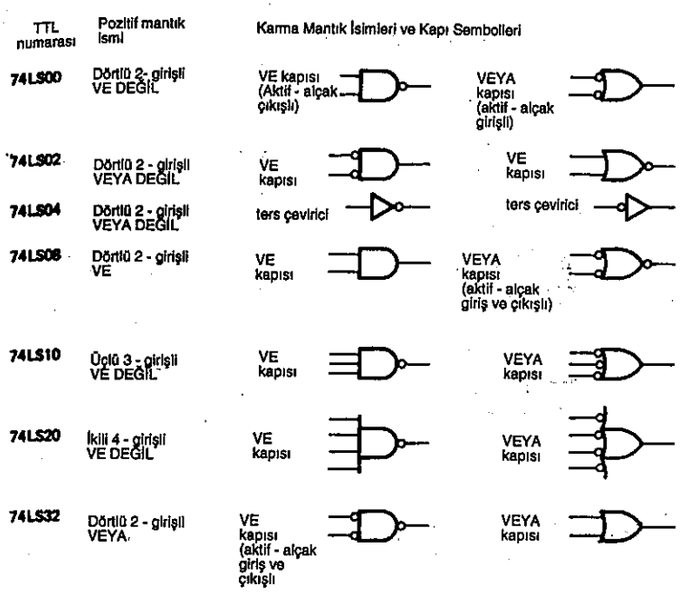
Şekil 3-9*a*, bir 74LS00 kapısının karma mantık sisteminde şemasının nasıl temsil edilebileceğini göstermektedir. Karma mantıkta, bu aygıt aktif yüksek girişleri ve aktif alçak çıkışları olan bir VE kapısı olarak düşünülür. Eğer *A* girişi yüksekse (high, *H*) ve *B* girişi yüksekse *(H)*, bu taktirde çıkış alçak (low, L) olacaktır. O halde bu karma mantıktır, çünkü girişlerin *aktif seviyesi yüksek (yuvarlak yok)* ve çıkışın *aktif seviyesi alçaktır (yuvarlak var).*



Şekil 3-9 Pozitif mantık VE DEĞİL kapısının karma mantık dengi. *(a)* VE gösterilişi. *(b)* VEYA gösterilişi

De Morgan'ın teoremi ve Şekil 3-3 tarafından gösterildiği gibi bir pozitif mantık VE DEĞİL kapısı, girişleri ters çevrilmiş bir VEYA kapısıyla temsil edilebilir.Bunun karma mantık şematik sembolü Şekil 3-9*b* de gösterilmiştir. Girişlerdeki yuvarlaklar girişlerin aktif seviyelerinin alçak olduğunu göstermektedir. Çıkışta yuvarlağın olmaması çıkışın aktif seviyesinin yüksek olduğunu gösterir. O halde, bu sembol *A* girişi aktif *(L)* veya B girişi aktif *(L)* ise çıkışın aktif *(H)* olacağını söylemektedir.

Şekil 3-10, karma mantık sembolleri kullanarak bir çok bilinen tip mantık devresinin iki şekilde temsil edilişini göstermektedir. 74LS02 deki gibi bir pozitif mantık VEYA DEĞİL kapısı, aktif yüksek girişleri ve aktif alçak çıkışı olan bir VEYA kapısı olarak düşünülür. Bu kapı ayrıca aktif alçak girişleri ve aktif yüksek çıkışı olan bir VE kapısı olarak temsil edilebilir. Bir ters çeviricinin sadece bir sinyalin aktif seviyesini değiştirdiğine dikkat edin. Şekil 3-10, sizin için De Morgan'ın teoreminin yeniden bir gözden geçirilmesi olmalıdır. Gelecek kısımda, bir karma mantık sisteminde aktif yüksek ve aktif alçak sinyallerle bu sembollerin nasıl kullanıldığını göreceğiz.



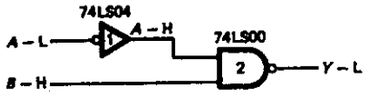
Şekil 3-10 Çok kullanılan mantık kapıları için karma mantık isimleri ve sembolleri

KARMA MANTIK DEVRELERİNİN ANALİZİNİ YAPMAK

Şekil 3-11, bir sistemde, analizini yapmanız gerekebilecek basit bir karma mantık devresini göstermektedir. *A-L* sinyalinin aktif seviyesi bir ters çevirici ile *A-H* ye çevrilmiştir. Bu sinyal B-H ile VE lenmiştir. Çıkıştaki yuvarlak çıkışın aktif alçak olacağını gösterir. Bu devre şöyle tanımlanabilir: Eğer *A* aktifse *(L)* ve *B* aktifse *(H)*, çıkış aktif *(L)* olacaktır. Bu fonksiyon

$$Y\text{-}L = (A\text{-}L).\ (B\text{-}H)$$

şeklinde yazılacaktır.

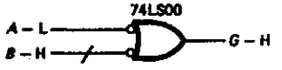


Şekil 3-11 Karma mantık VE kapısının devresi.

Şekil 3-12, analiz etmeniz için başka bir karma mantık devresini göstermektedir. Sembolden, hemen çıkış fonksiyonunun bir VEYA fonksiyonu olduğunu ve aktif yüksek olduğunu görebilirsiniz. *A-L* sinyali aktif alçaktır ve bir aktif alçak girişine bağlıdır. O halde çıkış ifadesinde sadece *A-L* olarak görülecektir. *B-H* sinyali bir aktif alçak girişine bağlıdır, böylece çıkış ifadesinde tümleyeni ile görülecektir. Bazı şemalarda bunu göstermek için sinyal hattı üzerine bir çapraz çizgi konmaktadır. Bu durumda çıkış ifadesi

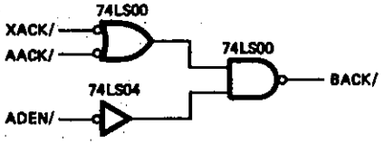
$$G\text{-}H=(A\text{-}L) + \overline{(B\text{-}H)}$$

olur. Eğer A aktif *(L)* ise veya eğer B aktif değilse (L), çıkış aktif (H) dir.



Şekil 3-12 Karma mantık VEYA kapısının devresi.

Karma mantığın diğer bir örneği Şekil 3-13 te verilmiştir. Eğer XACK aktif *(L)* veya AACK aktif *(L)*, ve ADEN aktif *(L)* ise BACK aktif *(L)* olacaktır.



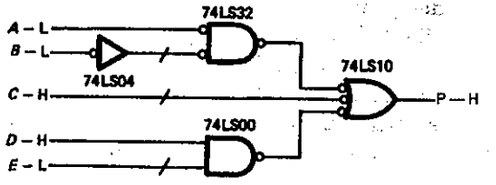
Şekil 3-13 Karma mantık devresi örneği.

Karma mantığın son bir örneği Şekil 3-14 te gösterilmiştir. Çıkış üç terimin toplamı (VEYA) olacaktır. 74LS32 nin çıkışındaki yuvarlak ve 74LS10 un

girişindeki yuvarlak birbirini kısaltıyorlar diye düşünülebilir, böylece bu kısıma ait terim *A-L*. $\overline{B\text{-}L}$ olur. *C-H* sinyali bir aktif alçak girişine uygulanmıştır, o halde çıkış ifadesinde buna ait terimin tümleyeni alınır. Bu da C-H dır. 74LS00 kısmı için ise, bunun çıkışındaki yuvarlak 74LS10 girişindeki yuvarlakla kısalır. Böylece bu kısım için olan terim *D-H*. $\overline{E\text{-}L}$ dır. O halde toplam ifade

$$P\text{-}H= (A\text{-}L.\ \overline{B\text{-}L}) + \overline{C\text{-}H} + (D\text{-}H.\overline{E\text{-}L})$$

olur.

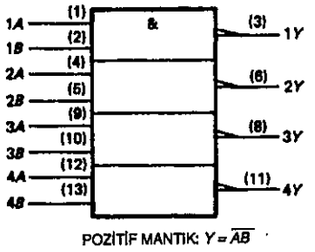


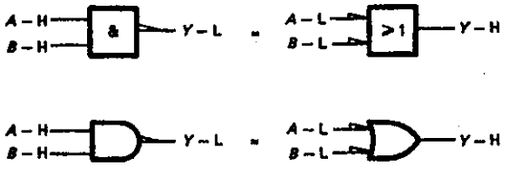
Şekil 3-14 Karma mantık devresi örneği.

Karma mantık şemalarının analizi modern sayısal sistemlerin arızalarının bulunmasında çok önemlidir. Çalışmanız için bölümün sonunda daha fazla problem verilmiştir.

KARMA MANTIK ŞEMALARI İÇİN YENİ SEMBOLLER

Bir sistem şemasında karma mantık gösterilişlerinin kullanıldığını açıkça belirtmek için, *IEC (International Electrotechnical Commission)* ve *IEEE (Institute of Electrical and Electronic Engineers)* yeni sembol standartları kabul etmişlerdir (Ek *B* ye bakın). Ek *B* de gösterildiği gibi bu standartlar bütün mantık elemanları için, içlerinde elemanın gerçekleştirdiği mantık fonksiyonunu gösteren semboller bulunan dikdörtgen şekiller kullanmaktadır.Örneğin (&) işareti VE fonksiyonu için kullanılır. ≥ 1 sembolü VEYA fonksiyonunu gösterir. Karma mantığı kullanan şemalarda, bu standartlar bildiğimiz sırf negatif veya sırf pozitif mantık şemalarında kullanılan ters çevirici yuvarlağı bir üçgen kama ile değiştirmektedir. Şekil 3-15a, 74LS00 dörtlü iki -girişli pozitif VE DEĞİL kapılarının yeni standart mantık sembolünü göstermektedir. Şekil 3-15b, bu aygıttaki bir kapının iki şekilde temsil edilişini göstermektedir.





Şekil 3-15 *(a)* Dörtlü iki-girişli pozitif mantık VE DEĞİL kapısının bağımlılık gösterilişi sembolü. *(b)* Bir tek karma mantık VE DEĞİL kapısı için denk semboller. *(c)* Karma mantık VE DEĞİL kapısı için hibrit semboller.

Bu standartlar oldukça yeni olduğu için, varolan çoğu karma mantık şemaları geleneksel sembol şekillerini ve giriş ve çıkışlarda yuvarlakları kullanmaktadır. Açıklık olması için, bu kitapta (ikinci baskı) geleneksel VE, VEYA, ve ters çevirici şekillerini kullanmaktayız, fakat Şekil 3-15c deki gibi, karma mantığın kullanıldığını göstermek için giriş ve çıkışlarda kamalar koymaktayız.

MANTIK FONKSİYONLARININ BASİTLEŞMESİNE AİT ÖRNEKLER

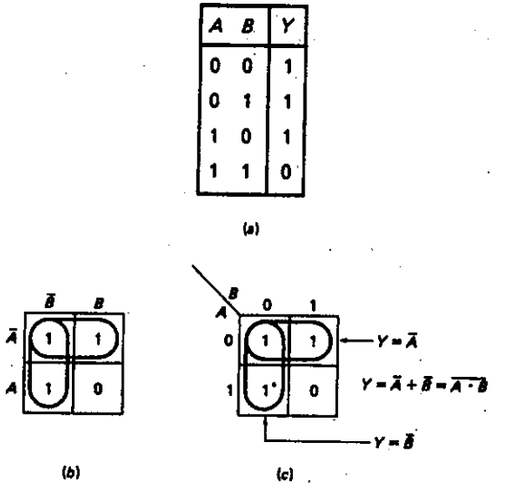
Önceki kısımlarda mantık fonksiyonlarının boole cebiri kullanılarak basitleştirilebileceklerini görmüştük. Paranteze alarak ve yerine koyarak indirgenmiş ifadeler elde etmeniz mümkün olabilir. Bu yaklaşımla ilgili tek problem, geometride bir teoremi ispat etmeye çalıştığınızda karşılaştığınız gibi, bir yanlış dönüş veya adım, sizin aynı yerde dönüp durmanıza sebep olabilir. Daha güvenilir ve hatırlanması kolay bir yol Karnaugh haritalamasıdır.

KARNAUGH HARİTALAMASI

Karnaugh haritalaması bir mantık ifadesinin doğruluk tablosunun grafik yöntemle çizimi ve sonra bu çizimden basitleştirilmiş ifadelerin elde edilmesidir. Karnaugh haritaları (*K*-haritaları) sekiz giriş değişkenine kadar kullanılabilir, fakat biz sadece dört giriş değişkenine kadar kullanımları üzerinde duracağız. Beş veya daha fazla değişkenli ifadeleri bilgisayar programları kullanarak basitleştirmek artık daha pratik olmuştur ve daha sık karşılaşılmaktadır.

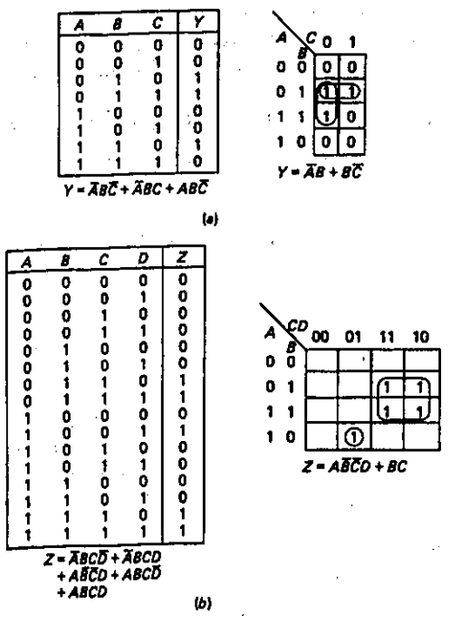
K-HARİTASI YAPISININ TANIMLANMASI

K-haritası, birbirine bitişik hücre veya kare dizilerinden oluşan kare veya dikdörtgen şeklinde iki boyutlu bir doğruluk tablosudur. Şekil 3-16*a*, *b* ve *c*, basit bir iki-girişli VE DEĞİL kapısının doğruluk tablosunu ve iki türlü Karnaugh haritası çizimini göstermektedir.



Şekil 3-16 Bir VE DEĞİL kapısının Karnaugh haritasının yapılması. *(a)* Doğruluk tablosu. *(b)* Harf değişkenli Karnaugh haritası. *(c)* 1 ler ve 0 larla Karnaugh haritası.

Giriş değişkenlerinin her mümkün birleşimi için haritada bir kare vardır. İki değişken dört kare verir, üç değişken sekiz kare verir ve dört değişken onaltı kare verir. O halde n değişkenli bir boole fonksiyonu 2^n hücre veya kareden oluşan bir *K*-haritası gerektirir. 3 ve 4 değişkenli ifadeler, bunların doğruluk tabloları ve karşılık gelen *K*-Haritalarının örnekleri için Şekil 3-17a ve b ye bakın.



Şekil 3-17 *(a)* Bir üç-değişkenli ifade örneği, doğruluk tablosu ve karşılık gelen *K*-haritası. *(b)* Bir dört değişkenli ifade örneği, doğruluk tablosu ve karşılık gelen *K*-haritası.

Haritadaki her kare dıştan isimlendirilmiş bir satır ve sütunun kesim noktasındadır. Böylece her hücre veya kare, kesişen satırın ve sütunun isimleri ile belirlenen tek bir adrese sahiptir.

Satır ve sütunlar, her seferinde sadece bir basamağın değiştiği bir ikilik dizi kullanılarak isimlendirilirler. Örneğin, Şekil 3-17, *AB* dizisini 00,01, 11, 10

olarak göstermektedir.(Diğer ikilik kod çeşitleri üzerine olan ilerideki bir kısım bunu Gray kod dizisi olarak tanımlar.) Bu isimlendirme, herhangi bir karenin tam adresinin (satır ismi artı sütün ismi), komşusu olduğu herhangi bir karenin tam adresinden sadece bir basamakta farklı olan bir harita oluşturur. Ayrıca n giriş değişkenli boole ifadeleri için, bu değişkenlerden r tanesinin satır isimleri ve *n-r* tanesinin sütun isimleri için kullanıldığına dikkat edin. Herhangi bir satır veya sütunun adres ismi için ikiden fazla değişken kullanılmaz.

Her kutunun içindeki mantık durumu, bulunduğu sütunun üstündeki ve satırın solundaki giriş değişkenleri için olan çıkış durumudur. Diğer bir deyişle, her kutunun (hücrenin) içine, satır ve sütun adreslerinin belirlediği giriş şartlarını orijinal boole fonksiyonunda yerine koyarak elde edilen 1 veya 0 değeri yazılır. Örneğin, Şekil 3-16b ve c nin sağ alt köşesindeki kutu, eğer *A* yüksek ve *B* yüksekse, çıkışın alçak olduğunu gösterir.

K-HARİTASINI BASİTLEŞTİRMEK : Şimdi bir K-haritasının ne olduğunu bildiğinize göre, bir sonraki soru bundan nasıl basitleştirilmiş ifade elde edileceğidir.

1. Önce *K*-haritasını kurun ve daha önce anlatıldığı gibi 1 leri ve 0 ları girin.
2. Daha sonra *K*-haritasında komşu olan 1 leri inceleyin. İki hücrenin adresleri, bir basamaktan daha fazla farklı değil ise bu hücrelerin komşu olduğuna dikkat edin.Bu nedenle çapraz kareler komşu değildir. Ancak herhangi bir satırın veya sütunun uzak kenarları komşudur. Hiçbir yönde komşusu olmayan tek 1 leri daire içine alın.
3. Kalan 1'ler, aşağıdaki kurallara göre ikişer, dörder ve sekizerli gruplar halinde daire veya halka içine alınır.
   a) Bu gruplamalar (halkalar) 2 nin tam kuvvetlerinden (2^n) oluşur.
   b) Gruplamalar kare veya dikdörtgen yapısında olacaktır.
   c) Halkalar mümkün olduğunca büyük olmalıdır.
   d) Her 1, en az bir halka içinde yer almalıdır, ancak bir 1, eğer daha büyük 2^n grubu oluşmasını sağlıyorsa, birden fazla halka içinde kullanılabilir.
   e) Mümkün olduğunca az halka olmalıdır, fakat bütün 1 ler en az bir defa bir halka içinde yer almalıdır.
   f) Halkalar haritanın sınırlarını (kenarlarını) içine alabilir, çünkü sınırların adresleri komşudur.
4. Her halka, son basitleştirilmiş çıkış ifadesi için bir çarpım terimine çevrilecektir.

a) Halka içinde bir kareden diğer bir kareye hareket ederken değişen adres basamakları çarpım teriminde ihmal edilecektir (kullanılmayacaktır).

b) Halka içinde bir kareden diğer bir kareye hareket ederken değişmeyen adres basamakları çarpım teriminde gereklidir (kullanılacaktır). Sabit 1 ler çarpım teriminde yüksek (tümleyeni alınmamış) olarak yer alır. Sabit 0 lar çarpım teriminde alçak (tümleyeni alınmış) olarak yer alır.

Örneğin, Şekil 3-16*b* deki haritada $\overline{A}$ ın sağındaki halka içine alınmış iki kutudaki yükseklere (1 lere) bakarsanız, *A* alçak ise, *B* alçak veya yüksek ne olursa olsun, çıkış yüksektir. Bu nedenle *A* alçak olduğu sürece *B* girişinin durumu önemli değildir. Bu durum için, kısmi çıkış ifadesini elde etmek üzere, giriş durumu önemli olmayan değişkenleri ihmal eder ve yüksek çıkışı veren değişkenin giriş durumunu yazarsınız: $Y=\overline{A}$. Şimdi, $\overline{B}$ sütunundaki halkalanmış yükseklere bakarsanız, *B* alçak olduğu sürece A nın durumunun önemli olmadığını görebilirsiniz: $Y=\overline{B}$. Haritadaki bir 1 in birden fazla terimde içerilebileceğini ve son ifadede bütün 1 lerin en az bir terimde içerilmesi gerektiğini unutmayın.

5. Son olarak, bütün terimlerin VEYA sını oluşturun. Şekil 3-16*b* ye tekrar bakarak, bütün çıkışlar hesaba alınmış olduğuna göre, eğer *A* alçaksa veya *B* alçaksa çıkışın yüksek olduğunu görebilirsiniz, yani çıkışın tam ifadesi

$$Y = \overline{A} + \overline{B}$$

olarak yazılabilecektir.

Şekil 3-16c yi bu çıkış ifadesini elde etmek üzere aynı şekilde kullanabilirsiniz. Bu ifade ($Y = \overline{A}+B$), hâlâ bir VE DEĞİL kapısının ifadesine benzememektedir, ancak De Morgan'ın teoreminden girişleri ters çevrilmiş bir VEYA kapısının bir VE DEĞİL kapısına denk olduğunu hatırlayın:

$$\overline{A} + \overline{B} = \overline{A \cdot B}$$

Şekil 3-16a' daki doğruluk tablosunun bir VE DEĞİL kapısına ait olduğunu farketmemiş ve doğruluk tablosundan çarpımlar toplamını, yani minterm ifadesini, doğrudan yazmış olsaydınız.

$$Y = \overline{A}.\overline{B} + \overline{A}.B + A.\overline{B}$$

elde edecektiniz. Boole cebiri kuralları ifadeyi bildiğimiz şekle indirgemek için kullanılır, ancak Karnaugh haritası yöntemi daha doğrudandır. Daha çok giriş değişkeni olan durumlarda, bu, çok daha açık olacaktır.

Şimdi Şekil 3- 17*a* daki üç giriş değişkenli örneği inceleyin.

1. *K*- haritasının kurulmasından sonra komşusu olmayan tek 1 ler aranır.

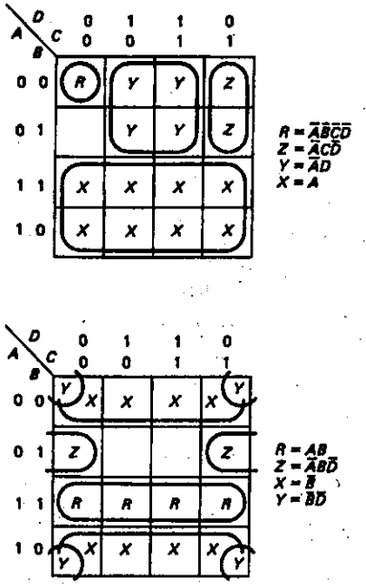
2. Hiç tek 1 olmadığına göre, sadece komşu olan çiftler (bir 1 sadece diğer bir tek 1 e komşu) halkalanır. İki çiftin halkalandığına ve halkaların üst üste bindiğine dikkat edin.

3. Bu noktada bütün 1 ler hesaba alınmış olduğuna göre, çarpım terimlerini yazmaya hazırız demektir. Haritadan *A* alçak ve *B* yüksekse, *C* alçak veya yüksek olabilir böylece $A\overline{B}$ terimini yazabilirsiniz. Diğer halka için *C* alçak ve *B* yüksekse, *A* alçak veya yüksek olabilir, böylece $B\overline{C}$ terimini yazabilirsiniz.

4. Son olarak, bu iki terim VEYA lanarak son çıkış ifadesi oluşturulur :

$$Y = A\overline{B} + B\overline{C}$$

Şekil 3- 18 deki dört- değişkenli Karnaugh haritaları, komşu kutuların uygun şekilde gruplandırılarak bir ifadedeki değişken sayısını nasıl azalttığını göstermek için kullanılacaktır. Dört değişkenli bir harita, daha önce belirtildiği gibi, onaltı kutu gerektirir. Eğer bir R ifadesinin R isimli kutuda olan bir tek 1 i varsa, bu tek *R* kutusuna ait $R = A\,\overline{B}\,\overline{C}\,\overline{D}$ ifadesini oluşturmak için dört değişken de gerekir.



Şekil 3- 18 Bir terimdeki veya ifadedeki değişken sayısını azaltmak için Karnaugh haritalarındaki bazı mümkün gruplamalar.

Bir ifadeye ait haritada, iki komşu kutuda 1 varsa, (Şekil 3- 18*a* daki Z ler gibi), bu takdirde bu terimi tanımlamak için sadece üç değişken gerekir, çünkü değişkenlerden biri 0 da olabilir, 1 de olabilir. Gösterilen örnekte B önemli değildir ve $Z = A\,C\,D$ olur. Eğer dört tane komşu 1, yatay olarak, düşey olarak veya gösterildiği gibi bir blokla gruplandırılırsa, iki değişken yok edilir. Gösterilen örnekteki Y nin ifadesi sadece $Y = \bar{A}D$ dir. X ifadesi için, gösterildiği gibi sekiz kutu birlikte gruplandırılırsa, üç değişken yok olur ve Şekil 3- 18 deki örnekteki sonuç ifade sadece $X = A$ olur.

Karnaugh haritası hem yatay hem de düşey yönlerde bir silindir etrafına sarılmış gibi düşünülebileceğinden, Şekil 3- 18*b*, çıkış ifadesini basitleştirmek üzere kutuların başka şekillerde gruplandırılmasını göstermektedir. Haritadaki bütün 1 lerin çıkış ifadesindeki terimlerden birinde içerilmesi gerektiğini ve en basit çıkış ifadesi için, uygun büyüklükteki (2 nin kuvvetleri) gruplarda mümkün olan en fazla sayıda 1 bulunması gerektiğini unutmayın.

Bir 4 - giriş değişkenli ifadenin veya doğruluk tablosunun *K*- haritası ile basitleştirilmesine ait basamakları gösteren bir örnek için Şekil 3-17*b* ye bakın.

1. *K*- haritası daha önce bahsedildiği gibi kurulur ve uygun karelere 1 ler girilir.
2. Daha sonra haritada tek 1 ler aranır (komşusu olmayanlar). Şekil 3- 17*b* de bir tane tek 1 bulunmuş ve halkalanmıştır.
3. Bundan sonra, daha büyük gruplar içine konmayacak ikili gruplar aranır. Şekil 3- 17*b* de böyle çiftler yoktur.
4. Bu, 4- giriş değişkenli bir harita olduğuna göre, şimdi de 8 li mümkün gruplamayı ararız. Bu durumda 8 li grup yoktur.
5. Hâlâ halkalanmayan 1 ler olduğuna göre hiç halkalanmamış 1 lerin yanısıra daha önce halkalanmış 1 leri de içermesi mümkün olan 4 lü grupları ararız. Bir tane bulur ve halkalarız.
6. Hâlâ daha önceki basamaklarda halka içine alınmamış çiftler kalmışsa bunlar halkalanır. Şekil 3- 17*b* deki haritadaki örnekte bütün 1 ler halkalanmıştır.
7. Şimdi oluşturulan her halka için çarpım terimini yazarız. Tekli halkaya ait terim $A\,B\,\bar{C}\,D$ dir. Dörtlü halkada *B* yüksek ve *C* yüksekken *A* ve *D* alçak veya yüksek olabilir ve bu nedenle terimde kullanılmazlar. Böylece *BC* çarpımını elde ederiz.
8. Son olarak çarpım terimleri VEYA lanarak basitleştirmiş çıkış ifadesi

   $Z = A\,\bar{B}\,\bar{C}\,\bar{D} + B\,C$ elde edilir.

Bir K- haritasını, doğrudan doğruluk tablosundan çizmekle, doğrudan boole ifadesinden çizmek arasındaki farklılıkları göstermek üzere son iki basitleştirme örneği aşağıda verilmiştir.

DOĞRULUK TABLOSUNDAN K- HARİTASINA : Şekil 3- 19*a*, bir doğruluk tablosu ve buna ait basitleştirilmemiş boole ifadesini göstermektedir. Şekil 3- 19b, bu fonksiyona ait doğruluk tablosundaki her 1 çıkış terimine karşılık bir 1 içeren Karnaugh haritasını göstermektedir. Haritanın ortasındaki yatay dört tane 1, $\bar{A}B$ terimini vermek üzere gruplandırılabilir. Haritanın altındaki iki tane 1 de $A\bar{B}\bar{C}$ terimini vermek üzere gruplandırılabilir. Bu iki terim toplanarak (VEYA lanarak), toplam, çarpımlar toplamı şeklindeki ifade elde edilir:

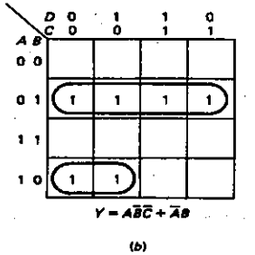
$$Y = A\,B\,\bar{C}\,\bar{D} + \bar{A}\,B$$

Bu ifadenin kapılarla gerçekleştirilmesi 6 terim veren orijinal ifadeden çok daha kolaydır. Şekil 3- 19c basitleştirilmiş ifadeye ait devrenin sadece VE DEĞİL ve ters çeviriciler kullanılarak nasıl kurulabileceğini göstermektedir.

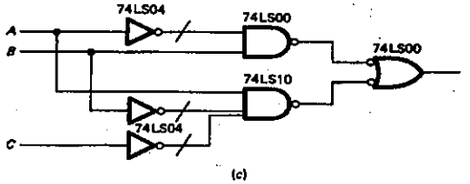
A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

$= \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}BC\bar{D} + \bar{A}BCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}C\bar{D}$

(a)



(b)



(c)

Şekil 3- 19 *(a)* Çıkış ifadesinin basitleştirilmesi gereken doğruluk tablosu. *(b)* Karnaugh haritası ve basitleştirilmiş çıkış ifadesi. *(c)* VE DEĞİL kapıları ve ters çeviricilerle kurulmuş basitleştirilmiş devre.

BOOLE İFADESİNDEN K- HARİTASINA : Bir mantık fonksiyonunu doğrudan boole ifadesinden basitleştirmek için, önce gerekli sayıdaki değişken için bir Karnaugh haritası çizip uygun şekilde isimlendirin. Sonra boole ifadesindeki her terim için Karnaugh haritasında karşılık gelen kareye bir 1 koyun. Karnaugh haritasından basitleştirilmiş ifade yukarıda gösterildiği gibi elde edilebilir.

Örnek olarak aşağıdaki boole ifadesini ele alalım;

$$Y = A\bar{B}\bar{C} + \bar{A}\bar{B}C + AB\bar{C} + A\bar{B}\bar{C}$$

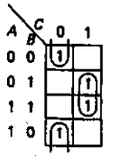
Bu ifade üç değişken içermektedir, o halde 8 kutulu bir Karnaugh haritası gerekmektedir. Şekil 3- 20 orijinal boole ifadesindeki her terime karşılık gelen her kutuya bir 1 konarak elde edilen haritayı göstermektedir. $A\bar{B}\bar{C}$ ve $\bar{A}\bar{B}\bar{C}$ ye ait kutular halkalanarak $\bar{B}\bar{C}$ elde edilmiştir. Diğer iki terim de halkalanarak BC ifadesini verir. Sonuç, bulunan bu terimlerin toplamıdır.

$$Y = B\bar{C} + BC$$

Bu ifadenin

$$Y = B\overline{\oplus C}$$ *(B özel VEYA DEĞİL C)*

olarakta yazılabildiğini not edin.



Şekil 3- 20 Üç- değişkenli bir ifadeyi basitleştirmek için kullanılan bir Karnaugh haritası.

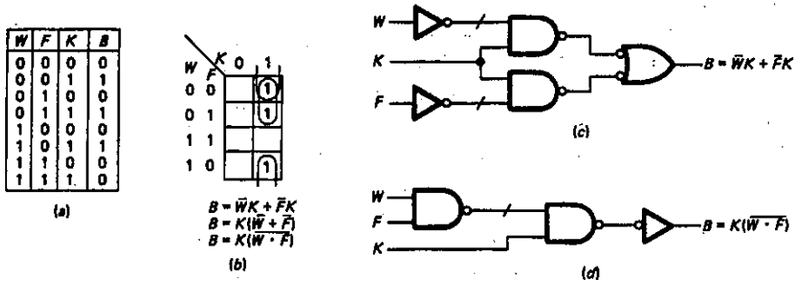
BASİT MANTIK KAPI DEVRELERİNİN SENTEZİ

Mantık devrelerinin analizi, eğer onların gerçek dünyada iş yapmak üzere nasıl kullanıldıklarını anlarsanız çoğunlukla daha kolaydır. Bu kısım, bu tür iki işin, bir otomobil koltuk kemeri uyarı zili ve bir gazla çalışan sıcak hava ev kaloriferinin kontrolörünün, mantık kapılarıyla nasıl yapıldığını göstermektedir.

OTOMOBİL KOLTUK KEMERİ UYARI SİSTEMİ

Otomobil koltuk kemeri uyarı sistemi şu şekilde çalışır. Üretici firma, eğer koltuğun üzerinde ağırlık varsa, kontak anahtarı çevrilmişse ve kemer bağlanmamışsa 15 s süreyle bir zilin çalmasını istemektedir. Ayrıca, zil ve algılayıcılardan ikinci bir fayda sağlamak üzere koltuk üzerinde ağırlık yoksa ve anahtar kontağın içindeyse zilin çalması istenmektedir. Bu sürücünün anahtarları arabada bırakıp dışarı çıkmasını önlemek için bir uyarı olarak kullanılacaktır. Mantık devresi, kontak anahtarına sadece açık ve kapalı konumlarında çalışacak şekilde bağlanabilir, aksesuvar konumuna bağlanmaz. Bu bağlantı, bir kimse sadece arabada oturmak ve ay ışığında radyo dinlemek isterse zilin çalmasını önlemiş olur.

Bu sistemdeki değişkenler şunlardır: Koltuktaki ağırlık W, koltuk kemerinin bağlı olma durumu F, anahtarın açık veya kapalı durumda olması (aksesuvar konumunda değil) K ve zil B. Şekil 3.21*a*, bu devre için mümkün bir doğruluk tablosu göstermektedir.

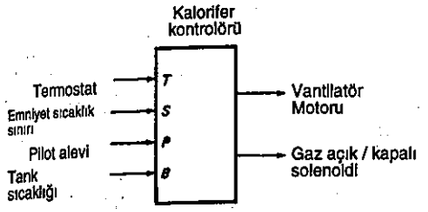


Şekil 3- 21 Bir otomobil emniyet kemeri uyarı mantık devresinin sentezini yapmak için kullanılan Karnaugh haritaları. *(a)* Doğruluk tablosu. *(b)* Karnaugh haritası. *(c)* Çarpımlar- toplamı devre. *(d)* Daha da basitleştirilmiş devre.

Doğruluk tablosundaki bir 1, ilgili değişkenin "aktif " durumunu gösterir. Şekil 3- 21*b*, Karnaugh haritasını ve elde edilen basitleştirilmiş ifadeyi göstermektedir. Karnaugh haritasından elde edilen çarpımlar toplamı şeklindeki ifade, Şekil 3- 21*c* de görüldüğü gibi üç VE DEĞİL kapısı ve iki ters çevirici ile gerçekleştirilebilir. Diğer taraftan, eğer çarpımlar toplamı şeklindeki ifade, Şekil 3- 21*b* de görüldüğü gibi paranteze alınır ve De Morgan'ın teoremi kullanılırsa, sonuç Şekil 3- 21*d* de görüldüğü gibi sadece üç VE DEĞİL kapısıyla gerçekleştirilebilir. Bu, Karnaugh haritaları ve boole cebiri birleşiminin bir devre için gerekli IC paketlerinin sayısını azaltmakta nasıl kullanıldığını göstermektedir.

GAZLI SICAK HAVA KALORİFERİ KONTROLÖRÜ

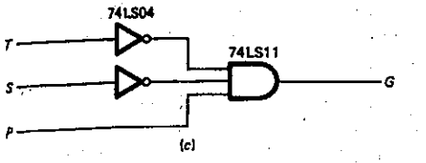
Bu gibi bir sistemi anlamadaki ilk iş, giriş ve çıkışları tanımlamaktır. Şekil 3- 22*a* 'da gördüğümüz gibi, kontrolörün dört girişi ve iki çıkışı vardır.



(a)

Girişler				Çıkışlar		
T	*S*	*B*	*P*	*F1*	*G*	*F2*
L = SOĞUK	L = OK	L = SOGUK	L = KAPALI	L = KAPALI	L = KAPALI	L = KAPALI
L	L	L	L	L	L	L
L	L	L	H	L	H	L
L	L	H	L	H	L	H
L	L	H	H	H	H	H
L	H	L	L	L	L	L
L	H	L	H	L	L	L
L	H	H	L	H	L	H
L	H	H	H	H	L	H
H	L	L	L	L	L	L
H	L	L	H	L	L	L
H	L	H	L	L	L	H
H	L	H	H	L	L	H
H	H	L	L	L	L	L
H	H	L	H	L	L	L
H	H	H	L	L	L	H
H	H	H	H	L	L	H

(b)



(c)

Şekil 3- 22 Mantık kapılı kalorifer kontrolörü. *(a)* Blok şeması. *(b)* Doğruluk tablosu. *(c)* Gaz kontrol devresi.

Termostattan gelen giriş *T* için, bir yükseğin anlamı, oda sıcaklığının ayarlanan sıcaklığın üzerinde olduğu ve bir alçağın anlamı da oda sıcaklığının ayarlanan sıcaklığın altında olduğu şeklindedir. Emniyet sıcaklık sınır algılayıcısı *S* den gelen bir alçak, sıcak hava tankının güvenli çalışma sıcaklığında olduğunu, fakat bir yüksek, aşırı sıcaklık olduğunu ve gazın kesilmesi gerektiğini gösterir. Pilot alevi emniyet algılayıcısı *P* den gelen bir yüksek, pilot alevinin yandığını ve ana brülörün açıldığında alevleneceğini gösterir. Pilot alevi emniyet algılayıcısından gelen bir alçak, pilot alevinin yanmadığı ve gazın açılmasına müsade edilmemesi gerektiği anlamına gelir. (Bu güvenlik öntedbiri, genellikle pilot alevini yakarken yaklaşık bir dakika basılı tutmanız gereken kırmızı düğme ile gerçekleştirilir.) Hava tankı sıcaklık algılayıcısı B, fırındaki hava, dolaşım vantilatörünü çalıştıracak sıcaklığa ulaştığında, bir yüksek verir. Vantilatörü çalıştırma kontrölü F ve gaz vanası solenoidini açma kontrolu *G* de iki çıkıştır. Her ikisi için de yüksek (1) aktif demektir.

Şekil 3- 22b bu girişler ve çıkışlar için mümkün bir doğruluk tablosunu göstermektedir. İlk bakışta, bu doğruluk tablosu karışık görülmektedir, fakat daha yakından bakıldığında sadece iki giriş birleşiminin GAZ AÇIK çıkışını yüksek yaptığını farkedersiniz.

$$G = \overline{T}.\overline{S}.\overline{B}.P + \overline{T}.\overline{S}.B.P$$

Bu fonksiyon için bu haliyle kapılar kullanarak devreyi kurabilirsiniz veya aşağıdaki yöntemlerden biri ile ifadeyi basitleştirmeye çalışabilirsiniz.

İÇGÜDÜSEL YAKLAŞIMLA BASİTLEŞTİRME : Doğruluk tablosunda yüksek G çıkışı veren iki satırı karşılaştırırsanız, *T*, *S* ve *P* girişlerinin aynı olduğunu görürsünüz. Bir birleşimde *B* alçaktır ve diğer birleşimde *B* yüksektir. Açıkcası, ifadeniz için *B* nin giriş durumunun ne olduğu farketmez. *G* nin ifadesinde bütün baktığınız *T. S. P* dir, böylece basitleştirilmiş ifadeyi,

$$G = \overline{T.S.P}$$

olarak yazabilirsiniz.

BOOLE CEBİRİNİ KULLANARAK BASİTLEŞTİRME : G nin yukarıdaki orijinal ifadesiyle başlayarak $\overline{T}.\overline{S}.\overline{P}$ parantezine alırsınız. Bu takdirde

$$G = \overline{T}.\overline{S}.P.\ (B+\overline{B})$$

elde edilir. $B+\overline{B}$, 1 e eşit olduğuna göre, ifade

$G = \overline{T}.\overline{S}.\overline{P}$

olarak basitleşir ki bu da yukarıdaki içgüdüsel sonuçla aynıdır. Bu son ifade, oda sıcaklığı alçaksa, tank sıcaklığı güvenliyse ve pilot alevi açıksa gazı açacağınızı söylemektedir. Bu devre, Şekil 3- 22c de görüldüğü gibi kapılarla kurulabilir.

VANTİLATÖRÜ AÇ için olan ifade biraz daha karışıktır, fakat şu şekilde açıklanabilir. Önce, yüksek çıkış veren giriş birleşimlerine ait bütün terimleri toplayın:

$$F1 = \overline{T}.\overline{S}.B.\overline{P} + \overline{T}.S.B.\overline{P} + T.\overline{S}.B.P + T.S.B.P$$

Bütün terimler T. B içermektedir, O halde $T.B$ parantezine alarak

$$F1 = T.B\ (\overline{S}.\overline{P} + S.\overline{P} + \overline{S}.P + S.P)$$

elde edilir. Parantezler $S.\ P$ nin bütün birleşimlerini içerdiğine göre, parantezlerin içi 1 e indirgenir ve ifadenin tamamı

$$F1 = T.B$$

ye basitleşir. Bu ifade basitçe, oda ısısı alçaksa ve tank sıcaksa, vantilatörü çalıştırmak istediğinizi söylemektedir.

Termostat yükseldikten ve gazı kapattıktan sonra vantilatörün çalışmaya devam ederek tankı soğutmasını istiyorsanız, Şekil 3- 22*b* deki F_2 sütunundaki gibi, vantilatör sütununa dört tane daha yüksek *(H)* ekleyebilirsiniz. F_2 için elde edilecek sonuç,

$$F_2 = \overline{T}.\overline{S}.B.\overline{P} + \overline{T}.S.B.\overline{P} + T.\overline{S}.B.\overline{P} + T.S.B.\overline{P} + \overline{T}.\overline{S}.B.P + \overline{T}.S.B.P + T.\overline{S}.B.P + T.S.B.P$$

olur. Bütün terimlerde B vardır, böylece B parantezine alarak

$$F_2 = B(\overline{T}.\overline{S}.\overline{P}. + \overline{T}.S.\overline{P} + T.\overline{S}.\overline{P} + T.S.\overline{P} + \overline{T}.\overline{S}.P + \overline{T}.S.P + T.\overline{S}.P + T.S.P)$$

elde edilir. parantez içindeki ifade $T.S.P$ nin bütün birleşimlerini içerdiği için, her zaman parantez içindeki terimlerden biri 1 olacaktır. O halde parantezin tamamı 1 e eşittir. Diğer bir deyişle, T,S ve P girişlerinin, durumlarının ne olduğu fark etmez. Böylece ifadenin tamamı

$$F_2 = B$$

ye indirgenmiş olur, ki bu ifade $F_1 = \overline{T}.B$ den de daha kolay kurulur. Hiçbir kapı gerekmemektedir. B girişinden VANTİLATÖR MOTOR rölesine doğrudan bir bağlantı yaparsınız. Sonuç, basit olarak "Tank ne zaman sıcaksa vantilatörü çalıştır" demektedir.

KARNAUGH HARİTALAMASINI KULLANARAK BASİTLEŞTİRME

Eğer VANTİLATÖRÜ AÇ için olan ilk ifadeyi gerçekleştirmek için gerekli olan kapı sayısını, son ifade için gerekli kapı sayısı ile karşılaştırırsanız, basitleştirmenin sebebi açık olacaktır. Şimdi, geri gidip, Şekil 3- 22*b* deki kalorifer kontrolörünün doğruluk tablosunu bir Karnaugh haritası üzerinde deneyin. Değişkenleri her zaman *A, B, C,* v.b. gibi isimlendirmek zorunda olmadığınıza dikkat edin. Gaz çıkış ifadesinin haritasını kendiniz kurun ve sonra sonuçlarını Şekil 3- 23*a* dakilerle karşılaştırın. Halkalanan iki 1 den çıkış ifadesinin

$$G = \overline{T}.\overline{S}.P$$

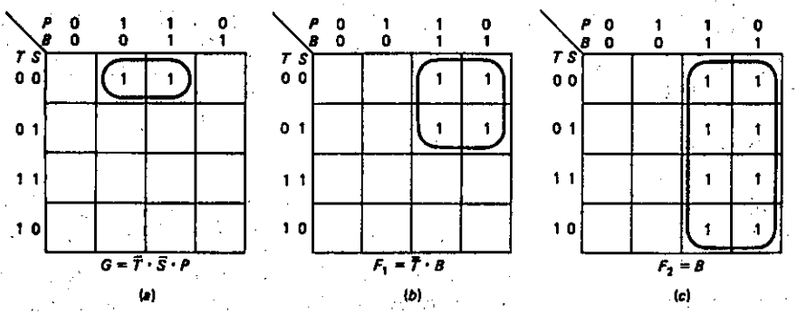
olduğunu hemen görebilirsiniz. Şimdi, ilk vantilatör ifadesinin haritasını yapmaya çalışın ve sonuçlarınızı Şekil 3- 23*b* ile karşılaştırın. Şekil 3- 23*b* deki halkalanmış dört 1, size basit çıkış ifadesi

$$F_1 = \overline{T}.B$$

yi derhal vermektedir. Şekil 3- 23*c* de, F_2 ifadesinin bir haritası gösterilmiştir. Sekiz kutu birlikte alınabileceğinden, çıkış ifadesi

$$F_2 = B$$

olur.



Şekil 3-23 Kalorifer kontrolörünün gaz ve vantilatör ifadelerini basitleştirmede kullanılan Karnaugh haritaları. *(a)* Gaz. *(b)* Vantilatör 1. *(c)*Vantilatör 2.

Karnaugh haritalarının kullanımı, bu bölümün daha sonraki kod çeviriciler kısmında daha derinlemesine gösterilecektir, fakat önce temel kapıların daha başka ilginç uygulamaları üzerinde duracağız.

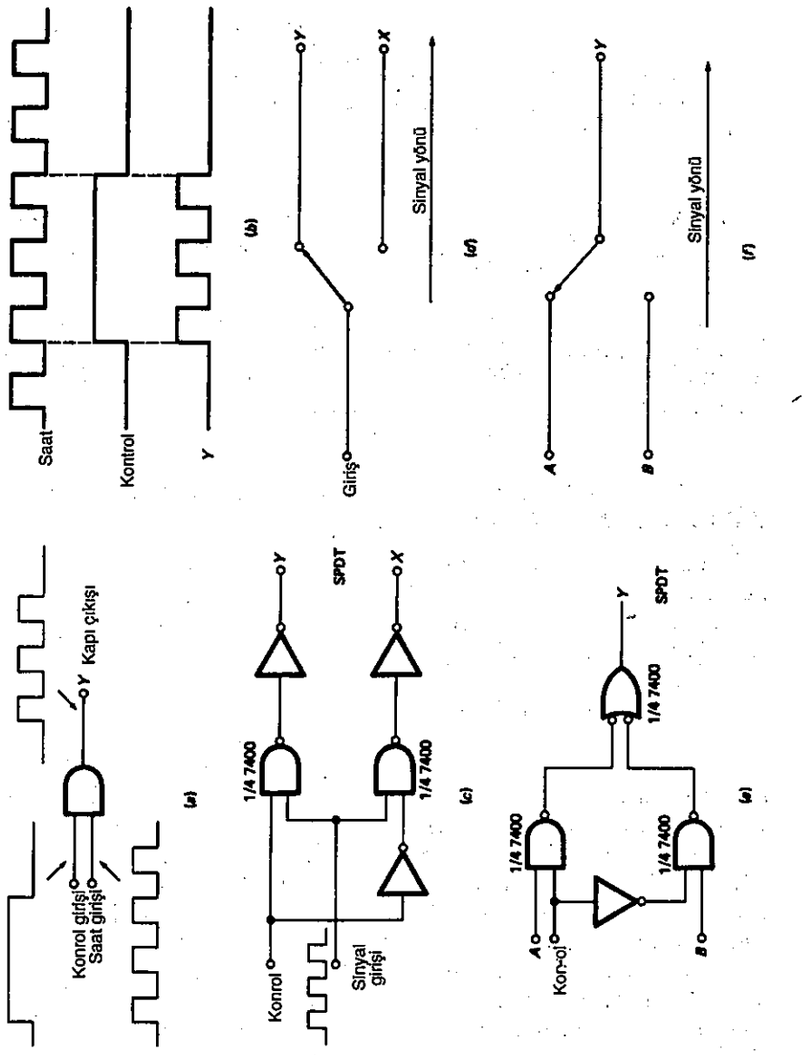
ANAHTARLAR ,VERİ SEÇİCİLER (ÇOĞULLAYICILAR YANİ MULTİPLEKSÖRLER) VE VERİ DAĞITICILAR (YOL ÇOKLAYICILAR YANİ DEMULTİPLEKSÖRLER)

Bu kısımda kapıları nasıl mantık kontrollu anahtarlar olarak kullanabileceğinizi, birçok veri hattının nasıl tek bir hat üzerinden zaman- çoğullanabileceğini ve beşe kadar giriş değişkeni olan herhangi bir mantık ifadesinin basit bir IC paketiyle nasıl kurulabileceğini göstereceğiz.

BASİT KAPILARIN VE VE-DEĞİL KAPI ANAHTARLARININ ZAMANLAMA DİYAGRAMINA BAKIŞ

Bu noktaya kadar, mantık kapı devrelerini tanımlamak ve analizini yapmak için doğruluk tabloları kullanılmıştır. Sürekli değişen girişlere sahip mantık devreleri için giriş ve çıkış durumları arasındaki ilişkiyi temsil etmenin diğer bir yolu, zamanlama diyagramlarıdır. Bir zamanlama diyagramı giriş ve çıkış gerilimlerinin zamana göre çizilerek karşılaştırılmalarından ibarettir. Şekil 3- 24*a* darbe girişli basit bir VE kapısını göstermektedir. Bu girişler ve karşılık gelen çıkış için olan zamanlama diyagramı Şekil 3- 24*b* de gösterilmiştir. Çıkışın her an bir VE kapısının doğruluk tablosunu izlediğine dikkat edin. Çıkış sadece saat ve kontrol girişlerinin her ikisi de yüksek olduğunda yüksektir. Girişlerden birisi alçak ise, diğer girişin durumu ne olursa olsun, çıkış alçaktır. Zamanlama diyagramının gösterdiği gibi, eğer kontrol hattı yüksekse, bu takdirde darbe, yani saat sinyali, *Y* noktasında görülecektir. Eğer kontrol hattı alçaksa, *Y* de saat görülmeyecektir ve saat girişi ne olursa olsun VE kapısının çıkışı alçak değerde sabit kalacaktır. VE kapısı bir mantık kontrollu anahtar gibi çalışmaktadır. Bir VE DEĞİL kapısı da bir anahtar gibi kullanılabilir, fakat çıkış ters çevrilecektir. Bir VEYA veya VEYA DEĞİL kapısı anahtar olarak kullanıldığında ise kontrol girişi alçak olduğunda çıkış saat sinyalini izleyecektir.

Şekil 3- 24*c*, bir saat sinyalini iki noktadan birine gönderen bir devreyi göstermektedir. Kontrol girişindeki bir yüksek, üst kapıyı yetkilendirir ve sinyali *Y* ye gönderir. Kontrol girişindeki bir alçak, alt VE DEĞİL kapısını yetkilendirir ve sinyali *X a* gönderir. Birçok kullanılışın yanısıra, bu devre Bölüm 4 te anlatıldığı gibi bir sayıcının yukarı- sayma ve aşağı- sayma girişlerine saati göndermek için de kullanılabilir. Devrenin anahtar dengi Şekil 3- 24*d* de gösterilmiştir.

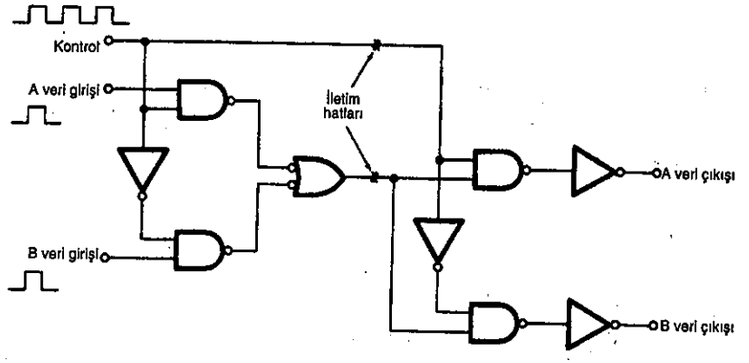


Şekil 3- 24 *(a)* Mantık- kontrollu anahtar olarak kullanılan VE kapısı, SPST(tek- kutuplu tek- bağlantılı anahtar) şeması ve sinyaller. *(b)* a kısmında gösterilen VE kapısı mantık- kontrollu anahtarın zamanlama diyagramı. *(c)* Mantık- kontrollu anahtarlar olarak kullanılan VE DEĞİL kapıları, SPDT (tek- kutuplu çift- bağlantılı anahtar) çıkış şeması. *(d)* c kısmında gösterilen devrenin mekanik anahtar dengi. *(e)* Mantık- kontrollu anahtarlar olarak kullanılan VE DEĞİL kapıları, SPDT giriş şeması. *(f)* e kısmında gösterilen devrenin mekanik anahtar dengi.

Şekil 3- 24e' de gösterilen bir başka anahtar çeşidi, iki girişten birindeki veriyi seçerek *Y* ye göndermek için kullanılır. Şekil 3-24f, denk anahtar devresini göstermektedir. Kontrol hattı yüksek ise veri *A* girişinden seçilir; kontrol hattı alçak ise veri *B* girişinden seçilir. Bu devre 74157 dörtlü iki- girişli veri seçici adı altında TTL için hazır olarak vardır.

ÇOĞULLAYICILAR VE YOL ÇOKLAYICILAR

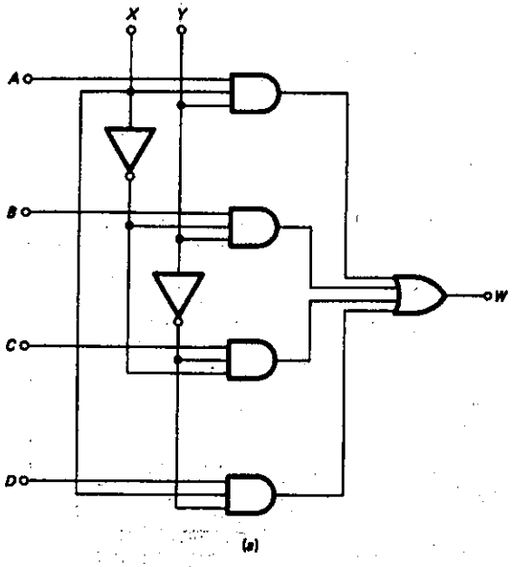
Şekil 3- 24e ve c deki devreler bir iki- girişli çoğullayıcı- yol çoklayıcı devresi yapmak üzere Şekil 3- 25 te gösterildiği gibi birleştirilebilirler. Bu durumda çoğullamak, aynı hat üzerinde her girişten alınan veri örneklerinin gönderilmesi demektir. Önce *A* dan bir örnek gönderilir, sonra *B* den bir örnek, sonra *A* dan bir örnek daha ve *B* den bir örnek daha ve bu böylece devam eder. Kontrol girişindeki bir yüksek frekanslı kare dalga sinyal, çoğullayıcının iki giriş arasında bir ona bir buna değişmesini sağlar. Diğer uçtaki senkronize edilmiş bir yol çoklayıcı, hattan gelen verileri doğru çıkışlara gönderir. Kontrol girişindeki sinyalin frekansı gönderilen verilerinkinden çok daha yüksek olmalıdır. Bu, zaman bölmeli çoğullamanın amacı birçok kaynaktan veri göndermek için gerekli hat sayısını azaltmaktır. İki- girişli örnek için toplam hat sayısında bir kazanç görülmez, fakat çoğullanacak hat sayısı arttıkça, hat sayısındaki kazanç artar.



Şekil 3- 25 Bir hat üzerinde iki veri sinyalini göndermek için VE DEĞİL kapılı anahtar devrelerinin kullanılışı.

Şekil 3- 26, dört- girişli bir çoğullayıcının devresini göstermektedir. Dört girişten hangisinin çıkışa bağlanacağını belirlemek için iki tane veri seçme hattına gerek vardır. Şekil 3- 26*b*, her veriyi seçmek için seçme girişlerinin ne olması gerektiğini göstermektedir. Bazan bu veri seçme girişlerini adres hatları olarak düşünmek daha uygun olmaktadır. Bu takdirde A girişinin adresinin 11 olduğunu, *B* girişinin adresinin 01, vb. olduğunu söyleyebilirsiniz. 74153 ikili dört- girişli çoğullayıcıda bu devrelerden iki tane vardır. Şekil 3- 26 daki gibi, bu devreyi veri çoğullayıcı olarak kullanırsanız dört hattaki verileri iki adres, yani veri seçme, hattı ve bir sinyal hattı üzerinden gönderebilirsiniz. Bu bir hatlık bir kazançtır ve çok etkileyici bir sonuç değildir, fakat şimdi bir sonraki duruma bakalım.

74151, bir sekiz- girişli çoğullayıcıdır. Sekiz girişten birini seçmek için, üç tane veri seçme, yani adres, hattına gerek vardır. Bunun anlamı sekiz kanaldaki veriyi sadece dört hatla (bir sinyal ve üç adres) gönderebileceğinizdir. Telefon konuşmaları gibi analog sinyaller de benzer şekilde CD4051 sekiz- girişli CMOS multipleksör kullanılarak çoğullanabilir.



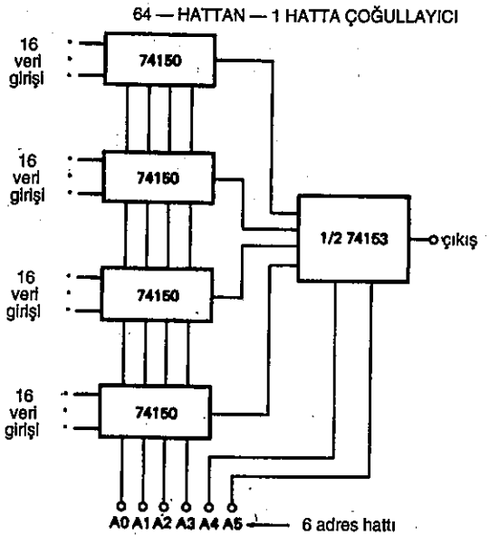
(a)

Seçilecek Giriş	X	Y
A	1	1
B	0	1
C	0	0
D	1	0

(b)

Şekil 3- 26 Dört girişli çoğullayıcı. *(a)* Şema. *(b)* Seçme tablosu

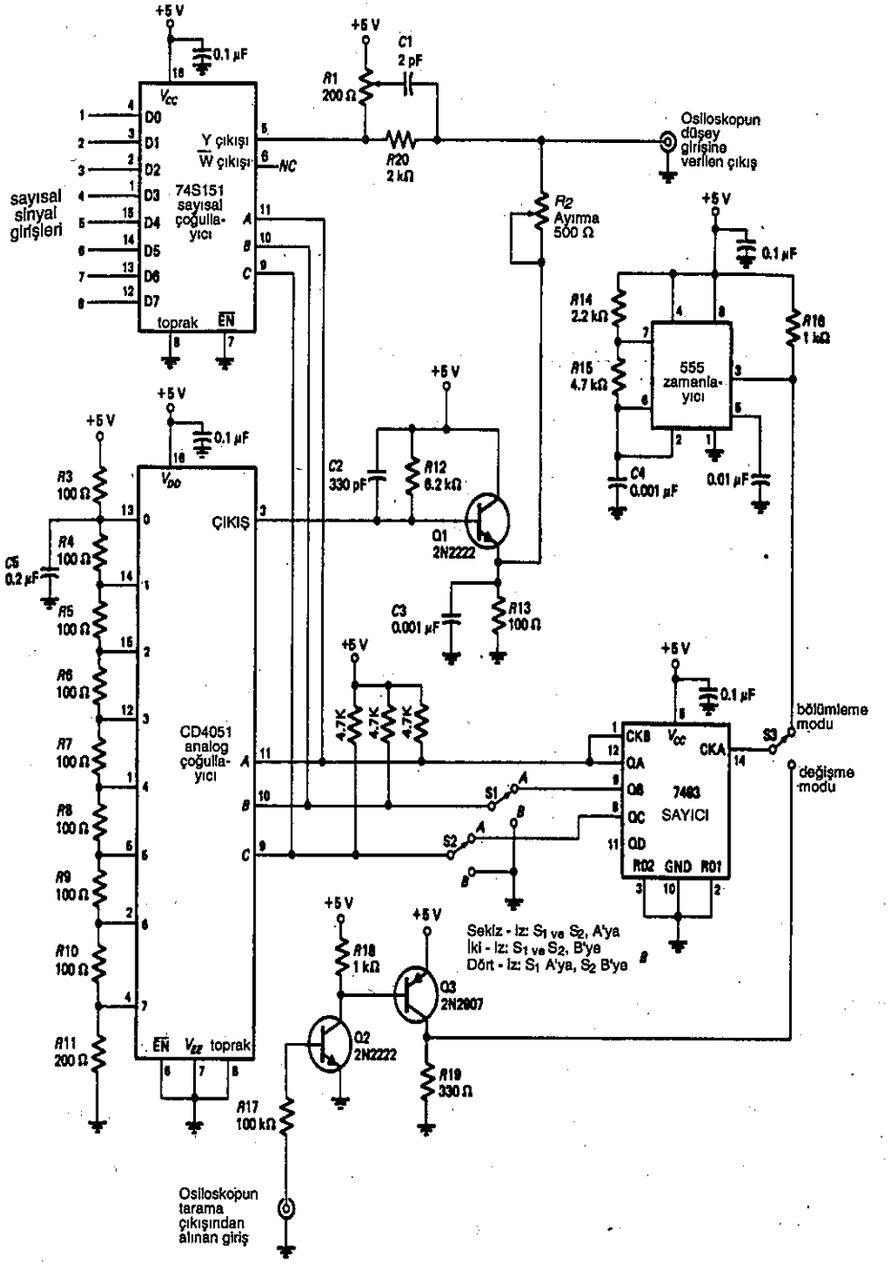
74150 bir 16- girişli çoğullayıcıdır ve girişindeki 16 hattan birisini seçmek için 4 adres hattı vardır. Şekil 3- 27 de gösterildiği gibi multipleksörler birçok hattaki verileri bir tek hatta sıkıştırmak için bir çoğullayıcı *"ağaç"* şeklinde yığılabilir. Alıcı sondaki bir yol çoklayıcı "ağaç"taki verileri paralel hatlara dağıtır. 74154, bir 4- den- 16- hatta yol çoklayıcıdır.



Şekil 3- 27 Altmış- dört- hattan- bir hatta çoğullayıcı.

SEKİZ- GİRİŞLİ OSİLOSKOP ÇOĞULLAYICISI

Bir veri kanalından diğerine değişmek size Bölüm 1 de anlatılan bölümlenmiş- mod osiloskop görüntüsünü hatırlatabilir. Prensip aynıdır. Şekil 3- 28, sekiz kanala kadar sayısal veriyi bir standart osiloskopta gösterebilecek bir devreyi göstermektedir. Devre, bir 74S151 sekiz- girişli sayısal multipleksör kullanarak çoğullama yapar ve bir CD4051 sekiz- girişli analog multipleksör kullanarak ekran üzerindeki izleri ayırmak üzere gerilim basamakları ekler. Bir 555 kararsız multivibratör ana saat görevini yapar ve bir 7493 dört- bit ikilik sayıcı sekiz adresi sürekli tarar.



Şekil 3- 28 Sekiz sayısal giriş sinyalini bir osiloskopta göstermek için devre. Bu devre, McGraw Hill tarafından basılan Electronics dergisinin 14 Ekim 1976 (Cilt: 49) tarihli sayısında bulunan devrenin değiştirilmişidir.

RASTGELE MANTIK FONKSİYONLARININ SENTEZİNDE ÇOĞULLAYICI KULLANILMASI

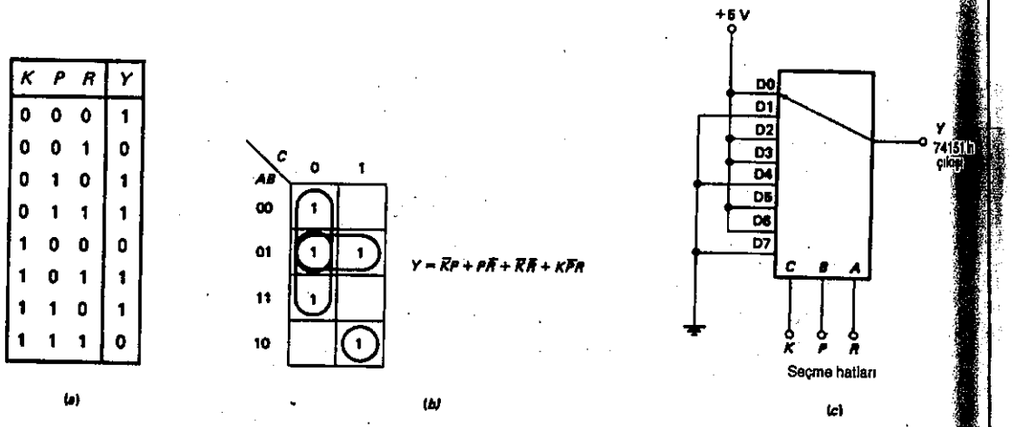
Çoğullayıcı ve yol çoklayıcıların isimlerinin ima ettiklerinden çok fazla kullanış yerleri vardır. Bir çoğullayıcı, daha önce Karnaugh haritaları kullanarak basitleştirdiklerinize benzer, rastgele- mantık ifadeleri denilen fonksiyonların devrelerini kurmak için ayrık kapılar yerine kullanılabilir. Bu şöyle yapılır: Şekil 3- 29*a*, üç değişkenli bir mantık ifadesi

$$Y=\overline{K}\,\overline{P}\,\overline{R}+\overline{K}\,P\,\overline{R}+K\,P\,\overline{R}+K\,\overline{P}\,R+\overline{K}P\,R$$

nin doğruluk tablosunu göstermektedir. Şekil 3- 29b deki bir Karnaugh haritasıyla bu ifadeyi

$$Y=\overline{K}\,P+P\,\overline{R}+\overline{K}\,\overline{R}+K\,\overline{P}\,R$$

şekline indirgeyebilirsiniz, fakat bu ifade hâlâ bir çok kapı paketi gerektirecektir. Şekil 3- 29c bu fonksiyonun tek bir IC paketiyle, 74151 sekiz- girişli multipleksörle, doğrudan doğruluk tablosundan nasıl kurulabileceğini göstermektedir. Giriş değişkenleri doğruluk tablosundaki aynı sırada seçme hatlarına bağlanır. Giriş değişkenleri *K*, *P* ve *R* nin hepsi alçaksa, yani 0 sa, çoğullayıcının *D0* girişini çıkışa bağladığını biliyorsunuz. Bütün yapmanız gereken giriş değişken birleşimi *K*, *P* ve *R*= 0, yani $K\,P\,R$, için Y nin hangi değeri almasını istiyorsanız; D0 girişini, 1 için +5 V a, 0 için toprağa bağlamaktır. Bir sonraki giriş birleşimi *KPR* için *D1* girişi adreslenecektir. Bu giriş için bir alçak, yani 0, çıkışı istediğinize göre *D1* girişini toprağa bağlamanız yeterlidir. Daha sonra, doğruluk tablosunda aşağıya doğru *D* girişlerini +5 V veya toprağa, hangisi uygunsa, bağlayarak devam edin.



Şekil 3- 29 Bir mantık ifadesini gerçekleştirmek için kullanılan bir çoğullayıcı *(a)* Doğruluk tablosu. *(b)* Karnaugh haritası. *(c)* Devre.

Bu devrenin güzel tarafı, üç- girişli herhangi bir ifadeyi hemen hemen doğruluk tablosunu yazmak kadar hızlı bir şekilde kurabilmenizdir. Ayrıca çıkış ifadesini değiştirmek isterseniz, örneğin *KPR* için alçak çıkış isterseniz, bütün yapmamız gereken *D2* giriş ucunu +5 V tan toprağa değiştirmektir. Önceki yöntemlerde, Karnaugh haritasını yeniden yapmanız, ifadeyi yeniden basitleştirmeniz ve devreyi yeniden kurmanız gerekecekti. Hâttâ bu örnek için, IC paketlerini değiştirmeniz gerekecekti. Çoğullayıcı yöntemi ile, bir Karnaugh haritası gerekmez, fakat doğruluk tablosunda bir çok 1 varsa, bir haritayla kontrol ederek, kalorifer kontrolörü örneğindeki *F2=B* de olduğu gibi, çok açık bir basitleştirmeyi kaçırmadığınızdan emin olun. (Bu arada 74151 in ters çevrilmemiş *Y* çıkışı ve bir de ters çevrilmiş *W* çıkışı olduğunu hatırlatalım).

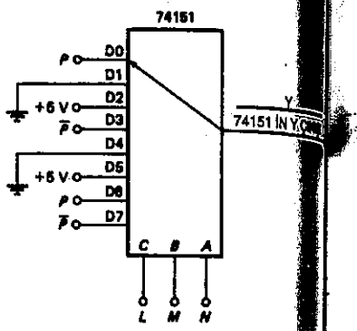
Yukarıdaki tekniği kullanarak, 74150 16- girişli multipleksör, herhangi bir dört giriş değişkenli mantık fonksiyonunu doğrudan sentezleyebilir. 74150, daha büyük ve daha pahalı 24- bacaklı pakette gelmektedir ve çıkışı ters çevrilmiştir. Bir katlama hilesi, herhangi bir dört giriş değişkenli fonksiyonu 74151 gibi bir sekiz- girişli multipleksör ile üretmenizi sağlar.

KATLAMA HİLESİ : Şekil 3-30a, dört- giriş- değişkenli bir ifadenin doğruluk tablosunu göstermektedir ve Şekil 3- 30b, buna ait devrenin sekiz-girişli bir çoğullayıcı kullanarak yapılışını göstermektedir. *L*, *M* ve *N* değişkenleri doğrudan *C*, *B* ve *A* seçme girişlerine bağlanmıştır. *P* değişkenini şu şekilde kullanırsınız. Doğruluk tablosunda, *L*, *M* ve *N* nin 0 olduğu iki giriş durumunda, eğer *P* sıfırsa *Y* sıfırdır, *P* birse *Y* birdir. O halde, *C*=0, *B*=0 ve *A*=0 seçme girişleri *D0* ı çıkışa bağlar. Eğer bu girişe *P* değişkenini bağlarsanız, çıkış, *P* değişkeninin her durumu için doğru olacaktır. L,M ve N nin bir sonraki birleşimi *L*=0, *M*=0 ve *N*=1 dir. *P*=0 ise, *Y*=0 ve *P*=1 ise *Y*= 0 olduğunu görebilirsiniz. O halde *Y*= 0' dır. Bunu, çoğulayıcının *D1* girişini toprağa bağlayarak gerçekleştirebilirsiniz. Şekil 3- 30a' dan görebileceğiniz gibi, *L*=0, *M*=1 ve *N*=0 için, P ister 0, ister 1 olsun, çıkış 1 dir. O halde çoğullayıcının karşılık gelen girişi +5 V a bağlanır. Bir sonraki *L* =0, *M*=1 ve *N*=1 birleşimi *Y*=*P* verir. Basit bir ters çevirici çoğullayıcı *D3* girişine bağlanacak $\overline{P}$ yi üretir. Bu işlemlerin altında yatan fikri anladığınızdan emin oluncaya kadar doğruluk tablosundaki diğer işlemleri yapmaya devam edin. Çoğullayıcının veri girişleri için mümkün girişlerin 0,1,P ve $\overline{P}$ olduğuna dikkat edin. Eğer doğruluk tablosunda bir değişiklik yapmak isterseniz, bütün yapmanız gereken bir veri giriş hattını bunlardan uygun olan birine değiştirmekten ibarettir.

L	M	N	P	Y
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

(a)

L	M	N	P	Y	
0	0	0	0	0	$Y = P$
0	0	0	1	1	
0	0	1	0	0	$Y = 0$
0	0	1	1	0	
0	1	0	0	1	$Y = 1$
0	1	0	1	1	
0	1	1	0	1	$Y = \overline{P}$
0	1	1	1	0	
1	0	0	0	0	$Y = 0$
1	0	0	1	0	
1	0	1	0	1	$Y = 1$
1	0	1	1	1	
1	1	0	0	0	$Y = P$
1	1	0	1	1	
1	1	1	0	1	$Y = \overline{P}$
1	1	1	1	0	



(b)

Şekil 3- 30 Bir mantık ifadesinin, bir çoğullayıcı ve katlama hilesini kullanarak gerçekleştirilmesi. *(a)* Doğruluk tablosu analizi. *(b)* 'Katlanmış' doğruluk tablosu için devre şeması.

74150 onaltı- girişli multipleksörü bu katlama hilesi ile birlikte kullanarak beş giriş değişkenli herhangi bir çıkış ifadesi üretebilirsiniz. Beşten fazla giriş değişkenli ifadeler için mantık tasarımcıları genellikle bir sonraki bölümde anlatılan yalnız- okunur- bellekleri veya programlanabilen- mantık dizilerini kullanmaktadır.

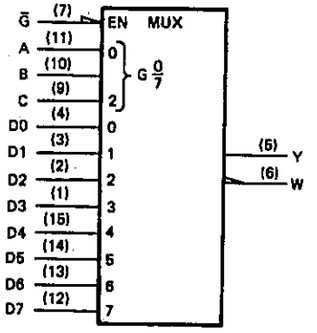
ÇOĞULLAYICILAR VE YOL ÇOKLAYICILAR İÇİN BAĞIMLILIK GÖSTERİLİŞİ SEMBOLLERİ

Ek B de anlatıldığı gibi, IEC- IEEE bağımlılık gösterilişi, her mantık aygı-

tına bütün girişleri ve çıkışları arasındaki ilişkileri açıkça gösteren bir mantık sembolü oluşturma teşebbüsüdür. Daha önce, MSI (orta- ölçekli entegrasyon, middle- scale integration) ve LSI (büyük- ölçekli entegrasyon, large- scale integration) aygıtları sadece bacak numaraları ve bacak isimleri olan dikdörtgen bloklarla gösterilmişti. Çıkışların girişlere nasıl bağımlı olduklarını belirlemek için veri kitabından aygıtın doğruluk tablosunun incelenmesi gerekiyordu. Bağımlılık gösterilişi daha fazla anlaşılır ve kullanılır hale geldikçe, MSI ve LSI aygıtlarının çalışmalarını doğrudan şematik diyagramlarına bakarak anlamak mümkün olacaktır. Bir sonraki kısmı okumadan önce, kullanılan sembollere ve isimlere alışmak için Ek B deki bağımlılık gösterilişi tarifini bir gözden geçirin.

BAĞIMLILIK GÖSTERİLİŞİ KULLANAN BİR ÇOKLAYICI

Şekil 3- 31, bir 74151 sekiz- girişli çoğulayıcının bağımlılık gösterilişi mantık sembolünü göstermektedir.Üst taraftaki MUX, elemanın bir multipleksör (çoğullayıcı) olduğunu gösterir. Girişler sol tarafta, çıkışlar sağ tarafta gösterilmiştir. Elemanın içindeki EN, *yetki (enable)* girişini belirtir. Bu girişteki kama, bu girişin aktif alçak olduğunu gösterir. Eğer bu giriş yüksekse *yetkisiz kılınmışsa, (disabled)*, diğer girişlerdeki sinyaller ne olursa olsun, *Y* çıkışı alçak ve W çıkışı yüksek olacaktır.



Şekil 3- 31 Sekiz- girişli çoğulayıcı 74151 için bağımlılık gösterilişi mantık sembolü.

0 dan 7 ye kadar numaralandırılmış bacaklar *D0* dan *D7* ye kadar olan sekiz veri girişidir. *A, B* ve *C* seçme girişlerinin ikilik kuvvetleri en küçük değerlikli giriş için 0 ile ve en büyük değerlikli giriş için 2 ile gösterilmiştir. Ek *B* de belirtilidiği gibi, *G* harfi VE bağımlılığını gösterir. Bu durumda, *G* den önceki parantez ve 0/7 kesri, çıkışın, seçme girişlerindeki ikilik sayının fonksiyonunun olduğunu belirtir.

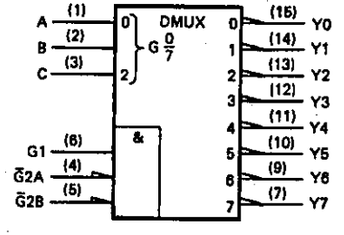
Diğer bir deyişle, eğer yetki girişi aktif yapılırsa *(EN= alçak)*, adreslenen girişte bulunan veri, *Y* çıkışında görülecektir. Üzerindeki kama ile belirtildiği üzere *W* çıkışı, *Y* deki mantık seviyesinin tümleyeni olacaktır.

Karma mantık ve bağımlılık gösterilişi dilinde bir örnek verelim. Eğer EN girişi aktif (alçak), 5 in ikilik kodu seçme girişlerinde ve kutu içinde yanında 5 yazan bacak girişi aktif (yüksek) ise, bu takdirde *Y* çıkışı aktif (yüksek) olacaktır. Eğer 5 girişi aktif değilse (alçaksa), *Y* çıkışı da aktif değildir (alçaktır).

BAĞIMLILIK GÖSTERİLİŞİNİ KULLANAN BİR YOL ÇOKLAYICI

Şekil 3- 32, bir yol çoklayıcı (demultipleksör) olarak kullanılan 74LS138 in bağımlılık gösterilişi sembolünü göstermektedir.Sembolün üst tarafındaki *DMUX*, demultipleksörün kısa yazılışıdır. & işaretiyle belirtildiğine göre *G1*, *G2A* ve *G2B* girişleri birlikte VE lenmiştir. Parantez, *G* ve 0/7 ise, seçme girişlerindeki ikilik sayı ile *G1*, *G2A*, *G2B* fonksiyonları arasında bir VE ilişisi olduğunu belirtir. Diğer bir deyişle, eğer *G1*, $\overline{G2A}$, ve $\overline{G2B}$ aktif ise VE seçme girişlerinde bir ikilik sayı varsa, bu takdirde aynı numaralı çıkış aktif olacaktır. Örneğin, eğer G girişleri aktifse ve *A*, *B* ve *C* seçme girişlerinde 5 in ikilik karşılığı varsa, yanında 5 yazan çıkış aktif (alçak) olacaktır.

74LS138 i bir yol çoklayıcı olarak kullanmak için, veri $\overline{G2A}$ girişine gönderilir. Adres hatları, A B ve C seçme girişlerine bağlanır. Bu üç hatta 5 gibi bir adres varsa, 5 numaralı çıkış gösteriliyor demektir. Eğer $\overline{G2A}$ girişindeki veri alçaksa, 5 çıkışı aktif (alçak) olacaktır. Eğer $\overline{G2A}$ girişindeki veri yüksekse, 5 çıkışı aktif olmayacaktır (yüksek olacaktır). Bir çıkışın, adreslendiği ve veri girişinin alçak olduğu hal dışında, yüksek olduğunu unutmayın.



Şekil 3- 32 74LS138 yol çoklayıcı için bağımlılık gösterilişi mantık sembolü.

YENİ BİR BAĞIMLILIK SEMBOLÜNÜ ANLAMAK

Ne zaman daha önce görmediğiniz bir elemanın bağımlılık gösterilişi sembolünü görürseniz, girişler ve çıkışlar arasındaki ilişkileri anlamak için Ek *B* deki bağımlılık gösterilişi tarifini ve aygıtın doğruluk tablosunu kullanın. Sonra, aygıtı tekrar gördüğünüzde, bağımlılık gösterilişi tek başına giriş ve çıkış fonksiyonlarını size belirtebilecektir.

İKİLİK TABANLI KODLAR

İKİLİK (BINARY)

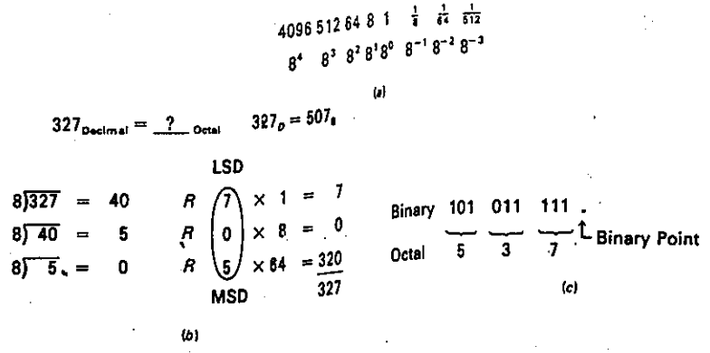
Bölüm 2 de, bilgisayarlar ve diğer sayısal mantık için kullanılan ikilik sayı sistemini tanıtmıştık. Ayrıca onluk sayıların ikilik sayılara ve ikilik sayıların onluk sayılara çevrilmesini de göstermiştik.

SEKİZLİK (OCTAL)

İkilik pek derli toplu bir kod değildir. Bunun anlamı, bir sayıyı ifade etmek için ikilik, örneğin onluktan, çok daha fazla basamak gerektirir. Oniki basamak sadece 4095_{10} e kadar olan sayıları gösterebilir. Bilgisayarlar ikilik veri isterler, fakat bilgisayarlarla çalışan insanlar bu uzun ikilik kelimeleri hatırlamakta zorluk çekerler. Bir çözüm, sekizlik, yani sekiz- tabanlı, kodu kullanmaktır. Şekil 3- 33*a* da gördüğünüz gibi, bu kodda basamaklar 8 in kuvvetlerini gösterir. Kullanılan semboller 0 ile 7 arasındadır. Bir onluk sayıyı dengi sekizlik sayıya çevirmek için, onluk sayıyı ikilik sayıya çevirmede kullandığınız aynı oyunu kullanabilirisiniz. Onluk 327 nin eşiti 507_8 dir. Sağlanması Şekil 3- 33b nin ikinci yarısında sekizlikten- onluğa gerisin geri çevrimle yapılmıştır.

8 sayısı 2 nin bir tam kuvveti olduğuna göre ikilikten sekizliğe ve sekizlikten ikiliğe çevrimler çok kolaydır. 101011111 gibi bir ikilik sayının olduğunu kabul edin. İkilik virgülden başlamak üzere ve sola doğru giderken ikilik basamakları üçerli gruplar halinde işaretleyin. Her üç ikilik basamak grubu bir sekizlik basamağa eşittir. Yukarıdaki örnek için 111 in karşılığı 7, 011 in karşılığı 3 ve 101 in karşılığı 5 tir. O halde ikilik 101011111 sayısı sekizlik 537 ye eşittir.

Sekizlikten ikiliğe, her sekizlik basamağı karşılığı olan 3- bitlik ikilikle değiştirerek çevirebilirsiniz.



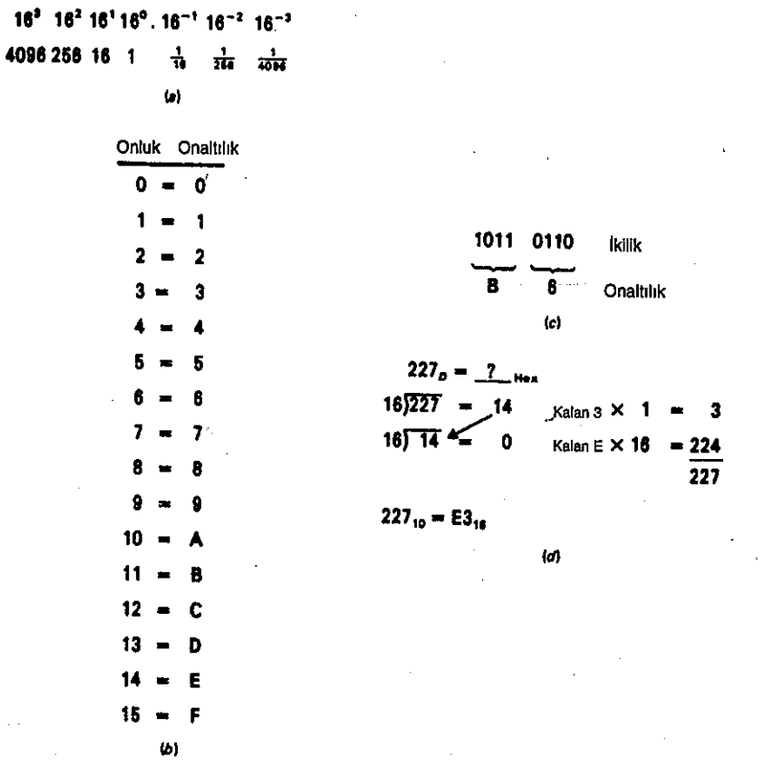
Şekil 3- 33 *(a)* **Sekizlik sayılar - basamakların değerleri.** *(b)* **Onluktan- sekizliğe ve sekizlikten- onluğa çevrim.** *(c)* **İkilikten- sekizliğe çevrim.**

ONALTILIK (HEXADECIMAL)

PDP- 8 gibi birçok popüler minibilgisayarda 12 paralel veri hattı vardır. Dört sekizlik basamak bu 12 paralel hat üzerindeki veri kelimesini temsil etmek için kolayca kullanılabilir. Örneğin 100001010111 ikiliği, 4127 sekizliği olarak kolayca yazılabilir ve hatırlanabilir. Çoğu mikroişlemcilerin 4, 8, veya 16 bitlik veri kelimeleri vardır. Bu mikroişlemciler için, ikilik basamakları üçer üçer değil de dörder dörder gruplayan bir kod kullanmanın çok daha kolay olacağı açıktır. Onaltılık (hexadecimal, çoğunlukla HEX denir), yani 16 tabanlı, kod bunu yapar. Şekil 3- 34*a*, onaltılık sistemde her basamağın değerini gösterir. Bu hex sistem 16- tabanlı olduğundan bütün rakamlar için 16 tane sembol gerekir. Şekil 3- 34*b* deki tablo, hex kodunun sembollerini göstermektedir. 0 ile 9 arasındaki onluk semboller bittikten sonra,10 ile 15 arasındaki değerler için *A* dan *F* ye kadar olan harfler kullanılmaktadır *(A, B, C, D, E, F)*.

Yukarıda bahsedildiği gibi, her hex basamak dört ikilik basamağa eşittir. 10110110 ikiliğini hex'e çevirmek için ikilik noktadan başlayıp sola doğru hareket ederken dörderli grupları işaretleyin. Sonra her dörtlü grup için hex kodunu yazın. 0110 grubu 6 ya eşittir ve 1011 grubu onluk sistemde 11 sayısına eşittir. 11 in hex kodu *B* olduğundan 10110110 ikiliği B6 hex e eşittir. Sekiz ikilik basamağı temsil etmek için, iki hex basamak gereklidir. (Şekil 3- 34*c* ye bakın).

Onluktan onaltılığa çevirmek isterseniz, Şekil 3- 34d bunu yapmak için tanıdık bir oyunu göstermektedir. Sonuç 227_{10} sayısının $E3_{16}$ e eşit olduğunu göstermektedir. Gördüğünüz gibi, hex kodu onluk koddan bile daha kısadır. İki hex basamak 255 *e* kadar olan onluk sayıları gösterebilir. 16- bitlik bir ikilik sayıyı temsil etmek için sadece 4 hex basamağı gerekir.



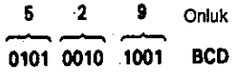
Şekil 3- 34 Onaltılık (hex) sayılar. *(a)* **Basamakların değerleri.** *(b)* **Semboller.** *(c)* **İkilikten- onaltılığa çevrim.** *(d)* **Onluktan- hex e ve hex den- on luğa çevrim**

Onaltılık sayıların sayısal mantıkta nasıl kullanıldığını göstermek üzere şu örneği ele alalım: Bir bakım kılavuzu, belirli bir işlem sırasında 8080A mikroişlemcinin 8- bit genişliğindeki veri yolunda $3F_{16}$ olması gerektiğini söylemektedir. Bir osiloskop veya mantık analizörü ile paralel hatlar üzerinde nasıl bir 1 ler ve 0 lar deseni bulacağınızı belirlemek üzere $3F_{16}$ yi 00111111 ikiliğine çevirin. Hex 3F şeklindeki ifade hatırlanması daha kolay ve hatalara daha az açık bir kısa yazılıştır.

Sekizlikten hex koduna çevirmek için en kolay yol, önce sekizliğin karşılığı olan ikiliği yazmak, sonra ikilik basamakları dörder dörder hex e dönüştürmektir. Hex ten sekizliğe çevirmek için bu yordamı ters çevirin.

BCD (BINARY- CODED- DECIMAL, İKİLİK- KODLANMIŞ- ONLUK) KODLAR

Çıkışın bir onluk görüntü olduğu frekans sayıcılar, sayısal voltmetreler veya hesap makineleri gibi uygulamalarda, çoğunlukla bir ikilik- kodlanmış-onluk (BCD) kodu kullanılır. Bu uygulamalarda, BCD nin avantajı, her onluk basamaktaki bilginin ayrı bir 4- bitlik ikilik kelimede içerilmesidir. Tablo 3- 1 den görebileceğiniz gibi, en basit BCD kodu, 0 dan 9 a kadar olan BCD sayılar için standart ikilik kodun ilk 10 sayısını kullanmaktadır. O halde, her onluk basamak, karşılığı olan 4- bitlik ikilikle ayrı olarak temsil edilir. Şekil 3- 35 bunu göstermektedir.



Şekil 3- 35 Onluktan- BCD ye çevrim.

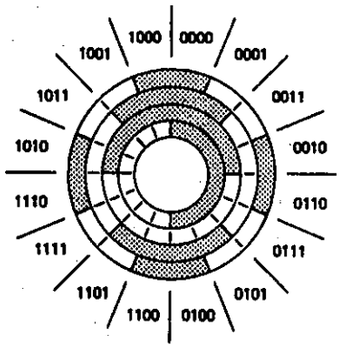
Tablo 3- 1 de bilgi edinmeniz için birçok diğer BCD kodları da içerilmiştir. Bunlardan önemli bir tanesi, bazı hesap makinalarında iç aritmetik işlemleri basitleştirdiği için kullanılan 3- FAZLA BCD kodudur. 3- FAZLA BCD kodu, standart 8421 BCD kodundaki her sayıya 3 ekleyerek elde edilir. Bunu Tablo 3- 1 deki 3- FAZLA ve 8421 BCD sayılarını karşılaştırarak kolayca görebilirsiniz.

GRAY KODU

Diğer bir önemli ikilik kod, çoğunlukla torna gibi bilgisayar kontrollu makinalarda veri kodlamada kullanılan Gray kodudur. Kod, standart ikilikteki aynı 16 mümkün birleşime sahiptir, fakat Tablo 3- 1 den de gördüğünüz gibi, bu birleşimler farklı bir şekilde sıralanmıştır. Bu kodda yukarı doğru sayarken her seferinde sadece bir bitin değiştiğine dikkat edin. Kodun sıralamasını bu şekilde değiştirmek, Şekil 3- 36 da gösterilen mil konum- kodlama diski gibi kaynaklardan veri alınırken, mümkün hata büyüklüğünü azaltmaktadır. Disk, dönen milin konumunu ikilik Gray kodu dengine çevirir. Bit sayısı (disk üzerindeki sektör sayısı) çözünürlüğü belirler. Gösterilen disk 4 bitliktir. O halde 22,5° lik çözünürlük verir. Sekiz bit, 2^8 de 1 lik, yani 1,4° lik bir çözünürlük verir.

TABLO 3.1
KULLANILAN SAYI KODLARI

ONLUK SİSTEM	İKİLİK	SEKİZ-LİK	ONALTI-LIK	ONLUK KODLAR				YANSITIL-MIŞ GRAY KODU	YEDİ PARÇALI GÖSTERGE (1=YANIYOR)							GÖSTERGE
				8421 BCD	2421	5421	3 - FAZLA		A	B	C	D	E	F	G	
0	0000	0	0	0000	0000	0000	0011	0000	1	1	1	1	1	1	0	0
1	0001	1	1	0001	0001	0001	0100	0001	0	1	1	0	0	0	0	1
2	0010	2	2	0010	0010	0010	0101	0011	1	1	0	1	1	0	1	2
3	0011	3	3	0011	0011	0011	0110	0010	1	1	1	1	0	0	1	3
4	0100	4	4	0100	0100	0100	0111	0110	0	1	1	0	0	1	1	4
5	0101	5	5	0101	1011	1000	1000	0111	1	0	1	1	0	1	1	5
6	0110	6	6	0110	1100	1001	1001	0101	1	0	1	1	1	1	1	6
7	0111	7	7	0111	1101	1010	1010	0100	1	1	1	1	0	0	0	7
8	1000	10	8	1000	1110	1011	1011	1100	1	1	1	1	1	1	1	8
9	1001	11	9	1001	1111	1100	1100	1101	1	1	1	1	0	1	1	9
10	1010	12	A	0001 0000	0001 0000	0001 0000	0100 0011	1111	1	1	1	1	1	0	1	A
11	1011	13	B	0001 0001	0001 0001	0001 0001	0100 0100	1110	0	0	1	1	1	1	1	B
12	1100	14	C	0001 0010	0001 0010	0001 0010	0100 0101	1010	0	0	0	1	1	0	1	C
13	1101	15	D	0001 0011	0001 0011	0001 0011	0100 0110	1011	0	1	1	1	1	0	1	D
14	1110	16	E	0001 0100	0001 0100	0001 0100	0100 0111	1001	1	1	0	1	1	1	1	E
15	1111	17	F	0001 0101	0001 1011	0001 1000	0100 1000	1000	1	0	0	1	1	1	1	F



Şekil 3- 36 Gray kodu mil konumu kodlayıcı diski.

Gray kodu ile, en büyük mümkün hata, bir en küçük değerlikli basamak kadardır. Standart ikilik kodlanmış diskte, en büyük değerlikli basamak kadar büyük bir hata elde edebilirsiniz. Örneğin, 7 den 8 e değişirken, algılayıcı ilk 3 bitin 0 a değiştiğini algılarda, dördüncü bitin 1 e değiştiğini algılamazsa, çıkış 0000 olacaktır ki bu da en büyük değerlikli bitte olan bir hatadır.

Gray kodunun avantajını göstermek için aynı geçişi onunla deneyin. Olabilecek en kötü hata miktarı ,1 en küçük değerlikli bit kadardır, çünkü dördüncü biti algılasaydınız 8 elde edecektiniz, algılamazsanız 7 elde edersiniz. Eğer Tablo 3- 1 dekinden daha büyük bir Gray kodu tablosu kurmanız gerekirse, 0 lar ve 1 lerdeki deseni gözlemek ve bu deseni genişletmek çok kolay bir yöntemdir. En değerliksiz bit sütunu 0 ile başlar ve sütunda aşağı doğru inerken değişen ikişerli 1 ler ve ikişerli 0 lar grupları görülür. İkinci en büyük değerlikli- basamak sütunu iki 0 la başlar ve değişen dörderli 1 ler ve dörderli 0 lar grupları ile devam eder. Üçüncü sütun, dört sıfırla başlar ve değişen sekizerli 1 ler ve sekizerli 0 lar grupları ile devam eder. Şimdiye kadar deseni görebilmiş olmanız gerekir. Onluk 16 sayısının Gray kodunu bulmaya çalışın. 11000 bulmanız gerekir.

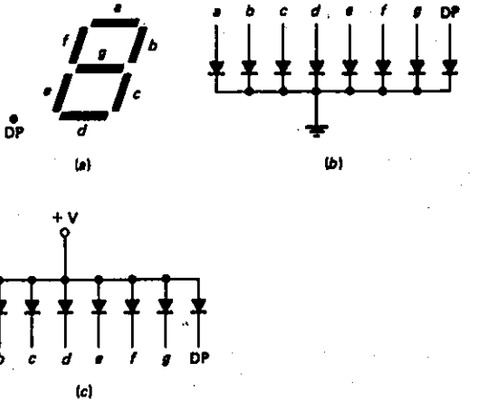
YEDİ- PARÇALI GÖSTERGE KODU

Şekil 3- 37 de gösterilene benzer yedi- parçalı- göstergeler, hesap makinalarından benzin pompalarına kadar artık her yerde kullandıklarından, bu gösterge parçalarına ait kod Tablo 3- 1 de verilmiştir. Tek bir yedi- parçalı- gösterge bu koddaki son altı sayıyı (10 dan 15 *e*), *A* dan *F* ye kadar hex sembolleriyle gösterir. Tablo 3- 1 de bir 1, Şekil 3- 37b deki ortak- katotlu *LED* lerde

olduğu gibi, parçanın yanmakta olduğunu gösterir. Şekil 3- 37c de gösterilen ortak- anotlu LED göstergelerde olduğu gibi, bazı göstergelerde bir alçak, parçayı yakar, bu nedenle bütün değerleri ters çevirmeniz gerekir.

ÇOK BİTLİ KODLAR

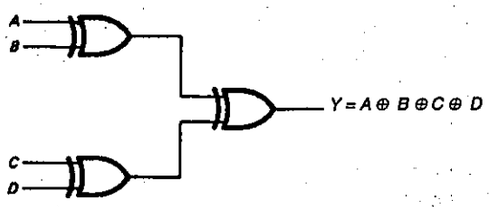
PARİTE : Bilgisayarlarla yazıcılar, delikli kartlar veya ekranlar aralarında haberleşirken, sayıların yanısıra alfabenin harflerini de temsil edebilecek ikilik- tabanlı kodlara gerek vardır. Bu amaçla, çok kullanılan kodların her kelimesinde 5 ile 12 arasında bit kullanılmaktadır. Çok bitli kodlarda olabilecek mümkün hataları bulmak için parite biti denilen ilave bir bit, çoğunlukla en büyük değerlikli bit olarak, kelimelere eklenir.



Şekil 3- 37 Yedi- parçalı *LED* gösterge. *(a)* Parça isimleri. *(b)* Ortak- katotlu tipin şeması. *(c)* Ortak- anotlu tipin şeması.

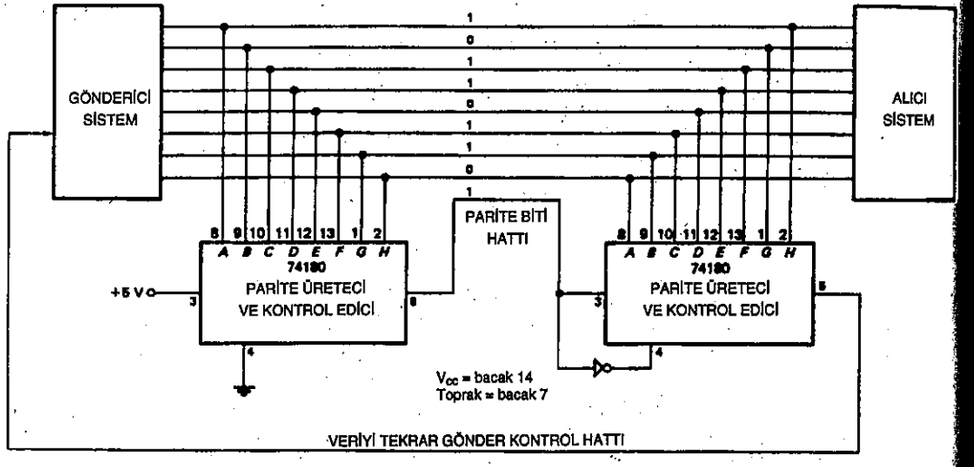
Parite, bir veri kelimesinin tek veya çift sayıda 1 e sahip olduğunu belirtmek üzere kullanılan bir terimdir. Eğer veri kelimesindeki 1 lerin sayısı tekse, kelime tek pariteye sahiptir denir. 0110111 ikilik kelimesi beş tane 1 i olduğu için tek pariteye sahiptir. 0110000 ikilik kelimesi ise çift sayıda 1 e sahiptir, bu nedenle de çift pariteye sahiptir.

İkilik bir kelimenin paritesi bir özel VEYA kapı devresi ile belirlenebilir. Şekil 3- 38, dört girişli özel VEYA "ağaç" devresini göstermektedir. Bu devrenin çıkışı eğer tek sayıda yüksek giriş olmuşsa yüksektir, çift sayıda yüksek giriş olmuşsa alçaktır.



Şekil 3- 38 Dört değişken için özel VEYA "ağaç" devresi.

74180 Parite Üretici ve Kontrol Edici entegresindekine benzer bir sekiz-girişli parite "ağaç" ı, 8 bite kadar olan veri kelimelerinin paritesinin tek mi çift mi olduğunu size söyleyebilir. Şekil 3- 39 daki gibi bağlandığında, gönderen uçtaki 74180, veri kelimesindeki tek pariteyi algılayacak ve bir yüksek çıkış üretecektir. Bu üretilen parite bilgisi, yani parite biti, veri ile birlikte bir dokuzuncu hattan gönderilir. Dokuz hattın tamamındaki parite şimdi çifttir. Alıcı uçtaki diğer bir 74180 dokuz hattaki pariteyi kontrol eder ve beklenildiği gibi çift parite bulursa alçak çıkış verir. Böylece gönderen sisteme, verinin doğru parite ile alındığını söyler. Alıcı uçtaki 74180 in bir yüksek çıkış vermesi alınan veride hata olduğunu gösterir. Bu yüksek çıkış, gönderen sisteme bir hata bayrağı sallar ve kelimenin yeniden gönderilmesi için kullanılabilir.



Şekil 3- 39 İletilen verideki hataları bulmak için parite üretici ve kontrol ediciyi kullanan sistem.

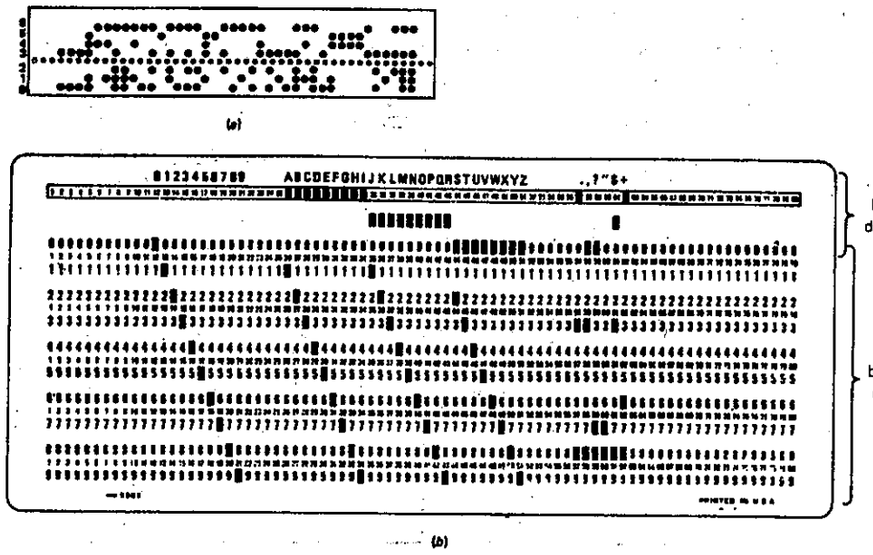
Bu hata bulma olayını bir özetleyelim. Gönderen uçtaki 74180, veri kelimesinin paritesini inceler ve dokuz hatta her zaman çift parite olacak şekilde bir 0 veya 1 çıkışı üretir. Alıcı uçtaki 74180 dokuz hattı kontrol eder ve dokuz hattaki parite çift değilse bir hata bayrağı olarak bir yüksek çıkış verir.

İletim sırasında oluşan hataları bulmak için kullanılan bu yöntemin bir sakıncası, bir veri kelimesinde oluşacak iki hatanın, kelimenin paritesini değiştirmeyecek ve böylece parite kontrol edicinin hata göstermeyecek olmasıdır. Daha fazla hat ve "Hamming kodları" kullanan kompleks bir yöntem, gönderilen verideki çoklu hataları bulmada ve hâttâ hataları düzeltmede kullanılabilir. Hamming kodu hata bulan/düzelten devreler de özel- VEYA ağacı tabanlı parite üreteç/ kontrol edicilerini kullanmaktadır.

ASCII Tablo 3- 2 çeşitli çok bitli kodları göstermektedir. Bunlardan ilki ASCII, (American Standart Code for Information Interchange = Bilgi Alışverişi için Amerikan Standart Kodu) dur. Bu kod tabloda 7- bitlik bir kod olarak gösterilmiştir. 7 bitle 128 taneye kadar karakteri kodlayabilirsiniz ki, bu da bütün büyük ve küçük harfler, rakamlar, noktalama işaretleri ve kontrol karakterleri için yeterlidir. Sadece büyük harfler, rakamlar ve bazı kontrol karakterleri kullanılmak istenirse, ilk 6 bitin yeterli olmasını sağlamak için kod özel olarak düzenlemiştir. Eğer bir parite kontrolu istenirse, temel 7- bitlik kodun en önemli bit konumuna bir parite biti eklenir. Örneğin, 11000100 ikilik kelimesi büyük harf D için tek pariteli ASCII kodudur. Şekil 3- 40a da görüldüğü gibi ,ASCII kodunun 1 inç (2,5 cm) genişliğinde kağıt şerit üzerine açılan deliklerle de gösterildiğini görmüş olabilirsiniz. Tablo 3- 3, ASCII tablosunda kullanılan kontrol karakter sembollerinin anlamlarını vermektedir.

BCDIC BCDIC, bazı bilgisayarlarda kullanılan İkilik- Kodlu Onluk Alışveriş Kodu (Binary- Coded Decimal Interchange Code) demektir. 7 bit ve 1 de parite biti kullanır. Alt 4 bite sayısal bitler denir. Üst 4 bitin 1 tanesi parite biti, 3 tanesi de bölge bitleridir. Bu bitlerin düzenlenişi Tablo 3- 2 nin altında verilmiştir. Tablo 3-2 de yer kazanmak için çift pariteli BCDIC için, ikilik basamakların hex dengi verilmiştir.

EBCDIC IBM cihazlarında sık karşılaşılan diğer bir alfanümerik kod *Genişletilmiş Ikilik- Kodlu Onluk Alışveriş Kodudur (Extended Binary- Coded Decimal Interchange Code).* Bu parite biti olmayan 8 bitlik bir koddur. Parite için bir dokuzuncu bit eklenebilir. Tablo 3- 2 de yer kazanmak için EBCDIC in 8 ikilik basamağı, iki basamaklı hex karşılıkları ile gösterilmiştir.



Şekil 3- 40 Şeritler ve kartlar üzerinde bilgi saklamak için kullanılan kodlar. *(a)* Kağıt şerit üzerinde ASCII. *(b)* Delikli kart üzerinde Hollerith.

SELECTRIC Selectric bildiğiniz IBM döner- top- kafalı daktilolarda ve printerlerde kullanılan 7 bitlik bir koddur. Tablo 3- 2, bu kodu da kaynak olması amacıyla göstermektedir. Koddaki her bitin konumu döner- topun bir işlemini kontrol eder. En büyük değerlikli bitten en küçük değerlisine doğru 7 bitin anlamları sırasıyla DÖNME5, EĞİLME 1, EĞİLME2, KAYMA, DÖNME 2A,DÖNME 2 ve DÖNME 1 dir (ROTATE 5, TILT1, TILT2, SHIFT, ROTATE 2A, ROTATE2, ve ROTATE 1). Bu 7- bitlik koda ek olarak, boşluk, satırbaşı, geri, tab, zil ve indis için Selectric, ayrı makina kontrollarına sahiptir.

BAUDOT Teletayp makinalarında kullanılan modası geçmiş 5- bitlik bir koddur. Bu kodun yerini ASCII almış olduğu için Tablo 3- 2 de gösterilmemiştir.

TABLO 3.2
KULLANILAN ALFANÜMERİK KODLAR

ASCII SEMBOLÜ	7 - BİTLİK ASCII İÇİN HEX KODU	BCDIC SEMBOLÜ	ÇİFT PARİTELİ BCDIC İÇİN HEX KODU	EBCDIC SEMBOLÜ	EBCDIC İÇİN HEX KODU	SELECTRIC SEMBOLÜ	SELECTRIC İÇİN HEX KODU	HOLLERITH SEMBOLÜ	HOLLERITH KODUNDA AÇILAN DELİKLER
NUL	00			NUL	00			NUL	12 09 8 1
SOH	01			SOH	01			SCH	12 9 1
STX	02			STX	02			STX	12 9 2
ETX	03			ETX	03			ETX	12 9 3
EOT	04			EOT	37			ECT	9 7
ENQ	05			ENQ	2D			ENQ	0 9 8 5
ACK	06			ACK	2E			ACK	0 9 8 6
BEL	07			BEL	2F			BEL	0 9 8 7
BS	08			BS	16			BS	11 9 6
HT	09			HT	05			HT	12 9 5
LF	0A			LF	25			LF	0 9 5
VT	0B	‡	9A	VT	0B			VT	12 9 8 3
FF	0C			FF	0C			FF	12 9 8 4
CR	0D	#	FF	CR	0D			CR	12 9 8 5
SO	0E			SO	0E			SO	12 9 8 6
SI	0F			SI	0F			SI	12 9 8 7
DLE	10			DLE	10			DLE	12 11 9 8 1
DC1	11			DC1	11			DC1	11 9 1
DC2	12			DC2	12			DC2	11 9 2
DC3	13			DC3	13			DC3	11 9 3
DC4	14			DC4	35			DC4	9 8 4
NAK	15			NAK	3D			NAK	9 8 5
SYN	16			SYN	32			SYN	9 2
ETB	17			EOB	26			ETB	0 9 6
CAN	18			CAN	18			CAN	11 9 8
EM	19			EM	19			EM	11 9 8 1
SUB	1A			SUB	3F			SUB	9 8 7
ESC	1B			BYP	24			ESC	0 9 7
FS	1C			FLS	1C			FS	11 9 8 4
GS	1D			GS	1D			GS	11 9 8 5
RS	1E			RDS	1E			RS	11 9 8 6
US	1F			US	1F			US	11 9 8 7
SP	20	SP	00	SP	40			SP	DELİK YOK
!	21	!	8A	!	5A	‡!	27	!	12 8 7
"	22	##	5F	"	7F	"	2D	"	8 7
#	23	#	4B	#	7B	#	7E	#	8 3
$	24	$	2B	$	5B	$	79	$	11 8 3
%	25	%	5C	%	6C	%	3D	%	0 8 4
&	26	&	30	&	50	&	7D		12
'	27	V	1D	'	7D	'	25	'	8 5
(	28	Boş	50	(	4D	(	38	(	12 8 5
)	29	Δ	6F	)	5D	)	39	)	11 8 5
*	2A	*	6C	*	5C	*	7C	*	11 8 4

TABLO 3.2 (Devamı)
KULLANILAN ALFANÜMERİK KODLAR

ASCII SEMBOLÜ	7 - BİTLİK ASCII İÇİN HEX KODU	BCDIC SEMBOLÜ	ÇİFT PARİTELİ BCDIC İÇİN HEX KODU	EBCDIC SEMBOLÜ	EBCDIC İÇİN HEX KODU	SELECTRIC SEMBOLÜ	SELECTRIC İÇİN HEX KODU	HOLLERITH SEMBOLÜ	HOLLERITH KODUNDA AÇILAN DELİKLER
+	2B			+	4E	+	0E	+	12 8 6
.	2C	,	1B	.	6B	.	44	.	0 8 3
–	2D			–	60	–	00	–	11
.	2E	.	7B	.	4B	.	26	.	12 8 3
/	2F	/	11	/	61	/	41	/	0 1
0	30	0	0A	0	F0	0	31	0	0
1	31	1	41	1	F1	1	77	1	1
2	32	2	42	2	F2	2	36	2	2
3	33	3	03	3	F3	3	76	3	3
4	34	4	44	4	F4	4	71	4	4
5	35	5	05	5	F5	5	35	5	5
6	36	6	06	6	F6	6	34	6	6
7	37	7	47	7	F7	7	75	7	7
8	38	8	48	8	F8	8	74	8	8
9	39	9	09	9	F9	9	30	9	9
:	3A	:	4D	:	7A	:	4D	:	8 2
;	3B	;	2E	;	5E	;	45	;	11 8 6
<	3C	<	7E	<	4C			<	12 8 4
=	3D	✓	0F	=	7E	=	06	=	8 6
>	3E	>	4E	>	6E			>	0 8 6
?	3F	?	3A	?	6F	?	49	?	0 8 7
@	40	@	0C	@	7C	@	3E	@	8 4
A	41	A	71	A	C1	A	6C	A	12 1
B	42	B	72	B	C2	B	18	B	12 2
C	43	C	33	C	C3	C	5C	C	12 3
D	44	D	74	D	C4	D	5D	D	12 4
E	45	E	35	E	C5	E	1D	E	12 5
F	46	F	36	F	C6	F	4E	F	12 6
G	47	G	77	G	C7	G	4F	G	12 7
H	48	H	78	H	C8	H	19	H	12 8
I	49	I	39	I	C9	I	2C	I	12 9
J	4A	J	21	J	D1	J	07	J	11 1
K	4B	K	22	K	D2	K	1C	K	11 2
L	4C	L	63	L	D3	L	59	L	11 3
M	4D	M	24	M	D4	M	6F	M	11 4
N	4E	N	65	N	D5	N	1E	N	11 5
O	4F	O	66	O	D6	O	69	O	11 6
P	50	P	27	P	D7	P	0D	P	11 7
Q	51	Q	28	Q	D8	Q	0C	Q	11 8
R	52	R	69	R	D9	R	6D	R	11 9
S	53	S	12	S	E2	S	29	S	0 2
T	54	T	53	T	E3	T	1F	T	0 3

TABLO 3.2 (Devamı)
KULLANILAN ALFANÜMERİK KODLAR

ASCII SEMBOLÜ	7 - BİTLİK ASCII İÇİN HEX KODU	BCDIC SEMBOLÜ	ÇİFT PARİTELİ BCDIC İÇİN HEX KODU	EBCDIC SEMBOLÜ	EBCDIC İÇİN HEX KODU	SELECTRIC SEMBOLÜ	SELECTRIC İÇİN HEX KODU	HOLLERITH SEMBOLÜ	HOLLERITH KODUNDA AÇILAN DELİKLER
U	55	U	14	U	E4	U	5E	U	0 4
V	56	V	55	V	E5	V	6E	V	0 5
W	57	W	56	W	E6	W	28	W	0 6
X	58	X	17	X	E7	X	5F	X	0 7
Y	59	Y	18	Y	E8	Y	09	Y	0 8
Z	5A	Z	59	Z	E9	Z	3F	Z	0 9
[	5B	[	7D	[	AD	[	7F	[	12 8 2
\	5C	\	1E	NL	15			\	0 8 2
]	5D	]	2D	]	DD			]	11 8 2
^	5E	□	3C	¬	5F			^	11 8 7
_	5F	—	60	—	6D	—	08	_	0 8 5
`	60			RES	14			`	8 1
a	61			a	81	a	64	a	12 0 1
b	62			b	82	b	10	b	12 0 2
c	63			c	83	c	54	c	12 0 3
d	64			d	84	d	55	d	12 0 4
e	65			e	85	e	15	e	12 0 5
f	66			f	86	f	46	f	12 0 6
g	67			g	87	g	47	g	12 0 7
h	68			h	88	h	11	h	12 0 8
i	69			i	89	i	24	i	12 0 9
j	6A			j	91	j	07	j	12 11 1
k	6B			k	92	k	14	k	12 11 2
l	6C			l	93	l	51	l	12 11 3
m	6D			m	94	m	67	m	12 11 4
n	6E			n	95	n	16	n	12 11 5
o	6F			o	96	o	61	o	12 11 6
p	70			p	97	p	05	p	12 11 7
q	71			q	98	q	04	q	12 11 8
r	72			r	99	r	65	r	12 11 9
s	73			s	A2	s	21	s	11 0 2
t	74			t	A3	t	17	t	11 0 3
u	75			u	A4	u	56	u	11 0 4
v	76			v	A5	v	66	v	11 0 5
w	77			w	A6	w	20	w	11 0 6
x	78			x	A7	x	57	x	11 0 7
y	79			y	A8	y	01	y	11 0 8
z	7A			z	A9	z	37	z	11 0 9
{	7B			{	8B			{	12 0
|	7C			|	4F			|	12 11
}	7D			}	9B			}	11 0
~	7E			¢	4A			~	11 0 1
DEL	7F			DEL	07			DEL	12 9 7

BCDIC: HEX BASAMAK (PCBA) HEX BASAMAK ($2^3 2^2 2^1 2^0$)
SELECTRIC: $R_5T_1T_2$ HEX BASAMAK, $SR_{2_A}R_2R_1$ HEX BASAMAK

HOLLERITH Hollerith, eğer "katlar, kıvırır veya bozarsanız" sizi ölümden de kötü bir gelecek ile tehdit eden bilgisayar kartlarındaki veriyi kodlamak için kullanılan 12- bitlik bir koddur. Şekil 3- 40b standart bir 12 satır ve 80 sütunlu kartı göstermektedir. 12 veri satırı yukarıdan başlamak üzere 12, 11, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 olarak etiketlendirilmiştir. Üstteki 3 sıra bölge delgileri adını alır ve alttaki 10 sıra da basamak delgileri adını alır. 0 satırının her iki grupta da yer aldığına dikkat edin. Açılmış bir delik 1 i temsil eder ve bir veri kelimesi bir düşey sütundaki 12 bitle gösterilir. Şekil 3- 40b deki kart, kartın üstünde yazılı rakam ve harfler için Hollerith kodunu göstermektedir. Tablo 3- 2, kodun tamamını ve her karakterin delik yerlerini göstermektedir. Hollerith kodu, 12 bitin mümkün olan birleşimlerinin sadece az bir kısmını kullandığı için pek verimli değildir. Bu nedenle, çoğunlukla bu kod ASCII veya EBCDIC e çevrilerek kullanılır.

KOD ÇEVİRİCİLER

Elinizde bir kodda bir harf veya karakter üreten bir devre varsa ve siz bunun başka bir kodda olmasını istiyorsanız ne yaparsınız? Örneğin, bir onluk sayıcının çıkışı BCD dir; fakat göstergeyi sürmek için yedi- parçalı koda gerek duymaktasınız. Aşağıda anlatılan ilk yöntem, Karnaugh haritalarının ve mantık kapılarının kullanıldığı geleneksel kod çevrim yöntemidir. Bunu, kod çevriminde MSI ve LSI yöntemleri izlemektedir.

KOD ÇEVİRİCİLER İÇİN GELENEKSEL YÖNTEM

Şekil 3- 41a, 0 dan 9 a kadar BCD sayıların değerlerini ve bunların yedi-parçalı gösterge kodu karşılıklarını göstermektedir. Şekil 3- 41a, ayrıca BCD girişlerinin diğer mümkün birleşimlerini de göstermektedir. Bu birleşimlerin karşısındaki yedi parçalı gösterge kodunda "farketmez" durumları bulunmaktadır. Bunun sebebi, bu giriş birleşimlerini BCD kodda hiçbir zaman elde etmeyeceğiniz kabul edildiği içindir. Diğer taraftan göreceğiniz gibi, bu fark etmez durumlarının tabloda içerilmesi, son mantık ifadesini basitleştirmektedir. BCD den yedi- parçalıya bir çevirici yapmak isterseniz, her parça için, BCD değişkenleri A, B, C ve D cinsinden basitleştirilmiş mantık ifadelerini bulmanız gerekmektedir. Bunun anlamı, her parça için bir Karnaugh haritası yapılması demektir.

Şekil 3- 41b, a parçasının ifadesi için olan Karnaugh haritasını göstermektedir. Daire içine alınmış karelerden gördüğünüz gibi basitleştirilmiş ifadeyi elde ederken farketmez durumları da içerilebilir. Karnaugh haritaları üzerinde daha önceki açıklamalarımızda, ne kadar çok kareyi halkalayabilirseniz çıkış ifadesinde de o kadar az değişken elde edeceğinizi belirtmiştik. Eğer bir farketmez durumu bir 1 in yanında olursa ve farketmez durumunu içermek daha büyük bir nizami gurup (2^n) oluşturuyorsa, o farketmez

durumunu da halka içine koymalısınız. Aksi halde farketmez durumlarını ihmal edin. K- haritasını kullanarak ve uygun farketmez durumları içererek a parçası için elde edilen ifade

$$a = B\,D + \overline{B}\,\overline{D} + A + C$$

olur.

Şekil 3- 41c, bilinen TTL mantık kapılarıyla a parçası çıkışının oluşturulmasının bir yolunu göstermektedir. Sadece a parçası için iki farklı IC paketinin bazı kısımları ile dört ters çevirici kullanıldığına dikkat edin. Diğer altı Karnaugh haritasını, alıştırma olması için siz yapabilirsiniz, çünkü bazı hallerde bu yönteme gerek duyulmaktadır.

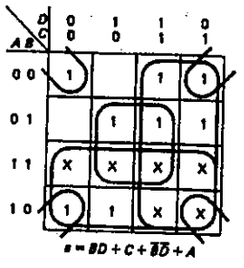
TABLO 3- 3

KONTROL KARAKTERLERİNİN TANIMLARI

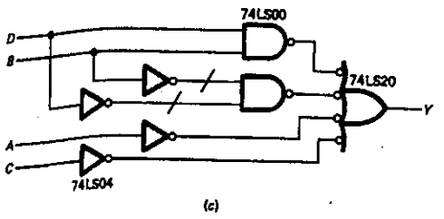
NUL	BOŞ
SOH	BAŞLIĞIN BAŞI
STX	YAZIYA BAŞLA
ETX	YAZIYI BİTİR
EOT	İLETİM SONU
ENQ	SORUŞTURMA
ACK	ALINDI
BEL	ZİL
BS	BİR KARAKTER GERİ
HT	YATAY TAB
LF	SATIR BESLEME
VT	DÜŞEY TAB
FF	SAYFA BESLEME
CR	SATIRBAŞI
SO	DEĞİŞİKLİĞİ ÇIKAR
SI	DEĞİŞİKLİĞE GİR
DLE	VERİ BAĞI KAÇMA
DC1	DOĞRUDAN KONTROL 1
DC2	DOĞRUDAN KONTROL 2
DC3	DOĞRUDAN KONTROL 3
DC4	DOĞRUDAN KONTROL 4
NAK	NEGATİFALINDI
SYN	SENKRON BOŞTA
ETB	İLETİM BLOĞU SONU
CAN	İPTAL
EM	ORTAM SONU
SUB	DEĞİŞTİR
ESC	KAÇMA
FS	SAYFA AYIRICI
GS	GRUP AYIRICI
RS	KAYIT AYIRICI
US	BİRİM AYIRICI

	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1
10	1	0	1	0	X	X	X	X	X	X	X
11	1	0	1	1	X	X	X	X	X	X	X
12	1	1	0	0	X	X	X	X	X	X	X
13	1	1	0	1	X	X	X	X	X	X	X
14	1	1	1	0	X	X	X	X	X	X	X
15	1	1	1	1	X	X	X	X	X	X	X

(a)



(b)



(c)

Şekil 3- 41 BCD den yedi- parçalı koda çevrim. *(a)* Doğruluk tablosu. *(b)* Karnaugh haritası ve *a* parçası için türetilen ifade. *(c)* a parçası ifadesini gerçekleştiren devre.

Diğer altı parça için olan sonuçlar şöyledir:

$b = \overline{B} + \overline{C}\,\overline{D} + C\,D$

$c = \overline{C} + B + D$

$d = \overline{B}\,\overline{D} + A + \overline{B}\,C + C\,\overline{D} + B\,\overline{C}\,D$

$e = \overline{B}\,\overline{D} + C\,\overline{D}$

$f = A + B\,\overline{D} + B\,\overline{C} + \overline{C}\,\overline{D}$

$g = A + \overline{B}\,C + B\,\overline{C} + C\,\overline{D}$

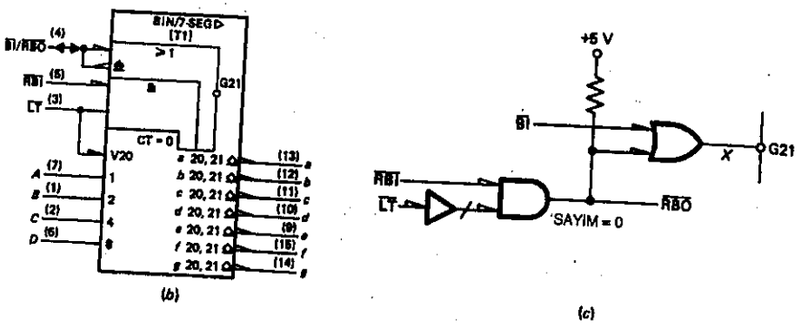
KOD ÇEVİRİCİLERE MSI YAKLAŞIMI

TTL veya CMOS veri kitabına bir göz atıldığında, birçok kod çevrimleri için hazır MSI aygıtlarının var olduğu görülür. Bunların bazılarına kod çözücüler (decoder), bazılarına da kodlayıcılar (encoders) denmektedir. Bir kod çözücü genellikle BCD gibi daha sıkıştırılmış bir koddan, yedi- parçalı gibi daha az sıkışık bir koda çevirir. Bir kodlayıcı, sekizden- biri- alçak gibi, daha az sıkışık bir koddan, ikilik gibi, daha çok sıkışık bir koda çevirir.

KOD ÇÖZÜCÜLER : 7447 BCD den-yedi-parçalıya kod çözücü bunlara iyi bir örnektir. Bu aygıt, bir önceki kısımda, geleneksel kod çevrim metodunda anlatılanların hepsini yapar. Şekil 4-15, bir 4- bitlik BCD giriş kodundan, bir ortak- anotlu yedi- parçalı LED göstergenin doğru sürüş sinyallerini elde etmek üzere bağlanmış bir 7447 yi göstermektedir. Bir benzer aygıt, 7448, bir 4- bitlik BCD girişi, bir ortak- katotlu yedi- parçalı LED göstergeyi sürmek için çevrim yapar. 7448 in CMOS dengi 4511 dir. Bu entegrelerin, test amacıyla bütün göstergeyi yakmak, göstergeyi karartmak veya sayının solunda bir sıfır varsa, örneğin bir DVM deki gibi, bunları karartmak için ilave kontrol girişleri vardır.

Şekil 3- 42*a* ve *b*, bir 7447 BCD den- yedi- parçalıya kod çözücünün fonksiyon tablosunu ve bağımlılık gösterilişinin mantık sembolünü göstermektedir. Bunu, bağımlılık gösterilişinin daha kompleks bir aygıttaki giriş ve çıkış ilişkilerini nasıl gösterdiğinin bir başka örneği olarak kullanacağız. Önce fonksiyon tablosunu ve altındaki notları dikkatle inceleyin. Sonra bağımlılık gösterilişi sembollerin, bu bilgileri nasıl anlattığını anlamaya çalışın. Daha önce görmediğiniz veya hatırlamadığınız bağımlılık gösterilişi sembolleri için EK *B* ye bakınız.

Fonksiyon tablosundaki a ile g arasındaki parça çıkışlarını inceleyin. Herbiri üzerindeki kamalar bunların aktif alçak olduğunu göstermektedir ve her birinin yanındaki paralelkenar sembolü bu çıkışların açık kollektörlü olduğunu söylemektedir. BCD girişleri *A, B, C* ve *D*, ikilik sistemdeki ağırlıkları ile etiketlendirilmiştir. Fonksiyon tablosundaki ilk 16 girişten, eğer *lamba testi* ($\overline{LT}$) *(lamp test)* yüksekse, dalgacık- karartma girişi $\overline{RBI}$ *(ripple- blanking input) yüksekse ve karartma girişi/ dalgacık- karartma çıkışı* ($\overline{BI}/\overline{RBO}$) *(Blanking input/ ripple- blanking output)* yüksekse bu durumda yedi- parçalı çıkışları, $\overline{A}$, $\overline{B}$, $\overline{C}$ ve *D* girişlerindeki ikilik koda karşılık gelecektir. Eğer karartma girişi ($\overline{BI}$ / $\overline{RBO}$) alçağa çekilirse veya eğer $\overline{RBI}$ alçak ve $\overline{LT}$ yüksek ise bütün çıkışlar kapalı olacaktır. Bu göstergeyi karartır. Eğer lamba testi ($\overline{LT}$) alçak ise ve $\overline{BI}$ / $\overline{RBO}$ alçağa çekilmezse, bütün çıkışlar yanacaktır. Bu mod, bir göstergenin bütün parçalarının bağlanmış olup olmadığını ve çalışıp çalışmadığını anlamak için kullanılabilir. Ayrıca, eğer $\overline{LT}$ yüksekse ve *A, B, C* ve *D* girişlerinde 8 kodu varsa çıkışların hepsi yanacaktır.





PARÇA TANITIMI

Sayısal Atamalar ve Sonuç Görüntüler

'46A, '47A, 'L46, 'L47, 'LS47 FUNCTION TABLE

Onluk ve İşlev	Girişler						BI/RBO†	Çıkışlar							Not
	LT	RBI	D	C	B	A		a	b	c	d	e	f	g	
0	H	H	L	L	L	L	H	ON	ON	ON	ON	ON	ON	OFF	
1	H	X	L	L	L	H	H	OFF	ON	ON	OFF	OFF	OFF	OFF	
2	H	X	L	L	H	L	H	ON	ON	OFF	ON	ON	OFF	ON	
3	H	X	L	L	H	H	H	ON	ON	ON	ON	OFF	OFF	ON	
4	H	X	L	H	L	L	H	OFF	ON	ON	OFF	OFF	ON	ON	
5	H	X	L	H	L	H	H	ON	OFF	ON	ON	OFF	ON	ON	
6	H	X	L	H	H	L	H	OFF	OFF	ON	ON	ON	ON	ON	
7	H	X	L	H	H	H	H	ON	ON	ON	OFF	OFF	OFF	OFF	
8	H	X	H	L	L	L	H	ON	ON	ON	ON	ON	ON	ON	
9	H	X	H	L	L	H	H	ON	ON	ON	OFF	OFF	ON	ON	
10	H	X	H	L	H	L	H	OFF	OFF	OFF	ON	ON	OFF	ON	
11	H	X	H	L	H	H	H	OFF	OFF	ON	ON	OFF	OFF	ON	
12	H	X	H	H	L	L	H	OFF	ON	OFF	OFF	OFF	ON	ON	
13	H	X	H	H	L	H	H	ON	OFF	OFF	ON	OFF	ON	ON	
14	H	X	H	H	H	L	H	OFF	OFF	OFF	ON	ON	ON	ON	
15	H	X	H	H	H	H	H	OFF	OFF	OFF	OFF	OFF	OFF	OFF	
BI	X	X	X	X	X	X	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	2
RBI	H	L	L	L	L	L	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	3
LT	L	X	X	X	X	X	H	ON	ON	ON	ON	ON	ON	ON	4

H=Yüksek seviye, *L*=Alçak seviye, *X*=Farketmez, *ON*=Yanıyor, *OFF*= Sönük

NOTLAR :

1. Çıkış fonksiyonları 0- 15 istendiğinde, *BI* karartma girişi açık olmalı veya yüksek mantık seviyesinde tutulmalıdır.Eğer onluk bir 0 ın karaltılması istenmiyorsa, dalgacık- karartma girişi *RBI* açık olmalıdır.
2. BI karartma girişine doğrudan bir alçak mantık seviyesi uygulandığında, diğer herhangi bir girişin seviyesine bakılmaksızın, bütün parça çıkışları OFF (sönük) durumda olur.
3. RBI dalgacık- karartma girişi ve *A, B, C* ve *D* girişleri alçak seviyede, ve lamba test girişi yüksek ise, bütün parça çıkışları OFF (sönük) olur ve *RBO* dalgacık- karartma çıkışı alçak seviyeye gider (tepki durumu).
4. BI /RBO karartma girişi/dalgacık- karartma çıkışı açıksa veya yüksek tutulursa ve lamba test girişine bir alçak uygulanırsa, bütün parça çıkışları ON (yanıyor) olur.

+ BI / RBO bacağı, *BI* karartma girişi ve/veya *RBO* dalgacık- karartma çıkışı olarak görev yapan bağlanmış- VE mantığıdır.

Şekil 3- 42 7447 BCD den- yedi- parçalı- koda kod çözücüsü. *(a)* Fonksiyon tablosu. *(b)* Bağımlılık mantık sembolü. *(c)* $\overline{BI}$, $\overline{RBI}$ ve $\overline{LT}$ için dahili mantık.

Şimdi bütün bunların nasıl özetlendiğini görmek için Şekil 3- 42*b* deki bağımlılık gösterilişi sembolüne bakın. Size yardımcı olmak için, Şekil 3- 42*c* daha geleneksel mantık sembolleri kullanarak iç mantık bloklarını göstermektedir.Eğer $\overline{BI}$ aktifse veya *A, B, C* ve *D* girişleri sıfır iken $\overline{RBI}$ aktifse ve $\overline{LT}$ aktif değilse bu durumda *X* taki sinyalin aktif olacağını görebilirsiniz. G21 deki yuvarlak, içerde bir tersini çevirme bağlantısını temsil eder. Bunun anlamı, eğer bu noktada sinyal aktif değilse VE *A, B, C* ve *D* girişlerinde bir kod varsa, 21 sayısıyla gösterilen çıkışlar giriş ikilik koduna göre aktif olacaktır. Eğer bu noktadaki sinyal aktif ise, bu takdirde 21 sayısı ile belirlenen çıkışlar aktif olmayacaktır. Bu göstergeyi karartır. $\overline{RBI}$ aktifse, $\overline{LT}$ aktif değilse ve *A, B, C* ve *D* deki kod 0000 ise *RBO* nin aktif olduğuna dikkat edin.

Eğer $\overline{LT}$ girişi aktifse, 20 sayısıyla belirtilen bütün çıkışlar aktiftir. Böylece bütün parçalar yanar.

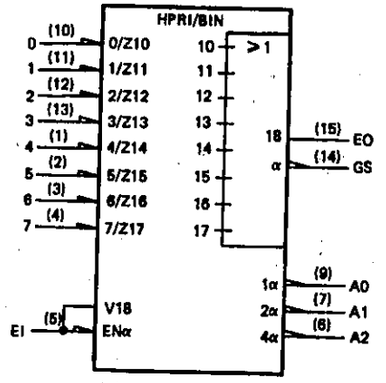
Bağımlılık gösterilişi sembolleri bir fonksiyon tablosuna olan gereksinimi yok etmemektedir; fakat sembollerin ne anlama geldiğini öğrendiğinizde, aygıtın fonksiyonlarından çoğunu doğrudan şemadan anlamanıza imkân vermektedir.

Diğer bir kullanışlı kod çözücü grubu 7442, 7445 ve 74145 BCD giriş kodunu ondan- biri alçak koduna çevirir. Bu kod çözücülerden biri, ortak anodu yüksek gerilime bağlanmış ve her biri tam bir karakter olarak şekillendirilimiş ayrı katotları olan gaz- deşarjlı göstergeleri sürmek için kullanılır. Katot topraklandığında, etrafındaki bölge parlar ve karakterin izi görülür.

KODLAYICILAR : Kodlayıcıların iyi bir örneği 74148 sekiz- hattan- üçhatta öncelik kodlayıcısıdır. Şekil 3- 43, bu kodlayıcının bağımlılık gösterilişi sembolünü ve fonksiyon tablosunu vermektedir. Aygıtı yetkilendirmek için EI (Enable Input) girişi alçak yapılır. Elemanın sekiz veri girişi (0 dan 7 ye), üç veri çıkışı *(A2, A1 ve A0)* ve iki kontrol çıkışı *(GS ve EO)* vardır. Bütün giriş ve çıkışlar aktif alçaktır. El alçak ve bütün girişler yüksek olduğunda yetkilendirme çıkışı EO *(Enable Output)* alçak olur. Sekiz veri girişinden herhangi birinde bir alçak varsa, grup yükleme *GS (Group Strobe)* çıkışı bunu göstermek üzere alçak olacaktır. Eğer *EI* alçaksa ve yalnız bir veri girişi alçaksa, alçak girişin numarasının ikilik kodu aktif alçak olarak veri çıkışlarında görülecektir. Örneğin, yalnız giriş 3 alçaksa, bu takdirde *A2* yüksek, *A1* alçak ve *A0* alçak olacaktır.

Bu seviyeler 3 sayısı için aktif alçak veya ters çevrilmiş ikilik kodun karşılığıdır. Eğer birden fazla giriş alçak ise, bu durumda en yüksek- sayılı alçak girişin kodu çıkışta görülecektir. Bu, Şekil 3- 43 teki fonksiyon tablosundaki farketmez sembolü *X* larla belirtilmiştir. Örneğin, eğer giriş 3 ve giriş 5 alçaksa, bu durumda çıkışlar $A2 = 0$, $A1 = 1$ ve $A0 = 0$ olacaktır ki bu da 5 in ters

çevrilmiş kodudur. 74148 öncelik kodlayıcısının çalışmasını özetlersek veri çıkışları, alçak olan en yüksek- sayılı girişin ters çevrilmiş ikilik kodunu gösterir.



(a)

'148, 'LS148
Fonksiyon Tablosu

Girişler									Çıkışlar				
EI	0	1	2	3	4	5	6	7	A2	A1	A0	GS	EO
H	X	X	X	X	X	X	X	X	H	H	H	H	H
L	H	H	H	H	H	H	H	H	H	H	H	H	L
L	X	X	X	X	X	X	X	L	L	L	L	L	H
L	X	X	X	X	X	X	L	H	L	L	H	L	H
L	X	X	X	X	X	L	H	H	L	H	L	L	H
L	X	X	X	X	L	H	H	H	L	H	H	L	H
L	X	X	X	L	H	H	H	H	H	L	L	L	H
L	X	X	L	H	H	H	H	H	H	L	H	L	H
L	X	L	H	H	H	H	H	H	H	H	L	L	H
L	L	H	H	H	H	H	H	H	H	H	H	L	H

(b)

Şekil 3- 43 74148 Sekiz- hattan- üç- hatta öncelik kodlayıcısı.
***(a)* Bağımlılık gösterilişi mantık sembolü. *(b)* Fonksiyon tablosu.**

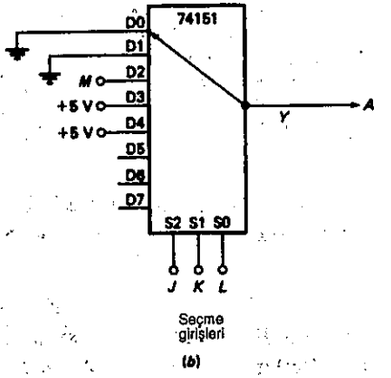
74148 in bağımlılık gösterilişi sembolü, iç mantık arabağlantılarını göstermek için Z harfini kullanır. Eğer 74148 in 0 girişi aktif yapılırsa, Z10 sembolü, iç VEYA blokunun 10 (on) girişi, bazı bağlantılar aracılığıyla, aktif hale gelecektir. Eğer *ENα* girişi aktif yapılırsa, bu takdirde α harfiyle işaretlenen bütün çıkışlar yetkilendirilir. Bu durumda, çıkışları göstermede ikilik ağırlıklı sayılar kullanıldığından, karışıklık olmaması için bir Yunan harfi kullanılmıştır. Eğer *ENα* aktif yapılırsa, en öncelikle aktif girişin kodu *A2*, *A1* ve *A0* çıkışlarında görülecektir. Eğer *EI* aktif değilse VEYA veri girişlerinden biri aktif değilse *EO* aktif olmayacaktır. Eğer *ENα* aktifse ve veri girişlerinden biri aktifse, bu takdirde *GS* aktif olur.

DİĞER BASİT KOD ÇEVİRİCİLER

İkilik kodu BCD ye çevirmek için 74185 kullanılabilir; BCD yi ikilik koda çevirmek için genellikle 74184 kullanılır.

BCD				3 — FAZLA			
J	*K*	*L*	*M*	*A*	*B*	*C*	*D*
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

(a)



(b)

Şekil 3- 44 Bir çoğullayıcı kullanarak BCD den 3- FAZLA koduna çevrim.
***(a)* Doğruluk tablosu. *(b)* Şema.**

Bir BCD kodunu başka bir BCD koduna çevirmeniz gerekiyorsa ve bunu yapacak hazır bir IC bulamazsınız, yine de bir MSI kullanarak bunu gerçekleştirmenin bir yolu vardır. Daha önce bu bölümde, dört giriş değişkenli bir fonksiyonun 74151 veya CD4051 gibi tek bir sekiz- girişli çoğullayıcı kullanarak nasıl gerçekleştirilebileceğini göstermiştik. Aynı tekniği kullanarak, örneğin 4 bitlik standart BCD yi dengi olan 4 bitlik 3- FAZLA BCD koduna çevirebilirsiniz. Şekil 3- 44*a*, her iki kodu da göstermektedir. Şekil 3- 44*b*, 3- FAZLA kodununun A basamağı için uygun çıkışları verecek çoğulayıcı girişlerini göstermektedir. Kalan 3- FAZLA basamakları için çıkışları elde etmek üzere üç tane daha çoğulayıcı kullanılabilir. Dört sekiz- girişli çoğullayıcı, 4 bitlik herhangi bir BCD kodunu 4 bitlik herhangi başka bir BCD koduna çevirebilir. Bu minimum bir çözüm olmayabilir, fakat çok çabuktur ve sayısal elektronikteki bir eğilimin bir örneğidir. Bu eğilim, birçok farklı fonksiyonlar için, sadece programlanması değiştirilerek çok güçlü genel devrelerin kullanılması şeklindedir.

Alfanümerik kodlar arasındaki, örneğin ASCII den EBCDIC e gibi, çevrimler veya dörtten fazla değişkenli birleşimsel mantık tasarımları, çok fazla kapı veya çoğullayıcı gerektirir, bu nedenle de bunlar için yalnız- okunur bel lekleri kullanıyoruz.

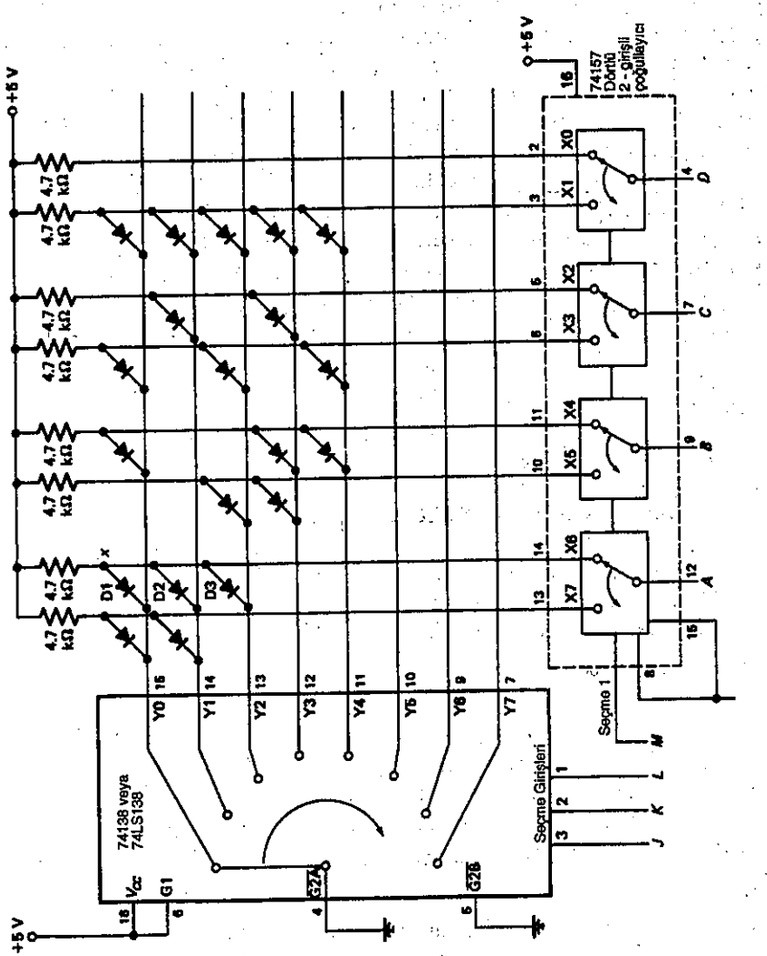
YALNIZ- OKUNUR BELLEKLER

Yalnız- okunur bellekler, yani ROM lar (Read- only memory), bir *X- Y* hatlar ızgarasının kesişme noktalarında ikilik veriyi 1 ler veya 0 lar olarak saklar. Uygun şekilde bu noktaları adresleyerek, burada saklanan veriyi okuyabilirsiniz.

BİR EV- İŞİ ROM

Şekil 3- 45, ROM ların nasıl, çalıştığını ve bir ROM un yukarıda anlatılan standart BCD den- 3- FAZLA- BCD ye kod çevrimini nasıl gerçekleştiğini gösteren bir "ev işi" ROM un devresidir. Ev- işi ROM devresinin bir 74LS138 kod çözücü, bir 74158 çoğullayıcı ve diyotlar içeren bir matristen oluştuğuna dikkat edin.

Veri kitapları, 74LS138 den 1- hattan- 8- hatta yol çoklayıcı (demultipleksör) veya 3- hattan- 8- hatta kod çözücü olarak bahsederler. İkinci moddaki çalışması, daha önce anlatılan 7442 BCD den- onluğa kod çözücünün çalışmasına benzer. Eğer *G1* girişi yükseğe bağlanırsa ve *G2A* ve $\overline{G2B}$ girişleri topraklanırsa, bu takdirde seçme, yani adres, girişlerindeki 3- bitlik ikilik sayı, sekiz çıkış hattından hangisinin alçak olacağını belirler. Her giriş birleşimi için sadece bir çıkış hattı adreslenir ve alçağa çekilir. Örneğin, eğer *JKL*=000 ise Y0 çıkış hattı alçaktır, diğer bütün *Y* çıkışları yüksektir. Eğer *JKL*= 001 ise sadece *Y1* hattı alçak olur.



Şekil 3-45 Diyot matrisi, 74138 ve 74157 kullanarak yapılmış "ev işi" ROM.

Şimdi de 74157 dörtlü iki- girişli çoğullayıcıya bakalım. Eğer seçme hattı *M*= 0 ise, anahtarlar gösterildiği gibi sağdadır ve çıkış verisi *X0, X2, X4* ve *X6* sütunlarından seçilir. 74157 nin seçme girişine uygulanacak bir 1, anahtarları diğer yöne çevirir ve çıkışlar *X1, X3, X5* ve *X7* sütunlarından seçilir. *JKLM*= 0000 girişi için çıkışların ne olacağını bulmaya çalışın. D çıkışı 1 dir, yani yüksektir; çünkü 4,7 KΩ luk bir direnç üzerinden +5 V a bağlanmıştır. Ancak *A* çıkışı biraz daha karışıktır. 74158 in adres hatlarında *JKL*= 000 olduğuna göre *Y0* hattı alçak olacaktır. Bu alçak, *D1* germanyum diyodunu ileri kutuplayacak ve X noktasında yaklaşık 0,6 V luk bir gerilim verecektir. Bu 0,6 V, *D1* in uçları arasındaki gerilim ile *Y0* hattının *TTL* alçak çıkış geriliminin toplamıdır. Çoğulayıcı, 0,6 V u bir 0, yani alçak, olarak görür ve çıkış A alçak olur. (0,6 V luk alçak gürültü payını azaltmıştır, fakat uçlar kısa tutulursa bu bir problem oluşturmaz). *Y1* ve *Y2* hatları açık olduğundan *D2* ve *D3* diyotlarının etkisi yoktur. *X4* sütunu *Y0* hattına bir diyotla bağlıdır, o halde B çıkışı da alçak yani 0 olur. *X2* sütununun, onu aşağı çekecek Y0 hattına bağlı bir diyodu olmadığından, çıkışı 1 olacaktır. O halde *JKLM*= 0000 girişi için ABCD çıkışı 0011 olur. Şekil 3- 44 den bunların birbirine karşılık gelen BCD ve 3- FAZLA kod sayıları olduğunu kontrol edebilirsiniz.

Şimdi *JKLM*= 0001 giriş birleşimini deneyin. 74LS138 in *JKL* adres girişlerinin hepsi hâlâ 0 olduğuna göre *Y0* hattı alçaktır. 74157 nin seçme girişindeki bir 1 ise anahtarı sola çevirmiştir, böylece çıkışlar *X1, X3, X5* ve *X7* den alınacaktır. *X7* hattı alçak *Y0* hattına bir diyotla bağlıdır, o halde *A* çıkışı alçaktır. *X5* sütununun *Y0* hattına bağlı diyodu yoktur, böylece *B* çıkışı yüksektir. X3 ve X1 sütunlarının da *Y0* a bağlı diyotları olduğu için *C* ve *D* çıkışları alçaktır. *JKLM* den verilecek bir 0001 girişi 0100 a eşit, beklenilen ABCD çıkışını vermektedir.

Bu örnekten görebileceğiniz gibi, sadece diyotları koymak veya koymamakla bu ROM da istediğiniz doğruluk tablosunu saklayabilirsiniz. Şekil 3- 45 te bacak numaraları da verilmiştir, böylece isterseniz devreyi kurar ve diğer kodları deneyebilirsiniz. Bu örnek için, bu küçük ROM daki bütün mümkün hatları kullanmanıza bile gerek kalmadığına dikkat edin.

Bir ROM da saklanan bilgiler uçucu değildir, yani gücü kapatıp tekrar açtığınızda sakladığınız bilgiler hâlâ orada olacaktır. Bölüm 5 te karşılaşacağınız bazı başka belleklerde bunun böyle olmadığını göreceksiniz.

ROM TİPLERİ

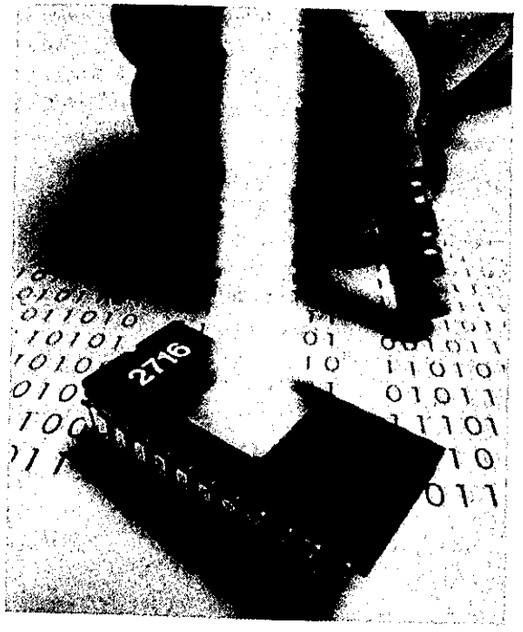
Üç temel tip ROM vardır: Maske programlanmış ROM lar; programlanabilir ROM lar yani PROM lar ve silinebilir programlanabilir ROM lar; yani EPROM ve EEPROM lar.

MASKE PROGRAMLANMIŞ ROMLAR : Maske programlanmış ROM lar, ev- işi ROM da yapıldığı gibi, diyotları veya transistörleri üretim sırasında ekleyerek veya koymayarak sabit programlanmış birimlerdir. Siz üreticiye istediğiniz doğruluk tablosunu delikli kart veya kağıt şerit üzerinde gönderirseniz, orada bir bilgisayar istenilen doğruluk tablosunu verecek ROM entegresi için gerekli maskeyi oluşturur. Programlama sabittir.

PROMLAR : 74S287 gibi programlanabilir ROM lar, yani PROM larda her bit konumunda bir diyot bulunmaktadır. Bu nedenle de başta çıkışların hepsi 0 dır. Ancak her diyot ile seri bir sigorta bağlantısı vardır. Eğer doğru biti adresler ve karşılık olan çıkışa uygun bir darbe gönderirseniz sigorta atar ve o konumda sabit bir 1 kalmış olur. Programlama sabittir, bu nedenle de hataları düzeltmeniz mümkün değildir. Hataları en aza indirmek için kâğıt şerit veya doğru ROM larla kontrol edilen PROM programlama makinaları kullanılır. Nikelkrom sigortalı PROM ların bir problemi zamanla Nikelkrom sigortanın yeniden büyüyerek bir veya daha fazla bitin programlamasını değiştirebilmesidir.

EPROMLAR (ERASABLE PROMS) : Silinebilir programlanabilir ROM lar MOS devreleri kullanır. 1 ve 0 ları IC yongasında gömülü bir tabakada yük paketleri olarak saklarlar. EPROM ların programlanması daha sonra anlatılacaktır. Şimdilik önemli olan nokta, EPROM larda saklanmış verileri, entegrenin üzerindeki kuvars pencereye bir yüksek- şiddetli kısa- dalga boylu ultraviyole ışığı yaklaşık 20 dakika süreyle tutarak silebileceğinizdir. Bu ışık, saklanmış yükün sızarak kaçmasına sebep olur ve çıkış doğruluk tablosunu değiştirmek üzere yeniden programlamanıza imkân verir. Ancak entegrenin içeriğinin tamamının silinmesi bir dezavantajdır. Bir baytı değiştirmek için bütün aygıtı yeniden programlamanız gerekmektedir. Şekil 3- 46, kuvars penceresiyle birlikte Intel 2716 EPROM unun bir fotoğrafını göstermektedir. 2716 gibi bir EPROM, bir ürünü geliştirmekte kullanılabilir; ve sonra tasarım kesinleştikten sonra Intel 2316E gibi daha ucuz maske programlanmış dengi kullanılabilir.

EEPROMLAR (ELECTRICALLY ERASABLE PROMS) : *Elektrikle silinebilir programlanabilir ROM lar (EEPROM lar)* da yukarıda anlattığımız ultraviyole ışıkla silinebilen aygıtlarınkilere çok benzer MOS devreleri kullanır. Veri, yalıtılmış bir tabaka veya yalıtılmış bir kayan kapı üzerine yük koyarak veya koymayarak saklanır. Yalıtılmış tabaka çok ince yapılmıştır. (< 200 *A°*). Böylece 20- 25 V kadar az bir gerilimle, elektrik yükleri bu ince tabakadan, programlamak veya silmek için istenilen yönde geçirilebilir.



Şekil 3- 46 Kuvars penceresi ile Intel 2716 EPROM *(Intel Corporation).*

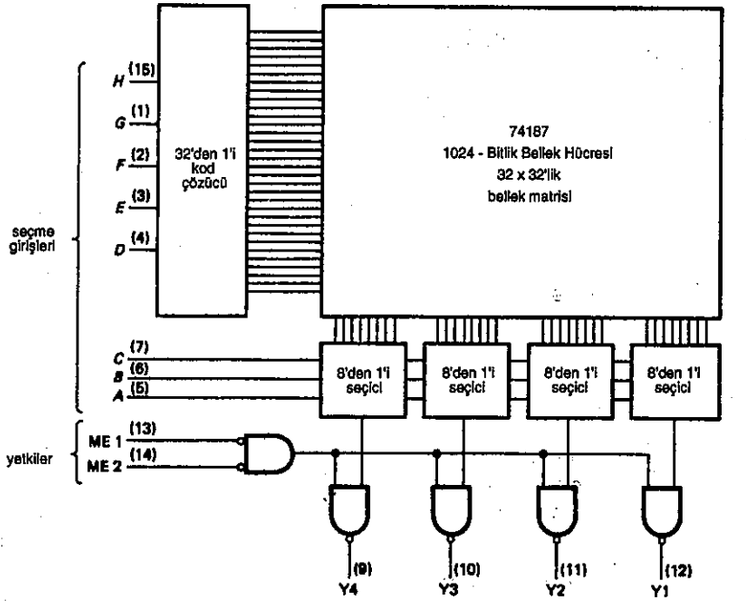
Bu tip EEPROM ların bir örneği 2048 bayt (16384 bit) saklayabilen Intel/ Motorola 2817 dir. Her baytı aygıta yazması, yaklaşık 10 ms gerektirir. Yaklaşık 21 V luk bir elektrik sinyali kullanarak tek bir veri baytı veya bütün aygıt, yaklaşık 10 ms de silinebilir. Bu silme zamanı, ultraviyole EPROM larınkinden çok daha kısadır. Ayrıca, ultraviyole ışık kaynağı gerekmediğinden, aygıt devre içindeyken bile, silme ve programlama çok kolay bir şekilde yapılabilir.

ROM YAPISI VE ADRESLEME

Ev- işi ROM un 8 satırı ve 8 sütunu vardı. Bu 64 mümkün kesişme noktası verir, böylece 64 ikilik bit saklanabilir. Bu bitler 4- bitlik kelimeler olarak çözümlenip çıkışa verildiklerine göre, bu 64- bitlik ROM da onaltı tane 4- bitlik kelime saklayabilirsiniz. Böyle bir ROM a '16x 4 ROM denir.

Şekil 3- 47, 1024 bitlik 74187 ROM unun blok diyagramını göstermektedir. Gerçek bellek matrisi 32x 32 dir (çünkü diziyi kare şeklinde veya ona yakın yapmak çok daha kolaydır), fakat 4 bitlik 256 çıkış kelimesi vermek üzere çözümlenmektedir. 32 sıradan birini seçmek için 5 adres hattı ve 8 sütundan birini seçmek için 3 adres hattı daha gerekir. 256 kelimeden birini seçmek için toplam 8 adres hattı gerekir. 256 sayısı 2^8 e eşit olduğuna göre, 2^N çıkış kelimesinden birini seçmek için N adres hattına gerek olacağını kolayca görebilirsiniz.

Diğer bir örnek, 4096x8 lik düzenlemesi olan 32768- bitlik TMS 2532 dir. 8- bitlik 4096 çıkış kelimesinden birini seçmek için kaç tane adres hattı gerekir? 4096 sayısı 2^{12} ye eşit olduğuna göre, 12 adres hattı gerekir.



Şekil 3- 47 DM 74187 256x 4 ROM un mantık şeması. *(National Semiconductor).*

Çoğu büyük ROM ların üç- durumlu çıkışları vardır, böylece Şekil 3- 48 de gösterildiği gibi, birçok ROM ortak bir paralel veri yoluna bağlanabilir. İstenilen ROM, yonga seçme ($\overline{CS}$) girişini alçak yaparak seçilir. Eğer $\overline{CS}$ yüksekse, ROM un çıkışları kayar durumdadır. Yonga seçme sinyalleri, adres hatlarından aşağıdaki şekilde çözümlenir.

2732, bir 4096x 8 lik EPROM dur, böylece ROM daki 4096 kelimeden birini adreslemek için 12 adres hattı gerekmektedir. 74LS138 veya 3205 üç- hattan-sekiz hatta kod çözücünün girişlerini seçmek için de daha önce açıklandığı gibi 3 ilave adres hattı kullanılır. Her üç- bitlik birleşim sekiz çıkıştan birinde bir alçak üretecektir. Eğer kod çözücü çıkışlarından herbiri ayrı bir ROM un yonga seçme girişine bağlanırsa, herhangi bir anda ROM larda yalnız birisi seçilecektir. Bazı ROM ların birden fazla yonga seçme veya yetkilendirme girişleri vardır.

ROM VERİ SAYFASI PARAMETRELERİ VE EPROM PROGRAMLAMA

Ek C, 8- bitlik 2048 kelime saklayan Intel 2716 EPROM unun veri sayfalarının tamamını vermektedir. Bu aygıt, daha önceki EPROM larda bulunmayan bazı ilginç özelliklere sahip olduğu için seçilmiştir. İlk olarak, yüksek-yoğunluklu, yüksek- hızlı bir MOS aygıtı olmasına rağmen, programlama dışında, tek bir +5 V luk kaynakla çalıştığına dikkat edin. İkinci olarak, girişler ve çıkışlar kesinlikle TTL ile uyuşmaktadır. Çıkışlar, bir Schottky- TTL girişi bile sürebilir. Üçüncü olarak, aygıt, seçilmediği zaman, aktif modda kullandığı gücün dörte biri kadar az güç kullandığı bekleme moduna alınabilir.

Veri sayfalarındaki Tablo 1 deki mod seçme çizelgesi, normal okuma işlemi için hem $\overline{CE}$ / $\overline{PGM}$ hem de $\overline{OE}$ girişlerinin alçak olması gerektiğini göstermektedir. Bu iki girişten biri yüksekse, çıkışlar yüksek- empedans durumundadır (kayar durumda). $\overline{CE}$ / $\overline{PGM}$ girişi yüksek yapıldığında gücü düşürüp, çıkışları kayar duruma getirdiğinden, bir kod çözücü ile birçok ROM dan birinin seçilmesi için $\overline{OE}$ girişi yerine kullanılabilir.

Şimdi de ROM hakkında bazı önemli zamanlama bilgileri edinmek için veri sayfalarındaki a. a. karakteristikleri tablosuna bakın. İlk bakışta, tablo karmaşık bir indisler koleksiyonu gibi görülebilir, fakat tablonun altındaki dalga şekilleri açıklanmasına yardımcı olmaktadır.

Erişim zamanı t_{ACC}, ROM un girişlerine geçerli bir adresin gelmesiyle çıkışlarında geçerli verinin görülmesi arasında geçen zamandır. Bu erişim zamanında, yonga seçme girişindeki bir alçağın yongayı bekleme modundan çıkarması için gerekli t_{CE} zamanı da içerilmiştir. $\overline{CE}$ sinyali büyük bir ihtimalle adres hatlarını çözümleyerek üretildiğinden, $\overline{CE}$ nin yüksekten- alçağa geçişi, gösterildiği gibi, geçerli adresler oluştuktan sonra gerçekleşecektir. Eğer bir CMOS kod çözücü kullanılırsa, bunun çok uzun yayılma gecikmesi, t_{CE} ye eklenip toplam erişim zamanını artırabilir. Bütün bunlara dikkat etmenizdeki sebep, eğer erişim zamanı çok uzunsa, ROM belli bir sistemde çalışabilecek kadar hızlı bir şekilde verileri dışarıya veremeyebilir.

İkinci erişim zaman t_{OE}, çıkışı, yüksek empedans modundan aktif okuma moduna geçirmek için gereken zamandır. Eğer t_{OE} sinyali de yukarıda önerildiği gibi adres hatlarını çözümleyerek üretiliyorsa, kod çözücünün t_{PD} si de t_{ACC} içine konmalıdır. Tablodaki son iki terim için verilen 0 ns lik minimum zamanlar, $\overline{OE}$ yükseldiği, $\overline{CS}$ yükseldiği veya adresler değiştiği andan itibaren çıkış verilerinin hâlâ geçerli olacağına güvenmemenizi söylemektedir.

Şimdi üzerinde duracağınız nokta bu EPROM un programlanmasıdır. Her EPROM tipinin kendi özel programlama yordamı vardır, bu nedenle, herhangi bir tiple ilgili yordamı, bu tiple çok miktarda programlama yapmayacaksanız, ezberlemenin hiçbir yararı yoktur. Çoğu hallerde bir EPROM programlama makinası, işi sizin için yapacaktır. Ancak, size kabaca bir fikir verebilmek için 2716 nın programlama yordamını özetleyeceğiz. Yongaya +5 V luk bir V_{cc} ve +25 V luk bir V_{PP} güç vermektedir. $\overline{OE}$ yüksek tutulur. Kelimeyi saklamak istediğiniz yer adreslenir. Saklamak istediğiniz veri kelimesi 2716 nın çıkışlarına uygulanır. Bundan sonra bir tek 50 ms lik, pozitif TTL seviyesi darbe $\overline{CE}$/PGM girişine uygulanır. Veriyi çıkışlardan kaldırarak ve $\overline{OE}$ yi alçağa çekerek programlamanın yapıldığını görebilirsiniz. Şimdi çıkışların, programladığınız kelimeyi size gerisin geriye okuması gerekir. Belleğin tamamı veri sayfalarında belirtildiği gibi ultraviyole ışıkla silinebilir.

ROM UYGULAMALARI

KOD ÇEVİRİCİ OLARAK : Daha önce bu bölümde bir ROM un kod çevirici olarak kullanılışını göstermiştik. Bu kullanışın başka örnekleri MM4221RR ASCII- 7 den- EBCDIC e çevirici, MM4230KP ASCII- 7 den- Selectric koduna çevirici ve MM4230BO Hollerithden- ASCII- 8 e çeviricidir. Gerekli çevrim için bir standart ROM yoksa, her zaman bir PROM veya EPROM u işi yapmak üzere programlayabilirsiniz.

BAŞVURUSU TABLOSU OLARAK : ROM lar başvuru tablosu olarak da kullanılabilirler. Örneğin, MM4232, 0 ile 90° arasındaki açıların 9- bitlik ikilik denklerini giriş olarak kabul eder. Çıkışlar ise giriş açısının sinüsünün 8-bitlik ikilik dengidir. Birçoğu birleştirilerek 16- bitlik çözünürlük sağlanabilir.

KARAKTER ÜRETİCİ : 2513 gibi ROM karakter üreteçleri, bir video terminal ekranında harf ve sayıları göstermek için gerekli nokta desenlerini üretmek üzere kullanırlar.

KLAVYE KODLAYICISI : MM5740 gibi ROM klavye kodlayıcıları, daktilo tipi klavyelerde üretilen anahtar- kapatmalı kodları bilgisayar kullanımı için ASCII ye çevirmek üzere kullanılırlar.

MANTIK KAPISI YERİNE: ROM ların diğer bir uygulaması, çok sayıda giriş değişkeni olan mantık ifadelerini gerçekleştirmek için kullanılması gereken birçok mantık kapısının yerine kullanılmalarıdır. Bir sekiz- girişli ROM, sekiz giriş değişkenli herhangi bir çıkış ifadesini vermek üzere programlanabilir.

BİLGİSAYAR BELLEĞİ : ROM ların ana kullanım yeri, bir mikroişlemci veya bilgisayar için ikilik kodlanmış komutları ve verileri saklamak içindir. Komutlar ROM da sıralı bir şekilde saklanmışsa, bu takdirde mikroişlemci bir komut okur, komutun gereklerini yerine getirir ve sonra bir sonraki komutu almak için bir üstteki adrese geçer. Şekil 3- 48 deki ROM dizisi 8- bitlik 32000 kelime, yani bayt, saklayabilir.

PLA LAR, FPLA LAR, FPGA LAR VE PAL LAR

PROGRAMLANABİLİR MANTIK DİZİLERİ (PROGRAMMABLE LOGIC ARRAYS: PLAS) VE YERİNDE PROGRAMLANABİLİR MANTIK DİZİLERİ (FIELD PROGRAMMABLE LOGIC ARRAYS: FPLAS)

Bir PLA bir üniversal çarpımlar- toplamı (VE- VEYA) fonksiyon üreteci olarak düşünülebilir. Şekil 3- 49, National DM7575 maske programlanabilir mantık dizisinin mantık şemasını göstermektedir. *I1* den *I14* e kadar *14* tane veri veya adres vardır. Ters çeviriciler, bu veri girişlerinin kendilerini ve tümleyenlerini yatay hatlar halinde üretmektedir. Düşey hatlar 96 tane öndört- girişli VE kapısının girişlerine bağlanmıştır. Her VE kapısının girişinin hangi yatay hatlara bağlanacağını kullanıcı belirler. Hiçbir yatay hatta bağlanmamış bir VE girişi yüksekmiş gibi davranır. Bu, giriş değişkenlerinin hangi fonksiyonunun her VE kapısının çıkışında yüksek oluşturacağını belirler. VE kapılarının çıkışları bir başka matrisin sütunlarını oluşturur. Bu yatay hatlar sekiz tane 8- girişli VEYA kapısının girişleridir. Hangi düşey hatların VEYA girişlerine bağlanacağını kullanıcı belirler. Hiçbir düşey hatta bağlanmayan VEYA girişleri içerden alçak tutulur. Normal veya ters çevrilmiş çıkışlar da maske programlanabilir.

Çıkış ifadesi

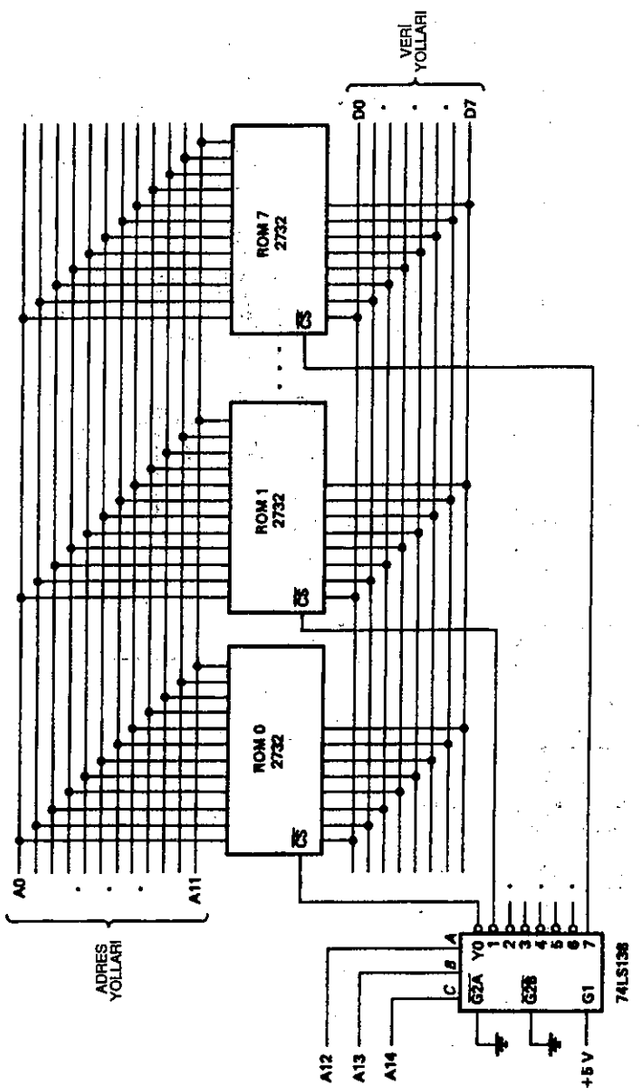
$F1 = I1\ I2\ I3\ I4\ I5 + \overline{I1}\ \overline{I2}\ I3\ I11\ I14 \ldots$

gibi bir çarpımlar- toplamı yapısına sahiptir.

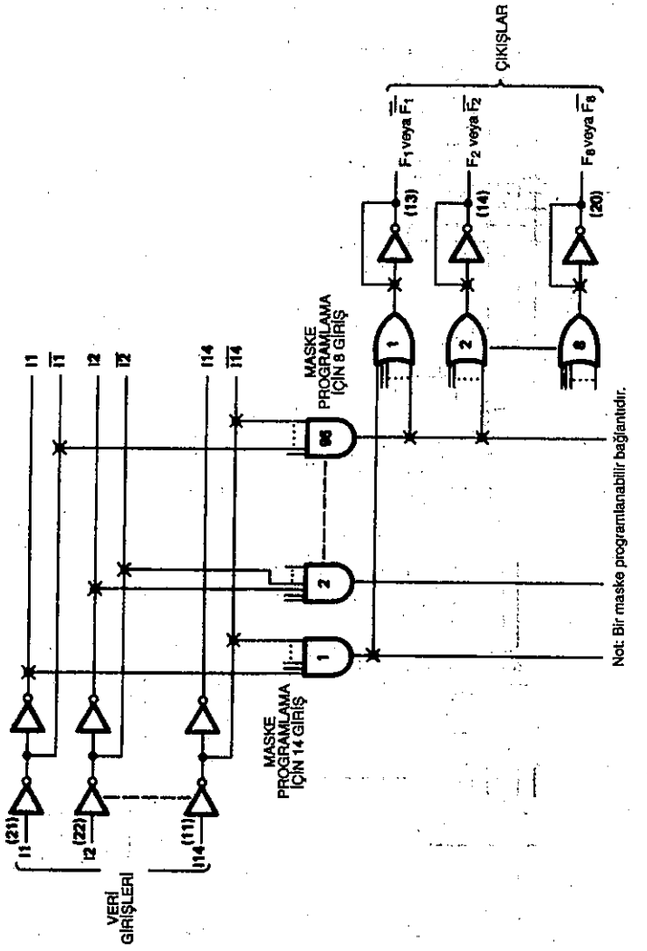
Bir çıkış ifadesi 8 taneye kadar çarpım terimi içerebilir. Her çarpım terimi 14 taneye kadar değişken içerebilir. Ancak, sadece 96 tane VE kapısı olduğundan, farklı çarpım terimlerinin toplam sayısı sadece 96 dır.

Bir PLA, bir kart dolusu mantık kapısının yerine veya özel amaçlı kod çevirici olarak kullanılabilir. Çarpımlar- toplamı şeklindeki çıkış formatı iç matris bağlantılarının doğrudan doğruluk tablosundan belirlenmesini kolaylaştırır.

Bu PLA için iyi bir uygulama örneği 12 bitlik Hollerithden- ASCII- 7 ye kod çeviricidir. Hollerith kodu 12 bitliktir, fakat mümkün birleşimlerin sadece 96 tane kadarını kullanır. O halde, PLA, Hollerith de kullanılan 96 giriş birleşiminin her biri için ASCII 7- bitlik çıkış kodunu vermek üzere doğrudan programlanabilir. Bu işi doğrudan ROM la yapmak için çok daha büyük bir ROM gerekmektedir. MM4230BO Hollerith den- ASCII- 8 e ROM u, çevrimi yapmadan önce 12 Hollerith hattını 8 e düşürmek için ilave harici mantık gerektirir. Signetics 82S100 gibi yerinde programlanabilir mantık dizileri, yani PLA lar, bipolar PROM larda yapıldığı gibi sigortalar attırılarak programlanır.



Şekil 3- 48 8- bitlik 32768 kelime saklayabilmek için sekiz Intel 2732 EP-ROM kullanan bellek dizilimi.



Şekil 3- 49 Bir programlanabilir mantık dizisinin (PLA) mantık şeması *(National Semiconductor).*

YERİNDE PROGRAMLANABİLİR KAPI DİZİLERİ (FIELD PROGRAMMABLE GATE ARRAYS: FPGA) VE PROGRAMLANABİLİR DİZİ MANTIĞI (PROGRAMMABLE ARRAY LOGIC: PAL)

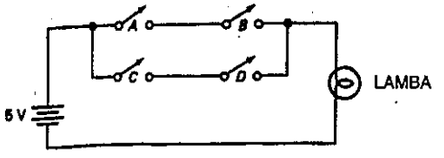
PLAveya FPLA ya benzer diğer bir aygıt yerinde programlanabilir kapı dizisi, yani FPGA dır. FPGA, Şekil 3- 49 daki PLA gibi bir VE kapısı matrisine sahiptir, ancak çıkış VEYA kapıları yoktur. Bu nedenle, çıkış ifadesi sadece giriş değişkenlerinin programlanabilir VE fonksiyonudur. Bu aygıtlar bazan rastgele- mantık kapılarını değiştirmek üzere kullanılırlar.

VE- VEYA matris tipi diğer bir aygıt da programlanabilir dizi mantığı, yani PAL, diye bilinir. PAL ler Şekil 3- 49 daki PLA gibi programlanabilir VE kapısı matrisi kullanır, fakat çıkış VEYA kapı bağlantıları sabittir. Bu aygıtı programlamayı kolaylaştırır. Her çıkış için çıkış ifadesi PLA daki gibi çarpımlar- toplamı şeklindedir.

Buradaki önemli nokta, bu aygıtlardan birinin birçok zaman, ayrık kapı entegreleriyle dolu bir baskılı devre kartının tamamını değiştirebileceğidir.

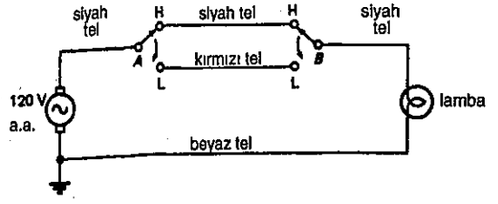
GÖZDEN GEÇİRME SORULARI VE PROBLEMLER

1. Şekil 3- 50 deki anahtar devresinin boole ifadesini yazın ve mantık kapısı şemasını çizin. Kapalı anahtar= 1 dir.



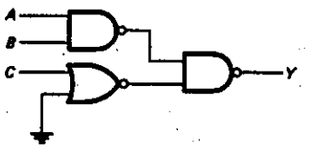
Şekil 3- 50

2. Şekil 3- 51, evde kullanılan bir anahtar devresinin şemasını göstermektedir. Işık her bir anahtarla yakılıp söndürülebilmektedir. Devrenin boole ifadesini yazın. Anahtar yukarıda *H*, aşağıda *L* dir ve ışık yanık çıkış *H* dir.



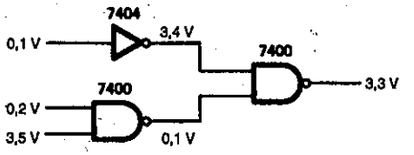
Şekil 3- 51

3. Şekil 3- 52 deki devrenin doğruluk tablosunu kurun ve çıkış ifadesini bulun.



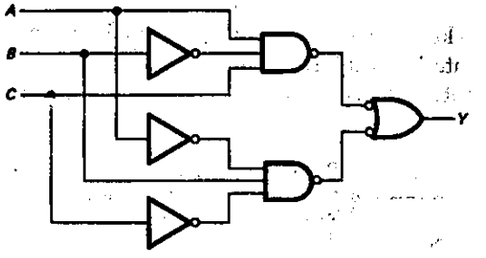
Şekil 3- 52

4. *Y= B. C+D. E* ifadesinin denk devresini çiziniz.
5. *Y= (A+B). (C+D)* ifadesinin denk devresini çiziniz.
6. Şekil 3- 53 te gösterilen TTL mantık devresi doğru çalışmamaktadır. Bir teknisyen her kapının giriş ve çıkış gerilimlerini ölçmüş ve sonuçları şekil üzerine yazmıştır. Devrenin neresinde bir problem vardır? İki mümkün sebep söyleyin.



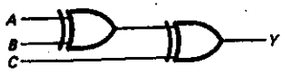
Şekil 3- 53

7. Şekil 3- 54 teki devrenin çıkış ifadesini yazınız.



Şekil 3- 54

8. a. Şekil 3- 55 teki devrenin doğruluk tablosunu gösterin ve çıkış ifadesini yazın.



Şekil 3- 55

b. Bir doğruluk tablosuyla, bir özel- VEYA kapısının bir giriş sinyalini, diğer girişin durumuna bağlı olarak ters çevirmek veya ters çevirmemek için nasıl kullanılabileceğini gösterin.

9. Boole cebirini kullanarak, aşağıdakileri basitleştirin.

a. $(A+\overline{B}).(A\overline{B}+C).C$

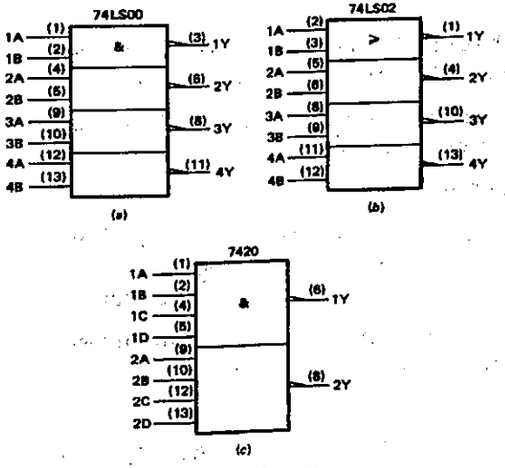
b. $(A+\overline{B}).C+(A\overline{B}+C)$

c. $AB+ABC+\overline{A}B+A\overline{C}$

d. $\overline{\overline{A.B}+\overline{C.D}}$

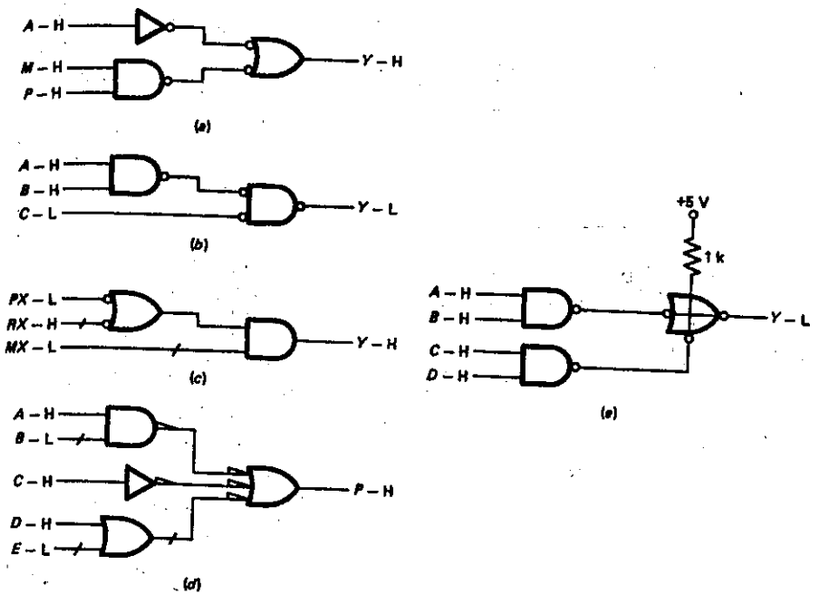
e. $\overline{A+\overline{B+\overline{CD}}}$

10. Şekil 3- 56 da bağımlılık gösterilişi sembolleri ile gösterilmiş olan her aygıt için girişler ve çıkışlar arasındaki ilişkiyi tarif edin.



Şekil 3- 56

11. Şekil 3- 57 de gösterilen her mantık devresinin boole ifadesini yazınız.



Şekil 3- 57

12. Problem 9a, b ve c deki ifadelerin karma mantık devrelerini çizin. Bütün giriş sinyallerinin aktif yüksek olduğunu kabul ediniz.

13. Aşağıdaki ifadeyi boole cebiri kullanarak basitleştirin. Ayrıca Karnaugh haritası kullanarak basitleştirme yapın.

$X= AB+ABC+\overline{A}B+AC$

Basitleştirilmiş ifadeyi kapılarla nasıl gerçekleştireceğinizi gösterin. Minimum ifadeyi elde etmek için mümkün olduğunca fazla kareyi 2, 4, 8 veya 16 lık gruplar olarak gruplamaya çalışmayı unutmayın.

14. Aşağıdaki ifadeyi basitleştirmek için Karnaugh haritası kullanın ve sonra basitleştirilmiş ifadenin devresini gösterin.

$X=AB\overline{C}\overline{D}+A\overline{B}CD+ABCD+AB\overline{C}D+\overline{A}B\overline{C}D+\overline{A}BCD$

15. Üç sekreterle bir patron bir büroda çalışmaktadır. Eğer patron veya en az iki sekreter isterse radyoyu açacaklardır.

a. Radyo AÇIK fonksiyonu için doğruluk tablosunu kurun. (Doğruluk tablosunda bir EVET i 1 eşit alın ve radyo AÇIK ı 1 e eşit alın.)

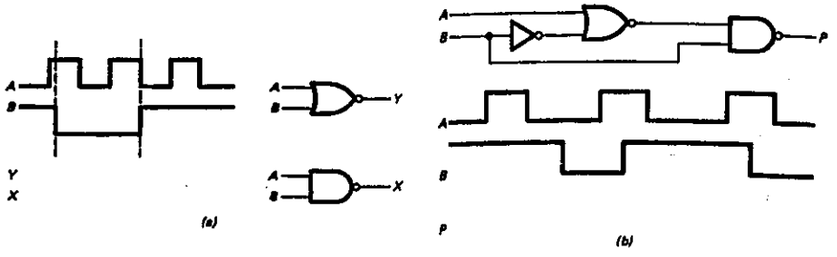
b. Karnaugh haritası kullanarak basitleştirilmiş ifadeyi bulun.

c. Sadece VE DEĞİL kapıları kullanarak fonksiyonun nasıl geçekleştirilebileceğini gösterin.

d. Sadece VEYA DEĞİL kapıları kullanarak fonksiyonun nasıl gerçekleştirileceğini gösterin.

16. Şekil 3- 58 deki devrelerde verilen giriş dalgaşekilleri için üretilecek çıkış dalgaşeklilerini gösterin. Çıkışın, devrenin doğruluk tablosuna her an uyması gerektiğini unutmayın.

Bu gerçeği kullanarak, çıkış dalgaşeklini giriş dalga şeklinin altına parça parça çizerek oluşturabilirsiniz.



Şekil 3- 58

17. İki girişli ve bir çıkışlı bir anahtarın sadece VE DEĞİL kapıları ile yapılmış ve sadece VEYA DEĞİL kapıları ile yapılmış devresini gösteriniz.

18. Çoğullamanın avantajı nedir?

19. Problem 14 teki fonksiyonun bir 74LS151 çoğullayıcı kullanarak nasıl gerçekleştirilebileceğini gösteriniz.

20. Aşağıdaki sayıları sekiz- tabanlı sayıları çevirin.

a. 1010101_2

b. 10_{10}

c. 111001011_2

21. On- tabanlı (onluk) sayılara çevirin.

a. 314_8

b. 72_8

c. 40_8

22. Onaltılık sayılara çevirin.

a. 42_{10}

b. 798_{10}

c. 01011101_2

d. 11000011_2

23. Onluk sayılara çevirin.

a. $A7_{16}$

b. $3FF_{16}$

c. 40_{16}

24. BCD ye çevirin.

a. 72_{10}

b. 59_{10}

c. 37_{10}

25. ASCII- kodlanmış bir klavyenin *M* tuşuna basılmıştır. Klavyeden gelen yedi paralel veri hattında nasıl bir 1 ler ve 0 lar deseni bulmayı beklersiniz? Bir satırbaşı *(CR)* hangi deseni verecektir?

26. Pariteyi tarif edin ve iletilen verideki hatayı bulmada nasıl kullanıldığını anlatın.

27. EBCDIC ve Selectric kodlarında *G* harfinin 1 ler ve 0 lar desenleri nelerdir?

28. Şekil 3- 43 te gösterilen 74148 öncelik kodlayıcıda, eğer 2 numaralı veri girişine bir alçak uygulanırsa *A0, A1* ve *A2* çıkışlarında hangi kod görülecektir?

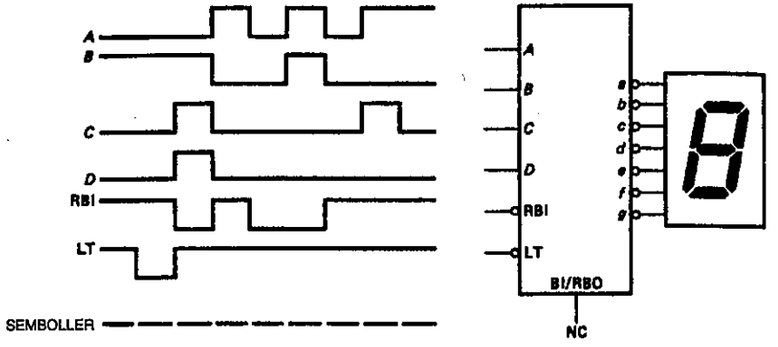
29. Şekil 3- 42 deki 7447 BCD- den- yedi parçalıya kod çözücüye bakın.

    a. $\overline{BI}$ girişini alçak yapmanın parça çıkışları üzerindeki etkisi ne olacaktır?

    b. Lamba test girişi aktif alçak mı, yoksa aktif yüksek midir?

    c. Girişlere 5 in BCD kodu uygulanmışsa, yedi- parçalı çıkışlarında hangi mantık seviyeleri görülecektir?

30. Bir 7447 BCD-den- yedi- parçalıya kod çözücü/sürücü verildiğine göre, Şekil 3- 59 daki göstergede görülecek olan sayı dizisini belirleyin.



Şekil 3- 59

31. Şekil 3- 45 teki ev- işi ROM matrisinde 1110 giriş adresi için 0011 çıkış kelimesini üretmek için nerelere diyotlar koyacağınızı gösterin.

32. National Semiconductor INS8298, 8- bitlik 8192 kelime yani bayt olarak düzenlenmiş 65536 bitlik bir ROM dur. 8192 bayttan birini adreslemek için kaç tane adres hattı gerekmektedir?

33. PROM larla EPROM lar arasındaki farkları anlatın.

34. EPROM lar nasıl silinir?

35. Niçin çoğu ROM ların üç- durumlu çıkışları vardır?

36. Şekil 3- 48 de ROM 7 yi seçmek için 74LS138 kod çözücünün $A12$, $A13$ ve $A14$ girişlerinde hangi mantık seviyeleri bulunmalıdır?

37. 2716 EPROM un en- kötü- durum erişim zamanı t_{ACC} nedir? (Ek C deki veri sayfasını kullanın).

38. Bir PLA ile bir ROM arasındaki farklılıklar nedir?

REFERANSLAR / BİBLİYOĞRAFYA

Winkel, David, and Franklin Prosser: The Art of Digital Design, Prentice-Hall, Englewood Cliffs, NJ, 1980.

Fletcher, William I.: An Engineering Approach to Digital Design, Prentice-Hall, Englewood Cliffs, NJ, 1980.

Peatman, John B.: Digital Hardware Desing, McGraw- Hill, New York, 1980.

Texas Instruments Staff: TTL Data Book for Desing Engineers, 2. baskı., Texas Instruments, Inc., Dallas, 1981

Texas Instruments Staff: Bipolar Microcomputer Components Data Book, 3. baskı., Texas Instruments, Inc., Dallas, 1981

Biakeslee, Thomas R.: Digital Desing with Standard MSI & LSI, Wiley, New York, 1975.

National Semiconductor Staff: CMOS Databook, National Semiconductor Corporation, Santa Clara, CA, 1980.

National Semiconductor Staff: Logic Databook, National Semiconductor Corporation, Santa Clara, CA, 1981.

Intel Staff: Intel Component Data Catalog, Intel Corporation, Santa Clara, CA, 1982

National Semiconductor Staff: MOS/LSI Databook, National Semiconductor Corporation, Santa Clara, CA, 1980.

BÖLÜM 4

FLIP-FLOPLAR, SAYICILAR VE KAYDEDİCİLER

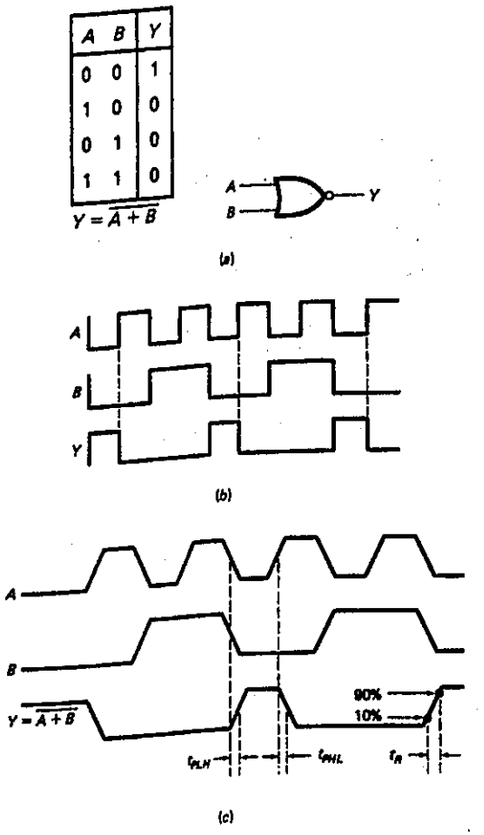
Bu kitabın kalan bütün kısımlarında anlatılan saatli devrelerde zamanlama çok önemlidir. Saatli devrelerin analizini yapmak için sadece doğruluk tabloları değil ayrıca zamanlama diyagramları da kullanırız. Zamanlama diyagramları, devreye çok kanallı bir osiloskopla baktığınızda göreceğiniz türde giriş ve çıkış dalgaşekillerinin çizimlerinden ibarettir. Şekil 4-1 *a* ve *b*, bildiğimiz VEYA DEĞİL kapısının doğruluk tablosunu ve onun zamanlama diyagramı dengini göstermektedir. Şekil 4-1 *c* de, zamanlama diyagramlarının bir kapının yükselme zamanlarını ve yayılma gecikmelerini gösterecek şekilde çizilebildiğine dikkat edin. Saatli devrelerin analizini yapmak için diyagramlara niçin gereksinim olduğunun bir örneği Şekil 4-2 de gösterilmiştir. Giriş dalga şekilleri Şekil 4.1-*b* dekilerle aynıdır, fakat *B* dalgası bir şekilde 30 ns geciktirilmiştir. $Y = \overline{A+B}$ nin elde edilen çıkışı, *X* ve *Y* noktalarında 30 ns genişliğinde istenmiyen darbeler, (çoğunlukla bunlara "gliç" denir) göstermektedir. Şekil 4-2 *b* bu problemi oluşturacak bir devre göstermektedir. Eğer *C* ve *D* girişleri yüksekse, 1 ve 2 kapıları B sinyali için sadece ters çeviriciler olarak davranırlar. İki ters çevrim kısaldığına göre, Şekil 4-2 a da görüldüğü gibi, 1 ve 2 kapılarının yayılma gecikmeleri VEYA DEĞİL kapısına (kapı 3) *B* sinyali girişini sadece geciktirir. Bir çözüm, *A* sinyalinin yoluna da aynı miktarda bir gecikme eklemektir. Bunu, *A* sinyali ile seri iki ters çevirici veya ters çevirici olarak kullanılan iki VE DEĞİL kapısı ekleyerek yapabilirsiniz.

Bu tartışmalardaki ana nokta, darbeli veya saatli mantık devrelerinin analizini yaparken zamanlama diyagramları kullanmanın önemini göstermektir.

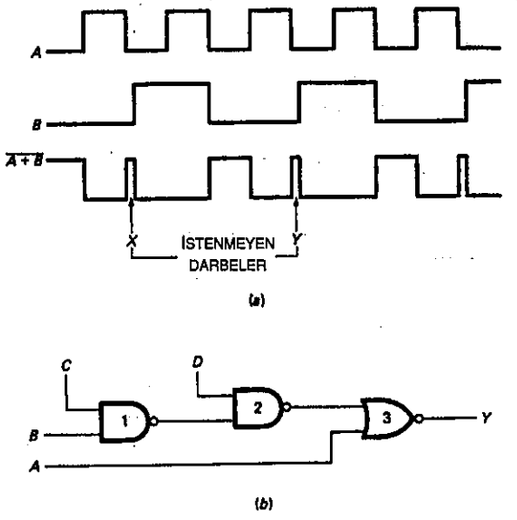
Bu bölümü tamamladıktan sonra şunları yapabileceksiniz:

1. *D* mandalı, *D* flip-flopu, *JK* flip-flopu ve *T* flip-flopunun doğruluk tablolarını yazabileceksiniz.
2. *D* mandallara, *D* flip-floplara ve *JK* flip-floplara uygulanan giriş dalgaşekilleri verildiğinde oluşacak çıkış dalgaşekillerini gösterebileceksiniz.
3. Bir 4-bitlik ikilik veya herhangi bir modulo sayıcının çıkış zamanlama diyagramını çizebileceksiniz.

4. Bir sayıcının herhangi bir durumunu çözümleyecek bir devreyi çizebileceksiniz.
5. Bir 4-bitlik ikilik senkron sayıcıyı 2 ile 15 arasında herhangi bir moda çevirecek bir devre çizebileceksiniz.
6. Bir sayısal saatin çalışmasını açıklamak için bir şematik diyagram kullanabileceksiniz.
7. Bir MSI veya LSI frekans sayıcının çalışmasını açıklamak için bir şematik diyagram kullanabileceksiniz.
8. Paralel verilerin geçici olarak saklanabilmesi için bir *D* flip-flop kaydedicisinin devresini çizebileceksiniz.
9. *D* veya *JK* flip-flopları kullanarak bir kaydırma kaydedicisinin şemasını çizebileceksiniz.
10. Statik MOS ve dinamik MOS kaydırma kaydedicilerinin aralarındaki farkları listeleyebileceksiniz.
11. *UART* ların (Universal asynchronous receiver-transmitters = *Üniversal asenkron alıcı-vericiler*) paralel verileri seri şekle veya seri verileri paralel şekle çevirmede nasıl kullanılacağını gösteren bir devre çizebileceksiniz.
12. MOS kova grupları ve CCD (charge-coupled device = yük - kupleli eleman) lerin çalışmalarını anlatabileceksiniz.
13. Flip - flopların, sayıcıların ve kaydırma kaydedicilerinin işlevlerini bunların bağımlılık gösterilişi sembollerine bakarak açıklayabileceksiniz.



Şekil 4.1 VEYA DEĞİL kapısı.
(a) Doğruluk tablosu.
(b) İdeal zamanlama diyagramı.
(c) Yükselme zamanlarını ve yayılma gecikmesini gösteren zamanlama diyagramı.



Şekil 4.2 "İstenmeyen darbe"ler.
(a) İstenmeyen darbeleri gösteren zamanlama diyagramı.
(b) İstenmeyen darbeleri üretebilen bir devre.

MANDALLAR VE FLİP-FLOPLAR

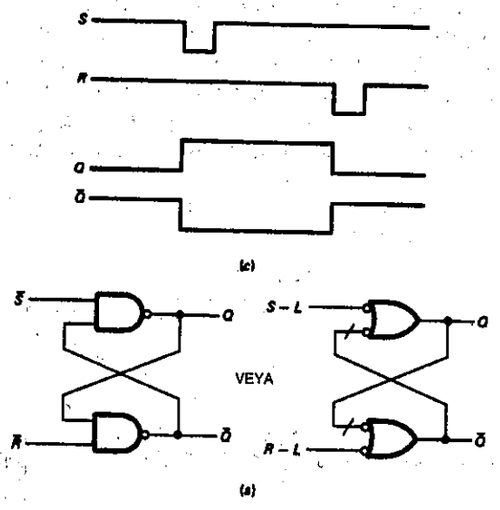
Mandallar ve flip-floplar çıkışlarında ya bir 1 veya bir 0 saklayan bellek devreleridir. En basiti dışında hepsinin çıkışı, yalnız bir özel saat veya yetki girişine darbe gönderildiğinde değişebilir.

VE DEĞİL MANDALI (NAND LATCH)

Şekil 4-3 basit bir VE DEĞİL mandalının şemasını, doğruluk tablosunu ve zamanlama diyagramını göstermektedir. Önce giriş ve çıkış isimlerine dikkat edin. Q ve $\overline{Q}$ tümleyici çıkışları gösterir. S, set veya preset girişi ve R de reset veya temizleme girişidir. Şimdi Şekil 4-3 b deki doğruluk tablosuna bakın. $S = 0$ ve $R = 1$ için Q çıkışı 1 dir. Q çıkışındaki bu 1 devre set edilmiştir diye anlatılır. $R = 0$ ve $S = 1$ için Q çıkışı 0 dır. Buna da reset durumu denir. Şimdi $S = 0$ ve $R = 0$ ı deneyin. Bölüm 3 ten VE DEĞİL kapısının özelliklerini hatırlarsanız, bir VE DEĞİL kapısına uygulanacak herhangi bir alçak giriş çıkışını yüksek yaptığına göre Q ve $\overline{Q}$ çıkışlarının her ikisi de yüksektir. Bu, Q ve $\overline{Q}$ ın bir-

birinin tümleyeni olma tanımını zedelemektedir, bu nedenle bu duruma **belirsiz** veya **yasaklanmış durum** deriz ve tabloda bir yıldız (*) la gösteririz. Diğer bir deyişle, S ve R nin her ikisi de alçaksa, çıkışlardan faydalı bir bilgi elde edemezsiniz. $S = 1$ ve $R = 1$ için tablodaki Q harfi, S veya R den birisi alçak yapılıncaya kadar Q çıkışının bulunduğu durumda sabit kalacağını söylemektedir. Bu duruma (**mandallanmış**) durum denir.

Aşağıdaki örnek, devrenin nasıl kullanıldığını göstermektedir. Şekil 4-3 c de gösterildiği gibi eğer başlangıçta her iki girişte yüksek ve Q alçaksa ve S girişine bir alçak darbe uygulanırsa, Q çıkışı yükseğe sürülür. Belli bir minimumun ötesinde S deki darbenin genişliğinin önemi yoktur. Şimdi Q çıkışındaki yüksek orada tutulur, yani oraya kilitlenir. Devreye set oldu denir.



FONKSİYON	S	R	Q	$\bar{Q}$
Geçersiz	0	0	*	*
Set	0	1	1	0
Reset	1	0	0	1
Kilitle	1	1	Q	$\bar{Q}$

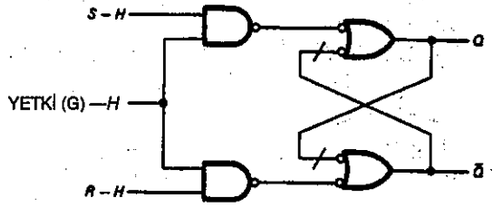
(b)

Şekil 4.3 Çapraz bağlı VE DEĞİL kapısı mandalı. *(a)* Şema. *(b)* Doğruluk tablosu. *(c)* S ve R darbelerinin çıkışlar üzerindeki etkilerini gösteren dalgaşekilleri.

Devreyi reset etmenin, yani Q çıkışını tekrar alçağa değiştirmenin tek yolu, Şekil 4-3 *c* de görüldüğü gibi S girişi yükseldikten sonra R girişine bir alçak darbe uygulamaktır. Şekil 2-31 *b* deki anahtar zıplamalarını giderici devreyi gözden geçirin. Bu devrenin aslında, bir defa çıkış kilitlendimi girişte ne olduğunun önemi yoktur; gerçeğinden yararlanan bir VE DEĞİL mandalı olduğuna dikkat edin.

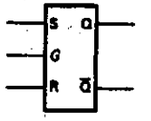
YETKİ (ENABLE) GİRİŞLİ RS MANDALI

Biraz önce anlattığımız VE DEĞİL mandalının iki önemli problemi vardır. Birincisi, girişlerin ikisinin de aynı anda alçak olmamasını garanti etmeniz gerekmektedir. İkincisi, devre asenkrondur. Bunun anlamı, S veya R darbesinin çıkışı ne zaman değiştireceğini kontrol eden bir girişi yoktur. Şekil 4-4 a daki devrede gösterildiği gibi bir yetki girişi ilave edilebilir. Bu devrede, yetki girişi alçak olduğu sürece, girişteki VE DEĞİL kapılarının çıkışları yüksektir ve Q ve $\bar{Q}$ çıkışlar kilitlenmiştir. Sadece yetki girişi yüksekken R ve S girişleri Q ve $\bar{Q}$ çıkışlarını etkileyebilir. Yetki girişi alçağa gitmeden hemen önceki S ve R girişlerinde bulunan durumlar Q ve $\bar{Q}$ çıkışlarına kilitlenecek durumları belirler. Şekil 4-4*b* deki doğruluk tablosunda, girişteki VE DEĞİL kapılarının, mandalı set veya reset etmek için gerekli S veya R darbelerinin yönlerini ters çevirdiğine dikkat edin.



G	S	R	Q	$\bar{Q}$
0	X	X	Q	$\bar{Q}$
1	0	0	Q	$\bar{Q}$
1	0	1	0	1
1	1	0	1	0
1	1	1	*	*

(b)



(c)

Şekil 4.4 Yetkili RS mandalı. *(a)* İç şeması. *(b)* Doğruluk tablosu. *(c)* Şematik sembol.

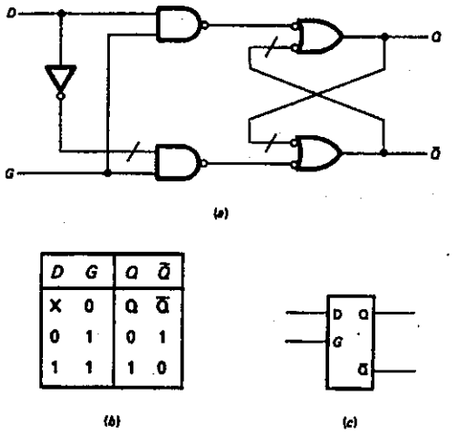
D MANDALI

Basit *RS* devresinde hala var olan bir problem de $S = 1$ ve $R = 1$ olduğunda belirsiz durumun oluşmasıdır. Bir çok uygulamada, Şekil 4-5 a da gösterildiği gibi, *S* ve *R* girişleri arasına bir ters çevirici bağlayarak problemi çözen bir devreyi kullanabilirsiniz. Elde edilen devreye *D* mandalı denir ve 7475 dörtlü *D* mandalı veya CMOS 4042 hazır bulunan entegrelerdir. Şekil 4-5 *b* deki doğruluk tablosundan veya bu devrenin analizinden görebileceğiniz gibi, yetki girişi *(G)* yüksek olduğu sürece, *Q* çıkışı *D* girişini izler. Yetki alçağa gitmeden hemen önceki *D* girişinde bulunan mantık durumu çıkışa kilitlenecektir. Mandallarda *yetki (enable)* girişine bazan *yükleme (strobe)* veya *kapılama (gate)* da denilmektedir.

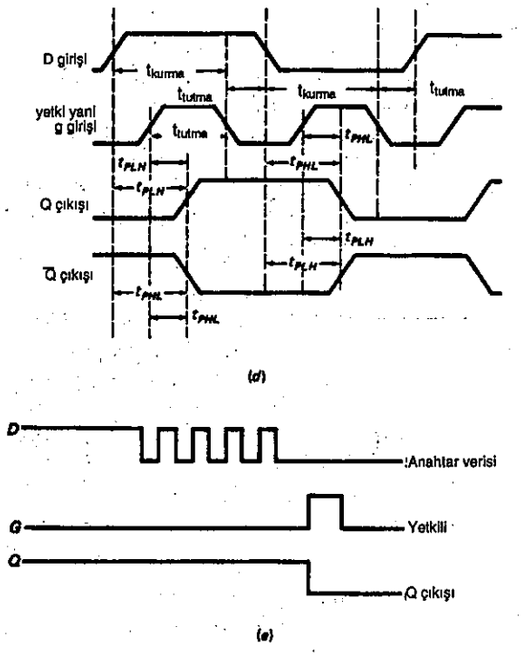
Şekil 4-5 *c*, bir *D* mandalın şematik sembolünü ve Şekil 4-4 *d* birçok önemli zamanlama parametrelerini vermektedir. Bunlardan üçü şunlardır: $t_{PW,MIN}$, yetki darbesinin minimum genişliği; t_{SETUP}, yetki düşmeden önce verilerin geçerli olarak kalması gereken minimum süre; ve t_{HOLD}, yetki alçağa düştükten sonra verinin değiştirilmeden tutulması gereken minimum süre olarak tanımlanır. Şekil 4-5 d de gösterilen örnek için, bu zamanlar yeterince uzun değilse, *Q* çıkışının *D* girişindeki alçağı yakalayacağından emin olamazsınız. Bir 7475 için bu değerler şöyledir:

$t_{PW,MIN} = 20$ ns $\quad t_{SETUP} = 20$ ns $\quad t_{HOLD} = 5$ ns

Şekil 4-5*d* ayrıca T_{PLH} ve T_{PHL} yi de göstermektedir. Bu sürelerin, girişteki değişmenin %50 noktası ile buna karşılık gelen çıkıştaki değişmenin % 50 noktası arasında ölçüldüğüne dikkat edin. Bir 7475 için T_{PLH} maksimum 80 ns ve T_{PHL} de maksimum 50 ns dir.



Şekil 4.5 D Mandalı. *(a)* İç şeması. *(b)* Doğruluk tablosu. *(c)* Şematik sembol. (Devam ediyor)



Şekil 4.5 (Devamı) *D* Mandalı. *(d)* SETUP (kurma) ve HOLD (tutma) zamanlarını gösteren zamanlama diyagramı. *(e)* Anahtar kontaklarındaki zıplamanın giderilmesi için zamanlama.

D mandalını nasıl kullanabileceğinize kısa bir örnek olarak, Şekil 4-5 e, bir anahtarın zıplayan çıkışını, *D* mandalının veri girişi olarak göstermektedir. Zıplamalar bitinceye kadar mandalı yetkilendirmezseniz, görüldüğü gibi, zıplamaların hiçbiri *Q* çıkışında görülmeyecektir. Anahtar sinyallerindeki zıplamayı gidermenin diğer bir yolu da budur. Yetki darbesi, tek kararlı olarak kullanılan iki *555* zamanlayıcı ile anahtar geçişinden elde edilebilir (Devreyi gözden geçirmek üzere Şekil 2-35 e bakın). Bir negatif anahtar geçişi ilk *555* i 10-15 ms lik bir çıkış için tetikler. Bu çıkışın düşen kenarı, ikinci *555* zamanlayıcıyı istenilen genişlikte bir yetki darbesi üretmek üzere tetikler. Mandallarda yetkilendirmek (enabling) yerine yükleme (strobing) terimi sıkça kullanılmaktadır.

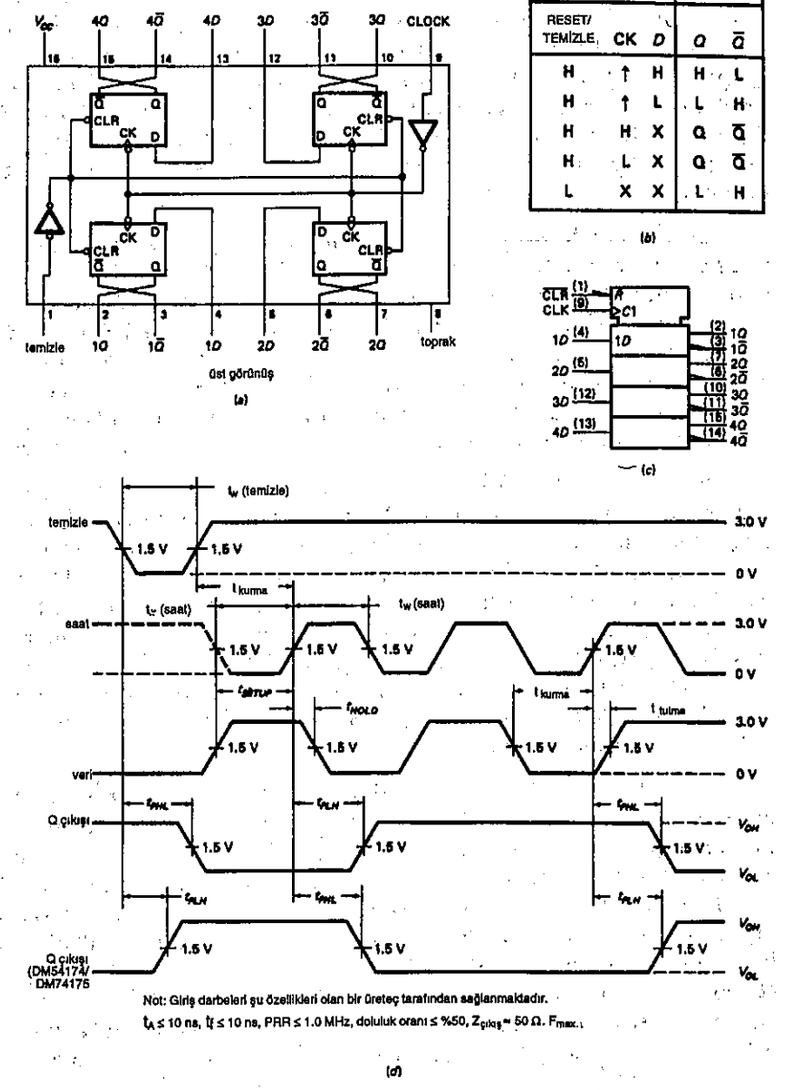
D FLİP - FLOP

D mandalıyla sık karıştırılan bir aygıt *D* flip-flopudur. Önceki kısımdan hatırladığınıza göre, bir *D* mandalında yetki girişi yüksek olduğu sürece *Q* çıkışı *D* girişini izlemekteydi. Halbuki bir *D* flip-flop, sadece saat girişinin pozitif geçişi sırasında *D* girişinde bulunan veriyi *Q* çıkışına gönderir. Bu, Şekil 4-6 *b* de gösterilmiştir. Bu şekil 74175 dörtlü kenar-tetiklemeli D flip-flopun doğruluk tablosudur. Tablonun saat sütunundaki iki ok, sadece saatin pozitif geçişinde *Q* çıkışının *D* ye eşit olmak üzere değiştiğini göstermektedir.

Doğruluk tablosundaki üçüncü satır, eğer saat yüksek durumda ise D sütunundaki *X* işareti ile gösterildiği üzere *D* girişi bir "farketmez"dir. Bunun anlamı *D* deki bir değişikliğin *Q* veya $\overline{Q}$ çıkışı üzerinde hiçbir etkisi olmayacağıdır. Tablodaki son satır, RESET/CLEAR girişi alçak yapılırsa saat ve *D* girişlerinin durumuna bakmaksızın, *Q* çıkışının reset durumuna geçeceğini söylemektedir. Bazı flip-flopların doğrudan RESET girişlerinin yanı sıra doğrudan PRESET girişleri de vardır. Bu girişlere *asenkron* denir, çünkü bun-lar saatin durumuna bakmadan çıkışları doğrudan set veya reset ederler.

Şekil 4-6 a daki 74175 'in bağlantı şemasında, entegrenin içindeki dört aygıtın herbirinin saat girişinde küçük bir üçgen ve küçük bir yuvarlak bulunduğuna dikkat edin. Üçgen, gerçek flip-flopun giriş saat sinyalinin bir kenarı ile çalıştığını gösterir. Yuvarlak ise flip-flopun saat sinyalinin negatif (yüksekten-alçağa) kenarı ile çalıştığını gösterir. Dört saat girişinin hepsi de bir ters çevirici ile sürülmekte olduklarından, gerçekte dışarıdan uygulanan saat sinyalinin pozitif (alçaktan - yükseğe) kenarı flip-flopu çalıştıracaktır. Saat hattı ters çeviricisinin çıkışındaki yuvarlak ile flip-flopların saat girişlerindeki yuvarlakların kısaldığını düşünürseniz, saatin hangi kenarının aktif olduğunu doğrudan görebilirsiniz.

Şekil 4-6 c, 74175 için bağımlılık gösterilişi mantık sembolünü göstermektedir. Tepedeki ezilmiş kutu, ortak kontrol bloğunun sembolüdür. Bu bloğa giren kontrol sinyalleri, tersi gösterilmedikçe, kontrol bloğunun altındaki bütün elemanları etkiler. Örneğin *CLR* girişi aktif yapılırsa bütün flip-floplar reset olur. *C* harfi, kontrol bağımlılığını gösterir. *C* den sonraki sayı ile aynı tanıtma sayısına sahip girişler, kontrol girişi ile yetkilendirilecekleridir. Bu durumda *C* kontrol bloğunda olduğuna göre, bütün *D* girişlerini etkiler. Sadece bir *D* girişine 1 *D* isminin verilmesi yeterlidir. CLK girişindeki üçgen tarafından belirtildiğine göre, *CLK* girişi alçaktan yükseğe giderken (*CLK* ın pozitif kenarı) *D* girişlerindeki veriler çıkışlara transfer edilecektir.



Şekil 4.6 D flip-flop. *(a)* 74175 dörtlü *D* flip-flopun bağlantı şeması. *(b)* Pozitif-kenar tetiklemesi gösteren doğruluk tablosu. *(c)* Bağımlılık gösterilişi mantık sembolü. *(d)* Zamanlama diyagramı.

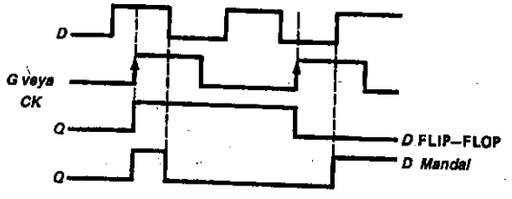
Bir *D* flip - flop için *SETUP (kurma)* ve *HOLD (tutma)* süreleri, Şekil 4-6 d de gösterildiği gibi saatin pozitif kenarından itibaren ölçülür. Size bir fikir vermek üzere, bir veri kitabı tipik değerler olarak $t_{SETUP,\ MIN}$ = 20 ns, $t_{HOLD,\ MIN}$ = 0 ns ve $t_{CW,\ MIN}$ ($t_{pw,\ MIN}$) = 20 ns vermektedir. *CLR* darbesinin minimum genişliği 20 ns dir ve *CLR* darbesi, saat darbesinin pozitif geçişinden en az 30 ns önce yükseğe çıkmalıdır ($t_{SETUP,\ CLEAR}$). Tabii ki bu değerleri ezberlemenize gerek yoktur; fakat bir flop-flop devresinde zamanlama problemleri ile ilgili arıza ararken veri kitabına başvurduğunuzda bu parametreleri anlamak zorundasınız. Sınırda bir zamanlama tasarımı olan bir flip-flop veya mandal, bazan doğru veriyi veren, bazan vermeyen, insanı sinir edici bir çalışma gösterir. Bir çift -iz - tetiklemeli osiloskop genellikle zamanlama problemlerini açığa çıkaracaktır.

4013 bir CMOS dörtlü *D* flip-flop örneğidir.

D MANDALI VE D FLİP-FLOPUN ÇIKIŞ DALGAŞEKİLLERİ

Şekil 4-7, aynı *D* girişi ve aynı saat dalgaşekilleri için, *D* mandalı ve *D* flip-flop çıkış dalgaşekillerinin farklılıklarını göstermektedir. Her saat darbesinin yükselen kenarında, *D* flip-flopu, o anda *D* girişinde bulunan durumu göstermek üzere *Q* çıkışını yeniler. Daha sonra da bu durum, saatin bir sonraki pozitif kenarıyla yenileninceye kadar *Q* çıkışında tutulur. Esas olarak, bu aygıt saatin yükselen kenarında D nin bir fotoğrafını çekmekte ve sonra da onu *Q* çıkışında göstermektedir diye düşünülebilir.

D mandalının saat veya yetki girişi yüksek olduğunda ise, *Q* ve *D* birbirine bir telle bağlıymış gibi, *Q* çıkışı *D* girişini izler. Saat alçalınca, *Q* da kalan durum, saatin düşmesinden hemen önceki *D* girişinin durumudur.



Şekil 4.7 Aynı saat ve *D* giriş sinyalleri için, *D* mandalı ile *D* flip-flopun çıkış dalgaşekillerinin karşılaştırılması.

JK FLİP-FLOP

Flip-flopların birçok kullanılış şekillerini görmeden önce çok önemli bir tipi üzerinde daha durmak isteriz. Bu, *RS* devresinde bulduğumuz yasaklı giriş birleşimi problemini yok eden ve hâlâ *J* ve *K* gibi iki girişi olan *JK* flip-floptur. Bunu gerçekleştirmek için kulanılan değişik iç devre yapılarıyla bu kitabı veya kafanızı doldurmaya hiç gerek yoktur. Esas anlaşılması gereken noktalar *JK* flip-flopların doğruluk tablosunu ve sahip olabileceği üç türlü saatli çalışmasını anlamaktır.

POZİTİF - KENAR - TETİKLEMELİ JK FLİP-FLOP

Şekil 4-8 a, bir CMOS 4027 *JK* flip-flopun doğruluk tablosunu göstermektedir. Oklar, çıkışın J ve *K* girişlerince belirlenen duruma değişmesinin, saatin pozitif geçişinde olduğunu belirtmektedir. Her iki girişin yüksek olması durumunda, basit *RS* de belirsiz durum varken, *JK* flip-flopta çıkış saatin her pozitif geçişi ile durum değiştirir (toggle eder). Diğer giriş birleşimleri basit RS mandalı ile aynı sonuçları verirler. *J* ve *K* nın her ikisi de alçaksa, saat darbesi geldiğinde çıkış değişmeyecektir. Eğer *J* yüksek ve *K* alçaksa, saat darbesinden sonra *Q* yüksek olacaktır ve eğer *J* alçak ve *K* yüksekse saat darbesinden sonra *Q* alçak olacaktır. Doğrudan set ve reset girişleri, saat, *J* ve *K* girişlerinden bağımsızdır. Bunları kullanarak *Q* çıkışı istenildiği anda yükseğe set edilebilir veya alçağa reset edilebilir. Şekil 4-8a dan görebildiğiniz gibi doğrudan set ve reset girişleri aktif yüksektir. Her ikisi de aynı anda yüksek olamaz. Bu tip flip-floplarda kurma ve tutma zamanları saatin pozitif kenarından ölçülür.

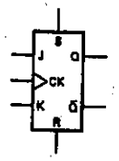
NEGATİF - KENAR - TETİKLEMELİ JK FLİP-FLOP

Şekil 4-8 d, bir 74LS76 ikili *JK* flip-flopun doğruluk tablosunu göstermektedir. Oklar, bu aygıtın *J* ve *K* girişlerine saatin negatif kenarında tepki gösterdiğini belirtmektedir. Ancak *J* ve *K* girişlerinin etkileri CMOS 4027 dekilerle aynıdır. Örneğin, *J* ve *K* nın her ikisi de yüksek ise her saat darbesiyle çıkış durum değiştirir. 74LS76 nın doğrudan set ve reset girişlerinin aktif alçak olduğuna dikkat edin. Her ikisi birden aynı anda alçak yapılmamalıdır. Kurma ve tutma zamanları saatin negatif kenarından ölçülür. Negatif-kenar-tetiklemeli *JK* flip-flopların diğer örnekleri 74103, 74113 ve 74114 tür.

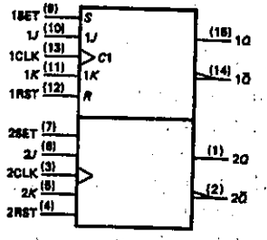
4027

GİRİŞLER					ÇIKIŞLAR	
S	R	CK	J	K	Q	$\bar{Q}$
L	H	X	X	X	0	1
H	L	X	X	X	1	0
H	H	X	X	X	H*	H*
L	L	↑	L	L	Q0	$\bar{Q}0$
L	L	↑	H	L	H	L
L	L	↑	L	H	L	H
L	L	↑	H	H	durum değiştirir	

(a)



(b)

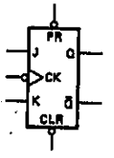


(c)

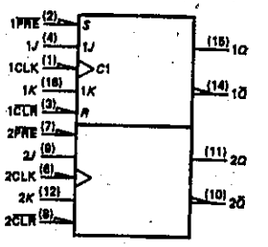
74LS76

GİRİŞLER					ÇIKIŞLAR	
PR	CLR	CK	J	K	Q	$\bar{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H*	H*
H	H	↓	L	L	Q0	$\bar{Q}0$
H	H	↓	H	L	H	L
H	H	↓	L	H	L	H
H	H	↓	H	H	durum değiştirir	
H	H	H	X	X	Q0	$\bar{Q}0$

(d)



(e)

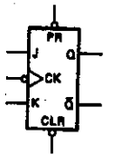


(f)

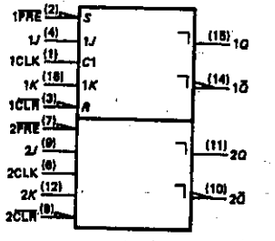
7476, H76

GİRİŞLER					ÇIKIŞLAR	
PR	CLR	CK	J	K	Q	$\bar{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H*	H*
H	H	⎍	L	L	Q0	$\bar{Q}0$
H	H	⎍	H	L	H	L
H	H	⎍	L	H	L	H
H	H	⎍	H	H	durum değiştirir	

(g)



(h)



(i)

NOTLAR ⎍ = yüksek - seviye darbe; saat yüksekken veri girişleri sabit tutulmalıdır; veri çıkışa darbelerin düşen kenarında transfer edilir.
Q0 = Belirtilen giriş koşulları oluşturulmadan önce Q nun seviyesi.
DURUM DEĞİŞTİR : Saatin (darbenin) her aktif geçişinde, her çıkış daha önceki seviyesinin tümleyenine değişir.
* Bu durum kararlı değildir; yani doğrudan set (direct set yani preset) ve doğrudan reset (direct clear veya clear) girişleri aktif olmayan seviyelerine (yüksek) döndüklerinde bu durum kalıcılık göstermez.

Şekil 4-8 JK flip-floplar. *(a)* Pozitif - kenar - tetiklemeli 4027 'nin doğruluk tablosu. *(b)* 4027 nin şematik sembolü. *(c)* 4027 nin bağımlılık gösterilişi sembolü. *(d)* Negatif - kenar - tetiklemeli 74LS76 nın doğruluk tablosu. *(e)* 74LS76 nın şematik sembolü. *(f)* 74LS76 nın bağımlılık gösterilişi sembolü. *(g)* Seviye tetiklemeli 7476 nın doğruluk tablosu *(h)* 7476 nın şematik sembolü. *(i)* 7476 nın bağımlılık gösterilişi sembolü.

ANA-UYDU (MASTER-SLAVE) JK FLİP-FLOP

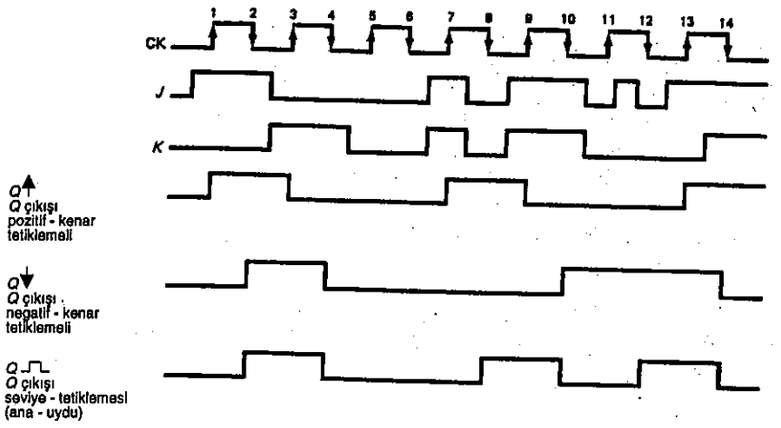
Diğer bir tip *JK* zamanlaması Şekil 4-8 g de bir 7476 ikili *JK* flip-flop için gösterilmiştir. Birçok *JK* aygıtlarında olduğu gibi, 7476 daki her flip-flopun bir ana (master) ve bir uydu (slave) olmak üzere iki kısmı vardır. 7476 nın devresi, saat yüksek seviyeye çıktığında *J* ve *K* verileri, ana yani giriş kısmına yüklenecek şekilde tasarlanmışır. Saat tekrar alçağa düşünce *J* ve *K* verileri, uydu yani çıkış kısmına aktarılır ve çıkışta görülür. Bu seviye-saatli tiplerin bir problemi, saat yüksekken *J* ve *K* girişlerinin değişmemesi gerekliliği, aksi halde çıkışın ne olacağının bilinemiyeceğidir. Bu problemden kaçınmak için saat darbeleri mümkün olduğunca dar tutulmalıdır.

Bağımlılık gösterilişinin negatif-kenar-tetiklemeli 74LS76 (Şekil 4-8 *f*) ile seviye-tetiklemeli 7476 (Şekil 4-8 *i*) arasındaki farkı nasıl gösterdiğine dikkat edin.

JK FLİP-FLOP ZAMANLAMA DİYAGRAMLARI

JK flip-flopda, verilen *J* ve *K* giriş dalgaşekilleri için *Q* çıkış dalgaşekillerinin ne olacağını öngörmek önemlidir. Şekil 4-9, aynı *J* ve *K* giriş dalgaşekilleri için her üç tip *JK* flip-flop tarafından üretilen çıkış dalgaşekillerini göstermektedir. İlk bakışta bu çıkış dalgaşekillerini öngörmenin zor olduğu zannedilebilir, fakat kısım kısım çizilirlerse, oldukça kolaydır.

Şekil 4-9 daki pozitif-kenar-tetiklemeli *JK* flip-flop çıkış dalgaşekli için, başta Q nun alçak olduğunu varsayalım. İlk pozitif saat geçişi sırasında *J* girişi yüksek, *K* girişi alçaktır. *JK* doğruluk tablosunda belirtildiği gibi, bu Q çıkışını yükseğe götürür. Hiçbir doğrudan set veya reset (PRESET veya CLEAR) girişi olmadığını kabul edersek, saat girişinin 3 le gösterilen bir sonraki pozitif geçişine kadar *Q* çıkışı bir daha değişemez. O halde, saat kenarı 1 den saat kenarı 3 e kadar dalgaşekli yüksek olarak çizilebilir. Saat kenarı 3 te, *J* alçak ve *K* yüksektir. Doğruluk tablosunda gösterildiğine göre, bu Q çıkışının alçağa gitmesine sebep olur ve çıkış en az saat kenarı 5 e kadar bu seviyede kalır. Saat kenarı 5 te, hem *J* hem de *K* alçaktır; o halde çıkış değişmez. Saat kenarı 7 de *J* ve *K* nin her ikisi de yüksektir, çıkış durum değiştirecektir ve bu durumda alçaktan yükseğe çıkacaktır. Saat kenarı 9 da yine *J* ve *K* yüksektir, Q yine durum değiştirir ve bu sefer yüksekten alçağa düşer. Saat kenarı 11 de, *J* ve *K* nın her ikisi de alçaktır, böylece *Q* aynı kalır. Son olarak, geçiş 13 te *J* girişi yüksek ve *K* girişi alçaktır, böylece *Q* çıkışı yükselir. Aygıt, saatin her pozitif geçişinde *J* ve *K* girişlerini örnekliyormuş ve *Q* çıkışın *JK* doğruluk tablosu kurallarına göre yeniliyormuş gibi düşünülebilir. Bu *Q* durumu saatin bir sonraki pozitif geçişiyle tekrar yenileninceye kadar aynı kalır. Dalgaşekli bir saat darbesi uzunluğundaki parçalar halinde çizilir.



Şekil 4-9 **Verilen saat, *J* ve *K* giriş dalgaşekilleri için üç tip *JK* flip-flopun ürettiği çıkış dalgaşekilleri.**

Şekil 4-9 daki negatif-kenar-tetiklemeli flip-flopa ait dalgaşekli aynı şekilde fakat *J* ve *K* girişlerinin örneklemeleri giriş saatinin düşen kenarında olmak üzere çizilmiştir.

Bir *JK* ana-uydu flip-flopunun dalgaşeklini çizmek için, sadece doğruluk tablosunu değil ayrıca ana-uydu işleminin nasıl olduğunu da hatırlamalısınız. Saat yüksekken, *J* ve *K* girişlerindeki veri ana kısmının çıkışını yeniler. Saat alçağa düşünce, ana kısmın çıkışındaki durum uydu kısmının çıkışına taşınır. Böylece, *Q* çıkışındaki değişiklikler saatin düşen kenarıyla çakışır. Şekil 4-9 un alt kısmındaki dalgaşekli bunun nasıl olduğunu göstermektedir. Saat kenarı 1 'de, *J* ve *K* nın her ikisi de alçaktır, böylece ana set olur. Saat kenarı 2 de, set durumu çıkışa taşınır. Saat kenarı 3 te, *J* alçak ve *K* yüksektir, böylece ana reset olur. Saat kenar 4 te, reset durumu çıkışa aktarılır. Kenar 5 ve 6 da, *J* ve *K* nın her ikisi de alçaktır, böylece çıkış değişmez. Ancak saat kenarı 7 de, *J* ve *K* nın her ikisi de yüksektir, böylece ana yükseğe değişir. Saat yüksekken *J* ve *K* nın ikisi de alçağa gitmekte ve anayı yükseğe kilitlemektedir. Saat kenarı 8 de, *J* ve *K* nın her ikisi de yüksektir, ana alçağa değişir. Kenar 10 da, çıkış alçağa değişir. Saat kenarı 11 de, *J* ve *K* nın her ikisi de alçaktır, çıkış alçakta kalır. Ancak, saat kenarları 11 ve 12 arasında *K* alçaktır ve *J* yükselir. Bu birleşim aygıtın ana kısmını set eder. Kenar 12 de saat düşünce, bu durum gösterildiği gibi çıkışa taşınır. Saatin alçalmasından

önce *J* nin tekrar alçağa düşmesi önemli değildir; çünkü *J* ve *K* nın ikisi de alçak iken çıkışta bir değişiklik olmaz. Saat yüksekken J veya K daki değişmeyle olacak beklenmedik çıkışlara "1's catching" (1 i yakalama) denir. Şekil 4-9 da kenar 13 ve 14 arasında, *J* yüksektir ve *K* yükseğe çıkmaktadır. Bu, saatin negatif kenarında Q çıkışının gösterildiği gibi durum değiştirmesine sebep olur.

FLİP-FLOPLARININ PROBLEMLERİ

Flip-flop ve mandalların çok karşılaşılan üç tür problemleri vardır:

1. Q çıkışı, saat geldiğinde *D* girişi veya *J* ve *K* girişlerince öngörülen duruma değişmemektedir. Genellikle bu, entegre bozuksa veya çıkış V_{cc} ye, toprağa veya baskılı kart üzerindeki başka bir yola kısa devre edilmişse olur.

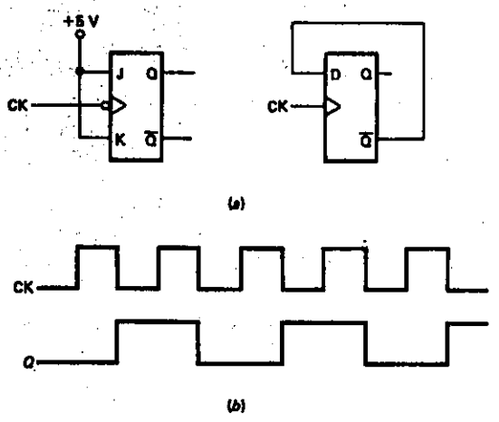
2. Çıkış gelişigüzel zamanlarda set veya reset etmektedir. Bu genellikle entegrenin set veya reset girişi açık bırakılmışsa olur. Açık bırakıldıklarında, bu girişler diğer hatlardan gürültü (darbe) kapabilirler. Kullanılmayan set veya reset girişleri, toprak veya V_{cc} den hangisi onları aktif olmayan durumda tutacaksa oraya bağlanmalıdır.

3. Çıkış bazan doğrudur, bazan hatalıdır. Bu, entegrenin bozuk olmasından veya aygıtın kurma ve/veya tutma zamanlarına uyulmadığından olabilir. Eğer bir giriş veri değişimi saat kenarına çok yakın olursa, çıkış istediğiniz seviyeye değişebilir de değişmeyebilir de. Bu parametreler sıcaklıkla ve entegreden entegreye değiştiklerinden, sınırda olan zamanlama problemlerini izlemek zor olabilir. Örneğin, bir cihaz kapağı açıkken iyi çalışmaktadır, fakat kapağını kapadıktan 5 dakika sonra göstergesinde yanlış veri göstermektedir. Bu takdirde yapmanız gereken, zamanlamanın sınırda olup olmadığını anlamak için şüpheli entegrenin giriş ve saatine bir osiloskopla bakmaktır. Daha sonra, entegreye bir ısı kaynağını yaklaştırarak bunun bir arızaya sebep olup olmadığına bakın. Eğer hatalı çalışma olursa, yonganın mı arızalı (kurma ve tutma zamanları veri sayfasındaki minimum değerlerden daha uzun), yoksa devre tasarımının mı sınırda olduğunu osiloskoptaki görüntüden belirleyebilirsiniz. Bu durumlarda sadece entegreleri değiştirmek iyi bir yol değildir; çünkü cihazı gideceği yere varıncaya kadar iyi çalıştıran bir entegre bulabilirsiniz.

Diğer problemler çok dar saat darbeleri veya reset darbeleridir. Ayrıca bir reset veya set darbesi yeterince hızlı değilse, flip-flopun çıkışı salınarak birçok darbe üretebilir.

ASENKRON VE DALGACIK SAYICILARI

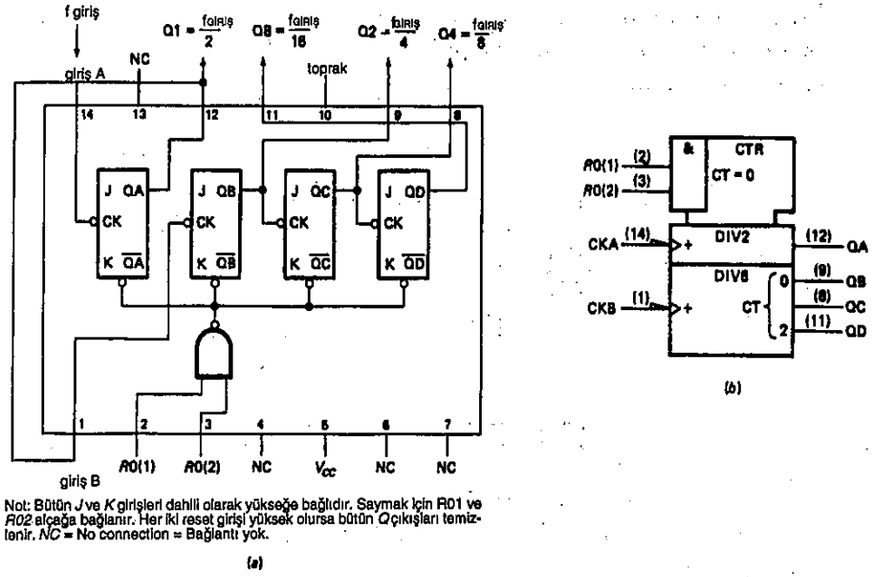
J ve *K* girişleri yükseğe bağlı bir *JK* flip-flopu veya $\overline{Q}$ çıkışı *D* girişine bağlı bir *D* flip-flopu; *T* veya toggle flip-flop denilen flip-flopu oluşturur (Şekil 4-10a). Her iki yapıda da, her saat darbesiyle çıkış durumunun değişecek olması (toggle etmesi) nedeniyle bu isim verilmiştir. Bir çıkış çevrimi oluşturmak için iki saat darbesi gerektiğinden (Şekil 4-10 b) çıkış frekansı giriş saatinin frekansının yarısıdır. İki *T* flip-flop seri bağlanarak giriş frekansının dörtte biri frekansta bir çıkış da elde edilebilir. Üçüncü bir *T* flip-flopu seri olarak eklemek $f_{GİRİŞ}$ / 8 çıkışını ve bir dördüncü flip-flop $f_{GİRİŞ}$ /16 çıkışını verir.



Şekil 4-10 ***JK* ve *D* flip-floplardan yapılmış T flip-floplar. *(a)* Devreler. *(b)* Negatif-kenar-tetiklemeli *JK* flip-floptan yapılmış *T* flip-flopun dalgaşekilleri.**

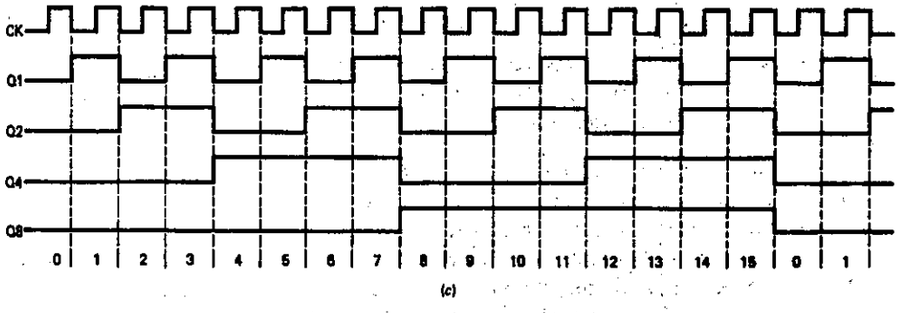
Şekil 4-11 a, aslında 4 tane seri bağlı flip-flop olan 7493 dört-kademeli ikilik bölücü yani 4-bit ikilik dalgacık sayıcıyı göstermektedir. Bölücü olarak, giriş frekansının bir kesri olan bir frekansı elde etmek için çıkışlardan biri veya bir çoğu kullanılır. Buna ayrıca sayıcı da denilmektedir, çünkü Şekil 4-11 c deki ideal dalgaşekillerinden de görüldüğü gibi, dört çıkış birlikte 0 ile 15 arasındaki bir ikilik sayıyı gösterir. Her giriş saat darbesinin negatif kenarı sayıcıyı bir sonraki ikilik sayıya artırır. Sayıcı 15 e, yani ikilik 1111 e, ulaşınca, saatin negatif kenarıyla birlikte çıkışların hepsi 0 a döner. Bundan sonra sayım yeniden başlar.

7493 ün Şekil 4-11 b deki bağımlılık gösterilişi mantık sembolü aygıtın, 2-ile-böl kısmı ve 8-ile-böl kısmı olmak üzere iki kısımdan oluştuğunu göstermektedir. Saat girişlerinin yanındaki artı işaretleri, her saat darbesinin düşen kenarında, sayıcı kısımlarının 1 arttığını (yukarı saydığını) belirtmektedir. Eğer *R0*(1) ve *R0*(2) aktif *(H)* ise çıkışlar 0 yapılacaktır (sayım 0 olacaktır).



Not: Bütün *J* ve *K* girişleri dahili olarak yükseğe bağlıdır. Saymak için R01 ve *R02* alçağa bağlanır. Her iki reset girişi yüksek olursa bütün *Q* çıkışları temizlenir. *NC* = No connection = Bağlantı yok.

(a)



Şekil 4-11 ***(a)*. Dört-kademeli ikilik bölücü veya sayıcı olarak bağlanmış 7493 ün şeması. *(b)* 7493 için bağımlılık gösterilişi *(c)* Saat ve çıkış dalgaşekilleri.**

Sayıcılar, daha fazla sayabilmek veya daha fazla bit elde etmek için seri bağlanabilir. Örneğin CMOS 4060, tek bir IC paketinde 14 ikilik bölücü kademesine sahiptir.

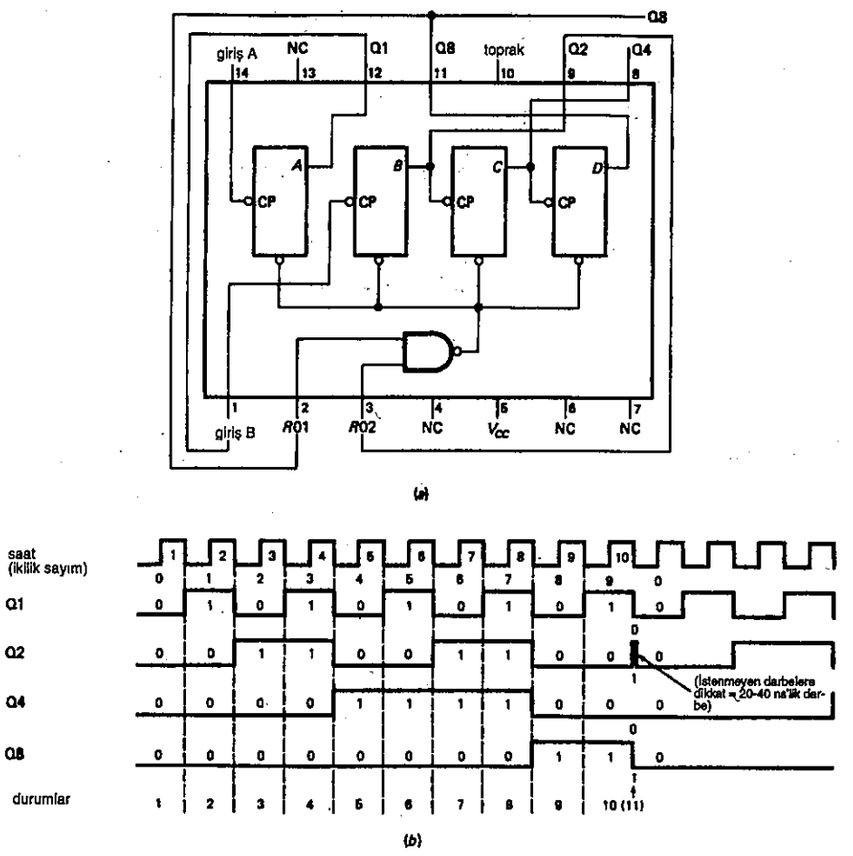
Dalgacık terimi, bir saat darbesinin etkisinin flip-floplar zinciri üzerinde art arda yayılarak gittiğini göstermektedir. Şekil 4-11 a daki devre için, son flip-flopun çıkışı, giriş saatinin negatif kenarından dört flip-flop yayılma gecikmesi sonra değişecektir. Bunun anlamı bu sayıcının asenkron olduğudur, çünkü çıkışlar saatle doğrudan senkronize değildir. Daha sonraki bir kısımda, bazı devrelerde bu gecikmenin sebep olduğu problemler üzerinde duracağız.

DALGACIK SAYICILARIN MOD-N BÖLÜCÜLER OLARAK KULLANILMASI

Flip-floplar, N flip-flop sayısı olmak üzere, çıkış frekansı $f_{GIRIŞ}/2^N$ olan sayıcılar oluşturmak amacıyla seri bağlanabilir. Örneğin, CD4044 CMOS 14-kademeli ikilik bölücünün son kademesindeki çıkış frekansı $f_{GIRIŞ}/2^{14}$ yani $f_{GIRIŞ}/16384$ olacaktır.

Bir sayıcının giriş frekansının çıkış frekansına oranına mod denir. Tek bir flip-flop bir mod-2 bölücü veya sayıcıdır, çünkü giriş frekansı çıkış frekansının iki katıdır. Şekil 4-11 a daki 7493 sayıcının maksimum modu, giriş frekansı en alçak çıkış frekansının 16 katı olduğundan, 16 dır. Değişikliğe uğratılmamış dalgacık sayıcılarında, mod, 2, 4, 8, 16, 32 v.b. gibi her zaman için 2 nin bir tam kuvveti olacaktır. Sayıcıların frekans bölücü olarak kullanımlarının çoğunda (örneğin, sayısal saatler) 6 veya 10 gibi başka bir mod gerekecektir. Verilen bir sayıcının maksimum mod değerinden daha küçük bir mod oluşturmak için, sayıcı maksimum sayıma ulaşmadan, sayıcıyı sıfıra reset edecek sayısal geribesleme kullanırsınız.

Şekil 4-12 a bir 7493 ün nasıl bir mod-10 sayıcıya veya frekans bölücüsüne dönüştürülebileceğini göstermektedir. 7493 bütün çıkışlarda 0 larla başlayıp, çıkış 1010 ikilik durumuna ulaşıncaya kadar Şekil 4-11 c dekine benzer şekilde ikilik sıraya göre sayar. *Q8* çıkışındaki yüksek, *Q2* çıkışındaki yüksek ile içerdeki VE DEĞİL kapısında VE DEĞİL lenerek bütün çıkışları derhal 0 a reset eder. Bundan sonra sayım yeniden başlar. Şekil 4-12 *b* saat dalgaşeklini ve oluşan çıkış dalgaşekillerini göstermektedir. Dalgaşekillerinden görebileceğiniz gibi, bir *Q8* çıkış darbesi üretmek için 10 saat darbesi gerekmektedir. Saati giriş olarak ve *Q8* i çıkış olarak kullanmak bu devreyi bir mod-10 frekans bölücü yapar. Saati giriş olarak ve çıkış için *Q* çıkışlarının dördünü birlikte kullanmak bir devreyi bir onluk, yani mod-10 sayıcı yapar. Çıkışlar bir arabanın kilometre saatindeki basamaklar gibi 0 dan 9 a kadar 10 durumu sayar. Bir onbirinci durumun var olduğuna fakat bunun sadece çıkışları sıfıra resetlemek için gerekli 30-40 ns sürdüğüne dikkat edin. Bir sayısal saatin göstergelerini sürmek gibi birçok uygulamada bu ilave durum gözle görülemeyecek kadar kısa olduğundan önemli değildir. İstenmeyen darbelerin problem çıkarabileceği uygulamalarda senkron sayıcılar kullanılır. Senkron sayıcılar bu bölümde daha sonra anlatılacaktır.

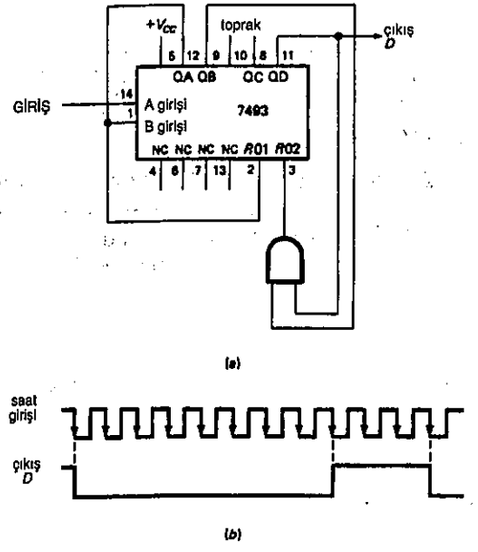


Şekil 4-12 ***(a)*** **Bir mod-10 sayıcıya dönüştürülmüş 7493 ü gösteren devre.** ***(b)*** **Saat ve çıkış dalgaşekilleri.**

Çoğunlukla bir onluk sayıcıyı gerçekten sentezlemek zorunda kalmayacaksınız; çünkü bu sayıcı TTL ailesinde 7490 olarak hazırda vardır. 7490 ın bir 5-ile-böl kısmı ve bir de T flip-floptan yapılmış 2-ile-böl kısmı vardır. Eğer giriş sinyali önce 2-ile-böl kısmından sonra 5-ile-böl kısmından geçirilirse çıkışlar BCD sırasında (0-9 arasındaki ikilik ile aynı) sayacaktır. Giriş sinyali önce 5-ile-böl kısmından ve sonra da flip-floptan geçirilirse, son çıkış, giriş frekansının onda biri frekansında bir simetrik karedalga olacaktır. Bir 12-ile-böl devresi TTL de 7492 dalgacık sayıcı ismiyle hazır olarak vardır.

Bir giriş frekansını 2 ile 16 arasında herhangi bir tamsayı ile bölmek için Şekil 4-12 a dakine benzer bir yapı bir 7493 ile kullanılabilir. Buradaki hile bir sonraki durumun kodunu çözdürmek ve bunu çıkışları 0 lara reset etmek için kullanmaktan ibarettir. Bu hile ayrık flip-floplardan yapılmış sayıcılarda bir **yarış** durumunun oluşmasından dolayı çalışmayabilir. Çıkışlardan birisi reset olur olmaz kodu çözülmüş çıkış durumu kaybolabilir. Eğer bu durum reset girişlerinde yeterince uzun bulunmazsa bazı flip-floplar reset olmayabilir. O halde bu bir ***yarıştır***. Kodu çözülmüş çıkış, reset darbesi kaybolmadan önce bütün flip-flopları reset edilebilecek midir? Entegre devre sayıcılar, yarış problemleri çoğunlukla yok edilmiş olarak tasarlanırlar.

Şekil 4-13, bir 7493 ü kullanan 11-ile-böl, yani mod-11, sayıcının devresini göstermektedir. Mod-11 devresinin, *QA*, *QB* ve *QD* deki 1 lerin algılanması gerektiğinden, bir harici kapı kullanmak zorunda olduğuna dikkat edin.



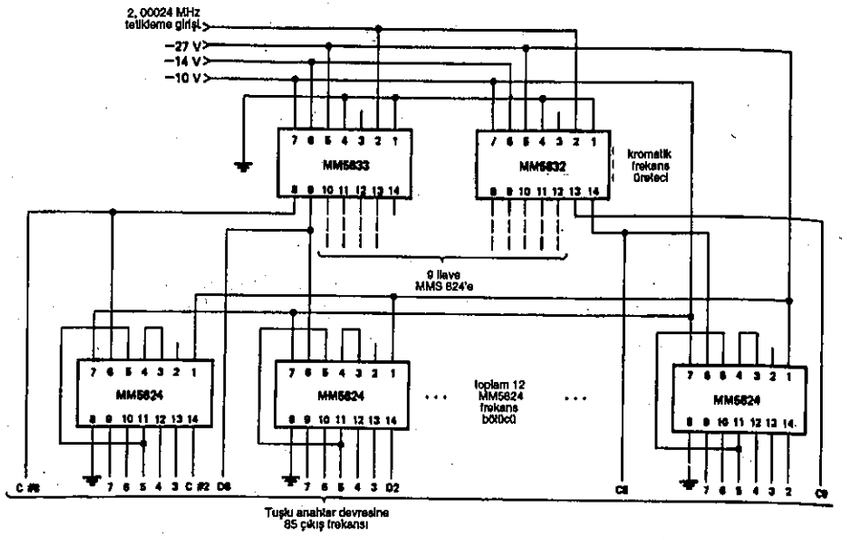
Şekil 4-13 ***(a)*** **Mod-11 sayıcı için 7493 devresi.**
(b) **Saat ve çıkış dalgaşekilleri.**

16 nın üzerindeki modlar için iki veya daha fazla sayıcıyı seri bağlayabilirsiniz. İstediğiniz moda eşit olan durumun kodunu daha önceki gibi çözerek bunu bütün çıkışları sıfırlamak için kullanabilirsiniz. Bir giriş frekansını 256 ya kadar herhangi bir sayıya bölmek için iki tane 7493 kullanılabilir.

DALGACIK SAYICILARIN BAZI UYGULAMALARI

ELEKTRONİK ORG Batı müziği, oktav başına eşit aralıklı 12 notalı bir kromatik sisteme dayanır. Oktav içindeki notaların birbirleri arasında kesin matematik ilişkiler vardır ve bir oktavdaki bir notanın frekansı bir üst oktavdaki aynı notanın frekansının tam yarısıdır. Örneğin, standart *"A"* 440 Hz dir ve bir alttaki oktavdaki *"A"* 220 Hz dir. Bunun bir altındaki oktavdaki *"A"* 110 Hz ve bir sonraki oktavdaki *"A"* da 55 Hz dir. Elektronik org gibi bir alet için, Şekil 4-11 a da gösterildiği gibi bütün gerekli frekanslar, tek bir referans frekansını bölerek üretilebilir. National'ın MM5833 ve MM5832 kromatik frekans üreteçleri, 2,00024 MHz lik bir ana saat tarafından sürülen 12 paralel bölücü zinciri ile üst oktavın 12 notasını üretir. Örneğin, bu dahili bölme zincirlerinden birisi ana frekans 2,00024 MHz i 284 e bölerek 7040 Hz lik yüksek *A* notasına çok yakın olan 7043 Hz verir. Şekil 4-14 *b*, MM5832 ve MM5833 ün bölücü modlarını ve çıkış frekanslarını göstermektedir. Karşılaştırma amacıyla eşit aralıklı ölçekteki standart frekanslar da verilmiştir. Daha alttaki oktavların frekansları, basit ikilik bölücü zincirlerle en üst oktavın frekanslarını bölerek türetilebilir. National MM5824 bir altı-kademeli ikilik bölücü zinciri içerir. Böylece, Şekil 2-14 a da gösterilen devre yedi tam oktav ve bir de fazladan D0 üretir. Bu 14 entegre, bir elektronik org yapmak için gerekli 85 frekansın tamamını verir. Devrede gösterilen tipik PMOS aygıtların standart olmayan gerilimlerine dikkat edin.

Çıkışların hepsi kare dalgadır ve yükseltilip bir hoparlöre verildiğinde biraz klarnet sesine benzer ses elde edilir. Diğer sesleri üretmek için, bu notalar harmonikleri ile karıştırılıp formant filitrelerinden geçirilir. Monoton bir tondan kaçınmak için 2,00024 MHz lik ana osilatöre hafif bir titreşim frekans modulasyonu uygulanır.



Tipik bir org ton üreteci

(a)

MM5832

NOT	BÖLEN	ÇIKIŞ FREKANSI	EŞİT ARALIKLI FREKANS	YÜZDE HATA
C8	478	4184.61	4186.01	−0.565
C9	239	8369.21	8372.02	−0.565
B8	253	7906.09	7902.13	0.842
A #8	268	7463.58	7458.62	1.119
A8	284	7043.10	7040.00	0.740
G #8	301	6645.32	6644.88	0.112
G8	319	6270.34	6271.93	−0.424

MM5833

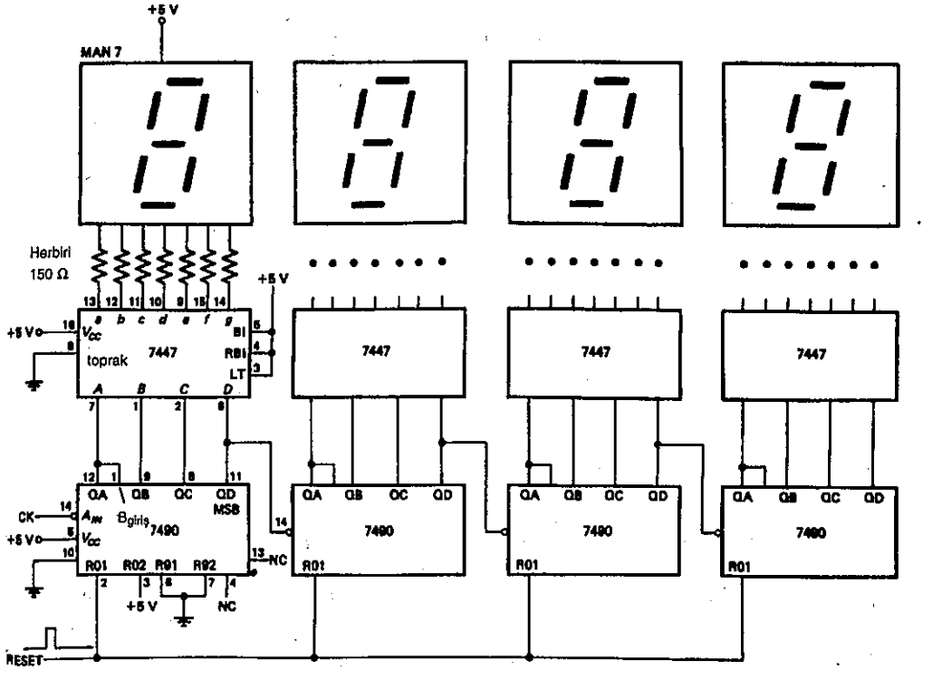
NOT	BÖLEN	ÇIKIŞ FREKANSI	EŞİT ARALIKLI FREKANS	YÜZDE HATA
F #8	338	5917.87	5919.91	−0.580
F8	358	5587.26	5587.65	−0.117
E8	379	5277.68	5274.04	1.180
D #8	402	4975.72	4978.03	−0.780
D8	426	4695.40	4698.64	−1.159
C #8	451	4435.12	4434.92	0.076

(b)

Şekil 4-14 *(a)* National MM5833 kromatik frekans üretecini ve MM5824 altı-kademeli ikilik bölücüleri kullanan elektronik org ton üretecinin şeması. *(b)* Eşit aralıklı frekansları ve kromatik üretecin frekanslarının karşılaştırılması.

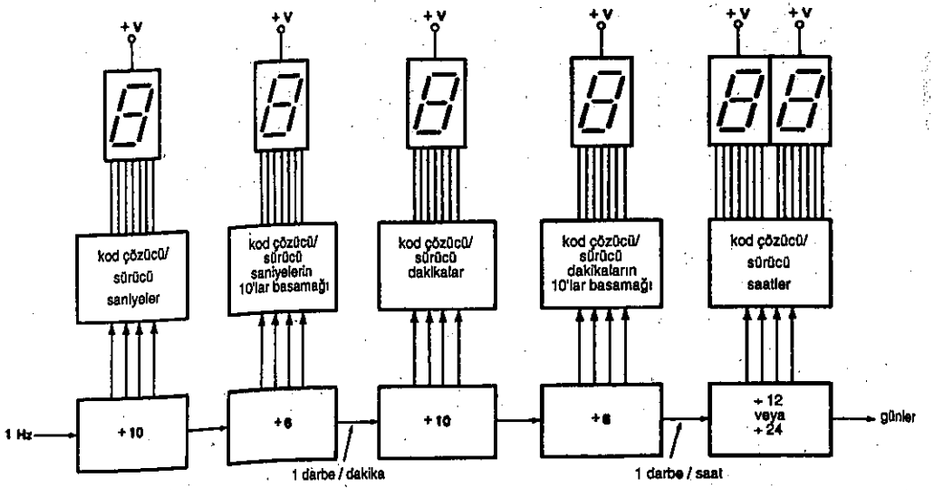
SAYMA BİRİMLERİ VE SAYISAL SAATLER

Şekil 4-15, onluk-sayma birimleri oluşturmak üzere 7490 onluk sayıcıların; 7447 BCD-den-yedi-parçalı göstergeye kod çözücüler ve ortak - anotlu yedi-parçalı LED göstergelerle nasıl bağlanabileceğini göstermektedir. Her gösterge, sayıcının BCD çıkışına karşılık gelen 0 ile 9 arasındaki onluk sayıyı gösterir. Sayıcı çıkışı, giren saat darbesi sayısını gösterecektir. En az değerlikli sayıcı 9 dan 0 a geçerken, QD çıkışının düşen kenarı bir sonraki sayıcıyı bir artırır. İstediğiniz kadar büyük bir sayıya sayabilmek için daha fazla sayıcılar eklenebilir. Bir taşıyıcı bant üzerinde geçen tenekeleri saymak veya bir futbol maçına giren insanları saymak için böyle bir devrenin girişinde darbe kaynağı olarak bir ışık kaynağı ve fototransistör kullanabilirsiniz. Not : Şekil 4-15 teki göstergede basamakların yeri sayıların normal yazılışındakilerin tersidir.



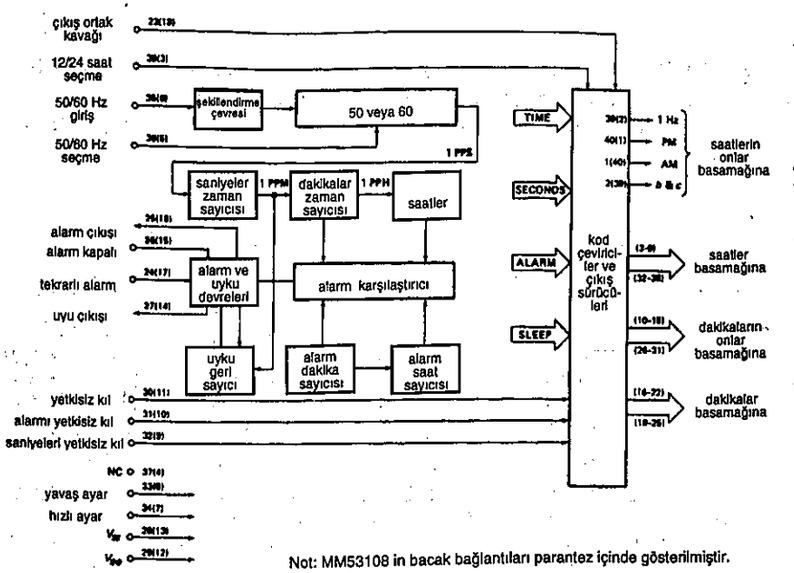
Şekil 4-15 74190, 7447 ve LED göstergelerden yapılmış onluk sayma birimleri.

Şekil 4-15 teki devrenin girişine, saniyede 1 darbe, yani frekansı 1 Hz olan bir saat uygularsanız, gösterge saniye cinsinden geçen zamanı gösterir. Bu sayısal göstergeli saattir, fakat bildiğiniz saat, dakika, saniye formatında değildir.



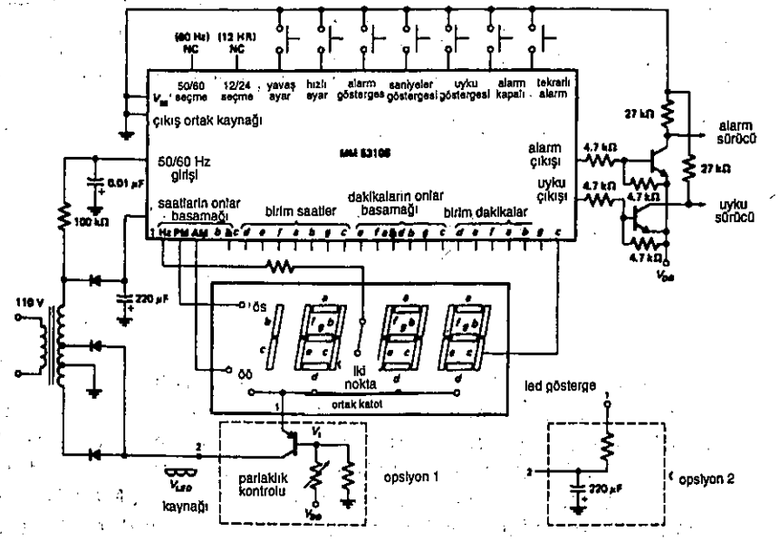
Şekil 4-16 Yedi-parçalı göstergeli bir sayısal saatin blok şeması.

Şekil 4-16, bazı sayıcıların modunu 6 veya 12 ye değiştirerek nasıl saat, dakika ve saniye göstergesi olan bir sayısal saat yapabileceğinizi göstermektedir. Bu saatin problemleri çok fazla entegre devre paketi kullanması ve epeyce bir güç harcamasıdır. Her iki problem de National MM53108 gibi MOS saat entegrelerinden birini kullanmakla çözülebilir. Şekil 4-17 a daki MM53108 in blok şemasında gösterildiği gibi, bu entegre sadece temel saat için gerekli sayıcıları ve kod çözücüleri değil ayrıca *"öğleden önce, (ÖÖ)"* ve *"öğleden sonra (ÖS)"* devrelerini, bir alarm, bir tekrarlı alarm ve siz uyuduktan sonra bir radyoyu kapatmak üzere uyku zamanlayıcı devrelerini içerir.



Not: MM53108 in bacak bağlantıları parantez içinde gösterilmiştir.

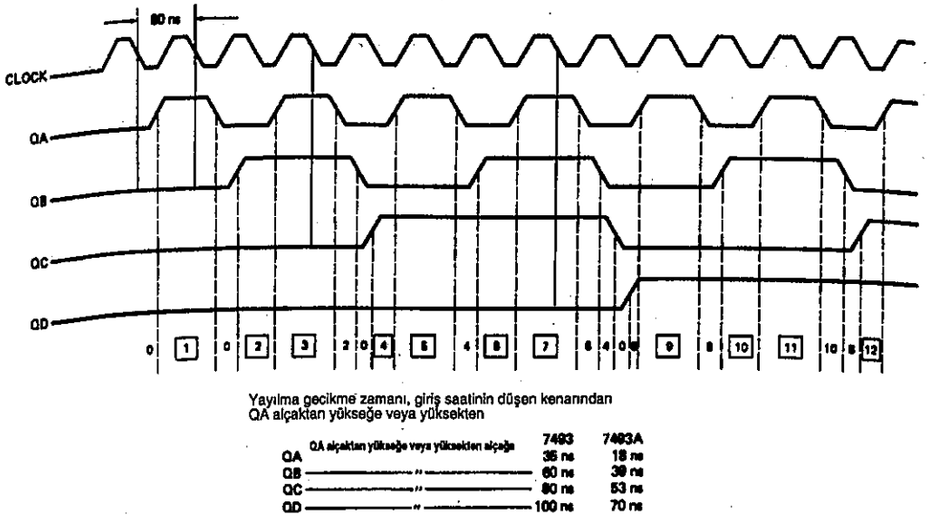
(a)



(b)

Şekil 4-17 Büyük-ölçekli- entegrasyon sayısal saat yapısı.(a) MM53108 saat entegresinin blok şeması. (b) MM53108 kullanan sayısal alarmlı saatin tam şeması (National Semiconductor.)

Şekil 4-17 *b*, MM53108 kullanan tam bir sayısal alarmlı saatin devresini göstermektedir. Gerekli olan tek ilave giriş 50 veya 60 Hz lik darbedir. Bu darbe sinyali, güç transformatörü üzerindeki bir uçtan bir 100 KΩ luk direnç üzerinden yongaya verilir. İsterseniz bir kristal osilatörün çıkışını bölerek de bu sinyali sağlayabilirsiniz. Şekil 4-17 *a* daki saat entegresi saniye düğmesi basılmadıkça saniyeleri göstermez, fakat MM5309 gibi başka entegreler saniyeleri de göstermektedir.



Yayılma gecikme zamanı, giriş saatinin düşen kenarından
QA alçaktan yükseğe veya yüksekten

QA alçaktan yükseğe veya yüksekten alçağa	7493	7493A
QA	35 ns	18 ns
QB	60 ns	39 ns
QC	80 ns	53 ns
QD	100 ns	70 ns

Şekil 4-18 7493 sayıcının istenmeyen sayım durumlarını gösteren zamanlama diyagramı.

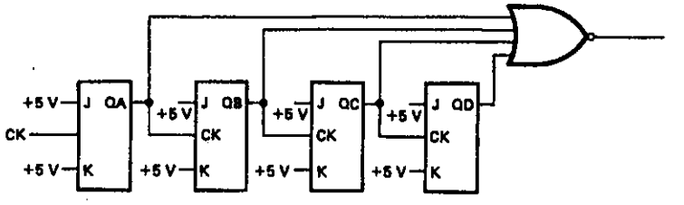
DALGACIK VEYA ASENKRON SAYICILARIN PROBLEMLERİ

Dalgacık sayıcılarındaki problemlerin çoğu, bir zincirdeki birinci flip-flop çıkışının durumunu değiştirmesi ile daha sonraki bir kademenin çıkışını değiştirmesi arasındaki gecikmeden doğmaktadır. Bu mümkün problemlerden ikisini göstermek için bildiğimiz 7493 ü kullanacağız. İlk problem sayım çıkışındaki hatalardır. Şekil 4-18, 12,5 MHz lik giriş saati olan bir 7493 ün

zamanlama diyagramını göstermektedir. Flip-flop yayılma gecikmelerini içeren bu zamanlama diyagramını Şekil 4-11 c deki 7493 ün ideal zamanlama diyagramı ile karşılaştırın. 12,5 MHz lik giriş için çıkışların gösterdiği ikilik durumlara dikkat edin. Bu hatalı durumlar dalgacık sayıcısında her zaman üretilir. Bir saat göstergesini sürmek gibi alçak-frekans uygulamalarında, bir periyodun çok ufak bir parçasını oluşturduklarından gözle görülmezler. 12,5 MHz de giriş saatinin bir periyodu gösterildiği gibi 80 ns dir. Bu, sayıcının flip-floplarındaki yayılma gecikmesi ile yaklaşık aynı boydadır ve böylece hatalı durumlar ciddi boyutlara ulaşır. Çıkıştaki sayımın darbe sayısıyla pek bir ilişkisi yoktur. Bu nedenle, yüksek - frekans sayıcı uygulamalarında bu aygıtın bir yararı yoktur, fakat frekans bölücü olarak hala faydalıdır. Zamanlama diyagramından *QA* nın frekansının, saat frekansının tam yarısı, *QB* nin f_c /4, *QC* 'nin f_c /8 ve *QD* 'nin f_c /16 olduğunu görebilirsiniz.

Dalgacık sayıcılarında ikinci problem, kod çözmeye çalıştığınızda, bu hatalı durum yani sayımlar tarafından üretilir. Bir sayıcının kodunu çözmek, sayıcının verilen bir durumunu algılamak ve bu durum algılandığında bir çıkış darbesi üretmektir. Şekil 4-19 daki dört-girişli VEYA DEĞİL kapısı sayıcının bütün çıkışlarının her sıfır oluşunda bir çıkış darbesi verecektir. İdeal olarak, bu her 16 giriş saat darbesinde bir defa olmalıdır. Şekil 4-18 deki zamanlama diyagramı, bu sayıcının çıkışlarının, her 16 saat darbesinde dört defa sıfır durumundan geçtiğini göstermektedir. Herhangi bir çalışma frekansında bu böyledir. VEYA DEĞİL kapısı gerçek 0 durumunu gösteren bir darbe ile üç tane istenmiyen 20 ns genişlikli darbe vermektedir. İstenmiyen darbeler, çıkışın 1 den 2 ye geçişinde, 3 ten 4 e geçişinde ve 7 den 8 e geçişinde olmaktadır. Burada hatırda tutulması gereken nokta her asenkron veya dalgacık devrenin, kod çözümünde mutlaka bu istenmiyen darbeleri üreteceğidir. Bu darbeler 20-30 ns genişliğinde olabilir, fakat çoğunlukla da problem çıkarırlar. Bu nedenle onlara dikkat edin. Bölüm 1 de anlatıldığı gibi bir gecikme taramalı zaman tabanlı osiloskop gerçek devrede onları görmenize yardımcı olabilecektir.

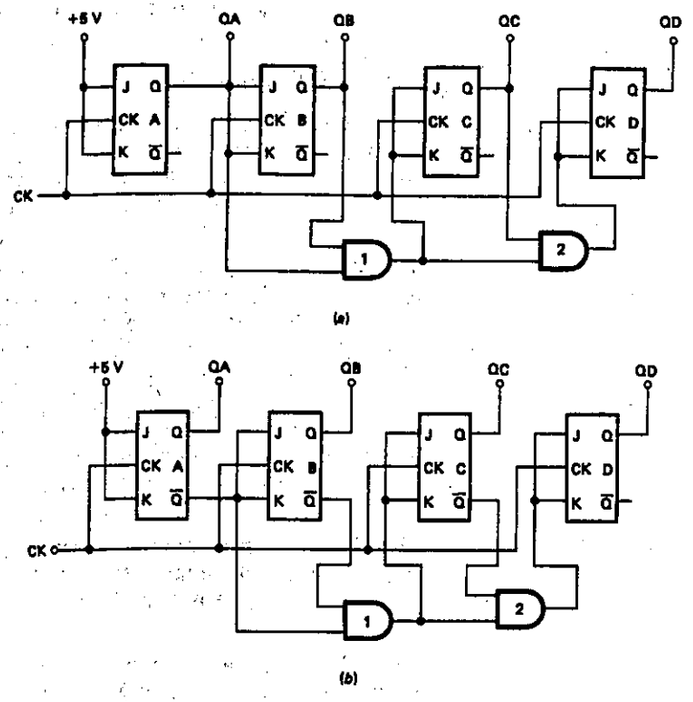
Bütün flip-flopların çıkışlarının aynı anda değişmesini sağlayacak şekilde devre değiştirilirse her iki problem de ortadan kalkacaktır.



Şekil 4-19 Dalgacık sayıcının 0 durumunu çözümlemek için kullanılan dört-girişli bir VEYA DEĞİL kapısı.

SENKRON SAYICILAR

Bir senkron sayıcı, bütün flip floplar aynı saat girişi ile aynı anda saat geçişlerini gördüklerinden, çıkışları aynı anda değişen bir sayıcıdır. Şekil 4-20 a, dört *JK* ana-uydu flip-flopla yapılmış bir 4 - bitlik senkron sayıcının şemasını göstermektedir. *J* ve *K* girişleri alçak iken saat gelse de bir *JK* flip-flopun çıkışlarının değişmediğini ve *J* ve *K* girişleri yüksekse saatle birlikte çıkışlarının durum değiştirdiğini hatırlayın. Şimdi, bu devrenin nasıl çalıştığını görmek için bir sayım dizisi üzerinde çalışabilirsiniz.



Şekil 4-20 JK flip-floplarla yapılmış senkron sayıcılar.
***(a)* İkilik yukarı sayıcı. *(b)* İkilik aşağı sayıcı.**

Bütün çıkışların 0 olduğu reset durumu ile başlarsanız, *B*, *C* ve *D* flip-floplarının *J* ve *K* girişlerinin hepsi alçaktır; bu nedenle de bir saat darbesi çıkışlarını değiştirmeyecektir. Ancak, *A* flip-flopunun *J* ve *K* girişleri sürekli yükseğe bağlıdır ve ilk saat darbesi ile çıkışı 0 dan 1 durumuna geçer. Şimdi B flip-flopunun *J* ve *K* girişleri yüksek olduğuna göre, bir sonraki saat darbesi, 2 sayım çıkışını vermek üzere *QB* yi 1 e ve *QA* yı 0 a değiştirecektir. *QA* şimdi alçak olduğuna göre, bir sonraki saat darbesi ile sadece *A* flip-flopu durum değiştirecektir. Bu durumda sayım çıkışı 0011, yani 3, olacaktır. *QA* ve *QB*'nin her ikisi de yüksek olduğundan VE kapısı 1,*C* flip-flopunun *J* ve K girişlerine bir yüksek verecek ve *C* nin çıkışı bir sonraki saat darbesiyle 1 e yükselecektir. Sayım dizisinin gerisi buna benzer bir desenle devam eder. *QA*, *QB* ve *QC* nin hepsinin 1 olduğu 7 sayımında, VE kapıları, sayımı 8 yapmak üzere *D* flip-flopunun durumunu 1 e değiştirmesini sağlar. Sayım 15 te, bir sonraki saat darbesinde bütün flip-flopların çıkışları 0 olmak üzere durumları değişir.

MOD-N BÖLÜCÜLER VE SAYICILAR

Senkron sayıcılarda yapılan iş, çıkış durumlarının kodlarının çözülmesi ve bunun *J* ve *K* girişlerini hazırlamakta kullanılmasıdır. Böylece bir sonraki saat darbesi, her flip-flopu istenilen çıkış durumuna toggle eder veya etmez. Bu çok güçlü teknik, istenilen mod veya çıkış sayım dizisi için sayıcılar yapmakta kullanılabilir.

Geribeslemeli bir *JK* flip-flop devresinin analizini yapmak için, bilinen çıkış durumları ile başlayın. Bu durumlardan kod çözme mantığı (geribesleme kapıları) üzerinden giderek flip-flopların *J* ve *K* girişlerinde oluşacak durumları belirleyin. *J* ve *K* girişlerini bilmek, bir saat darbesinden sonra her flip-flopun çıkış durumunu öngörmenizi sağlayacaktır. Daha sonra bu çıkışlar, kod çözücü mantık üzerinden izlenerek *J* ve *K* girişleri bulunur ve bir sonraki saat darbesiyle çıkışların ne olacağı öngörülür. Bu sürece, çıkışlar başlangıç durumlarına dönünceye kadar devam edin. Ek *A* da bunun gibi bir devrenin durum dizisini gösterme de çok kullanılan mantık durum diyagramları anlatılmaktadır.

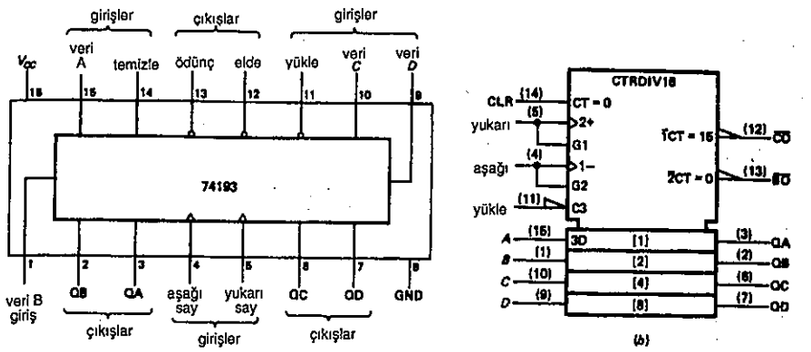
Şekil 4-20 *b*, başka bir faydalı senkron sayıcının devresini göstermektedir. Bu devrenin sayım dizisini izlemeye çalışın. Başlangıç olarak, bu devrede çıkışların hepsi 0 olduğunda, bir sonraki saat darbesiyle flip-flopların hepsinin 1 'e durum değiştireceğine dikkat edin.

Şekil 4-20 *b* deki sayıcı, herhalde keşfetmiş olduğunuz gibi, bir 4-bitlik ikilik aşağı sayıcıdır. Bir senkron sayıcıya her ihtiyaç duyduğunuzda oturup bir tane sentezlemenize gerek yoktur.

Bu sayıcıların birçok hazır tipleri vardır. TTL 74193 ile CMOS 74C193 ve CD40193 senkron 4-bitlik yukarı / aşağı ikilik sayıcılardır. TTL74192 ile CMOS 74C192 ve CD40192 ise senkron 4-bitlik yukarı /aşağı onluk sayıcılardır. 74161, bir senkron ikilik yukarı sayıcıdır, ve 74160 bir senkron onluk yukarı sayıcıdır. Bu sayıcıların standart TTL tiplerinin hepsi 20 MHz ve hatta üzerindeki frekanslara kadar olan giriş frekanslarında doğru sayım yapabilir. Kodları çözüldüğünde, flip-floplardaki yayılma gecikmelerindeki farklılıklardan ve çıkışın alçaktan yükseğe geçişi ile yüksekten alçağa geçişi arasındaki zaman farklılıklarından doğan çok küçük istenmeyen darbeler üretmektedirler.

Sık kullanılan 4-bitlik senkron ikilik sayıcı 74193 'ün bacak şeması, bağımlılık gösterilişi mantık sembolü ve tipik sayım dizisi diyagramı Şekil 4-21 'de gösterilmiştir. Bunlardan bu sayıcının birçok özelliklerini belirleyebilirsiniz. Bu sayıcının bir avantajı, eğer saat yukarı-say girişine gönderilirse yukarı saymakta, aşağı-say girişine gönderilirse aşağı saymaktadır. Sayıcı, saatin pozitif kenarında bir artmakta veya azalmaktadır. Kullanılmayan girişler yüksek tutulmalıdır. Yukarı sayarken, bir son sayım 15, elde (carry) çıkışında bir alçağa giden darbe oluşturur. Bu, giriş saatinin düşen kenarında olur. Aşağı sayarken, bir son sayım 0, *ödünç (borrow)* çıkışında bir alçağa giden darbe oluşturur. Şekil 4-21 a da çıkışlardaki yuvarlaklar, çıkış darbesinin negatif, yani aktif alçak, olduğunu belirtmektedir. *Yükleme (load)* girişindeki yuvarlak, bu girişteki bir alçak durumun sayıcıya yükleme yaptıracağını gösterir. Temizleme (clear) girişi asenkrondur, yani bu giriş yüksek yapıldığında, saatin durumuna bakılmaksızın, bütün çıkışlar temizlenecek yani 0 a reset edilecektir.

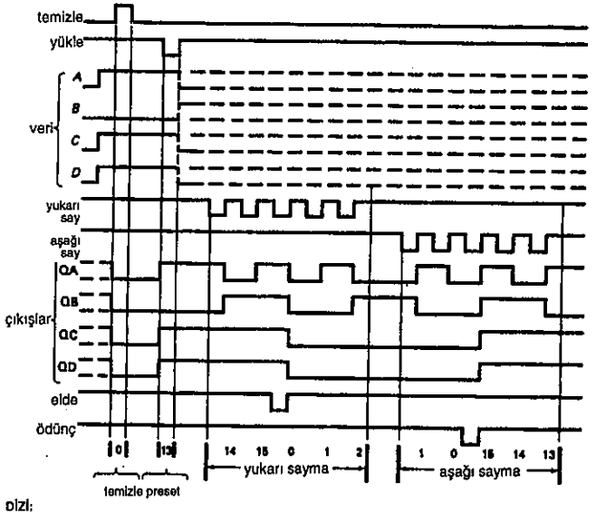
Sayıcı önceden ayarlanabilir, yani dört veri girişine yerleştirilen ikilik sayı, yükleme girişi alçak yapıldığı anda çıkışlara yüklenecektir. Çıkış sayım dizisi, bu yüklenen sayıdan yukarı veya aşağı doğru ilerleyecektir. Bu özellik, bu sayıcının 1 ile maksimum durum sayısı arasında herhangi bir mod da çalışan sayıcı haline dönüştürülmesi için kullanılabilir. Şekil 4-22, bunun nasıl yapılacağını göstermektedir. Sayıcı aşağı saymak üzere ve ödünç çıkışı doğrudan paralel yükleme girişine bağlanmıştır. Örneğin, eğer bir modulo-11 sayıcı istiyorsanız, gösterildiği gibi veri girişlerine ikilik onbir (1011) sabit bağlantısını yapın. Sayıcı 11 den 0 a kadar aşağı sayacaktır. Sayım çıkışı 0 a ulaştığında, ödünç çıkışından bir darbe verilecektir. Bu darbe veri girişlerindeki onbir sayısını sayıcı çıkışlarına yükleyecektir. Bundan sonraki saat darbeleri sayıcıyı tekrar 11 den 0 a doğru azaltacak, ve sıfırda yine bir ödünç darbesi üretilerek çevrim devam edecektir. Devre bir mod-11 bölücüdür, çünkü her 11 giriş darbesinde bir ödünç çıkışında bir çıkış darbesi üretmektedir.



(b)

Not: Yükledeki alçak gibi QA = A, QB = B, QC = C ve QD = D yapar.

(a)



DİZİ:

(1) Çıkışlarısıfırla.
(2) İkilik onüçü yükle
(3) Ondört, onbeş, elde, sıfır bir ve iki diye yukarı say.
(4) Bir, sıfır, ödünç, onbeş, ondört ve onüç diye aşağı say.

NOTLAR:

(A) Temizle girişi yükle, veri ve sayma girişlerine göre üstündür.
(B) Yukarı sayarken, aşağı say girişi yüksek olmalıdır; aşağı sayarken yukarı say, girişi yüksek olmalıdır.

(c)

Şekil 4-21 74193 ikilik yukarı/aşağı sayıcı. *(a)* Bağlantı şeması. *(b)* Bağımlılık gösterilişi mantık şeması. *(c)* Tipik temizle, yükle ve sayma dizileri.

Şekil 4-22 *b*, bir mod-11 bölücü olarak kullanılan bir 74193 ün zamanlama dalgaşekillerini göstermektedir. Bu dalgaşekillerinde dikkat edilmesi gereken üç önemli nokta vardır. Birincisi ödünç çıkışı yükleme girişine bağlıdır ve ödünç çıkışı düşer düşmez yükleme girişi alçak olacaktır. Bu *Q* çıkışlarına 1011 yükleyecektir ve ödünç çıkışını derhal yüksek yapacaktır. Sonuçta oluşan ödünç-yükleme darbesi sadece 30-40 ns genişliğinde olacaktır.

İkincisi, *Q* çıkışlarında bulunan durumların sayısını sayarsanız, 0 dan 11 e kadar olmak üzere 12 tane olduğunu göreceksiniz. Ancak, 0 durumu ile 11 durumu sadece yarım periyotluk sürelerle vardır ve ödünç çıkış darbeleri arasındaki toplam giriş saat darbesi sayısı 11 dir. Bu iki yarım periyotluk durumdan dolayı devre bazı uygulamalarda sayıcı olarak kullanılamaz, fakat frekans bölücü olarak kullanılabilir.

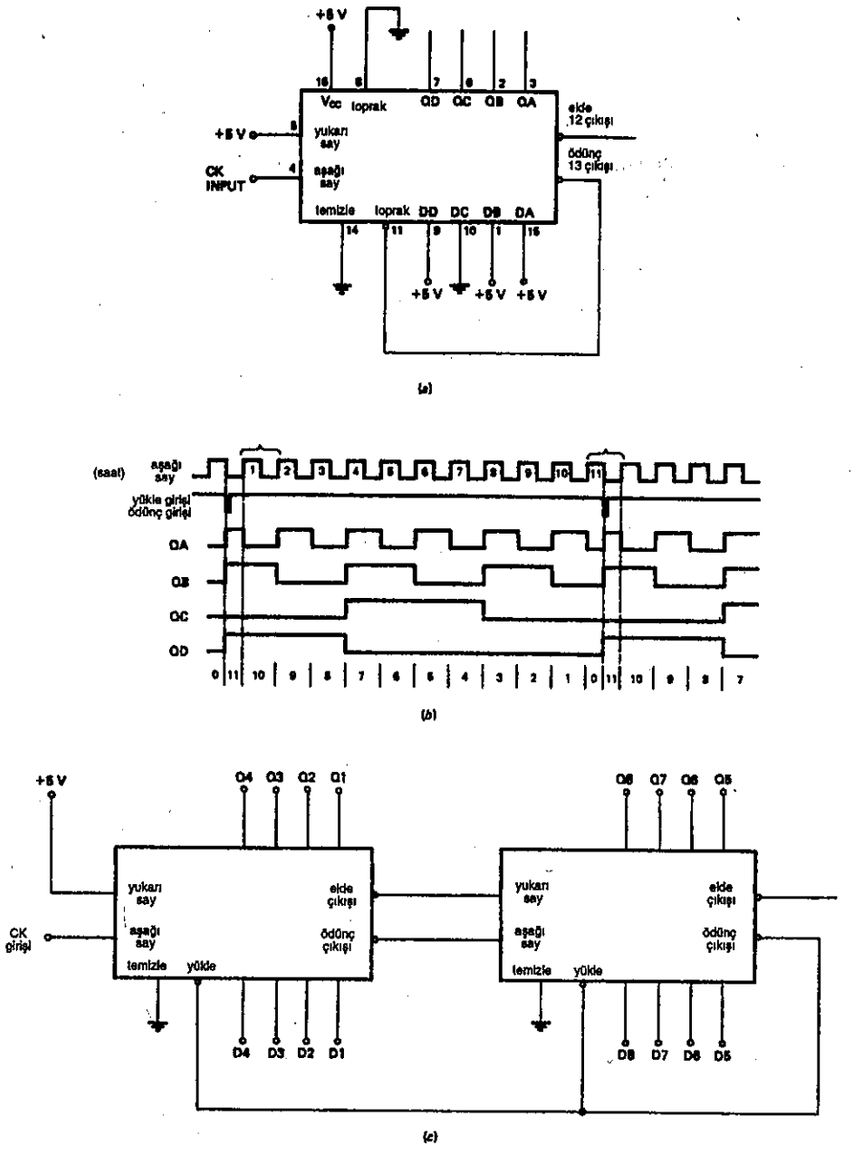
Üçüncüsü, bu devrede *QC* ve *QD* çıkışlarının her ikisi de, her 11 giriş saat darbesi için bir çıkış darbesi üretmektedir. Bunların doluluk oranı % 50 ye daha yakın olduğundan, ödünç çıkışı darbesine göre daha çok kullanılışlı olabilmektedirler.

İstediğiniz herhangi bir modu, karşılığı olan ikilik sayıyı veri girişlerine bağlayarak basitçe üretebilirsiniz. 15 den daha büyük bir mod, Şekil 4-22 *c* de gösterildiği gibi iki veya daha fazla sayıcıyı seri bağlayarak üretilebilir. Son sayım ulaşıldığında, en önemli sayıcının ödünç çıkışı yükleme darbesini üretir ve bütün sayıcılar yeniden yüklenir.

Şimdi kapıları, mandalları ve sayıcıları öğrendiğimize göre, giriş darbelerinin frekansını sayısal göstergede veren bir frekans sayıcısını bu parçaları kullanarak nasıl kuracağınızı göstereceğiz.

FREKANS SAYICILAR : BİR MSI SAYICI

Şekil 4-23, hazır bulunan TTL parçalarını kullanan bir frekans sayıcının şemasını göstermektedir. Bu, bir çok entegreyi içeren oldukça büyük ve kompleks bir şema olduğundan, tanıdık "işaretleri" yani işlevini bildiğiniz devre bloklarını arayıp bularak onu basitleştirmek en iyi yoldur. Bu frekans sayıcıda önce 1 MHz lik kristal osilatörü ve bu referans frekansını bölerek istenilen çıkış frekanslarını elde etmekte kullanılan onluk sayıcılar zincirini tanıyabilirsiniz. Daha sonra her biri bir 74175 dört-bitlik D flip-flop, bir 7447 BCD-den-yedi-parçalıya kod çözücü ve bir yedi-parçalı LED göstergeden oluşan senkron onluk sayıcılar bölücü zincirini fark edebilirsiniz. Bu büyük devre bloku, kısacası 999.999 a kadar sayan ve gösteren bir birim sayıcıdır.



Şekil 4-22
74193 ile yapılmış mod-N sayıcı/bölücü. *(a)* Mod-11 şeması. *(b)* Mod-11 dalgaşekilleri. *(c)* 16 dan büyük modlar için devre;

Bu arada, 74160 sayıcıların, soldaki ilk sayıcıdan sonraki sayıcıların bir büyük senkron sayıcı olarak çalışmasını sağlayacak şekilde bağlandıklarını belirtelim. Bu özellik, sayıcı kısmının yüksek giriş frekanslarını da doğru olarak saymasına imkân verir.

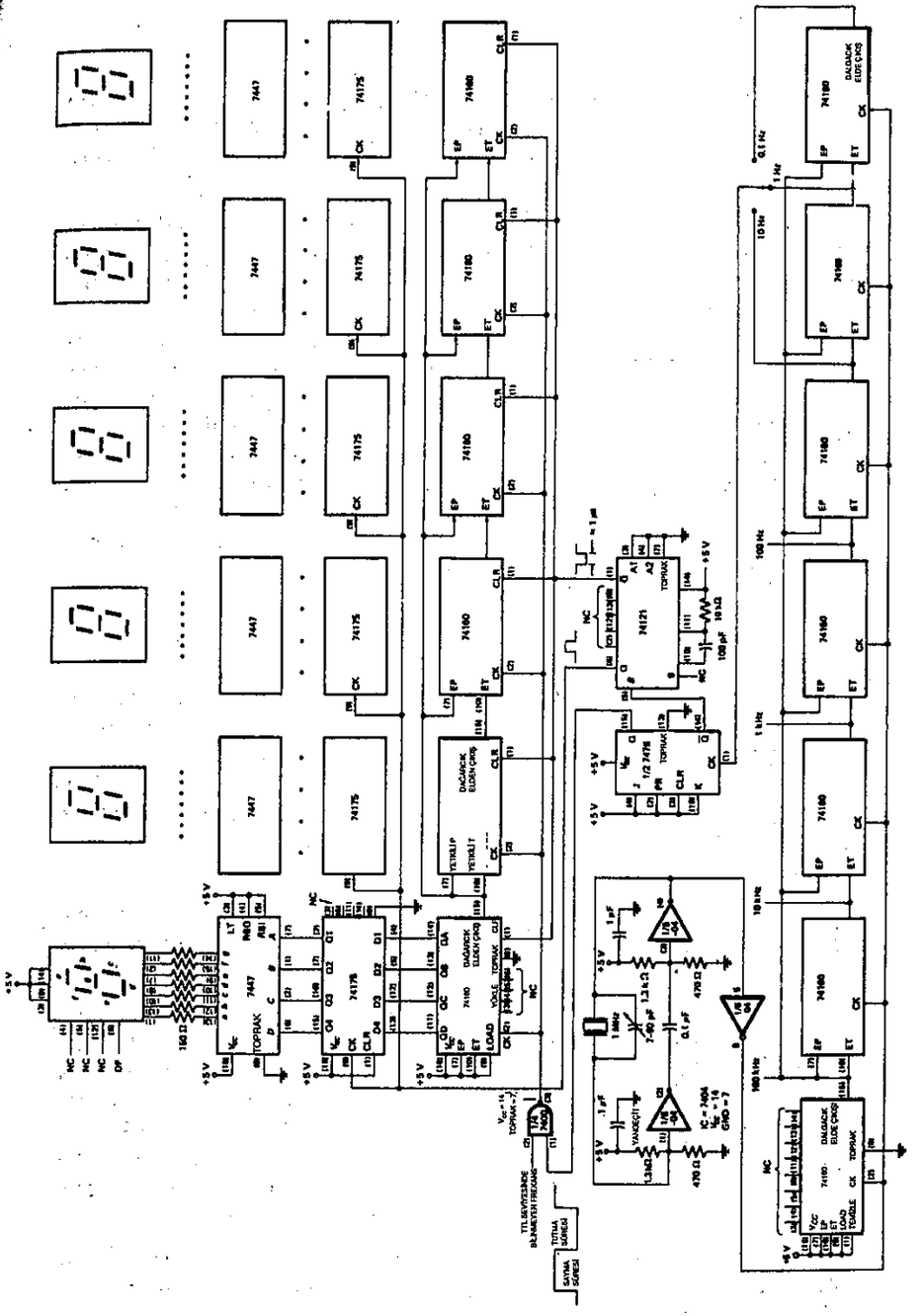
Devrenin büyük bloklarını anlamış olduğunuza göre, şimdi bunları birbirine bağlayan küçük parçalara bakmaya başlayabilirsiniz. Bilinmeyen frekansın sayıcı kısmına bir VE DEĞİL kapısıyla verildiğine dikkat edin. Bilinmeyen frekans darbeleri, eğer bu VE DEĞİL kapısının bacak 1 girişi yüksek ise, sayılmak üzere içeri girebilirler. Eğer bu kapı girişi tam 1 sn yüksek yapılırsa, bu takdirde göstergede 1 sn içinde sayılan bilinmeyen giriş darbelerinin sayısı okunacaktır. Tabii ki bu, bilinmeyen sinyalin Hz cinsinden frekansıdır. Şimdi bu 1 sn lik sayım penceresinin nasıl üretildiğine bakalım. Yüksek zamanı 1 sn olan bir kare dalganın frekansı 0,5 Hz dir. Kristal osilatörün önündeki bölücü zinciri, bir *JK* flip-flopun saat girişine giden bir 1 Hz lik çıkışa sahiptir. *J* ve *K* girişleri yükseğe bağlanmış 7476 JK flip-flopu, bu giriş frekansının yarısı frekansta bir simetrik kare dalga çıkışı verecektir. O halde, bu *JK* flip-flop 1 sn lik hassas sayım penceresini üretmektedir.

Bundan sonra yaklaşacağımız devre parçası 74121 tek kararlı devresidir. Tek kararlı sayım pencere darbesinin negatif kenarı onu tetikleyerek pozitif bir darbe üretecek şekilde bağlanmıştır. Bu pozitif darbenin ön kenarı sayıcıların çıkış durumlarını 74175 D flip-flopların çıkışlarına yükler. Böylece, flip-floplar, bütün sayım çevrimi boyunca kararlı veri tutarlar. Flip-floplar olmasaydı, göstergeler 1 sn lik sayım sırasında bir sayı kargaşası gösterecek, pencere darbesinin 1 sn lik tutma zamanında da son okumayı verecekti. Bu göstergeyi okumayı çok zorlaştıracaktı. Flip-floplar, göstergedeki verilerin yenilendiği birkaç nanosaniyelik süre dışında ikişer saniyelik sürelerle göstergedeki verileri sabit tutarlar.

74121 in $\overline{Q}$ çıkışı, bütün sayıcıları, bir sonraki sayım dizisine başlamaya hazırlık olmak üzere, 0 a reset etmek için kullanılır. Şekil 4-24 bu darbelerin zamanlamasını göstermektedir.

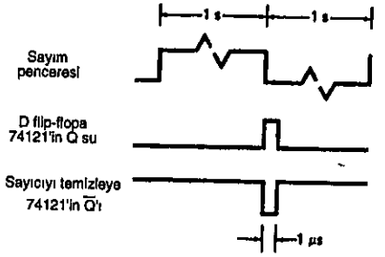
Bir saniyelik sayım penceresi ve altı basamakla, sayıcı, gösterildiği gibi, taşmadan önce sadece 999.999 a kadar olan frekansları ölçebilir. Göstergeye bir virgül eklenerek, maksimum frekans 999,999 KHz olmak üzere okuma KHz cinsinden yapılabilir. Cihazın frekans alanı, sayım penceresi zamanını 0,1 saniyeye kısaltarak ve ondalık virgülü gösterge maksimum 9999,99 KHz i gösterecek şekilde kaydırarak, yaklaşık 10 MHz e kadar okuyacak şekilde artırılabilir. Sayım penceresi sadece onda bir uzunluğunda olduğuna göre, sayıcıları taşırmadan 10 misli daha büyük frekanslar ölçülebilir.

Frekans alanını artırmanın diğer bir yolu, bir başka onluk sayıcı, kod çözücü ve gösterge ilave etmektir. Alanı daha da yüksek giriş frekanslarına



Şekil 4.23 Bir MSI frekans sayıcının tam şeması

genişletmek için bir veya daha fazla önölçekleyici (prescaler) kullanılabilirsiniz. Bir önölçekleyici, giriş sinyali ile seri bağlanan bir 74S196 gibi çok yüksek hızlı onluk sayıcıdan ibarettir. Böylece giriş sinyalinin frekansı bölünerek sayıcı zincirini sayabileceği aralığa düşürülür. Yüksek frekans ölçümleri için önölçekleyiciyi devreye sokan anahtarın bir kısmı aynı zamanda gösterge doğru okuyacak şekilde ondalık virgülü kaydırabilir. Önölçekleyecilerin doğrudan kendilerine bağlanmış göstergeleri yoktur.



Şekil 4-24 Şekil 4-23 deki frekans sayıcının kontrol devresinin zamanlama dalga şekilleri.

Alçak-frekans ölçümleri için, 1 sn lik pencere kötü bir çözünürlük verir. Örneğin 20 Hz de, bir saniyelik bir pencere, sayıcılar 1 Hz in altındaki kesirleri sayamadıklarından, sadece ± 1 Hz lik bir çözünürlük verir. Bu % ± 5 lik bir hatadır. Eğer sayım penceresini 10 sn ye uzatır ve ondalık virgülü uygun şekilde kaydırırsanız, çözünürlük ± 0,1 Hz olur, çünkü şimdi 200 ± 1 darbe sayılmaktadır.

Temel frekans sayıcıda yapabileceğiniz diğer bir iyileştirme, giriş sinyal uyumlaması yapmaktır. Gösterilen şekliyle devre TTL giriş seviyeleri gerektirmektedir. AM 686 gibi bir yüksek-hızlı karşılaştırıcı sayıcıların önüne eklenerek küçük sinüs dalgaları veya diğer dalga şekilleri TTL seviyelerine dönüştürülebilir. Tetikleme gerilim seviyesinin gürültü darbelerinin üzerine ayarlanabilmesi için karşılaştırıcının eşiği değişken yapılabilir.

Devrenin çalışmasını böylece özetlemiş olduk. Şekil 4-23, eleman değerlerini ve bacak numaralarını göstermektedir, böylece isterseniz devreyi kurup çalışmasını görebilirsiniz. Not : Şemada en küçük değerlikli basamak soldadır.

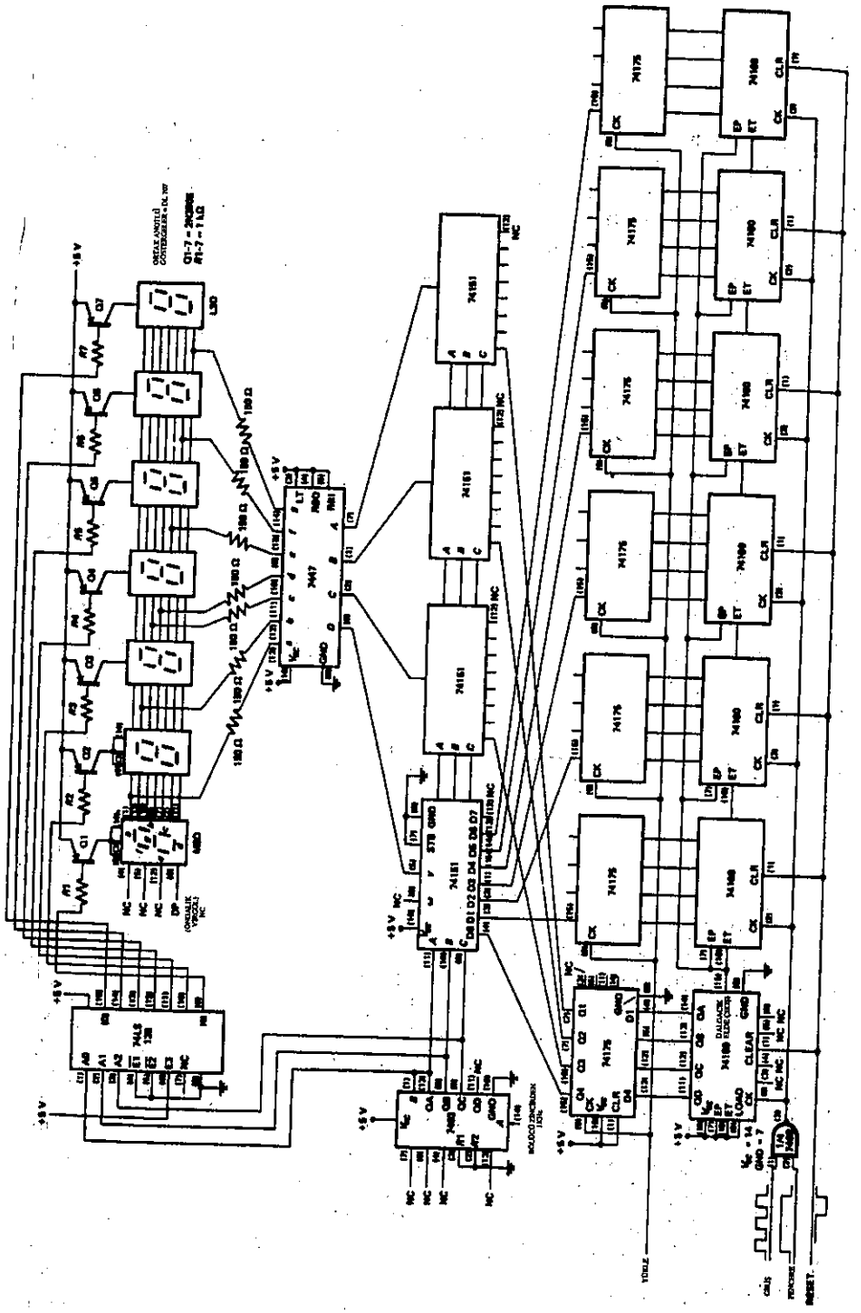
ÇOĞULLAMALI GÖSTERGELER

Şekil 4-23 teki devre gayet iyi çalışır, fakat pratik açıdan iki problemi vardır. Çok sayıda entegre paketi kullanmaktadır ve oldukça büyük bir miktar akım çekmektedir. Her iki problem de kolayca çözülebilir. Güç problemi için ilk bakılacak yer, gücün en fazla harcandığı yer olan, göstergelerdir. Yedi-parçalı göstergenin her parçası başına 20 mA lik bir akım ve her basamakta 7 parça olduğu düşünülürse, her basamaktaki maksimum akım 140 mA olur. Sadece 6 basamaklı göstergedeki toplam maksimum akım 840 mA olur. Altı tane 7447 de 260 mA ekleyince, toplam 1,1 A olur. Bu miktar göstergeyi çoğullayarak büyük çapta azaltılabilir. Bölüm 3 ten hatırlayacağınız gibi, çoğullamak, aynı hat üzerine birçok sinyal parçasını sıralamaktır. Çoğullanmış görüntü, Bölüm 3 te tek ışınla sekiz osiloskop izi üretildiği gibi, aynı yolla üretilmektedir. Verilen herhangi bir anda sadece bir giriş görüntülenmektedir; fakat girişler arasında hızlı bir şekilde değişirseniz (yani girişleri hızlı bir şekilde bölümlersiniz), gözünüz sonucu sürekli bir görüntü olarak görecektir. LED göstergelerde, her seferinde sadece bir basamak yakılmaktadır.

Şekil 4-25, bir 7 basamaklı frekans sayıcının çoğullamalı gösterge, sayıcı ve flip-flop kısımlarını göstermektedir. Burada, dört 74151 sekiz-girişli çoğullayıcı, yedi 74175 dörtlü flip-flopun çıkışını tek bir 7447 BCD - den - yedi - parçalı kod çözücüye çoğullamaktadır. O halde 7447 nin herbir çıkışı, bütün yedi-parçalı göstergelerin karşılık gelen girişlerine bağlanır. Böylece, bir 5 in yedi-parçalı kodu 7447 nin çıkışlarında ise, bu bütün göstergelere taşınır. Ancak, herhangi bir anda her gösterge ile seri olan PNP transistörlerinden yalnız birisi iletimde olduğundan, göstergelerden yalnız bir tanesi bu 5 i gösterir.

3-bitlik ikilik sayıcının çıkışları, dört 74151 in hangi onluk sayıcının çıkışlarını 7447 ye göndereceğini belirler. Aynı 3-bitlik sayıcının çıkışları, hangi transistörün iletimde olacağını ve böylece hangi göstergenin yanacağını seçen 74LS138 sekizinden-biri-alçak kod çözücüsüne bağlıdır. 74LS138 ve 74151 ler senkronize edilmişlerdir. Böylece, örneğin 74151 'ler en küçük değerlikli basamak sayıcısından göstergelere veri gönderirlerken, 74LS138 de göstergenin en küçük değerlikli basamağının basamak sürücü transistörünü aktif hale getirir. Her basamak 50 ile 200 Hz arasında bir oranda tazelenmelidir.

Bu çoğullamalı sistemdeki kazanç 600 mA civarındadır. Daha fazla kazanç mümkün olabilirdi, ancak LED göstergeleri çoğulladığınızda, görüntü parlaklığını arttırmak için her parçaya giden akımı en az iki misline çıkarmanız gerekmektedir. Bu devre aynı sayıda entegre kullanmaktadır ancak 36 direnç ve birçok bağlantıyı yok etmektedir.



Şekil 4.25 Çoğullamalı göstergeli MSI frekans sayıcının şeması

BİR MSI FREKANS SAYICIDA ARIZA BULMA

Şekil 4-23 teki zaman tabanı ve kontrol devreleri Şekil 4-25 in sayıcı, flip-flop ve gösterge kısımlarıyla birleştirilerek çalışan bir MSI frekans sayıcı yapılabilir. Bir sayısal sistemde, sistemli arıza bulma işlemini anlatırken bu devreyi kullacağız. Her sistemin ayrıntıları farklıdır, fakat temel yaklaşım aynıdır. Önemli noktalar şunlardır :

1. Önce analiz et.
2. Böl ve yen.

Bunun gibi bir devrenin temel örneğini oluştururken, önce hangi kısımların ayrıca kurulup kolayca test edilebileceğini bulmak için şemanın analizi ile işe başlayın. Örneğin, kristal osilatörle başlayabilirsiniz. Sonra, bu doğru çalışıyorsa zaman-tabanı bölücü zincirinin her sayıcısını sırayla ekleyip test edin. Bir sonraki basamak 7476 pencere üretecinin ve 74121 kontrol-darbesi üretecinin kurulup test edilmesi olacaktır.

Bundan sonra, gösterge kısmını kurup ayrı bir modül olarak test edebilirsiniz. Parçaların doğru bağlandığını kontrol etmek için geçici olarak 7447 nin girişlerine 2 veya 5 in BCD kodunu bağlayın. Her basamak sürücü transistörün baz direncinin ucu topraklandığında, yani alçak yapıldığında, o basamak doğru sayıyı göstermelidir. 74LS138 ve 7493 adres sayıcı eklendiğinde ise bütün göstergeler bu sayıyı göstermelidir.

Daha sonra, ana sayıcı zincirini ayrı bir modül olarak kurun. Sayım pencere girişi geçici olarak 7476 yerine yükseğe bağlanabilir ve bilinmeyen sinyal girişine bir 1MHz lik sinyal uygulanabilir. Böylece sayıcılar sürekli olarak bir sinyal alacaktır. Sayıcılar, zaman-tabanı zincirinde yapıldığı gibi, birer birer eklenip kontrol edilebilirler.

Son olarak, flip-flopları ve çoğullayıcıları ekleyin. Sayım penceresi, mandal yükleme ve reset sinyallerini doğru noktalara bağlayın ve sayıcının tamamının doğru olarak çalışıp çalışmadığını kontrol edin. Çalışırsa mükemmel ! Gidip bir çay için. Çalışmazsa, bu son kısım aşağıdaki sırada izlenebilir. 7493 ü yerinden çıkarın ve geçici olarak *QA*, *QB* ve *QC* çıkış bacaklarını toprağa bağlayın. Şimdi göstergenin en önemsiz basamağı yetkilendirilmiş olacaktır ve çoğullayıcı girişleri soldaki en önemsiz 74175 flip-flopa yönelir. 74121 den yükleme (strobe) girişi çıkarıp toprağa bağlayın. 74121 den reset ucunu çıkarıp +5 V a bağlayın. 7476 dan sayım penceresi ucunu çıkarıp sayıcılara bir sayı yüklemek üzere bir 1 MHz lik sinyali sayma girişine uygulayın. Sonra pencere girişini toprağa bağlayın. Şimdi sayıcıların çıkışlarında bir sayım bulunacaktır. Bu sayım, saat girişi ucunu kısa bir süre topraktan çıkararak 74175 flip-flopların çıkışlarına yüklenebilir. En önemsiz sayıcının mantık durumları şimdi soldaki 74175 in çıkışlarında ve dört 74151 in çıkışlarında

görülecektir. Buradaki önemli nokta, bütün seviyeler sabit tutulduğunda, hepsi de bir voltmetre veya mantık probuyla kontrol edilebilirler. Bu, devre normal saat ve darbelerle çalışıyorken görülecek karmaşık darbe dizilerini çözmeye çalışmaktan çok daha kolaydır.

7493 yerine koyduğunuz bağlantıları QA = +5 V, QB = toprak ve QC = toprak olarak değiştirerek 74151 çoğullayıcılarının bir sonraki 74175 in çıkışlarını alması sağlanabilir. Böylece, her sayıcıdan 7447 girişlerine olan bağlantılar doğru akım temelinde bir mantık probuyla kontrol edilebilir.

DAHA ÖNCE ÇALIŞAN BİR CİHAZIN ARIZASINI BULMA

Daha önce çalışmakta olan bir cihazın arızasını bulma yordamı biraz daha farklıdır, çünkü genellikle bağlantıların doğru olduğunu kabul edebilirsiniz. Burada bütün belirti ve onarımların ayrıntılarına girmemiz mümkün değildir. Fakat bir kaç tanesi size yol gösterecektir.

Önemli parçaların çalışmasını anlamak için sistemin şemasının ve/veya blok diyagramının analizini yaparak işe başlayın. Başlangıçta her entegrenin işlevini belirlemeye gerek yoktur. Çoğu cihaz veya sistemlerde bu çok fazla zaman alacaktır. Bir defa bir problemin belli bir kısma ait olduğunu belirledikten sonra o kısmın ayrıntıları üzerinde durmaya başlayabilirsiniz.

Eğer sayıcı sayma yapıyorsa fakat hatalı sayımlar gösteriyorsa, ilk kontrol edilecek nokta, güç kaynağı geriliminin belirtilen sınırlar içinde olup olmadığıdır. Alçak besleme gerilimlerinde TTL aygıtları hatalı çalışır.

Eğer göstergenin bir basamağı parlak olarak yanıyorsa ve diğerleri karanlıksa, bu 74LS138 basamak tarama devresinin doğru çalışmamasından kaynaklanmaktadır. Ya 74LS138 arızalıdır ya da 7493 ona gerekli ardışıl adresleri sağlamamaktadır. Bir osiloskop, problemin 74LS138 de mi, 7493 de mi yoksa 7493 e gelen saat darbesinde mi olduğunu size derhal gösterecektir. 7493 e saat sinyali zaman-tabanı bölücü zincirinden geldiğinden, sinyalin 7493 e gelmemesi sizi bölücü zincire ve belki de çalışmayan ana osilatöre yönlendirecektir. Bu, bir sayısal sistemde, çoğunlukla bir problemin sebebinin görüldüğü yerden çok uzakta olabileceğini göstermektedir.

Eğer bütün basamaklar eşit olarak yanıyorsa fakat giriş frekansı değiştirildiğinde sayılar değişmiyorsa, problem sayma penceresi olmamasından veya 74175 flip-floplarına saat darbesi gitmemesinden kaynaklanabilir. Eğer bütün basamaklar eşit olarak yanıyorsa, fakat sayım sürekli rastgele sayılara değişiyorsa, sayıcılar bir reset darbesi almıyor olabilirler. Buradaki önemli nokta kontrol ve zamanlama sinyallerinin kontrol edilmesidir. Bu sistemde yükleme ve reset darbeleri sayım penceresine göre çok dardırlar. Bölüm 1 de anlatıldığı gibi bir osiloskopun gecikmeli tarama işlevini kullanmak bu darbeleri açıkça görülür hale getirecektir.

Eğer sayıcı, çoğullayıcı ve flip-flop kısmında bir problem varsa, 7493, 74121 ve 7476 yı yerlerinden çıkarın. Sonra, bu kısımda, daha önce anlatıldığı gibi statik sinyaller elde etmek ve seviyelerini ölçmek üzere uygun bağlantılar yapın.

Özet olarak, sistemli arıza bulmanın esası, belirtilerin ve şemanın analizini yapmak ve sonra problemi kolayca bulabileceğiniz küçük bölgelere sıkıştırmak üzere gerçekleştirebileceğiniz testler kurmaktır.

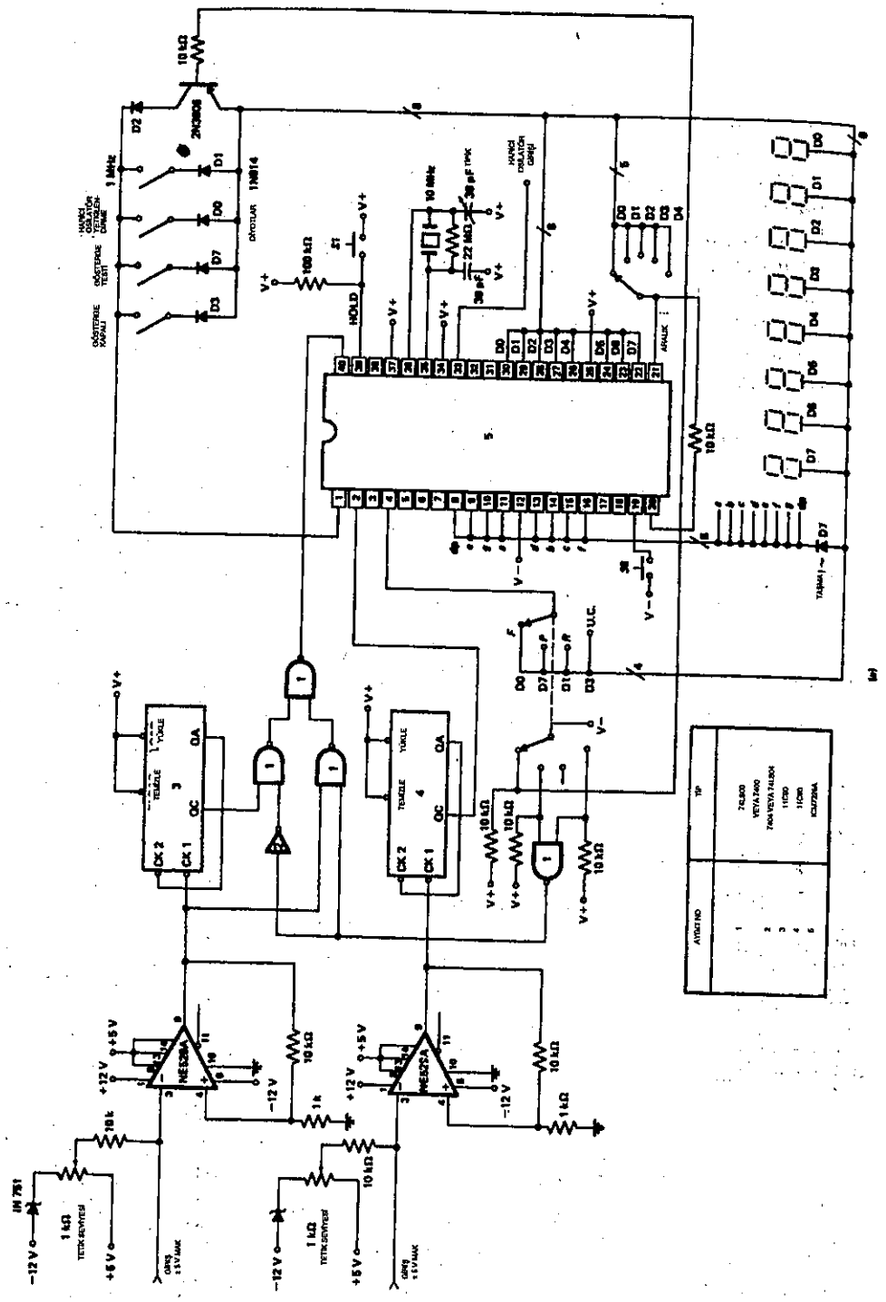
BİR LSI FREKANS SAYICI

Şekil 4-25 te gösterilen devre gayet iyi çalışır, ancak 1970 lerin frekans sayıcısıdır. Devrenin çoğu kısmı alçak frekansta çalıştığından, CMOS ve MOS LSİ, frekans sayıcının çoğunu bir veya iki entegre paketine koymayı mümkün hale getirmiştir. Şekil 4-26 a, Intersil ICM 7226 A kullanan 100 MHz lik bir çok işlevli sayıcının şemasını göstermektedir. CMOS 7226, bir 10 MHz lik referans osilatörü, zaman-tabanı bölücü zinciri, giriş sinyali sayıcı zincirleri, mandallar, kod çözücüler, gösterge parçası sürücüleri, basamak sürücüleri, kontrol devresi ve bir 8-basamaklı ortak-anotlu gösterge sayıcı sistemi için çoğullayıcı devre içerir. NE 529A karşılaştırıcıları giriş sinyali uyumlaştırılması ve 11C90 yüksek hızlı onluk sayıcılar, önölçekleme için kullanılmıştır. Önölçekleyiciler, cihazın frekans alanını, 7226A nın 10 MHz inden 100 MHz e kadar genişletir.

ICM 7226A çok kullanışlı bir entegredir. A ve B olmak üzere iki sinyal girişi olduğuna dikkat edin. Frekansın yanısıra, 7226, periyot, iki frekansın oranını, zaman aralıklarını ve toplam sayımları da ölçebilmektedir. Frekans, periyot ve miktar ölçümleri için sinyal A girişine uygulanır. A girişi 10 MHz e kadar sayabilir. Frekans oranı ve zaman aralığı ölçümleri için, bir sinyal A girişine diğer sinyal de B girişine uygulanır. B girişi 2 MHz e kadar sayabilir. Zaman aralığı ölçümlerinde, 7226, A girişindeki yüksekten - alçağa geçiş ile B girişindeki yüksekten- alçağa geçiş arasındaki zamanı 0,1 μs çözünürlükle ölçer.

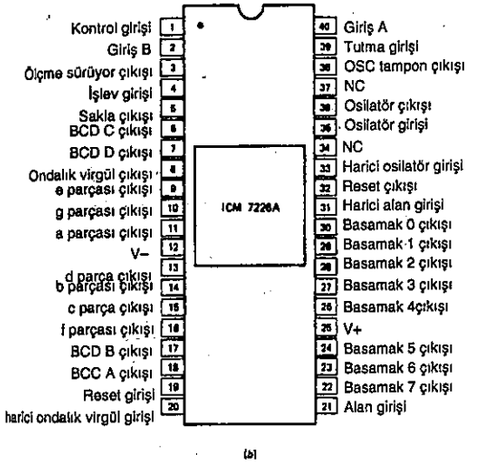
Şekil 4-26 b, ICM 7226A nın bütün bacaklarını göstermektedir. Aygıtın, çoğullanmış yedi-parçalı ve basamak-sürücü çıkışları olduğu gibi çoğullanmış BCD çıkışları da vardır. Yedi-parçalının her parçasının sürücüsü 25 mA e kadar akım çöktürebilir ve her basamak sürücüsü 170 mA e kadar akım kaynaklandırabilir. Ancak, gösterge kapalıyken, 7226A nın kendisi +5 V luk kaynaktan sadece 5 mA çeker.

Şekil 4-26 a daki şemada sekiz basamak yükleme hattının ve sekiz parçalı sürücü hattının birer tek hat olarak nasıl gösterildiğine dikkat edin. Bağlantılar belirgin olduğunda, şemanın çok daha kolay okunmasını sağladığı için, bu yol sık sık tercih edilir.



Şekil 4.26 Intersil ICM 7226A kullanan bir 100 MHz 'lik çok işlevli sayıcı. *(a)* şeması.

7226A nın kendine has diğer bir özelliği de, işlev, alan ve kontrol sinyallerinin aygıta çoğullanış şeklidir. Şekil 4-26 *a* da gösterildiği gibi, basamak-yükleme hatları alan, işlev ve kontrol anahtarlarına bağlanmıştır. İç devre, hangi basamak-yükleme hattının, örneğin, 21 bacağındaki alan girişine bağlandığını algılar. Sonra bu alan için uygun pencereyi ve ondalık virgülün konumunu seçer. Şekil 4-26 *c*, bütün alanlar, işlevler ve kontrollar için basamak- yükleme bağlantılarını göstermektedir. Bu yolla, aygıta verilen bütün sinyaller için gerekli bacak sayısı ciddi bir şekilde azaltılmıştır. Ayrıca, eğer mekanik alan ve işlev anahtarları, CD4051 gibi bir analog çoğullayıcı ile değiştirilirse, seçilen alan ve işlev, çoğullayıcının adres girişlerindeki ikilik kelime ile belirlenir. Diğer bir deyişle, ölçme işlevi ve alanı, örneğin bir mikrobilgisayar veya bilgisayardan gönderilecek ikilik kelimelerle programlanabilir.



(b)

GİRİŞ	İŞLEV	BASAMAK
İşlev giriş bacak 4	Frekans	D0
	Periyot	D7
	Frekans oranı	D1
	Zaman aralığı	D4
	Birim sayıcı	D3
	Osilatör Frekansı	D2
Alan girişi bacak 21	0,01 sn/1 çevrim	D0
	0,1 sn/10 çevrim	D1
	1 sn/100 çevrim	D2
Harici alan girişi bacak 31	10 sn/1k çevrim	D3
	Yetkili	D4
Kontrol girişi bacak 1	Boş gösterge	D3 ve Tutma
	Test görüntüsü	D7
	1 MHz'i seçme	D1
	Harici osilatör yetkili	D0
	Harici ondalık virgül yetkili	D2
	Test	D4
Harici ondalık virgül girişi, bacak 20	Bu girişe bağlanan basamakta ondalık virgül çıkışı verilir.	

(c)

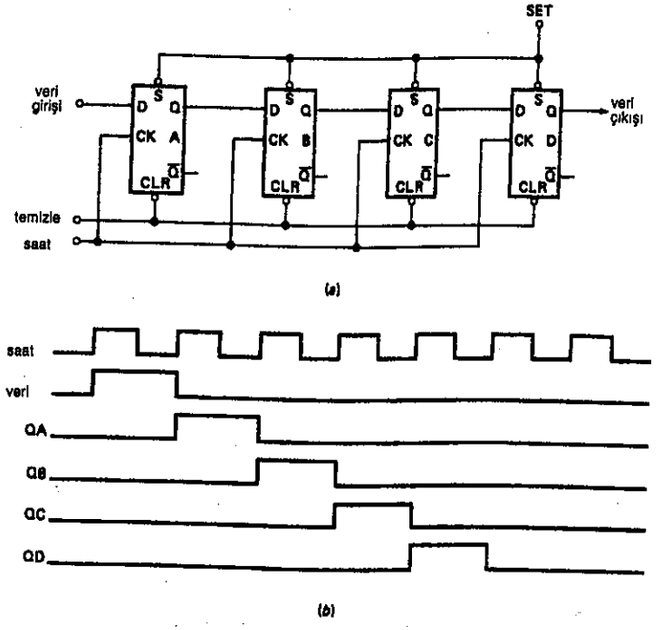
Şekil 4-26 (Devam) *(b)*ICM 7226 A nın bacakları. *(c)* ICM 7226 A nın alan, işlev ve kontrol girişlerinin basamak-yükleme bağlantıları.

KAYDEDİCİLER VE KAYDIRMALI KAYDEDİCİLERİ

Bir kaydedici, veri bitlerini geçici olarak saklamak için kullanılan, genellikle D tipi, bir paralel flip-flop gurubudur. Kaydediciler herhangi bir sayıda bit genişliğinde olabilir. Çok kullanılan genişlikler 4, 8, 12 veya 16 bittir. Bir kaydırma kaydedicisinde, her flip-flopun çıkışı yanındaki flip-flopun girişine bağlıdır, böylece her gelen saat darbesi veri bitlerini bir flip-flop, bağlantı sırasına bağlı olarak, sağa veya sola kaydırır. İzleyen kısımlarda tek tek inceleyeceğimiz dört temel tipte kaydırma kaydedicisi vardır.

SERİ - GİRİŞ, SERİ-ÇIKIŞ KAYDIRMALI KAYDEDİCİ

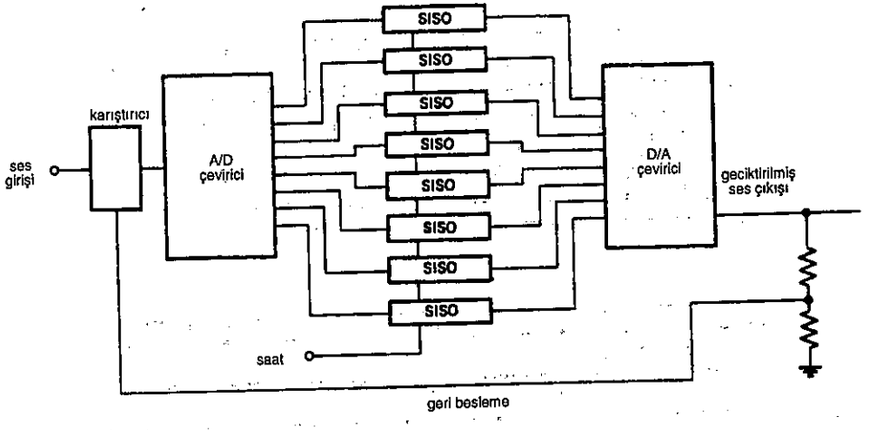
Şekil 4-27 a, *seri-giriş, seri-çıkış (serial-in, serial-out-SISO)* kaydırma kaydedici olarak bağlanmış dört D flip-flopu göstermektedir ve Şekil 4-27 b de devrenin saat, veri çıkış dalgaşekillerini göstermektedir. Bu dalgaşekillerinden gördüğünüz gibi, her saat darbesi, veri darbesini bir flip-flop sağa kaydırır. Çıkış, girişe göre 4 saat darbesi geciktirilmiştir, çünkü veri darbesinin çıkışa ulaşması için 4 saat darbesi gerekmektedir.



Şekil 4-27 **Seri-giriş, seri-çıkış kaydırma kaydedicisi. *(a)* Mantık şeması. *(b)* Saat, veri ve çıkış dalgaşekilleri.**

GECİKTİRME İÇİN SISO

SISO kaydırma kaydedicilerinin bir kullanılış yeri sayısal sinyalleri geciktirmek içindir. Gecikme miktarı saat frekansına ve kaydedicideki flip-flop sayısına bağlıdır. Bir saat darbesinin süresi 1/frekans (Hz olarak) dır. Bu süre flip-flop sayısı ile çarpıldığında kaydedici için toplam gecikmeyi verir. Bazı MOS *SISO* kaydırma kaydedicilerinin 4096 tane flip-flopu vardır. Şekil 4-28, gitar yükselteçlerinde çoğunlukla bulunan mekanik yaylı yankı sisteminin yerine nasıl bir analogdan -sayısala *(A / D)* çevirici, bazı uzun kaydırma kaydedicileri ve bir sayısaldan- analoga *(D / A)* çeviricinin kullanılabileceğini göstermektedir. *A / D* çevirici, girişteki sinüs dalgasını ikilik ağırlıklı 8-bitlik sayısal kelimelere çevirir. Bu sayısal kelimeler sekiz paralel kaydırma kaydedicilerinden geçirilerek geciktirir. Gecikme miktarı, saat frekansı ile kontrol edilebilir. Kaydırma kaydedicilerinin sonunda, bir *D / A* çevirici sayısal kelimeyi tekrar analog sinüs dalgasına çevirir. Çıkışın ufak bir kısmı bir yankılanma etkisi vermek üzere girişe geri beslenir.



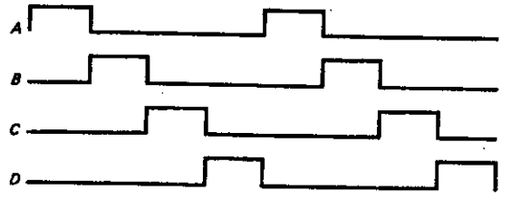
Şekil 4-28 **Bir ses yankı birimi yapmak için *A / D* ve *D / A* çeviricilerle birlikte kullanılan seri - giriş, seri-çıkış kaydırma kaydedicisi.**

SERİ-GİRİŞ, PARALEL-ÇIKIŞ KAYDIRMA KAYDEDİCİLERİ

Bir *seri-giriş, paralel, çıkış (serial-in, parallel-out SIPO)* kaydırma kaydedicisi Şekil 4-27 a daki SISO kaydırma kaydedicisi gibi bağlanmıştır, fakat her flip-flopun çıkışı entegre devre paketinin bacaklarına verilmiştir.

HALKA SAYICI (RING COUNTER) OLARAK SIPO

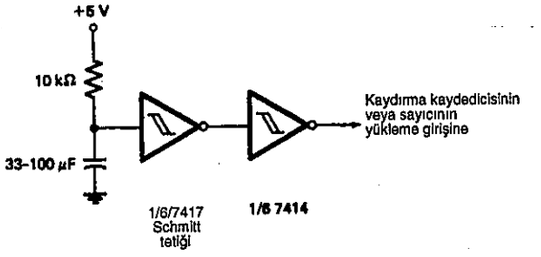
SIPO kaydırma kaydedicisinin önemli bir kullanılış yeri halka sayıcı yapımındadır. Şekil 4-27 a daki kaydırma kaydedicisinin çıkışı girişine bağlanırsa, devreye konulan bir darbe halka içinde dolaşıp duracaktır. Bu halka sayıcının çok kullanışlı paralel çıkış dalgaşekilleri Şekil 4-29 da gösterilmiştir. Gördüğünüz gibi çıkışlar ardışıl ve kısmen bile çakışmayan darbelerdir. Dizi, başlangıçta çıkışlardan birini 1 e set ederek ve diğer çıkışları 0 a reset ederek başlatılabilir. Bunun için doğrudan set ve reset girişleri kullanılabilir. Paralel-yükleme girişleri olan TTL 74195 gibi kaydırma kaydedicilerinde, yükleme girişlerine 1000 bağlantısı yapar ve sayımı başlatmak üzere bunları kaydediciye bir açılışta reset etme devresi ile yükleyebilirsiniz. Açılışta reset etme devresi Şekil 4-30 da gösterilmiştir. Güç açıldığında, büyük kapasitör Schmitt tetiğinin eşik gerilimine şarj edinceye kadar çıkış alçak tutulur. Schmitt tetiği, çıkış sinyalinin alçaktan yükseğe keskin bir şekilde geçmesini sağlar. Başlangıçta kaydediciye 0111 yükleyerek bütün darbeleri ters çevirebilirsiniz.



Şekil 4-29 Seri-giriş, paralel-çıkış kaydırma kaydedicisi ile yapılmış halka sayıcının çıkış dalgaşekilleri.

HALKA SAYICILARIN KULLANILIŞLARI

Halka sayıcıların basamaklı çıkış dalgaşekilleri onları birçok uygulamalar için kullanışlı yapmaktadır. Bazı mantık-kontrollu makinalarda, her çıkışın ardışıl olarak bir sonraki durumu veya işlemi yetkilendirdiği kontrol-durum sayıcıları olarak kullanılabilirler. Ayrıca, küçük artmalarla yani adımlarla dönen özel bir tip motor olan adımlı motor (stepping motor), adım adım milini döndürmek için bu ardışıl darbelere gerek duyar.



Şekil 4-30 Açılışta reset etme devresi

Halka sayıcılar ayrıca hesap makinaları ve bilgisayarlarda kullanılanlar gibi elektronik tuş takımlarının çıkışlarını kodlamakta kullanılabilir. Kodlama, bir anahtar-kapamalı sinyali hex, ASCII veya EBCDIC gibi bir standart karakter koduna çevirmek demektir. Şekil 4-31, standart bir 63 tuşlu klavyenin sekiz satır ve sekiz sütunlu bir matris olarak bir bağlanış şeklini ve bir halka sayıcının yardımıyla kodlanışını göstermektedir. 74199, Şekil 4-31 deki U1, bir halka sayıcı olarak bağlanmış bir 8-bitlik kaydırma kaydedicisidir. Başlangıçta, bir açılışta reset etme devresi ile buna 11111110 yüklenir. Bir harici 5 KHz lik tarama osilatörü kaydedicinin saat girişine bağlandığından, bir alçak darbe ardışıl olarak satır hatlarından aşağı iner. Herhangi bir anda yalnız bir satır alçaktır. Hiçbir tuşa basılmamışsa, +5 V a bağlı dirençler dolayısıyla bütün sütunlar yüksektir. Bir tuşa basılırsa, o tuşa ait satır ve sütun birbirine kısa devre edilir. Halka sayıcı o satıra bir alçak gönderdiğinde, bu alçak, tuşun basıldığı sütunda da görülecektir. 1N270 germanyum diyotlar, eğer aynı sütundaki iki tuş aynı anda basılmışsa, halka sayıcının iki çıkışının kısa devre olmasını engeller. Bu diyotların üzerindeki küçük gerilim düşümü *U3* ve *U7* nin $V_{IL,\,MAK}$ değerini aşmaz.

Hiçbir tuşa basılmamışsa, sekiz-girişli VE DEĞİL kapısı U7 nin bütün girişleri yüksek ve çıkışı alçaktır. Herhangi bir sütunda bir alçak görülür görülmez VE DEĞİL kapısının çıkışı yüksek olur. Ancak, kontak zıplamasından dolayı bu çıkış birçok defa yükselip alçalabilir. Bu çıkışın ilk alçaktan-yükseğe geçişi *U4* ü, çıkış darbe genişliği yaklaşık 10 ms ye ayarlanmış bir 74121 tek kararlı, tetikler. 74121 in *Q* çıkışı, tuş zıplaması duruncaya ve doğru kod çıkışa verilinceye kadar, *U9* üzerinden halka sayıcının saat girişini yasaklar. Eğer tuş bundan daha uzun basılı tutulursa, 7430 dan gelip *U8C* den geçen sinyal anahtar bırakılıncaya kadar halka sayıcının saatini yasaklar. Bu tuş kodunun, her tuşa basılışında defalarca çıkışa verilmesini önlemek için gereklidir.

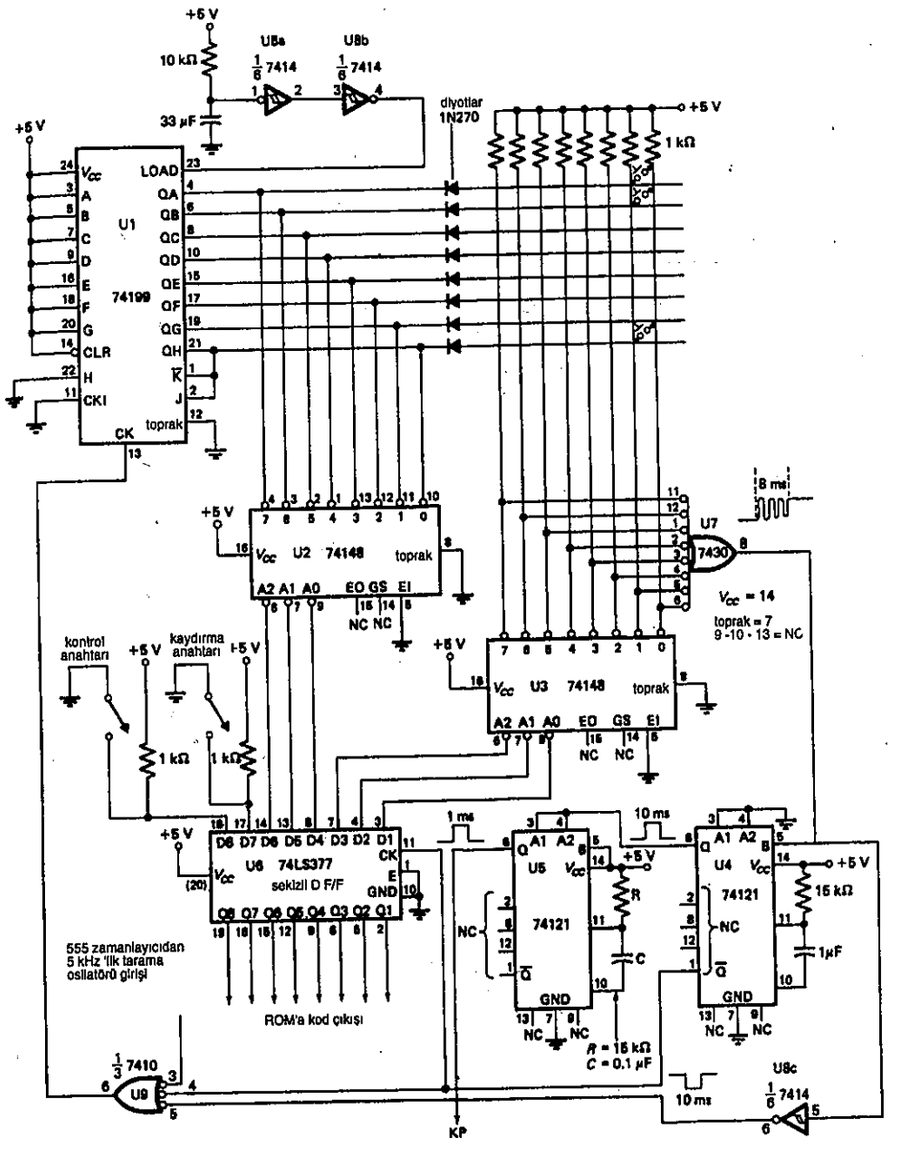
Tarama osilatörü durdurulduğunda, satır ve sütun kodları *U2* ve *U3* (iki 74148) tarafından işlenir. Bir 74148 bir 8-hattan-3-hatta öncelik kodlayıcıdır. EI girişini topraklayarak yetkilendirildiğinde, bu aygıt, üzerinde alçak olan en yüksek numaralı girişin 3-bitlik ters çevrilmiş ikilik kodunu çıkışa verecektir. Örneğin, eğer 74148 in 5. girişinde bir alçak var ise, çıkışlarda 5 in ikilik kodunun tersi olan 010 olacaktır. Eğer girişlerden ikisi alçak ise, sadece en yüksek numaralı (öncelikli) girişin ters çevrilmiş ikilik kodu çıkışa verilecektir.

Halka sayıcı bir tuşun basılmasıyla durdurulduğunda satırlardan biri alçak olacaktır. Bu satır numarasının 3-bitlik kodu *U2* nin çıkışına verilecektir. Benzer şekilde, tuş basıldığında sütunlardan biri alçak olacaktır. Bu sütun numarasının 3-bitlik kodu *U3* ün çıkışına verilecektir.

U2 den gelen 3-bitlik satır kodu, *U3* ten gelen 3-bitlik sütun kodu, ve kaydırma ve kontrol anahtarı çıkışları *U6* sekizli *D* flip-flopun *D* girişlerine bağlanmıştır. *U4* ten gelen saat yasaklama darbesinin yükselen kenarı bu kodları flip-flopların çıkışlarına yükler. Bu 8-bitlik kod ASCII veya başka bir standart koda çevrilmek üzere bir ROM a gönderilir.

İkinci bir 74121 *(U5)* pozitif ve negatif anahtar-basıldı yükleme sinyallerini üretir. Bunlardan birisi bir alıcı sisteme U6 nın çıkışlarında yeni basılan bir tuşun geçerli kodunun bulunduğunu bildirmek için kullanılabilir.

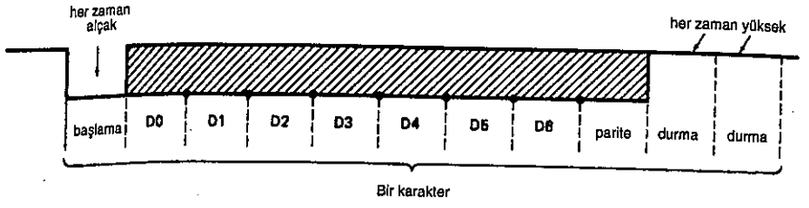
Sadece tuş anahtarlarına bağlantı gerektiren, General Instruments AY-5-2376 ve National Semiconductor MM5740 gibi, komple MOS LSI klavye kodlayıcıları hazır olarak bulunabilmektedir. Bu kodlayıcıların ilave bir özelliği de iki tuşa çok yakın zaman aralıklarında basılmış ise ikinci basılan tuşun kodunun saklanıp birincisi çıkışa verildikten sonra ikincisinin de çıkışa verilmesidir.



Şekil 4-31 Sekiz kademeli halka sayıcı kullanan klavye kodlama devresi.

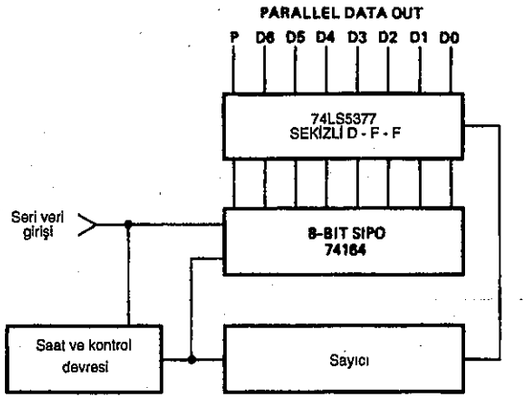
BİR SIPO NUN SERİDEN-PARALELE VERİ ÇEVİRİCİ OLARAK KULLANILIŞI

Bağlantı tellerinin sayısını azaltmak için, sayısal veri kelimeleri bir sistemden diğerine çoğunlukla seri şekilde gönderilir. Diğer bir deyişle, bir kelimenin bitleri bir tek hat üzerinden birer birer gönderilir. Şekil 4-32, ASR-33 gibi bir teletayp terminalinden gönderilen ASCII kodun seri formatını göstermektedir. Hiçbir karakter gönderilmiyorsa, hat sürekli yüksektir. Bir karakterin geleceğini alıcıya söyleyen başlama (start) biti alçaktır. Gösterildiği gibi, ASCII kodlanmış karakterin veri bitleri başlama bitini izler. Bundan sonra 1 parite biti ve her zaman yüksek olan 2 durma (stop) biti gelir. Durma bitleri karakterin gönderilmesinin tamamlandığını belirtir. Şekil 4-32 de gördüğünüz gibi, her karakter 11 bit zamanı alır. Teletayplardan seri veri göndermedeki standart oranlardan biri 110 bit/sn yani 110 baud (Bd) dir. Saniyede 110 bit, karakter başına gerekli 11 bite bölününce saniyede 10 karakter değerini verir.



Şekil 4-32 Bir teletayp için seri veri formatı.

Mikrobilgisayarlar ve diğer cihazlar genellikle verinin paralel şekilde girilmesini isterler. Şekil 4-33 te gösterildiği gibi, bir SIPO kaydırma kaydedicisi veri formatını seriden paralele çevirmek için kullanılabilir. Seri veri, kaydırma kaydedicisinin seri girişine uygulanmaktadır. Kontrol devresi başlama bitinin 1 den 0 a geçişini algılar ve veriyi kaydediciye kaydıran bir saat sinyalini kapıdan geçirir. Bir sayıcı, kaydırma darbelerinin sayısını tutar ve her sekiz kaydırma darbesi için bir *veri hazır (data available : DAV)* darbesi üreterek 8-bitlik *D* flip-floplara veriyi kilitler.



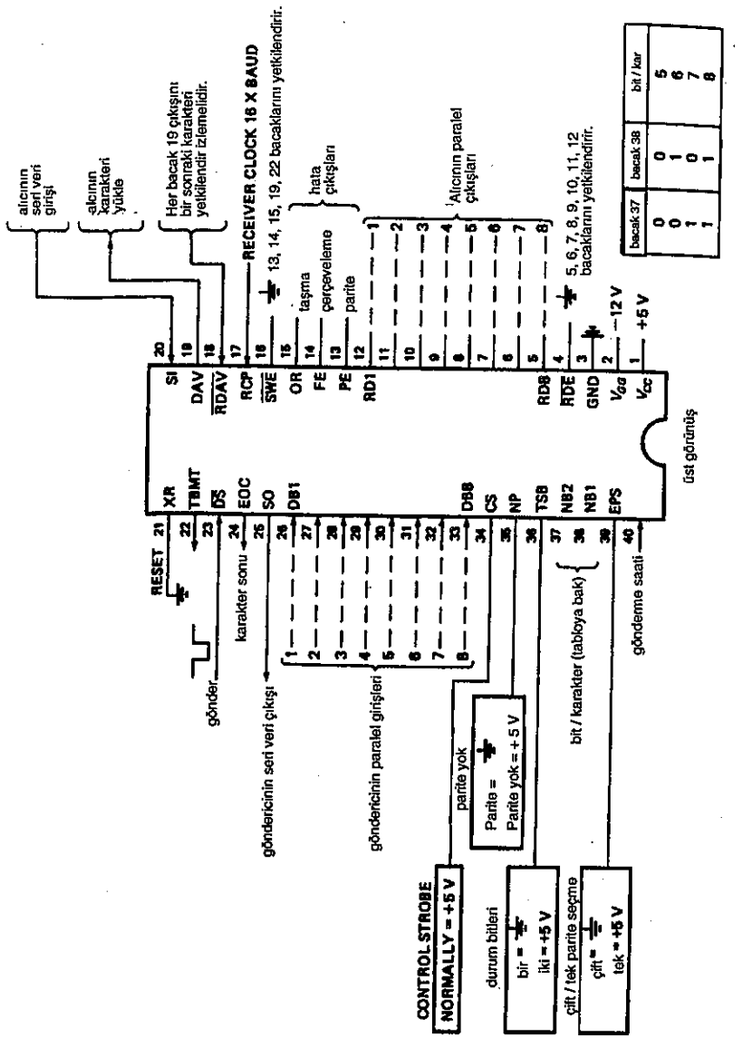
Şekil 4-33 SIPO kaydırma kaydedicisini kullanan seriden-paralele veri çevirici.

PARALEL-GİRİŞ, SERİ-ÇIKIŞ KAYDIRMA KAYDEDİCİLERİ

Paralel-giriş, seri-çıkış kaydırma kaydedicileri, isminden de anlaşıldığı üzere, kaydedici bir sayıyı paralel yüklemenize ve sonra kaydedicinin ucundan kaydırarak almanıza yani seri çıkışa imkân veren kaydedicilerdir. Bu tip bir kaydedicinin açık bir kullanım yeri paralel-formatlı veriyi seri-formatlı veriye çevirmektedir. Bir başlama biti, 7 veri biti, 1 parite biti ve 2 durma biti, bir 11-bitlik PISO kaydırma kaydedicisine yüklenip saat göndererek seri veri üretmesi sağlanabilir.

UARTLAR VE USARTLAR

Ayrık kaydırma kaydedicilerinden, seriden-paralele veya paralelden-seriye veri çeviricileri kurmanıza nadiren gerek vardır, çünkü General Instruments AY-5-1013 veya Signetics 2536 gibi MOS LSI aygıtları 40 bacaklı bir tek pakette, seri veriyi paralele ve paralel veriyi seriye çevirmek için gerekli bütün devreleri içermektedir. Bu LSI aygıtlara üniversal asenkron alıcı-vericiler (Universal asynchronous receiver-transmitter : UART) denir. Şekil 4-34, AY-5-1013 ün bacak diyagramını göstermektedir. Bu aygıt 30.000 baud yani bit/sn ye kadar olan hızlarda veri gönderebilir ve alabilir. İstenilen hızın 16 katı olan bir giriş saati gerektirir. İstenilen hızdan daha yüksek bir saat frekansı kul-



Şekil 4-34 AY-5-1030 üniversal asenkron alıcı vericinin *(UART)* bacak diyagramı. *(General Instruments.)*

lanılması salınım problemlerinden kaçınmak üzere her bit süresinin ortasında örnekleme yapılmasına imkân verir. Her karakter için 5 ile 8 arasında veri biti, istenirse bir parite biti ve tek veya çift parite, ve 1 veya 2 durma biti seçilebilir. 13 ve 15. bacaklardaki çıkış darbeleri gelen seri verilerde bir parite hatası, bir çerçeveleme hatası veya bir taşma hatası olup olmadığını belirtir. Bir çerçeve-

leme hatası, gelen verinin, UART ın kabul etmeye programlandığından daha farklı bir bit sayısı olduğunu belirtir. Bacak 4 ü yüksek yaparak paralel veri çıkışları 3-durumlu hale getirilebilir, böylece çıkışlar doğrudan bir çift yönlü veri taşıtına bağlanabilir ve istendiğinde yetkilendirilebilir. Bacak 23 teki gönder (send) girişi UART a bir sonraki seri karakteri göndermesini söyler. Bacak 19 daki çıkış, bir alıcı sisteme paralel çıkışlardaki verinin geçerli olduğunu söyler. Bacak 18 deki giriş, alıcı sistemin veriyi aldığını ve UART ın bir sonraki veri karakterini göndermekte serbest olduğunu UART a bildirir.

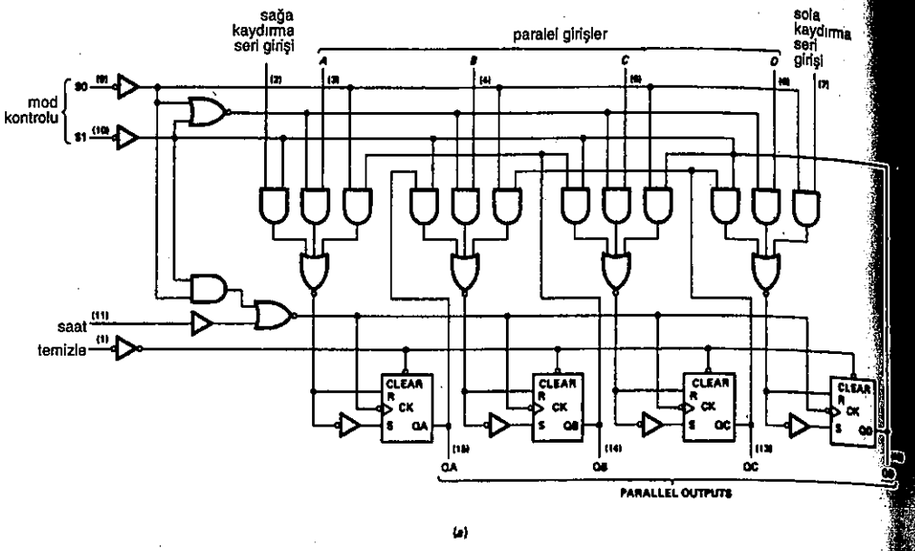
UART a benzer bir devrede üniversal senkron veya asenkron alıcı-vericidir (Universal synchronous or asynchronous receiver-transmitter : USART). Senkron haberleşmede karakterler sabit bir hızda gönderilir. Eğer veri karakterleri gönderilmiyorsa, sabit hızda karakter gönderimini sürdürmek için, USART *"boş" (null)* karakterler gönderir. UART lar ve USART lar üzerinde Bölüm 11 de daha fazla durulacaktır.

PARALEL - GİRİŞ, PARALEL - ÇIKIŞ KAYDIRMA KAYDEDİCİLERİ

Daha önce anlatıldığı gibi, bir paralel girişli kaydırma kaydedici, paralel giriş bacaklarına konan bir ikilik sayının iç kaydedicilerine yüklenmesine imkân verir. Bazı paralel yüklemeli kaydırma kaydedicilerinde, bu paralel veri girişi derhal paralel çıkışlarda görülür, fakat diğer bazılarında paralel yüklenmiş veriyi çıkışa göndermek için bir saat darbesi gerekir. Şekil 4-35 a bir 74194 üniversal kaydırma kaydedicisinin şemasını göstermektedir. Bu kaydedici, anlatılan dört tipten herhangi birisi olarak kullanılabilir. S0 ve S1 girişlerindeki programlamaya göre paralel-yüklenebilir, sola kaydırabilir, sağa kaydırabilir veya hiç bir şey yapmayabilir. Kapı devrelerini izleme bilgilerinizi tazelemek için her işlemi hangi *S0* ve *S1* giriş durumlarının gerçekleştirdiğini belirleyebilirsiniz. Sonuçlarınızı Şekil 4-35 *b* deki doğruluk tablosuyla karşılaştırın.

Şekil 4-36, bir 74194 kaydırma kaydedicinin bağımlılık gösterilişi mantık şemasını göstermektedir. Bu sembol S0 ve S1 girişlerinin yanındaki M harfiyle belirtildiği üzere mod bağımlılığını göstermektedir. Bu aygıt için mod S0 ve S1 girişlerindeki ikilik sayı ile seçilmektedir. Mod 3 te ($S1 = 1$ ve $S0 = 1$) Q çıkışlarına, A, B, C ve D girişlerindeki veriler yüklenir.

Bu durum, bu girişlerin yanındaki 3 ile gösterilmiştir. Mod 2 de (S1 = 1 ve $S0 = 0$) sola-kaydırma seri girişi aktiftir. Bu durum, bu girişin yanındaki 2 ile belirtilmiştir. Mod 1 de ($S1 = 0$ ve $S0 = 1$) sağa-kaydırma seri girişi aktiftir. Bu da bu girişin yanındaki 1 ile belirtilmiştir. Mod 0 da ($S1 = 0$ ve $S0 = 0$) girişlerin hiç birinin çıkış üzerinde etkisi yoktur, bu nedenle de hiçbir girişin yanına 0



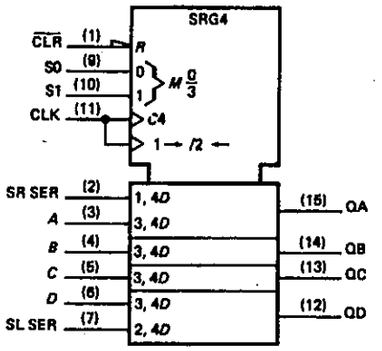
(a)

GİRİŞLER										ÇIKIŞLAR			
	MOD			SERİ		PARALEL							
TEMİZLE	S1	S0	SAAT	SOL	SAĞA	*A*	*B*	*C*	*D*	QA	QB	QC	QD
L	X	X	X	X	X	X	X	X	X	L	L	L	L
H	X	X	L	X	X	X	X	X	X	QA0	QB0	QC0	QD0
H	H	H	↑	X	X	*a*	*b*	*c*	*d*	*a*	*b*	*c*	*d*
H	L	H	↑	X	H	X	X	X	X	H	QAn	QBn	QCn
H	L	H	↑	X	L	X	X	X	X	L	QAn	QBn	QCn
H	H	L	↑	H	X	X	X	X	X	QBn	QCn	QDn	H
H	H	L	↑	L	X	X	X	X	X	QBn	QCn	QDn	L
H	L	L	X	X	X	X	X	X	X	QA0	QB0	QC0	QD0

H = Yüksek seviye (sürekli durum)
L = Alçak seviye (sürekli durum)
X = (önemsiz, geçişler dahil herhangi bir giriş)
↑ = Alçaktan yükseğe geçiş
QA0, QB0, QC0, QD0 = Belirtilen sürekli durum girişler oluşmadan önceki *QA, QB, QC* ve *QD* çıkışları
QAn, QBn, QCn, QDn = En son rahat geçişinden önceki *QA, QB, QC*n ve *QD* seviveleri mod kontrolları saat yüksekken değiştirilmelidir.

Şekil 4-35 74194 paralel-giriş, paralel-çıkış kaydırma kaydedicisi *(a)* Mantık şeması. *(b)* Doğruluk tablosu.

konulmamıştır. Daha alttaki saat girişi CLK nın yanındaki semboller, mod 1de aygıtın sağa doğru kaydırmayı saatin yükselen kenarında yaptığını ve Mod 2 de sola doğru kaydırmayı yine saatin yükselen kenarında yaptığını belirtmektedir. Daha üstteki CLK girişinin yanındaki *C4* ile belirtilen kontrol bağımlılığı ise 4 ile numaralandırılmış girişlerin, çıkışı saat sinyalinin yükselen kenarında etkileyeceğini göstermektedir. *R* girişi, eğer CLR sinyali aktif ise bütün çıkışların alçağa reset edileceğini belirtmektedir.



Şekil 4-36 74194 ün bağımlılık gösterilişi mantık şeması

PIPO KAYDIRMA KAYDEDİCİLERİNİN KULLANILIŞI

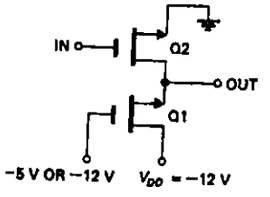
PIPO kaydırma kaydedicilerinin en çok kullanıldığı yer herhalde ikilik sayıların matematik işlemleridir. Bölüm 7 de, iki ikilik sayının bir seri toplama ve sağa kaydırma işlemleri ile nasıl çarpılabileceğini göstereceğiz. İkilik bölme de bir seri çıkarma ve sola kaydırma işlemleri ile gerçekleştirilebilir. Şimdilik, bir ikilik sayıyı bir basamak sağa kaydırmanın sayıyı 2 ile böldüğünü ve bir ikilik sayıyı bir basamak sola kaydırmanın sayıyı 2 ile çarptığını kendinize göstermeniz yeterli olacaktır. Örneğin, ikilik 111 (onluk 7) sayısı bir basamak sola kaydırılırsa 1110, yani 14_{10}, elde edilir.

MOS KAYDIRMA KAYDEDİCİLERİ

Çok kademeli uzun bir kaydırma kaydedicisini tek bir entegre paketine koymak için her kademe minimum yonga alanı kullanmalı ve minimum güç harcamalıdır. Yaklaşık 16 kademeden fazla uzunluklar için bunun anlamı bir çeşit MOS devresinin kullanılması demektir. Yer ve güç yönünden çok verimli bir çeşit, dinamik MOS devreleridir.

DİNAMİK MOS

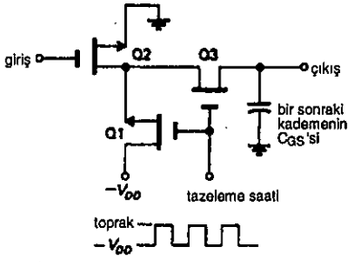
Kendinize dinamik MOS u tanıtmak için önce Bölüm 2 de anlatılan PMOS devrelerine geri dönüp bir hafıza tazelemesi yapın. Şekil 4-37 bir PMOS ters çeviriciyi göstermektedir. Negatif besleme gerilimine dikkat edin. Devredeki transistörler yalnız -E MOSFET lerdir, fakat basit olması için görüldüğü gibi çizilmişlerdir. *Q1* kutuplanmıştır ve *Q2* için aktif yük direnci görevini görür. Eğer giriş toprak yakınında bir gerilimde ise *Q2* kesimdedir ve çıkış *Q1* tarafından V_{DD} yakınında bir gerilime çekilir. Eğer giriş V_{DD} yakınında bir gerilimde ise *Q2* iletimdedir. *Q2* nin iletimdeki direnci *Q1* inkinden az yapıldığından, çıkış, toprak yakınında bir gerilime çekilecektir ki bu da PMOS la genelde kullanılan mantıkta bir alçaktır. Bu devre ters çevirici olarak iyi çalışır, fakat *Q2* iletimdeyken toprağa sürekli bir akım yolu olması gibi bir problemi vardır. Bunun anlamı, ters çeviricinin bu durumda sürekli güç harcadığıdır.



Şekil 4-37 PMOS ters çevirici. Negatif kaynağa dikkat edin.

Harcanan gücü azaltmanın bir yolu, Şekil 4-38 deki dinamik MOS ters çeviricide gösterilmektedir. Bu devrede, çıkış durumu gerçekte bir sonraki kademenin MOSFET girişinin kapı-kaynak arasındaki kapasitansında saklanır. Tazeleme saati (refresh clock), V_{DD} yakınında bir gerilim seviyesinde olduğunda hem *Q1* hem de *Q3* iletimdedir. *Q3* sadece kapalı bir anahtar

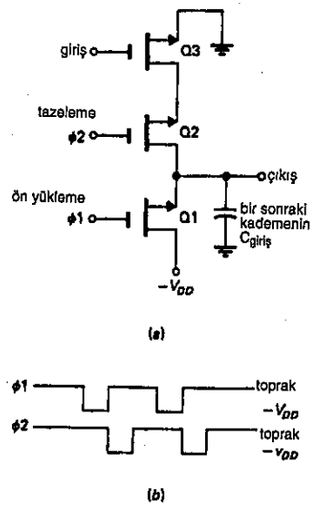
gibi davrandığından bir sonraki kademenin giriş kapasitansı C_{GS}, *Q1* ve *Q2* kavşağında çıkış gerilimine şarj eder. Tazeleme saati (refresh clock) toprağa döndüğünde, *Q1* ve *Q3* kesime gider. Çıkış seviyesi C_{GS} de tutulur ve şimdi *Q1* kesimde olduğundan hiç güç harcamaz. Ters çeviricinin girişi negatif mantık alçakta, yani toprak civarında, ise *Q2* kesimdedir ve tazeleme saatinin çıkıştaki C_{GS} yi yüksek durumunda ($-V_{DD}$) şarjlı tutması yeterlidir ki bu da çok az bir akım ve güç harcamasını gerektirir. Eğer giriş mantık yüksek, yani $-V_{DD}$ yakınında, ise *Q2* iletim için kutuplanmıştır. Fakat tazeleme saatinin dalgaşeklinden görebildiğiniz gibi *Q1* aktif yükü zamanın sadece % 50 sinde iletimde olacak ve akım geçirecektir. Bu, zamanın sadece % 50 sinde güç harcarken, C_{GS} kapasitörünü deşarj etmeye ve çıkışı mantık alçakta tutmaya yeterli olacaktır. Bu, Şekil 4-37 deki statik PMOS ters çeviriciye göre çok büyük bir gelişmedir. Bir saat üreteci gerekmektedir, fakat bu çok ciddi bir problem değildir, çünkü bir saat üreteci birçok kapıda kullanılabilir. Bu yapıdan **dinamik** diye bahsedilir, çünkü mantık durumları, 1 KHz den daha yüksek bir frekansta saat gönderilerek, sürekli tazelenmek zorundadır. 1 KHz in altında bir tazeleme saati, C_{GS} deki elektrik yükünün sızarak kaçmasına imkân verir ve bütün mantık seviyeleri geçersiz hale gelir.



Şekil 4-38 Tek - faz saatli dinamik MOS ters çevirici.

Kaydırma kaydedicilerinde kullanıldığında daha da az güç harcayan diğer bir dinamik ters çevirici yapısı Şekil 4-39a da gösterilmiştir. Bu yapı, Şekil 4-39 b de dalgaşekilleri gösterilen iki-fazlı saat gerektirmektedir. Yine çıkış mantık seviyesi bir sonraki kademenin girişindeki C_{GS} sinde saklanır. Bir tazeleme çevrimi *Ø1* önşarj saatinin *Q1* i iletime geçirip kapasitansı $-V_{DD}$ ye şarj etmesiyle başlar. *Q1* kesime geçirildikten sonra *Ø2* tazeleme saati *Q2* yi iletime geçirir. Eğer giriş yüksek ($-V_{DD}$) ise, *Q3* iletimdedir ve kapasitans negatif mantık alçak çıkışı vermek üzere *Q2* ve *Q3* üzerinden toprağa deşarj edilecektir. Eğer giriş alçak (0 V) ise, *Q3* kesimdedir. Böylece *Q2* iletime geç-

tiğinde, toprağa bir akım iletim yolu olmayacaktır ve kapasitans negatif mantık yüksekte ($-V_{DD}$) şarjlı kalacaktır. Bu devre daha az güç harcar, çünkü hiç bir zaman $-V_{DD}$ den toprağa doğrudan bir yol yoktur. Bu yapının tek dezavantajı, $C_{GIRIŞ}$ te tutulan mantık durumunu sürekli tazelemek için iki - fazlı saate gerek olmasıdır.



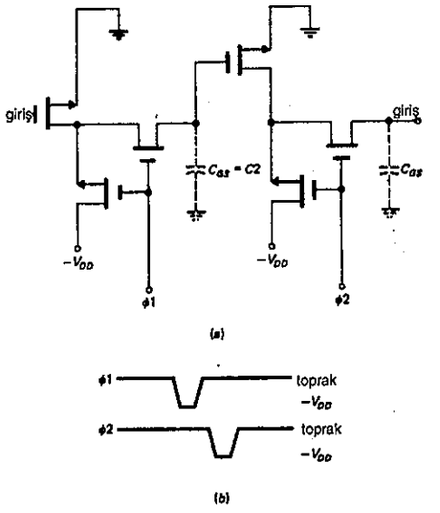
Şekil 4-39 İki-faz saatli dinamik MOS ters çevirici.
***(a)* Şema. *(b)* Saat dalgaşekilleri.**

Şekil 4-38 deki türden veya Şekil 4-39 *a* daki türden iki dinamik ters çevirici bir dinamik kaydırma kaydedici hücresi oluşturmak üzere birleştirilebilir, İki tek-faz ters çevirici kullanan bir dinamik kaydırma kaydedici hücresinin devresi, Şekil 4-40 a da ve dalgaşekilleri Şekil 4-40 *b* de gösterilmiştir. Ø1 saati, girişteki veriyi *C2* ye aktarır ve sonra Ø2 saati veriyi çıkışa aktarır. Bütün saat girişlerine sadece bir tek-fazlı saat bağlanmış olsaydı, girişteki veri, bir kaydırma kaydedicisinde olması gerektiği gibi her saat darbesiyle hücreden hücreye geçmek yerine bütün ters çeviriciler zincirini bir hamlede geçecekti. İki-fazlı saatler, her seferinde ters çevirici leri bir atlayarak yetkilendirip, bu yarış durumunu engeller.

National Semiconductor MM4027, tipik bir PMOS dinamik kaydırma kaydedicidir. Bu entegre +5 ve -12 V luk kaynaklarla çalışır ve 2048 kademe, yani hücre, içerir. Minimum çalışma frekansı 0°C da 200 Hz den 100°C da 20 KHz e kadar değişir. Bu 2048 kademenin ve ilgili devrelerinin, toplam güç harcaması, 1 MHz de çalışırken, 250 mW dan azdır.

STATİK MOS KAYDIRMA KAYDEDİCİLERİ

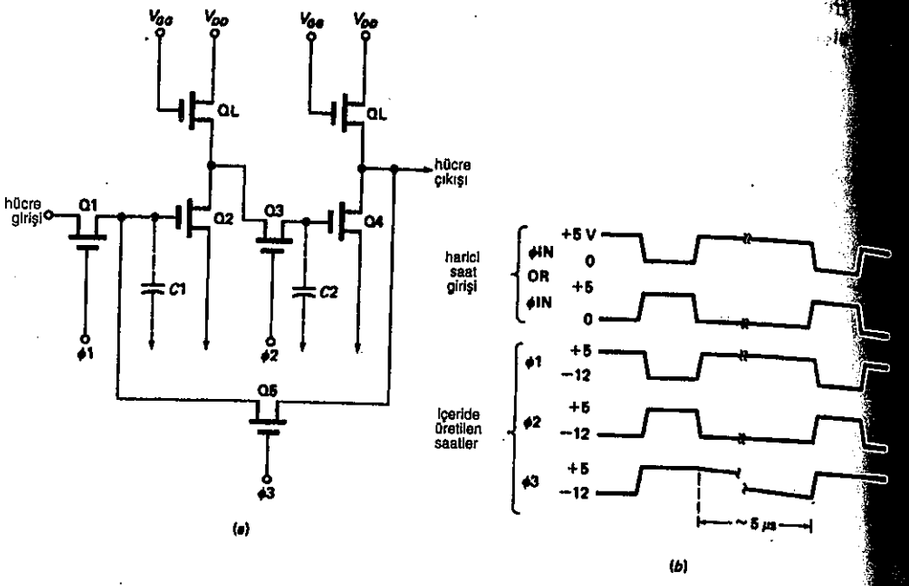
Önceki kısımda anlatılan dinamik kaydırma kaydedicilerinin alçak güç ve entegre paketi başına çok kademe avantajları vardı, fakat minimum çalışma frekansı dezavantajları bulunmaktadır. Alçak-frekans uygulamalarında, bir statik kaydırma kaydedici kullanılmalıdır. Şekil 4-41 *a* da gösterilen statik



Şekil 4-40 Dinamik MOS kaydırma yazmacı hücresi.
***(a)* Şema. *(b)* Saat dalgaşekilleri.**

MOS kaydırma kaydedici hücresi, iki statik ters çevirici kullanmaktadır (*Q2* ve *Q4*). Şekil 4-41 b de gösterildiği gibi entegre üzerindeki saat üreteçleri bir tek-faz saat girişinden *Ø1*, *Ø2* ve *Ø3* saatlerini üretir. *Ø1* saati, veriyi birinci kademenin çıkışına kaydırır ve *Ø2* saati de onu ikinci kademenin çıkışına kaydırır. Saat darbeleri arasında, *Ø2* saati alçak ve *Q3* iletimde olduğundan, birinci kademenin çıkışı, ikinci kademenin girişine doğrudan bağlanmış olur. Ancak, *Q1* kesimde olduğundan, saat darbeleri arasında birkaç mikrosaniyeden fazla bir süre geçerse, *C1* deşarj olabilir ve belki de saklanan mantık seviyesini kaybedebilir. Bu kayıp *Q5* tarafından önlenmektedir. Eğer saat darbeleri arasında 5 µs den fazla zaman geçerse, dahili olarak üretilen *Ø3* saati *Q5* i iletime geçirir. Böylece, *C1* üzerindeki yükü eski haline getirmek üzere, *Q5*, çıkıştan pozitif geribesleme sağlar. Alçak frekanslarda, bu devre çapraz-kuplajlı ters çeviricilerden yapılmış bir kararlı mandal gibi davranır. Statik devre, daha fazla güç gerektirir ve daha fazla yonga alanı kullanır; ve bu nedenle de

bir entegre paketine daha az sayıda kademe konulabilmektedir. Avantajı ise doğru akıma kadar bütün frekanslarda çalışabilmesidir. Tipik bir MOS statik kaydırma kaydedici, 1024-bitlik Signetics 2533 tür.

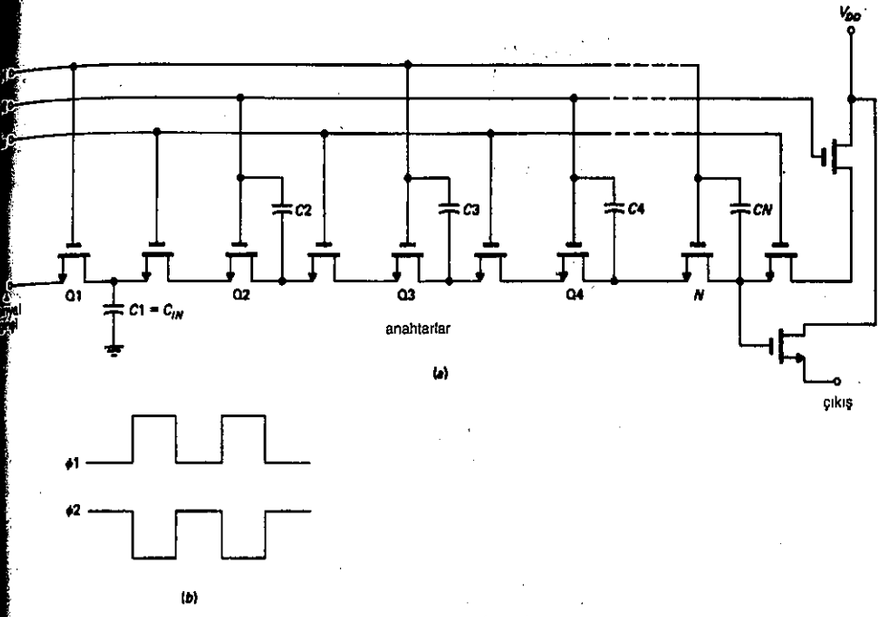


Şekil 4-41 Statik MOS kaydırma kaydedici hücresi. (a) Şema. (b) Saat dalgaşekilleri.

KOVA GRUBU (BUCKET BRIGADE) KAYDIRMALI KAYDEDİCİLERİ

Sayısal sinyallerin yanısıra analog sinyalleri de geciktirmede kullanılabilen diğer bir seri-giriş, seri-çıkış MOS kaydırma kaydedici kova grubu tipidir. İsmini eski bir yangın söndürme tekniğinden almaktadır. Şekil 4-42, bir PMOS kova grubunun devresini ve zamanlama diyagramını göstermektedir. Yapı, akaçtan kaynağa bağlanmış bir MOSFET ler zincirinden ibarettir. *P*-kanal E-tipi MOSFET lerin altlıklarının hepsi içeriden toprağa bağlanmıştır. *Ø1* saati V_{DD} ye ulaştığında, *Q1* giriş FET i giriş gerilimini örnekler ve bu gerilimi $C_{GIRIŞ}$ te saklar. Kapıları V_{BB} ye bağlı isimsiz MOSFET ler kutuplamayı sağlar. *Ø1* toprağa geri döndüğünde, bu gerilimin değeri $C_{GIRIŞ}$ te tutulur.

Ø2 saati V_{DD} ye indiğinde, *Q2* iletime geçer ve $C_{GIRIŞ}$ üzerindeki yükün yaklaşık % 99,9 u *Q2* nin kapı-akaç kapasitansına çekilir. *Ø1* in bir sonraki negatif saat darbesi bu bir kova yükü, *Q3* ün kapı-akaç kapasitansına kaydıracak ve girişin yeni bir örneğini $C_{GIRIŞ}$ e koyacaktır. Her saat darbesi, gerilim örneğini, yani yük kovasını, grupta bir kademe kaydırır. Zincirin sonundaki bir kaynak-izleyici, bir alçak-empadanslı çıkış verir. Veri, küçük kapı-akaç kapasitansı üzerinde saklandığından, eğer devre belli bir minimum frekansın üstünde bir saatle çalıştırılmazsa, yük sızarak kaçacak ve veri kaybolacaktır.



Şekil 4-42 MOS kova grubu. *(a)* Şema. *(b)* Saat dalgaşekilleri.

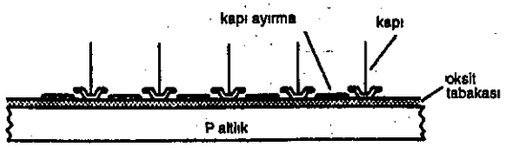
Bu devrede kaydırılabilecek gerilim, sayısal kaydırma kaydedicilerindeki gibi yüksek ve alçakla sınırlı değildir. Bu nedenle, ses sinüs dalgası gibi analog sinyaller de örneklenip bir kova grubu aygıtında kaydırılarak geciktirilebilir. Gecikme miktarı, saat frekansı ve kademe sayısı ile belirlenir. Çıkış dal-

gaşekli bölümlenmiş osiloskop görüntüsüne benzer şekilde noktalardan oluşmaktadır. Bir alçak-geçiren filitre çıkış dalgaşeklini, noktaları birleştirecek şekilde, yuvarlayarak yumuşatır. Eğer çıkış sinyalinin bir kısmını girişe geri beslerseniz, bir yankı (eko) etkisi oluşturursunuz. Şekil 4-24 deki gitar yükselteci yankı biriminin A/D çeviricisi, kaydırma kaydedicileri ve D/A çeviricisi, Reticon Corporation SAD-1024 gibi bir 1024 -elemanlı kova grubu ile değiştirilebilir. SAD-1024 ün minimum saat frekansı 1,5 KHz ve maksimum saat frekansı 1 MHz dir. Ses geciktirme uygulamalarında, bu aygıt, sinyalin tepeden tepeye olan değeri 75 mV dan az ise en az bozulmayı vermektedir.

YÜK-KUPLELİ ELEMANLAR (CHARGE-COUPLED DEVICES : CCD)

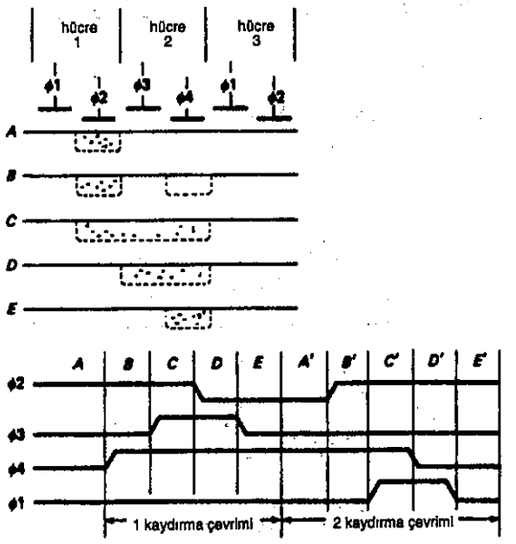
Yük-kupleli aygıtlar da, yani *CCD* ler de, seri-giriş, seri çıkışlı MOS kaydırma kaydedicileri olarak yapılmaktadırlar, fakat bunların yapıları kova gruplarınınkinden daha basit ve daha derli topludur (sıkıştırılmıştır).

Şekil 4-43, bir *CCD* kaydırma kaydedicisinin bir kısmının yapısını göstermektedir. Gördüğünüz gibi, yapı basitçe bir *P* altlık, bir yalıtkan oksit tabakası ve ayrılmış kapılardan oluşmaktadır. Yanyana olan kaydırma kaydedici kısımları, sürekli bir yapı oluşturmak üzere aynı *P* altlık üzerine konulmuşlardır. Veri bir *CCD* de yük olarak saklanır. *D* altlık içinde bir potansiyel kuyusu, yani yük saklama bölgesi, yaratılabilmektedir. Şekil 4-44 te, A anında $\varnothing 2$ kapısındaki pozitif bir gerilim, enjekte edilmiş bir elektron yükünü $\varnothing 2$ kapısının altındaki bölgede esir tutar. Bu yük paketi, Şekil 4-44 te gösterilen bir dizi saat darbesiyle bir sonraki hücreye kaydırılabilir. Önce $\varnothing 4$ saati, B zamanı süresince $\varnothing 4$ kapısını pozitif yaparak altında boş bir kuyu oluşturur. Sonra C süresinde $\varnothing 3$ saati $\varnothing 3$ kapısını pozitif yaparak bu boş kuyu ile $\varnothing 2$ kapısı altındaki dolu kuyu arasında bir bağ oluşturur. Sonra da *D* süresinde $\varnothing 2$ saati alçağa düşer. Yükün tamamı $\varnothing 3$ ve $\varnothing 4$ kapı bölgelerine çekilir. E süresinin başlangıcında, $\varnothing 3$ saati alçağa düşer. Bu, bütün yükü $\varnothing 4$ kapısı altındaki bölgeye gitmeye zorlar ve hücre 1 den hücre 2 ye yerinin kayması tamamlanır. Bu süreç ilk başta çok karmaşık görülebilir, fakat gerçekte çok basittir ve daha az basamak kullanılması halinde oluşabilecek yarış durumuna engellemek için gereklidir. Yük paketinin hücre 2 deki $\varnothing 4$ altındaki bölgeden hücre 3 ün $\varnothing 2$ altındaki bölgesine nasıl kaydırıldığını göstermek için, Şekil 4-44 teki dalgaşekillerinin gerisinden faydalanabilirsiniz.



Şekil 4-43 MOS yük-kupleli aygıtın yapısı.

CCD dinamiktir ve belli bir minimum hızda kaydırılması gerekmektedir. Aksi halde, boş potansiyel kuyuları yavaş yavaş *P* altlığında sıcaklıkla serbest kalan elektronlarla dolar. Bu, saklanmış veriyi değiştirdiği için istenmez. *CCD* nin girişindeki bir dahili yükselteç, giriş verisine uygun olarak yük enjekte eder.

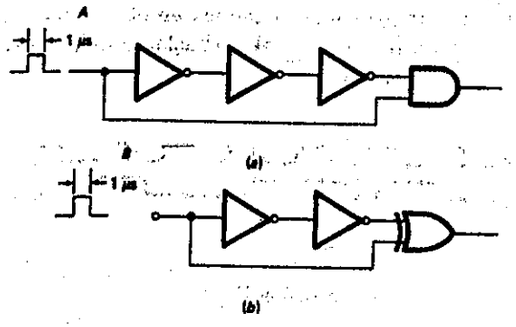


Şekil 4-44 CCD kaydırma sırası.

CCD ler, sayısal veya analog gecikmeler için ve Bölüm 5 te anlatılan seri veri bellekleri olarak kullanılırlar. *CCD* lerin diğer bir ilginç uygulaması televizyon kameralarında ışığa-duyarlı görüntü algılayıcıları olarak kullanılmalarıdır. Bir görüntü algılayıcı yapmak için yüzlerce yük-kupleli kaydırma kaydedici aynı yonga üzerine paralel olarak konulurlar. Her ardışıl kapının altına bir fotodiyot katkılanır. Altlarında fotodiyotlar olan bütün kapılar pozitif yapıldığında potansiyel kuyuları oluşturulmuş olur. Bir kamera mercek sistemi *CCD* dizisinin üzerine bir görüntü düşürür. Fotodiyot üzerine düşen ışık her kuyuya o ışığın şiddetiyle orantılı bir yük koyar. Bu yükler, bir televizyon resminin nokta nokta değerlerini vermek üzere sırayla satır satır kaydırılabilir. Işıksız yük-kupleli kaydırma kaydedicileri ile ışıklı yük-kupleli kaydırma kaydedicileri ardarda konarak çözünürlük iyileştirilebilir. Görüntü verileri ışıklı kaydediciden ışıksız kaydediciye paralel olarak kaydırılır ve ışıksız kaydediciden seri olarak kaydırılıp alınır. *CCD* görüntü algılayıcıları çok daha az yer almaktadır ve geleneksel boşluklu - tüp görüntü algılayıcılarından çok daha sağlamdır. Çözünürlükleri de boşluklu - tüplerinkine hızla yaklaşmaktadır.

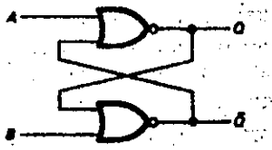
GÖZDEN GEÇİRME SORULARI VE PROBLEMLER

1. Sayısal devrelerle çalışırken niçin zamanlama diyagramları önemlidir?
2. Şekil 4-45 teki devrelerin herbiri için çalış dalgaşekillerini çizin.



Şekil 4-45

3. Bir mandal set edildiğinde Q çıkışının durumu nedir?
4. Şekil 4-3 a daki çapraz-bağlı VE DEĞİL kapıları mandalının iki önemli problemi nedir?
5. Şekil 4-46 daki devrenin doğruluk tablosunu kurun. Hangisinin SET girişi, hangisinin RESET girişi olduğunu belirlemek için doğruluk tablosunu kullanın.



Şekil 4-46

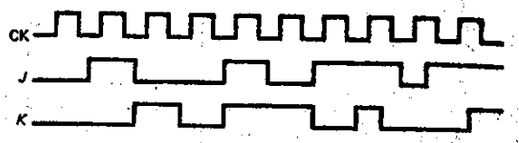
6. Bir D mandalının şematik sembolünü gösterin. Aktif yüksek yetki girişi olan bir D mandalının doğruluk tablosunu zamanlama-diyagramı şeklinde çizin. Veri gerçekte ne zaman çıkışa kilitlenir?
7. Bir *D* mandalı ile bir *D* flip-flop arasındaki ana fark nedir?
8. Şekil 4-47 de gösterilen saat ve *D* girişleri tarafından *(a)* bir *D* mandalının ve *(b)* bir *D* flip-flopun (pozitif-kenar-tetiklemeli) *Q* çıkışında oluşacak dalgaşekillerini gösterin. *Q* nun alçaktan başladığını kabul edin.



Şekil 4-47

9. Bir flip-flop için kurma zamanını ve tutma zamanını tanımlayın.
10. Bir *JK* flip-flopun çıkış doğruluk tablosunu kurun.

11. Şekil 4-48 de gösterilen *J*, *K* ve saat girişlerinin dalgaşekilleri verildiğine göre, bunların bir pozitif-kenar-tetiklemeli *JK* flip-flopta, bir negatif-kenar-tetiklemeli *JK* flip-flopta ve seviye-saatli ana-uydu *JK* flip-flopta oluşturacağı *Q* çıkışı dalgaşekillerini gösterin. *Q* nun başlangıçta alçak olduğunu kabul edin.



Şekil 4-48

12. Flip-flop devrelerinin sık karşılaşılan iki problemi nedir?

13. a. Bir *JK* flip-flopun nasıl bir *T* flip-flopa çevrildiğini gösterin.
    b. Bir *D* flip-flopun nasıl bir *T* flip-flopa çevrildiğini gösterin.

14. 1 KHz lik saati olan bir *T* flip-flopun saat ve çıkış dalgaşekillerini çizin.

15. 1 KHz saat girişli bir 4-bitlik ikilik sayıcının saat ve çıkış dalgaşekillerini çizin. Her sayımı ikilik değeri ile gösterin. Her çıkışın frekansını hesaplayın. (Flip-flopların, saat girişinin negatif kenarında durum değiştirdiğini kabul edin.)

16. Giriş frekansı 10240 Hz olan bir sekiz-kademeli ikilik bölücünün son çıkış frekansı nedir?

17. 8-bitlik, 12-bitlik ve 16-bitlik ikilik sayıcıların maksimum sayımları nedir?

18. Bir mod-10 (yani onluk) sayıcı, bir mod-12 sayıcı ve bir mod-13 sayıcıya çevrilmiş bir 7493 ün devrelerini gösteriniz. Her birindeki *QD* çıkış dalgaşeklini gösteriniz.

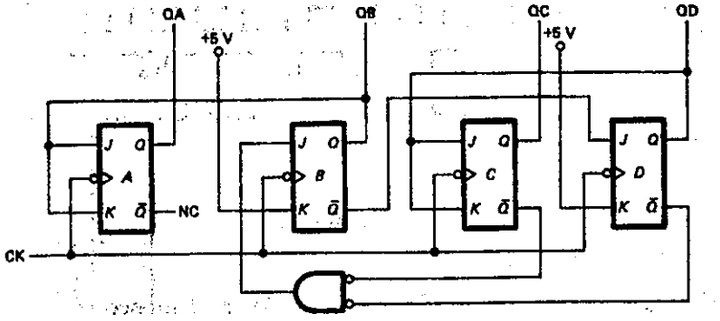
19. Bir 4-bitlik ikilik sayıcıda aşağıdaki durumların kodunu çözmek için kullanacağınız mantık kapı devresini çiziniz:

    a. Durum 4
    b. Durum 9
    c. Durum 12

20. Basit bir sayısal saatin blok diyagramını gösterin.

21. Dalgacık, yani asenkron, sayıcıların iki ana problemi nedir? Senkron sayıcılar bu problemleri nasıl en aza indirger?

22. *JK* flip-flopları ve sayısal geribesleme kullanan Şekil 4-49 daki devrenin *Q* çıkış sayım dizisini belirleyin. Başlangıçta bütün *Q* ların alçak olduğunu kabul edin.



Şekil 4-49

23. Bir mod-11 sayıcıya çevrilmiş bir 74193 ün devresini gösterin. Ödünç çıkışı darbesi de dahil olmak üzere bu mod-11 sayıcının zamanlama diyagramını çizin.

24. Asenkron temizleme terimiyle ne anlatılır?

25. Bir ROM u ardışıl olarak adreslemekte kullanılan bir sayıcı niçin senkron olmalıdır?

26. Şekil 4-23 teki frekans sayıcıya bakın.

    a. Kristal osilatörün ve onluk bölücüler zincirinin amacı nedir?
    b. Bölücü zincirden sonraki *JK* flip-flopun amacı nedir?
    c. Gösterilebilecek en yüksek sayım nedir?
    d. Ana sayıcı zincirinin her çıkışına niçin *D* flip-floplar konulmuştur?
    e. Flip-floplarn kilitleme darbeleri ve ana sayıcı reset darbeleri nasıl üretilmektedirler?
    f. Alçak frekansları doğru olarak ölçmek için niçin bir frekans sayıcıda 10 sn lik sayım penceresine ihtiyaç vardır?

27. Şekil 4-25 teki frekans sayıcıda kullanılana benzer bir çoğullamalı göstergenin avantajları nelerdir? Şekil 4-25 teki frekans sayıcının çoğullamalı göstergesinde, en önemli basamaktaki verinin o basamakta nasıl gösterildiğini anlatın.

28. Bir öğrenci tarafından kurulan bir çoğullamalı göstergeli frekans sayıcıda basamakların sırası bozuktur. Bunun muhtemel sebebi nedir?

29. Şekil 4-25 tekine benzer bir çoğullamalı göstergeli frekans sayıcı daha önce çalışmaktaydı. Aşağıdaki belirtilerin herbiri için mümkün bir sebebi açıklayın.

    a. Yeni bir frekans girişe verildiğinde görüntüdeki sayım hiç değişmemektedir.

    b. Görüntüdeki basamaklardan birisi parlak olarak yanmaktadır ve hep 8 göstermektedir. Diğer basamaklar zayıf yanmakta ve doğru sayıları göstermektedir.

    c. Bir giriş frekansı için en küçük değerlikli basamaklar doğrudur, fakat en büyük değerlikli üç basamak yanlıştır.

30. Bir önölçekleyici nedir?

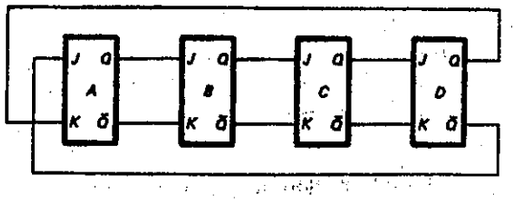
31. Bir kaydedici nedir?

32. *D* flip-floplarıyla yapılmış bir 4-bitlik seri-çıkış kaydırma kaydedicisinin devresini gösterin. *D* mandalları için kullanılamaz? İpucu : *D* flip-flopuyla *D* mandalları arasındaki farkı hatırlayın.

33. Bir 200 kademeli SISO kaydırma kaydedicisine 10 KHz lik saat uygulanmıştır. Veriler bu kaydediciden geçerken ne kadar gecikecektir?

34. *D* flip-floplarıyla yapılmış 4-bitlik bir halka sayıcının devresini gösterin. Sayıcı başlangıçta 1000 ile yüklenmişse oluşacak zamanlama diyagramını gösterin. Ayrıca başlangıçta 1100 ve 0111 yüklenmesi durumunda oluşacak çıkışları gösterin.

35. JK flip-floplarla yapılmış enteresan bir kaydırma kaydedici sayıcının şeması Şekil 4-50 de gösterilmiştir. Buna bazan bükülmüş halka sayıcı (twisted ring counter) veya Johnson sayıcı da denilmektedir. Bütün *Q* çıkışlarında 0 larla başlayarak, çıkış dalgaşekillerini çizin. Bu sayıcının avantajı, her seferinde sadece bir flip-flop değiştiği için, bütün durumların kodu istenmeyen darbeleri oluşmadan çözülebilmektedir.



Şekil 4-50

36. Şekil 4-31 de gösterilen klavye kodlayıcısının zamanlama diyagramını çiziniz.

37. ASR-33 gibi bir teletayp tarafından gönderilen 7-bitlik ASCII kodunun seri formatını gösteriniz. Başlama ve durma bitlerini de içeriniz.

38. UART, USART ve baud oranını tanımlayın.

39. Dinamik MOS ile statik MOS arasındaki farklılıklar nelerdir?

40. Bir MOS kova grubu kaydırma kaydedicisinin çalışmasını anlatın.

41. Bir CCD kaydırma kaydedicisinin çalışmasını anlatın ve kova grubu ile karşılaştırın. CCD lerin bazı kullanılış yerleri nelerdir?

REFERANSLAR / BİBLİYOĞRAFYA

Texas Instruments Elemanları, *Bipolar Microcomputer Components Data Book*, 3. Baskı. Dallas, 1981.

Texas Instruments Elemanları, *TTL Data Book for Design Engineers*, 2. Baskı. Dallas, 1981.

Blakeslee, Thomas R.: *Digital Design with Standard MSI& LSI*, Interscience-Wiley, New York. 1975.

MOS *Databook*, National Semiconductor Corporation, Santa Clara, CA. 1980.

CMOS *Databook*, National Semiconductor Corporation, Santa Clara, CA. 1978.

Logic *Databook*, National Semiconductor Corporation, Santa Clara. CA. 1981.

Garen, Eric R.: "Charge-Transfer Devices-Part 1: The Technologies", *Computer Design*, "Around the IC Loop", cilt. 16, no. 11, Kasım 1977, sayfa 146-152.

"Principles of Analog Discrete-Time Signal Processing Devices", *Uygulama Notu*, no. 102, Reticon Corporation, Sunnyvale, CA, 1975.

Intersil ICM 7226 A/B 10 MHz Universal Counter System, Veri sayfası. Intersil, Inc., Cupertino, CA, Ekim 1978.

Intersil Timer / Counter Applications Brochure, Intersil, Inc., Cupertino. CA.

General Instrument Corporation-Microelectronics: AY: 5-1013 UAR/T *Universal Asynchronous Receiver/Transmitter*, Veri sayfası, General Instrument Advance Nitride Technology (GIANT), Mart 1974.

BÖLÜM 5

OKU - YAZ BELLEKLER

Bilgisayarlar, mikroişlemciler ve daha bir çok cihaz, sayısal verileri ve ikilik-kodlanmış komutları saklamak için belleğe gereksinim duyarlar. Bölüm 3 te anlatılan yalnız-okunur-bellekler kullanılanların bir türüdür. ROM ların avantajarı uçucu olmamalarıdır. Bu, güç kesildiğinde de verilerin ROM matrisi üzerinde saklı kalması demektir. Bu nedenle, güç tekrar verildiğinde geçerli veriler hazır bulunur. Ancak ROM ların bir dezavantajı vardır. Bir plak gibi, bir defa programlandılar mı saklanan verileri değiştirmek çok zor, hatta imkânsızdır. EPROM lar ve EEPROM lar silinip yeniden yazılabilir, fakat bu çok kolayca ve normal devre çalışma hızlarında yapılamaz.

Bu bölümde, içine kolayca yazılabilen, okunabilen ve normal devre çalışmasının bir parçası olarak gerektiği kadar sayıda yeni verilerin tekrar tekrar anında yazılabildiği belleklerden bahsedeceğiz.

Bu bölümü bitirdikten sonra, şunları yapabileceksiniz :

1. Kaydedici, akümülatör, kaydedici dosyası, RAM, MBM ve Josephson kavşak belleğini tanımlayabileceksiniz.
2. Bir RAM ın matris yapısını ve adreslemesini anlatabileceksiniz.
3. Erişim zamanını açıklayıp, bir RAM ın veri sayfası verildiğinde, yazma ve okuma işlemlerinin önemli zamanlama parametrelerini bulabileceksiniz.
4. Statik ve dinamik MOS RAM arasındaki farkları söyleyebileceksiniz.
5. 16 adres bitinin bir yongaya sadece 8 bacak üzerinden nasıl yüklenebileceğini anlatabileceksiniz.
6. Birçok bilinen sayısal veri saklama yöntemlerinin avantaj ve dezavantajlarını listeleyebileceksiniz.
7. Manyetik teyp ve diskler üzerine veri saklanılışını anlatabileceksiniz.
8. Basit bir mantık analizörünün blok şemasını çizebileceksiniz.
9. Kullanılan dört mantık-analizörü gösterge formatını ve kullanılışlarını anlatabileceksiniz.

RASTGELE ERİŞİLEBİLİR BELLEK (RANDOM ACCESS MEMORY : RAM)

Bir flip-flop 1 bitlik bir oku-yaz belleğidir. Girişine bir veri biti uygulayarak ve bir saat darbesi kullanarak içine yazılabilir. Veri biti Q çıkışından oku-

nabilir. Saklanmış veri, girişe yeni bir veri uygulayıp bir saat darbesi göndererek kolayca değiştirilebilir. Bu tip bir bellek uçucudur, çünkü güç kesilince veriler kaybedilir. Yeniden güç verildiğinde çıkışın durumunun ne olacağı önceden bilinemez.

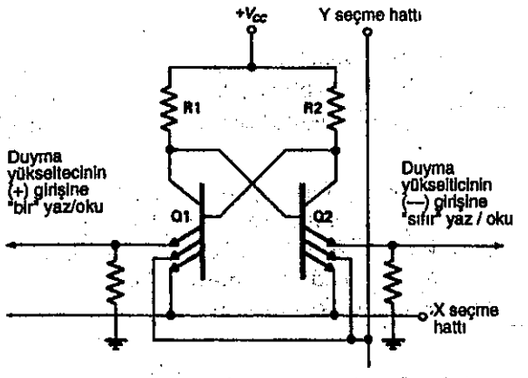
Kaydedici adı verilen bir flip-floplar gurubu, bir ikilik paralel veri kelimesini saklamak için kullanılabilir. Bir kelimeden daha fazlasını saklamak için birçok kaydedici kullanabilirsiniz. TTL 74170 4 x 4 kaydedici dosyası, 4-bitlik 4-kelimeyi saklamak üzere düzenlenmiş 16 flip-flop içerir. Okunmak veya yazılmak istenen kaydedici iki adres girişi ile seçilir.

TTL RAMLAR

Standart TTL *D* flip-floplar, tek bir entegre paketi içine birkaç taneden daha fazla kaydedici konulmasına izin vermeyecek kadar çok yonga alanı kullanır ve fazla güç (ısı) harcar. Bu nedenle, daha fazla bit saklamak için, Şekil 5.1 deki bipolar tipindekiler gibi, basitleştirilmiş flip-floplardan yapılmış *X* - *Y* matrisi kullanılır. Elde edilen matrise, yani dizileme, ratsgele-erişilebilir-bellek (random-access-memory : RAM) denir. Bu isim, matristeki herhangi bir yere, başka bir yeri etkilemeden, okumak veya yazmak için erişmek mümkün olduğundan verilmiştir.

YAPI

Şekil 5.1 deki RAM hücresi aşağıdaki şekilde çalışır. *X* seçme hattı, matrisin bir satırındaki bütün hücrelerin *X* seçme girişlerine bağlıdır. *Y* seçme hattı, matrisin bir sütunundaki bütün hücrelerin *Y* seçme girişlerine bağlıdır. *Q1* ve *Q2* çapraz bağlı ters çeviriciler olduklarından, birisi iletimdeyken diğeri mutlaka kesimdedir. *X* ve *Y* seçme hatları normalde alçaktır, bu durumda iletimde olan transistör üzerindeki akım *X* ve *Y* seçme hattı üzerinden toprağa akar. Bir hücre, o hücreyi adresleyen *X* ve *Y* seçme girişleri yüksek duruma çekilerek seçilir. Bu *X* ve *Y* emitörlerini yetkisiz kılar ve iletimdeki transistörden geçen akımın, duyma yükselteç emitörlerinin birinden akmasına sebep olur. Bir duyma yükselteci, akımın *Q1* emitöründen mi yoksa *Q2* emitöründen mi aktığını algılar. Gösterilen devrede, eğer *Q1* iletimde ve *Q2* kesimde ise hücreye bir 1 saklanmıştır. Hücrenin durumu, *Q1* duyma/yazma emitörüne bir yüksek darbe göndererek, 0 a değiştirilebilir. Bu *Q1* i kesime götürür. *Q1* kesimdeyken, *Q2* yi iletime geçirecek baz akımını şimdi *R1* sağlar. *Q2* iletimde olduğu sürece, kollektörü alçaktır ve *Q1* kesimde tutulur. *Q2* yi kesime götürmek için *Q2* duyma/yazma emitörüne bir yüksek darbe gönderilerek yeniden bir 1 yazılabilir.

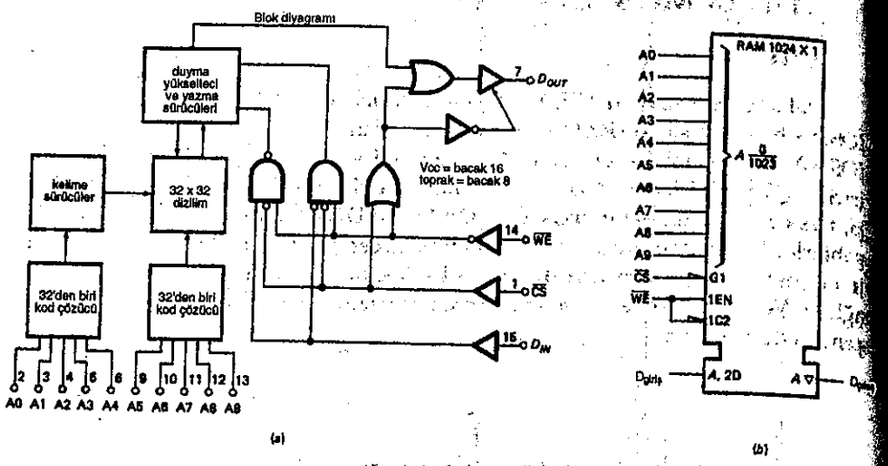


Şekil 5.1 TTL RAM hücresi.

Şekil 5.2a, bir Motorola MCM 93425 TTL RAM ın blok şemasını göstermektedir. Aygıtın, 1024 biti saklayabilen 32 x 32 lik bir matrisi vardır. A0 ile *A4* arasındaki hatlardaki ikilik adres, matrisin 32 satırından birini seçer. A5 ile *A9* arasındaki hatlardaki ikilik adres de 32 sütundan birisini seçer. Eğer $\overline{CS}$ alçak ve $\overline{WE}$ yüksek ise seçilen satır ve seçilen sütunun kesişme noktasında saklanmış olan veri $D_{ÇIKIŞ}$ a gönderilir. Eğer $\overline{CS}$ alçak ve $\overline{WE}$ alçak ise, $D_{GİRİŞ}$ te bulunan veri, adreslenen hücreye yazılır (saklanır). Yazma işlemi sırasında, çıkış tamponu yetkisiz kılınır (üç-durumlu). Çıkış tamponu, $\overline{CS}$ yüksek yapılarak da üç-durumlu hale getirilebilir.

Şekil 5.2b, MCM 93425 RAM ın bağımlılık gösterilişi sembolünü göstermektedir. Adres bağımlılığı *A* harfi ve onu izleyen 0 bölü 1023 ile belirtilmiştir. *A* ile işaretlenmiş diğer giriş ve çıkışlar, uygulanan adrese göre kullanılır veya etkilenir. *G* harfi VE bağımlılığını gösterir. Eğer $\overline{CS}$ aktifse (*L*) VE $\overline{WE}$ aktif yüksekse (*H*) veya $\overline{WE}$ aktif alçaksa (*L*), aygıt aktiftir. Buradaki VEYA, her iki WE girişinin 1 ile isimlendirilmesiyle anlatılmaktadır. Eğer $\overline{WE}$ aktif yüksekse, üç-durumlu çıkış yetkilendirilmiştir ve adreslenen hücredeki veri çıkışta görülür. Eğer $\overline{WE}$ aktif alçaksa, çıkış yetkisiz kılınmıştır ve $D_{GİRİŞ}$ teki veri, adreslenen hücreye yazılır. Bu ilişkileri görebilmenize yardımcı olabilmesi için Şekil 5.2a daki mantığı kullanın.

Belleklerin hızlarını karşılaştırmak için kullanılan önemli bir parametre olan T_{AA}, okumada adres erişim zamanıdır. Bu, adres girişlerine bir adres uygulandıktan sonra bellek hücresindeki verinin çıkışlarda geçerli olarak görülebilmesi için gerekli zamandır. Diğer bir deyişle, uygulanan adresle geçerli veri çıkışı arasındaki yayılma gecikmesi T_{AA} dır. Daha sonra kullanmak üzere MCM 93425 in maksimum T_{AA} sının 45 ns olduğunu belirtelim.



Şekil 5.2 Motorola MCM 93425 TTL RAM. *(a)* Blok diyagramı. *(b)* Bağımlılık gösterilişi sembolü

TTL RAM lar çoğunlukla, daha hızlı olan ve/veya daha fazla bit saklayabilen ECL veya yüksek-hızlı MOS RAM larla değiştirilmektedir. Daha sonra, kolayca bulunan MOS aygıtlarını örnek alarak, RAM ların okuma ve yazmadaki zamanlama parametrelerinin ayrıntıları üzerinde duracağız.

ECL RAMLAR

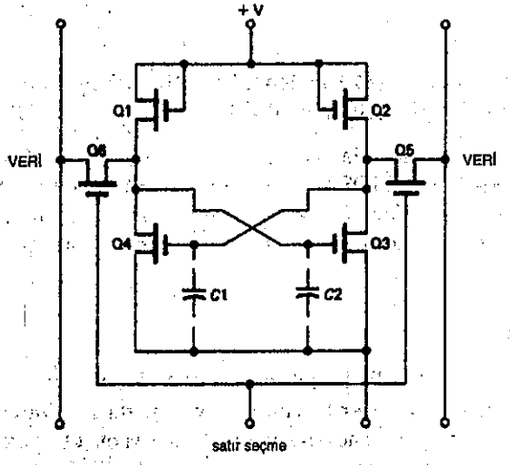
Emitör-kupleli mantık (ECL) çok fazla güç kullanmakla ün yapmıştır. Ancak, küçültülmüş transistörler kullanarak, çok fazla güç harcamayan ve TTL RAM larınkinden daha kısa adres erişim zamanları olan büyük ECL RAM lar üretilmiştir. Örneğin National Semiconductor DMI0470A, maksimum adres erişim zamanı 15 ns olan bir 4096-bitX1 lik aygıttır. Fairchild, maksimum adres erişim zamanı 35 ns olan daha büyük bir aygıtı, F100480 16K X 1 ECL RAM ı duyurmuştur.

8 veya 16-bitlik kelimeler saklamak için, Şekil 3-48 deki ROM larda gösterildiği gibi, bu aygıtların birçoğu paralel olarak adreslenebilir. Bu aygıtların çıkış transistörleri açık-emitörlüdür, böylece bir çoğunun çıkışları, daha fazla kelime saklamak için, bağlanmış- VEYA lanabilir. Her çıkışta $-V_{EE}$ ye bağlanan bir tek harici aşağı-çekme direnci veya -2 V a bağlanan sonlandırma direnci kullanılır. ECL sinyal iletim hatlarının, bir kaç inçten daha uzunsa, bir dirençle sonlandırılmaları gerektiğini unutmayın.

STATİK MOS RAM LAR

YAPI

Daha önce anlatılan TTL hücresine çok benzer şekilde artma-modu MOSFET transistörleri de RAM yapmak için kullanılabilir. Şekil 5-3, bir statik MOS RAM hücresini göstermektedir. Burada *Q1* ve *Q2*, çapraz bağlı ters çeviriciler *Q3* ve *Q4* için sadece yük direnci görevini görmektedirler. *Q5* ve *Q6*, ters çevirici mandalın çıkışlarını DATA ve $\overline{\text{DATA}}$ hatlarına bağlayan anahtarlardır. Satır seçilince *Q5* ve *Q6* iletime geçer. DATA ve $\overline{\text{DATA}}$ hatları, bir matristeki her sütun için duygu yükseltecine ve yazma yükseltecine bağlıdır. Veri bir hücreye, o hücreyi seçerek ve yazma yükselteci ile o hücreyi istenilen duruma zorlayarak yazılır.

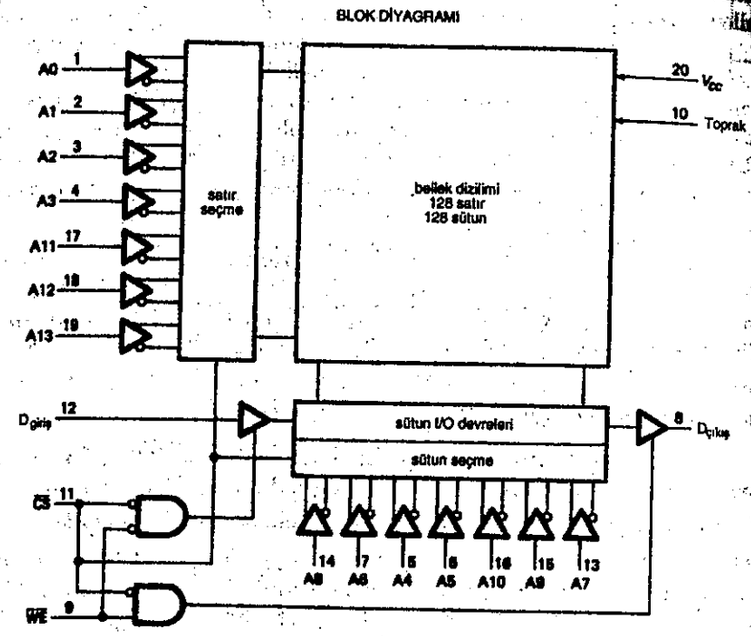


Şekil 5.3 MOS statik RAM hücresi.

Şekil 5.4, sadece 5 V luk bir besleme gerektiren Intel 2167 16384 x 1 statik MOS RAM ının blok diyagramını göstermektedir. 16384 kelimeden birini seçebilmek için 14 adres hattı gerektiğine dikkat ediniz. Aygıt yonga seçme (chip select : $\overline{\text{CS}}$) alçak olduğunda seçilir. $\overline{\text{CS}}$ alçak ise ve $\overline{\text{WE}}$ alçak olduğunda, girişteki veri adreslenen hücreye yazılır. Çıkış, yüksek-empedans (yüksek-z) durumunda olacaktır. $\overline{\text{WE}}$ yüksek olduğunda, adreslenen hücredeki veri çıkışta görülecektir. Giriş tamponu yetkisiz kılınmıştır. $\overline{\text{CS}}$ yi yüksek yapmak hem giriş hem de çıkış tamponlarını kayan duruma sokar (yüksek-z). Ayrıca, paketteki devrenin çoğu düşürülmüş güçte bir bekleme moduna geçirilir. Bir

2167-55, 125 mA lık maksimum aktif akıma karşın, bekleme modunda mak-
mum 40 mA kullanır. Bu, devre seçilmediğinde, güç harcamasını büyü
ölçüde azaltır. V_{cc} üzerinde fazladan yangeçit kapasitörleri kullanılmazsa,
aygıtın seçilmesi veya seçilmemesi sırasındaki büyük akım değişikliği besle-
me geriliminde büyük gürültülere sebep olabilir. Intel, her iki aygıtta bir V_{cc}
ile toprak arasına bir 0,1 µF lık seramik kapasitör kullanılmasını, her 16 ayg
bir de 22-47µF lik bir elektrolitik kapasitör kullanılmasını önermektedir.

Ayrıca, büyük bir baskılı-devre kartı üzerinde bu yüksek-hızlı aygıtlar-
dan çok sayıda kullanılmaktaysa, bütün adres hatları, veri hatları ve kontrol
hatları, uygun şekilde sonlandırılmış iletim hatları olmalıdır.



Şekil 5.4 Intel 2167 16K -bit statik RAM ın işlevsel blok diygramı.

Birçok RAM ın 1-bitlik çıkış kelimesi vermek üzere kodu çözülmüştür.
Bu, entegre devre paketi üzerindeki gerekli bacak sayısını azaltmak için yapıl-
maktadır. Çok-bitli kelimeleri saklamak için, bu 1-bit kelimeli (*X1* lik) RAM
lardan bir çoğu paralel olarak bağlanabilir. Örneğin 8 tane 2167 paralel bağla-
narak 4096 x 8 lik bir bellek oluşturulabilir. Ancak *X1* lik RAM lar kul-
lanıldığında, 8-bit genişliğinde bir bellek elde etmek için her zaman en az 8 ta-

ne entegre gerekmektedir. 8-bitlik mikroişlemcilerin popüler olması, bazı üre-
tici firmaları 8-bitlik çıkış kelimeli RAM lar üretmeye yöneltmiştir. Mostek
MK 4802, bir 2048 x 8 lik statik MOS RAM, bunun iyi bir örneğidir.

ZAMANLAMA

RAM veri sayfaları, birçok önemli zamanlama parametresini göster-
mektedir. Çoğu üreticiler; bu parametreleri, parametre isminin başharflerini
indis olarak kullanmakla tanıtmaktadır. Bu parametre değerlerini ezberleme-
nize gerek yoktur, ancak bu indislerin dilini öğrenmeniz ve verilen zaman-
ların dalga şekilleri ile ilişkilerini kurabilmeniz gerekmektedir. Bunun nedeni,
RAM devrelerinde bulunan problemlerin çoğunun (açıkça bozuk olan yonga-
lar hariç) zamanlama ile ilişkili olmasıdır. Belli bir bellek entegresi alçak fre-
kanslarda çalışabilir, fakat bir yüksek-frekans devresinde sağlıklı olarak
çalışabilmesi için erişim zamanı çok uzun olabilir. Bir yazma darbesi çok geniş
olabilir ve adres hatları değişmeden önce yeterli yazmayı kurtarma zamanına
izin vermeyebilir. Buradaki esas nokta, bir osiloskop ve bu zamanlama çizim-
leri ile genellikle problemi bulabileceğinizdir.

Şekil 5.5, Intel 2167 nin okuma çevrimi zamanlama dalgaşekillerini ve
değerlerini göstermektedir. Farklı sonekli numaraları olan parçalar için za-
manlama değerlerinin farklı olduğuna dikkat edin. Bütün 2167 ler aynı değil-
dir. Örneğin 2167-55 in maksimum T_{AA} sı 55 ns dir, 2167-70 ve 2167-L70 in
maksimum T_{AA} sı 70 ns dir. Kazara yanlış sonekli parça kullanmak bir sistem-
de zamanlama problemlerine neden olabilir.

Şimdi okuma çevrimi dalgaşekillerine ve altındaki notlara bakın. Oku-
ma çevrimi 1, $\overline{CS}$ nin alçak, $\overline{WE}$ nin yüksek olduğunu varsaymaktadır. T_{RC},
T_{OH} ve T_{AA} parametreleri adres girişlerindeki adresin bütün bitlerinin ge-
çerli olduğu andan itibaren ölçülmektedir. T_{OH} için belirtilen 5 ns lik mini-
mum değer, daha önceki adresten okunan verinin, yeni adres uygulandıktan
sonra en az 5 ns daha çıkışta bulunacağını göstermektedir. T_{AA} için belirtilen
55 ns lik maksimum değer (2167-55 için), yeni bir adres uygulandıktan en çok
55 ns sonra geçerli yeni verinin çıkışta olacağını belirtir. T_{OH} ve T_{AA} arasında-
ki çarpraz-taranmış bölge, çıkışın eski adreslenen veriden yeni adreslenen ve-
riye bu zaman aralığında herhangi bir yerde değişebileceğini belirtir. Diğer bir
deyişle, bu süre içinde çıkışın ne olacağı öngörülemez. Belirsizlik aralığı,
aygıttaki yayılma gecikmelerindeki farklılıklardan ve parçalar arasındaki
değişikliklerden doğmaktadır. Bu aralığa genellikle **bükülme bölgesi** veya
sadece **bükülme** denir. Okuma çevrimindeki T_{RC} zamanı, adresin sabit tutul-
ması gereken minimum zamandır. Bu süre geçmeden bir sonraki okuma çev-
rimi için yeni bir adres uygulanamaz.

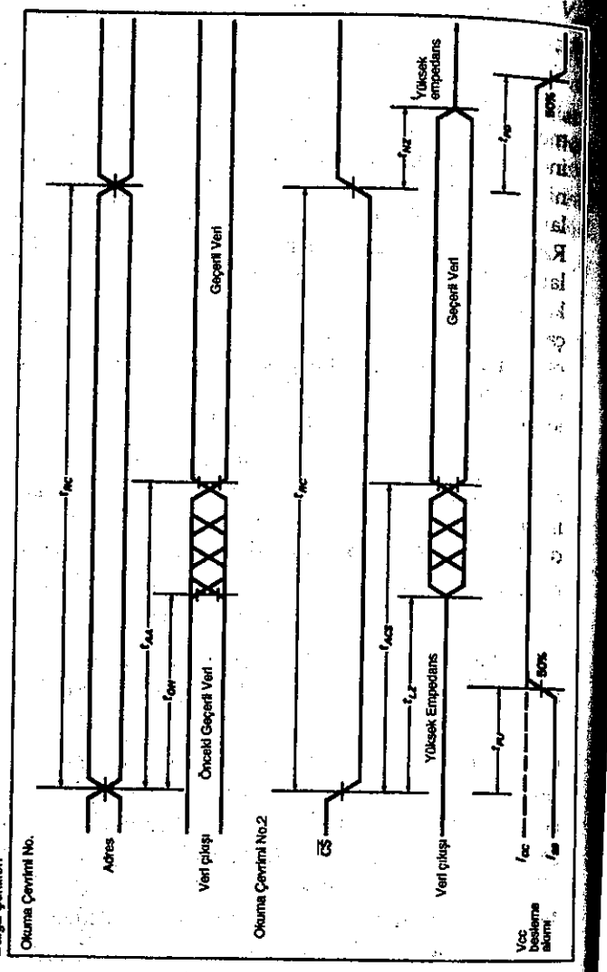
Okuma çevrimi 2 dalgaşekli, eğer $\overline{WE}$ yüksekse ve $\overline{CS}$ alçak yapıldığında bir adres halen varsa veya geçerli hale gelirse oluşan zamanlamayı göster

A.A. Karakteristikleri T_A = 0°C ve 70° C, V_{CC} = +5V ± 10%; aksi belirtilmedikçe

okuma çevrimi

SEMBOL	PARAMETRE	2167-08 MIN.	2167-08 MAX.	2167-70,2167L-70 MIN.	2167-70,2167L-70 MAX.	2167-10 MIN.	2167-10 MAX.	UNIT
t_{RC}[1]	Okuma Çevrim Zamanı	85		70		100		ns
t_{AA}	Adres Erişim Zamanı		85		70		100	ns
t_{ACS}	Yonga Seçme Erişim Zamanı		85		70		100	ns
t_{OH}	Adres Değişiminden Sonra Çıkışı Tutma Zamanı	5		5		5		ns
t_{LZ} [2,3]	Yonga Seçiminden Sonra Çıkışa Alçak Z Geçişi	10		10		10		ns
t_{HZ} [2,3]	Yonga Seçilmesinden Sonra Çıkışın Yüksek Z Geçişi	0	30	0	40	0	40	ns
t_{PU}	Yonga Seçiminden Sonra Güç Yükselme Zamanı	40		60		55		ns
t_{PD}	Yonga Seçilmesinden Sonra Güç Düşme Zamanı		85		70		80	ns

Dalga Şekilleri



Şekil 5.5 Intel 2167 statik RAM için okuma çevrimi zamanlama dalga şekilleri ve zamanlama değerleri.

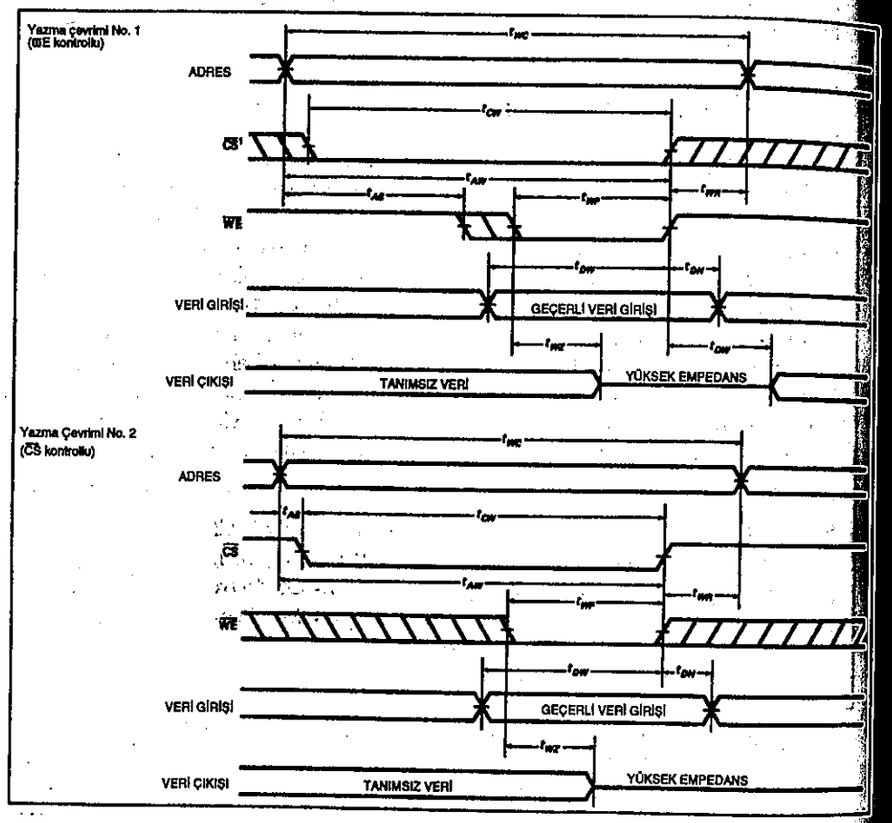
Önemli
HASAN'a
MEB. den
Kitap ve

Temel Bil
Elek. Hid Kon
Pompa
Hidrolik Araç
NC Mak Pro
7 CNC Parça Prog
5 Prog
6 (NC ve)
Mikro işlemciler ve Zam 2.)

Adres: SELANİK CADDESİ NO: 45 ANKARA TELEFON: 118 32 69 - 118 32 70 - 118 32 71 - 118 32 72 Fax: 125 75 26

YAZMA ÇEVRİMİ

sembol	PARAMETRE	2167-55 MIN.	2167-55 MAX.	2167-70, 2167L-70 MIN.	2167-70, 2167L-70 MAX.	2167-10 MIN.	2167-10 MAX.	Birim
t_{WC}[2]	Yazma çevrim zamanı	55		70		100		ns
t_{CW}	Yonga seçmeden yazmanın sonuna	55		70		90		ns
t_{AW}	Geçerli adresten yazmanın sonuna	55		70		90		ns
t_{AS}	Adres kurma zamanı	0		0		5		ns
t_{WP}	Yazma darbesi genişliği	35		40		45		ns
t_{WR}	Yazma nakalat süresi	0		0		5		ns
t_{DW}	Geçerli veriden yazmanın sonuna	25		30		35		ns
t_{DH}	Veri tutuma zamanı	0		0		0		ns
t_{WZ}[3]	Yazma yetkilendirmeden yüksek Z çıkışa	0	25	0	35	0	35	ns
T_{OW}	Yazmanın sonunda çıkış aktife	0		0		0		ns



Şekil 5.6 Intel 2167 statik RAM ın yazma çevrimi zamanlama dalgaşeklileri ve zamanlama değerleri.

Farklı üreticiler bu zamanlama parametreleri için değişik indisler kullanmaktadırlar, fakat dalgaşekillerinin yardımıyla bir sistemi kontrol etmek için gerekli değerleri bulabilirsiniz.

UÇUCU OLMAYAN RAM LAR

Xicor, Inc. firması statik RAM ile elektrikle silinebilir PROM u aynı yonga üzerinde birleştirerek normal RAM olarak çalışabilen ve güç kesilmesi durumunda saklanmış verileri EEPROM da saklayabilen bir aygıt yapmıştır. Bu X2201 aygıtı bir 1024-bitlik RAM ile bir 1024-bitlik EEPROM içermektedir. RAM ın bütün içeriği, $\overline{\text{STORE}}$ isimli tek bir sinyalle 4 ns den az bir sürede paralel olarak EEPROM a taşınabilir. Bu aygıtların avantajı güç kesilmesi durumunda kullanılmakta olan verileri saklamak için bir pil yedeklemesine gerek olmamasıdır.

CMOS RAM LAR

Bir güç kesilmesi durumunda verilerin saklanması gereken uygulamalarda, çoğunlukla bir pil destekleme kaynağı ile birlikte CMOS RAM lar kullanılmaktadır. Statik CMOS RAM hücresi Şekil 5.3 teki statik MOS RAM hücresi ile aynıdır, sadece *Q1* ve *Q2* yük transistörleri P-kanal yalnız -E MOSFET lerle değiştirilmiştir. Bu, yapıyı Bölüm 2 de anlatılan sadece durum değiştirirken güç harcayan çapraz-bağlantılı CMOS ters çeviricler haline dönüştürür. CMOS RAM ın erişilmediğinde son derece az akım çekmesi, ufak bir nikel-kadmiyum pilin destekleme kaynağı olarak bellek kartı üzerine konmasına imkan vermektedir. Örneğin Harris HM6514, bekleme modunda sadece maksimum 50µA gerektiren bir 1024 x 4 CMOS RAM dır. HM6514, 2,0V kadar düşük bir V_{cc} besleme gerilimi ile bile verileri tutacaktır. V_{cc} gerilimi 4,5 ile 5,5 V arasında iken aygıtın maksimum adres erişim zamanı 220 ns dir.

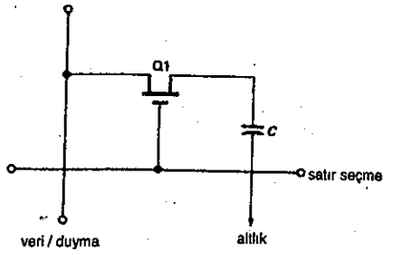
Bu kitabın yazıldığı sıralarla, birçok üretici 2K x 8 CMOS RAM ların duyurusunu yapmışlardır. Bazı büyük statik RAM aygıtları, alanı en aza indirgemek için saklama dizilimleri için standart NMOS devreleri ve bekleme durumu güç harcamasını en aza indirgemek için CMOS çevre devreleri kullanmaktadır.

DİNAMİK MOS RAM LAR

YAPI

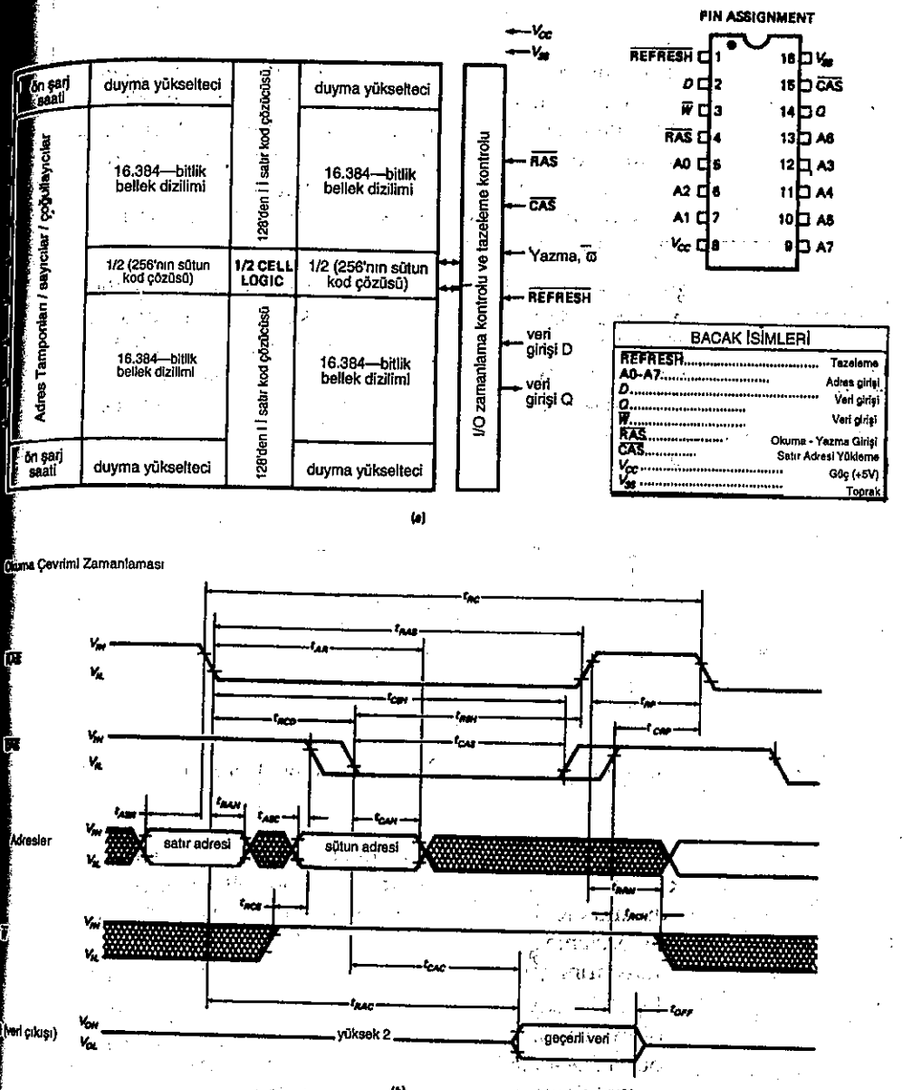
Dinamik MOS RAM hücreleri, verileri statik RAM hücrelerindeki gibi çapraz-bağlantılı ters çevirici mandallarda saklamak yerine, küçük kapasitörler üzerinde yük olarak saklamaktadırlar. Bu nedenle bu dinamik RAM hücrelerinin, Bölüm 4 te anlatılan dinamik kaydırma kaydedici hücrelerinde olduğu gibi, her 2 ms veya daha az sürelerde tazelenmeleri gerekmektedir. Dinamik hücrelerin avantajları daha basit ve daha küçük , daha ucuz olmaları ve daha az bekleme durumu gücü gerektirmeleridir.

Şu andaki aygıtlar, Şekil 5.7 de gösterilene benzer tek bir transistör hücresi kullanmaktadır. Burada Q1, satırı seçilince saklama kapasitörünü VERİ/DUYMA (DATA/SENSE) hattına bağlayan bir anahtardan ibarettir. Bu hücrelerden oluşan bir RAM diziliminde, her sütunda, adreslendiğinde hücrenin durumunu okumak için bir duyma yükselteci ve hücreye yazmak için bir yazma yükselteci vardır.



Şekil 5.7 Tek-transistörlü dinamik RAM saklama hücresi.

Hücre başına olan alanın küçük olmasından dolayı bir tek yonga üzerine çok sayıda dinamik RAM hücresi konulabilir. Motorola MCM 6664 bir 65536x1 yüksek-hızlı dinamik RAM dır. Standart 16 bacaklı DIP olarak paketlenmiştir ve sadece +5 V besleme gerektirir. Şekil 5.8a, MCM 6664 ün blok diyagramını ve bacak atamalarını göstermektedir. Gördüğünüz gibi aygıtın sadece 8 adres giriş vardır. 65536 sayısı 2^{16} ya eşit olduğuna göre aygıtta saklanan 65536 kelimeden birini seçebilmek için 16 adres bitinin gerekeceğini beklerdiniz (bu konuda Bölüm 3 te anlatılanlara bakın). Gerçektende 16 adres biti gerekmektedir. Ancak aygıt üzerindeki bacak sayısını azaltmak için, 16 adres biti, *A0* ile *A7* arasındaki adres girişine sekizer sekizer olmak üzere çoğullanmıştır.



Şekil 5.8 Motorola MCM6664 64K-bitlik dinamik RAM *(a)* Blok diyagramı ve bacak ataması. *(b)* Okuma çevrimi zamanlama dalgaşekilleri.

PARAMETER	SYMBOL	MCM6664-15		MCM6664-20		UNITS	NOTES
		MIN	MAX	MIN	MAX		
Random Read or Write Cycle Time	t_{RC}	300	—	350	—	ns	8, 9
Read-Write Cycle Time	t_{RWC}	300	—	350	—	ns	8, 9
Access Time from Row Address Strobe	t_{RAC}	—	150	—	200	ns	10, 12
Access Time from Column Address Strobe	t_{CAC}	—	75	—	110	ns	11, 12
Output Buffer and Turn-Off Delay	t_{OFF}	0	30	0	40	ns	18
Row Address Strobe Precharge Time	t_{RP}	120	—	140	—	ns	—
Row Address Strobe Pulse Width	t_{RAS}	150	10,000	200	10,000	ns	—
Column Address Strobe Pulse Width	t_{CAS}	75	10,000	110	10,000	ns	—
Row to Column Strobe Lead Time	t_{RCD}	30	75	35	90	ns	13
Row Address Setup Time	t_{ASR}	0	—	0	—	ns	—
Row Address Hold Time	t_{RAH}	25	—	30	—	ns	—
Column Address Setup Time	t_{ASC}	0	—	0	—	ns	—
Column Address Hold Time	t_{CAH}	45	—	55	—	ns	—
Column Address Hold Time Referenced to $\overline{RAS}$	t_{AR}	120	—	155	—	ns	—
Transition Time (Rise and Fall)	t_T	3	50	3	50	ns	6
Read Command Setup Time	t_{RCS}	0	—	0	—	ns	—
Read Command Hold Time	t_{RCH}	10	—	10	—	ns	14
Read Command Hold Time Referenced to $\overline{RAS}$	t_{RRH}	30	—	35	—	ns	14
Write Command Hold Time	t_{WCH}	45	—	55	—	ns	—
Write Command Hold Time Referenced to $\overline{RAS}$	t_{WCR}	120	—	155	—	ns	—
Write Command Pulse Width	t_{WP}	45	—	55	—	ns	—
Write Command to Row Strobe Lead Time	t_{RWL}	45	—	55	—	ns	—
Write Command to Column Strobe Lead Time	t_{CWL}	45	—	55	—	ns	—
Data in Setup Time	t_{DS}	0	—	0	—	ns	15
Data in Hold Time	t_{DH}	45	—	55	—	ns	15
Data in Hold Time Referenced to $\overline{RAS}$	t_{DHR}	120	—	155	—	ns	—
Column to Row Strobe Precharge Time	t_{CRP}	−10	—	−10	—	ns	—
$\overline{RAS}$ Hold Time	t_{RSH}	75	—	110	—	ns	—
Refresh Period	t_{REFH}	—	2.0	—	2.0	ms	—
$\overline{WRITE}$ Command Setup Time	t_{WCS}	−10	—	−10	—	ns	16
$\overline{CAS}$ to $\overline{WRITE}$ Delay	t_{CWD}	45	—	55	—	ns	16
$\overline{RAS}$ to $\overline{WRITE}$ Delay	t_{RWD}	125	—	160	—	ns	16
$\overline{CAS}$ Hold Time	t_{CSH}	150	—	200	—	ns	—
$\overline{RAS}$ to $\overline{REFRESH}$ Delay	t_{RFD}	0	—	0	—	ns	—
$\overline{REFRESH}$ Period (Battery Backup Model)	t_{REF}	2000	—	2000	—	ns	—
$\overline{REFRESH}$ to $\overline{RAS}$ Precharge Time (Battery Backup Model)	t_{RPH}	390	—	460	—	ns	—
$\overline{REFRESH}$ Cycle Time (Auto Pulse Mode)	t_{FC}	330	—	390	—	ns	—
$\overline{REFRESH}$ Pulse Period (Auto Period Mode)	t_{FP}	60	2000	60	2000	ns	—
$\overline{REFRESH}$ to $\overline{RAS}$ Setup Time (Auto Pulse Mode)	t_{FSR}	30	—	30	—	ns	—
$\overline{REFRESH}$ to $\overline{RAS}$ Delay Time (Auto Pulse Mode)	t_{FRD}	390	—	460	—	ns	—
$\overline{REFRESH}$ Inactive Time	t_{FI}	30	—	30	—	ns	—
$\overline{RAS}$ to $\overline{REFRESH}$ Lead Time	t_{FRL}	390	—	460	—	ns	—

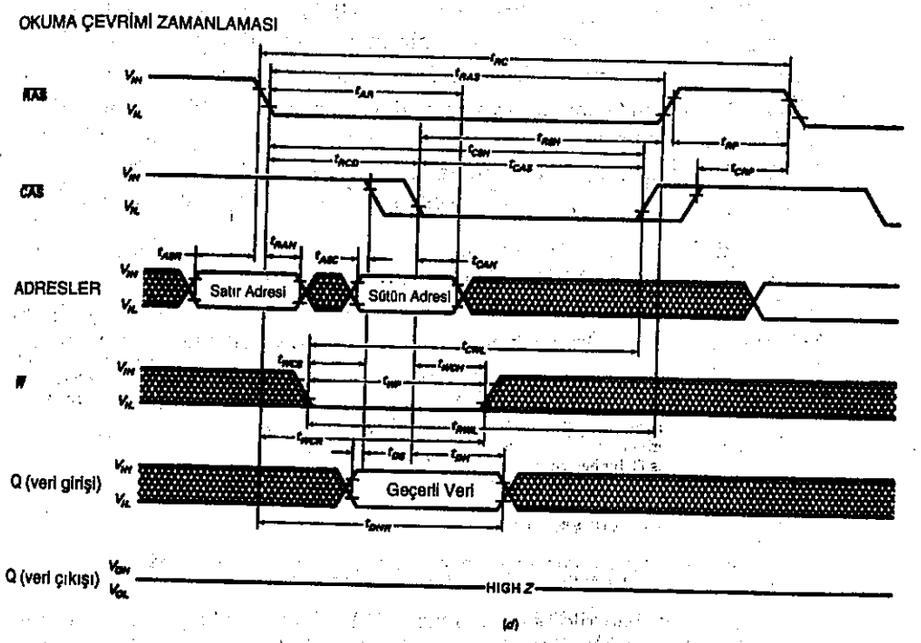
Şekil 5.8 (c) Zamanlama değerleri.

NOTLAR:

1. Bütün gerilimler için V_{SS} referans alınmıştır.
2. V_{IH} min ve V_{IL} mak, giriş sinyallerinin zamanlamasını ölçmek için kullanılan referans seviyeleridir.
3. Devrenin doğru çalışmasını garantilemek için güç verilmesini izleyen herhangi 8 RAS çevriminden sonra başlangıçta 100 µs lik bir bekleme gerekmektedir.
4. Akım, çevrim oranı ve çıkış yüklemesinin bir fonksiyonudur; maksimum akım, çıkış açıkken en hızlı çevrim oranında ölçülür.
5. Çıkış yetkisiz kılınmıştır (açık-devre) ve $\overline{RAS}$ ve $\overline{CAS}$ nin her ikisi de mantık 1 dedir.
6. Geçiş sinyali özellikleri bütün giriş sinyalleri için geçerlidir. Geçiş oranı özelliklerini karşılamanın yanısıra bütün giriş sinyalleri V_{IH} ile V_{IL} (veya V_{IL} ile V_{IH}) arasında monoton bir şekilde geçiş yapmalıdır.
7. Boonton Metre ile ölçülen kapasitans veya $C = \frac{\Delta t}{\Delta V}$ denkleminden hesaplanan etkin kapasitans.
8. t_{RC} (min) ve t_{RWC} (min) için verilen değerler, doğru çalışmanın garantilendiği tüm sıcaklık aralığındaki (0°C ≤ TA ≤ 70°C) çevrim zamanını göstermek için kullanılır.
9. A.a. ölçümleri $t_T = 5{,}0$ ns olduğunu varsayar.
10. $t_{RCD} \leq t_{RCD}$ (mak) olduğunu varsayar.
11. $t_{RCD} \geq t_{RCD}$ (mak) olduğunu varsayar.
12. 2 TTL yüküne (+ 200µA, - 4mA) denk akım yükünde ve 100 pF (V_{OH} = 2,0V, V_{OL} = - 0,8 V) da ölçülmüştür.
13. t_{RCD} (mak) sınırları içinde çalışmak t_{RAC} (mak) karşılanabileceğini garantiler. t_{RCD} (mak) sadece referans noktası olarak belirtilmiştir; eğer t_{RCD} belirtilen t_{RCD} (mak) sınırından daha büyükse, bu takdirde erişim zamanı sadece t_{CAC} tarafından kontrol edilir.

14. Bir okuma çevrimi için ya t_{RRH} veya t_{RCH} sağlanmalıdır.
15. Bu parametrelere, rastgele yazma çevrimlerinde $\overline{CAS}$ nin ön kenarı ve gecikmeli yazma veya oku-değiştir-yaz çevrimlerinde $\overline{WRITE}$ nin ön kenarı referans alınmıştır.
16. t_{WCS}, t_{CWD} ve t_{RWD} sınırlayıcı çalışma parametreleri değildir. Veri sayfalarında sadece elektriksel özellikler olarak içerilmişlerdir: Eğer $t_{WCS} \geq t_{WCS}$ (min) ise çevrim bir erken yazma çevrimidir ve veri çıkış bacağı bütün çevrim boyunca açık-devre (yüksek empedans) olarak kalacaktır; eğer $t_{CWD} \geq t_{CWD}$ (min) ve $t_{RWD} \geq t_{RWD}$ (min) ise çevrim bir okuma-yazma çevrimidir ve veri çıkışı seçilen hücreden okunan veriyi içerecektir; eğer yukardaki koşul takımlarının hiçbiri de sağlanmıyorsa, veri çıkışının durumu (erişim zamanında) belirsizdir.
17. Adresler, veri-girişi ve $\overline{WRITE}$ ın ne olduğu farketmez. Veri-çıkışı $\overline{CAS}$ ın durumuna bağlıdır. Eğer $\overline{CAS}$ alçak kalırsa, önceki çıkış geçerli kalacaktır. Bacak # 1 tazeleme çevrimi sırasında $\overline{CAS}$ nin aktifden aktif olmayan duruma geçişine müsade edilmektedir. $\overline{CAS}$ yükseğe çıkarıldığında, çıkış yüksek-empedans durumuna geçer.
18. t_{OFF} (mak), çıkışın açık devre durumuna ulaştığı ve çıkış gerilim seviyelerine referans alınmadığı süreyi tanımlar.

Şekil 5.8(c) devam



Şekil 5.8(*d*) Yazma çevrimi zamanlama dalgaşekilleri.

Şekil 5.8 b deki MCM6664 ün okuma çevrimi zamanlama dalgaşeklini kullanarak bunun nasıl yapıldığını görebilirsiniz. Bir harici çoğullayıcı tarafından bir 8-bitlik satır adresi, aygıtın adres girişlerine uygulanır. 6664 ün içindeki mandallara satır adresini kilitlemek için, satır adresi yükleme girişi $\overline{RAS}$, harici devre tarafından aktif alçağa çekilir. Bir T_{RAH} kadarlık bekleme süresinden sonra satır adresi kaldırılır, yazma girişi $\overline{W}$ yüksek yapılır ve 8-bitlik sütun adresi adres girişlerine uygulanır. Bundan sonra, 6664 ün içindeki mandallara sütun adreslerini kilitlemek için harici devre, sütun adresi yükleme girişi $\overline{CAS}$ ye bir alçak darbe gönderir. Bir T_{CAC} kadar süre sonra, adreslenen hücredeki veri, Q (veri) çıkışında geçerli veri olarak hazır olacaktır. Satır ve sütun adreslerinin yüklenmesi dışında, bu zamanlama diyagramı, daha önce anlatılan 2167 MOS statik RAM ınkilere benzemektedir. Dalgaların taranmış bölgeleri bu hatların durumlarının bilinmediği ve "fark etmez" oldukları zamanları gösterir. Harici donanımın uygun zamanlama ile satır adresi $\overline{RAS}$, sütun adresi $\overline{CAS}$ ve $\overline{W}$ sinyallerini sağlaması gerektiğini unutmayın. Şekil 5.8c, MCM6664-15 ve MCM6664-25 için zamanlama değerlerini göstermektedir. Bunların çoğu dikkate değer. 6664-15 i referans olarak kullanırsak, toplam okuma erişim zamanı T_{RAC} nin maksimum 150 ns ve sütun adres erişim zamanı T_{CAC} nin maksimum 75 ns olduğuna dikkat edin. Bu ikisinin arasındaki fark çoğullanmış adres girişleri olan daha küçük bir paketin bedelidir. Toplam okuma çevriminin (T_{RC}) minimum değeri 300 ns dir. - 10 ns lik T_{CRP} değeri, $\overline{RAS}$ nin yeni bir okuma çevrimi için aşağı geçmesinden sonra 10 ns ye kadar bir süre içinde $\overline{CAS}$ ın yükseğe çıkabileceğini belirtmektedir.

Şekil 5.8d, MCM6664 ün yazma çevrimi zamanlama dalgaşekillerini göstermektedir. Yazma işlemi için, adres girişlerine bir satır adresi uygulanır ve $\overline{RAS}$ aktif alçak yapılır. Tutma süresi T_{RAH} kadar sonra satır adresi kaldırılır. Sonra, eğer $\overline{W}$ alçak değilse alçak yapılır, veri girişlerine veri uygulanır ve aygıtın adres girişlerine sütun adresi gönderilir. Arkasından sütun adresini kilitlemek için $\overline{CAS}$ aktif alçak yapılır. CAS alçak olduktan sonra sütun adresi T_{CAH} kadar bir süre sabit tutulmalıdır ve yazılacak olan veri yine CAS alçak olduktan sonra T_{DH} kadar sabit tutulmalıdır.

BİR DİNAMİK RAM IN TAZELENMESİ

Daha önce belirtildiği gibi, dinamik RAM lar bitleri küçük bir kapasitör üzerinde yükün varlığı veya yokluğu olarak saklarlar. Kapasitör her 2 ms de bir tazelenmezse, yük sızıp kaçacak ve saklanmış veri kaybolacaktır. Tazelemeyi yapmak için aygıtın her satırına her 2 ms de en az bir defa erişilmelidir. MCM6664 bunun birkaç değişik şekilde yapılmasına imkân verir.

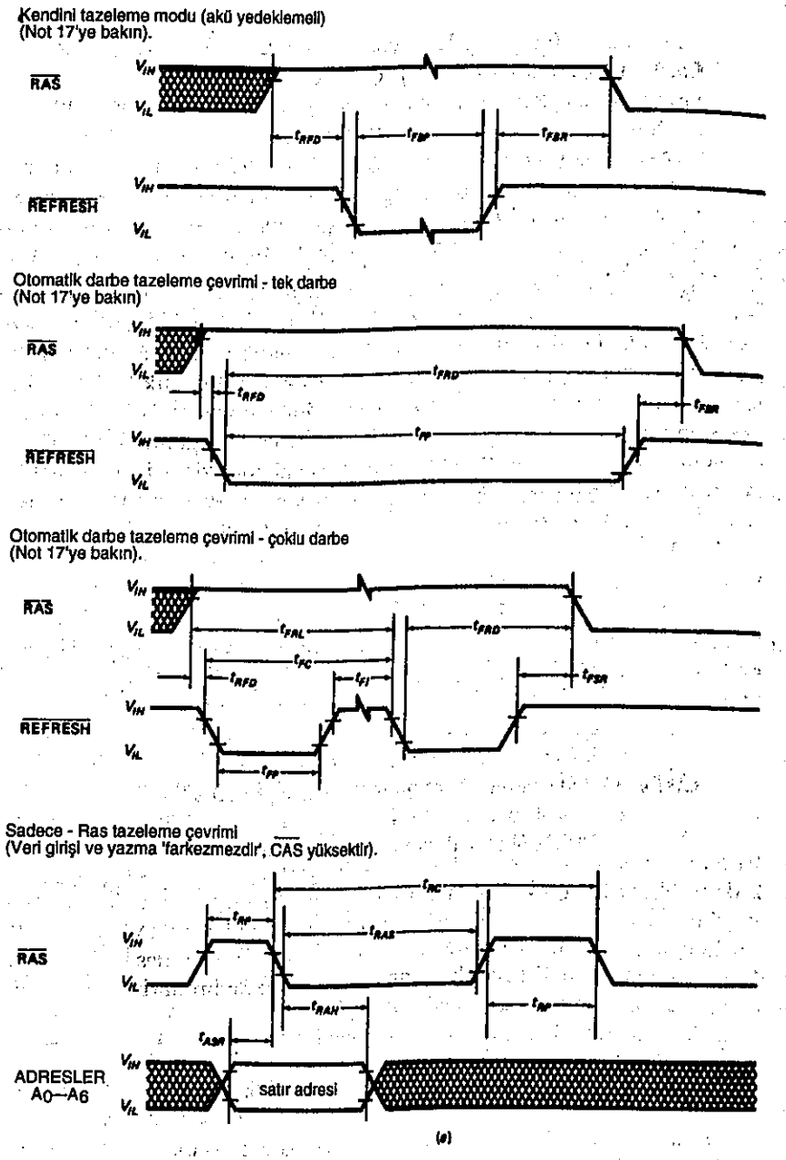
Şekil 5.8e, üç farklı tazeleme yönteminin zamanlama dalgaşekillerini göstermektedir. Yalnız-RAS tazeleme çevrim yöntemi, eski nesil 16K dinamik RAM ların tazelenmesinde kullanılan yöntemle aynıdır. Bu yöntemde $\overline{CAS}$ yüksek tutulur, $\overline{REFRESH}$ yüksek tutulur ve $D_{GIRIŞ}$ ile $\overline{W}$ nin durumları farketmez. A0 ile A6 arasındaki adres girişlerine 7-bitlik bir adres göndermek için bir harici sayıcı kullanılır ve $\overline{RAS}$ alçağa çekilir. Gerekli T_{RAS} süresinden sonra $\overline{RAS}$, T_{AP} süresince yüksek yapılır. Harici sayıcı, bir sonraki satırın adresini elde etmek için, bir artırılır ve $\overline{RAS}$ tekrar alçağa çekilir. Bu süreç her 2 ms de bir olmak üzere 128 defa tekrar edilir. 6664 te aynı anda iki tane 8-bitlik satır tazelenmektedir, bu nedenle sadece 128 işlem gerekmektedir. 128 satırın tamamı ya her 2 µs lik sürenin sonunda toplu olarak veya okuma ve yanma işlemleri arasına dağıtılarak tazelenir. Buradaki temel nokta her satırın her 2 ms de en az bir defa adreslenip yüklenmesi gerektiğidir. Bir yalnız-RAS tazeleme çevriminı minimum süresi, T_{RC} , 300 ms olduğuna göre 128 satır için : 128 satır x 300 ns/satır = 38,4 µs gerekir. O halde, aygıtı toplu olarak tazelemek için her 2 ms de 38,4 µs harcamak gerekir. Eğer dağıtılmış mod, 2 ms nin tamamına eşit olarak yayılırsa, bu takdirde her

$$\frac{2000\,\mu s}{128} = 15{,}6\,\mu s$$

de bir satırın tazelenmesi gerekecektir. Başka bir deyişle dağıtılmış tazeleme, bir satırı tazelemek için her 15,6 µs nin 300 ns sini kullanacaktır.

MCM 6664 ün otomatik darbe tazeleme çevirim modunda, $\overline{RAS}$ yüksek yapılır ve $\overline{REFRESH}$ e her 2 µs de 128 defa darbe gönderilir. Bir dahili satır adres kaydedici her darbeden sonra otomatik olarak bir artırılır, böylece bir harici sayıcıya gerek yoktur. REFRESH darbeleri toplu veya dağınık modda gönderilebilir. $\overline{REFRESH}$ alçak iken aygıt okunup yazılamaz.

Kendi-kendini-tazeleme modunda, $\overline{RAS}$ yüksek ve $\overline{REFRESH}$ alçak yapılır. Eğer $\overline{REFRESH}$ 2000 ns den daha fazla alçak ise bir dahili osilatör çalışmaya başlar ve satırları her 2 ms de bir tazelemek üzere dahili satır adres kaydedicinin otomatik olarak artırır. Bu modda aygıt okunamaz veya aygıta yazılamaz. Bu modun esas işlevi, bir güç kesilmesi halinde, bir sistem, RAM ı pille destekleme güç kaynağına bağlamak üzere anahtarlarken verilerin korunmasıdır.



Şekil 5.8(e) Tazeleme zamanlama dalgaşekilleri.

DİNAMİK RAM LAR İÇİN GEREKLİ HARİCİ DONANIM

Piyasadaki dinamik RAM ların çoğu epeyce bir miktar harici donanım gerektirir. Yalnız-RAS tazeleme çevrimleri olanlar bir harici sayıcı, sayıcı için bir saat ve aygıta satır adreslerini, sütun adreslerini veya tazeleme sayısını göndermek için bir çoğullayıcı gerektirir. Ayrıca bir mikrobilgisyar, aygıttan okumak isterken veya aygıta yazmak isterken aynı zamanda tazeleme devresi de bir satırı tazelemek isterse, bir de karar verme devresi gerekmektedir.

Bu noktada akla gelebilecek bir soru, niçin bir kimsenin bütün bu karmaşık tazeleme ve çoğullama devrelerini gerektiren dinamik RAM ları bunları gerektirmeyen statik RAM lar yerine kullanmak isteyeceğidir. Bu sorunun birçok cevabı vardır. Birincisi, dinamik RAM lar daha yoğundur (her entegrede daha fazla bit) ve birim başına fiatı daha ucuzdur. İkincisi, dinamik RAM lar bit başına çok daha az güç gerektirirler. Örneğin, MCM6664 erişildiğinde sadece 50 mA ve beklemede 5mA çeker. Not: Bu akımlar arasındaki büyük farklardan dolayı, dinamik RAM bellek kartları ardarda gelen her iki aygıttan birinin yanına toprak ile V_{CC} arasına bağlanan 0,1µF lık kapasitör gerektirir. Üçüncüsü, bütün bir dinamik RAM kartının tazeleme ve karar verme işlemleri için bir tek tazeleme kontrol entegresi yeterlidir. Intel 8202 bu işleri 16K lık aygıtlar için yapmaktadır. Hiç şüphesiz, daha geniş kullanım alanları buldukça 64K lık RAM lar içinde benzer kontrol entegreleri piyasaya çıkacaktır.

DİNAMİK RAM LARIN GELECEĞİ

256 K bit saklayabilen dinamik RAM lar halen hazırda bulunmaktadır. Bacak 1, gerekli dokuzuncu adres hattı girişi olarak kullanılır. Bu aygıtlar "tekrarlama" kullanır. Bu, verimi artırmak için aygıtın ekstra satır ve sütunlar ile yapıldığı anlamına gelir. Son testte, bunlar arızalı olanların yerine bir lazer yardımıyla değiştirilerek konur. Zilog Z6132 gibi bazı yeni nesil aygıtlar tazeleme ve karar verme devrelerini de yonga üzerinde bulundurmaktadır ve böylece kullanıcıya statik olarak görülürler.

Büyük dinamik RAM ların bir diğer problemi de, paketleme malzemelerinin alfa parçacık radyasyonunun neden olduğu bir-defalık, "yumuşak" hatalardır. Şimdi çoğu üreticiler bellek yongalarını, alfa parçacıklarını emmesi için, polimid ile kaplamaktadırlar. Bu tür hatalara müsamaha gösterilemeyecek büyük bellek sistemlerinde, tazeleme kontrol devresiyle birlikte hata bulma ve düzeltme devreleri de eklenmiştir.

JOSEPHSON KAVŞAĞI BELLEKLERİ

Süperiletken malzemelere ve elektron tünellerine dayalı bir yeni teknoloji, 2 inçlik (5,1 cm) bir tek kübe bütün bir bilgisayarı ve birkaç megabaytlık RAM ı sıkıştırma ihtimalini getirmiştir. Bu bilgisayarın çevrim zamanı 1982 nin en hızlı işlemcilerinden 50 defa daha hızlı olabilir ve 1000 defa daha az güç kullanacaktır. Bu bilgisayarla ilgili bir zorluk bütün birimin mutlak sıfır (- 273,2^{o}C) civarında tutulması zorunluluğudur.

Saklama elemanı, mutlak sıfır civarında hiç elektrik direnci olmayan bir superiletken metal halkasıdır. Bu halka içinde bir defa bir akım akmaya başladı mı, teorik olarak sonsuza kadar akmaya devam edecektir. Halka üzerinde bir yönde akan akım saklanmış bir 0 yani alçağı, diğer yönde akan akım da saklanmış bir 1 yani yükseği temsil edebilir. Bir duyma hattı akımın hangi yönde aktığını algılar.

Halka içinde içerilen bir veya iki Josephson kavşağı anahtar, dönen akımı başlatmak veya yönünü değiştirmek için kullanılır. Bir Josephson kavşağı anahtar, 30 ile 50 *Å* kalınlığında bir izolatörle birbirinden ayrılmış iki süperiletken metal şeridinden oluşmaktadır. Bir manyetik alanın yokluğunda, bir tünel diyotun çalışmasına benzer şekilde, elektronlar bu ince izolatörden tünel açarak geçerler. Josephson kavşağına uygulanan ufak bir manyetik alan tünel açma işlemini durdurur. Bu durumda kavşak açık bir anahtar gibi davranır. Josephson kavşağı anahtarların avantajı birkaç pikosaniye içinde açılıp kapanabilmeleridir.

Bu teknolojiyi kullanarak bellek hücrelerinin yanısıra mantık kapıları da yapılabileceğinden, bir bilgisayarın tamamı bu teknoloji ile üretilebilir. Son zamanlarda, oda sıcaklığında da süperiletken olan malzemeler bulunmuştur. Bu, mutlak sıfır civarında tutulmasına gerek olmayan Josephson kavşağı bilgisayarlar yapılmasının mümkün olabileceği ümitlerini doğurmuştur.

KAYDIRMALI KAYDEDİCİ BELLEKLER

Flip-flopları paralel dizilimler halinde bağlayarak RAM ların sayısal verileri nasıl sakladıklarını anlatmıştık. Kaydırma kaydedici olarak bağlanmış flip-floplar da sayısal verileri saklamak için kullanılabilir. Veriler seri olarak veya kısa kaydedicilerde, paralel olarak yüklenebilir. Saklanabilecek bit sayısı, kademe yani hücre sayısına eşittir. Bir tek entegre devre paketine maksimum sayıda kaydırma kaydedici hücresi sığdırabilmek için yük-kupleli aygıt veya manyetik balon bellek teknolojisi sıkça kullanılmaktadır.

Kaydırma kaydedicileri gibi seri yarıiletken bellekleri daha kolay yapıldıkları ve bit başına daha az yonga alanı kullandıklarından RAM tiplerine

göre daha avantajlıdır. Seri belleklerin bir dezavantajı ise saklanmış veri kelimesinin ilk bitine kaydırıncaya kadar daha uzun zaman alabilmesidir.

YÜK - KUPLELİ AYGIT BELLEKLERİ

Şekil 4.43 e bakın. Bir *CCD* nin yapısı bir altlık, bir oksit dielektrik katmanı ve bir dizi ayrık kapıdan ibarettir.Bir kapının altındaki potansiyel kuyusunda bir yük paketi varsa bir 1, bir yük paketi yoksa bir 0 saklanmıştır. Şekil 4.44 te gösterildiği gibi, bir yük paketi *CCD* kaydedicisinden dört fazlı bir saatle kaydırılmaktadır. *CCD* dinamiktir. Kaydedici belli bir minimum hızda kaydırılmak zorundadır, aksi halde veriler geçersiz hale gelir. Saklanmış verilerin kaydedicinin sonundan kayıp gitmesini ve kaybolmasını önlemek için dolaştıran bir kaydırma kaydedici yapmak üzere çıkış girişe bağlanabilir. Böylece, verileri taze tutmak için, veriler bu halka üzerinde kaydırılarak dolaştırılıp durulur.

CCD bellekler, kısa halka ve uzun halka olmak üzere halen iki formatta gelmektedir. Kısa halka tiplerin her halkasında tipik olarak 128 veya 256 bit vardır ve uzun halka tiplerin her halkasında 1024 ile 4096 arasında bit vardır.

Uzun halka tipinin bir örneği, toplam 65536 bit saklayabilen Texas Instruments TMS 3064 tür. İçinde herbiri 4096 bitlik 16 kaydırma kaydedicisi vardır. Dört adres hattı 16 kaydediciden birinin seçilmesine ve seri şekilde okunup yazılmasına imkân verir. Ayrı bir 4096 bitlik kaydedici, algılama ve yeniden üretme devreleri için referans olarak kullanılmaktadır. Her saklanan bit her 4 ms de en az bir defa tazelenmelidir. Bu, her kaydediciye yeniden üretme devresi üzerinden saat gönderilerek yapılır. İki-fazlı 12 V luk giriş saatinden dahili olarak yedi saat fazı üretilmektedir.

Seri bellekler için önemli bir parametre gecikme süresidir. Bu, saklanmış veri bloğunun ilk bitine ulaşıncaya kadar geçen zamandır. Veri sayfaları genellikle ortalama gecikme süresini verirler. TMS 3064 için bu süre yaklaşık 410 µs dir. Bu sürenin böyle uzun olması istenilen başlama bitine ulaşıncaya kadar birçok bitin kaydırılması gerekebileceği içindir. Veri bloğunun ilk biti bir defa bulunduktan sonra bir sonraki bite erişim zamanı TMS 3064 için sadece 200 ns kadardır (aygıtın 5 MHz lik saat kullandığı kabul edilerek).

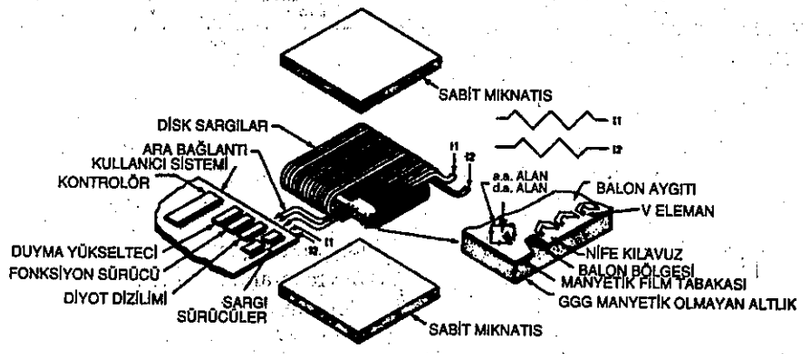
Uzun gecikme süresi olmayan dinamik RAM belleklerdeki hızlı gelişmelerden dolayı CCD bellekler beklenildiği kadar tutulmamışlardır. Büyüme belirtileri gösteren bir yeni seri bellek tipi manyetik balon belleğidir.

MANYETİK BALON BELLEKLER (MAGNETIC BUBBLE MEMORIES)

Bu bölümde henüz anlatılmış olan oku-yaz bellekleri silikon tabanlıdır ve verileri mandallarda veya elektrik yükleri olarak saklar. Yeni bir teknoloji 'garnet'i (çoğunlukla zımpara kağıdında kullanılan kırmızımsı turuncu bir mineral) kullanır ve verileri mikroskobik mıknatıslar üzerinde saklar. Bu teknoloji ile tek bir entegre paketine 1 milyon bitten fazlasını saklama imkanı olacağı beklenmektedir.

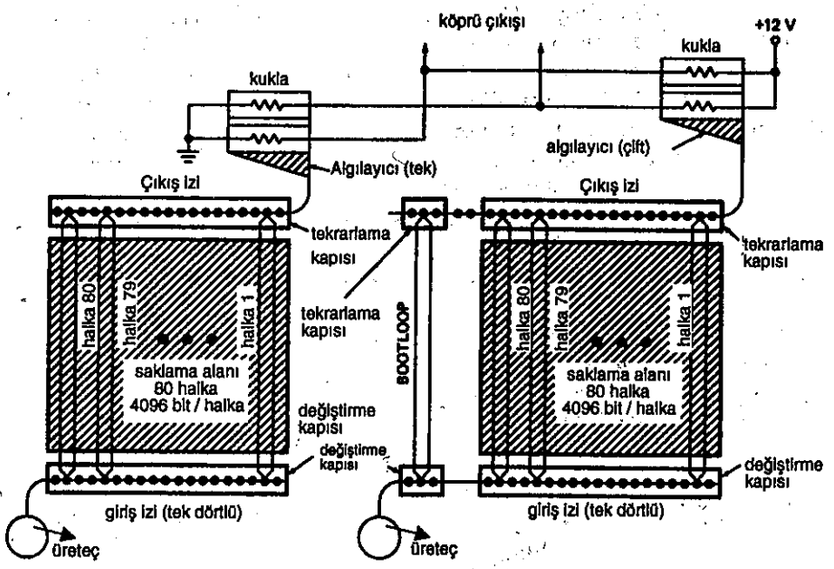
YAPI

Şekil 5.9, tipik bir manyetik balon belleğin (MBM) yapısını göstermektedir. Altlık gadolinyum-galyum garnet, yani kısaca GGG dir. Bu atlık üzerine ince bir tabaka saf manyetik garnet büyütülür. Bu yonganın altına ve üstüne kuvvetli sabit mıknatıslar, akı çizgileri yongaya dik olacak şekilde, yerleştirilir. Bu kuvvetli mıknatıslar, manyetik garnetteki mikroskobik manyetik bölgelerin çoğunun akı çizgilerine paralel ve onlarla aynı yönde dizilmesine sebep olur. Ancak bir çok bölge ters yönlü kalır ve bu harici alan tarafından küçük, yuvarlak manyetik balonlar haline sıkıştırılır. Bu balonlar küçük bir elektromıknatıs ile oluşturulabilirler veya yok edilebilirler ve garnet malzemenin içinde kolayca taşınabilirler. Çok ince nikel-demir yani permalloy mıknatıslar garnetin üzerine "V" deseninde konur. Balonlar bu küçük mıknatısların altına dizilirler. Balonlar, bir küçük mıknatıstan diğerine manyetik alanı döndürerek kaydırılır. Bu alan, Şekil 5.9 da gösterilen dikgen (ortogonal) sargılardan akım geçirilerek elde edilir. (Dikgen, iki sargının birbirine dik açılarda sarıldığını ifade eder.)



Şekil 5.9 Bir manyetik balon bellek aygıtının yapısı *(Texas Instruments).*

Sayısal veri saklarken bir balon 1 i belirtir, balonun yokluğu da 0 ı belirtir. Şekil 5.10, yaklaşık $1\frac{3}{4}$ inçlik (4,4-cm lik) kare pakette 1 milyon bit saklayan Intel 7110 manyetik balon aygıtının saklama mimarisini göstermektedir. Elemanın dört kısmı vardır. Her kısımda herbiri 4096 bitlik 80 halka vardır. Bu, 1 megabitlik bellekten 16 halka fazladır, fakat ekstra halkalar iki amaca hizmet eder. Birincisi, bu ekstra halkalar üretim verimini artırır. Eğer kısımlardan birinde bazı bozuk halkalar varsa aygıt içindeki programlanabilir mantık ile bunlar atlanır. (Bir makina her kısmı test edip iyi halkaların bir haritasını bir halkaya otomatik olarak programlayabilir.) İkincisi, ekstra saklama halkaları hata bulma ve düzeltme sistemleri için kullanılabilir.



Şekil 5.10 Intel 7110 MBM saklama mimarisi.

Balonlar halka içerisinde X ve Y dikgen alan sargıları ile döndürülür. Bir algılayıcının yanından geçen bir balon dış devre tarafından okunabilen birkaç milivoltluk bir darbe oluşturur. Okuma işlemi tahripsizdir. Veri, halkaya bir tekrarlayan üreteçle girilir. Üreteçler bir 1 için bir balon içeri atarlar, 0 için balon içeri atmazlar. Veri yazılmadan önce, bir halka daha önce saklanmış ve-

rilerin bütün balonlarından temizlenmelidir. Eğer bunu canlandırmakta zorluk çekiyorsanız, bir taşıyıcı bant üzerinde kaydırılan manyetik bilyeleri düşünün. Intel 7110 un ortalama gecikme süresi birkaç milisaniyedir. Saklanmış veri bloğunun ilk biti bulunduktan sonraki veri transfer hızı 100 K bit/sn dir.

DIŞ KONTROL DEVRESİ

Intel 7110 gibi manyetik balon bellekler, hem analog hem de sayısal kontrol devreleri gerektirir. Analog kontrol devresi, X ve Y sargılarının sürücülerini içerir. Bu devre ayrıca algılayıcılardan gelen milivolt seviyesindeki sinyalleri mantık seviyelerine çıkaran yükselteçler ve mantık seviyesi sinyalleri uygun iç seviyelere çeviren tamponlar içerir. Intel'in hazırladığı BPK-70, bir 1 megabitlik aygıt ve gerekli bütün analog bağdaştırma devresini içeren bir baskılı devre kartıdır.

Sayısal kontrol devresi oldukça karmaşıktır. İlk kısmı, aygıttaki hangi halkalara yazıldığını belirler. Ayrıca, zamanlama için bir devre ve seriden-paralele veri çevrimi için bir UART gereklidir. Çoğunlukla giriş-çıkış transferlerini düzenlemek için bir mikroişlemci kullanılmaktadır.

AVANTAJLAR VE DEZAVANTAJLAR

MBM in avantajları şunlardır:

1. Bir tek entegre paketinde çok sayıda, bir milyon veya daha fazla, bit saklayabilir.
2. *Saklama statiktir (tazeleme gerektirmez)* ve uçucu değildir. Küçük V mıknatıslar güç kesildiğinde balonları yerlerinde tutar.
3. Motorlar ve birçok garip mekanik parçalar gerektiren manyetik teyp ve disk sistemleri gibi toplu saklama sistemlerine göre MBM ler daha az zorluk çıkarırlar. Fujitsu, manyetik balon bellekleri içeren takılabilir kasetler kullanan bir bellek sistemi satmaktadır.

MBM lerin dezavantajları şunlardır :

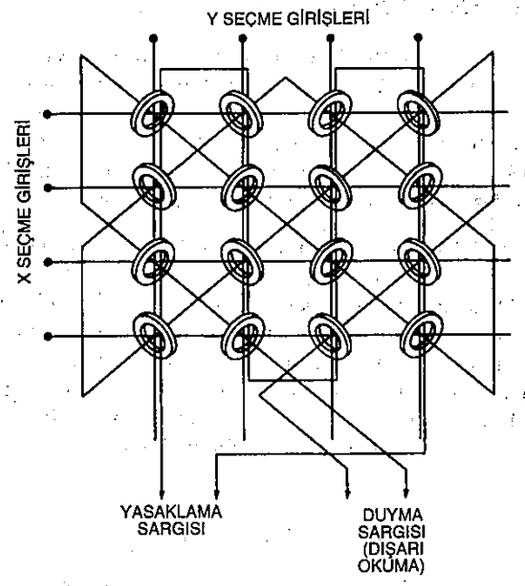
1. MBM ler çok yavaştır. 100 K bit/sn lik tipik kaydırma hızıyla, bir veri bitine ortalama gecikme süresi, TMS 3064 gibi bir *CCD* nin ortalama gecikme süresinin yaklaşık 10 katıdır. Hızlı erişimin gerektiği yerlerde bu aygıtlar hiç uygun değildir.
2. Kontrol devresi, ROM ların, RAM ların ve *CCD* lerinkinden çok daha karmaşıktır.

MANYETİK ÇEKİRDEKLİ BELLEK

Manyetik çekirdekli bellek büyük boy bilgisayar belleklerinde çok kullanılmaktaydı, fakat şimdi yeni tasarımlarda sadece aşırı sıcaklık ve radyasyon ortamı ile karşılaşılacaksa kullanılmaktadır. Çekirdekli bellekteki saklama elemanı küçük bir manyetik ferit boncuktur. Boncuk simit şeklindedir ve dış çapı yaklaşık 0,050 inç (1,27 mm) ve ortasındaki deliğin çapı 0,030 inç (0,76 mm) dir. Boncuğun deliğinden bir elektrik akımı taşıyan bir tel geçirilirse, ferit boncuk yani çekirdekte bir manyetik alan oluşur. Endüklenen manyetik alanın yönü, akımın akış yönüne bağlıdır. Alanın yönünü belirlemek için sol el kuralı kullanılabilir. Baş parmağınız elektron akış yönünü gösterirse, diğer dört parmağınız çekirdekteki manyetik akımın yönünü gösterecektir. Boncuk bir defa bir yönde mıknatıslandı mı, akım kesilse bile bu mıknatıslılığını korur. O halde bu uçucu olmayan saklamadır. Bir 1, keyfi olarak akının belli bir yönde olması ile ve bir 0 da akının ters yönde olması ile tanımlanabilir.

Bir çekirdekten veri okumak için, delikten bir duyma teli geçirilmiş ve bir duyma yükseltecine bağlanmıştır. Daha sonra yazma telinden uygun yönde bir akım geçirilerek çekirdeğe bir 0 yazılır. Eğer okunmakta olan çekirdek 1 durumunda ise, 1 den 0 a geçişteki akı değişimi, duyma telinde bir gerilim endükler. Çekirdek 0 da ise, yeniden çekirdeğe 0 yazmak duyma telinde hemen hemen hiç bir gerilim endüklemez. Bu okuma tahriplidir, çünkü okunduktan sonra çekirdek her zaman 0 durumundadır. Bir okumadan sonra veriyi kaybetmemek için, veri bir mandala saklanıp tekrar çekirdeğe yazılmalıdır.

RAM tipi veri saklaması, Şekil 5.11 de gösterildiği gibi, çekirdekleri X - Y matrisi şeklinde telle bağlayarak yapılabilir. Bu dizilimde belli bir çekirdeği okumak için çekirdeğin X hattına 0 yazmak için gerekli akımın yarısına eşit bir seçme akımı gönderilir. Yine bu yarım 0 yazma akımı çekirdeğin Y hattına gönderilir. Yarım akımlar, X ve Y hatlarındaki diğer çekirdeklerin durumlarını değiştirecek kadar büyük değildir. Duyma yükselteci akı değişimini adreslenen hücredeki bir 1, akı değişmemesini de bir 0 olarak algılar. Matristeki bir çekirdeğe veri yazması, 1 mi yoksa 0 mı yazılacağına göre iki farklı mekanizma gerektirir. 1 yazmak için, çekirdek, pozitif bir yarım 1 yazma akımı ile X ve Y hatlarından adreslenir. İkisinin etkisi toplanarak bir 1 yazılması gerçekleştirilmiş olur. Yarım 0 yazma akımını X ve Y hatlarından göndermekle, çekirdeği okurken yapıldığı gibi, bir çekirdeğe 0 yazılabilir. Bu ikisinin etkisi birlikte çekirdeği diğer yönde mıknatıslayacak ve bir 0 yazacaktır. Ancak, genellikle bir 0, *yasaklama (inhibit)* hattı denilen bir başka hat kullanılarak yapılır. Bu hat, bir matris yani düzlem üzerindeki bütün çekirdeklerden geçirilmiştir. Çekirdek 1 yazıldığı zamanki gibi adreslenir, fakat yasaklama hattına gönderilen negatif bir yarım 1 yazma akımı seçme akımlarından bir tanesinin etkisini yok eder ve bir 1 in yazılmasını önler. Bu yöntem daha kolay kontrol sağlar.



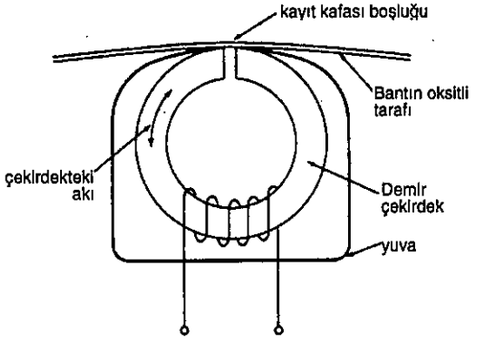
Şekil 5.11 Manyetik çekirdekli bellek düzlemine yazma şeması.

Çekirdek düzlemleri, bellek miktarını artırmak için üç-boyutlu diziliml-er oluşturacak şekilde üstüste yığılabilir. Çekirdek belleğin avantajları, sağlamlığı, sıcaklık ve radyasyona dayanıklılığı ve uçucu olmamasıdır. Bu belleklerin bir problemi ise fiziksel hacmidir. Bir çekirdekli bellek düzlemi dizilimi, dengi olan modern yarıiletken bellekten çok daha büyüktür. Çekirdekli bellek, 2 inç x 2 inç (5,1 cm x 5,1 cm) lik bir çekirdek düzleminde 4096 bit içerir. 4096 x 8 lik bir bellek elde etmek için bunlardan 8 tanesi, 2 inç (5,1 cm) lik bir küp oluşturmak üzere üstüste yığılabilir .Mostek MK4801 gibi 16 bacaklı statik MOS RAM lardan dört tanesi bu manyetik çekirdekli bellek yığını yerine, çok daha ucuz bir fiata, konabilir. Ayrıca, çekirdekli belleklerdeki okumalar tahribatlıdır ve verinin saklanıp yeniden yazılması isteniyorsa gerekli devreler eklenmelidir. Diğer bir problem de çekirdekli belleklerin seçme hatlarında yüksek akımlı sürücüler gerektirmeleridir. Birçok kullanımda çekirdek erişim süresini kısaltmak için seçme hattı akımları 1 A e kadar çıkarılabilmektedir.

MANYETİK BANTTA SAKLAMA

Manyetik bantta saklama, büyük miktarlardaki sayısal verinin bit başına düşük bir maliyetle saklanması için önemli bir yöntemdir. Bant genellikle üzerinde ince bir tabaka manyetik demir oksit bulunan Mylar'dan yapılır. Philips Portatif kaset kayıt cihazlarında kullanılan 0,15 inç (3,8 mm) lik bantlardan büyükboy bilgisayarlarda çok kullanılan 2 veya 3 inç (5,1 veya 7,6 cm) lik geniş bantlara kadar değişik ende bantlar vardır.

Şekil 5.12, bir banta veri yazmak veya banttan veri okumak için kullanılan bir *kayıt/çalma (record/play)* kafasının şemasını göstermektedir. Yazma modunda, sargıdan geçen bir akım, kafanın demir çekirdeğinde bir manyetik akı yaratır. Kafadaki bir açıklık, manyetik akının dışarı çıkmasına ve bant üzerindeki manyetik malzemeyi mıknatıslamasına imkan verir. Bir defa mıknatıslanınca, bant tekrar değiştirilinceye kadar öyle kalır. Bant üzerindeki mıknatıslanmış bölgenin yönü, kafadaki akımın akış yönü ile belirlenir. Bir yöndeki akım akışı 1 i, diğer yöndeki de 0 ı temsil eder.

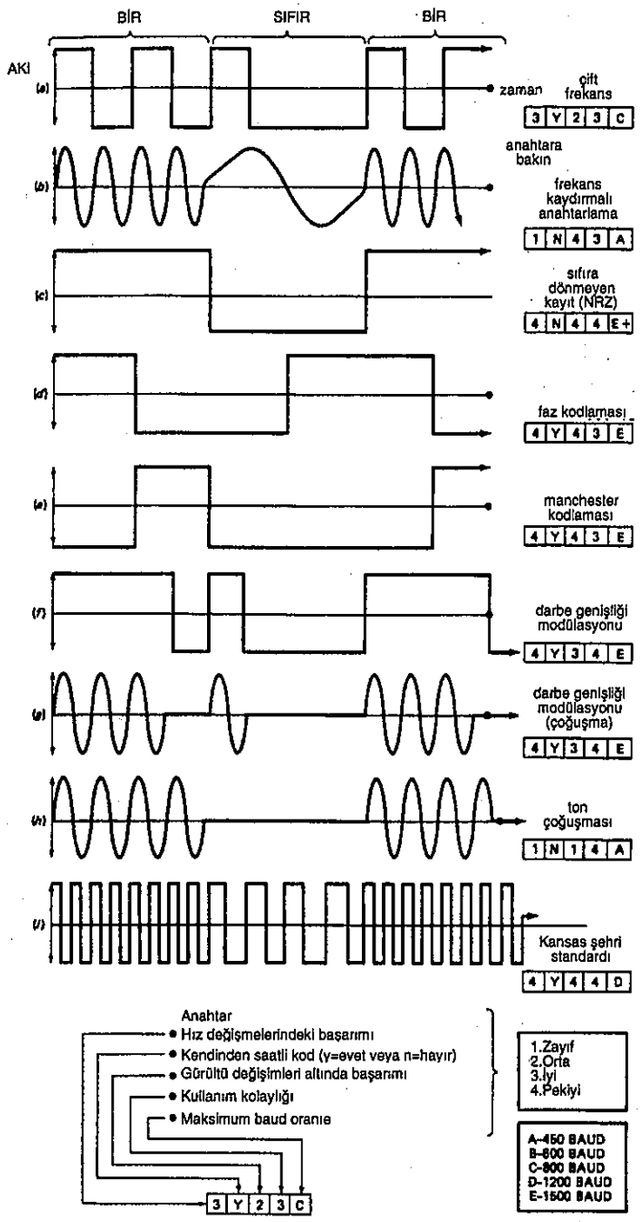


Şekil 5.12 Bir manyetik bant için kayıt/çalma kafasının yapısı.

Veriler aynı kafayla banttan okunabilirler. Bantın mıknatıslanmış bölgesi kafadaki açıklık üzerinden geçtiğinde, tipik olarak birkaç milivoltluk bir gerilim sargıda endüklenir. Endüklenen gerilimin yönü, okunan mıknatıslanmış bölgenin yönüne karşılık gelir. Bu küçük sinyali, bir yükselteç mantık seviyelerine yükseltir.

KAYIT MODLARI

Pratikte, bir 1 veya bir 0 ı bant üzerinde mıknatıslanmış bölgeler olarak temsil etmenin bir çok yolu vardır. Şekil 5.13 bunlardan dokuzunu göstermektedir. (Her dalgaşeklinde, yatay zaman çizgisi dalgaşeklinin 0 akı seviyesinde çizilmiştir.) Birincisi, çift frekans ,yani F2F, bir 1 i bir bit zamanı içinde iki darbe



Şekil 5.13 Manyetik bant kayıt modları.

ile, bir 0 ı bir bit zamanı içinde bir darbe ile temsil eder. Bu sisteme kendinden saatli denir, çünkü veri darbelerinden hız bilgisi türetilebilir. Hız bilgisi, bir UART ı veri ile senkronize tutmak için kullanılabilir.

İkinci yöntem *frekans kaydırmalı anahtarlama (frequency shift keying :* FSK) dir. Burada, bu frekanstaki bir sinyal parçası 1 i, bir başka frekanstaki bir sinyal parçası da 0 ı temsil eder. Bu yöntem kendinden saatli değildir, çünkü iki frekans arasında bir harmonik ilişkisi yoktur.

Sıfıra-dönmeyen (non-return-to-zero : NRZ) kayıt yönteminde bir 1 akının bir yönü ve bir 0 da akının diğer yönü olarak saklanır. *NRZ* aynı zamanda bantı ya bir yönde veya diğer yönde mıknatıslayan herhangi bir kayıt modu için de kullanılır. Şekil 5.13a daki *F2F* kaydı, gördüğünüz gibi 0 akı seviyesi dalgaşeklinin ortasından geçtiği için, *NRZ* modundadır. Bu yöntem bir çok IBM-uyumlu teyplerde kullanılmaktadır. Şekilde gösterilmemiş olan bir sıfıra-dönen (return-to-zero) kayıt yöntemi bir 1 i akı ile ve bir 0 ı akının olmaması ile saklar. *NRZ* verileri kendinden saatli değildir. Bir alıcı sistemi senkronize etmek için ayrı bir saat izi gerekmektedir.

Faz kodlaması (phase encoding) yöntemi, bit zamanının ortasında akı geçişi kullanır. Bir yöndeki akı değişimi bir 1 dir ve diğer yöndeki bir 0 dır (Şekil 5.13d).

Diğer bir faz kodlaması türünün adı Manchester'dir. Burada akı geçişi bit zamanının başındadır. Bit zamanının başındaki bir yüksekten alçağı geçiş bir 0 ı, geçiş olmaması da bir 1 i gösterir.

Darbe genişliği modülasyonu (pulse width modulation) bir 1 i temsil etmek için yüksek doluluk oranlı, yani geniş, bir darbe kullanır ve bir 0 ı temsil etmek için alçak doluluk oranlı, yani dar, bir darbe kullanır (Şekil 5.13f).

Darbe genişliği modülasyonu çoğuşma (burst) ve ton çoğuşması (tone burst) yöntemleri, Şekil 5.13*g* ve *h* de gösterildiği gibi 1 ve 0 ları temsil etmek için kapılanmış sinüs dalgalarını kullanırlar. Bu ve sinüs dalgası kullanan diğer teknikler standart doyuma varmayan ses kayıt cihazlarında kullanılabilir. Bu tür bir kayıt cihazı diğer tekniklerin gerektirdiği doymuş darbeleri kaydetmez.

Kansas Şehri Standardı (Kansas City Standard) bir çok amatör sistemde kullanılmaktadır. Bu yöntemin standart 300-baudluk tipi, bir 1 i temsil etmek için sekiz tane 2400 Hz periyodu, bir 0 ı temsil etmek için dört tane 1200 Hz periyodu kullanır. Dalgaşekilleri genellikle sinüs dalgaları şeklinde veya sinüse benzeyecek şekilde filitrelenmiş yapıdadır, böylece standart ses kayıt cihazları kullanılabilir. Kansas Şehri Standardı ayrıca seri veri kaydı için de formatı belirler. Yol göstericinin (seri veri bloğunun ön tarafına konan kısım) 30 sn lik 2400 Hz lik sinyalden oluşmasını söyler. Her veri karakterinin bir 0 başlama biti, parite dahil 8 veri biti ve iki 1 durma bitinden oluşacağını söyler. Veri blokları 5 sn lik 2400 Hz yol göstericilerle ayrılmalıdır.

BANT ÜZERİNDEKİ VERİ FORMATLARI

Bant üzerine bitlerin çeşitli şekillerde saklanmasının yanısıra, bitlerin kelimeler halinde saklanmasında kullanılan birçok format vardır. Geniş bantlı büyük ticari kayıt cihazlarında veri kelimeleri, paralel izler üzerinde paralel şekilde saklanır. Çok-kısımlı okuma/yazma kafaları bir kelimenin bütün bitlerini aynı anda okur veya yazar. Bir standart IBM formatı her kelimeyi kaydetmek için yedi paralel iz kullanır. Yedi izli şeritlerden birkaç tanesi aynı bant üzerine konabilir.

Philips kasetlerindeki gibi dar bantlı sistemlerde sadece bir veya iki iz vardır, bu nedenle veri kelimeleri tek bir iz üzerinde seri şekilde saklanır. Seri kelimeler başlama ve durma bitleri dahil 8 ile 19 arasında bit içerebilir. Bu kelimeler, 256 veya 512 kelimelik bloklar yani kayıtlar halinde gruplanır. Kayıtlar, *boş banttan oluşan IRG (Interrecord gaps)* lerle ayrılmıştır. İlgili verilerin iki veya daha fazla bloğuna dosya (file) denir. *Dosya-sonu (end-of-file : EOF)* karakterleri bir dosyanın sonunu işaretler. IRG den sonra gelen bir *dosya-başı (beginning-of-file)* karakteri bir sonraki dosyanın başladığını belirtir.

BANTTA SAKLAMANIN AVANTAJLARI VE DEZAVANTAJLARI

Bir bantta saklama sisteminin ana avantajı çok fazla sayıda bitin, bit başına çok küçük bir maliyetle saklanabilmesidir. Bantta saklama uçucu değildir ve bant tekrar tekrar silinip yeniden kullanılabilir.

Bantta saklamanın ana dezavantajı seri erişimdir. Yavaş bir sistemin bir dosyayı bulması bir çok saniye alabilir. Ancak bir defa dosya bulundu mu bazı paralel izli kayıt cihazları saniyede 250 000 karakter dökebilir. Uzun bir bant yerine bir çok kısa kaset kullanarak ilk erişim zamanı kısaltılabilmektedir. Büyük ticari cihazların diğer bir dezavantajı, bant hızını sabit tutmak ve bandı koparmadan durdurup çalıştırmak için karmaşık mekanik sistemlere gerek olmasıdır.

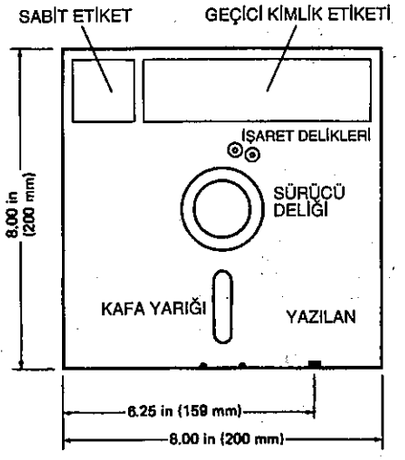
MANYETİK DİSKTE VERİ SAKLAMA

Manyetik bantta veri saklamadaki seri erişim yani gecikme zamanı problemi, verileri, üzeri manyetik olarak kaplanmış bir plağa benzer disk üzerine koyarak kısmen çözülebilir. Veriler bu diskler üzerinde, bir plaktaki gibi spiral şekilde değil ayrı ayrı daire şeklindeki izler olarak saklanır. Okuma-yazma kafası, verinin okunacağı ize, bütün önceki izleri okumaya gerek kalmadan, atlatılabilir.

Disk üzerindeki manyetik malzeme ve okuma-yazma kafasının yapısı manyetik banttakilerle aynıdır. İzleyen kısımlarda çok kullanılan iki disk türünü, yumuşak diskleri ve sert diskleri, anlatacağız.

YUMUŞAK DİSK (FLOPPY DISK) SİSTEMİ

Bir yumuşak disk sistemi, manyetik malzeme ile kaplanmış ve Şekil 5.14 te gösterildiği gibi 8 inçlik bir az sürtünmeli zarf içine konmuş $7\frac{3}{4}$ inç çapında bir Mylar disk kullanır. Mylar disk sadece onda 1 mm kadar kalınlıktadır, bu nedenle de yumuşak (floppy) adını alır. Disk, sürücü birime sokulduğunda, ortadaki büyük delikten sıkıştırılıp dakikada 360 devirle döndürülür. Okuma-yazma kafası zarf üzerindeki ince uzun açıklıktan diske dokunur. Yumuşak disklerde kafalar okuma veya yazma sırasında diskle sürekli temas halindedir. Bu diskin aşınmasını artırır, fakat yumuşak disklerin daha yavaş hızlarından dolayı gereklidir.



Şekil 5.14 Koruyucu zarflı yumuşak disk.

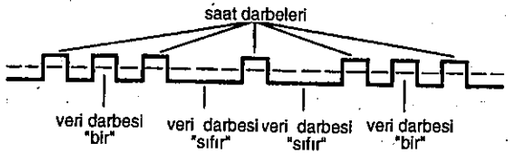
Diskteki yazılanları koruma çentiği, teyp kasetlerindeki kırılabilen plastik kulaklar gibi, saklanmış verileri yeniden üzerlerine yazılmaktan korumak için kullanılabilir. Bir fotoalgılayıcı (LED ve fototransistor birlikte) çentiğin var olup olmadığını belirleyebilir ve çentik varsa yazma devresini yetkisiz

denetim-toplamı, veya çevrimsel-tekrarlama, baytından oluşur. Denetim toplamı (check-sum) baytları, bütün veri baytlarını toplayarak ve iki baytı aşan eldeleri ihmal ederek oluşturulur. Sonuçta elde edilen 2-baytlık toplam, veriden sonraki iki baytlık konuma saklanır. Veriler okunurken, yeniden toplanırlar ve toplam, kaydedilmiş denetim-toplam baytları ile karşılaştırılır. Kaydedilmiş verilerde hata yoksa, iki toplam uyuşur. Veri hatalarını kontrol etmek için diğer bir yöntem çevrimsel-tekrarlama kontroludur (cyclic-redundancy control: *CRC*). Bu yöntemde veri baytları dizisi bir sabite bölünür. Bu bölmeden elde edilen kalan, 2 *CRC* baytı veya karakteri olarak yazılır. Veri okunurken, bu 2 bayt, veri dizisinden çıkarılır. Sonuç, orjinal sabite bölünür. Eğer verilerde hata yoksa, bu bölme sıfır kalan vermelidir. *CRC*, diskte saklamada hata kontrolu için çoğunlukla kullanılan yöntemdir.

Veriler bir diske yazılabilir ve hemen gerisin geriye okunup hatalar için test edilebilir. Bir hata bulunursa, veriler yeniden yazılabilir ve yeniden test edilebilir. Eğer 10 yazma denemesi sonunda başarılı olunamazsa, bu takdirde bir dış devre başka bir sektöre yazmayı dener. Böyle bir çalışma şekli, Philips kaset kayıtlarından 1 milyon defa daha iyi yumuşak disk hata oranı vermektedir.

Veri alanı, bir veri alanının sonucu ile bir sonraki sektörün başlama işareti arasında tampon olarak görev yapan kayıt boşluk alanı tarafından izlenir. Bu boşluk, ayrıca sistem bir sektöre veri yazdıktan sonra bir sonrakinin ID alanını okumaya gittiğinde, sisteme yazma modundan okuma moduna geçmesi için zaman tanır.

Daha önce anlatıldığı gibi, IBM 3740-uyumlu yumuşak-sektörlü yumuşak disk formatı, her birinde 26 sektör olan 76 veri izine sahiptir. Her sektör 128 baytlık veri içerebildiğine göre, bir IBM-uyumlu diskte toplam veri saklama kapasitesi yaklaşık 250 000 bayttır. Bir sektöre ortalama erişim zamanı yaklaşık 500 ms dir ki bu da çoğu teyp sistemlerininkinden çok daha kısadır. Bir defa bir sektör veya dosya bulundu mu, veri diskten yaklaşık 250 000 bit/sn hızla okunabilir. IBM uyumlu bir disk üzerinde bitleri saklamakta kullanılan gerçek desen, NRZ şeklinde kaydedilmiş F2F tipindedir. Bu, Şekil 5.17 de gösterilmiştir. Her bit zamanı, bir saat darbesi içerir. Bir bit zamanında, bir 1 saat darbeleri arasına bir başka darbe koyarak yazılır; bir 0 ise bu darbeyi koy-



Şekil 5.17 Manyetik diskler için F2F kayıt modu.

mayarak yazılır. Bu sıfıra-dönmeyen türdendir, çünkü veri izindeki her bölgede ya bir yönde ya diğer yönde manyetik akı vardır. Şekil 5.17 deki dalgaşeklinin ortasından geçen 0 akı hattı bunu belirtmektedir.

DEĞİŞİK YUMUŞAK DİSKLER

Bir disk üzerinde saklanabilecek veri miktarını artırmak için çeşitli yöntemler kullanılmıştır. Birincisi, çift-taraflı yöntemdir. IBM in yaptığı gibi diskin tek tarafını kullanmak yerine, diskin her iki tarafı da kullanılır. Her bir taraf, hâlâ IBM yumuşak-sektör uyuşması için formatlanabilir. İkinci yöntem çift-yoğunluktur ve bu her izdeki sektör sayısını ikiye katlar. Her iki yöntem disk kapasitesini 500 000 baytın üzerine çıkarır, fakat çift-yoğunluk IBM uyumlu değildir. Bu iki yöntem birleştirilerek, bir tek 8 inç (20cm) lik disk üzerinde 1 milyon bayttan fazla saklama imkanı, çift-taraflı- çift-yoğunluklu kayıtla sağlanmıştır.

Disk saklama biriminin büyüklük ve maliyetini bir miktar azaltmak için mini-flopi sistem geliştirilmiştir. Mini-flopi sistemler, standart 8 inçlik flopilerle aynı türde disk ve sürücü mekanizma kullanırlar, fakat disk büyüklüğü sadece $5\frac{1}{4}$ inçtir. Mini-disketlerin, dizin izi dahil sadece 35-40 izi vardır. Bir 35-izli disk yaklaşık 80 000 bayt saklayabilir. Çoğu mini-flopi sistemler, kaybedilen saklama alanını, çift-taraflı kayıtla, çift-yoğunluklu kayıtla veya her ikisini birden kullanarak telafi ederler. Flopi diskleri ve bunların bilgisayar ve mikroişlemcilerle kullanışlarını daha sonra anlatacağız.

YUMUŞAK DİSKLERDE ÖN TEDBİRLER

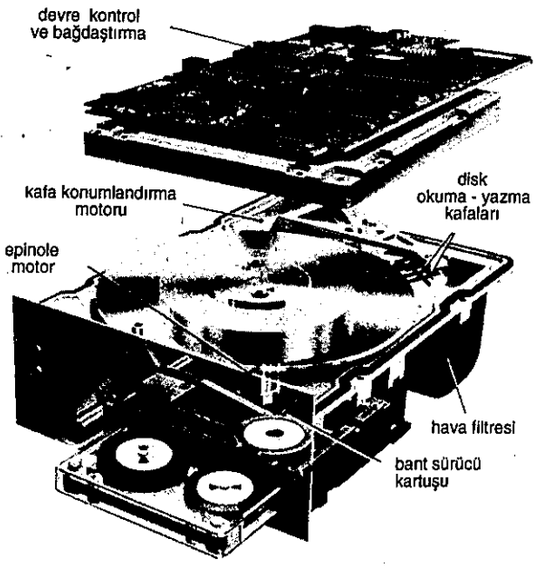
Manyetik diskler; kir, ısı veya manyetik alanlar tarafından zarar görmeye açıktır. Kayıt yüzeyinin açık kısımlarına parmaklarınızla dokunmayın. Kullanmıyorsanız diski kutusunun içinde saklayın. Diskler, diğer elektronik cihazardan uzak tutulmalıdır. Kazara bir güç kaynağının üzerine konan bir disk, bir transformatörün değişen manyetik alanı tarafından silinebilir. Ayrıca, disk, sürücü içinde iken, disk sürücü biriminin gücünü kapatmayınız. Bazı birimlerde, formatlama devreleri veya motorların çökmekte olan elektrik alanları bu disk üzerinde saklanmış verileri bozabilir.

SERT DİSKLER (HARD DİSKS) : WINCHESTER SÜRÜCÜLERİ

Sert disklerin de bir veya iki tarafı manyetik malzeme ile kaplıdır. Çoğu hallerde, yumuşak diskler gibi takılıp çıkarılmazlar, sürücü ünite için sabit

olarak konmuştur. Bazı standart disk büyüklükleri $5\frac{1}{4}$, 8, $10\frac{1}{2}$, 14,20 ve 40 inçtir. Her sürücünün saklama kapasitesini artırmak için diskler, aralarında boşluk ve bu boşluklarda her diskin kendi okuma-yazma kafası olmak üzere üst üste konabilir. Sert diskler ,boyutları yönünden daha sabit olduklarından, yumuşak disklere göre inç başına daha fazla ize ve iz başına daha fazla bite sahiptirler. Bu, sert disklerin, aynı büyüklükteki yumuşak disklere göre, daha fazla veriyi kolayca saklayabileceği anlamına gelir. Örneğin $5\frac{1}{4}$ inçlik birimler disk başına 3 ile 10 milyon bayt arasında saklayabilirler. Bu miktar disk başına 8 inçlik birimlerde 5 - 20 milyon, $10\frac{1}{2}$ inçlik birimlerde 30-50 milyon ve 14 inçlik birimlerde 40-100 milyon arasındadır.

Sert diskler dakikada 1000 ile 3600 devir hızla döndürülürler. Bu yüksek hız, sadece daha hızlı okuma ve yazmayı mümkün kılmaz, ayrıca okuma-yazma kafasını disk yüzeyinden 10μ inç kadar uzak tutan ince bir hava yastığı oluşturur. Kafa arızalanmadığı sürece diskin veri bölgesine hiç dokunmaz, böylece aşınma en aza indirgenmiş olur. Genellikle 10μ inç ten daha büyük



Şekil 5.18 Irwin $5\frac{1}{4}$ inçlik Winchester sert disk sürücü.

(Irwin International, Inc.)

olan toz ve duman parçacıklarının disk kafasını bozmaması için, ya diskler hava almayacak şekilde tozsuz bir ortam içinde kutulanmışlardır veya soğutma havası çok ince gözenekli bir hava filitresinden geçirilmektedir.

Şekil 5.18, bir Irwin $5\frac{1}{4}$ inçlik Winchester sert disk sürücüsünün bir fotoğrafıdır. Bu sürücülere Winchester adının verilmesinin, her sürücüde 30 milyon bayt saklaması planlanan eski bir IBM çift-sürücülü üniteden kaynaklandığı söylenir. 30-30 yapısı herhalde birisine Winchester tüfeklerini hatırlatmıştı ve bu isim tutuldu.

Kafalar istenilen iz üzerine, bir adımlı motor ve kılavuz vida veya Şekil 5.15 te yumuşak diskler için gösterilen çelik bant veya lineer bir aktuatör (eyleyici) yardımıyla konumlandırılır. Çeşitli $5\frac{1}{4}$ inçlik birimler için konumlandırma süresi 25 ile 150 ms arasında değişmektedir. Bir defa istenilen ize ulaşıldı mı, veriler 500 000 ile 900 000 bayt/sn hızla okunur. Bağdaştırma devresi, okuma ve yazma işlemlerinde kafanın konumlandırılmasını kontrol eder ve giriş verilerini saklamak için formatlar.

Sert disklerin avantajı, çabuk erişilebilir büyük miktarda veriyi saklayabilmeleridir. Örneğin, sert diskler bir kelime işlemci ile birlikte kullanılırlarsa, bunun gibi bütün bir kitap, bir yığın mini-flopi disk üzerine saklamak yerine, bir disk üzerinde saklanabilirdi.

Bir sert disk sisteminin bozulması halinde verilerin kaybını önlemek için, disk üzerindeki bütün dosyaların yedekleme kopyaları, manyetik bant veya flopi diskler gibi başka bir ortamda saklanır. Şekil 5.18 de gösterilen Irwin 510 un içine konulmuş bulunan teyp birimi gibi yüksek hızlı teyp birimleri, 4 dakikadan az bir süre içinde diskten 10 milyon bayt okuyup kendilerine yükleyebilir veya diske 10 milyon bayt yükleyebilirler. Bazı birimler yedekleme problemini, kapalı disk kutusunu sökülüp takılabilir yaparak çözmektedir. Fakat bu, hızlı teyp yaklaşımından daha pahalıdır.

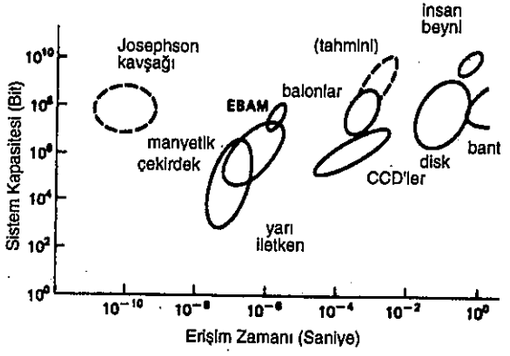
ELEKTRONİK IŞINLA ERİŞİLEN BELLEK

Elektronik ışınla erişilen bellekler (electronic beam accessed memory : EBAM) bir televizyon resim tüpüne benzer şekilde yapılırlar, fakat ekran bir MOS yongalarının dizilimi şeklindedir. Veri, yonga üzerinde adreslenen yere bir yük paketi olarak saklanır. Yük buraya bir elektron ışınıyla yerleştirilir. Işın, osiloskop veya televizyonda yatay ve düşey sapmalar için kullanılan yöntemlerin aynısı kullanılarak belli bir adrese yönlendirilebilir. Veri, ışını belli bir hücreye adresleyerek ve ışın akımının akışına bakarak okunur. Eğer saklama hücresinde bir yük paketi varsa daha az ışın akımı akacaktır. Okuma tahrip edicidir ve saklama uçucudur.

EBAM lar, çeşitli güç kaynakları gerektiği için biraz büyükçedir, fakat her birim 30-40 milyon bit saklayabilir. Erişim zamanı sadece onlarca mikrosaniyedir.

SAKLAMA KAPASİTESİNE GÖRE ERİŞİM ZAMANI

Şekil 5.19, bu bölümde anlatılan çeşitli bellek türlerinin saklama kapasitesi ile erişim zamanlarını bir grafik olarak karşılaştırmasını göstermektedir. Bu grafik her türün kullanım aralığını hatırlamanıza yardımcı olacaktır. Ayrıca, her birinin yanına saklamanın uçucu olup olmadığını ve rastgele veya seri erişilebilir tür olduğunu yazabilirsiniz. İnsan beyni ile Josephson eklemlerinin grafiğin ayrı uçlarında olduğuna dikkat ediniz. Verilen bir uygulamada hangi tür bellek kullanılacağını belirleyen faktörler maliyet, erişim zamanı, kolay bulunması, saklama kapasitesi, erişim kolaylığı (seri veya rastgele), uçuculuk ve fiziksel büyüklüktür.

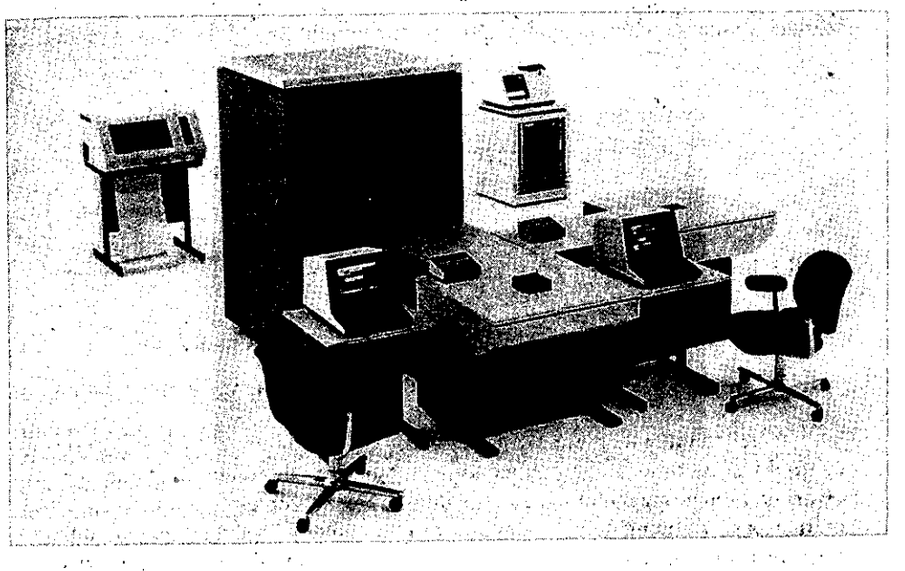


Şekil 5.19 Çeşitli bellek türleri için erişim zamanı ve saklama kapasitelerinin karşılaştırılması.

YARİİLETKEN BELLEKLERİN TEST EDİLMESİ

Yarıiletken belleklerin fabrikadaki testleri parametrik ve işlevsel olmak üzere iki temel tiptedir. Parametrik test maksimum ve minimum gerilim seviyeleri ile giriş ve çıkış akımlarının ölçülmesini içerir. Bu test, ayrıca, erişim zamanlama ilişkilerinin ölçümlerini de içerir. En-kötü-hal zamanlama parametreleri ölçülürken, bellekteki her adres için ölçüm yapılmalıdır, çünkü matristeki değişik yerlere erişim zamanı genellikle farklıdır.

İşlevsel test, her adrese verinin yazılması veya programlanmasının, doğru okunmasının ve silinmesi veya yeniden programlanmasının gerçekleştirilip gerçekleştirilemediğini belirlemeyi içerir. Şüphesiz her adres ayrı ayrı test edilmelidir. İlk akla gelen, 2147 gibi 4 K lık bir statik RAM ın, her adrese bir 1 yazarak ve sonra her adresi okuyup hepsinden 1 okunup okunmadığını kontrol ederek ve daha sonra da aynı işlemleri 0 için yaparak test edilebileceğidir. Ancak, pratikte bir belleği sadece bütün 1 ler ve bütün 0 lar yöntemiyle test etmek yeterli değildir, çünkü çoğu bellekler desenlere hassastır. Bu, bazı desenlerin belleğe doğru programlanma veya yazılmasının mümkün olamayabileceği anlamına gelir. Bir belleğin desen hassasiyetini tamamen kontrol etmek için, teorik olarak bütün mümkün birleşimlerin denenmesi gerekir. 4 K lık bir RAM için mümkün birleşimlerin sayısı 2^{4096} dır. Bu kadar çok olasılığı test etmek imkansız olduğundan, bir problemin varlığını gösterebilecek çeşitli test desenleri kullanılmakla yetinilir. Önce, bütün 1 ler, bütün 0 lar veya değiştiren 1 ler ve 0 lar gibi sabit desenler kullanılarak büyük boyutlu hataların varlığı kontrol edilir. Sonra desen hassasiyetini kontrol etmek için bir "dörtnala" 0 veya 1 deseni kullanılabilir. Çoğunlukla "galpat" (galloping pattern) denilen bu desen önce bütün belleği 0 la doldurur. Sonra bir yere bir 1 yazılır ve belleğin gerisinde etkilenen başka bir yer olup olmadığı kontrol edilir. Eğer başka bir yer etkilenmişse entegre bozuktur; eğer etkilenme-



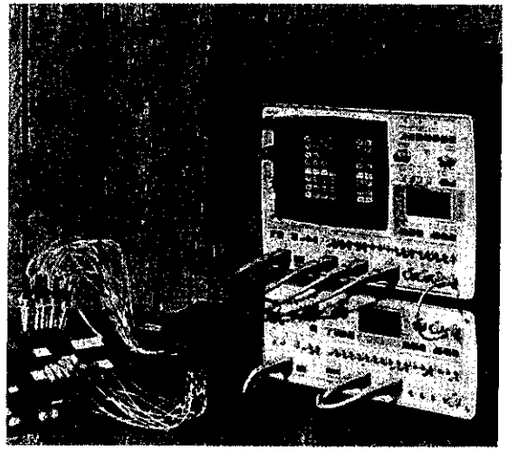
Şekil 5.20 Fairchild Sentry entegre devre test sistemi.

mişse, yazılmış olan 1 değiştirilerek 0 yapılır ve bir sonraki yere bir 1 yazılır. Belleğin gerisinde değişiklik olup olmadığı kontrol edilir. Bu 1, bütün bellek içinde yürütülür yani "dörtnala koşturulur." Daha sonra belleğe hep 1 ler yazılıp bir 0 dörtnala koşturulabilir.

Test sayısı çok büyük olduğundan ve geçerli olması için testlerin, belleğin maksimum çalışma hızında yapılması gerektiğinden, bu testleri anahtarlarla ve LED lerle yapamazsınız. Entegre devre üreticileri, Şekil 5.20 de gösterilen Fairchild Sentry gibi karmaşık bilgisayar-kontrollu test sistemleri kullanırlar. Uygun opsiyon seçimleriyle, Sentry, 'galpat'larla yüksek hızda işlevsel test yapmasının yanısıra bir entegrenin bütün d.a. ve a.a. parametrelerini ölçmek üzere programlanabilir. Bu yüksek hızlı test cihazlarıyla bile büyük bir belleği test etmek saniyeler alır.

MANTIK ANALİZÖRLERİ: BİR DEVREDEKİ BELLEKLERİ TEST ETMEK

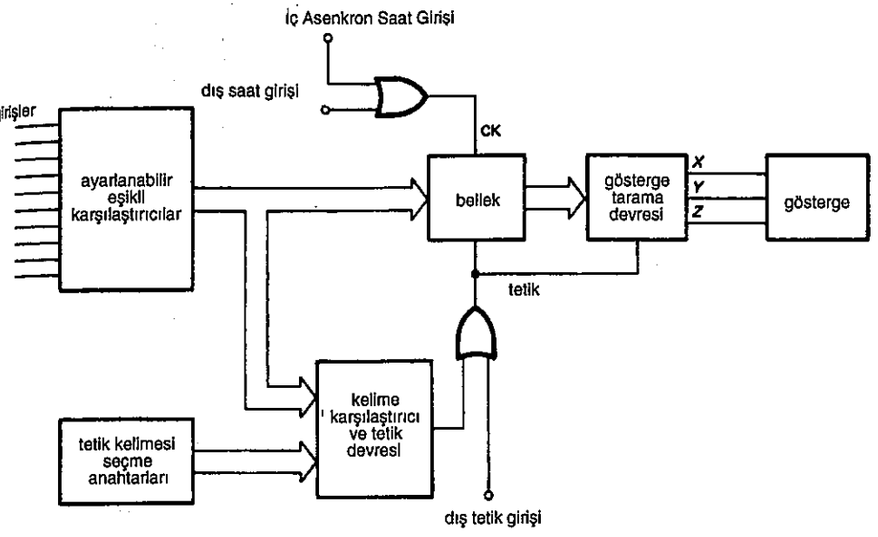
Bir baskılı devre kartında devreye halen lehimlenmiş bulunan bir belleğin çalışmasının analizini yapmak için, adres taşıtındaki sinyalleri, veri taşıtındaki sinyalleri ve okuma, yazma ve yonga seçme sinyallerini aynı anda gösterebilecek bir cihaza ihtiyaç vardır. Bunu yapmak için 40 ın üzerinde kanala gerek duyabilirsiniz. 40 kanal elde etmek için bir sürü osiloskop kullanmayı düşünebilirsiniz, ancak bu pratik değildir. Bu nedenle mantık analizörleri ve mantık durum analizörleri geliştirilmiştir. Şekil 5.21, 32 taneye kadar veri kanalını gösterebilen bir Hewlett-Packard 1600A ve 1607A mantık analizör sistemini göstermektedir. Şekil 5.21 in solundaki prob "sürüsüne" dikkat edin. Bunlar, 32 standart osiloskop probu kullanmaya göre çok daha derli toplu bağlantılar oluşturmaktadır.



Şekil 5.21 Hewlett-Packard 1600/1607A mantık analizör sistemi.

Bir mantık analizörü en basit şekliyle, Bölüm 3 te anlatılan sekiz-kanallı çoğullanmış tip gibi, temelde çok kanallı bir osiloskoptur. Şekil 5.22, basit bir mantık analizörünün blok şemasını göstermektedir. Mantık analizörleri sadece 1 leri ve 0 ları (yüksekleri ve alçakları) algılamak için kullanıldıklarından, her girişe bir karşılaştırıcı konulmuştur. Karşılaştırıcının referans girişi, test edilen ailenin mantık eşiğine eşit bir gerilime ayarlanır. Karşılaştırıcının çıkışından analizörün geri kalanına giden sinyaller artık kesin 1 ler ve 0 lardır.

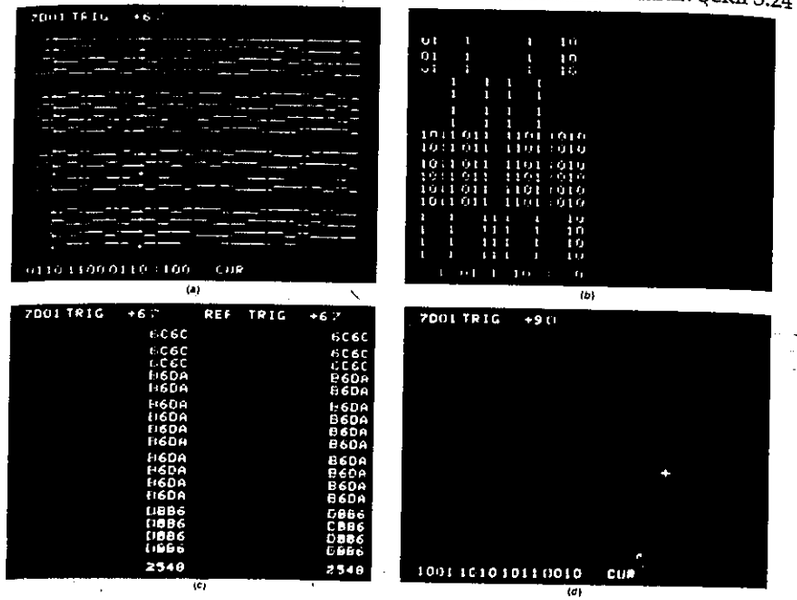
Mantık analizörlerinin, genellikle standart osiloskoplarda bulunmayan iki ayrı özellikleri daha vardır. Birincisi, standart dış tetiklemenin yanısıra,



Şekil 5.22 Basit bir mantık analizörünün blok şeması.

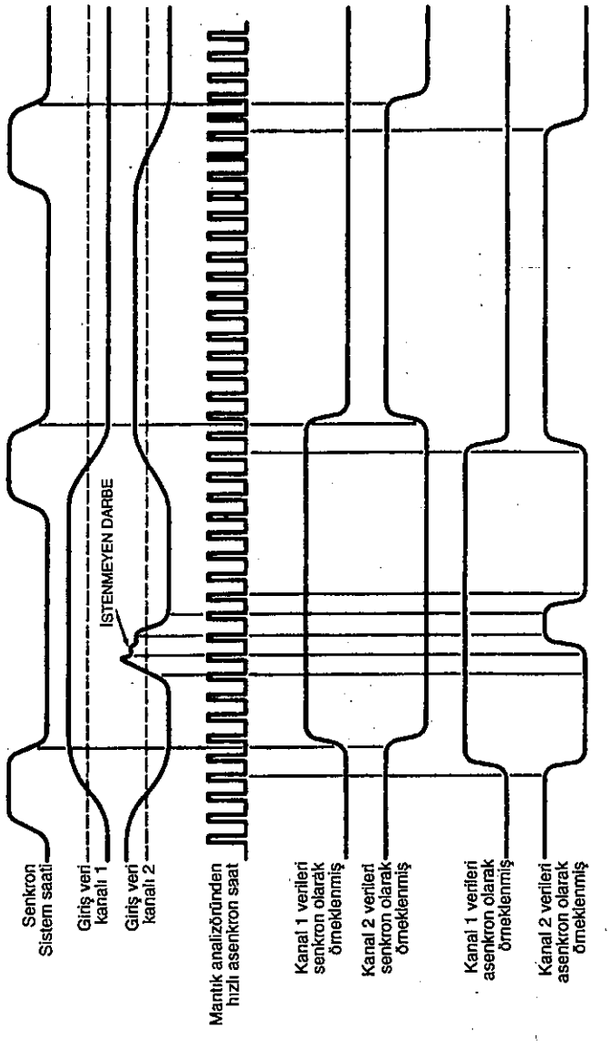
mantık analizörleri bir birleşik , yani kelime-tanıyıcı, tetikleme modu içerir. Kelime tanıyıcı devre, gelen verileri, işletmenin ön pano anahtarlarıyla programladığı kelime ile karşılaştırır. İki kelime birbirine eşit olduğunda bir tetikleme sinyali üretir. İkincisi, mantık analizörlerinin belleği vardır. Bellek, her kanal için belki 256 bitlik paralel kaydırma kaydedicisi gibi çalışır. Giriş verileri, ya iç ya da dış saatle örneklenir ve bellekte kaydırılır. Veri örnekleri, bir tetikleme oluncaya kadar sürekli olarak alınmakta ve belleğe yazılmaktadır. Bir tetikleme olunca, bellekte saklanan örnekler CRT ekranında gösterilir.

GÖSTERGE FORMATLARI İç bellekte saklanan 1 ler ve 0 lar analiz için çeşitli formatlarda görüntülenebilir. Şekil 5.23, Tektronix 7D01 mantık analizörü ve DF2 gösterge formatlayıcısı ile mümkün olan dört formatı göstermektedir. Birincisi, Şekil 5.23a da gösterilen zamanlama diyagramı formatıdır. Bu format, en iyi, istenmeyen darbeleri bulmada ve uzun veri dizilerini görüntülemekte kullanılır. Şekil 5.24 te gösterildiği gibi, zamanlama diyagramı formatı, girişleri senkron veya asenkron saatle örnekleyerek oluşturulabilir. Senkron tip, sadece test edilen sistemin sistem saatini kullanır. Şekil 5.24



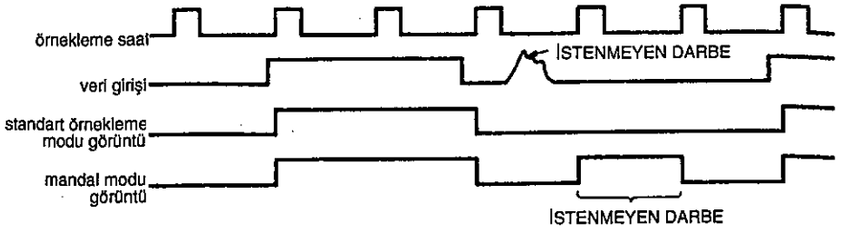
Şekil 5.23 Mantık analizörü gösterge formatları *(Tektronix).*

te gösterilen dalgaşekillerinde, örnekleme, sistem saatinin negatif kenarında olmaktadır. Senkron saatin dezavantajı, giriş veri kanalı 2 de gösterilen istenmeyen darbeyi tamamen kaçırma ihtimali olmasıdır. Şekil 5.24 ün alt kısmında gösterildiği gibi, istenmeyen darbe, bir yüksek-frekans asenkron saat kullanarak yakalanıp görüntülenebilir. Yüksek-frekans asenkron saatin dezavantajları, ayrı bir saat gerekmesi ve örnekler daha sonra görüntülenmek üzere saklanacaksa, daha fazla bellek ihtiyacıdır. Bazı analizörler tarafından kullanılan diğer bir istenmeyen darbe yakalama yöntemi bir 'mandal-modu' saklamadır. Eğer mandal-modu saklama seçilirse, bir istenmeyen darbe iç flip-flopu set veya reset eder. Bir sonraki saat darbesi, bir istenmeyen darbenin varlığını göstermek üzere görüntüyü tazeler. Şekil 5.25, normal örnekleme-moduyla mandal-modu saklamayı karşılaştırmaktadır. Birkaç nanosaniye genişliğindeki bir istenmeyen darbe, bir saat periyodu genişliğinde bir darbeye uzatılmıştır ve kolayca görülür hale gelmiştir.



Şekil 5.24 Bir istenmeyen darbenin, bir mantık analizöründe asenkron örnekleme ile nasıl yakalandığını gösteren dalgaşekilleri.

İkinci gösterge formatı, Şekil 5.23b de gösterilen 1 ler ve 0 lar "mantık durumu" tipidir. Bir veri veya adres dizisi kontrol edilirken, bu format, zamanlama diyagramı formatından daha kolay okunur. Sayısal kelimeler soldan sağa doğru okunur. Ekranın sağ yarısı, sağlam bir birimin verilerini gösterebilir, sol yarısı da arızalı birimin verilerini gösterebilir. Böylece, problemi bulmak için ikisi karşılaştırılabilir. Bazı modellerde, iki yarıyı nokta nokta karşılaştıran ve bulunan hataları sol yarıda şiddetlendiren iç devreler vardır.



Şekil 5.25 Bir mantık analizöründe, bir istenmeyen darbenin mandal modunda nasıl yakalandığını gösteren dalgaşekilleri.

Okuması ve karşılaştırması daha da kolay olan bir format, Şekil 5.23c deki onaltılık (hex) formattır. Bölüm 3 te öğrendiğiniz gibi dört onaltılık sayı, 16-bitlik ikilik kelimelerdeki 1 ler ve 0 ları temsil etmenin kısa yoludur. Bazı analizörler, onaltılık, sekizlik, onluk veya ikilik okumaları seçme imkânı verir.

Dördüncü format, Şekil 5.23d deki harita görüntüsüdür. Ekrandaki her nokta bir veri kelimesini temsil eder. Harita görüntülerinin ana kullanılış şekli, çalışan bir sistemle arızalı bir sistemin haritalarını karşılaştırarak arızayı tanımlamaktır. Haritalar herhalde kullanılması en zor öğrenilen formattır.

Bellek, mantık analizörlerine, tetiklemeden önceki kelimeleri görüntüleme yeteneğini verir. Bir tetikleme olunca, görüntü belleğinde saklanan veriler, örneğin tetiklemeden önceki 1024 örnek, görüntülenecektir. Bu, size, gözlenen bir hata kelimesinde tetikleme yapma ve hataya yol açan koşulları gözleyebilme imkânı verir. Çoğunlukla buna 'tetikönü kayıt' (pretrigger recording) denir. Eğer tetikleme bir sayıcı ile 512 saat darbesi geciktirilirse, bu takdirde göstergede tetiklemeden önceki 512 örnekle tetiklemeden sonraki 512 örnek görülür. Böyle bir görüntüye "ortada tetikli kayıt" (center trigger recording) denir. Tetikleme 1024 saat darbesi geciktirildiğinde, sadece tetiklemeden hemen sonraki örnekler görüntülenir. Bellek, ayrıca, uzun bir dizide bir defa olan olayları saklama, görüntüleme ve analizini yapma imkânını verir.

BİR UYGULAMA ÖRNEĞİ

Bir mantık analizörünün, bir bellek sisteminin arızasını bulmada nasıl kullanılabileceğine bir örnek olarak aşağıdaki durumu ele alalım: Bir bellek sistemi okunmakta olan bir veri dizisinin bir noktasında hatalı bir kelime vermektedir. Hatanın bellekte saklanmış olan yanlış bir veriden mi olduğunu, yoksa o anda belleğe giden yanlış bir adresten mi olduğunu bilmiyorsunuz. Bellek sisteminin adres taşıtına ve veri taşıtına bir mantık analizörü bağlayarak problemi bulabilirsiniz. Ön pano anahtarlarını, kelime tanıyıcı, veri yollardında hatalı kelime olunca tetikleme üretecek şekilde ayarlayın. Sonra, adresin doğru olup olmadığını kontrol etmek için adres yolu görüntülerine bakabilirsiniz. Eğer adres doğruysa ve yanlış veri saklanmadığı biliniyorsa, bu takdirde bellek entegrelerinden biri hatalıdır ve değiştirilmelidir. Mantık analizörlerinin daha başka kullanılışları daha sonraki bir bölümde anlatılmaktadır.

GÖZDEN GEÇİRME SORULARI VE PROBLEMLER

1. Aşağıdakileri tanımlayın:

   a) Kaydedici *b)* Kaydedici dosyası *c)* RAM

2. Bir RAM için aşağıdakileri tanımlayın :

   a) Adres erişim zamanı *b)* Yonga-seçme erişim zamanı

3. Bir *RAM* da 8192 yeri adreslemek için kaç adres hattına gerek vardır.?

4. Bir veri kelimesi belleğin 04FA hex adresinde saklıdır. Bu yer adreslendiğinde, 16 adres hattı üzerinde hangi 1 ler ve 0 lar deseni görülecektir?

5. Bir *RAM* veri sayfasındaki "1024 x 4" ifadesinin anlamı nedir?

6. 16384 x 8 lik bir bellek sistemi oluşturmak için 2167 *RAM* ların nasıl bağlanabileceğini gösteren bir devre şeması çizin.

7. Şekil 5.5 teki 2167-70 in zamanlama dalga şekillerini ve parametrelerini kullanarak aşağıdaki en-kötü-durum zamanlarını bulun:

   a) Adres erişim zamanı
   b) Yonga-seçme erişim zamanı
   c) Bir okuma işleminin sonunda $\overline{CS}$ yükseğe döndükten sonra verinin çıkışta geçerli kalacağı süre
   d) $\overline{CS}$ yükseğe döndükten sonra, gücü kesmek için gerekli süre.
   e) Bir yazma işlemi sırasında, $\overline{WE}$ yükseğe çıkmadan önce geçerli verilerin girişte bulunması gereken süre.
   f) Miminum yazma darbesi genişliği.

8. Niçin 2167 gibi bellekler her iki entegrede bir yangeçit kapasitörleri gerektirir?

9. *CMOS RAM* ların ana avantajı nedir?

10. Bir statik *MOS RAM* ile bir dinamik *MOS RAM* arasındaki farkları söyleyiniz. Her birinde veri nasıl saklanır?

11. Bir dinamik *RAM* hücresi nasıl tazelenir? Çoğu dinamik RAM lar ne kadar sık tazelenmelidir? Dinamik *RAM* ların avantaj ve dezavantajlarını sıralayın.

12. Dinamik *RAM* larda 16 adres bitinin sadece sekiz bacak üzerinden aygıta vermek için kullanılan sistemi anlatın.

13. Bir *MCM6664*-15 dinamik *RAM* için aşağıdaki en-kötü-durum zamanlarını belirlemek üzere Şekil 5.8 deki zamanlama dalgaşekillerini ve parametreleri kullanınız:

    a) Okuma erişim zamanı
    b) Okuma çevrimi
    c) Minimum $\overline{RAS}$ darbe genişliği
    d) $\overline{CAS}$ alçağa gittikten sonra sütun adresinin tutulması gereken süre
    e) Bir yazma işleminde, $\overline{CAS}$ alçağa gittikten sonra geçerli verinin girişte tutulması gereken süre.
    f) Kendi kendini tazeleme olması için $\overline{REFRESH}$ ın alçak tutulması gereken minimum süre.

14. a. Bir Josephson kavşağı bellekte veri, hangi şekilde saklanır?
    b. Bu bellekler, hangi ortamda çalışmalıdır?

15. a. *CCD* ler gibi seri belleklerin avantaj ve dezavantajları nelerdir?
    b. Gecikme zamanı terimi ile ne anlatılır?

16. *a.* Garnette manyetik balonlar nasıl oluşturulur?
    b. Manyetik balon bellekleri verileri, seri şekilde mi yoksa ratsgele erişilebilir şekilde mi saklar?
    c. MBM lerin yapısını anlatın.
    d. MBM saklama uçucu mudur, uçucu değil midir?
    e. MBM lerin ana problemleri nelerdir?

17. *a.* Çekirdek bellekte 1 ler ve 0 lar nasıl saklanır?
    b. Bir düzlemdeki belli bir çekirdeğin nasıl adreslendiğini ve okunduğunu anlatın.
    c. Yasaklama hattı kullanılarak, bir 0 ın belli bir çekirdeğe nasıl yazıldığını açıklayınız.
    d. Çekirdek belleğin avantaj ve dezavantajlarını söyleyin.

18. *a.* Bir manyetik bant okuma-yazma kafasının çalışmasını anlatın.
    b. Bir 0 veya 1 in manyetik bant üzerine F2F formatında nasıl yazıldığını gösteren bir dalgaşekli çizin.
    c. Bant-kayıtlı veriler için Kansas Şehri Standardı nedir?
    d. Bant-kayıtlı verilerle ilgili olarak, bir 'blok' veya 'kayıt' ne demektir? Bir 'dosya' nedir?
    e. Sayısal verilerin bantta saklanmasının avantaj ve dezavantajlarını sıralayınız.

19. Manyetik disklerde erişim, manyetik banttakinden nasıl farklıdır?

20. Bir disk sisteminde, okuma-yazma kafası istenilen iz üzerine nasıl konumlandırılır?

21. *a.* Yumuşak diskte veri saklamayla ilgili olarak, *"sert-sektörlü"* ve *"yumuşak-sektörlü"* terimlerini tanımlayın.
    b. IBM 3740 yumuşak-sektörlü formatı kullanarak her sektöre kaç bayt veri saklayabilirsiniz?
    c. Bir veri dizisinin sonundaki denetim-toplamı veya CRC ne için kullanılır?

22. Yumuşak diskleri kullanırken alınması gereken öntedbirleri sıralayın.

23. Niçin sert diskler yumuşak disklerden daha fazla veri saklayabilir ve daha hızlı okuyabilirler?

24. EBAM terimini tanımlayın.

25. *a.* Niçin yarıiletken bellekleri sadece 0 lar yazıp okuyarak ve sadece 1 ler yazıp okuyarak test etmek yeterli değildir?
    b. Galpat nedir?

26. Bir bellek sisteminin arızasını bulmak için, niçin çoğunlukla bir mantık analizörü gerekir?

27. Bir mantık analizörünün basit bir blok şemasını çiziniz.

28. *a.* Dar istenmeyen darbeleri bulmak için hangi gösterge formatı en iyisidir?
    b. İstenmeyen darbeyi hangi tür saat en iyi gösterecektir?

29. Mandal-modu saklama terimi ile ne ifade edilmektedir?

30. Bir mantık analizör, nasıl bir veri akışı içindeki belli bir kelime veya adresle tetiklenmek üzere ayarlanabilir?

31. Bir mantık analizörünün tetiklemeden önceki verileri göstermesi nasıl mümkün olmaktadır?

REFERANSLAR / BİBLİYOGRAFYA

Down, Robert L.: "Technical Note: Understanding Logic Analyzers, "Computer Design, cilt 16, no. 6, Haziran 1977, sayfa 188-191.

Tektronix 1982 Catalogue, Tektronix, Inc., Beaverton, OR, 1982.

Memory Data Manual, Motorola, Inc., Austin, Texas, 1980.

Intel Component Data Catalog, Intel Corporation, Santa Clara, CA, 1982.

Harris Digital Data Book, Harris Semiconductor Products Division, Melbourne, FL, 1981.

Memory Data Book, National Semiconductor, Santa Clara, CA, 1980.

BÖLÜM 6

D/A VE A/D ÇEVİRİCİLER

Bir sıcaklık veya basınç transduseri gibi bir algılayıcının veri çıkışını bir bellekte saklamak istediğiniz zaman, genellikle verilerin analog şekilden, belleğin kabul edebileceği sayısal 1 ler ve 0 lar şekline çevrilmesi gerekir. Bu çevrimi yapmak için, bir *analogdan-sayısala (analog-to-digital : A/D)* çevirici kullanılır. Bir analog miktar, bir aralıktaki herhangi bir değeri alabilir. Bir miktarın sayısal gösterilişi sadece sabit sayıda mümkün değerlere sahiptir.

Günlük dünyamızda kaldırımdaki bir rampa bir analog sistemdir. Çünkü rampanın altıyla üstü arasında ayağınızı koyabileceğiniz sonsuz sayıda konum vardır. Bir merdiven bir sayısal sistemdir; çünkü ayağınızı koyabileceğiniz sabit sayıda konum vardır.

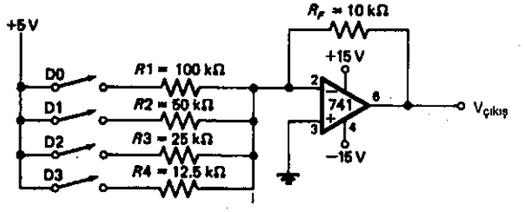
Bir *A/D* çeviricinin tersi, *sayısaldan-analoğa (digital-to-analog:D/A)* çeviricidir. *D/A* çevirici, girişine uygulanan ikilik kelimenin büyüklüğü ile orantılı bir çıkış akımı veya gerilimi üretir. Bu bölümde önce *D/A* çeviricileri anlatacağız; çünkü bunlar, *A/D* çeviricilerden daha basittir ve birçok *A/D* çevirici türü, devrelerinin bir parçası olarak *D/A* çeviriciler kullanır.

Bu bölümün sonunda şunları yapabileceksiniz:

1. Bir ağırlıklı direnç *D/A* çeviricinin devresini çizebilecek ve çalışmasını anlatabileceksiniz.
2. Bir *R/2R* merdiven tipi *D/A* çeviricinin devresini çizebilecek ve çalışmasını anlatabileceksiniz.
3. *D/A* ve *A/D* çevirıcileri tanımlamakta kullanılan çözünürlük, doğruluk, lineerlik, monotonluk ve çevrim süresi gibi terimleri tanımlayabileceksiniz.
4. Bir *A/D* veya *D/A* çeviricinin, verilen bir çözünürlük için gerekli bit sayısını belirleyebileceksiniz.
5. Bir paralel tip *A/D*, tek-rampalı *A/D*, çift-rampalı A/D, sayıcı tipi *A/D*, izleyen *A/D* veya ardarda-yuvarlamalı *A/D* çeviricinin blok şemasını çizebilecek ve çalışmasını anlatabileceksiniz.
6. Bir *A/D* çeviriciyi, bir sayısal multimetreye çevirmek için gerekli devreleri anlatabileceksiniz.
7. *CODEC* i tanımlayabileceksiniz.

SAYISALDAN-ANALOĞA ÇEVİRİCİLER
İKİLİK AĞIRLIKLI DİRENÇ D/A ÇEVİRİCİ

En basit tip *D/A* çevirici, ikilik ağırlıklı mantık sinyallerini, onlarla orantılı bir gerilime çevirmek için bir *işlemsel kuvvetlendirici (op amp)* ve ikilik ağırlıklı dirençler kullanır. Şekil 6-1, bu tür bir 4-bitlik çeviricinin devresini göstermektedir. Bir işlemsel kuvvetlendiricinin, çok yüksek kazancı (genelikle 100.000 den büyük), alçak çıkış empedansı ve yüksek giriş empedansı vardır. İşlemsel kuvvetlendiricinin iki girişi vardır: Üçgen şematik sembolünde eksi işaretiyle belirtilen bir ters çevirici giriş ve sembolde artı işaretiyle belirtilen bir ters çevirmeyen giriş. Her halde işlemsel kuvvetlendiriciler hakkında hatırlamanız gereken en önemli nokta, ne zaman çıkışın bir kısmı ters çevirici girişe geri beslenirse (negatif geri besleme), ters çeviren ve ters çevirmeyen girişlerde aynı gerilimi ölçeceğinizdir. İşlemsel kuvvetlendiricinin çıkışı, bu gerilimleri aynı tutmak için gerekli olan akımı kaynaklandıracak veya çöktürecektir. Şekil 6.1 deki devrede, ters çevirmeyen giriş topraklanmıştır. Bu nedenle ters çeviren giriş 0 V ta tutulacaktır. Ters çeviren giriş 0 V ta olduğu halde gerçekten toprağa bağlanmadığı için bazen buna zahiri (varsayalım) toprak denir.



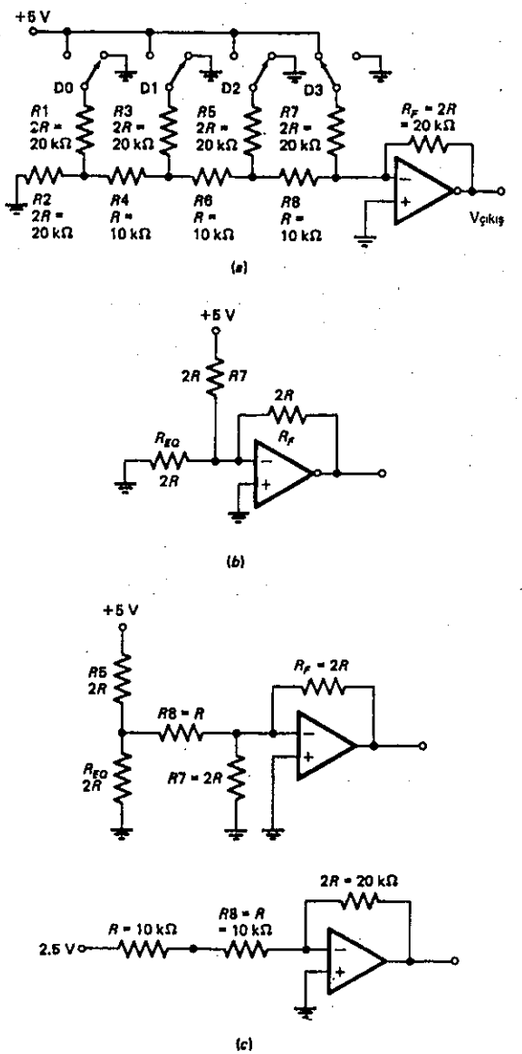
Şekil 6.1: İkilik ağırlıklı direnç *D/A* çevirici.

Şimdi D0 anahtarı kapandığında ne olacağına bakalım. 100KΩ luk R1 direncinin bir ucunda 5 V ve diğer ucunda 0 V (zahiri toprak) vardır. Ohm kanunu, üzerinde 5 V olan bir 100 KΩ luk direncin içinden 0,05 mA lık akım geçeceğini söyler. Bu akım, işlemsel kuvvetlendiricinin girişinden içeri giremez, çünkü işlemsel kuvvetlendiricinin yüksek bir giriş empedansı vardır ve fazla akım çöktüremez veya kaynaklandıramaz. Bu nedenle 0,05 mA lik akım, 10 KΩ luk geri besleme direnci üzerinden işlemsel kuvvetlendiricinin çıkışına gitmek zorunda kalır. Ohm kanunu tekrar kullanılırsa, D0 anahtarından gelen akımı çöktürmek ve zahiri toprağı korumak üzere kuvvetlendiricinin çıkışındaki gerilimin 10 K Ω x (-0,05 mA)= - 0,5 V olması gerektiğini söyler. Eğer bunu görmekte zorluk çekiyorsanız bir ucu + 5 V ta, ortası 0 V ta ve diğer ucu - 0,5 V ta olan basit bir gerilim bölücü olarak çizin.

Şimdi aklınızdan *D0* anahtarını açın ve *D1* anahtarını kapayın. *R2* nin değeri *R1* in değerinin sadece yarısı olduğundan, iki misli, yani 0,1 mA lik bir akım, *R2*, zahiri toprak ve R_F üzerinden akacaktır. R_F den geçen iki misli akım, çıkış gerilimini de ikiye katlayarak - 1 V yapacaktır. Çıkış gerilimi, R_F den geçen akımla doğru orantılıdır. Bundan sonraki basamak, *D 0* ve *D 1* in her ikisini de kapayıp çıkış gerilimini belirlemektir. Her iki anahtar kapalıyken, *R1* den 0,05 m A ve R2 den 0,1 m A geçecektir. Zahiri toprak noktasında iki akım toplanmakta ve R_F ye doğru akmaktadır. Şimdi R_F den geçen toplam akım 0,15 mA olduğuna göre kuvvetlendiricinin çıkışındaki gerilim - 1,5 V olur. Anahtarların başka bileşimlerini deneyebilir ve çıkış gerilimlerini hesaplayabilirsiniz. Dirençler, *D0-D3* anahtarlarıyla programlanan sayısal kelimeye göre ikilik ağırlıklı akımlar oluşturur. Bu akımlar, zahiri toprak noktasında (toplama noktası) toplanır ve işlemsel kuvvetlendirici ile R_F tarafından bir orantılı gerilim çıkışına çevrilir. Şekil 6.1 deki devreyi kurmak ve anahtar yerine bir 4-bitlik ikilik *TTL* veya CMOS sayıcının çıkışlarına bağlamak isteyebilirsiniz. Çıkış, her biri - 0,5 V olan 15 basamaklı ilginç bir merdiven şeklinde olacaktır. R_F nin değeri değiştirilerek basamakların büyüklüğü değiştirilebilir. Ancak, eğer R_F çok büyükse, üst basamaklarda, işlemsel kuvvetlendiricinin çıkışı -14 V civarında doyuma sürülebilir.

R/2R MERDİVEN TİPİ D/A ÇEVİRİCİ

4-bitlikten daha fazla bir *D/A* çevirici için Şekil 6.1 de gösterilene benzer bir devrenin, gerekli direnç değerleri aralığının çok büyük olması gibi bir problemi vardır. Sadece iki direnç değeriyle ikilik ağırlıklı akımlar üreten bir başka devre Şekil 6.2 de gösterilmiştir. Daha önce olduğu gibi ikilik ağırlıklı akımlar işlemsel kuvvetlendirici ve R_F aracılığı ile, orantılı bir gerilime çevrilirler. İlk bakışta *R/2R* merdiveni bir Kirchhoff kanunu kabusu gibi görülebilir, fakat basit direnç oranlarından dolayı oldukça kolaydır. Önce, en büyük değerlikli bit anahtarı *D3* ün +5 V luk bir hassas referans gerilimine ve diğer anahtarların toprağa bağlandığını kabul edin. Bu takdirde *R1* ve *R2* paralel olarak toprağa bağlı olur. *2R* değerinde iki direncin paralel dengi bir tek R dir. Bu denk *R*, *R3* ile paralel toprağa bağlı *2R* lik bir başka direnç oluşturmak üzere *R4* e eklenir. *R3* ün diğer dirençlerle birleşimi, *R6* ile seri bir tek *R* ye indirgenir. Devrenin gerisinde de aynı tekniği kullanarak Şekil 6.2*b* de gösterilen basitleştirilmiş devre elde edilir. İşlemsel kuvvetlendiricinin varsayılan toprağı 0 V ta olduğundan, denk direnç üzerinden toprağa akım akmaz. Böylece bu direnç ihmal edilebilir. 20 KΩ luk *R7* direncinin ucundaki 5 V, toplama noktası ve 20 K Ω luk R_F üzerinden 0,25 mA lik bir akım oluşturur. En büyük değerlikli bit tarafından oluşturulan çıkış gerilimi -5 V tur.



Şekil 6.2: *R/2R* merdiven tipi *D/A* çevirici. *(a)* Tam şema. *(b)* MSB anahtarı yükseğe bağlıyken denk devre. *(c)* Bir sonraki MSB anahtarı yükseğe bağlıyken denk devre.

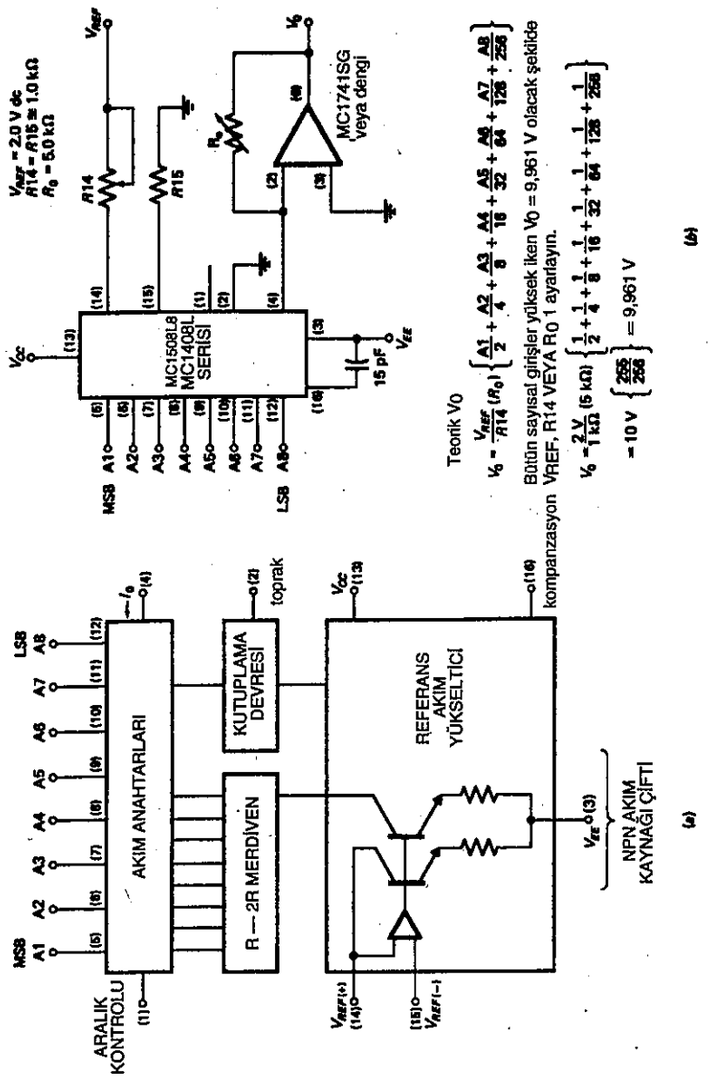
Bir sonraki en büyük değerlikli bitin ürettiği gerilimi bulmak için, aklınızdan *D2* anahtarını +5 V a ve *D3* anahtarını toprağa bağlayın. Şekil 6.2*a* da *R5* in solundaki bütün dirençler toprağa bağlı bir tek *2R* lik dirence indirgenir.Devre, *R5* ile toprağa bağlı *2R* nin oluşturduğu gerilim bölücünün Thevenin dengini Şekil 6.2*c* de gösterildiği gibi bularak daha da basitleştirilebilir. Thevenin gerilimi, kavşaktaki gerilim, yani 2,5 V tur. Thevenin direnci, paralel iki direncin değerine, yani *R* ye eşittir. *R8* in solundaki her şey 2,5 V a bağlanmış *R* değerinde bir tek direnç ile gösterilebilir. *R7* yi ihmal edebilirsiniz, çünkü her iki ucu topraktadır.Toplama noktasıyla 2,5 V luk Thevenin dengi gerilim arasındaki toplam direnç *2R* yani 20 K Ω dur. Toplama noktasına gelen akım 2,5 V/20 K Ω yani 0,125 mA dir. 20 K Ω luk R_F den geçen bu akım, bir sonraki en büyük değerlikli bit için, -2,5 V luk bir çıkış gerilimi oluşturur. Benzer analizle, bir sonraki anahtarın 1,25 V luk ve en küçük değerlikli bit anahtarının da 0,625 V luk bir çıkış ürettiğini gösterebilirsiniz. Bütün anahtarlar +5 V a bağlıyken elde edilecek tam-ölçek çıkış gerilimi, bu dört gerilimin toplamı, yani 9,375 V tur.

R/2R merdiven tipi *D/A* çevirici, ağırlıklı direnç tipine göre analizi daha zor olmasına rağmen, hassas olarak kurması daha kolaydır. Çünkü sadece bir veya iki değerde hassas metal-film direnç gerekmektedir. Bit sayısı, aynı *R/2R* değerleri olan daha fazla kısımlar ekleyerek artırılabilir. Bir 4-bitlik TTL veya *CMOS* ikilik sayıcı, bir negatif merdiven basamağı çıkış dalgaşekli vermek üzere Şekil 6.2 *a* daki anahtarların yerine bağlanabilir. Mantık yüksek gerilim seviyelerindeki farklılıklardan dolayı bütün basamaklar eşit olmayabilir.

MONOLİTİK VE HİBRİT D/A ÇEVİRİCİLER

Monolitik "tek parça" demektir ve entegre devrelerle ilgili olarak kullanıldığında, bütün devrenin tek bir silikon yonga üzerinde içerildiğini belirtir. Bir hibrit devre, bir veya daha fazla silikon yonga ve direnç ağlarını veya diğer mikrominyatür elemanları bir tek entegre devre tipi pakette içerir.

Çok kullanılan bir 8 bitlik monolitik *D/A* çevirici, blok şeması Şekil 6.3a da gösterilen MC1408L dir. 1408L, standart 16 bacaklı DIP olarak gelir ve +5 V luk V_{CC} ile minimum -5, maksimum -15 V luk V_{EE} gerilimi gerektirir. 1408L de, bir *R/2R* merdiveni, akım yükseltecinden gelen referans akımını, 8 ikilik ağırlıklı akıma böler. Bipolar transistör anahtarlar, *A1-A8*, TTL-uyumlu girişlerdeki ikilik kelimeye göre, ikilik ağırlıklı akımları çıkış hattına bağlar. MSB ve LSB etiketlerinin, sayıcılardaki normal etiketlendirmenin tersine olduğuna dikkat edin. Bazı *D/A* çeviriciler bunu yapar ve bazıları yapmaz, bu nedenle veri sayfalarını dikkatli okumanız gerekir. 1408 L nin, Şekil 6.3*b* de



Şekil 6.3: Motorola MC 1408 D/A çevirici.
***(a)* Blok diyagramı. *(b)* Çıkış gerilimi şeması.**

gösterildiği gibi bir işlemsel kuvvetlendirici ve bir dirençle gerilime çevrilebilen akım çıkışı vardır. Bu gerilim aşağıdaki formülü kullanarak hesaplanabilir.

$$V_{ÇIKIŞ} = \frac{V_{REF}}{R14} \times R0 \left(\frac{A1}{2} + \frac{A2}{4} + \frac{A3}{8} + \frac{A4}{16} + \frac{A5}{32} + \frac{A6}{64} + \frac{A7}{128} + \frac{A8}{256} \right)$$

Gösterilen değerler için, tam ölçek çıkış gerilimi,

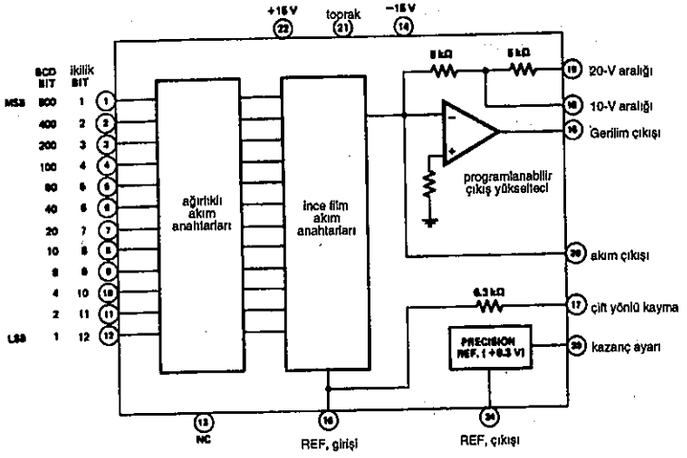
$$2\,V \times \frac{5\,k\Omega}{1\,k\Omega} \times \frac{255}{256} = 9{,}961\,V$$

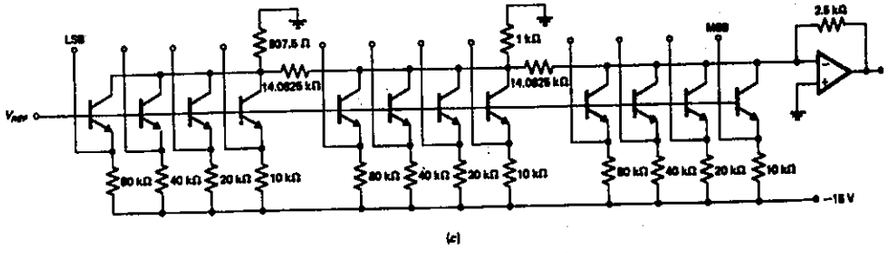
olarak bulunur. Bu çeviriciye 10 V tam ölçekli çevirici denir.

1408L gibi ucuz *D/A* çeviricilerin hoş bir uygulaması, özel ses ve dalga şekilleri üretmede yapılır. 8-bitlik bir ikilik sayıcının çıkışları *D/A* girişlerine bağlanabilir. Sayıcı saymaya devam ederken, *D/A* çevirici 255 küçük basamaktan oluşan bir testere dişi dalgaşekli verir. Çıkış frekansı, sayıcının giriş saat frekansını 256 ya bölerek bulunur. *D/A* çeviriciyi, 8-bit genişliğinde bir ROM veya RAM belleğin çıkışına bağlayarak daha karmaşık dalga şekilleri yaratılabilir. Bellek, nokta nokta istenen ikilik değerlere programlanır. Bellek, adres girişlerine bağlanan bir sayıcı ile sırayla okunarak *D/A* çeviriciye çıkış verir. Üretebileceğiniz seslerin sayısı sadece sizin hayal gücünüzle, belleğin büyüklüğüyle ve belleği programlamadaki sabrınızla sınırlıdır.

Daha fazla çözünürlük biti gerektiren uygulamalarda, Datel DAC-HZ 12 BCG gibi hibrit *D/A* çeviricileri, 24 bacaklı DIP paketler olarak bulunmaktadır. Kapağı açılmış bir hibrit devrenin resmi Şekil 6.4 a da ve blok diyagramı Şekil 6.4 b de gösterilmiştir. Bu, sadece +15 ve -15 V luk besleme gerektiren bir 12-bitlik çeviricidir. Dirençler ve bir işlemsel kuvvetlendirici pakette bulunmaktadır, böylece hem akım hem de gerilim çıkışları mümkündür. Tam-ölçek çıkış gerilimi işlemsel kuvvetlendiricinin üzerindeki geri besleme direnci için farklı değerler kullanarak değiştirilebilir. Akım-belirleyen ağ için bir hassas referans gerilimi de içerilmiştir. Bir 12-bitlik çeviricide gerekli hassasiyeti elde etmek için, ince-film direnç ağındaki dirençlerin her biri lazerle uygun değere ayrı ayrı düzeltilmiştir. Her direncin ayrı ayrı düzeltilmesi gerektiğinden, çeviricide gerekli direnç sayısının azaltılması istenir. *R/2R* merdiveni, her bit için iki direnç gerektirir ve ağırlıklı direnç yöntemi her bit için sadece bir direnç kullanır. Ağırlıklı direnç yönteminin problemi, 4-bitlikten büyük çeviricilerde gereken, çok büyük direnç aralığıdır. Bu problem, Şekil 6.4 c de gösterildiği gibi, üç tane 4-bitlik ağırlıklı direnç tipi kullanarak çözülebilir. Daha önemsiz kısımlarla seri olan dirençler, bu kısımların çıkışlarını zayıflatır, böylece aynı değerde akım kaynağı dirençleri, bütün kısımlarda kullanılabilir.







Şekil 6.4: Datel hibrit D/A çevirici.
***(a)* Fotoğraf. *(b)* Blok diyagram. *(c)* İkilik ağırlıklı akım kaynakları.**

Şekil 6.4 c de gösterildiği gibi hassas akım kaynaklarının hepsi çalışır vaziyettedir. Gösterilmemiş olan anahtar devresi, istenilen akım kaynaklarının emitörlerini yükseğe çekerek keser. Çeşitli anahtarlama yolları vardır ve bunların hepsini burada anlatmak mümkün değildir. *D/A* çeviricilerinin kullanıcısı olarak sizin için önemli olan, dahili olarak hangi anahtarlama modunun kullanıldığından çok, genel özellikleridir.

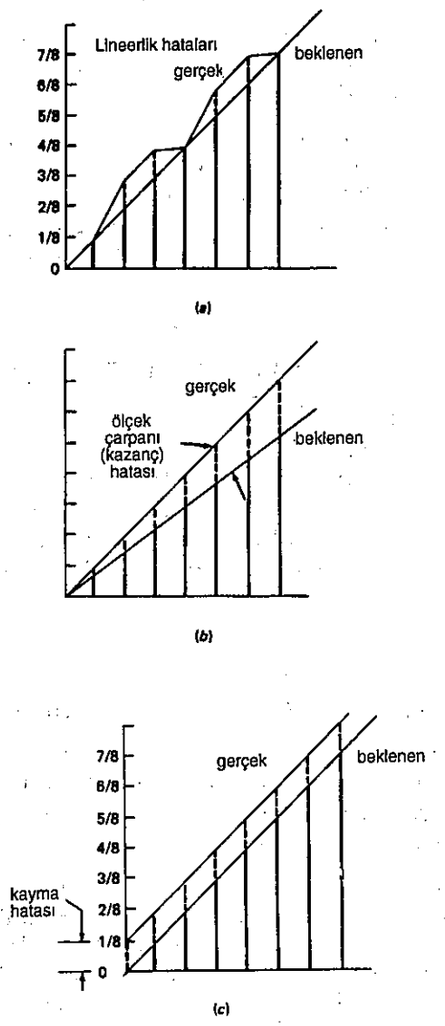
D/A ÇEVİRİCİNİN KARAKTERİSTİKLERİ, HATA KAYNAKLARI VE ÖZELLİKLERİ

Bir D/A çeviricinin dikkate alınacak ilk karakteristiği, çözünürlüğüdür. Bu, giriş kelimesindeki bit sayısıyla belirlenir. Örneğin bir 8-bitlik çevirici, 2^8 yani 256 mümkün çıkış seviyesine sahiptir. Böylece çözünürlüğü 256 da 1 parçadır. 12-bitlik bir çeviricinin çözünürlüğü 2^{12} yani 4096 da 1 parçadır. Çözünürlük bazen yüzde olarak ifade edilir. 4096 da 1 parça yaklaşık %0,024 tür.

Dikkate alınması gereken bir diğer nokta, *D/A* çeviricinin doğruluğudur. Doğruluk dediğimizde, gerçek çıkışla beklenen çıkışın karşılaştırılmasından bahsediyoruz. Doğruluk, tüm ölçeğin veya maksimum çıkış geriliminin yüzdesi olarak belirtilir. Eğer bir çeviricinin 10 V luk tam ölçek çıkışı ve yüzde ±0,2 doğruluğu varsa, herhangi bir çıkış için maksimum hata 0,002 x 10 V, yani 20 mV olacaktır. İdeal olarak, bir D/A çeviricinin doğruluğu, en kötü halde, en önemsiz bitinin ±1/2 si olmalıdır. 10-bitlik bir çeviricinin çözünürlüğü 1024 te 1 parçadır, yani yaklaşık %0,1 dir. O halde doğruluğu yüzde ±0,05 olmalıdır. Bu hata çeşitli şekillerde ortaya çıkabilir. Şekil 6-5, bunlardan üç tanesini göstermektedir.

Lineerlik hataları, gerçek çıkışın ideal düz çizgi çıkışından ne miktarda ayrıldığıdır. Bu hata genellikle, akım kaynağı direnç değerlerindeki hatalar tarafından oluşturulur. Diğer bir hata tipi, kazanç hatasıdır. Bu hata, akımdan gerilime çeviren işlemsel kuvvetlendiricinin üzerindeki geri besleme direncindeki hatalardan oluşur. Kayma hatası, bütün girişler 0 olduğunda, çıkışın 0 olmadığı anlamına gelir. Kayma hatası, Şekil 6.5 c de görüldüğü gibi, bütün çıkış değerlerine bir sabit ekler. Bu hataya, işlemsel kuvvetlendirici hataları veya akım anahtarlarındaki kaçak akımlar sebep olabilir.

Bir *D/A* çeviricinin doğruluğu, çözünürlüğünden daha az olabilir. Örneğin MC1408L nin çözünürlüğü 256 da 1 parça, yani %0,39 dur. Veri sayfası, MC1408L-8 in doğruluğunu yüzde ±0,19, yani en küçük değerlikli bitin ±1/2 si olarak belirtir. Bu da çözünürlükle uyuşmaktadır. Ancak Motorola, doğruluğu sadece yüzde ±0,78 olan 8-bitlik çevirici MC1408L-6 yı da pazarlamaktadır. Bu, 6-bitlik bir çeviricinin çözünürlüğüne eşittir. Doğruluğu daha



Şekil 6.5: *D/A* **hataları.** *(a)* **Lineerlik.** *(b)* **Ölçek veya kazanç.** *(c)* **Kayma.**

düşük olan aygıt, çok daha ucuzdur ve 8-bitlik aralığın gerektiği fakat doğruluğun gerekmediği yerlerde kullanılabilir.

D/A çeviricilerde kullanılan diğer bir terim "monotonluk"tur. Bir çevirici, bir sayıcı ile bütün aralığı adımlandığında, adım kaçırmaz veya geri adım atmazsa buna monotonik denir.

Çıkış yerleşim zamanı, *D/A* çeviricilerin verilen özelliklerinden biridir. Yerleşim zamanı, genellikle, bir giriş kelimesindeki değişiklikten sonra, çıkışın, son değerin ±1/2 LSB sine yerleşinceye kadar geçen zaman olarak tanımlanır. Eğer bir çevirici çok yüksek bir frekansta çalıştırılırsa, bir sonraki değere değiştirilmeden önce, o andaki doğru değere yerleşecek zaman bulamayabilir.

ANALOGDAN SAYISALA ÇEVİRME

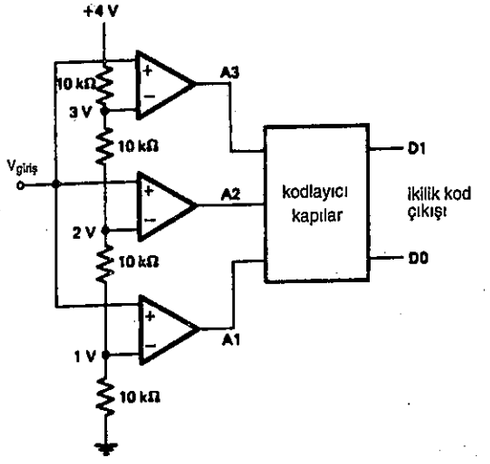
Geçen kısımda, bir sayısal kelimenin orantılı bir akım veya gerilime nasıl çevrildiğini gösterdik. Bu kısımda, bir analog gerilimi onu temsil eden en iyi sayısal kelimeye çevirme işlemini anlatacağız. *D/A* çeviricide olduğu gibi, bir *A/D* çevirici, bir giriş gerilimini 256 mümkün kelimeden biri ile temsil edebilir. Bu takdirde, çözünürlük 256 da 1 parçadır denir. Birçok çeşit *A/D* çevirici vardır. Çok bilinen tiplerden bazıları aşağıdaki kısımlarda anlatılmaktadır.

PARALEL KARŞILAŞTIRICI, SİMULTANE VEYA "FLAŞ" A/D ÇEVİRİCİLER

Kavram olarak en kolay ve en hızlı *A/D* çevirici tipi Şekil 6.6a da gösterilen paralel karşılaştırıcı tipidir. Bir rezistif gerilim bölücü üç karşılaştırıcının eşik gerilimlerini ayarlar. Bölücünün üstündeki referans gerilimi tam-ölçek, yani anlamlı maksimum giriş gerilimidir.

Bir karşılaştırıcının çıkışı, pozitif girişindeki giriş gerilimi, negatif girişindeki referans geriliminden büyükse, yüksektir. Sayısallaştırılacak olan analog sinyal, bütün karşılaştırıcılara paralel olarak uygulanır. Yüksek çıkışı olan karşılaştırıcıların sayısı, analog giriş geriliminin büyüklüğünü belirtir. Şekil 6.6 *b*, çeşitli giriş gerilimleri için, Şekil 6.6 *a* daki devrenin çıkışlarını göstermektedir. Eğer giriş gerilimi 1 V dan az ise, hiçbir karşılaştırıcının çıkışı yüksek olmaz. Eğer, en düşük eşikli karşılaştırıcının çıkışı yüksekse, 1 ile 2 V arasındaki bir giriş gerilimini belirtir. 2 ile 3 V arasındaki bir giriş gerilimi, *A1* ve *A2* karşılaştırıcı çıkışlarında yüksek verir. 3 V un üzerinde bir gerilim bütün karşılaştırıcıların çıkışlarında yüksek üretir.

Bu sistem, üzerinde sadece metre işaretleri olan 4m lik bir cetvele benzer. Çevirici, giriş gerilimini dört giriş seviyesinden birisine çözer. Bu, 2 ikilik bit çözünürlüğüne denktir. Çeviricinin çözünürlüğünü artırmak için daha fazla karşılaştırıcı kullanılabilir. 3-bitlik çözünürlük için 7 karşılaştırıcı ve 4-bitlik çözünürlük için 15 karşılaştırıcı gerekmektedir. Bir N-bitlik çevirici, 2^N-1 karşılaştırıcı gerektirir. O halde 8-bitlik bir çevirici 255 karşılaştırıcı gerektirir. Bu çok fazla sayıdaki karşılaştırıcı, paralel tip A/D çeviricilerin en önemli dezavantajıdır.



(a)

Vgiriş (Volt)	Karşılaştırıcı Çıkışları			İkilik Çıkış	
	A1	A2	A3	D1	D0
0 to 1	0	0	0	0	0
1 to 2	1	0	0	0	1
2 to 3	1	1	0	1	0
3 to 4	1	1	1	1	1

(b)

Şekil 6.6: Paralel karşılaştırıcılı *A/D* çevirici.
***(a)* Şema. *(b)* Çeşitli giriş gerilimleri için çıkışlar.**

Bu tip çeviricinin bir diğer dezavantajı, çıkış kodunun ikilik olmamasıdır. Ancak ilave kapılarla, çıkış, ikiliğe çevrilebilir. Şekil 6.6 a daki devrenin karşılaştırıcı çıkış kodunun çevrimi, mantık kapıları ve Karnaugh haritalarının bir tekrarı olması için okuyucuya bırakılmıştır.

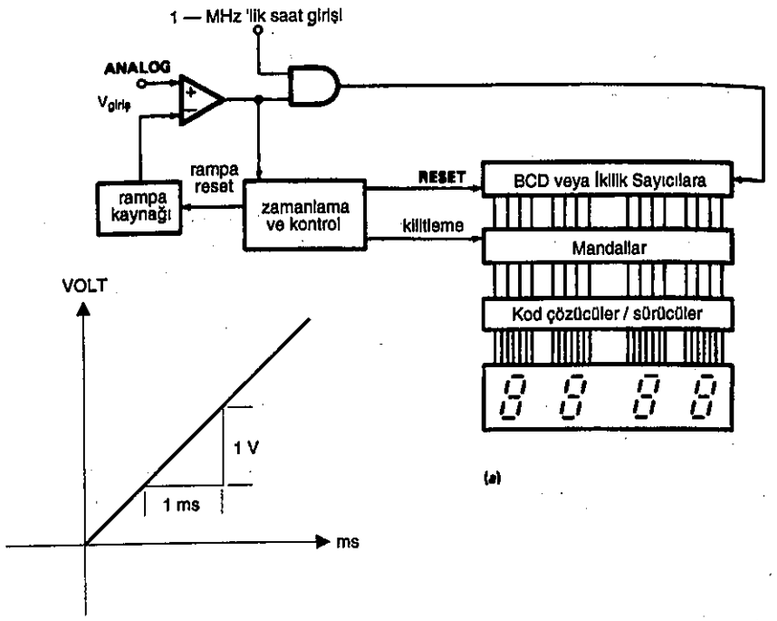
Paralel tip *A/D* çeviricinin en önemli avantajı hızıdır. Sayısal çıkış kelimesinin tamamı, karşılaştırıcılar ve kodlama kapılarındaki yayılma gecikme zamanından hemen sonra hazırdır. Bu nedenle, bu tip için bazen flaş kelimesi kullanılmaktadır. TRW TCD-1007J, 8-bitlik flaş çevrimi yaklaşık 33 ns de yapabilmektedir.

TEK RAMPALI VEYA TEK EĞİMLİ A/D ÇEVİRİCİ

Şekil 6.7*a*, bir rampa üreteci, karşılaştırıcı ve BCD yani ikilik sayıcılar kullanan bir A/D çeviriciyi göstermektedir. Çevrimin başlangıcında, rampa kaynağı ve sayıcılar 0 a reset edilir. Karşılaştırıcının artı girişine bir analog gerilim uygulanır. Bu giriş, karşılaştırıcının negatif girişinden daha pozitif olduğundan, karşılaştırıcının çıkışı yükseğe gider. Bu, hem VE kapısının saati sayıcılara ulaştırmasını hem de rampanın başlamasını sağlar. Rampa gerilimi, giriş gerilimini aşıncaya kadar pozitif yönde artar. Sonuçta karşılaştırıcı çıkışı, alçağa gider ve VE kapısını yetkisiz kılar. Böylece sayıcıya giden saat kapatılmış olur. Sayıcı verilerini kilitlemek, sayıcıları 0 a reset etmek ve rampa üretecini reset etmek için Bölüm 4 deki frekans sayıcı devresine benzer kontrol devreleri kullanılabilir.

Bu devrenin nasıl çalıştığını daha iyi görmek için, saat frekansının 1MHz, BCD sayıcıların dört tane ve $V_{GİRİŞ}$ in 2,000 V olduğunu varsayalım. Ayrıca, Şekil 6.7*b* de gösterildiği gibi rampanın eğiminin 1 V/ms olduğunu varsayalım. Rampanın 2 V a ulaşması ve sayıcılara giden saat darbelerini kapatması, çevrimin başlangıcından sonra 2 ms alacaktır. Bu 2ms de, 2000 saat darbesi sayıcılara ulaşacak ve sayılacaktır. Yükselen karşılaştırıcı çıkışı, mandallara yükleme yapacak ve sayımı göstergeye gönderecektir. Yedi - parçalı sayısal göstergede onluk virgül, uygun yere konulduğunda, 2,000 V luk analog giriş gerilimine doğrudan karşılık gelen 2,000 okumasını verecektir. 9,999 V a kadar herhangi bir pozitif giriş gerilimi, aynı şekilde BCD karşılığına çevrilip görüntülenebilir. Bu devre, basit bir sayısal voltmetredir. Sayısal voltmetreler, bu bölümde derinlemesine daha sonra anlatılacaktır. BCD sayıcılar yerine ikilik sayıcılar kullanılırsa, çıkış doğrudan ikilik kodlanmış olur.

20.000 de 1 parça veya daha yüksek çözünürlük gerektiren sayısal voltmetreler gibi uygulamalarda, tek rampalı, yani tek eğimli, *A/D* çeviriciler yeterince kararlı değildir. Zaman, sıcaklık veya giriş gerilim duyarlılığı, rampa üretecinde değişikliklere neden olur. Şimdi anlatacağımız çift-eğimli çeviricide, bu etkenlerin çoğu ortadan kaldırılmıştır.

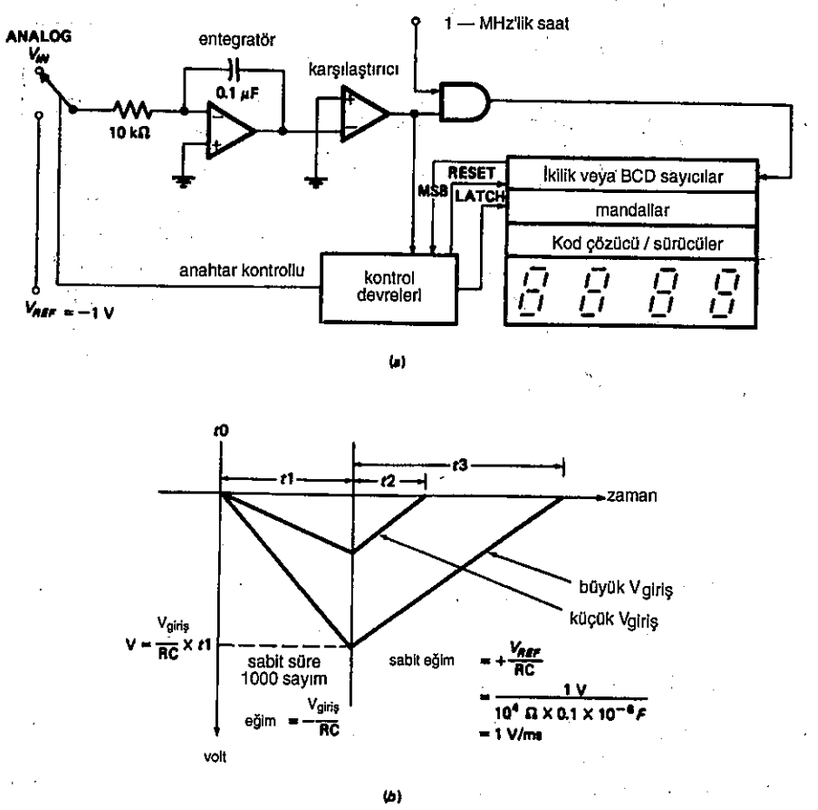


Şekil 6.7: Tek eğimli *A/D* çevirici. *(a)* Blok diyagramı. *(b)* Rampa eğimi.

ÇİFT EĞİMLİ A/D ÇEVİRİCİLER

Bir çift eğimli *A/D* çevirici devrenin blok diyagramı Şekil 6.8 a da gösterilmiştir. Devre, giriş gerilimini veya referans gerilimini seçen girişteki bir anahtar ile ters bağlanan karşılaştırıcı girişleri dışında tek-eğimli devreye çok benzemektedir.

Bu devrenin ilk kısmı rampa üreteci yani entegratördür. İşlemsel kuvvetlendiricinin ters çeviren girişi, işlemsel kuvvetlendirici tarafından varsayılan toprakta tutulur. Örneğin 10 K Ω direncin giriş ucuna uygulanan 2 V luk bir gerilim, direnç üzerinden toplama noktasına 0,2 mA lik sabit bir akım akmasına sebep olur. Bu akım, yüksek empedanslı işlemsel kuvvetlendirici girişinden içeri doğru akamayacağından, kapasitör plakalarından birine doğru akar. İşlemsel kuvvetlendiricinin ters çeviren girişini, varsayılan toprakta tutmak için, işlemsel kuvvetlendirici aynı akımı kapasitörün diğer plakasından çekmelidir. Kapasitör şarj ederken, işlemsel kuvvetlendiricinin çıkış gerilimi, sabit akımın akmasını sağlamak üzere sürekli daha negatif olmak zorundadır. Sabit bir akımla şarj edilen bir kapasitörün uçları arasındaki gerilim bir lineer rampadır. Pozitif bir giriş gerilimiyle, entegratörün çıkışı, Şekil 6.8 b de göste-



Şekil 6.8: Çift eğimli *A/D* çevirici *(a)* Blok diyagramı. *(b)* Zamana göre entegratör çıkışı.

rildiği gibi negatif bir rampa üretilir. Negatif bir gerilim ise pozitif yönde artan bir lineer rampa üretir.

Rampanın eğimi, kapasitörler için temel ifade $q=CV$ ve $q=It$ den kolayca belirlenebilir. Bu iki ifadeyi eşitleyerek ve terimleri düzenleyerek $\Delta V/\Delta t=I/C$ elde edilir. Akım, $V_{GIRIŞ}/R$ ye eşit olduğuna göre $\Delta V/\Delta t=V_{GIRIŞ}$ (RC) olur. Böylece verilen bir giriş gerilimi için, çıkışın eğimi $\Delta V/\Delta t$ nin sabit

olduğu görülmektedir. Gösterilen değerler ve +2 V luk bir giriş için, çıkışın eğimi 2V/ms dir.

Rampa üretecini açıkladığımıza göre, şimdi bir çevrimin nasıl yapıldığını görelim. Çevrim, rampa 0 da, sayıcılar 0 a reset edilmiş ve giriş anahtarı giriş gerilimine bağlanmış olarak başlar. Entegratörün girişindeki pozitif gerilim, çıkışının negatif rampa olmasına sebep olur.

Entegratör çıkışı, karşılaştırıcının ters çeviren girişini negatife itince, karşılaştırıcının çıkışı pozitife gider ve VE kapısını yetkilendirir. Böylece saat sayıcılara ulaşır. Entegratör çıkışının belli bir miktar sayım için negatif rampa üretmesine izin verilir. Bu, Şekil 6.8b de iki ayrı giriş gerilimi için t1 olarak gösterilmiştir. Sayıcı sabit sayıma ulaştığında, kontrol devreleri sayıcıları 0 a reset eder ve entegratör girişini negatif referans gerilimine çevirir. Negatif giriş gerilimi, Şekil 6.8 b de t2 olarak gösterildiği gibi, entegratör çıkışının pozitif rampa üretmesine sebep olur. Entegratör çıkışı 0 ın hemen üzerine ulaşınca, karşılaştırıcı çıkışı alçağa gider. Kontrol devresi, bu geçişi algılar ve sayıcı çıkışlarını mandallara yükler. Kontrol devresi, bundan sonra sayıcıları 0 a reset eder ve entegratör girişini tekrar giriş gerilimine çevirir. Bu, yeni bir çevrim başlatır. Mandallarda saklanan sayım miktarı, giriş gerilimi $V_{GİRİŞ}$ ile orantılıdır. Bunu şöyle açıklayabiliriz.

Entegratör çıkışı, sabit t1 zamanında $(V_{GİRİŞ}/RC).t1$ e eşit bir V gerilimine düşer. Tekrar 0 a ulaşmak için entegratör, aynı miktar bir gerilimi, bir pozitif rampayla üretmek zorundadır. Referansı entegre etme periyodu $t2$ için V gerilimi $(V_{REF}/RC).t2$ ye eşittir. V nin bu iki ifadesi eşitlendiğinde,

$$\frac{V_{GİRİŞ}}{RC} x\, t1 = \frac{V_{REF}}{RC} x\, t2$$

$$V_{GİRİŞ} x\, t1 = V_{REF} x\, t2$$

$$t2 = V_{GİRİŞ} x \frac{t1}{V_{REF}}$$

bulunur. RC denklemin her iki tarafında da bulunduğundan kısalır. Bunun pratik anlamı, sinyal entegre etme periyodu ile referans entegre etme periyodunda aynı direnç ve kapasitör kullanıldığından, R ve C deki değişimlerin, okunan çıkışın doğruluğu üzerinde hiçbir etkisi olmayacağı demektir. Bu, tek rampalı çeviriciye göre büyük bir avantajdır. Yukarıdaki denklemin düzenlenmesinden elde edilen sonuç, sayıcı çıkışı $t2$ nin, V_{REF} ve t1 sabit olduklarından, $V_{GİRİŞ}$ le doğru orantılı olduğunu göstermektedir.

Şekil 6.8 a da gösterilen devre için t1, 1 MHz lik saatin 1000 sayımına yani 1ms ye eşittir. V_{REF} ise -1 V tur. 2 V luk bir giriş sinyali için $t2$ nin değeri

(2 V/1 V)x1000=2000 sayımdır. Doğru yere konan onluk virgül, bunun 2,000 V u göstermesini sağlar. Şekil 6.8b de içeride kalan grafik, örneğin 0,8V gibi daha küçük bir giriş gerilimi için entegratör çıkışını göstermektedir. Bunun için, t2 nin değeri (0,8 V/1 V)x1000= 800 sayım olacaktır. Gösterge bunu 0,800 V olarak gösterir.

Çift eğimli *A/D* çevirici tipi birçok sayısal voltmetre ve diğer cihazlarda kullanıldığından, çalışmasını çok iyi anlamalısınız. Özetlersek: Bilinmeyen giriş gerilimi t1 sabit sayım miktarı kadar entegratöre uygulanır. Sayıcı 0 a reset edilir ve entegratör girişi referans gerilimine bağlanır. Entegratör çıkışının tekrar 0 a ulaşıncaya kadar gerekli sayım miktarı, giriş gerilimiyle doğru orantılıdır.

Çift eğimli çeviricinin avantajları doğruluğu, alçak maliyeti ve sıcaklığın R ve C de oluşturduğu değişimlerden etkilenmemesidir. Çift eğimli tipin dezavantajı hızının yavaş olmasıdır. 20.000 de 1 parça çözünürlüğü olan bir çevirici, her çevrim için 100 ms veya daha fazla süre alabilir. Bu tip, daha sonraki sayısal voltmetreler kısmında, daha fazla anlatılacaktır.

YÜK DENGELİ A/D ÇEVİRİCİLER

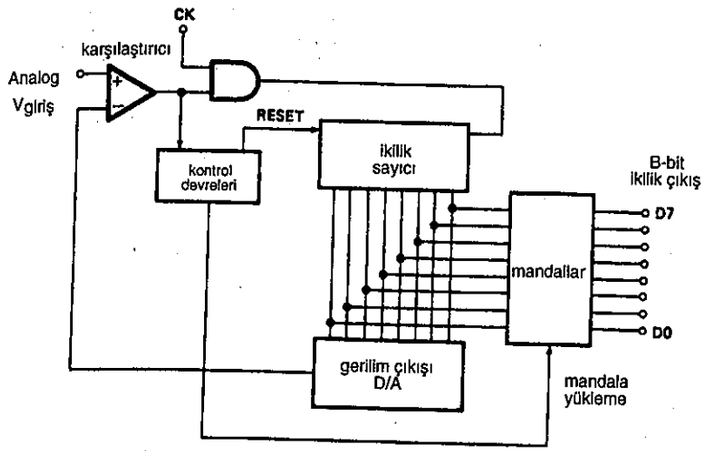
Yük dengeli tip *A/D* çeviriciler, temelde Şekil 6.8*a* daki çift eğimli çeviriciyle aynı devreyi kullanırlar. Ancak, girişin bilinmeyen gerilim ile referans gerilimi arasında ileri geri anahtarlanması yerine, giriş bilinmeyen gerilime bağlı bırakılır. Referans akımı darbeleri sabit bir süreyle doğrudan entegratörün toplama noktasına sokulur. Bu sabit süre içinde toplama noktasını 0V ta tutmak için gerekli referans akımı darbe sayısı, bilinmeyen giriş gerilimiyle orantılıdır. Bu tekniğin avantajı, entegre eden kapasitörün üzerindeki gerilimin 0 a çok yakın tutulmasıdır. Bu nedenle kapasitörün dielektrik soğurulması, yüksek hassasiyetli çift eğimli çeviricilerdeki gibi problem olmamaktadır.

SAYICILAR VE D/A ÇEVİRİCİLER KULLANAN A/D ÇEVİRİCİLER

TEK-SAYICI TİPİ : Daha önce bir sayıcının çıkışlarını, bir *D/A* çeviricinin girişlerine bağlayarak bir lineer rampanın nasıl üretilebileceğini anlatmıştık. Rampa, gerçekte küçük basamaklardan oluşmaktadır. Basamakların büyüklüğü çeviricinin çözünürlüğünün bit sayısıyla belirlenir.

Şekil 6.9, tek rampa tipi bir *A/D* çevirici yapmak için entegratörün yerine, gerilim çıkışlı *D/A* çeviriciler ve sayıcıların nasıl kullanılabileceğini göstermektedir. Çevrimin başında sayıcılar reset edilir ve *D/A* çeviricinin

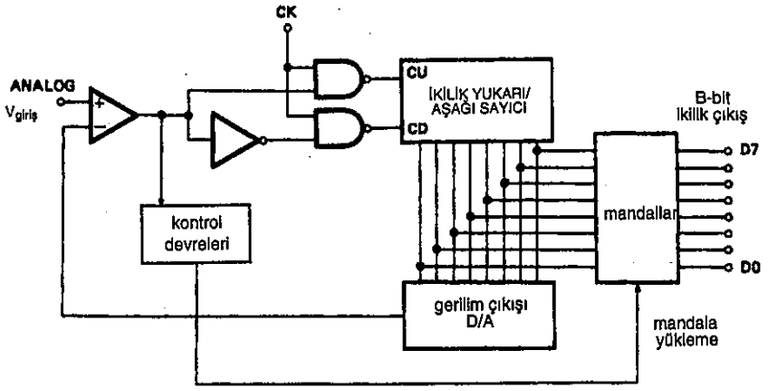
çıkışı 0 dır. Karşılaştırıcı girişine bir gerilim uygulandığında, çıkışı yükseğe gider ve saati sayıcılara bağlar. Her saat darbesi sayıcıları bir artırır ve *D/A* nın çıkış gerilimini bir adım yükseltir. *D/A* çeviricinin çıkışı, giriş geriliminden yüksek olunca karşılaştırıcının çıkışı alçağa düşer. Bu, sayıcılara giden saat darbelerini keser. Bundan sonra kontrol devreleri, sayıcı çıkışlarını kilitler ve tekrar yeni bir çevrime başlamak üzere sayıcıları reset eder.



Şekil 6.9: Tek rampa tipi *A/D* çevirici yapmak için kullanılan ikilik yukarı sayıcı ve *D/A* çevirici.

Bu yöntem çift rampa tipinden daha hızlıdır, fakat bir hassas *D/A* çevirici gerektirir. Diğer bir dezavantaj da, her çevrimde sayım 0 dan başlar ve *D/A* çıkışının $V_{GIRIŞ}$ e eşit olduğu yere kadar sayım artar.

İZLEMELİ *A/D* ÇEVİRİCİ: İzlemeli *A/D* çevirici, bir örnekten diğerine az bir miktar değişen ses sinyalleri gibi sinyallerin sayısallaştırılmasında biraz daha hızlı bir yöntemdir. Şekil 6.10 da gösterildiği gibi, bir izlemeli *A/D* çevirici, daha önceki tipteki sadece yukarı sayan sayıcı yerine, bir yukarı/aşağı sayıcı kullanır. Saat, karşılaştırıcının çıkışındaki VE DEĞİL kapı anahtarlarıyla, ya yukarı sayma ya da aşağı sayma girişine bağlanır. Bir giriş gerilimi için çevrimin başında *D/A* nın çıkışı 0 dır ve böylece karşılaştırıcının çıkışı yüksektir. Bu, saati yukarı sayma girişine bağlar. Sayıcı *D/A* çeviricinin çıkışı $V_{GİRİŞ}$ ten büyük oluncaya kadar yukarı doğru sayar.



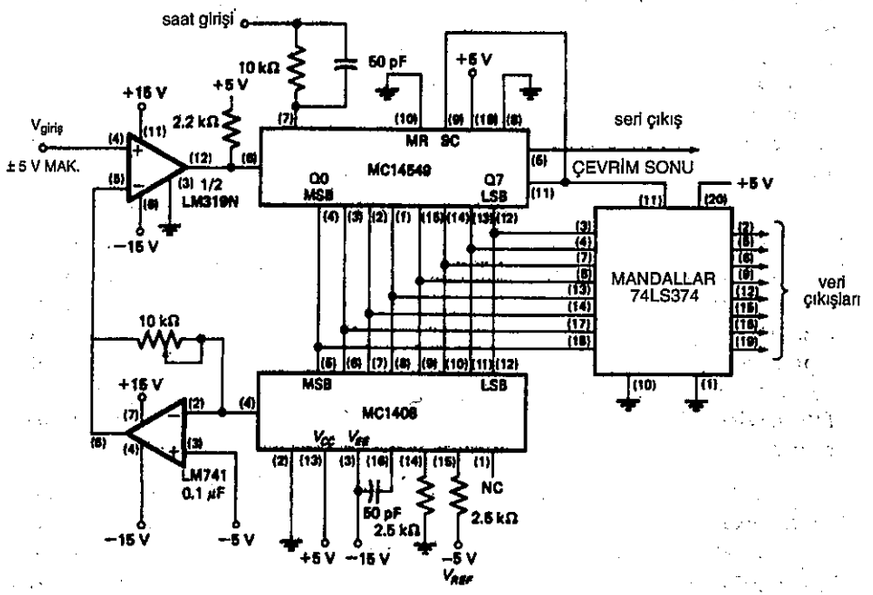
Şekil 6.10: İzlemeli *A/D* çevirici yapmak için kulanılan ikilik yukarı/aşağı sayıcı ve *D/A* çevirici.

Bundan sonra, karşılaştırıcı çıkışı alçağa düşer ve saati aşağı sayma girişine bağlar. Ancak, eğer $V_{GIRIŞ}$ değişmemişse, bir aşağı sayım, *D/A* çeviricinin çıkışını $V_{GIRIŞ}$ in altına düşürecektir. Karşılaştırıcı çıkışı, yükseğe çıkar ve saati tekrar yukarı sayma girişine bağlar. Bir yukarı sayma, karşılaştırıcı çıkışını bu döngüyü tekrarlamak üzere tekrar alçağa çeker. Bu tip bir çeviricide, sabit bir $V_{GİRİŞ}$ için en küçük değerlikli çıkış biti salınım gösterir. Bu bir dezavantajdır. Giriş sinyali değişirken, sayıcı onu izlemek üzere yukarı veya aşağı sayar. Çeviricinin adı, bu nedenle verilmiştir. Bu tipin avantajı, sinüs dalgalarını sayısallaştırırken, her seferinde 0 dan yukarı saymak yerine, bir örnekten diğerine geçerken, sadece bir kaç sayım yukarı veya aşağı saymasıdır.Bir sonraki kısımda anlatılan ardarda -yuvarlamalı tip bahsedilmiş olan her iki çeviriciden de daha hızlıdır ve salınım yapmaz.

ARDARDA-YUVARLAMALI A/D ÇEVİRİCİ

Ardarda-yuvarlamalı *A/D* çeviricinin önemli avantajı, sadece N saat darbesiyle, *N*-bitlik çözünürlüğün elde edilebilmesidir. Örneğin, 8-bitlik bir ardarda-yuvarlamalı tip, sayıcı tipinin gerektirdiği 256 ya kadar saat darbesine karşın, sadece 8 saat darbesi gerektirir. Şekil 6.11 ardarda-yuvarlamalı tip *A/D* çevirici için bir devre göstermektedir. Bu devrenin kalbi, aşağıdaki şekil-

de çalışan MC14549 gibi bir *ardarda -yuvarlama kaydedicisidir (Successive-approximation register:SAR).* Çevrimin başında, ilk saat darbesiyle, SAR, MC1408 *D/A* çeviriciye giden en büyük değerlikli çıkış bitini yüksek yapar. Sonra, SAR, *D/A* çıkışının giriş geriliminden daha büyük veya küçük olduğunu belirten LM319 karşılaştırıcı sinyalini bekler. Eğer karşılaştırıcı çıkışı yüksekse, bu takdirde *D/A* çıkışı $V_{GİRİŞ}$ ten daha azdır ve SAR en büyük değerlikli bitini set edilmiş olarak bırakır. Karşılaştırıcı çıkışı alçaksa, *D/A* çıkışı $V_{GİRİŞ}$ ten daha büyüktür ve SAR en büyük değerlikli bitini reset eder. Her iki halde de, bir sonraki saat darbesinde, SAR, bir sonraki en büyük değerlikli bitini set eder. Karşılaştırıcının çıkışına bağlı olarak, SAR, bu biti ya aynen bırakır veya reset eder. Her seferinde bir biti deneyerek SAR en küçük değerlikli bite kadar bu işlemlere devam eder. Eğer üretilen *D/A* çıkışı $V_{GIRIŞ}$ ten daha az ise SAR, ilgili biti set olarak bırakır ve *D/A* çıkışı $V_{GIRIŞ}$ ten daha büyükse o biti reset eder. Her bit için sadece bir saat darbesi gerekmektedir. Bütün bitler denendikten sonra SAR, bir *çevrim-sonu (end-of-conversion:EOC)* sinyali gönderir. EOC



Şekil 6.11: Ardarda - yuvarlamalı *A/D* çevirici.

sinyali, paralel çıkış hatlarının, analog giriş sinyalinin büyüklüğünü temsil eden geçerli bir ikilik kelime içerdiğini belirtmektedir. Eğer EOC sinyali, *çevrime başla (start conversion:SC)* girişine bağlanırsa çevirici sürekli çevrim yapacaktır. MC14549 un ayrıca, SAR belirledikçe sayısal kelimenin 1 bitini çıkışa veren bir seri çıkışı vardır.

A/D çevirmede ardarda yuvarlama yöntemi, 128, 64, 32, 16, 8, 4, 2 ve 1 cm yüksekliğinde ikilik ağırlıklı blokları üstüste koyarak bir masanın yüksekliğini bulmaya benzer. 128 cm lik en önemli blok önce denenir. Eğer bu blok çok yüksekse bir kenara konur ve ilgili bit konumuna 0 yazılır. Sonra bir sonraki en büyük değerlikli blok yığına eklenerek denenir. Bir blok, yığına eklendiğinde, yığın masadan yüksek oluyorsa, o blok bir kenara konur ve ilgili bite 0 yazılır. Eğer eklenen blok, yığını masadan daha yüksek yapmıyorsa, bu takdirde ilgili bit için 1 yazılır. Her bloğun sadece 1 defa denenmesi gerekir. Tüm bloklar denendikten sonra elde edilen sonuç, masanın yüksekliğini gösteren 8-bitlik bir ikilik kelimedir.

Şekil 6.11 de gösterilen devre hazırda varolan parçaları kullanmaktadır ve isterseniz devreyi kurabilmeniz için bacak numaralarıyla gösterilmiştir. Maksimum giriş aralığı ±5V tur. Saat girişindeki *R* ve *C*, saati sağlayan darbe üretecinin kapatılmasından önce, *A/D* çeviricinin gücünün kazara kapatılması durumunda, MC14549 a bir zarar gelmesini önlemek içindir. Akımdan gerilime çevirici işlemsel kuvvetlendiricinin pozitif girişinin, toprak yerine -5V a bağlanmış olduğuna dikkat edin. Bu, analog giriş aralığını 0 ile +10V arası yerine, -5 ile +5V arasına kaydırır. Böylece a.a. sinüs dalgaları doğrudan *A/D* girişine bağlanabilir.

Bir analog sinüs dalgası sinyalinin, sayısal değerlerinden nasıl yeniden oluşturulabileceğini göstermek için, bir MC1408L-8, bir LM741 ve Şekil 6.11 in *D/A* kısmındaki aynı değerleri kullanarak bir başka *D/A* çevirici kurabilirsiniz. Bu *D/A* çevirici mandalların çıkışlarına bağlanır. *A/D* çeviricinin bir 5V tepeli 1 KHz lik sinüs dalga girişi, *D/A* çeviricinin çıkışıyla karşılaştırılabilir. *A/D* nin giriş saat frekansını 1KHz den yaklaşık 1MHz e kadar değiştirmek ve *D/A* çeviricinin çıkış dalgaşekli üzerindeki etkileri gözlemek öğretici olacaktır.

Ardarda-yuvarlamalı *A/D* çeviricinin dezavantajı, bir *D/A* çevirici gerektirmesi, avantajı ise yüksek hızda mükemmel çözünürlüğü olmasıdır. Analog Devices, Inc., bir 10-bitlik çeviricinin bütün analog ve sayısal devrelerini AD571 de tek bir yonga üzerine koymak için I^2L kullanmaktadır. Micro Networks 'ün MN5245 i, 12-bitlik ardarda yuvarlamalı çevrimi yaklaşık 900 ns de yapmaktadır.

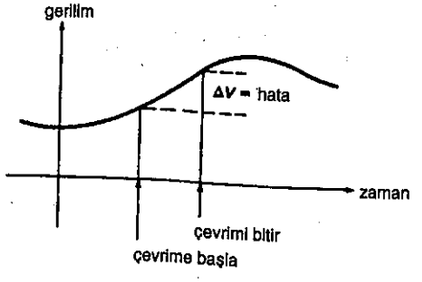
A/D ÇEVİRİCİLERİN ÖZELLİKLERİ

Çözünürlük, doğruluk, lineerlik vb. gibi daha önce *D/A* çeviricilerde anlatılan özellikler *A/D* çeviriciler için de kullanılır. *A/D* çeviricilerde sık kullanılan diğer bir terim de niceleme hatası, yani gerçek analog değerle bunun sayısal gösterilişi arasındaki hatadır. İdeal olarak maksimum niceleme hatası en küçük değerlikli bitin ±1/2 si kadardır, yani bu, çözünürlüğü belirtmenin diğer bir yoludur.

A/D çeviricilerde diğer bir önemli parametre çevrim süresi, yani her örneği sayısallaştırmak için geçen süredir. Bu süre, çevrim tipine ve bit sayısına bağlı olarak nanosaniyelerle milisaniyeler arasında değişir. Yüksek çözünürlüklü, alçak hızlı uygulamalar genellikle çift eğimli tipleri kullanır. Yüksek hızlı, orta çözünürlüklü uygulamalar, ardarda yuvarlamalı tip çevirici kullanır ve çok yüksek hızlı uygulamalarda paralel karşılaştırıcılı yöntem kullanılmak zorundadır.

ÖRNEKLE -VE- TUT DEVRELERİ

Hızla değişen bir giriş sinyali sayısallaştırılmaya çalışılırsa, giriş sinyalinin değeri çevrim tamamlanmadan önce değişecektir. Çeviricinin çıkışı, çevrimin başlangıcı yerine sonundaki girişi gösterecektir. Şekil 6.12 bu hatayı göstermektedir. Hata gerilimi ΔV, giriş sinyalinin ne kadar hızlı değiştiğinin bir fonksiyonudur. Bu hata, tampon yükselteçli bir örnekle -ve- tut devresini *A/D* çeviricinin girişinde kullanarak yok edilebilir veya büyük ölçüde azaltılabilir. Çok hızlı bir şekilde açılıp kapatılabilen bir elektronik anahtar, t1



Şekil 6.12: *A/D* çevrim süresi sırasında değişen giriş geriliminin sebep olduğu hata.

anında giriş dalgaşeklinin bir örneğini hızlı bir şekilde alıp tutma kapasitörü üzerinde saklamak üzere kullanılabilir. Sonra A/D çevirici saklanmış olan bu gerilimi sayısallaştırır. Çevrim tamamlandıktan sonra, yeni bir örnek alınır ve bir sonraki çevrim için tutulur.

Örnekle -ve- tut devrelerinin iki önemli parametresi, yakalama süresi (acquisition time) ve açıklık süresi (aperture time) dir. Yakalama süresi, anahtarın kapanması ve kapasitörün giriş sinyali gerilimine şarj etmesi için gerekli süredir. Açıklık süresi, tutma sinyalinden sonra anahtarın tamamen açılması için gerekli süredir. İdeal olarak, örnekleme sırasındaki sinyal değişmelerinin etkisini yok etmek için her iki sürenin de 0 olması gerekir.

Bir giriş sinyalini sayısallaştırmak üzere örneklerken hatırlanması gereken bir başka önemli nokta vardır. Örnekleme teoremine göre bir sinüs dalga girişi, dalgaşeklinin bir periyodunda en az 0 olmayan iki noktada örneklenmelidir. Gerçekte, bir sinüs dalga girişinin iyi bir sayısal gösterilişini elde etmek için çok daha yüksek bir örnekleme frekansı gerekmektedir. Çocukların çizdiği noktaları birleştirme resimleri gibi, ne kadar çok nokta varsa, resmin çözünürlüğü de o kadar iyidir.

Eğer Şekil 6.11 deki ardarda-yuvarlamalı A/D çeviriciyi kurar ve çıkışına bir D/A çevirici bağlarsanız, saatin frekansını değiştirerek bir sinüs dalgası girişinin, çıkışta yeniden üretilmesinde, örnekleme hızının etkisini gözleyebilirsiniz.

UYGULAMALAR

Bir gerilim olarak varolan hemen hemen bütün ölçülebilir miktarlar, bir A/D çeviriciyle sayısallaştırılabilir ve bir sayısal göstergeyle görüntülenebilir. Bir yarıiletken birim uzama ölçerinin gerilim çıkışı yükseltilir, sayısallaştırılır ve bir sayısal okuma ölçeğinde görüntülenir. Sayısal göstergeli termometreler, sıcaklığa hassas bir diyodun üzerindeki gerilim düşümünü görüntülemek için bir A/D çevirici kullanarak sayısal şekle çevirirler. Laboratuvarda, A/D çeviriciler, sayısal voltmetrelerin ve sayısal multimetrelerin kalbidir. Analog ses sinyalleri, uzun mesafelere iletim için sayısal şekle çevrilir ve hedeflerine vardıklarında yeniden analog şekle çevrilirler.

SAYISAL GÖSTERGELİ VOLTMETRELER VE MULTİMETRELER

Sayısal voltmetreler (DVM ler) halen 3 1/2 , 4 1/2 veya 5 1/2 basamaklı birimler olarak bulunmaktadırlar. 1/2 basamağı, en öndeki basamağın sadece 0 veya 1 i gösterebildiğini anlatmaktadır.

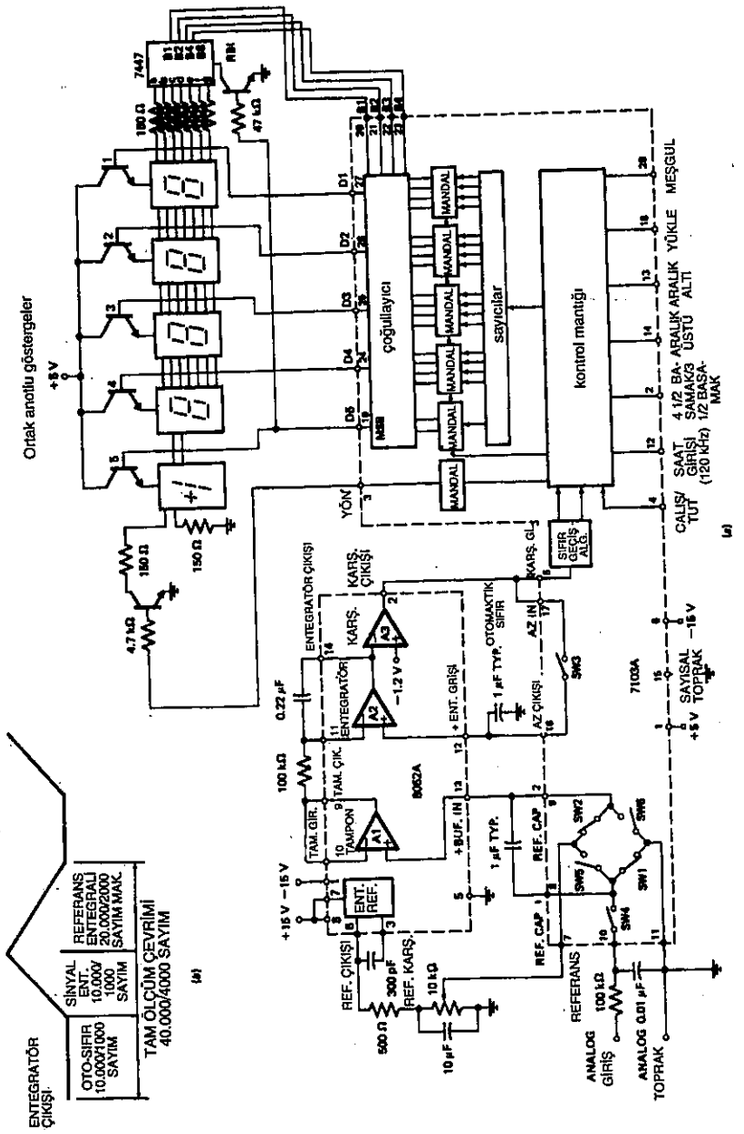
Şekil 6.13 a, Intersil 8052A ve 7103A kullanan bir 4 $^1/_2$ basamaklı DVM nin A/D çevirici ve gösterge kısımlarının şemasını göstermektedir. Devrenin tam-ölçek giriş gerilim aralığı ±1,9999 V tur. 8052A, bir çift eğimli çevirici için referans kaynağını, tampon yükseltecini, entegratörü ve karşılaştırıcıyı içerir. 7103A, 4 $^1/_2$ basamaklı gösterge için bir çoğullayıcı ve mandalların yanısıra çevirici için MOS anahtarlar, sayıcılar ve kontrol mantığını içerir.

Devre, doğruluğunu artıran ve pozitif ve negatif gerilimlerin sayısallaştırılmasını mümkün kılan iki değişiklik dışında, daha önce anlatılan çift eğimli çevirici olarak işlev görür. Çevrim süreci iki yerine üç fazdan oluşur. Bu üç faz, Şekil 6.13b de gösterildiği gibi, otomatik 0, sinyal integrali ve referans integrali fazlarıdır. Otomatik 0 fazı sırasında 1,2 ve 3 anahtarları kapalıdır. Anahtarlar 1 ve 2 yi kapamak V_{REF} gerilimine eşit bir gerilimi referans kapasitörü üzerinde saklar. Anahtar 3 ü kapamak, otomatik 0 kapasitörü üzerindeki kayma veya kaçakların etkilerini saklar. Saklanmış hatalar, sinyal ve referans integrali fazlarında entegratör için referans olarak alındığından yokedilmiş olur. Bu, çeviricinin doğruluğunu büyük ölçüde artırır.

Sinyal entegrali fazı sırasında 10.000 saat darbesi için anahtar 4 kapalı ve anahtarlar 1, 2 ve 3 açıktır. Eğer giriş gerilimi pozitif ise entegratör negatif bir rampa üretecektir ve yön flip-flopunu set edecektir. Giriş gerilimi negatif ise, entegratör pozitif bir rampa üretecek ve yön flip-flopu set edilmeyecektir.

Referans integrali fazı için anahtar 4 açılır ve yön flip-flopu anahtar 5 in mi yoksa 6 nın mı kapalı olacağını belirler. Eğer giriş pozitifse anahtar 6 kapatılır. Bu, otomatik 0 fazında uygulanan gerilimden V_{REF} kadar daha negatif olan tampon girişini toprağa bağladığından entegratör, karşılaştırıcı eşiğini geçinceye kadar bir pozitif rampa üretir. Entegratör çıkışının tekrar 0 a ulaşması için gerekli sayım miktarı, daha önce anlatıldığı gibi giriş gerilimiyle orantılıdır. Eğer giriş gerilimi negatifse, anahtar 5 kapatılır. Bu referans kapasitöründe saklanan V_{REF} gerilimine daha pozitif V_{REF} gerilimini de ekleyerek tampon girişine verir. Entegratör, 0 a ulaşıncaya kadar negatif bir rampa üretir. V_{REF} i bir kapasitörde saklamak ve referans integrali fazında entegratörün pozitif veya negatif rampa üretmesini sağlamak, bu çeviricinin pozitif veya negatif giriş sinyalleriyle çalışmasına imkan verir.

Bu devre, 7103 ün bacak 2 si yüksekken 4 $^1/_2$ basamaklı bir birim veya bacak 2 alçakken 3 $^1/_2$ basamaklı bir birim olarak çalıştırılabilir. 3 $^1/_2$ basamaklı birimler için 12 KHz lik bir saat kullanılır. Bir aralık üstü çıkışı, giriş geriliminin 2,0000 V tan büyük olduğunu ve bir aralık altı çıkışı, giriş geriliminin tam ölçeğin yüzde 9 undan az olduğunu gösterir. Bunlar LED li sürücü transistörlere bağlanarak, başka bir aralığın kullanılması gerektiğini belirtebilirler.



Şekil 6.13: *(a)* **Intersil 8052A ve 71013A yı kullanan tam bir 4 1/2 basamaklı *A/D* çeviricinin şeması.** *(b)* **Negatif giriş gerilimi için entegratör çıkışı.**

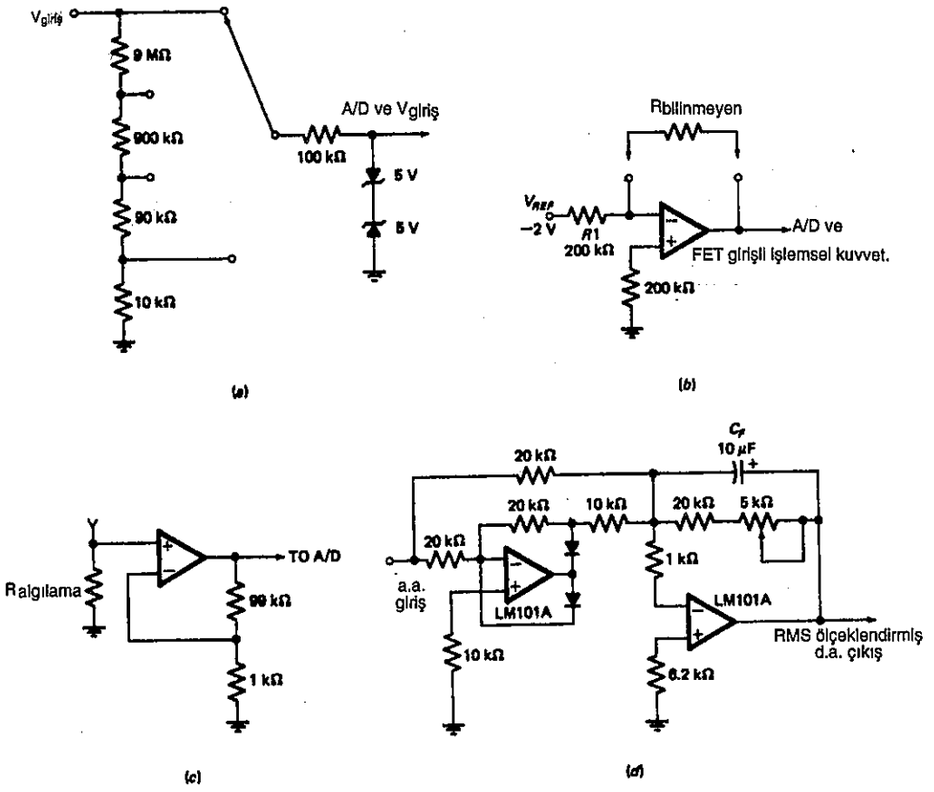
AKIM, DİRENÇ, A.A. GERİLİM ÖLÇMEK İÇİN EK DEVRELER

Temel *A/D* çeviriciyi faydalı bir laboratuvar cihazı yapmak için Şekil 6.13a daki temel *A/D* birimine birçok devre eklenmiştir. İlk eklenen Şekil 6.14a da gösterilene benzer bir hassas metal-film dirençlerden oluşan zayıflatıcı devredir. Bu, daha yüksek gerilimlerin ölçülmesine imkan verir. Devrede aşağı doğru her adım, giriş gerilimini 10 la böler. Örneğin 500 V luk bir girişle, B kavşağı 50V, C kavşağı 5V ve D kavşağı 0,5V olacaktır. Sadece D anahtar konumu, ±1,9999V luk *A/D* aralığı içinde olduğundan bu konum 500V u sayısal değere çevirmek için kullanılır. Aynı anahtar üzerinde başka bir bölüm, göstergedeki onluk virgülü uygun konuma getirebilir. 100KΩ luk bir dirençle sırt sırta olan zener diyotları aşırı gerilim-koruyucu olarak görev yapar. Bunlar, *A/D* çeviriciye giden gerilimi, aralık anahtarı başlangıçta yanlış konuma ayarlandıysa ±5,7V la sınırlar. 100KΩ luk direnç giriş sinyalini zayıflatmaz çünkü *A/D* çeviricinin giriş empedansı 1000MΩ dan büyüktür.

İkinci ilave, DVM nin bir ohmmetre olarak çalışmasını sağlayan devredir. Şekil 6.14 b bunu yapmanın bir yolunu göstermektedir. Değeri bilinmeyen bir direnç, işlemsel kuvvetlendiricinin çıkışı ile ters çeviren girişi arasına bağlandığında, negatif geribesleme, ters çeviren girişi 0 V ta, yani varsayılan toprakta, tutar. R1 in üzerinde sabit bir -2V luk gerilim olduğundan, R1 den geçen akım sabittir. Bu sabit akım bilinmeyen R den akarak *DVM* devresiyle ölçülebilecek bir çıkış gerilimi oluşturur. Değişik R1 değerleri anahtarlanarak farklı direnç aralıkları elde edilebilir. Gösterilen devre için maksimum bilinmeyen direnç 200KΩ dur.

Diğer bir geliştirme, akımın ölçülebilmesi için bir devre eklemekle yapılabilir. Şekil 6.14c bunu yapmanın bir yolunu göstermektedir. Bilinmeyen akım, bir gerilime çevrilmek üzere algılama direncinden geçirilir. Algılama direncinin üzerindeki küçük gerilim *A/D* çeviriciye gönderilmeden önce 100 defa yükseltir. Algılama direncinin üzerindeki gerilimi yükseltmek çok daha küçük bir algılama direnci kullanılmasına imkan verir. 2mA lik bir tam-ölçek okuma için sadece 10Ω luk bir algılama direnci gerekmektedir. Algılama direnci ne kadar küçükse, ölçüm yapılan devreyi o kadar daha az etkilemesi söz konusudur.

Bir *DVM* veya *DMM*, d.a. gerilimlerinin yanısıra a.a. gerilimleri de ölçebilmelidir. a.a. gerilimleri ölçmek için, bu gerilimler önce d.a. gerilimlere çevrilmelidir. Güç uygulamalarında basit bir diyot veya diyot köprüsü bu çevrimi yapar. Ancak ölçme uygulamalarında, diyot üzerindeki 0,6 V luk düşme çok büyük hataya sebep olur. Şekil 6.14d deki devre bir a.a. girişin tam d.a. dengini üretmek için iki işlemsel kuvvetlendirici kullanmaktadır. Birinci işlemsel kuvvetlendirici, bir aktif yarım dalga doğrultucusudur. İkinci işlem-



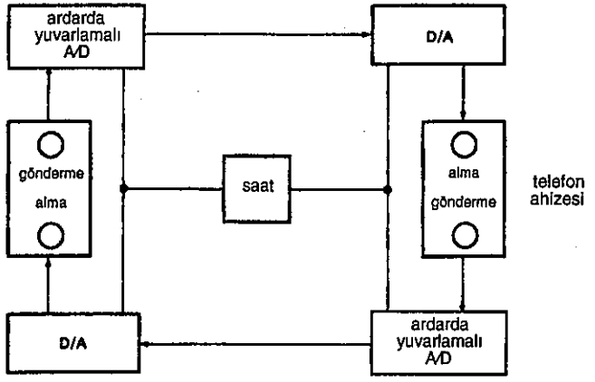
Şekil 6.14: *DMM* yapmak için *A/D* çeviriciye ilaveler. *(a)* Bir A/D çeviricinin giriş gerilim aralığını genişletmek için zayıflatıcı girişi. *(b)* Dirençten gerilime çevirici. *(c)* Akımdan gerilime çevirici. *(d)* a.a. dan d.a. ya gerilim çevirici.

sel kuvvetlendirici a.a. girişini yarım dalga doğrultucunun çıkışı ile birleştirerek tam dalga doğrultulmuş d.a. üreten bir toplayıcıdır. Bu d.a., C_F tarafından filitrelenir ve 5KΩ luk değişken direnci ayarlayarak etkin *(rms)* değere ölçeklenir.

SAYISAL SES İLETİMİ VE CODEC LER

Video veya ses iletişiminde olduğu gibi analog sinyaller uzun mesafelere iletildiklerinde gürültülü hale gelirler. Gürültü rastgele olduğundan filitrelemesi çok zor, hatta imkansızdır. Ancak, sayısal sinyaller, 1 ve 0 gibi sadece iki seviyeleri olduğundan, küçük gürültü gerilimleri tarafından etkilenmezler. Bu nedenle Ay'dan veya Mars'tan anlaşılabilir bir sinyali geri almak istiyorsanız, sinyali göndermeden önce sayısallaştırmakta ve alıcı ucunda, gerekiyorsa analog sinyali yeniden oluşturmak üzere, gelen sinyali bir *D/A* çeviriciden geçirmekte yarar vardır.

Şekil 6.15 bir tek kanallı sayısal iletişim bağını göstermektedir. Her uçta, bir ardarda yuvarlamalı *A/D* çevirici ses sinyalini, bir seri sayısal şekle çevi-



Şekil 6.15: Sayısal ses iletişim bağı.

rir. Alıcıda bir *D/A* çevirici, ses dalga şekillerini yeniden oluşturur. Ortak bir saat, her iki ucun eş zamanlı olarak çalışmasını sağlar.

Intel 2910 gibi bir kodlayıcı-kod çözücü (coder-decoder : *CODEC*) entegre devresi böyle bir sistemin her bir ucu için gerekli *A/D* ve *D/A* çeviricileri içermektedir. Hat veya radyo kanalı sayısını azaltmak için birçok *CODEC* çıkışı, tek bir çift hat veya kanala çoğullanabilir. Bu, sayısal şekilde veri göndermeye darbe kod modülasyonu (pulse code modulation: *PCM*) denir.

GÖZDEN GEÇİRME SORULARI VE PROBLEMLER

1. Negatif geri beslemeli bir işlemsel kuvvetlendiricinin pozitif girişi topraklanmışsa, negatif girişinde hangi gerilim ölçülür?

2. Negatif geri beslemeli bir işlemsel kuvvetlendiricinin ters çevirmeyen girişi -5V a bağlanmışsa, ters çeviren girişinde hangi gerilimi bulmayı beklersiniz?

3. a) Bir 6-bitlik ağırlıklı direnç tipi *D/A* çeviricinin en küçük değerlikli basamak direncinin değeri 80KΩ dur. Diğer beş direncin değeri ne olur?
   b) Bu çeviricinin kaç tane ayrık çıkış basamağı, yani seviyesi vardır?
   c) Ağırlıklı direnç tipi *D/A* çeviricilerin ne gibi bir sorunu vardır?

4. Bir 4-bitlik *R/2R D/A* çeviricinin devresini çizin. En büyük değerlikli bit anahtarının yanındaki anahtarın V_{REF} e bağlı olması ve diğer üçünün toprağa bağlı olması halinde denk devreyi çizin. *R/2R* tipinin, ağırlıklı direnç tipine göre avantajı nedir?

5. 12-bitlik bir *D/A* çeviricinin çözünürlüğü nedir? Eğer tam-ölçek çıkış gerilimi 10V ise, en küçük değerlikli bit hangi büyüklükte bir gerilim basamağını temsil eder?

6. Bir *D/A* çevirici için aşağıdakileri tanımlayın:

   a) Lineerlik hatası
   b) Kazanç hatası,
   c) Kayma hatası,
   d) Monotonluk,
   e) Bu hataların her birinin, gerçek çıkışın beklenen çıkıştan nasıl farklı olmasına sebep olacağını çizerek gösterin.

7. 12-bitlik bir *D/A* çeviricinin çözünürlüğüyle uyuşması için nasıl bir doğruluğu olması gerekir?

8. a) Bir 6-bitlik paralel *A/D* çevirici kurmak için kaç tane karşılaştırıcı gerekir? Bu sayı genelde pratik midir?
   b) Paralel tip *A/D* çeviricinin ana avantajı nedir?

9. Şekil 6.6b deki karşılaştırıcı çıkış kodunu ikiliğe çevirmek için gerekli kapı devrelerini gösterin.

10. Bir tek rampalı *A/D* çeviricinin blok diyagramını çizin ve çalışmasını kısaca anlatın. Tek rampalı tiplerin bir dezavantajı nedir?

11. Bir çift eğimli *A/D* çeviricinin blok diyagramını çizin ve çalışmasını kısaca anlatın. Açıklamalarınızda referans integrali ve sinyal integrali terimlerini içerin. Çift eğimli *A/D* çevirici, tek eğimli tipin ana problemini nasıl çözmektedir?

12. *a)* Şekil 6.8a daki entegratörde, 1,5V luk bir giriş gerilimi için hangi eğim çıkışı verilecektir?
    b) 1MHz lik saatin 1000 sayımından sonra, çıkış hangi negatif gerilime ulaşacaktır?
    c) Şekil 6.8a daki devreyle, 1,5V luk girşi sayısallaştırmak ne kadar zaman alacaktır?

13. 100 KHzlik bir giriş saati kullanarak, Şekil 6.9 daki 8-bitlik sayıcı tipi *A/D* çeviricinin bir tam ölçek sinyali sayısallaştırması ne kadar zaman alacaktır?

14. İzlemeli *A/D* çevirici, tek sayıcılı tipten nasıl farklıdır?

15. *a)* Bir 8-bitlik ardarda yuvarlamalı tip *A/D* çeviricinin her çevrimi için kaç saat darbesi gerekmektedir?
    b) 100 KHz lik bir saatle bir çevrim ne kadar zaman alır?
    c) Bu süreyi 8-bitlik sayıcı tipin süresi ile karşılaştırın.

16. Bir ardarda yuvarlamalı tip *A/D* çeviricinin blok diyagramını çizin ve çalışmasını kısaca anlatın.

17. *a)* Hızı değişen sinyalleri sayısallaştırırken niçin bir örnekle -ve- tut kullanmak gereklidir?
    b) Bir sinüs dalgası ne kadar sıklıkla örneklenmelidir?
    c) Bir örnekle-ve- tut için yakalama süresini ve açıklık süresini tanımlayın.

18. Bir $3\,1/2$ basamaklı *DVM* de 1/2 basamak ne demektir?

19. Bir *A/D* çeviriciyi bir sayısal multimetreye dönüştürmek için hangi ek devreler gerekir?

20. Şekil 6.13a daki devre için üç çevrim fazını anlatın.

21. Bir *A/D* çeviricinin gerilim aralığını genişletmek için, girişinde kullanılabilecek bir zayıflatıcı devrenin şemasını çizin.

22. Sayısal ses iletişiminin avantajı nedir?

23. *CODEC* i tanımlayın.

Referanslar / Bibliyografya

Datel Systems, Inc. in Teknik Elemanları: "DATEL Systems Products Handbook. "*Electronic Design's GOLD BOOK.* cilt 3, 1976-77. sayfa 3/6-3/18, 3/87-3/90.

Linear Applications Handbook, National Semiconductor Corporation, Santa Clara, CA, 1980.

McMOS Integrated Circuits Data Book. Motorola Semiconductor Products. Inc., Phoenix, AZ, 1973.

Linear Integrated Circuits Data Book. Technical Information Center. Motorola Semiconductor Products. Inc., Phoenix. AZ. 1979.

Analog Devices, Inc. in Mühendis Elemanları. (Editör: Daniel H.Sheingold): *Analog-Digital Conversion Handbook,* Norwood, MA. 1972.

Intersil $4\,1/2$ *Digit Pair 8052A/7103A:* $3\,1/2$ *Digit Pair 8052/7103,* Data Sheet, Intersil, Inc., Cupertino, CA. 1977.

Data Acquisition Handbook, National Semiconductor Corporation, Santa Clara, CA, 1978.

BÖLÜM 7

SAYISAL ARİTMETİK

A/D çeviricilerin çeşitli kullanılışlarından biri de *ağırlık, basınç, sıcaklık,* v.b. temsil eden *analog gerilimleri sayısal kelimelere çevirmektir.* Bu görüntülemek amacıyla yapılır, böylece ağırlıkları toplamak veya ağırlığı birim fiyatla çarpmak gibi matematik işlemleri daha kolay yapılır. Gerekirse, D/A çeviricileri sonuçları analog akım veya gerilimlere gerisin geri çevirebilir. Elektronik hesap makinaları bütün işlemlerini ikilik-kodlu sayılarla gerçekleştirir.

Bu bölümde ikilik aritmetiği ve işlemleri yapmak için kullanılan devreleri anlatacağız.

Bu bölümü bitirdiğinizde, şunları yapabileceksiniz:

1. İkilik, BCD, 3-fazla BCD, sekizlik ve onaltılık sayıları doğru olarak toplayabileceksiniz.
2. İkilik sayıları, 1 in-tümleyeni veya 2 nin-tümleyeni yöntemini ve bir işaret biti kullanarak, doğru olarak çıkarabileceksiniz.
3. BCD, 3-fazla BCD, sekizlik ve onaltılık sayıları doğru olarak çıkarabileceksiniz.
4. 74181 gibi bir ALU nun, mantık fonksiyonlarından herhangi birisini gerçekleştirmede veya iki 4-bitlik giriş değişkeni üzerinde 16 aritmetik fonksiyonu gerçekleştirmede nasıl kullanılabileceğini bir devre ile gösterebileceksiniz.
5. Kalem yöntemiyle veya topla - ve - sağa - kaydır algoritmasıyla ikilik çarpmayı yapabileceksiniz.
6. Kalem yöntemiyle veya çıkar - ve - sola - kaydır algoritmasıyla ikilik bölmeyi yapabileceksiniz.
7. Sayıları, bir ikilik kesir şekline normalize edebileceksiniz.
8. Kayan virgüllü sayıları, bir ikilik, bir ANSI FORTRAN onaltılık veya bir BCD formatında temsil edebileceksiniz.

TOPLAMA

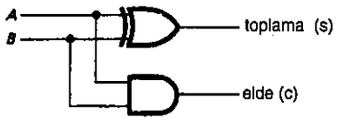
İKİLİK

Şekil 7.1 a, A ve B gibi iki tane tek-bitlik sayının toplanmasına ait doğruluk tablosunu göstermektedir. Burada S, A ve B' nin toplamını, ve C'de bir

sonraki basamağa olan ikilik eldeyi göstermektedir. Her basamak yalnız 0 veya 1 olabileceğine göre, $A = 1$ artı $B = 1$ toplamının 0 ve bir sonraki basamağa eldesinin 1 olduğuna dikkat edin. $S = 1$ yapan mantık ifadesi $A \oplus B$ dir ve C nin ifadesi $A \cdot B$ dir. Şekil 7.1 *b* de gösterildiği gibi, bu fonksiyonlar, kapılarla kolayca gerçekleştirilebilir.

GİRİŞLER		ÇIKIŞLAR	
A	*B*	*S*	*C*
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

(a)



(b)

Şekil 7.1 Yarım toplayıcı. *(a)* Doğruluk tablosu. *(b)* Devre.

Şekil 7.1 b deki devre sadece bir 'yarım' toplayıcıdır, çünkü bir önceki basamaktan gelecek eldeyi hesaba katmamaktadır. Şekil 7-2 *a*, bir önceki basamaktan gelen $C_{GİRİŞ}$ eldesini hesaba alan bir 'tam' toplayıcının doğruluk tablosunu göstermektedir. Eğer A, B ve $C_{GİRİŞ}$ in hepsi de 1 ise, bu takdirde hem toplamın hem de elde çıkışının 1 olduğuna dikkat ediniz. S nin mantık ifadesi $A \oplus B \oplus C_{GİRİŞ}$ tir. Üç veya daha fazla değişkenin özel VEYA sının *(XOR)*, tek sayıda değişken 1 ise, 1 olduğunu hatırlayın. C_0 elde çıkışının ifadesi $A \cdot B + C_{GİRİŞ}(A \oplus B)$ dir.

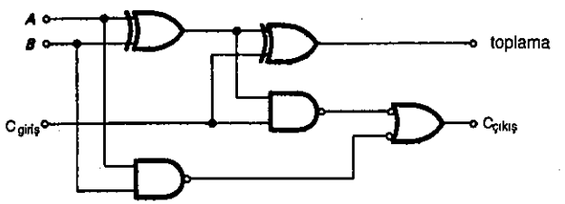
Şekil 7.2 *b* de gösterildiği gibi, tam toplayıcı, kapılar kullanarak kurulabilir. Genellikle ayrık kapılardan toplayıcılar kurmaya gerek yoktur; fakat şema 7,1 .*b* ve 7.2 *b* deki devreler, mantık ifadeleriyle ikilik toplama arasındaki ilişkiyi çok iyi bir şekilde göstermektedir. Tek-bitlik tam toplayıcı 7480 de, ve bir 4-bitlik tam toplayıcı da 7483 te hazır yapılmış olarak bulunmaktadır. 7483, iki 4-bitlik kelimeyi, bir 4-bitlik toplam ve bir elde çıkışı vermek üzere toplar. Her biri 4-bitten uzun olan iki sayısal kelimeyi toplamak için birçok 7483 paralel olarak bağlanabilir. Uzun kelimeleri toplamak için gerekli olan toplayıcı sayısını azaltmak için, Şekil 7.3 te gösterilene benzer bir devre kullanılabilir.

GİRİŞLER			ÇIKIŞLAR	
A	B	$C_{GİRİŞ}$	S	$C_{ÇIKIŞ}$
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$S = A \oplus B \oplus C_{giriş}$

$C_{çıkış} = A \cdot B + C_{GİRİŞ}\ (A \oplus B)$

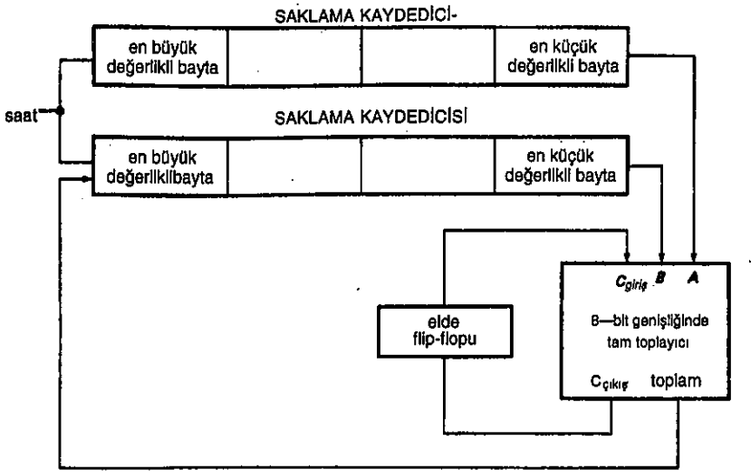
(a)



(b)

Şekil 7.2 Tam toplayıcı. *(a)* Doğruluk tablosu. *(b)* Devre.

Toplanacak kelimeler iki kaydedicide saklanmaktadır. Bunlar, her seferinde 8 er bit olarak toplayıcıya kaydırılır ve toplanır. Toplayıcı, her seferinde 1, 2, 4 veya 8 bit toplamak üzere tasarlanabilir. Elde çıkışı, bir sonraki toplamada elde girişi olarak kullanılmak üzere geciktirilir veya saklanır. Toplam çıkışı, saklanmak üzere kaydediciye kaydırılabilir.



Şekil 7.3 Uzun kelimeleri toplamak için kullanılan 8-bitlik tam toplayıcı.

BCD VE 3-FAZLA BCD TOPLAMA Bir hesap makinasında olduğu gibi hesapların en son sonucunun görüntüleneceği sistemlerde, BCD veya 3-fazla BCD formatındaki sayılarla çalışmak daha kolay olabilir. Tablo 3-1 de gösterildiği gibi, bu kodlar, 0 dan 9 a kadar olan her onluk basamağı 4-bitlik bir ikilik kelimeyle temsil eder. BCD kelimeler, 0 ile 9 arasındaki sayıların ikilik karşılıklarıyla aynıdır. Bunlara karşılık gelen 3-fazla BCD kelimeleri de, standart BCD kelimelerine 3 eklenerek oluşturulur.

BCD nin basamak kelimeleri 9 dan büyük olamaz. Bu nedenle, bir BCD toplamın sonucu, 1001 den, yani 9 dan, büyükse bir elde üretilmelidir. Şekil 7-4, BCD toplamayla ilgili üç örnek göstermektedir. Şekil 7-4 a daki birinci toplama oldukça kolaydır; çünkü toplam basamakları 9 dan azdır. Bu sonuç, standart ikilikte elde edilecek sonuçla aynıdır. Şekil 7-4 b de gösterildiği gibi, BCD 7 yi BCD 5 ile toplamak 1100 üretir. Bu, doğru ikilik sonuç 12 dir, fakat nizami olmayan bir BCD kodudur. Sonucu BCD formatına çevrimek için bir düzeltme faktörü, 6, eklenir. 6 ekleme sonucunda 10010 elde edilir ki bu da 12 nin nizami BCD kodudur. Şekil 7.4 *c*, düzeltme faktörünün eklenmesi gereken bir başka durumu göstermektedir. 9 ile 8 in ilk toplaması 10001 verir. Sağdaki dört basamak 9 dan az olmasına rağmen bu hatalı BCD sonucudur; çünkü toplama bir elde üretmiştir. Düzeltme faktörü 6 yı eklemekle 17 nin karşılığı olan doğru BCD sonuç 10111 elde edilir.

```
                BCD
  35        0011 0101
 +23       +0010 0011
 ---       ---------
  58        0101 1000
```

(*a*)

```
                BCD
   7            0111
 + 5          + 0101
 ---          ------
  12            1100   hata BCD
              +  110   ekle 6
              ------
              10010   doğru BCD 12
```

(*b*)

```
                BCD
   9            1001
 + 8          + 1000
 ---          ------
  17           10001   hata BCD
                 110   ekle 6
              ------
              10111   doğru BCD 12
```

(*c*)

Şekil 7.4 *BCD* toplama örnekleri

Özetlersek, BCD toplamada, eğer en küçük değerlikli 4 bitteki sonuç 9 dan büyükse veya ilk toplam bir elde üretirse, sonuca bir düzeltme faktörü 6 eklenmelidir. Bu düzeltmeye bazen *"onluğa ayarlama işlemi" (decimal adjust operation)* denir.

3-fazla BCD toplamı Şekil 7-5 te gösterildiği gibi yapılır. Şekil 7-5 a daki örnekte bir 3-fazla sayı artı bir 3-fazla sayı sonuçta bir 6-fazla sayı vermektedir. Doğru 3-fazla sonucu elde etmek için bir 3 çıkarılarak düzeltme yapılmalıdır. Şekil 7.5 *b* de gösterildiği gibi toplam sonucunda beşinci basamakta bir elde oluşursa, doğru sonucu elde etmek için 3 eklenmelidir.


```
   5        1000   3-fazla
 +2         0101   3-fazla
 ---      ------
   7        1101   6-fazla
          -  11   8-fazla
          ------
           1010   7₁₀ için 3 - fazla
```

(*a*)

```
   7        1010 }
 + 5      +  1000 } 3-fazla
 ---      -------
  12        1 0010   Hatalı
          + 11  11   Elde olduğundan 3 ekle
          ---------
          100 0101   12₁₀ için doğru 3 - fazla
```

(*b*)

Şekil 7.5 Üç-fazla toplama örnekleri

SEKİZLİK VE ONALTILIK TOPLAMA

Bilgisayar veya mikroişlemcilerle çalışan insanlar, ROM adresleri gibi uzun ikilik sayıları kısa şekilde temsil etmek için çoğunlukla sekizlik veya onaltılık sistemi kullanırlar. Bu nedenle, sekizlik ve onaltılık sayıları da toplayabilmek faydalı olacaktır.

Şekil 7.6, 47 ve 36 sekizlik sayılarını toplamanın iki yolunu göstermektedir. Birinci yol, iki sayıyı da ikilik denklerine çevirmektir. Her sekizlik basamağın, üç ikilik basamağı temsil ettiğini hatırlayın. Bu ikilik sayılar, Şekil 7-2 *a* daki ikilik toplama kurallarını kullanarak toplanır. Sonra elde edilen ikilik toplam sekizliğe çevrilir.

```
                                  Elde
                                    ↓
 47₈          100 111            ¹47₈
+36₈        + 011 110          + 36₈
 ---        ---------          -----
            1 000 101          8₁₀ 13₁₀
            1  0  5₈           1 0 5₈
```

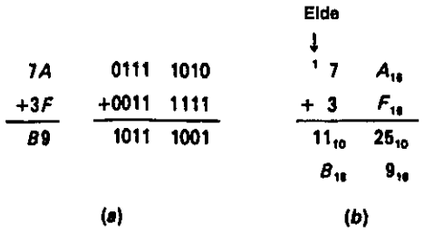
(*a*) (*b*)

Şekil 7.6 Sekizlik toplama örnekleri

İkinci yöntem doğrudan sekizlik şekille çalışır: 7 ve 6 eklenince 13 elde edilir ki bu da 5 kalanla bir sonraki basamağa bir elde demektir. 5 yazılır ve elde bir sonraki basamak sütununa eklenir. Sonra 4 artı 3 artı elde toplanarak 8 bulunur ve bu da 0 kalan ve bir elde verir. 0 yazılır ve elde bir sonraki basamak sütununa eklenir. Bu, onluk toplamada kullandığınız süreçle aynıdır, fakat, 10 yerine, toplamın her 8 veya daha büyük oluşunda bir elde üretilmektedir.

Şekil 7.7 de gösterildiği gibi, iki onaltılık sayıyı toplamak için aynı yaklaşımlar kullanılabilir. İkiliğe çevirirken her onaltılık (hex) basamağın, dört ikilik basamağı temsil ettiğini unutmayın. Daha sonra ikilik sayılar toplanır ve sonuç onaltılığa çevrilir.

İkinci yöntem doğrudan onaltılık sayılarla çalışır. Onaltılık toplamada, toplam ne zaman 16 veya daha büyük olursa bir elde üretilir. Hex A nın onluk karşılığı 10, F ninki de 15 tir. İkisinin toplamı 25 verir ve bu da kalan 9 ve bir elde demektir. 9 yazılır ve elde bir sonraki basamak sütununa eklenir. Böylece 7 artı 3 artı bir elde onluk 11 sayısını verir ki bu da hex B dir.



Şekil 7.7 Onaltılık toplama örnekleri

Hangi yöntem daha kolay gelirse ve sürekli doğru cevaplar elde edebiliyorsanız onu kullanabilirsiniz. Çok fazla sekizlik veya onaltılık aritmetik kullanıyorsanız, bir Texas Instruments TI programlayıcı alabilirsiniz. Bu, onluk, sekizlik veya onaltılık aritmetik işlemler için özellikle tasarlanmış bir elektronik hesap makinasıdır.

ÇIKARMA

STANDART İKİLİK

İkilik çıkarma, çok değişik yöntemlerle gerçekleştirilebildiği için biraz karmaşıktır. Anlatılan ilk yönteme standart denir, çünkü standart ikilik toplamaya benzemektedir. Şekil 7.8 a, A eksi B nin doğruluk tablosunu gösterir.

Eğer B, A dan büyükse, bir sonraki önemli basamaktan bir ödünç almasının etkisini de içeren 'tam' çıkarmanın doğruluk tablosu Şekil 7.8 *b* de gösterilmiştir. Şekil 7.8 c, bu kuralları kullanan bir ikilik çıkarma örneğini göstermektedir.

GİRİŞLER		ÇIKIŞLAR	
A	*B*	*D*	*B*
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Fark = $A \oplus B$

Ödünç = $\overline{A} \cdot B$

(a)

GİRİŞLER			ÇIKIŞLAR	
A	*B*	$B_{GİRİŞ}$	*D*	$B_{ÇIKIŞ}$
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Fark = $A \oplus B \oplus B_{GİRİŞ}$

Ödünç = $\overline{A \cdot B + (\overline{A \oplus B}) \cdot B_{GİRİŞ}}$

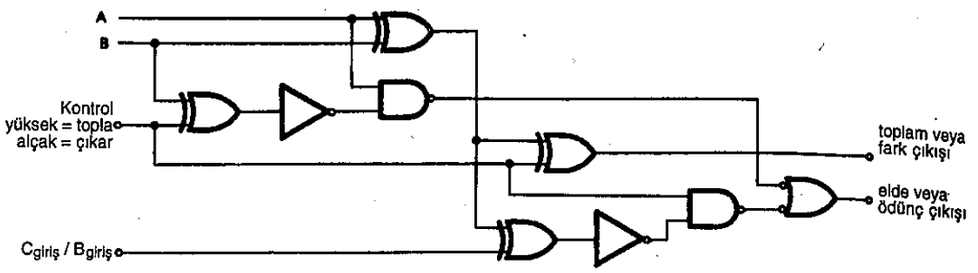
(b)

12	1100
− 2	− 0010
10	1010

(c)

Şekil 7.8 İkilik çıkarma. *(a)* Yarım çıkarıcının doğruluk tablosu. *(b)* Tam çıkarıcının doğruluk tablosu. *(c)* Örnek.

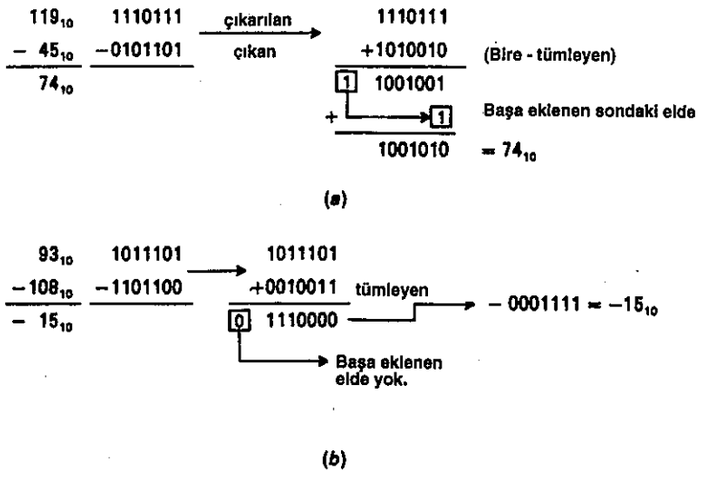
Çıkarmadaki fark çıkışının mantık ifadesi $D = A \oplus B \oplus B_{GİRİŞ}$ ve ödünç-çıkışı $B_{ÇIKIŞ}$ ın ifadesi $\overline{A} . B + (\overline{A \oplus B}) . B_{GİRİŞ}$ tir. Bu ifadeler, geçen kısımdaki tam toplayıcının ifadelerine çok benzediklerinden, iki ikilik basamağı toplayacak veya çıkaracak bir devre kurmak oldukça kolaydır. Şekil 7.9 böyle bir devreyi göstermektedir. Bir devre-izleme alıştırması olarak, seçme girişi yüksek ise bu devrenin, elde girişi ile iki girişi topladığını, seçme girişi alçak ise ödünçle birlikte A dan B yi çıkardığını gösterin.



Şekil 7.9 İki 1-bitlik sayıyı, komuta göre toplayan veya çıkaran devre.

BİRE-TÜMLEYEN ÇIKARMA

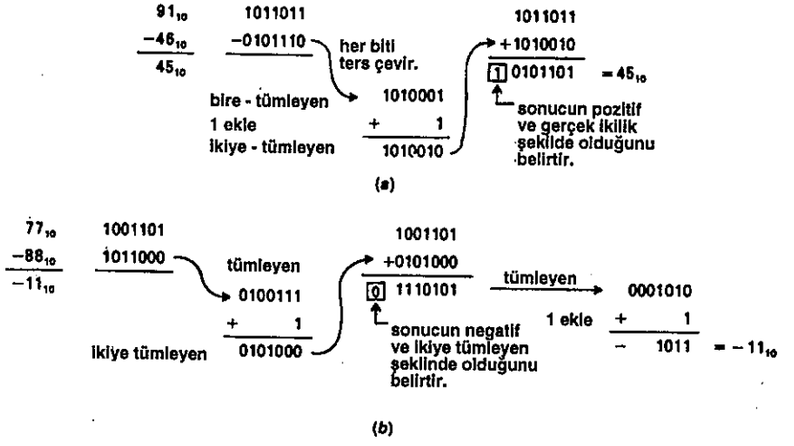
Diğer bir çıkarma yöntemi bire-tümleyen yöntemidir. Bir ikilik sayının bire-tümleyeni, basitçe sayının her bitini tersine çevirerek bulunur. İki sayıyı çıkarmak için, üstteki sayıya (çıkarılan), alttaki sayının (çıkan) bire-tümleyeni eklenir. Şekil 7.10 a bu işlemi göstermektedir. Tümleyeni bulmadan önce alt ve üstteki sayıların basamak sayısının aynı olması gerektiğine dikkat edin. Tümleyeni bulduktan sonra toplama yapılır. Eğer en büyük değerlikli basamak konumundan bir elde 1 üretilirse, doğru cevabı elde etmek üzere bu 1, buradan alınarak en küçük değerlikli basamakla toplanır. Pozitif bir sonuç için bu sonraki eldeyi başa alma her zaman gerekir. Eğer kutuda bir 1 görülmezse (yani en büyük değerlikli basamaktan bir elde üretilmezse), bu takdirde alttaki sayının üsttekinden daha büyük olduğunu anlarsınız. Kutudaki bir 0 ayrıca cevabın tümlenmiş şekilde olduğunu belirtir. Daha kolay tanınır bir cevabı bulmak için, her biti ters çevirin ve sayının önüne bir eksi işareti koyun. Şekil 7.10 b bunun bir örneğini göstermektedir.



Şekil 7.10 Birin-tümleyeni çıkarma örnekleri.

İKİYE - TÜMLEYEN ÇIKARMA

Bir ikilik sayının ikiye-tümleyeni, sayıdaki her biti ters çevirerek ve en küçük değerlikli basamağa bir 1 ekleyerek oluşturulur. İki ikilik sayıyı birbirinden çıkarmak için, alttaki sayının (çıkan) ikiye-tümleyenini, üstteki sayı (çıkarılan) ile toplayabilirsiniz. Şekil 7.11a bu işlemi göstermektedir. Alttaki sayının ikiye-tümleyenini bulmak için, sayının her bitinin tersini alır ve 1 eklersiniz. Sonra, bu sayı üstteki sayı ile toplanır. Sondaki elde başa eklenmez, çünkü aslında bu ikiye-tümleyen alınırken yapılmıştır. En büyük değerlikli basamaktan üretilen bir elde, yani kutudaki 1, sonucun pozitif olduğunu belirtir. Eğer kutuda bir 0 gözükürse, bu, sonucun negatif olduğunu ve ikiye-tümleyen şeklinde gösterildiğini belirtir. Şekil 7.11b bunun bir örneğini göstermektedir. İkiye tümleyen şeklinden sayıyı elde etmek için, her biti ters çevirin ve sonuca 1 ekleyin. Bu yöntem biraz garip gelebilir, fakat bir bilgisayar veya mikroişlemcide kullanılması kolaydır, çünkü sadece basit ters çevirme, bir artırma ve toplama işlemlerini gerektirir.



Şekil 7.11 İkiye-tümleyen çıkarma örnekleri.

Örneğin 8-bit genişliğinde bir RAM da hem pozitif hem de negatif sayılar saklanacaksa, sayıları pozitif veya negatif diye tanıyabilmek için bir yol saptanmalıdır. Genel yöntem en büyük değerlikli biti işaret biti olarak kullanmaktır. Pozitif sayılar, 0 işaret bitiyle gerçek ikilik şekilleriyle temsil edilirler. Negatif sayılar, 1 işaret bitiyle ikiye-tümleyen şekilleriyle temsil edilirler. Şekil 7.12, işaret bitiyle gösterilen sayıların örneklerini, 8-bit genişliğinde bir RAM da saklanabileceleri şekliyle göstermektedir.

	İşaret biti ↓	
+ 7	0	0000111
+ 46	0	0101110
+105	0	1101001
− 12	1	1110100
− 54	1	1001010
−117	1	0001011
− 46	1	1010010

İşaret ve büyüklüğün ikiye - tümleyeni

Şekil 7.12 İşaret bitiyle gösterilen pozitif ve negatif sayılar.

Bir negatif sayı için, İkiye-tümleyen işaret - ve - büyüklük şeklini elde etmek için:

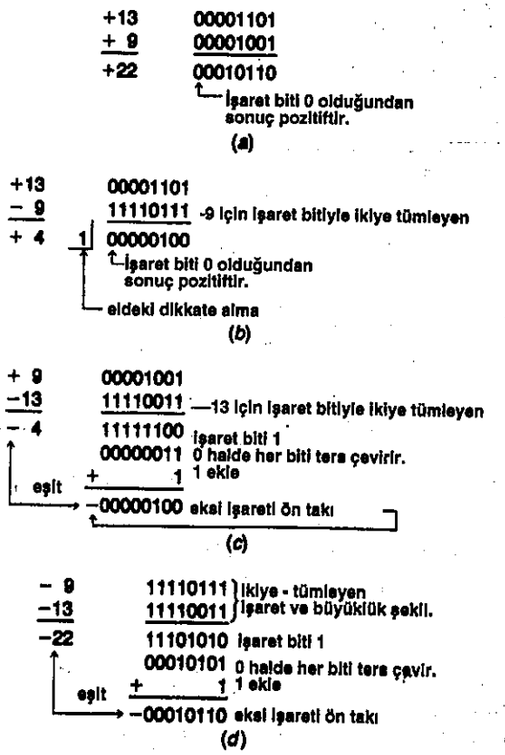
1. İşaret biti de dahil, sayının pozitif ikilik şeklini yazın.
2. İşaret biti de dahil, her biti ters çevirin.
3. Sonuca 1 ekleyin.

Örneğin, - 46 onluk sayısının, 8-bitlik ikiye-tümleyen işaret - ve - büyüklük şeklini elde etmek için, önce 8-bitlik +46 sayısını, yani 00101110 yazın. İkinci olarak, her biti ters çevirerek 11010001 i elde edin. Üçüncü olarak, buna 1 ekleyerek 11010010 u bulun. Bu, şekil 7.12 de de gösterildiği gibi, - 46 sayısının doğru 8-bitlik ikiye-tümleyen gösterilişidir.

8 bitin hâlâ 256 değeri temsil ettiğine dikkat edin, fakat 0 ile 255 arasını temsil etmek yerine, şimdi - 128 ile + 127 arasındaki sayıları temsil etmektedir. 0_{10} un işaret biti 0 dır, bu nedenle, bu sistemde, 0, pozitif bir sayı olarak kabul edilir. Eğer sayılardaki desenleri seviyorsanız, bu yöntemin, 128 ile 255 arasındaki normal kodları, - 128 ile - 1 arasını temsil etmek üzere aşağı doğru kaydırdığını fark edebilirsiniz.

Bu işaret - ve - büyüklük şekli herhangi bir sayıda biti olan sayılar içinde kullanılabilir. Onaltı bit - 32768 ile + 32767 arasındaki sayıları temsil edebilir.

Şekil 7.13, bu tür işaretli ikilik sayıların toplanmasına ait bazı örnekler göstermektedir. İşaret bitleri de diğer bitler gibi toplanmaktadır. Şekil 7.13a, iki pozitif sayının toplamının sonucunu göstermektedir. Sonucun işaret biti 0



Şekil 7.13 İşaret bitli sayılarla yapılan işlemlere örnekler.

dır, o halde sonuç pozitiftir. İkinci örnekte (Şekil 7.13*b*) + 13 e - 9 eklenmiş, yani, gerçekte, 13 ten 9 çıkarılmıştır. 0 işaret bitiyle belirtildiği gibi, bunun sonucu, 4, pozitif ve gerçek ikilik şekildedir.

Şekil 7.13*c* - 13 e daha küçük pozitif bir sayı, 9 un eklenmesinin sonucunu göstermektedir. Sonucun işaret biti 1 dir. Bu, sonucun negatif ve büyüklüğün ikiye-tümleyen şeklinde olduğunu belirtir. İkiye-tümleyen şeklindeki bir sonucu, gerçek ikilik şeklinde işaretli bir sayıya çevirmek için:

1. Her biti ters çevirin.

2. 1 ekleyin.

3. Sonucun negatif olduğunu göstermek için önüne bir eksi işareti koyun.

Şekil 7-13d'deki son örnek, iki negatif sayıyı toplamanın sonucunu göstermektedir. Sonucun işaret biti 1 dir ve sonuç, negatiftir ve ikiye-tümleyen şeklindedir. Yine her biti ters çevirip, 1 ekleyip, önüne de bir eksi işareti koyarak, sonuç daha kolay anlaşılır hale getirilebilir.

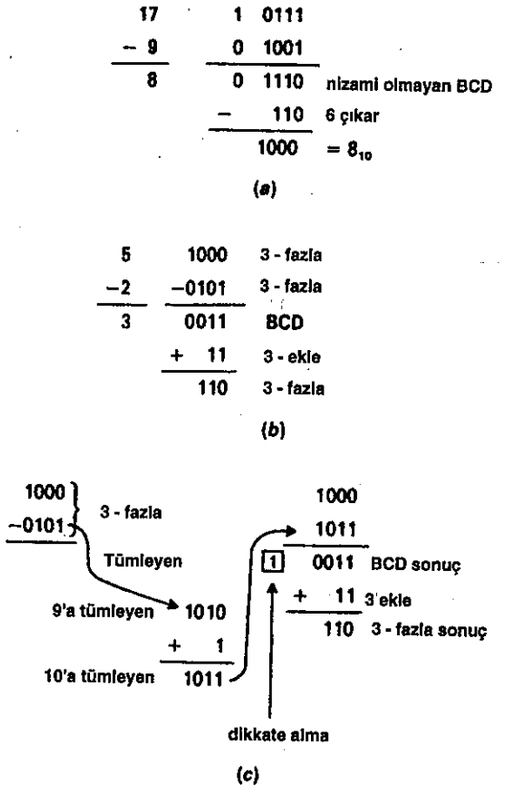
ÖZET Bu örneklerde gösterildiği gibi, ikilik sayıların çıkarılması, üstteki sayıya alttaki sayının ikiye-tümleyen işaret - ve - büyüklük şeklini ekleyerek gerçekleştirilebilir. Sonucun işaret biti 0 sa, sonuç pozitiftir ve büyüklük gerçek ikilik şeklindedir. Eğer sonucun işaret biti 1 ise, sonuç negatiftir ve büyüklük ikiye-tümleyen şeklindedir.

DİĞER SAYI SİSTEMLERİNDE ÇIKARMA

BCD ÇIKARMA Çıkarma üzerine anlattıklarımızı toparlamak üzere birçok çeşite ait örnekler verilmiştir. Şekil 7.14a 17 den 9 un BCD çıkarmasını göstermektedir. İlk sonuç bir nizami olmayan BCD sayıdır. BCD çıkarmada bu ne zaman olursa, nizami BCD sonucu elde etmek için - 6 lık bir düzeltme faktörü eklenmelidir. Bu onluğa ayarlama işlemi, BCD sayıları toplarken yaptığınız düzeltmenin tam tersidir.

3-FAZLA ÇIKARMA 3-fazla çıkarma Şekil 7.14b ve c de gösterilmiştir. Çıkarılan iki sayı da 3-fazla şeklinde olduğu için, fark, standart BCD şeklinde çıkar. Sonucu 3-fazla şeklinde bulmak için ilk sonuca 3 eklenmelidir. 3-fazla sayıların çıkarması, ikilik çıkarmada ikiye-tümleyeni eklediğiniz gibi, üstteki sayıya, alttaki sayının ana-tümleyenini ekleyerek de yapılabilir. 3-fazla çıkarma bazı hesap makinalarında standart BCD nin yerine kullanılmaktadır, çünkü ana-tümleyen daha kolay oluşturulmaktadır. Ana-tümleyeni oluşturmak için, Şekil 7.14c de gösterildiği gibi, 3-fazla sayı ters çevrilir ve 1 artırılır. Fark yine standart BCD olarak çıkar, bu nedenle 3-fazla şekline sokmak için 3 eklenmelidir.

SEKİZLİK ÇIKARMA Sekizlik çıkarma Şekil 7.15 te gösterilmiştir. Üstteki sayının en küçük değerlikli basamağı, alttaki sayının en küçük değerlikli basamağından daha küçük olduğundan, bir ödünç alma gerekir. Sekizlik çıkarmada, bir sonraki basamaktan sekiz ödünç alınır ve bir sonraki basamağa eklenir. Sonra alttaki basamak çıkarılır ve kalan yazılır. Bu sürece, bütün basamakların



Şekil 7.14 *BCD* **ve 3-fazla** *BCD* **çıkarma örnekleri.** *(a) BCD. (b)* **3-fazla** *BCD. (c)* **3-fazla** *BCD.*

çıkarması tamamlanıncaya kadar devam edilir. Eğer "8 leri ödünç alma" sizi rahatsız ediyorsa, sayıları onluğa çevirip çıkarır ve sonucu yeniden sekizliğe çevirebilirsiniz.

ONALTILIK ÇIKARMA Onaltılık çıkarma, sekizlik çıkarmaya benzer, sadece bir ödünç alma gerektiğinde, bir sonraki büyük değerlikli basamaktan alınan ödüncün değeri 16 dır. Şekil 7.16 bunu göstermektedir. Eğer kafanızda onluğa kısmi çevrimler yapıyorsanız, örneği izlemeniz kolaylaşabilir.

$$\begin{array}{rr} 34_8 & 28_{10} \\ -17_8 & -15_{10} \\ \hline 15_8 & 13_{10} \end{array}$$

Şekil 7.15 Sekizlik çıkarma örnekleri.

Örneğin 7 artı ödünç alınan 16, 23 yapar. Bundan B yi, yani 11, çıkarmakla 12, yani onaltılık C, kalır. Ödünç almadan sonra kalan 6 dan 3 çıkarmakla 3 kalır, o halde sonuç $3C_{16}$ dır.

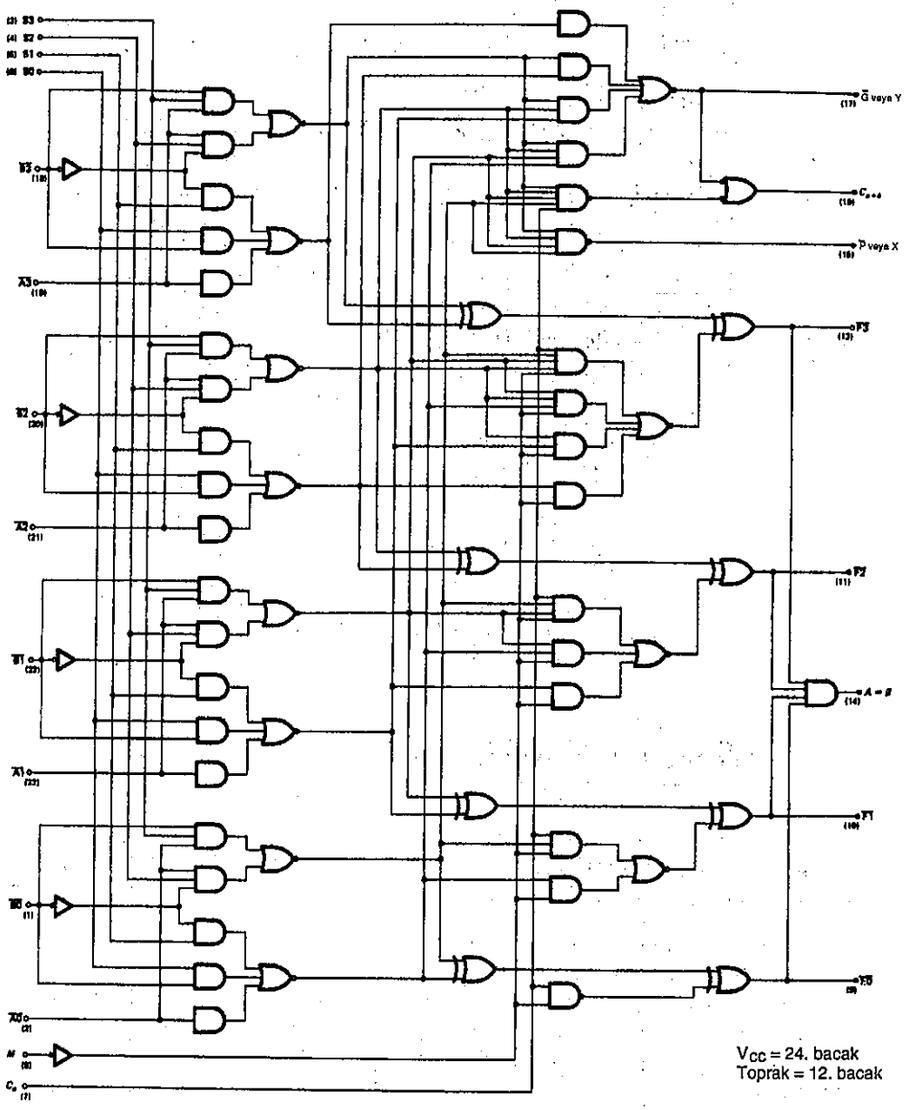
$$\begin{array}{rcr} 77_{16} & = & 119_{10} \\ -3B_{16} & = & -\ 59_{10} \\ \hline 3C_{16} & & 60_{10} \end{array}$$

Şekil 7.16 Onaltılık çıkarma örnekleri.

ARİTMETİK MANTIK BİRİMİ (ARITHMETIC LOGIC UNIT : ALU)

Bu kitabın geri kalan kısımlarında, birçok sistemde ikilik aritmetikle karşılaşılmaktadır. Toplama ve çıkarmanın yanı sıra diğer mantık fonksiyonlarını da gerçekleştiren bir eleman *aritmetik mantık birimi*, yani *ALU* dur.

Şekil 7.9, iki tek-bitlik sayıyı, elde veya ödünçlerle birlikte, toplayabilen veya çıkarabilen basit kapılardan yapılmış bir devreyi göstermektedir. Yapılacak işlem, seçme girişindeki mantık (1) bir seviyesi veya mantık sıfır seviyesi ile belirlenir. Bu devrenin daha genişletilmişi, TTL de 74181 aritmetik mantık birimi olarak bulunmaktadır. 74181 in mantık şeması Şekil 7.17 de gösterilmiştir. Bu aygıt, dört seçme girişi ve bir mod girişi ile, iki 4-bitlik, giriş kelimesi üzerinde mantık veya aritmetik işlemler yapmak üzere programlanabilir. Şekil 7.18, mod girişi M yüksek olduğunda, üretilen 16 mantık fonksiyonunun bir tablosunu göstermektedir. Bu tabloda A, 4 bitlik ikilik kelime $A_3\,A_2\,A_1\,A_0$, B de ikilik kelime $B_3\,B_2\,B_1\,B_0$ ı göstermektedir.



Şekil 7.17 DM 74181 aritmetik mantık biriminin mantık şeması (National Semiconductor).

Mantık Fonksiyonları Tablosu

Fonksiyon Seçme				Çıkış Fonksiyonu	
S3	**S2**	**S1**	**S0**	Negatif Mantık	Pozitif Mantık
L	L	L	L	$F = \bar{A}$	$F = \bar{A}$
L	L	L	H	$F = \overline{AB}$	$F = \overline{A + B}$
L	L	H	L	$F = \bar{A} + B$	$F = \bar{A}B$
L	L	H	H	$F =$ Mantık 1	$F =$ Mantık 0
L	H	L	L	$F = \overline{A + B}$	$F = \overline{AB}$
L	H	L	H	$F = \bar{B}$	$F = \bar{B}$
L	H	H	L	$F = \overline{A \oplus B}$	$F = A \oplus B$
L	H	H	H	$F = A + \bar{B}$	$F = A\bar{B}$
H	L	L	L	$F = \bar{A}B$	$F = \bar{A} + B$
H	L	L	H	$F = A \oplus B$	$F = \overline{A \oplus B}$
H	L	H	L	$F = B$	$F = B$
H	L	H	H	$F = A + B$	$F = AB$
H	H	L	L	$F =$ Mantık 0	$F =$ Mantık 1
H	H	L	H	$F = A\bar{B}$	$F = A + \bar{B}$
H	H	H	L	$F = AB$	$F = A + B$
H	H	H	H	$F = A$	$F = A$

Mod kontrol *(M)* yüksekse : *Cn* farketmez
Pozitif mantık için: Mantık 1 = Yüksek gerilim
Mantık 0: Alçak gerilim
Negatif mantık için: Mantık 1 = Alçak gerilim
Mantık 0 = Yüksek gerilim

Şekil 7.18 74181 ALU nun mantık fonksiyonlarının tablosu (National Semiconductor.)

Şekil 7.19, 74181 tarafından iki 4-bitlik kelime üzerinde gerçekleştirilen mantık fonksiyonlarına ait bazı örnekleri göstermektedir. İlk örnekte, seçme girişleri $S_3 S_2 S_1 S_0$ a *HHHL* bağlanmış ve *M* girişi yükseğe set edilmiştir. Bu, 74181 i A + B çıkış ifadesini vermek üzere programlar. *A* ifadesinin her biti, *B* ifadesinin karşılık gelen bitiyle VEYA lanır. VEYA lamanın sonucu karşılık gelen *F* çıkışında görülür. Çıkış ifadesi, *A* veya *B* ifadesinde 1 olan bit konumlarının herbirinde bir 1 içerecektir. *M* yüksek olduğunda, elde girişi C_n dikkate alınmaz.

Seçme girişleri HLHH' ye değiştirilir ve mod girişi M yüksek tutulursa, çıkış kelimesi F, A ifadesi ile B ifadesinin karşılık gelen bitlerinin VE lenmesi sonucuna eşit olacaktır. 74181 de, A ve B nin bütün mümkün 16 mantık birleşimi bulunduğundan, bu aygıt, pozitif veya negatif mantık sistemlerinde kullanılabilir. Bütün yapmanız gereken uygun seçme girişlerini seçmenizdir. Örneğin, seçme girişlerindeki LLLH pozitif mantık için $F = \overline{A + B}$ verir ve seçme girişlerindeki LHLL negatif mantıkta $F = \overline{A + B}$ verir. Seçme girişleri için L, topraktır ve H, + 5 V tur.

M = H
S3 S2 S1 S0 = H H H L

	BIT3	BIT2	BIT1	BIT0
A	1	1	0	0
B	1	0	1	0
F	1	1	1	0

$F = A + B$

(a)

M = H
S3 S2 S1 S0 = H L H H

	BIT3	BIT2	BIT1	BIT0
A	1	1	0	0
B	1	0	1	0
F	1	0	0	0

$F = A \cdot B$

(b)

Şekil 7.19 74181 i programlama örnekleri. *(a)* A + B. *(b)* A. B . (National Semiconductor.)

Negatif ve pozitif mantıkta karşılık gelen fonksiyonlarla ilgili olarak hafızanızı tazelemek için Şekil 7-18 i de kullanabilirsiniz. Örneğin, pozitif mantıkta bir VEYA DEĞİL fonksiyonu, negatif mantıkta bir VE DEĞİL fonksiyonudur.

Aritmetik fonksiyonları gerçekleştirmek için, 74181 in *M* girişi alçak yapılır. Bu, C_n i ve iç eldeleri yetkilendirir. Şekil 7.20, hem pozitif hem de negatif mantık için 16 seçme girişi birleşiminin oluşturduğu aritmetik fonksiyonları göstermektedir, ancak burada pozitif mantık fonksiyonları üzerinde duracağız.

Seçme girişleri HLLH olduğunda A ile B nin basit toplaması yapılır. Elde girişi C_n, aktif alçaktır; böylece bu giriş alçak olursa, A ve B toplamına bir de elde eklenecektir. Eğer toplam bir elde çıkışı üretirse, C_{n+4} alçak olacaktır. 74181, C_{n+4} çıkışı verilmeden önce toplamın hesaplanmasını gerektirmeyen bir "eldeyi önceden bulma" look ahead carry yöntemi kullanır. 4 bitten büyük kelimeleri alçak-hızda toplamak için, C_{n+4} çıkışı, bir başka 74181 in C_n girişine bağlanabilir. Daha büyük kelimeleri yüksek-hızda toplamak için, *'elde yayma' (carry propagate)* ve *'elde üretme' (carry generate)* çıkışları bir başka parça, 74182 eldeyi önceden bulma üreteci, ile birlikte kullanılır. 74182, dört taneye kadar 74181 ALU nun eldelerini önceden bulmak için 'elde yayma' ve 'elde üretme'yi kullanır. Bu, uzun kelimelerin daha büyük değerlikli bitlerini toplayan ALU ların, baştaki ALU ların eldelerinin dalgacık halinde yayılıp gelmesini beklemek zorunda olmadığı anlamına gelir.

Aritmetik İşlemlerin Tablosu

Fonksiyon Seçme				Çıkış Fonksiyonları	
S3	S2	S1	S0	Alçak seviyeler aktif	Yüksek Seviyeler Aktif
L	L	L	L	$F = A$ eksi 1	$F = A$
L	L	L	H	$F = AB$ eksi 1	$F = A + B$
L	L	H	L	$F = A\overline{B}$ eksi 1	$F = A + \overline{B}$
L	L	H	H	$F =$ eksi 1 (ikiye tümleyen)	$F =$ eksi 1 (ikiye tümleyen)
L	H	L	L	$F =$ A artı $[A + \overline{B}]$	$F =$ A artı AB
L	H	L	H	$F =$ A artı $[A + \overline{B}]$	$F = [A + B]$ artı $\overline{AB}$
L	H	H	L	$F =$ A eksi 1 B eksi 1	$F =$ A eksi B eksi 1
L	H	H	H	$F = A + \overline{B}$	$F = A\overline{B}$ eksi 1
H	L	L	L	$F =$ A artı $[A + B]$	$F =$ A artı B
H	L	L	H	$F =$ A artı B	$F =$ A artı B
H	L	H	L	$F = \overline{AB}$ artı $[A + B]$	$F = [A + \overline{B}]$ artı AB
H	L	H	H	$F = A + B$	$F =$ AB eksi 1
H	H	L	L	$F =$ A artı A +	$F = A$ artı A +
H	H	L	H	$F =$ AB artı A	$F = [A + B]$ artı A
H	H	H	L	$F = A\overline{B}$ A artı	$F = [A + \overline{B}]$ artı A
H	H	H	H	$F = A$	$F =$ A eksi 1

Negatif mantık için mode kontrol *(M)* ve *Cn* alçak; pozitif mantık için *M* alçak ve *Cn* yüksek. + her bit, bir sonraki daha büyük değerlikli konuma kaydırılır.

Şekil 7.20 74181 ALU nun aritmetik işlemlerinin tablosu (National Semiconductor).

Eğer bütün seçme girişleri alçak, *M* alçak ve C_n yüksekse, 74181 ALU, *A* çıkışını verir. Eğer C_n alçak yapılırsa, çıkış *A* + 1 i vermek üzere 1 artırılacaktır. Seçme girişlerinin hepsi yüksek, *M* alçak ve C_n yüksek olduğunda, çıkış, *A* nın 1 azaltılmışı, yani *A* - 1, olacaktır. Seçme girişleri HHLL olarak programlanarak *A* + *A*, yani 2*A* çıkışı üretilebilir. Önceki bölümde bahsedildiği gibi, bir ikilik sayıyı 2 ile çarpmak, her basamağı bir sola kaydırmakla aynıdır. Daha sonraki bir kısımda, ikilik çarpmada kaydırmanın nasıl kullanıldığını göreceksiniz.

Çıkarma, 74181 in diğer bir önemli işlemidir. 74181 çıkarmayı, kendi içinde *B* nin bire-tümleyenini üretip bunu *A* ile toplayarak yapar. LHHL seçme girişleri, *A-B-1* çıkışını verir. Doğru sonucu bulmak için, sondaki eldeyi başa ekleme işlemi yapılmalıdır. Bu, C_n yi alçak yaparak gerçekleştirilebilir. Bir çıkarma sırasında C_{n+4} çıkışı ödünçü temsil eder.

Çıkarma işlemi, bir ikilik *A* ifadesinin, bir ikilik *B* ifadesinden daha büyük, daha küçük veya ikisinin eşit olduğunu belirlemek için kullanılabilir. Seçme girişleri LHHL, *M* alçak ve C_n alçak yapılır. Eğer *A* = *B* ise, 14. bacaktaki *A* = *B* çıkışı yüksek olacaktır. Bu çıkış açık-kollektörlüdür, böylece daha büyük ifadeleri karşılaştırmak için birçok *A* = *B* çıkışı birbirine bağlanabilir. Eğer bütün *A* = *B* çıkışları yüksek ise birleşik çıkış yüksek olacaktır. Eğer, bir 74181 de, *A* ifadesi, *B* ifadesinden büyük veya *B* ifadesine eşitse, C_{n+4} çıkışı yüksek olacaktır (ödünç alma olmadığını gösterir). Eğer B ifadesi A ifadesinden büyükse, bu takdirde C_{n+4} çıkışı alçak olacaktır (ödünç almayı gösterir).

Şekil 7-17' deki 74181' in mantık şemasında, bir tane kurup deneyebilmeniz için, bacak numaraları da verilmiştir. Buradaki gerçekten önemli noktalar, ALU nun, iki ikilik ifade üzerinde birçok mantık ve aritmetik işlemler yapabildiği ve yapılan işlemin, seçme ve mod girişlerine programlanan ikilik kodlu komutla belirlendiğidir. Bir sonraki bölümde anlatıldığı gibi, bir ALU, bir mikroişlemcinin 'kalbi'dir.

İKİLİK ÇARPMA VE BÖLME

ÇARPMA

İkilik çarpma yapmanın değişik yöntemleri vardır. Şekil 7-21, onluk sayıları çarpmayı öğrendiğiniz yolla aynı olduğu için kalem yöntemi denilebilen bir yöntemi göstermektedir. Üstteki sayı, yani çarpılan, alttaki sayının, yani çarpanın, en küçük değerlikli basamağı ile çarpılır. Kısmi çarpım alta yazılır. Üstteki sayı, çarpanın bir sonraki basamağı ile çarpılır. Elde edilen çarpım, bir basamak sola kaydırılmış olarak daha öncekilerin altına yazılır.

Bütün kısmi çarpımlar toplanarak toplam çarpım bulunur. Çarpma elle yapıldığında, bu yöntem gayet kullanışlıdır, fakat bir bilgisayar için pratik değildir; çünkü gerekli kaydırma tipleri bu yöntemi biraz garip hale getirir.

Bilgisayarlar ve mikroişlemciler için kullanılan bir çarpma yöntemi tekrarlı toplamadır. Örneğin 7 x 55 i yapmak için, bilgisayar sadece 7 tane 55 i top-

```
  11          1011     Çarpılan
 × 9        × 1001     Çarpan
 --         -------
 99          1011  \
              0000   } Kısmi çarpımlar
             0000   /
            1011   /
            -------
            1100011    Çarpım
```

Şekil 7.21 İkilik sayıların kalemle çarpma örnekleri.

lar. Ancak büyük sayılar için, bu yöntem yavaştır. Örneğin, 786 x 253 ü yapmak 252 toplama işlemi gerektirir.

Bilgisayarlar ve mikroişlemciler için daha hızı bir yöntem, topla - ve - sağa kaydır algoritması kullanır. Bir algoritma, bir problemi çözmekte kullanılan bir formül veya yordamdır. Bir problemi çözmede belli bir algoritmanın iyi olduğu ilk başta hemen açık olmayabilir. Topla - ve - kaydır algoritması Şekil 7.22a'da gösterilmiştir. Bu yöntem, ikilik çarpmada kısmi çarpımın, çarpan basamak 1 ise üstteki sayıya, çarpan basamak 0 ise 0 a eşit olmasından yararlanır.

Çarpılacak sayılar iki kaydedicide saklanır. Sonuç, ilk başta sıfıra resetlenen ve akümülatör denen özel bir kaydedicide oluşturulacaktır. Önce, çarpanın en küçük basamağının 0 mı, 1 mi olduğu kontrol edilir. Eğer basamak, örnekte gösterildiği gibi, 1 ise, üstteki sayı, yani çarpılan, akümülatöre eklenir. Bundan sonra, akümülatör 1 basamak sağa kaydırılır, veya 'döndürülür'. Çarpanın bir sonraki basamağı kontrol edilir ve üstteki sayı ya da 0, akümülatörün en büyük değerlikli bitlerine şekilde gösterildiği gibi eklenir. Sonra, akümülatör bir daha döndürülür. İşlemler, çarpanın bütün basamakları test edilip kullanılıncaya kadar devam eder.

Özetlersek; eğer bir çarpan biti 1 ise, çarpılan akümülatöre eklenir; eğer çarpan biti 0 ise, akümülatöre 0 eklenir. Daha sonra akümülatör, 1 bit sağa kaydırılır, yani döndürülür. Bu süreç, bütün çarpan bitleri kullanılıncaya kadar devam eder.


```
  1011 |        B kaydedicisi
 ×1001 |        C kaydedicisi
 ------|
  0000 |        Akümülatör kaydedicisini sıfıra resetle.
  1011 |        İlk çarpan basamağı 1 olduğundan
 ------|        çarpılanı akümülatöre ekle.
  1011 |
   101 | 1      Sonucu sağa kaydır.
  0000 |        Sonraki çarpan basamağı 0'dır,
 ------|-       o halde akümülatör 0 ekle.
  0101 | 1
  0010 | 11     Sonucu sağa kaydır.
  0000 |        Sonraki çarpan basamağı 0'dır,
 ------|--      o halde akümülatör 0 ekle.
  0010 | 11
  0001 | 011    Sonucu sağa kaydır.
  1011 |        Sonraki çarpan basamağı 1'dir,
 ------|---     o halde çarpılanı akümülatöre ekle.
Product 1100 | 011
```

(a)

```
   101 |
 × 011 |
 ------|
   000 |        Akümülatör resetle.
   101 |        Topla
 ------|
   101 |
   010 | 1      Kaydır
   101 |        Topla
 ------|-
   111 | 1
   011 | 11     Kaydır
   000 |        Topla
 ------|--
   011 | 11     Çarpım
```

(b)

Şekil 7.22 İkilik çarpmada topla-ve- sağa-kaydır yönteminin örnekleri.

Bu algoritmaya yakından bakarsanız, kalem yöntemiyle aynı şeyleri yaptığını, sadece kısmi çarpımların elde edilir edilmez toplandığını göreceksiniz. Ayrıca, kalem yönteminde yapıldığı gibi her kısmi çarpımın sola kaydırılması yerine, kısmi çarpımlar toplamı sağa kaydırılmaktadır. Sonuçlar

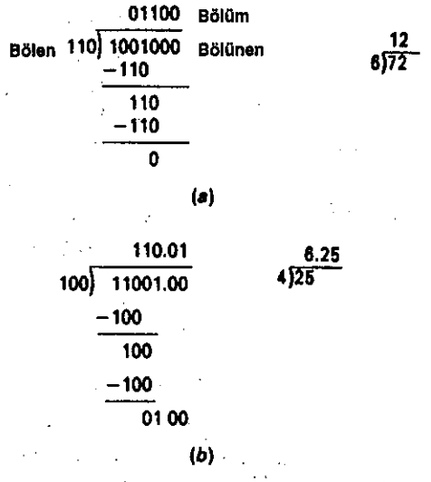
aynıdır, fakat topla-ve-sağa-kaydır algoritmasını bilgisayar ve mikroişlemcilerde kullanmak daha kolaydır. Şekil 7.22*b*, bu yönteme ait başka bir örnek göstermektedir.

4 bitlik 2 sayıyı çarpmak, maksimum 8-bitlik bir çarpım verebilir. 8 bitlik iki sayı, 16-bitlik çarpım verebilir. Çarpımı saklayacak olan kaydedici, beklenilen bit sayısını tutabilecek kadar büyük olmalıdır. Akümülatör yeteri kadar büyük değilse, çarpımın bir kısmını saklamak üzere, bir başka kaydedici kullanılabilir.

Bilgisayar dışında hızlı çarpma için, hazır bir veya iki yongalı çarpıcılar vardır. Birlikte bağlanan SN 74284 ile SN 74285, 4-bitlik iki sayının 8-bitlik çarpımını yaklaşık 40 ns de verecektir. Bu çarpıcılar, daha büyük sayıları çarpmak üzere genişletilebilirler.

İKİLİK BÖLME

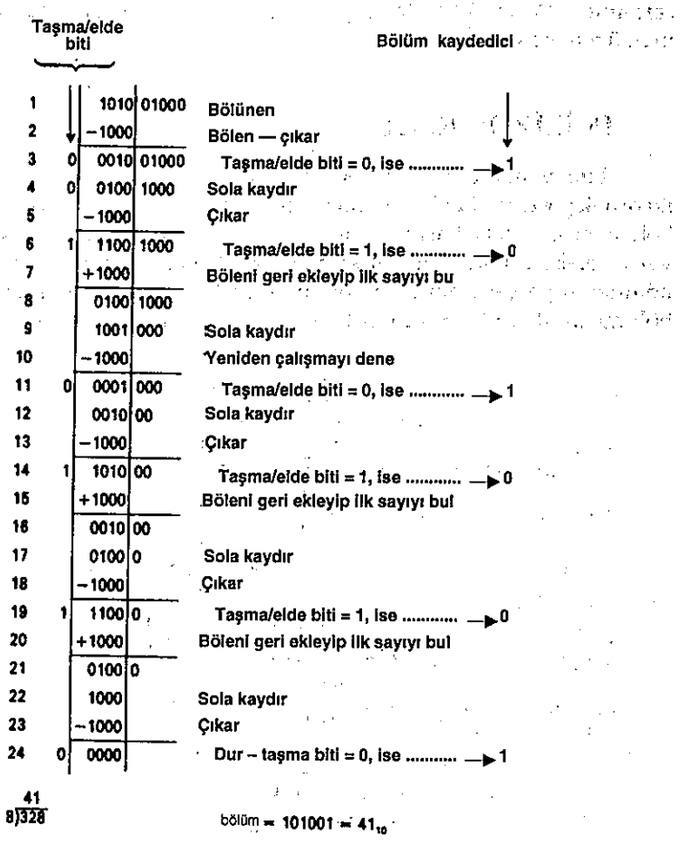
İkilik bölme, çeşitli şekillerde yapılabilir. Şekil 7.23, kalem yöntemine ait iki örnek göstermektedir. Bu, onluk bölmedekiyle aynı süreçtir. Ancak, onluk bölmeden çok daha basittir, çünkü sonucun (bölümün) basamakları sadece 0 veya 1 olabilir. Bölünenin bir kısmı üzerinde bölme yapmaya çalışılır. Bölen, bölünenin parçasından daha büyük olduğu için, bölme mümkün değilse, bölüme bir 0 yazılır. Sonra, bölünenin bir basamağını daha kullanarak yeni-



Şekil 7.23 İkilik sayıları kalemle bölme örnekleri.

den bölme yapmaya çalışılır. Bölme mümkün olursa, bölüme bir 1 girilir. Sonra bölen, bölünenin kullanılan kısmından çıkarılır. Bu sürece standart bölmedeki gibi, bölünenin tamamı kullanılıncaya kadar devam edilir. Şekil 7.23*b* de gösterildiği gibi, kalanı ikilik dengine çevirmek üzere ikilik virgülün sağına 0 lar eklenip bölmeye devam edilebilir.

Bilgisayar ve mikroişlemcilerin kullanması daha kolay olan bir başka bölme yöntemi, tekrarlı çıkarmadır. Bölen, bir ödünç alma oluşuncaya kadar, bölünenden ve her kalandan çıkarılır. Aranılan bölüm, ödünçü üreten çıkarma sayısından 1 eksiktir. Bu yöntem basittir, fakat büyük sayılar için yavaştır.



Şekil 7.24 İkilik bölmede çıkar-ve-sola-kaydır yöntemine ait örnek.

Büyük sayıları daha hızlı bölmek için, bilgisayarlarda bir çıkar-ve-sola-kaydır algoritması kullanılabilir. Şekil 7.24, bu yöntemin bir örneğini göstermektedir. Bu yöntem biraz karmaşık görülebilir, fakat temelde, kalemle yapılan bölme ile aynı süreçtir. Bölen önce bölünenin en büyük basamağından çıkarılır. Eğer çıkarma sonucu oluşan taşma, yani elde, biti 0 ise, çıkarma mümkündür ve Şekil 7.24 ün 3. satırındaki gibi, bölme kaydedicisine bir 1 yazılır. Çıkarmanın sonucu bir basamak sola kaydırılır. 5 ve 6. satırlarda gösterildiği gibi, bölen bu kaymış sonuçtan çıkarılır. Bu sefer bölen çok büyük olduğu için taşma/elde biti 1 olur. Bölüm kaydedicisinin bir sonraki basamağına bir 0 yazılır. Bölen, 7 ve 8. satırlarda görüldüğü gibi, daha önceki kaymış sonucu elde etmek üzere, çıkarma sonucuna eklenir. Bu sayı bir sola kaydırılır ve çıkarma yeniden denenir. Eğer çıkarma bir 0 taşma/elde biti verirse, çıkarma mümkündür ve 11. satırdaki gibi, bölme kaydedicisine bir 1 yazılır. Ne zaman taşma/elde biti 1 olursa, bölen gerisin geriye toplanır, elde edilen sayı bir basamak sola kaydırılır ve çıkarma yeniden denenir. Bu, kalemle bölmede kullandığınız sürecin aynısıdır. Bölme olmadığında, bölüme bir 0 yazarsınız, bölünene bir basamak daha eklersiniz ve yeni sayı ile bir daha denersiniz. Bu süreç, bir 0 taşma/elde biti bulununcaya kadar devam eder. Her 0 taşma/elde biti için bölüm kaydedicisine bir 1 yazılır. Bölünenin tamamı kullanıldığında, bölüm tamamlanmıştır. Bir kalan varsa, ikilik virgülün sağına 0 lar eklenebilir ve kalan ikilik dengine çevrilmek üzere bölmeye devam edilebilir.

Daha sonra, bu bölme algoritmasının ve daha önce anlatılan çarpma algoritmasının bir mikroişlemcide nasıl kullanılacağını göstereceğiz.

SABİT-VİRGÜLLÜ SAYILAR

TAMSAYILARIN SABİT-VİRGÜLLÜ HESAPLARI

Birim kilo fiatı ile ağırlığı çarpıp toplam fiatı gösteren bir terazi gibi çoğu basit mikroişlemci-tabanlı sistemler sabit-virgüllü sayıları kullanır. Bunun anlamı, bütün sayıların, ikilik veya onluk virgülün en küçük değerlikli bitin sağında olan tamsayılar olarak kabul edilmeleri demektir. Örneğin 9,6 kg, bir kaydedicide 10010110, BCD veya 01100000, ikiliği olarak saklanır. 0,29 dolar/kg lık birim fiat 00101001, BCD veya 00011101, ikiliği olarak saklanır. İkilik ağırlıkla ikilik birim fiat çarpıldığında sonuç 101011100000, ikiliği yani 2784, onluğu olarak bulunur. 2,78 dolarlık doğru görüntüyü vermek için tasarımcı onluk virgülün doğru konumunu belirlemek zorundadır.

Sabit virgüllü sistemde, çok büyük veya çok küçük sayıları temsil etmek için, bu sayıların ölçeklenmeleri gerekir. Örneğin 2 260 000 sayısı 226×10^4 olarak gösterilebilir. 226 sayısı yazmaçta 11100010 olarak saklanır. Ölçek çarpanı 10^4 ün kullanıcı tarafından hatırlanması gerekir. Benzer şekilde, 0,000078

sayısı 78×10^{-6} olarak gösterilebilir. 78, kaydedicide 01001110 olarak saklanır, 10^{-6} lık ölçek çarpanı akılda tutulur.

İKİLİK KESİRLERİN SABİT-VİRGÜLLÜ HESAPLARI

Yukarıda anlatılan sabit-virgüllü sistem, bazı makinalarda çarpmada problem çıkarır. Şekil 7.25a, iki 8-bitlik sayının çarpımını göstermektedir. Böyle bir çarpmayı kalemle yaparsanız, sola dört basamak sayarak ikilik virgülü konumlandırmanızda hiç problem olmaz. Ancak, makinanın ikilik virgülü konumlandırmak üzere sayma imkanı yoktur. Hatta, bellekte bir yerde saklamak için sonucun "kesilmesi", yani 8 bite kısaltılması gerekiyorsa, çoğu makinalar basitçe en büyük değerlikli 8 biti alacaktır. Bu durumda, bu bitlere göre ikilik virgülün nerede olduğunu belirlemek imkansız hale gelecektir.

Şekil 7.25b, bu problemi yenmenin bir yolunu göstermektedir. Bütün sayılar, ikilik virgül en büyük değerlikli bitin solunda olacak şekilde 8-bitlik

```
      100110.11        38.75
    × 100001.01      × 33.25
    ----------       --------
      10011011       1228.4375
     100110110
    1001101100000
    ------------
    1010000100.0111

En büyük değerlikli 8 basamak
```

(a)

```
        .10011011      ×2^8
        .10000101      ×2^8
        ---------
         10011011
        100110110
       1001101100000
       -------------
       .0101000010000111   ×2^12
        ↑
        └── Virgül daima burada
```

(b)

Şekil 7.25 Tamsayı olmayan ikilik sayılarla çarpma. *(a)* Kalem yöntemi. *(b)* İkilik kesir yöntemi.

ikilik kesirlere ölçeklenir (çevrilir). Örneğin, 100110,11 sayısı 0,10011011 x 2^6 olarak gösterilir. Bu tür makina 2^6 yı saklayamayacağı için, bu faktör akılda tutulmalıdır.

Çarpma sonucu, her zaman ikilik virgül en büyük değerlikli bitin solunda olmak üzere 16-bitlik bir çarpım verir. Sonuca ait 2 nin kuvveti, çarpılan sayıların ölçek çarpanlarının üsleri toplanarak belirlenir. Virgülün sağındaki en büyük değerlikli 8 bit ile belirlenen üs, her zaman, doğru cevabı verecektir. Bu sistem, ikilik virgülü kesilip atılmayacağı bir yere konumlandırdığı için çalışmaktadır. Hem negatif, hem de pozitif sayıcılarla çalışılabilir, fakat hâlâ üsleri hatırlama ve toplama problemi vardır. Bundan sonraki yöntem bu problemi çözmektedir.

KAYAN-VİRGÜLLÜ SAYILAR

Sabit- virgüllü sistemler, bütün sayıları, tamsayılar veya kesirler olarak temsil ederler. Daha önce anlatılan iki sabit-virgüllü sistem, işlemcinin ikilik veya onluk virgülün doğru yerinin nerede olduğunu izlemesini gerektiriyordu.

Bir kayan-virgüllü sistem, hem tamsayı hem de kesir kısmı olan sayıları temsil edebilir. Bu tür sayılara kayan-virgüllü sayılar veya reel sayılar denir.

İKİLİK KAYAN-VİRGÜLLÜ SAYI GÖSTERİLİŞLERİ

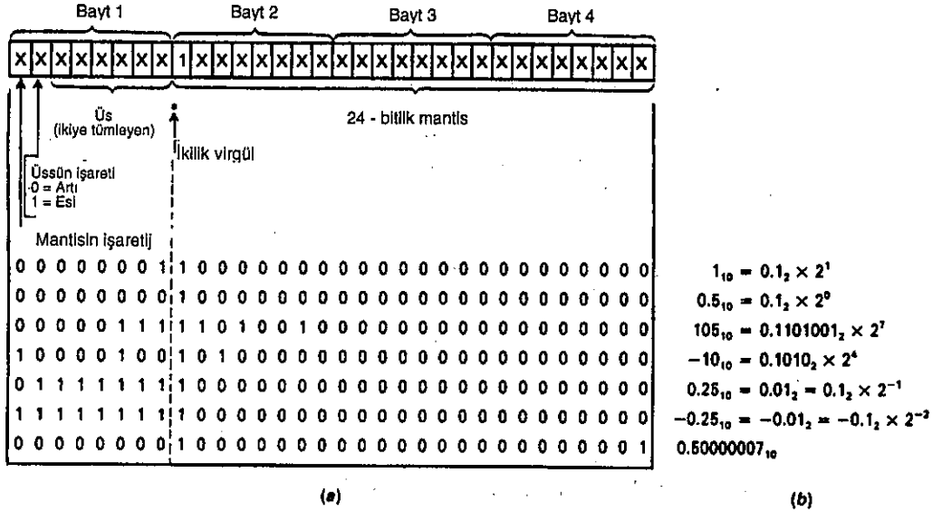
Kayan-virgüllü sayıları temsil etmede kullanılan birçok format vardır. Şekil 7.26a' birini göstermektedir. Her sayıyı temsil etmek için dört bayt, yani 32 bite, gerek vardır. Sayının büyüklüğü, 24-bitlik bir ikilik kesirle gösterilir. Bu kesir, ikilik virgülün sağındaki basamakta 1 olacak şekilde yeniden düzenlenir. Örneğin 1000,11 sayısı yeniden düzenlenince 0,100011 x 2^4 , ve 0,0001011 sayısı da 0,1011 x 2^{-3} olur. 24-bitlik ikilik kesire çoğunlukla mantis denir. İşaret-ve-üs bilgisi dört baytın ilkinde saklanır. En büyük değerlikli bit, sayının pozitif mi yoksa negatif mi olduğunu gösteren işaret bitidir: 0 pozitifi, 1 negatifi belirtir.

Sağdaki bir sonraki bit üssün işaret bitidir ve yine 0 pozitif, 1 negatiftir. Baytta kalan altı bit üssün büyüklüğünü göstermek için kullanılır. Üssün büyüklüğü ikiye-tümleyen şeklindedir. Bu formatın bazı örnekleri Şekil 7.26b de gösterilmiştir. Pozitif bir sayı için sayının ikiye-tümleyeni, standart ikilikle aynıdır. Negatif bir sayı için, sayının ikiye-tümleyeni, her biti ters çevirerek ve sonuca 1 ekleyerek bulunur.

24-bitlik mantisi olan bir formatın doğruluğu 2^{24} te 1 dir, yani yaklaşık yedi onluk basamakta 1 dir. İşaretiyle birlikte 6-bitlik üs, 2^{-63} ile 2^{+63} , yani onlukta yaklaşık 10^{-19} ile 10^{+19} arasında bir aralık verir. Bu bir çok uygulama için yeterli doğruluk ve aralıktır.

KAYAN-VİRGÜLLÜ SAYILARIN ANSI FORTRAN GÖSTERİLİŞİ

Kayan-virgüllü sayıları temsil etmekte kullanılan diğer bir format Amerikan Milli Standartlar Enstitüsünün (ANSI) FORTRAN formatıdır (Şekil 7.27a). Bu



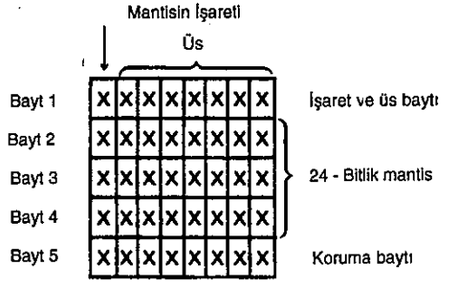
Şekil 7.26 İkilik kayan-virgüllü sayılar.

format, ikilik virgülün en solda olduğu kabul edilen 3-baytlık, yani 24-bitlik, bir mantis kullanır. Ancak, en büyük dört basamaktan herhangi birisi 1 ise, o sayı bu formatta normalize edilmiş kabul edilir. Böylece 0,1000 ; 0,0100; 0,0010 veya 0,0001 şeklinde en büyük değerlikli basamakları olan sayılar normalize edilmiş kabul edilmektedir.

En büyük değerlikli bayt, üssü ve mantisin işaret bitini içerir. Bir 0 işareti pozitif bir mantisi, bir 1 işareti negatif bir mantisi belirtir. Üs ise ikinin kuvveti yerine 16 nın kuvvetini gösterir. Üs, 64-fazla atama gösterimiyle ifade edilir. 64-fazla ataması, 7-bitlik üssü ikiye-tümleyen şeklinde ifade edip 64, yani ikilik 1000000, ekleyerek oluşturulur. Bu, üssün en büyük değerlikli bitini yani

işaret bitini ters çevirir. Böylece, 1 işareti pozitif veya sıfır üssünü, 0 işareti negatif üssünü belirtir. Üs 64 ise, 0000000; üs 0 ise 1000000; ve üst +63 ise, 1111111 ile temsil edilir. O halde, 7-bitlik üs gerçekte -64 ile +63 arasında bir ikiliktir. Şekil 7.27b, bu formatta ifade edilmiş bazı sayıları göstermektedir. Çoğu zaman yapıldığı gibi, her dört ikilik basamağı temsil etmek üzere hex basamaklar kullanılmıştır.

ANSI FORTRAN kayan-virgüllü format, $0{,}0001 \times 16^{-64}$ ile $0{,}FFFFFF \times 10^{+63}$, yani yaklaşık 10^{-79} ile 10^{+76} arasındaki sayıları temsil etmeyi mümkün kılar. Bu yöntem, öncekine göre daha büyük bir aralık vermekteyse de, 24-bitlik mantis kullanıldığı için, çözünürlük hâlâ yaklaşık olarak 7 onluk basamakta 1 dir. Bazı ara hesaplarda oluşan ilave bitlerini tutmak üzere, sayılar arasında koruma baytı denilen bir 8-bitlik bayt konulmuştur. Bu, hesapların genel doğruluğunu geliştirir, çünkü kesilmek yerine yuvarlanabilir.



(a)

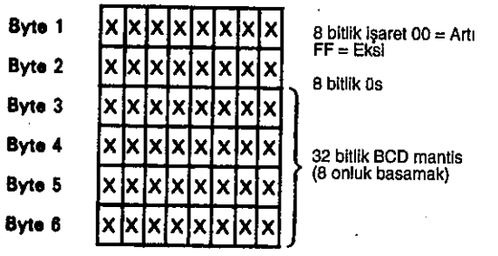
Onluk	ANSI FORTRAN Onaltılık İşaret ve Üs	Mantis
1.00	41	10 00 00
100	42	64 00 00
−1.00	C1	10 00 00
.50	40	80 00 00
−.25	C0	40 00 00
0.00	00	00 00 00

(b)

Şekil 7.27 ANSI FORTRAN onaltılık kayan-virgüllü sayılar. *(a)* Format. *(b)* Örnekler.

KAYAN-VİRGÜLLÜ BCD GÖSTERİLİŞLERİ

Bir işlemin sonucunu onluk basamaklar olarak görüntüleyen elektronik hesap makinaları, çoğunlukla kayan-virgüllü sayıların BCD gösterilişini kullanır. Şekil 7.28a, bu formatı göstermektedir. Bir 32-bitlik mantis, sekiz onluk basamağın temsil edilmesine imkân verir. 8-bitlik üs, çoğunlukla 128-fazla şeklinde ifade edilir. 128-fazla şekli, üssün ikiye-tümleyenine 128_{10}, yani 10000000_2, ekleyerek bulunur. Üs, 10 un kuvvetlerini gösterir. Mantisin işareti için tam bir bayt ayrılmıştır. FF_{16}, negatif bir mantisi, 00_{16} da pozitif bir mantisi gösterir. Şekil 7.28b, bazı örnekler vermektedir.



(a)

Onluk	BCD İşaret	Kayan Virgüllü Format (Hex Basamaklar) Üs	Mantis
1.00	00	81	10 00 00 00
5.00	00	81	50 00 00 00
−1.00	FF	81	10 00 00 00
100	00	83	10 00 00 00
−.25	FF	80	25 00 00 00
$2^{16} = 65536$	00	85	65 53 60 00

(b)

Şekil 7.28 BCD kayan-virgüllü sayılar. *(a)* Format. *(b)* Örnekler.

Kayan-virgüllü sayılarla aritmetik işlemler yapan algoritmalar, şu anda anlattıklarımızın alanının dışındadır.

GÖZDEN GEÇİRME SORULARI VE PROBLEMLER

1. Aşağıdakileri toplayın :

   a) İkilik olarak 10111 ile 1011.

   b) BCD olarak 37 ile 26.

   c) 3-fazla BCD olarak 37 ile 26.

   d) 37_8 ile 26_8

   e) $4A_{16}$ ile 78_{16}

2. Aşağıdaki onluk sayıların ikilikte çıkarılmasını gösterin.

   a) 7 - 4 *b)* 37 - 26 *c)* 125 - 93

3. Aşağıdaki onluk sayıları 8-bitlik işaret ve büyüklüğün ikiye-tümleyeni şeklinde ifade ediniz.

   a) +26 *b)* -7 *c)* -26 *d)* -125 *e)* 0 *f)* -1 *g)* -97 *h)* +85

4. Aşağıdaki 3-bitlik işaretli ikilik sayıları onluğa çevirin:

   a) 00110011 b) 10001100 c) 11110110 d) 01011101

5. Aşağıdaki onluk sayıları, bunların 8-bitlik ikiye-tümleyen işaret-ve-büyüklük gösterilişlerini kullanarak toplayın.

   a) +3 , -7 *b)* -3 , -7 *c)* -37 , +26

6. Aşağıdaki çıkarmaları yapınız.

   a) BCD olarak 17 - 8

   b) 3-fazla BCD olarak 17 - 5

   c) 22_8 - 13_8

   d) 72_{16} - $3A_{16}$

7. Şekil 7.18 i kullanarak, aşağıdaki pozitif mantık fonksiyonlarını gerçekleştirmek üzere 74181 in seçme ve mod girişlerinin nasıl programlanacağını gösteriniz.

   a) $\overline{A + B}$ *b)* $A \oplus B$ *c)* $A . \overline{B}$

8. Aşağıdaki ikilik kelimeler VE lendiğinde ve VEYA landığında elde edilecek çıkış kelimelerini bulun.

   a) 1010 ile 0111

   b) 1011 ile 1100

   c) 11010111 ile 111000

   d) 8-bitlik bir ikilik sayıyı 11110000 ile VE lemeye bazen az önemli 4 biti maskelemek denir. Bunun nedenini açıklayınız.

9. Şekil 7.20' yi kullanarak, aşağıdaki aritmetik fonksiyonları gerçekleştirmek için 74181'in gerektirdiği seçme ve mod girişleri programlamasını gösterin.

   a) A + B *b)* A - B - 1 *c)* A . B + A

10. Kalem yöntemini ve topla-ve-sağa-kaydır algoritmasını kullanarak, 1001 ile 011'i çarpın. Aynı yöntemlerle 11010 ile 101'i çarpın.

11. Kalem yöntemini ve çıkar-ve-sola-kaydır algoritmasını kullanarak, 1100100 sayısını 1010'a bölün.

12. Aşağıdaki sayıları ikilik kesir formatına normalize edin ve oluşan üsleri belirtin.

    a) 0,01101 *b)* 110001,0 *c)* 0,000101

13. Aşağıdaki sayıların her birinin, kayan-virgüllü ikilik, ANSI FORTRAN ve BCD olarak nasıl temsil edileceklerini gösterin.

    a) 2,0 *b)* 76 *c)* -22 *d)* 0,125 *e)* -0,75

REFERANSLAR / BİBLİYOĞRAFYA

Boney, Joel: "Math in the Real World, " BYTE: The Small Systems Journal. cilt 3. no. 9. Eylül 1978, sayfa: 114-119.

Digital Integrated Circuits. Digital Data Handbook. National Semiconductor Corporation, Santa Clara, CA, 1974.

Hashizume, Burt: "Floating Point Arithmetic," BYTE: The Small Systems Journal, cilt 2. no.11. Kasım 1977, sayfa 76-78, 180-188.

Nashelsky, Louis: Digital Computer Theory, chap. 1010_B. Wiley, New York, 1966, sayfa : 216-235.

BÖLÜM 8

MİKROİŞLEMCİ YAPISI VE PROGRAMLAMA

Önceki bölümlerde, bir bilgisayar veya mikroişlemci sisteminde kullanılan bütün elemanlar anlatılmıştı. Şimdi yapılması gereken, bir bilgisayar veya mikroişlemci sistemi oluşturmak üzere bu parçaların nasıl bağlanacağını göstermektir. Kavram olarak, mikroişlemcileri anlamak, Bölüm 7 'deki ALU yu anlamanın ufak bir adım ötesidir. 74181 in 4 bitlik iki tane iki tabanlı ifade üzerinde mantık ve aritmetik fonksiyonlar gerçekleştirebildiğini hatırlayın. Gerçekleştirilecek fonksiyon, mod girişiyle dört seçme girişine uygulanan ikilik komutta belirlenmektedir. Eğer, veri saklamak için RAM ve ardışıl ikilik-kodlanmış komutları saklamak için ROM ve kontrol devresi ALU 'ya eklenirse, ALU, birbiri ardından bir dizi işlemi gerçekleştirebilir. Bir bilgisayar veya mikroişlemci temelde bu şekilde bir yapıya sahiptir. Bölüm 7 'deki topla-ve-sağa kaydır çarpma yordamı, böyle bir mikroişlemci tarafından gerçekleştirilebilecek ardışıl işlemlerin bir örneğidir. Bellekte saklanan komutlar dizisine program denir.) İZLENCE denir

Bu bölümün sonunda şunları yapabileceksiniz:

1. Yolları ile birlikte basit bir bilgisayarın blok diyagramını çizebileceksiniz.
2. Komut getirme ve yürütmede yollar üzerindeki sinyal yollarını izleyebileceksiniz.
3. Bu bölümü referans olarak kullanarak, 8080A/8085A komutlarınca gerçekleştirilen işlemleri anlatabileceksiniz.
4. Basit bir programın akış tablosunu çizebileceksiniz.
5. Bir 8080A/8085A mikroişlemci için basit bir çeviri dili program yazabileceksiniz.
6. Bir 8080A/8085A programını elde çevirebileceksiniz.
7. Bir makina dili program üretmek için bir metin editör programının ve çevirici programın kullanılışını açıklayabileceksiniz.
8. Çeviri dili programların hatalarını ayıklamak için kullanılan teknikleri anlatabileceksiniz.

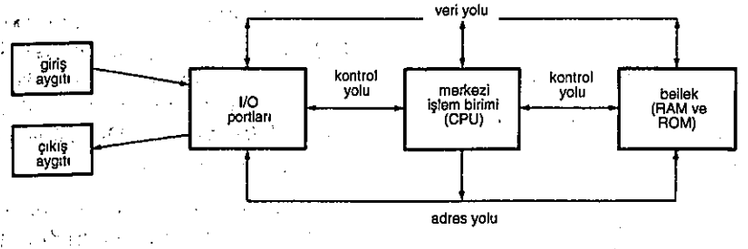
BİR SAYISAL BİLGİSAYARDA GENEL YAPI VE PROGRAM AKIŞI
BİLGİSAYAR SİSTEMİ BLOK DİYAGRAMI

Şekil 8-1, basit bir bilgisayar veya mikroişlemci sisteminin blok diyagramını göstermektedir. Üç ana kısım, bellek, CPU ve I/O portlarıdır.

BELLEK KISMI : Bellek kısmı genelde ROM,RAM, manyetik diskler veya banttan oluşur. Bellek, bilgisayarın yürüteceği komutlar dizisini yani programları saklamak için kullanılır. Bellek, ayrıca bilgisayarda işlenecek verileri veya işleme sonundaki verileri saklamak için kullanılır.

CPU KISMI: Daha sonra ayrıntılarıyla anlatılacağı üzere, merkezi işlem birimi (central processing unit: CPU), kontrol devresini, bir ALU, bazı kaydediciler ve bir adres/program sayıcı içerir. Bir programı yürütmek için, CPU, ilk yürütülecek komutun kodunun bulunduğu yerin adresini belleğe gönderir. CPU, ayrıca bir belleği yetkilendirme sinyali gönderir. Komut kodu, bellekten CPU ya gelir kodu çözülür ve yürütülür. Her işlemden sonra program sayıcı bellekte saklanan bir sonraki komut veya verinin yerini adreslemek üzere artırılır.

GİRİŞ-ÇIKIŞ BAĞDAŞTIRMA KISMI: Bilgisayarın üçüncü kısmı, bilgisayarın dış dünya ile olan bağdaştırma devresidir. Bu bağdaştırma devresine çoğunlukla *giriş-çıkış (Input-Output: I/O)* portları denir. Bir giriş portu, CPU kontrolu altında, bir klavyeden gelen verilerin, bir basınç algılayıcısından gelen sayısallaştırılmış çıkışın veya bir başka verinin bilgisayara alınmasına imkân verir. Bir çıkış portu, teletayp, video ekran gösterge veya motor-hız kontrol birimi gibi bir çıkış aygıtına veri göndermekte kullanılır. Fiziksel olarak, bir çıkış veya giriş portu genellikle CPU tarafından veri geçişi kontrol edilen paralel D flip-flop lardır.



Şekil 8-1 Basit bir bilgisayar veya mikroişlemci sisteminin blok diyagramı

YOL YAPISI:

Üç ana yol, bellek, CPU ve I/O portlarını birbirine bağlar: Adresyolu, veri yolu ve kontrol yolu.

ADRES YOLU: Okunacak veya yazılacak bellek yerinin adresini, CPU, adres yolu üzerinden gönderir. Adres yolu ayrıca CPU tarafından belli bir giriş veya çıkış portunu seçmede kullanılır. Adres yolu 8,12 veya daha fazla paralel hattan oluşabilir.

VERİ YOLU: Şekil 8-1 deki çift uçlu oklar tarafından belirtildiği gibi, veri yolu çift yönlüdür. İşleme bağlı olarak veri, belleğe, CPU ya veya I/O portlara gidiyor veya bunlardan geliyor olabilir. Veri yoluna bağlanan herhangi bir aygıt çıkışı, üç-durumlu olmalıdır. Böylece aygıt adreslenerek okunmadığı zaman çıkışı kayar durumda tutulabilir. Bir veri yolu, 4,8,12 veya daha fazla paralel hattan oluşabilir.

KONTROL YOLU: Kontrol yolu, belleği okuma, belleğe yazma, girişi okuma ve çıkışa yazma gibi sinyalleri taşır. Bu sinyaller, bellek veya I/O portlarını uygun çalışma için yetkilendirilir. Örneğin, bellekte bir yeri okumak için, CPU önce adres yolunda adresi ve kontrol yolunda belleği okuma sinyalini gönderir. Bundan sonra, bellek adreslenen yerdeki veriyi veri yoluna koyar.

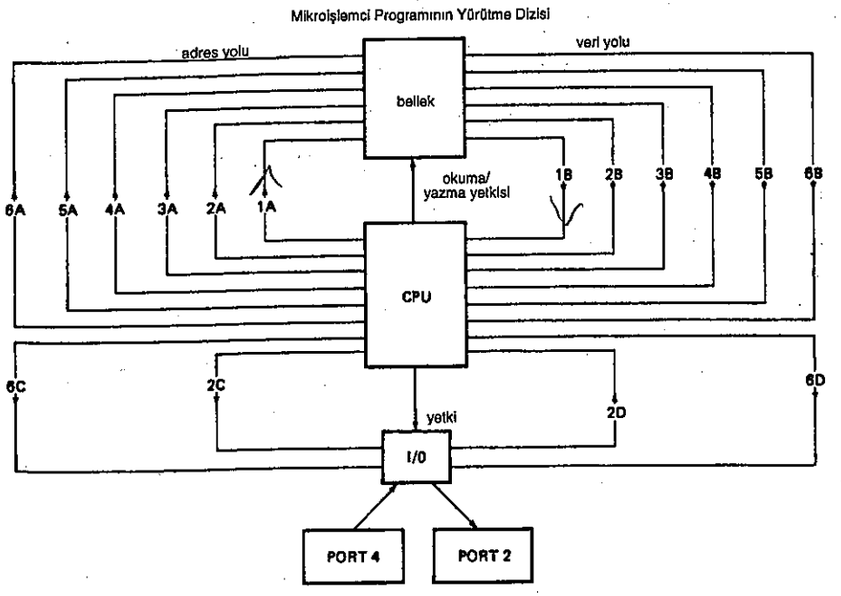
ÜÇ-ADIMLI BİR PROGRAMIN YÜRÜTÜLMESİ

Bir CPU nun ayrıntılarına girmeden önce, genel bir bilgisayarda basit bir programın çalışmasını anlatacağız. Bu, CPU, bellek ve I/O portların bir sistem olarak nasıl işlediğini size gösterecektir. Şekil 8-2 de gösterildiği gibi, örnek programda 3 adım bulunmaktadır:

1. Port 4 ten bir sayı gir.
2. Bu sayıya 7 ekle.
3. Sonucu port 2' den dışarı ver.

Güç ilk açıldığında, CPU otomatik olarak 0000H (adresin sonundaki *H*, adresin hex yani onaltılık olduğunu söyler) adresini okur, yani buradaki komutu getirir. Bellekteki bu adresin, gerçekleştirilecek ilk komutun ikilik kodunu içermesini programcının sağlaması gerekir. Örnek programdaki, ilk komut, 8-bit genişliğindeki bir makinada ikilik 11011011 yani hex DB olarak kodlanabilecek *IN (input from port: Porttan giriş yap)* komutudur.

Şekil 8-2a' daki *IA* isimli ok, ilk komutu getirme adresinin belleğe gönderildiğini göstermektedir. Böylece, bellek 0000H adresinin içeriğini CPU ya gönderir *(1B)*. CPU, bu ifadenin kodunu bir IN komutu olarak çözer. Sonra, istenilen giriş portunun adres numarasını almak üzere 0001H adresinden "getirme" yapar *(2A)*. Bellek bu sayıyı veri yolu üzerinden CPU' ya gönderir *(2B)*. Bundan sonra, CPU, 04H adresini adresyoluna ve giriş okuma sinyalini kontrol yoluna göndererek IN komutunu yerine getirir, yani yürütür *(2C)*. Giriş portu 4 deki ikilik sayı, CPU' ya gönderilir ve akümülatöründe saklanır *(2D)*.



DİZİ:

1A	İlk komutu getir
1B	Komutun gelişi (adresi bir sonraki bellek yerinde saklı portu adresle)
2A	Bir sonraki bellek yerini adresle (port adres için)
2B	Port adresinin gelişi (#4)
2C	Port 4 ün adreslenmesi
2D	Port 4 ten verinin girişi
3A	Bir sonraki komut için bir sonraki bellek yerini adresle
3B	Komutun gelişi (bir sonraki bellek yerinde saklanan sayıyı ekle)
4A	Bir sonraki bellek yerini adresle (veri için)
4B	Verinin gelişi (#7) ve CPU ya eklenişi
5A	Bir sonraki komut için belleği adresle
5B	Komutun gelişi (adresi bir sonraki bellek yerinde saklı portu adresle)
6A	Bir sonraki bellek yerini adresle (port adresi için)
6B	Port adresinin gelişi (#2)
6C	Port 2 yi adresle
6D	Veriyi port 2 ye gönder.

(a)

Bellek Adresi (Hex)	Belleğin İçeriği (ikilik)	Belleğin İçeriği (Hex)	Op Kodu	İşlenen
0000	11011011	DB	IN	04
0001	00000100	04		
0002	11000110	C6	ADI	07
0003	00000111	07		
0004	11010011	D3	OUT	02
0005	00000010	02		
0006				

Şekil 8.2: Üç adımlı bir mikroişlemci programı. *(a)* Yürütme dizisi. *(b)* Belleğin içerikleri ve komut bellenirleri.

Bu komut tamamlandığından, bir sonraki komutu almak üzere CPU bir başka getirme yapmalıdır *(3A)*. İkilik 11000110, yani hex C6, olarak kodlanmış ADI (add immediate: derhal ekle) komutu veri yolu üzerinden CPU ya gelir *(3B)*. Bu toplama komutu CPU'nun saklamakta olduğu sayıya hangi sayıyı ekleyeceğini bulması için, yeni bir getirme daha yapmasını gerektirir *(4A)*. Bellek, 00000111 ikiliğini, yani 07H' yi, toplanacak sayı olarak CPU ya gönderir *(4B)*. Daha sonra, CPU, toplama işlemini içinde gerçekleştirir ve sonucunu akümülatöre saklar.

Bir sonraki komutu almak üzere bir başka getirme gerçekleştirilir *(5A)*. İkilik 11010011, yani D3H, olarak kodlanmış OUT (output to port: porta çıkış ver) komutu veri yolundan CPU' ya gönderilir. OUT komutu, çıkış portunun adresini almak için ikinci bir getirme gerektirir *(6A)*. Bu adres, bellekten CPU ya veri yolu üzerinden gönderilir *(6B)*. CPU, çıkış port adresi 02H yi adres hatlarına *(6C)* ve toplamı veri yoluna *(6D)* koyar. Sonra, CPU, çıkış aygıtına veriyi yüklemek için, kontrol yoluna bir çıkış yazma darbesi gönderir. Şekil 8-2*b*, bu program için gerekli bellek içeriklerini göstermektedir. Hem adresler, hem de bellek içerikleri, yer kazanmak ve hataları azaltmak için hex olarak ifade edilmiştir. Bellek, tabii ki, sadece ikilik 1 ler ve 0 ları saklayabilir.

Bu işlemler, bir değer girip buna 7 ekleyip sonucu çıkıştan almak için çok fazla sayıda adım gerektiriyor gibi görünmektedir. Ancak, bir mikroişlemci, bu adımları bir kaç mikrosaniye içerisinde atabilmekte, yani gerçekleştirebilmektedir. Bir bilgisayar veya mikroişlemci çok hızlı olmasına karşın biraz aptaldır, çünkü ona yapacağı işlemlerin tam sırasının bir programla söylenmesi gerekmektedir. Hatta, CPU, bir veri baytı ile komut arasındaki farkı sadece

programdaki sıralanışlarından anlamaktadır. Bu nedenle, programlamada bilgisayar komut istiyorsa komut almasını; veri istiyorsa veri almasını garantilemek için çok dikkat edilmelidir. Şimdi CPU nun, komut ve verileri nasıl işlediğini görebilmeniz için, iç yapısını inceleyeceğiz.

MİKROİŞLEMCİ CPU LARİ: 8080A VE 8085A MİKROİŞLEMCİLERİ

Mikroişlemcileri belli örnekler kullanarak öğrenmek çok daha kolay olduğu için, bu ve bundan sonraki bölümde 8080A ve 8085 A yı kullanacağız. Her ne kadar 8085A, yeni tasarımlarda 8080 'nın yerini almışsa da, 8080A, yıllarca en çok satan mikroişlemciydi. Bu nedenle şu anda birçok cihazda bulunmaktadır ve referans amacıyla burada içerilmişlerdir.

Şekil 8-3*a*, 8080A CPU nun bir işlevsel blok diyagramını ve Şekil 8-3*b* 8085A CPU nun bir işlevsel blok diyagramını göstermektedir. CPU ların kalpleri, aritmetik mantık birimleri, yani ALU lardır.

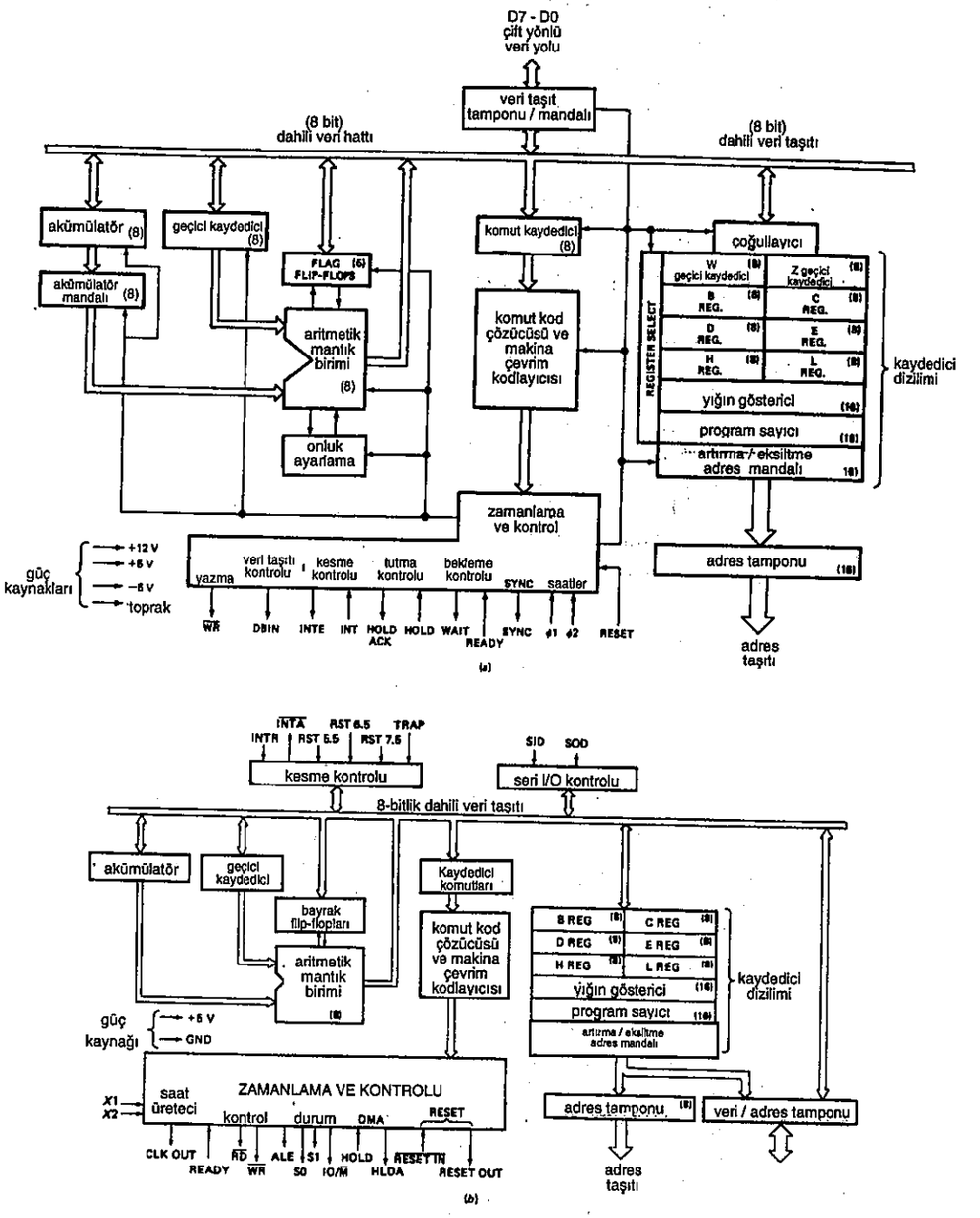
ALU VE İLGİLİ DEVRELERİ: ARİTMETİK MANTIK GRUBU

8080A ve 8085A, 8-bitlik işlemcilerdir, çünkü veri taşıtları 8-bit genişliğindedir ve ALU ları 8-bitlik kelimeler üzerinde işlem yapmak üzere tasarlanmıştır. Her işlemcideki ALU, bir ikilik miktarı bir artırabilir bir eksiltebilir ve iki 8-bitlik miktar üzerinde VE, VEYA, özel VEYA, toplama çıkarma veya karşılaştırma işlemleri yapılabilir. ALU; akümülatör, akümülatör mandalı, geçici kaydedici, bayrak flip-flopları ve onluğa ayarlama devresini içeren aritmetik mantık grubu devrelerin bir parçası olarak işlev görür.

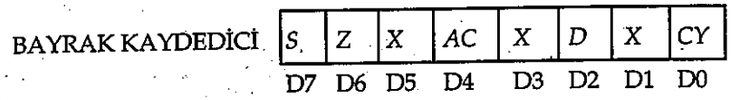
AKÜMÜLATÖR: Akümülatör, ALU tarafından üzerinde işlem yapılacak miktarlardan birini tutan ve ALU tarafından yapılan bir işlemin sonucunu saklayan 8-bitlik özel bir kaydedicidir. Örneğin, bir komut yürütüldüğünde, *B* kaydedicisinin içeriğini akümülatörün içeriğine ekleyip toplam sonucunu akümülatöre koyabilir.

ONLUĞA AYARLAMA DEVRESİ: Eğer BCD toplama veya çıkarma gerçekleştiriliyorsa, *akümülatörü onluğa ayarlama (decimal adjust accumulatör: DAA)* devresi, sonuçta doğru BCD formatı sağlamak için kullanılır.

BAYRAK FLİP-FLOPLARI: Her aritmetik veya mantık komutunun yürütülmesinden sonra, beş bayrak flip-flopu, işlem sonucunda oluşan elde gibi durumları belirtmek üzere set veya reset edilir. Bu bayrak flip-flopları bayrak kaydedicisini oluşturur. Şekil, 8-4 bayrak kaydedicisinde, bu beş bayrak flip-flopunun yerlerini göstermektedir. Kaydedicinin *D1, D3* ve *D5* bitleri kullanılmamaktadır, bu nedenle bunlar ihmal edilebilir.



Şekil 8-3 ***(a)*** **8080A CPU nun işlevsel blok diyagramı.** ***(Intel Corporation).*** ***(b)*** **8085A CPU nun işlevsel blok diyagramı.** ***(Intel Corporation)***



Şekil 8-4 8080A/8085A bayrak kaydedici formatı

Elde *bayrak biti, CY (Carry flag bit)*, bir işlem sonucunda bir elde oluşmuşsa 1 e set edilir, elde yoksa 0 a reset edilir. Örneğin, bir toplama komutu iki adet ikilik sayıyı toplarsa ve sonuç FFH den büyükse, elde bayrağı bu durumu göstermek üzere set edilecektir. Çıkarma işleminde, elde bayrağı, çıkarma işlemini gerçekleştirmek için bir ödünç alma gerekip gerekmediğini belirtir. Diğer bir deyişle, eğer akümülatörden çıkarılan sayı, akümülatördekinden büyükse, bu durumda, çıkarma işleminin gerçekleştirilmesi için, bir ödünç gerektiğini belirtmek üzere *elde/ödünç* bayrağı set edilecektir.

Parite bayrak biti, P (parity flag bit), eğer akümülatördeki sayıda bulunan 1 ler toplamı çift ise set edilir, tek ise reset edilir.

Yardımcı elde bayrak biti, AC (auxiliary carry flag bit), eğer akümülatör- deki bit, 3'ten bir elde çıkışı varsa set olur. BCD toplama veya çıkarma yapılırken, bu bayrak, onluğa ayarlama işleminin yapılması gerektiğini belirtmek için kullanılır. Onluğa ayarlama işlemi Bölüm 7'de anlatılmıştır.

Sıfır bayrak biti, Z (zero flag bit), eğer yürütülen bir komut sonunda, akümülatördeki veya bir kaydedicideki sayı 0 olursa, set edilir. Örneğin, akümülatörde 01H varsa ve sonra akümülatör 1 azaltırsa sonuç 00H olur. Bu durumda sıfır bayrağı set olur.

İşaret bayrağı biti S (sign flag bit), akümülatörün bit 7 sinin bir kopyasıdır. Bölüm 7 de anlatıldığı gibi, 8-bitlik işaretli (pozitif veya negatif) sayılarla çalışırken, en büyük değerlikli bit *D7*, işaret biti olarak kullanılmaktadır. Bu durumda, eğer işaret biti 1 ise akümülatördeki sayı negatifdir ve 8-bitlik ikiyetümleyen şeklindedir. Eğer işaret biti 0 ise, akümülatördeki sayı pozitifdir ve normal ikilik şeklindedir.

Bu bayrakların herhangi birisi, işlemciye bir başka programa veya program kısmına dallanmasını veya atlamasını söylemek için kullanılabilir. Örneğin, eğer bir işlemin sonucu tek parite verirse, reset edilmiş parite bayrağı, işlemciye bir hata mesajı çıkışı veren bir altyordama atlamasını söylemekte kullanılabilir.

KAYDEDİCİLER, PROGRAM SAYICI VE ADRES MANDALI

KAYDEDİCİ ÇİFTLERİ: Akümülatör ve bayrak flip-flop kaydedicilerinden başka, 8080A ve 8085A nın her ikisi de, çiftler oluşturacak şekilde altışar tane 8-bitlik kaydedicilere sahiptir. Çiftler, *B* ile *C*, *D* ile *E* ve *H* ile *L* dir. Geçici kaydedici çifti *W* ile *Z* kullanıcı tarafından kullanılamazlar. *BC*, *DE* ve HL kaydedicilerine bazan yaz-boz kaydedicileri denir. Örneğin, *B* kaydedici tek başına bir veri baytı saklamak için kullanılabilir. *C* kaydedici ile birlikte ise 16-bitlik veri veya 16-bitlik adres saklamak için kullanılabilir. Bir çoğullayıcı, iç veri hattındaki veri baytının *B* sütunu kaydedicilerine veya *C* sütunu kaydedicilerine yönlendirilmesini sağlar.

YIĞIN GÖSTERİCİSİ: *Yığın göstericisi (stack pointer)* genellikle bir adres saklamak için kullanılan 16-bitlik bir kaydedicidir. Bunu daha sonra ayrıntılarıyla anlatacağız.

PROGRAM SAYICI: Bir program sayıcı, bir kaydedici veya paralel-yüklemeli sayıcı olarak düşünülebilir. CPU tarafından okunmakta veya yazılmakta olan bellek adresini saklar. Her komuttan sonra, program sayıcı, bellekteki bir sonraki komut veya verinin yerini gösterecek şekilde otomatik olarak artırılır.

ADRES MANDALI: Adres mandalı kısmı iki işlev görür. Birincisi, adres mandalı, program sayıcıdan, yığın gösterici kaydediciden veya 16-bitlik kaydedici çiftlerinden birinden gönderilecek adresin seçimini yapar. İkincisi, adres mandalı, bu adresi, adres hatlarına, gerekli süre boyunca kilitler. 8080A ve 8085A daki 16-bitlik adresler, her mikroişlemcinin 2^{16}, yani 65536, bellek yerini adreslemesine imkân verir. Bir artırıcı-eksiltici, 16-bitlik kaydedicilerden herhangi birisinin içeriğinin azaltılıp çoğaltılmasına imkân verir.

KOMUT KAYDEDİCİ, KOMUT KOD ÇÖZÜCÜSÜ VE KONTROL DEVRELERİ

Komut kaydedici, veri taşıtı üzerinden kod çözücüsüne gelen komutları tutar. Kod çözücüsü, komutu yorumlar ve komutu yerine getirmek için uygun sinyalleri üretir. Örneğin, yukarıdaki üç-adımlı programın ADD komutu kod çözücüye gelince, kod çözücü, başka bir bellekten getirme gerektiğini farkedecektir. Akümülatöre eklenecek sayıyı almak için ikinci bir getirme gerekmektedir. Kod çözücü, kontrol devresini, yeni bir belleği okuma darbesi göndermek ve akümülatöre yüklenebilmesi için, veri yolu üzerinden gelen veriyi geçici kaydediciye yollamak üzere yönlendirir. Toplama bitirildiğinde kontrol devreleri sonucu tekrar akümülatöre gönderir. Sonra bu kontrol devreleri, program sayıcısını bir sonraki bellek adresine artırır ve bellekten bir sonraki komutu almak için yeni bir bellek okuma darbesi gönderirler. 8080A nın +12,

+5 ve -5 V luk besleme gerilimleri gerektirdiğine dikkat edin. 8085A sadece +5V luk besleme gerektirir. 8080A, dış saat üretecinden iki saat sinyali *Ø1* ve *Ø2*, gerektirir. 8085A, sadece *X1* ve *X2* girişlerine bağlanan bir dış kristal gerektirir. Bu mikroişlemcilerin geri kalan giriş ve çıkış sinyalleri Bölüm 9'da anlatılmaktadır.

8080A VE 8085A İÇİN ÇEVİRİ DİLİ (ASSEMBLY LANGUAGE) PROGRAMLARI

Bir programı çalıştırabilmek için, bu program mikroişlemcinin artarda gelen bellek yerlerinde ikilik şekilde saklanmış olmalıdır. Programın bu ikilik şekline makina dili denir; çünkü bu makina tarafından istenen şekildir. Ancak, programlayıcı kimseler için, örneğin bütün 8080A komutlarının 1 ler ve 0 lar şeklini hatırlaması çok zordur. İkilik sayıların onaltılık gösterilişlerini kullanmak biraz yardımcı olmaktadır; fakat hâlâ 256 'ya yakın sayıda farklı komutun ezberlenmesi gerekmektedir.

Çeviri dili programlama yapması daha kolaydır ve çok daha anlaşılır programlar verir. Bir çeviri dili program tamamlandıktan sonra çevrilir, yani assemble edilir. Bunun anlamı, mikroişlemcide çalıştırılmak için, programın makina dilinin 1 ler ve 0 larına çevrilmesi demektir.

Çeviri dili, her komutu temsil etmek üzere iki ile dört harf arasında uzunluğu olan bellenirler (mnemonic) kullanır. Bir bellenir, sizin birşeyi hatırlamanıza yardımcı olacak bir kısa yazılışdır. Bir çeviri dili bellenirinin harfleri, çoğunlukla komutun İngilizce kelimelerinin başharfleri veya kısa yazılışıdır. Örneğin, *"B kaydedicisinin içeriğini akümülatöre taşı" (move the contents of the B register into the accumulator)* komutu *MOV A,B* şeklinde gösterilmektedir. Bu şekil, komuta ait temel bilgileri saklar, fakat minimum alan kullanır. Gelecek kısımda, çeviri dilinde yani bellenir şeklindeki 8080A/8085A komut kümesine, ilk bakışımızı yapacağız.

8080A/8085A KOMUT KÜMESİ

Tablo 8-1, her 8080A/8085A komutunun bellenirini ve hex (onaltılık) kodunu göstermektedir. 8085A ya eklenmiş olan SIM ve RIM komutları dışında,iki işlemcinin komutları aynıdır. Her komut, birbirlerinden tek bir boşlukla ayrılan işlem kodu (operation code), yani op kod, ve işlenen (operand) olmak üzere iki kısımdan oluşur. Op kodu, yapılan işlemin belleniridir ve işlenen de üzerinde işlem yapılan kaydedici veya veridir.

Elinizin altında olması için, bütün bu komutlar birlikte izleyen kısımlarda açıklanmaktadır. Ancak bu komutların hepsini birden derhal özümlemeye çalışmayın. Bu kısımları ilk okuyuşunuzda, aşağıdaki komut verilerini anlamaya çalışın:

Veri aktarım grubu: Taşı, derhal taşı ve derhal yükle.

Aritmetik ve mantık grubu: Topla, çıkar, artır, eksilt, mantıksal, aritmetik ve derhal mantıksal.

Dallama kontrol grubu: Atla

I/O ve makina kontrol grubu: *Giriş-çıkış* ve *dur (HLT)*

Farklı birleşimlerin sayısı dolayısıyla bir sürü komut varmış gibi gözükmektedir, fakat gerçekte sadece 10 kadar farklı işlem tipi vardır. Bellenirler, her komutun ne yaptığını hatırlamanıza yardımcı olacaktır.

Bu komutların tanımlarını okuduktan sonra, basit programlarda bu komutların bazılarının nasıl kullanıldığını görmek için Şekil 8-20 ve 8-21'deki programlara bakınız.

Komut kümesi tanımlarının üzerinden ikinci geçişinizde, özeller grubunu, döndürme grubunu, çağırma grubunu, geri dönüş grubunu ve yığın op kod grubunu anlamaya çalışın. Sonra Şekil 8-22 deki programdan, bu komutların bazılarının nasıl kullanıldığına bakın.

Komut kümesineden bir üçüncü geçiş, yükleme /saklama ve çift-toplama gibi grupları halledecektir. Program yazmaya başladığınızda var olan komutların çoğunu çok çabuk öğreneceksiniz. Var olan komutların çoğunu bilmek, programınızın belli bir noktasında istenen işlemi en kolay gerçekleştiren birini seçmenize imkân verecektir.

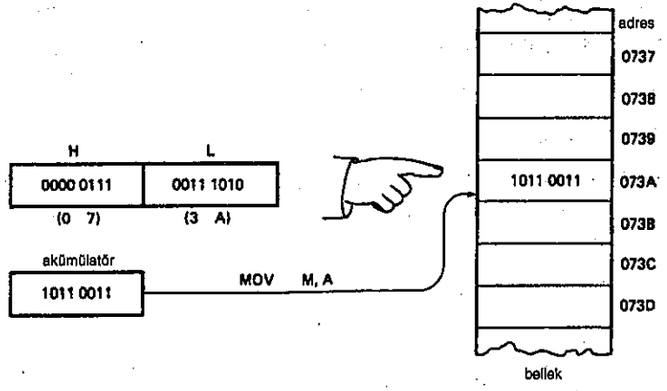
VERİ AKTARIM GRUBU KOMUTLARI

TAŞI KOMUTLARI: Komutların en büyük grubu taşı (move) grubudur. Bu grup, bir veri baytının bir kaydediciden diğerine veya bellekteki bir yer ile bir kaydedici arasında mümkün bütün aktarımları içerir. Hedef kaydedici, daima virgülden önce, ve kaynak kaydedici de daima virgülden sonra yazılır. Örneğin, *MOV C,D komutu, "D kaydedicisindeki baytı C kaydedicisine taşı"* demektir.

Taşıma işlemi, kopyalama olarak düşünülebilir, çünkü taşıma işlemi kaynak kaydedicideki veya bellek yerindeki veriyi değiştiremez. Örneğin, *D* kaydedicisinin 78H ve *C* kaydedicisinin 8AH içerdiğini kabul edin. Bir MOV *C,D* komutu yürütüldükten sonra, *C* kaydedicisi 78H içerir, *D* kaydedicisi de hâlâ 78H içerecektir. Komutların etkisini anlatmak için bazen bir sembolik şekil kullanılır. *MOV C, D* komutu için sembolik şekil *(C)← (D)* dir. Parantezler "içeriği" yönünü göstermektedir. Böylece sembolik şekil "*D* kaydedicisinin içeriği, *C* kaydedicisine kopyalanır" olarak okunur.

Eğer MOV M,A gibi bir belleğe taşıma veya bellekten bir kaydedici taşıma komutu kullanılıyorsa, önce istenilen bellek adresi *H* ve *L* kaydedici çiftine yüklenmiş olmalıdır. Diğer bir deyişle MOV M,A komutu, akümülatörün içeriğini adresi HL de saklanan bellek yerine kopyalar. Bu durumda, *HL* kaydedici çifti, bir bellek yerini gösterici olarak kullanılmaktadır.

Şekil 8-5, bir bellek taşıması için, göstericinin kullanılışını diyagram şeklinde göstermektedir. Bu durumda, *HL* kaydedici çifti 073AH içerdiğinden, MOV M,A komutu akümülatörün içeriği B3H' yi, belleğin 073A H adresine kopyalar. Eğer komut MOV A, M olsaydı ve *HL* kaydedici çifti 073AH içerseydi, belleğin 073AH adresinin içeriği akümülatöre kopyalanacaktı. Bu durumda belleğin 073AH adresinin içeriği değişmeyecekti.



Şekil 8-5 MOV M,A komutu için bellek göstericisi olarak kullanılan *H* ve *L* kaydedicileri.

DERHAL TAŞI KOMUTLARI: *Derhal taşı (move immediate)*, yani 'MVI kaydedici, bayt' şeklindeki, komutları, virgülden sonra yazılmış olan veri baytını belirtilen kaydediciye yükleyen komutlardır. Bu komutlar bellekte iki bayt gerektirir: Bir tane op kod için, bir tane de veri baytını saklamak için MVI A, 07H komutu için, ilk bellek yeri 3EH ın ikilik dengini ve bir sonraki bellek yeri de 07H verisini içerir. Çalıştırıldığında bu komut, akümülatöre 07H, yani 00000111, yükleyecektir. 'MVI M, bayt' komutuyla, verilen veri baytı belleğin HL tarafından gösterilen yerine taşınacaktır. Her 'derhal' komutu, op koddan sonra verilecek bir veya iki veri baytı gerektirir.

GENİŞLETİLMİŞ DERHAL YÜKLE: Bir sonraki grubun LXI belleri "genişletilmiş derhal yükle" (load extended immediate) anlamındadır. Burada "genişletilmiş", verilen 16-bitlik miktarın belirtilen kaydedici çiftine yükleneceği demektir. Örneğin, LXI *H*, 234AH komutu, *H* kaydedicisine 23 ve *L* kaydedicisine 4A yükleyecektir. Ancak, bu komut çalıştırılmak üzere belleğe yüklendiğinde, alt sıralı bayt, üst sıralı bayttan önce yüklenmelidir. LXI *H*, 234AH komutu için ardarda gelen üç bellek yeri şunları içerecektir: Tablo 8-1'den LXI H komutunun kodu 21H, L *kaydedicisine* gidecek alçak mertebeli bayt 4AH ve *H kaydedicisine* gidecek yüksek mertebeli bayt 23H.

YÜKLE VE SAKLA KOMUTLARI: Genel tanım olarak, bir yükle (load) komutu, bellek gibi bir yerden CPU kaydedici gruplarından birine veya akümülatöre veri getirmek demektir. Diğer taraftan, bir sakla (store) komutu, CPU grubundan bellek gibi bir başka yere veri göndermek demektir.

LDAX B, 16-bitlik genişletilmiş *BC* kaydedicisinde içerilen bellek adresindeki veriyi akümülatöre kopyalar. Diğer bir deyişle MOV A,M komutunda ve diğer M ile belirtilen bellekle ilgili olan komutlarda HL kaydedicisinin kullanıldığı gibi, genişletilmiş *BC* kaydedicisi bellek adresi için bir gösterici olarak kullanılmaktadır.

STAX *B* komutu, akümülatörün içeriğini, genişletilmiş *BC* kaydedicisi ile gösterilen bellek adresine kopyalar.

LDAX *D* ve STAX *D* komutları, genişletilmiş *DE* kaydedicisini bellek göstericisi olarak kullanır. LDAX *D*, *DE* nin gösterdiği adresteki veriyi akümülatöre kopyalar. STAX *D*, akümülatörün içeriğini, DE nin içeriğinin gösterdiği bellek adresine kopyalar.

"LHLD adres" ve "STHD adres" komutları, tek bir komutla genişletilmiş HL kaydedicisinin içeriğini saklamak veya saklandığı yerden geri getirmek için kullanılır. "LHLD adres" komutu, komutta verilen adresteki veriyi *L* kaydedicisine, bir sonraki adresteki veriyi *H* kaydedicisine yükler. Örneğin, LHLD 43A7H komutu, 43A7H adresinin içeriğini *L* kaydedicisine ve bir sonraki adres 43A8H ın içeriğini H kaydedicisine yükler. Bu komut artarda gelen üç bellek yerine şu şekilde kodlanır: Tablo 8-1 den komut kodu 2AH, adresin alçak mertebeli baytı A7H ve adresin yüksek mertebeli baytı 43H. "SHLD adres" komutu yukarıdaki komutun tersini gerçekleştirir. SHLD 43A7H komutu, L kaydedicisini 43A7H adresine ve *H* kaydedicisini 43A8H adresine kopyalar. Bu komut artarda gelen üç bellek yerinde 22, A7 ve 43 olarak kodlanacaktır.

"LDA adres" komutu, komutta verilen bellek adresindeki bir veri baytını akümülatöre kopyalar. Örneğin, LDA 3A78H, komutu, 3A78H adresinin içeriğini akümülatöre kopyalar. 3A78H adresinin içeriği aynı kalır.

"STA adres" komutu, LDA işleminin tersini yapar. Örneğin, STA 3A78H, akümülatörün içeriğini bellekteki 3A78H adresine kopyalar. Akümülatörün içeriği aynı kalır. Bu komutların avantajları, *HL* kaydedicisini gösterici olarak kullanmaya gerek kalmadan, bir veri baytını akümülatörden belleğe veya bellekten akümülatöre kopyalama imkânı vermesidir.

Bütün bu komutları, ilk defada anlamıyorsanız üzülmeyin. Bunlar burada, ileride başvurmak ve 8080A veya 8085A nın ne gibi işlemler yapabildiğini görmeniz için açıklanmaktadır. Daha sonraki bir kısımda verilen basit program örnekleri, çok kullanılan bazı komutların anlaşılmasını sağlayacaktır. Okumaya devam etmeden önce, Tablo 8-1 in kalan kısmına bir göz atın ve sadece bellenirlerine bakarak kaç tane komutu anlayabileceğinizi görün.

ARİTMETİK VE MANTIK GRUBU KOMUTLAR

TOPLA KOMUTLARI: "ADD kaydedici" şeklindeki topla (add) komutları, belirtilen kaydedicinin içeriğini akümülatöre ekler. ADD M komutu, *HL* kaydedicisi tarafından gösterilen bellek yerinin içeriğini akümülatörün içeriğine ekler. Eklemenin sonucu (toplam) akümülatörde bırakılır. Kaynak kaydedicinin veya bellek yerinin içeriği aynı kalır. Topla ve çıkar komutlarıyla bütün bayrak flip-flopları etkilenir.

Şekil 8-6, bir ADD *B* komutunun etkilerinin bir örneğini göstermektedir. Başlangıçta *B* kaydedicisi 01111000 ve akümülatör 00101001 içermektedir. ADD *B* komutu yürütüldükten sonra akümülatör toplamın sonucu 10100001 i içerir. ADD *B* komutu içeriğini değiştirmediği için, B kaydedicisinde hâlâ 01111000 vardır. Bayraklar, Şekil 8-6'da gösterildiği gibi olur. İşaret bayrak biti 1 dir, çünkü akümülatördeki bit 7 de bir 1 vardır. Sıfır bayrak biti 0 dır, çünkü akümülatördeki sonuç 0 değildir. *AC* bayrak biti 1 dir, çünkü, toplam sırasında bit 3 ten bir elde çıkışı olmuştur. Parite bayrak biti 0 dır, çünkü akümülatördeki 1 lerin sayısı tektir. *CY* bayrak biti 0 dır, çünkü sonuç FFH dan büyük değildir.

	D7	D6	D5	D4	D3	D2	D1	D0
B KAYDEDİCİSİ	0	1	1	1	1	0	0	0
AKÜMÜLATÖR	+ 0	0	1	0	1	0	0	1
AKÜMÜLATÖR (ADD	1	0	1	0	0	0	0	1
B KOMUTU YÜRÜTÜLDÜKTEN SONRA)								
B KAYDEDİCİSİ (ADD *B* DEN SONRA)	0	1	1	1	1	0	0	0

	S	Z		AC		P		CY
BAYRAK KAYDEDİCİSİ (ADD *B* DEN SONRA)	1	0	X	1	X	0	X	0

Şekil 8-6 ADD *B* komutunun, kaydedici ve bayraklar üzerindeki etkisi.

Eğer bir toplamanın sonucu FF_{16} (255_{10}) den büyükse elde bayrağı flip-flop set edilir. Çok baytlı sayılar toplanırken, bu elde bir üst mertebeden olan baytların toplamında kullanılabilir. "ADC kaydedici" yani elde ile birlikte-toplam komutları bu amaçla kullanılmaktadır.

"ADC kaydedici" komutu belirtilen kaydedicinin içeriği ile elde flip-flopunun içeriğini akümülatöre ekler. Eğer elde bayrağı set (1) ise, kaydedici ile akümülatör toplamına 1 eklenir. Eğer elde reset (0) ise 1 eklenmez. Eğer bir elde-ile-birlikte-topla işleminin sonucu FFH den büyükse, elde bayrağı set edilir. Sonuç FFH ye eşit veya ondan küçükse, elde bayrağı reset olur. Şekil 8-7, karşılaştırma amacıyla, ADD ve ADC komutlarının etkilerini göstermektedir. Her iki komutun da, toplam sonucu FFH den az olduğu için, elde bayrağını reset ettiğine dikkat ediniz.

	ELDE	*A*		*B*	
KOMUTTAN ÖNCE	1	1100	1010	0010	0010
ADD *B* DEN SONRA	0	1110	1100	0010	0010
ADC *B* DEN SONRA	0	1110	1101	0010	0010

Şekil 8-7 ADD *B* ve ADC *B* komutlarının karşılaştırılması

ÇIKARMA KOMUTLARI: "SUB kaydedici" komutu, belirtilen kaydedici veya bellek yerinin içeriğini, akümülatörün içeriğinden çıkarır. Sonuç akümülatör de bırakılır. Sonuç negatifse, elde bayrağı set edilir. Çıkarmada, set edilmiş elde bayrağı, çıkarmayı gerçekleştirmek için bir ödünç gerektiğini belirtir. Diğer bir deyişle, çıkarmada elde bayrağı ödüncü gösterir.

Şekil 8-8 çıkarmanın nasıl gerçekleştirildiğine ait örnekleri götermektedir. ALU çıkarmayı, üstteki sayıya alttaki sayının ikiye-tümleyenini ekleyerek yapar. A7H-73H işlemi kalemle yapıldığında fark 34H olarak bulunur ve elde set edilmiştir. Ancak, çıkarma işleminde, dahili olarak elde bayrağının tümleyeni alınır. Böylece, gerçek sonuç, çıkarmayı yapmak için bir ödünç alma gerekmediğini belirtmek üzere elde bayrağını reset olarak gösterir.

Şekil 8-8b deki 73H-A7H çıkarması için ise, çıkarmayı yapmak için bir ödünç alma gerektiğini belirtmek üzere elde bayrağı set edilmiştir. Tek baytlık çıkarmada, set edilmiş elde bayrağı ayrıca sonucun negatif ve ikiye-tümleyeni şeklinde olduğunu belirtir. O halde, 73H-A7H için sonuç-34H olur. Çok baytlı sayıları çıkarırken, elde (ödünç) bayrağının içeriği bir üst mertebe bayttan çıkarılabilir.

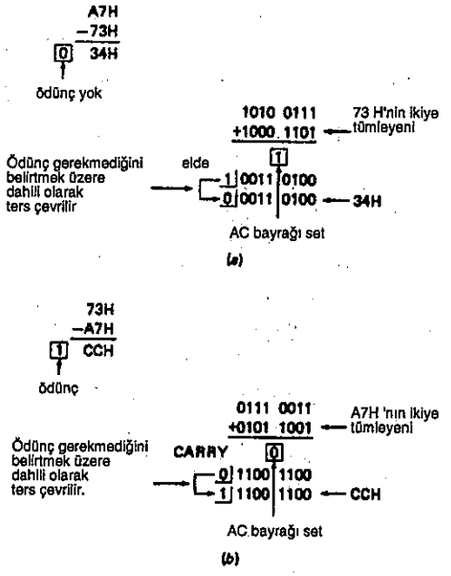
Topla ve çıkar komutlarının 8-bitlik işaret-ve-büyüklük olarak gösterilen sayılarla nasıl çalıştığına ait örnekler için Bölüm 7 deki örneklere başvurun.

"SBB kaydedici" (Subtract with borrow: ödünçle birlikte çıkar) komutları, belirtilen kaydedicinin içeriği ile ödünç (elde) bayrağının içeriğini akümülatörden çıkarır. Yine, sonuç akümülatörde bırakılır. Çıkarmanın sonucu negatif ise, elde (ödünç) bayrağı bunu belirtmek üzere set edilir.

ÇİFT-TOPLA KOMUTLARI: *Çift-topla, yani "DAD kaydedici çifti"* komutları, belirtilen kaydedici çiftinin içeriğini, genişletilmiş *HL* kaydedicisine ekler. Sonuç, *HL* kaydedici çiftinde bırakılır. FFFFH'den daha büyük bir sonuç, elde bayrağını set eder. Bu komut, iki 16-bitlik sayının tek bir komutla toplanmasına imkân verir.

DAD *H* komutu *HL* kaydedicisinin içeriğini ikiye katlar (2 ile çarpar). Bu, her biti bir basamak sola kaydırmak ve en önemsiz bite 0 girmekle aynıdır.

ARTIR VE AZALT KOMUTLARI: *Artır (increment)* ve *azalt (decrement)* komutları oldukça kolaydır. "INR kaydedici" komutu, belirtilen kaydedicinin içeriğini 1 artırır ve "DCR kaydedici" belirtilen kaydedicinin içeriğini 1 azaltır. INR ve DCR nin sonuçları belirtilen kaydedicide bırakılır. Tablo 8-1'de çift yıldızlı dipnotta gösterildiği gibi, artır ve azalt komutları elde bayrağı dışında bütün bayrakları etkiler. Örneğin, eğer bir DCR *C* komutu, *C* kaydedicisinde hep sıfır bırakırsa, sıfır bayrağı set (1) olur. Eğer akümülatör FFH içeriyorsa, bir INR *A* komutunu yürütmek akümülatörün içeriğini hep 0 yapar. Bunu göstermek üzere, sıfır bayrağı set edilir.



Şekil 8-8 İşaretsiz ikilik çıkarma

"INX kaydedici çifti" komutu belirtilen 16-bitlik genişletilmiş kaydediciyi 1 artırır ve "DCX kaydedici çifti" komutu belirtilen kaydedici çiftini 1 azaltır *(SP, yığın göstericisi (stack pointer)* demektir.) Tablo 8-1 de gösterildiği üzere, INX ve DCX komutlarının hiçbir bayrağı etkilemediğine dikkat edin. Genişletilmiş kaydedicileri geciktirme döngülerinde kullanırken, bu noktayı unutmamak önemlidir.

ÖZEL KOMUT GRUBU: Akümülatörü onluğa ayarla komutu DAA, BCD toplama gerçekleştirildiğinde, doğru sonucu üretmek için kullanılır. Bu düzeltme, Bölüm 7'de anlatılmıştı, Şekil 8-9, bu komutun etkisini göstermektedir. Başlangıçta, akümülatör 99 un BCD kodu olan 10011001 i ve *B* kaydedicisi de 28'in BCD kodu olan 00101000'i içermektedir. ADD *B* komutunun yürütülmesi sonucu akümülatörde 11000001 elde edilir ki, bu doğru BCD sonuç değildir. DAA komutu bu problemi çözer. Bu komut akümülatördeki toplamın her 4 biti (nibıl) üzerinde ayrı ayrı çalışır. Eğer daha küçük değerlikli, 4 bit nizami olmayan bir BCD kodu oluşturuyorsa, veya eğer toplama, bit 3 ten bir elde üreterek yardımcı elde (AC) bayrağını set etmişse, bu takdirde DAA komutu, akümülatördeki alt nibıla 0110 düzeltme faktörünü ekleyecektir. Benzer şekilde, eğer orijinal toplamada üstteki nibıldan bir elde üretilmişse veya eğer sonucun üst nibılı nizami olmayan bir BCD kod içeriyorsa, üstteki nibıl da düzeltilir. Gösterilen örnekte, alt ve üst nibılların her ikisi de doğru sonuç 0001 0010 0111 BCD, yani 127_{10}, vermek için düzeltme gerektirmiştir. Çoğunlukla bu konuda çok fazla düşünmenize gerek yoktur, çünkü DAA komutu bu işlemleri otomatik olarak yapar. Ancak, DAA komutunun ancak *yardımcı elde (AC) bayrağını* etkileyen bir komuttan sonra doğru çalışabileceğine dikkat edin.

CY	AC		
	0		
0	1001 1001	AKÜMÜLATÖR - BCD99	
+	0010 1000	B KAYDEDİCİ - BCD 28	
	1		
0	1100 0001	ADD B DEN SONRA AKÜMÜLATÖR	
+	0110 0110	DAA	
	0		
1	0010 0111	DAA DAN SONRA ELDE VE AKÜMÜLATÖR-BCD 127	

Şekil 8-9 BCD toplamada *DAA* komutunun etkileri

CMA (complement accumulator) komutu, akümülatördeki bütün bitleri ters çevirir yani tümleyenini alır.

STC (set carry) komutu, elde bayrağı flip-flopunu 1'e set eder.

CMC (complement carry) komutu, elde bayrağını ters çevirir.

DÖNDÜR KOMUTLARI: Döndür veya kaydır komutlarının elde bayrağı flip-flopunu kullanış şeklindeki farklılıklarına bağlı olarak, ikişer türü vardır. Şekil 8-10, bu dört komutu göstermektedir.

RAL, elde üzerinden sola döndür (rotate left through carry) demektir. Akümülatörün en büyük değerlikli biti sola kaydırılarak elde flip-flopuna konur ve akümülatörün her biti bir basamak sola kaydırılır. Elde flip-flopunun orijinal (ilk) içeriği de akümülatörün en küçük değerlikli bitine kaydırılır. Elde filip-flopu, temelde, akümülatörün bir uzantısı olarak görülmektedir.

RLC, *sola ve eldeye döndür (rotate left and to carry)* komutu, akümü-latörün en büyük değerlikli bitini sola kaydırıp elde flip-flopuna ve akümülatörün en küçük değerlikli bitine koyar. Akümülatörün diğer bitleri 1 basamak sola kayar.

RAR, Şekil 8-10c de görüldüğü gibi, akümülatörü elde üzerinden sağa döndür (rotate accumulator right through carry) demektir.

RRC, sağa ve eldeye döndür (rotate right and to carry), akümülatörün en küçük değerlikli bitini, en büyük değerlikli bitine ve elde flip-flopuna kaydırır. (Şekil 8-10d).

Kaydırma komutları, mikroişlemcilerde, topla-ve-sağa kaydır çarpma ve çıkar-ve-sola kaydır bölme yaparken faydalıdır.

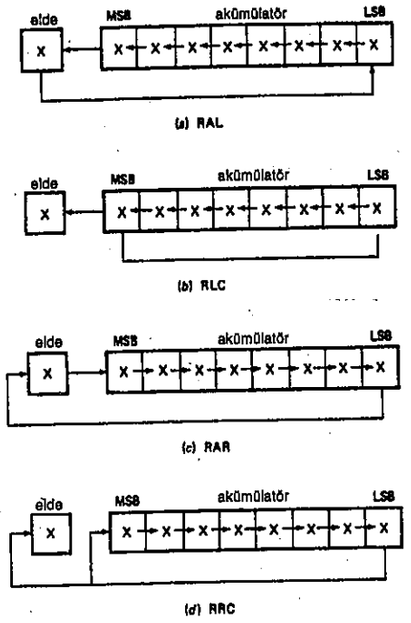
MANTIKSAL KOMUTLAR: ANA, XRA, ORA, CMP

Bu komutlar oldukça basittir. "ANA kaydedici" komutu belirtilen kaydedicinin içeriğini akümülatörün içeriği ile bit VE ler. Sonuç akümülatörde bırakılır. *CY*reset edilir.

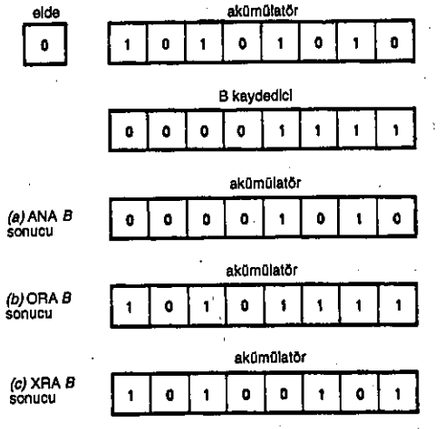
"XRA kaydedici" komutu, belirtilen kaydedicinin içeriği ile akümülatörün içeriğinin özel VEYA sını oluşturur. *CY* reset edilir.

"ORA kaydedici" komutu, belirtilen kaydedicinin içeriği ile akümülatörün içeriğini VEYA lar. *CY* reset edilir. Yeniden gözden geçirme amacıyla, Şekil 8-11, iki 8-bitlik sayı VE lendiğinde, VEYA landığında ve özel VEYA landığında elde edilen sonuçları göstermektedir.

ANA komutu, sık sık bir ifadenin bitlerini maskelemek için kullanılır. Şekil 8-11 de gösterildiği gibi bir ifadenin 0 la VE lenen herhangi bir biti için sonuç 0 dır. Bu bitlere maskelenmiş denir, çünkü ilk durumları ne olursa olsun, VE leme sonucunda bu bitler daima 0 olur. Ancak, 1 le VE lenen bitler aynı



Şekil 8-10 Döndür komutları. *(a)* RAL. *(b)* RLC. *(c)* RAR. *(d)* RRC



Şekil 8-11 İki 8-bitlik sayı üzerindeki mantıksal işlemlerin sonuçları. *(a)* ANA *B* sonucu. *(b)* ORA *B* sonucu. *(C)* XRA *B* sonucu.

kalır. Şekil 8-11 de gösterildiği gibi, akümülatörün AAH (10101010) ve *B* kaydedicinin OFH (00001111) içerdiğini kabul edin. Bir ANA *B* komutu yürütüldükten sonra, akümülatörün üst 4 biti maskelenmiş olur ve alt 4 biti aynı kalır. Şekil 8-11a da gösterildiği gibi akümülatördeki sonuç OAH (00001010) dır. Bir kelimenin istenilen sayıda biti maskelenebilir.

Bir biti 0'la VEYA lamak o biti değiştirmez. Bir biti 1'le VEYA lamak ise onu 1 set eder. Şekil 8-11b nin gösterdiği gibi, bir ifadeyi OFH ile VEYA lamak, alttaki 4 biti 1 e set eder, fakat üstteki 4 biti değiştirmeden bırakır.

NOT: ORA *A* komutu, akümülatörün içeriğini değiştirmeden elde bayrağını reset etme de oldukça sık kullanılır. Bir biti 0 ile özel-VEYA lamak, o biti değiştirmez. Bir biti 1 ile özel -VEYA lamak o bitin tersini alır. Şekil 8-11 de gösterildiği gibi, AAH, OFH ile özel-VEYA lamak, üstteki 4 biti aynı bırakıp ve alttaki 4 biti ters çevirerek A5H sonucunu verir. Bu komut, bir kelimedeki bitlerin tümleyenini almak için sık sık kullanılır. XRA A komutu akümülatörün içeriğini temizler ve elde bayrağını 0 a reset eder.

"CMP kaydedici" komutu, akümülatörün içeriği ile belirtilen kaydedicinin içeriğini karşılaştırır. Karşılaştırma akümülatörün içeriğinden belirtilen kaydedici veya bellek yerinin içeriğini çıkarmakla yapılır. Bayraklar sonuca göre ayarlanır, fakat kaydedicinin de akümülatörün de içeriği değişmez. Bir CMP B komutu, Şekil 8-12 de gösterildiği gibi bayrakları etkiler. O halde, bu komut, bir sayının akümülatöründeki başka bir sayıdan büyük, küçük veya bu sayıya eşit olup olmadığını belirlemekte kullanılabilir.

Eğer:		Bu durumda (C)	Sıfır (Z)
Akümülatör	= Kaydedici veya derhal veri	0	1
Kaydedici veya derhal veri	> Akümülatör	1	0
Akümülatör	> Kaydedici veya derhal veri	0	0

Şekil 8-12 CMP/CPI komutları ve etkilenen bayraklar.

ARİTMETİK VE MANTIK DOLAYSIZ KOMUTLARI: ADI, ACI, SUI, SBI, ANI, ORI, XRI VE CPI

Bu komutlar, belirtilen işlemi akümülatör ile komutta içerilen veri üzerinde gerçekleştirir. Örneğin ADI 3AH komutu, 3AH sayısını akümülatöre derhal ekler. Bütün bu komutlarda sonuçlar akümülatörde bırakılır. Kodlandığında bu komutların her biri, birincisi komut kodu, ikincisi de veri baytı olmak üzere, iki bayt gerektirir.

Bu gruptaki komutların işlemleri aşağıdaki gibidir.

"ADI bayt", verilen baytı akümülatöre dolaysız ekler.

"ACI bayt", verilen bayt ile eldeyi akümülatöre dolaysız ekler.

"SUI bayt", verilen baytı akümülatörden dolaysız çıkarır.

"SBI bayt", verilen bayt ile elde (ödünç) bayrağını akümülatörden dolaysız çıkarır.

"ANI bayt", verilen bayt ile akümülatörün içeriğini mantık olarak dolaysız VE ler. CY reset edilir.

"ORI bayt", verilen bayt ile akümülatörün içeriğini mantık olarak dolaysız VEYA lar. CY reset edilir.

"XRI bayt", verilen bayt ile akümülatörün içeriğini mantık olarak dolaysız özel-VEYA lar. CY reset edilir.

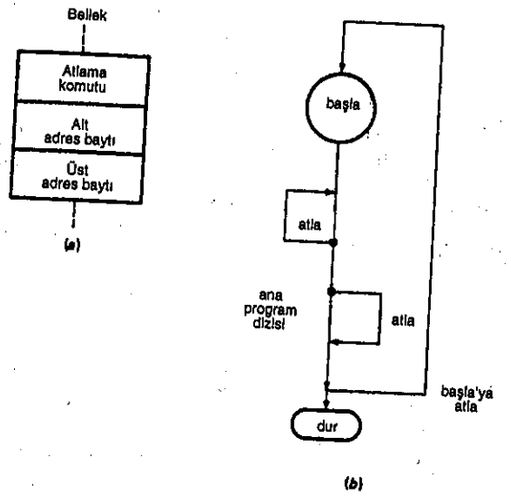
"CDI bayt", verilen baytı akümülatörle dolaysız karşılaştırır. Bayraklar Şekil 8-12 de gösterildiği gibi etkilenir, fakat akümülatörün içeriği değişmez.

DALLANMA VE KONTROL GRUBU KOMUTLARI

Bu gruptaki komutlar, program yürütmedeki normal sırayı değiştirir.

ATLAMALAR: Koşulsuz *atlama komutu JMP (jump)*, Şekil 8-13*a* da gösterildiği gibi, bayt 1 komutun op kodunu, bayt 2 ve 3 te adresini içermek üzere, 3-baytlık bir komuttur. Daha önce bahsedilen diğer komutlarda olduğu gibi, alt mertebeli adres baytı önce yüklenir, üst mertebeli bayt onu izler. JMP komutu yürütülürken, bayt 2 ve 3 teki adres program sayıcıya yüklenir. Böylece, CPU, bir sonraki komutu, belleğin bu yeni adresinden alıp getirecektir. Bundan sonra program, bu atlanılan adresten itibaren sırayla yürütmeye devam eder. Atlama adresi için 2 bayt kullanıldığından, belleğin 65536 yerinden herhangi birine atlamak mümkündür. Bir atlama, Şekil 8-13*b* de gösterildiği gibi, program dizisi içinde ileri veya geri yönde olabilir.

JMP komutunun bir uygulaması, mikroişlemci-tabanlı terazide yapıldığı gibi, bir programı bir sürekli döngü içinde tekrar tekrar döner şekilde tutmaktır. Bunu gerçekleştirmek için, programın son komutu JMP yapılır ve komutun adres baytlarına programın başlangıç adresi konur. Program bu komuta her ulaştığında, bu komut program sayıcıya yeniden başlangıç adresini yükler.



Şekil 8-13 Atlama komutu. ***(a)*** **Bellekte önce adresin alt adres baytı saklanır.** ***(b)*** **Atlama komutlarıyla program akışında oluşturulabilecek değişiklikler.**

JZ (jump if zero: sıfır ise atla) gibi koşullu atlama komutları, sadece belirtilen koşul doğru ise, program sayıcıya verilen atlama adresini yükler. Atlama adresi, daha önce olduğu gibi, bayt 2 ve 3 te bulunmaktadır. JZ komutu yürütülürken, bir atlama sıfır bayrağı ile belirtildiği üzere, eğer akümülatör veya kaydedici baytı 0 ise yapılır. Eğer akümülatör veya kaydedici 0 değilse program bulunduğu yerden normal akışına devam eder.

Diğer koşullu atlama komutları şunlardır:

JNZ *(jump if zero flag is not set) : Sıfır bayrağı set edilmemişse atla.*

JC *(jump if carry flag is-1): Elde bayrağı 1 ise atla.*

JNC *(jump if carry flag is 0)* : Elde bayrağı 0 ise atla.

JPE (jump if parity is even): Parite çift ise atla.

JPO (jump if parity is odd): Parite tek ise atla.

JP (jump if sign is plus): İşaret artı ise atla.

JM (jump if sign is minus): İşaret eksi ise atla.

Tablo 8-1 de atlama komutlarıyla birlikte yazılan PCHL komutu, program sayıcı ile HL kaydedicinin içeriklerini karşılıklı olarak değiştirir. Bu komut, yürütüldüğünde, HL kaydedicisinde saklanan adrese koşulsuz atlamayı gerçekleştirir.

Koşullu atlama komutları, sonuç verilen koşulu sağlayıncaya kadar, programı bir döngü içinde tutmakta kullanılabilir. Örneğin, JNZ komutu, bir işlemin sonucu 0 oluncaya kadar, işlemin başına geri dönüp işlemi tekrar tekrar yapmada kullanılabilir. Bunun örneklerini daha sonra vereceğiz.

ÇAĞIRMA, GERİ DÖNME VE ALTYORDAMLAR: Bir "CALL adres" çağırma komutu, Şekil 8-14a da gösterildiği gibi, program sayıcısını, bellekteki bir altprogram veya yordamın başlangıç adresine atlatmak için kullanılır. Bir altyordamın sonundaki bir RET (return) geri dönme komutu, program yürütmesinin ana diziye geri dönmesini sağlar. Yürütme, ana program dizisindeki CALL komutundan bir sonraki komuta döner.

Bir yordam, örneğin iki sayıyı çarpmak gibi özel bir görevi gerçekleştiren kısa program parçasıdır. Eğer bir yordam, bir programda defalarca kullanılacaksa, belleğe bir defa altyordam olarak yazılabilir ve sonra her gerektiğinde çağrılabilir. Bu yöntem, her seferinde ana programda kullanılması gereken yere yordamın tamamını yazmaya göre çok daha az bellek kullanır.

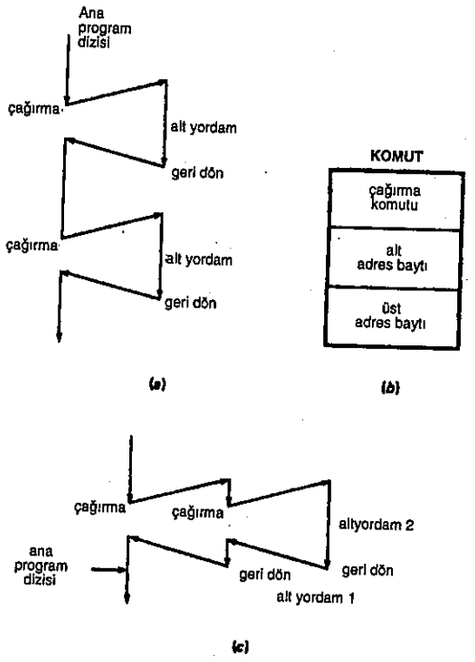
Çağırma komutları, Şekil 8-14b'de gösterildiği gibi, bayt 2 ve 3 te altyordamın başlangıç adresini içerirler. Eğer CALL komutu kullanılmışsa çağırma komutu koşulsuzdur, eğer CNZ, CZ, CNC, CC, CPO, CPE, CP veya CM kullanılmışsa koşulludur. Bu komutlardaki koşulların anlamı, koşullu atlama komutlarındakilerle aynıdır.

Daha önce bir altyordamın sonundaki geri dönme komutunun, program yürütmesini CALL komutundan sonraki komuta aktardığını söylemiştik. Bir altyordam, ana programda herhangi bir yerden çağrılabileceğine ve geri dönme komutu hiç bir adres içermediğine göre, geri dönme komutunun, geri döneceği yeri nasıl belirlediği sorusu ortaya çıkar.

CALL komutu, geri dönüş adresini, RAM'ın yığın denilen özel bir kısmında saklar. Yığının RAM içindeki başlangıç adresine programcı karar verir. Bu adres programın başında bir "LXI SP adres" komutuyla yığın gösterici yazmaca yüklenir. Bir CALL komutu yürütüldüğünde, çağırma komutundan bir sonraki ana program komutunun adresi yığına saklanır, yani itilir.

Şekil 8-15a, bir alt altyordam çağıran bir küçük örnek programdır ve Şekil 8-15b yığın göstericinin içeriğinin bu program tarafından nasıl etkilendiğini göstermektedir. Şekil 8-15a da gösterildiği gibi, yığın gösterici bir LXI SP, 20C2H komutuyla programın başında 20C2H'a ilklendirilmektedir. CALL komutundan sonraki ilk ana program komutunun adresi (geri dönüş adresi)

2054H dir. CALL 20A0H komutu yürütülürken Şekil 8-15*b*'de gösterildiği gibi, birçok adım gerçekleştirilir.



Şekil 8-14 Altyordamlar. ***(a)*** **Altyordamlarla program akışı.** ***(b)*** **Alt adres baytı, üst adres baytından önce saklanmış bir çağırma komutu.** ***(c)*** **İçiçe altyordamlar.**

İlk olarak, yığın gösterici 1 azaltılarak 20C1H yapılır ve geri dönüş adresinin üst baytı bu adrese saklanır (itilir). İkinci olarak, yığın gösterici tekrar 1 azaltılarak 20C0H yapılır ve geri dönüş adresinin alt baytı bu adrese saklanır (itilir). Üçüncü olarak, altyordamın başlangıç adresi 20A0H, program sayıcıya yüklenir. Böylece bir sonraki komut, bu adresten alınıp getirilecektir. Yığın gösterici, yığına itilen en son veri veya adres baytının adresini tutar.

Bir altyordamın sonunda bir RET komutu yürütülürken, mikroişlemci, yığındaki 20C0H dan geri dönüş adresinin alt baytını okur ve programı sayıcıya yükler. Bundan sonra mikroişlemci yığın göstericiyi 20C1H i elde etmek üzere 1 artırır ve geri dönüş adresinin üst baytını okuyarak program sayıcıya yükler. Son olarak yığın göstericiyi 1 daha artırarak orijinal değeri

ADRES	KOD	KOMUT	İŞLENEN	
2000	21	LXI	SP,20C2H	ANA PROGRAM
2001	C2			
2002	20			
	.			
	.	ANA PROGRAM DEVAM EDER.		
	.			
2051	CD	CALL	20A0H	
2052	A0			
2053	20			
2054	D3	OUT	01H	
2055	01			
	.			
	.	ANA PROGRAM DEVAM EDER.		
	.			
20A0	E5	PUSH	PSW	ALT YORDAM
20A1	C5	PUSH	B	
	.			
	.			
	.			
20B0	C1	POP	B	
20B1	F1	POP	PSW	
20B2	C9	RET		

(a)

	YIĞIN GÖSTERİCİ		BELLEK ADRESİ	YIĞIN BELLEK İÇERİĞİ
LXI SP, 20C2H'DEN SONRA	20C2	RETURN'DEN SONRA	20C2	XX
CALL	20C1	RET	20C1	20
CALL 20A0H'DEN SONRA	20C0	POP PSW'DAN SONRA	20C0	54
PUSH PSW	20BF	POP PSW	20BF	(A)
PUSH PSW'DEN SONRA	20BE	POP B'DEN SONRA	20BE	(BAYRAKLAR)
PUSH B	20BD	POP B	20BD	(B)
PUSH B'DEN SONRA	20BC	POP B'DEN ÖNCE	20BC	(C)

Not: (.)'ler, push komutu yürütülmeden önce belirtilen kaydedicinin içeriğini göstermektedir.

(b)

Şekil 8-15 ***(a)*** **CALL, PUSH, POP ve RET komutlarını kullanan program ve altyordam.** ***(b)*** **Altyordam çağırma ve geri dönüşte yığının içerikleri.**

20C2H elde eder. Program sayıcı şimdi ana programdaki bir sonraki komutun adresi 2054H yi içermektedir, böylece yürütme buradan devam eder.

Geri dönmeler koşullu veya koşulsuz olabilir. RET, koşulsuz geri dönme komutudur. Koşullu geri dönme komutları şunlardır:

RZ (return if 0): Sıfır ise geri dön.

RNZ (return if not 0): Sıfır değil ise geri dön

RC (return if carry): Elde varsa geri dön.

RNC (return if not carry): Elde yoksa geri dön.

RPO (return if parity odd): Parite tekse geri dön.

RPE (return if parity even): Parite çiftse geri dön.

RP (return if plus): Artı ise geri dön.

RM (return if minus): Eksi ise geri dön.

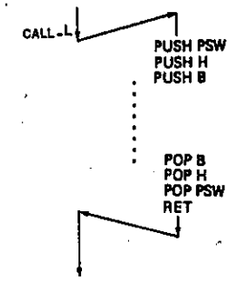
Altyordamlar iç içe yerleştirilebilir. Bu, bir altyordamdaki bir CALL komutu, bir başka altyordamı çağırabilir demektir. Şekil 8-14c, bunun bir diyagramını göstermektedir. Bir makina bir CALL komutuna rastlayıncaya kadar programı yürütmekte ve sonra altyordam 1'e gitmektedir. Bir başka CALL komutuna rastlayıncaya kadar, makina altyordam 1'i yürütür ve sonra altyordam 2'ye gider. Altyordam 2 tamamlandığında, makina altyordam 1'e geri döner. Altyordam 1 tamamlandığında da, makina ana programa geri döner. Her iki altyordamın geri dönüş adresleri, CALL komutları tarafından otomatik olarak sırayla yığına itilmektedirler.

YENİDEN BAŞLAMALAR: Yeniden başlama (restart) RST, komutları özel 1-baytlık koşulsuz çağırma komutlarıdır. Bu komutlar, geri dönüş adresini yığına saklar ve program sayıcıyı bellekteki bir alt seviyeli adrese atlatır. Bu komut, çoğunlukla Bölüm 9 da anlatılan "kesme" lerle kullanılır. Yeniden başlama komutlarıyla atlanan adresler RST 0 için 0000H; RST 1 için 0008H; RST 2 için 0010H; RST 3 için 0018H; RST4 için 0020H; RST 5 için 0028H; RST 6 için 0030H ve RST 7 için 0038 H'tir.

YIĞIN İŞLEMLERİ: Yığın, altyordam çağırmalarında geri dönüş adreslerini saklamak için RAM da ayrılan bir bölgedir. Yığın, ayrıca kaydedicilerin içeriklerinin bir altyordam tarafından değiştirilmesi veya tahrip edilmesi söz konusu olduğunda, bundan kaçınmak üzere, kaydedicilerin içeriklerini saklamak için kullanılabilir. Örneğin, ana programda *B* kaydedicisinde saklanan veri, eğer *B* kaydedicisi altyordamda kullanılırsa, kaybolacaktır. Şekil 8-16 da gösterildiği gibi, kaydedicinin içeriğini saklamak için altyordamın başında bir itme işlemi gerçekleştirilir. Örneğin, PUSH *B* komutu, genişletilmiş *BC* kaydedicisinin içeriğini yığına saklar. PUSH *D*, genişletilmiş *DE* kaydedicisini;

PUSH H, genişletilmiş HL kaydedicisini ve PUSH PSW, akümülatör ve bayrak baytını saklar. *PSW (program status word), program durum sözcük* demektir.

Bir altyordamın sonunda, bir POP komutuyla veri yeniden uygun kaydediciye konur. Bir "POP kaydedici" komutu, saklanmış verileri, yığından geriye alıp belirtilen genişletilmiş kaydediciye kopyalar. Örneğin, POP *B* komutu, yığındaki verileri *B* ve *C* kaydedicilerine koyar. Şekil 8-16 da, verilerin, yığına itilmelerindeki sıranın tersi sırada yığından alınması gerektiğine dikkat edin. Bu tür yığınlara bazan son-giren ilk-çıkar (last-in, first out: LIFO) bellek denir.



Şekil 8-16 PUSH ve POP komutlarının sırasını gösteren altyordam çağırma diyagramı.

PUSH komutu için, yığın gösterici, yığına itilen en son veri baytının adresini izler (tutar). Örneğin bir PUSH B komutu, yığın ve yığın göstericiyi şu şekilde etkiler. Yığın gösterici 1 azaltılır ve *B* kaydedicisinin içeriği yığının o adresine kopyalanır. Yığın gösterici 1 daha azaltılır ve *C* kaydedicisinin içeriği de bu adrese kopyalanır.

Bir POP *B* komutu, daha önce saklanmış olan veri baytını, yığından *C* kaydedicisine kopyalar, yığın göstericiyi 1 artırır, bir sonraki saklanmış olan veri baytını yığından *B* kaydedicisine kopyalar ve yığın göstericiyi 1 artırır. Bir PUSH B, POP *B* dizisinden sonra, yığın göstericinin gösterdiği adres, bu işlemlerden önceki adresle aynıdır.

Şekil 8-15b, tipik bir altyordam çağrılışı sırasında, yığın ve yığın göstericinin, CALL, PUSH, POP ve RET komutlarından nasıl etkilendiğinin bir örneğini göstermektedir. CALL komutu yığın, göstericiyi 20C1H ye azaltır ve geri dönüş adresinin üst baytını yığına iter. Sonra, yığın göstericisini 20C0H'ye azaltır ve geri dönüş adresinin alt baytını yığına iter. PUSH PSW komutu, yığın göstericiyi 20BFH ye azaltır, akümülatörün içeriğini yığına saklar,

yığın göstericiyi tekrar azaltır (20BEH olur) ve bayrak baytını yığına saklar. PUSH *B* komutu, yığın göstericiyi 20BDH'ye azaltır, *B* kaydedicisinin içeriğini saklar, yığın göstericiyi tekrar azaltır (20BCH olur) ve *C* kaydedicisinin içeriğini yığına saklar.

POP ve RET komutları yordamı ters çevirir. POP *B*, 20BCH'daki yığın içeriğini C kaydedicisine kopyalar ve yığın göstericiyi 20BDH ye artırır. Sonra, 20BDH daki yığın içeriğini *B* kaydedicisine kopyalar ve yığın göstericiyi 20BEH ye artırır. POP PSW komutu, bayrakları yerine koyar, yığın göstericiyi 20BFH ye artırır, akümülatörün içeriğini yığından alıp yerine koyar ve yığın göstericiyi 20C0H ye artırır. RET komutu, geri dönüş adresinin alt baytını program sayıcıya yükler, yığın göstericiyi 20C1H ye artırır, geri dönüş adresinin üst baytını program sayıcıya yükler ve yığın göstericiyi 20C2H ye artırır. Şimdi yığın gösterici, altyordam çağrılışından önceki adresle aynı adresi içermektedir.

Eğer, bir programda POP komutlarının sayısı, PUSH komutlarının sayısıyla aynı tutulmazsa, yığın dengesini kaybeder. Bir RET komutu, aslında yazmaç içeriği olan verileri bir geri dönüş adresi olarak alıp kullanmaya kalkabilir.

Yığın, bellekte başlangıç adresinden aşağıya doğru büyüdüğü için, başka saklanan verilerle veya programlarla karışmaması için genellikle RAM da bir üst adrese yerleştirilir.

XTHL komutu, yığının üstündeki 2 veri baytını *H* ve *L* kaydedicilerindeki 2 baytla karşılıklı olarak yer değiştirir. SPHL komutu, *H* ve *L* kaydedicilerinin içeriğini yığın gösterici kaydedicisine kopyalar.

GİRİŞ/ÇIKIŞ: "OUT bayt" komutu, akümülatördeki veri baytını, bu komutun ikinci baytında adresi içerilen çıkış portuna taşıyan 2-baytlık bir komuttur. Port adresi 8 bit olduğundan, 8080A/8085A mikroişlemcileri 256 taneye kadar çıkış portunu adresleyebilir.

"IN bayt" komutu, bir veri baytını *giriş portundan akümülatöre taşır.* Komutun ikinci baytında, 256 mümkün giriş port adresinden biri içerilir.

KONTROL: Kontrol komutları *DI* kesmeyi *yetkisiz kılma (disable interrupt)*ve EI kesmeyi yetkili kılma (enable interrupt) komutlarıdır. Bunlar kesmelerle birlikte Bölüm 9 da anlatılacaktır.

*NOP "işlem yok" (no operation)*komutu yürütülürken, işlemci, program sayıcıyı artırmaktan başka bir şey yapmaz. NOP komutları, genellikle zamanlama amacıyla programlarda kullanılırlar. Ayrıca, bazen unutulan veya henüz yazılmamış bazı komutların yerine NOP komutları yazılarak ve sonradan bütün programdaki komutları kaydırmadan, düzeltme, NOP'ları çıkarıp yerine esas komutları koyarak yapılır.

HLT bir *durma (halt)* komutu olup mikroişlemciyi durdurur. Bu komutun ayrıntılarını Bölüm 9 da inceleyeceğiz.

RIM ve SIM komutları da 8085A nın kesmeleriyle birlikte Bölüm 9'da anlatılmaktadır.

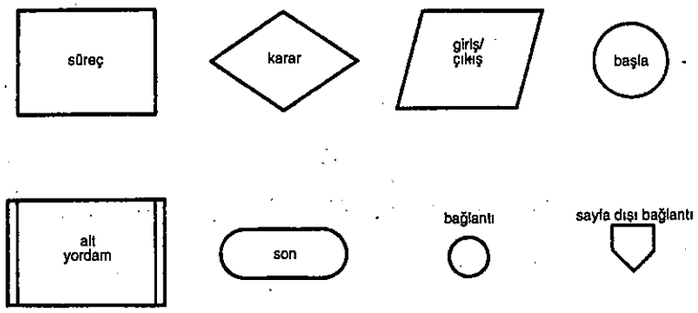
Böylece 8080A/8085A çeviri dilinin sözlüğü ile tanışmış oldunuz. Şimdi bu sözlüğü kullanarak nasıl basit programlar yazacağınızı göstereceğiz.

ÇEVİRİ DİLİNDE PROGRAM YAZMA

AKIŞTABLOLARI

Belli bir işlem için gerçek çeviri dili programı yazmadan önce, birçok programcı, programı akıştablosu denen şematik bir şekilde göstermektedir. Şekil 8-17, bazı standart akıştablosu sembollerini göstermektedir. Bu sembollerin hepsini içeren hazır şablonlar vardır.

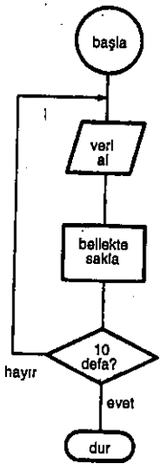
Şekil 8-18, bir giriş portundan 10 veri örneği alıp bellekte saklayan bir programın akıştablosunu göstermektedir. İşlemler dikdörtgen kutularda



Şekil 8-17 Akış tablosu sembolleri

gösterilir. Bir işlem bloğu gerçekte birçok komutu temsil edebilir. Karar noktaları baklava şeklindeki kutularla gösterilir. Akıştablosundan bu programın bir veri baytı alıp bellekte saklandığını ve sonra bu işlemin 10 defa yapılıp yapılmadığını kontrol ettiğini görebilirsiniz. Eğer 10 defa yapılmadıysa, makina geri dönüp yeni bir veri baytı alır ve onu belleğe saklar. Bu işlemler alınan ve bellekte saklanan örnek sayısı 10 oluncaya kadar yapılır.

Doğru yazılmış bir akış tablosu, belli bir makinanın çeviri dilinden hiç bahsetmez. Bu nedenle, bir akıştablosu, herhangi bir mikroişlemci için program yazmakta kullanılabilir. Akıştabloları, büyük programların küçük parçalara, yani modüllere, bölünerek üzerinde kolayca çalışılmasını sağlar.



Şekil 8-18 10 veri örneği alıp bellekte saklayan programın akıştablosu.

Bazı deneyimli programcılar, akıştablosunu, program yazılmadan önce çizmek yerine program tamamlandıktan sonra programı açıklama amacıyla çizerler.

ÇEVİRİ DİLİNİN DİLBİLGİSİ KURALLARI

Çeviri dili programları, bir çevirici program ile makina diline çevrilebilmeleri için genellikle standart bir şekilde yazılır. Bu standart şekilde, çeviri dili cümlelerinin dört alanı vardır. Bunlar Şekil 8-19 da gösterilmiştir.

İlk alan etiket alanıdır. Bir etiket, bilinmeyen bir adresi temsil etmekte kullanılan bir sembol veya semboller grubudur. Bazı çevirici programlar, etiketlerin 5 veya 6 karakterden uzun olmamasını ve ilk karakterin bir harf olmasını gerektirirler. Etiket alanı çoğunlukla bir "iki nokta üstüste" ile sonlandırılır. Etiketlerin kullanışlarını daha sonra anlatacağız.

Etiket Alanı	OP Kodu Alanı	İşlenen Alanı	Açıklama Alanı
LOOP:	IN	00H	; port 0'dan veri gir

Şekil 8-19 Doğru çeviri dili program satırının dört alanı

Op kodu alanı, gerçekleştirilecek komutun bellenirini içerir. Bellenirlere bazen işlem kodu veya op kod denir. 8080A /8085A için bu op kodlar aynen Tablo 8-1 de gösterildikleri gibi yazılmalıdır.

Op kod alanının arkasından işlenen alanı gelir. Bu alan, komutun gerçekleştirileceği kaydedicileri veriyi veya adresi içerir. Kaydedici başharfleri arasında veya bir kaydedici başharfiyle veri baytı arasında bir virgül olması gerekmektedir. MOV *B,A* komutunun, akümülatörü *B* kaydedici kopyaladı-ğını hatırlayalım.

Herhangi bir çeviri dili programının çok önemli bir kısmı açıklama alanıdır. Çoğu çeviriciler için, açıklama alanı bir noktalı virgülle veya bir yıldızla başlar. Açıklamalar, makina dili programın bir parçası olmaz. Bunlar, size, programcıya ve sizden sonra geleceklere yardımcı olması için yapılır. Eğer açıklamaları olmayan bir program yazarsanız ve 6 ay bir kenarda bırakırsanız, ona tekrar baktığınızda programı anlamakta çok zorluk çekebilirsiniz. Hatta programı bir başkasının yazmış olduğuna inanabilirsiniz.

Açıklamalar, her komutun veya komut grubunun ne yaptığını ayrıntılı olarak açıklayacak şekilde yazılmalıdır. Açıklamalar sadece belleniri açıklamamalı, ilgili yordam içinde komutun işlevini de tanımlamalıdır.En iyi sonuçlar için açıklamaları, başta programın amacı hakkında hiç bir şey bilmeyen birisine açıklıyormuş gibi yazın.

Her komuta veya komut grubuna ait açıklamaların yanı sıra, her program veya altyordam, programın ne yaptığını tanımlayan bir dizi açıklamalarla başlamalıdır. Başlangıç açıklamaları ayrıca, kullanılan parametre, kaydedici ve bellek yerlerinin bir listesini içermelidir. Bir program için açıklamalar yazmanın hemen hemen sonu yoktur. Şekil 8-20 oldukça iyi açıklanmış basit bir çeviri dili programı göstermektedir.

ÇEVİRİ DİLİ PROGRAMLAMA ÖRNEKLERİ

Bu kısımda çeviri dili programların nasıl yazıldığını örneklerle göstereceğiz. İlk örnek Şekil 8-18 deki akıştablosunun programıdır.

ÖRNEK 1: VERİYİ ALIP SAKLAYAN PROGRAM Şekil 8-20, 00H portundan 10 örnek alıp, bunları, ardarda gelen bellek yerlerine saklayan bir çeviri dili programı göstermektedir. Başlık, kullanılan yazmaçlar ve çağrılan altyordamların belirtilişine dikkat edin.

Program en iyi orta kısmından başlayarak açıklanabilir, çünkü akış tablosuyla ilgili ilk komut *IN* komutu olarak çözülmektedir. Bu komut, basitçe port 00H dan bir veri örneği alır ve akümülatöre saklar. Bir sonraki komut MOV *M,A* akümülatörün içeriğini, *HL* kaydedici tarafından gösterilen bellek yerine kopyalar. Bu nedenle, *HL* kaydedici çifti, veri örneklerini saklamaya başlamak istediğiniz bellek adresine, daha önce ilklendirilmiş olmalıdır. Bu, programın başında, H kaydedicisine 20H ve *L* kaydedicisine 50H yükleyen LXI *H*, 2050H komutuyla yapılmaktadır. İlk örnek bellekte saklandıktan sonra, genişletilmiş *HL* kaydedici çiftinin bellekteki bir sonraki saklama yerini göstermesi için INX *H* komutu *HL* yi artırır.

PROGRAMCI _____ SAYFA _____

PROGRAMIN ADI _____ İŞ NO. _____

ADRES	KOMUT	ETİKET	BELLENİR	İŞLENEN	AÇIKLAMALAR
2000	21		LXI	H,2050H	;H +L YAZMAÇLARINI RAM İÇİNDEKİ BİRİNCİ
2001	50				; DEPOLAMA ADRESİNİ AYARLA
2002	20				
2003	0E		MVI	C,0AH	; SAYICI KAYDEDİCİSİNİ 10 (ON)'A AYARLA
2004	0A				
2005	DB	LOOP:	IN	00H	; VERİRİ ÖRNEĞİNİ PORT 0'DAN AL
2006	00				
2007	77		MOV	M,A	; VERİ ÖRNEĞİNİ RAM DEPOYA SAKLA
2008	23		INX	H	;H+L'Yİ BİR SONRAKİ ADRESİ GÖSTERMEK İÇİN ARTIR
2009	0D		DCR	C	; SAYICIYI BİR AZALT
200A	C2		JNZ	LOOP	; SAYICI KAYDEDİCİ SIFIR DEĞİLSE LOOP'A ATLA
200B	05				
200C	20				
200D	76		HLT		; DUR

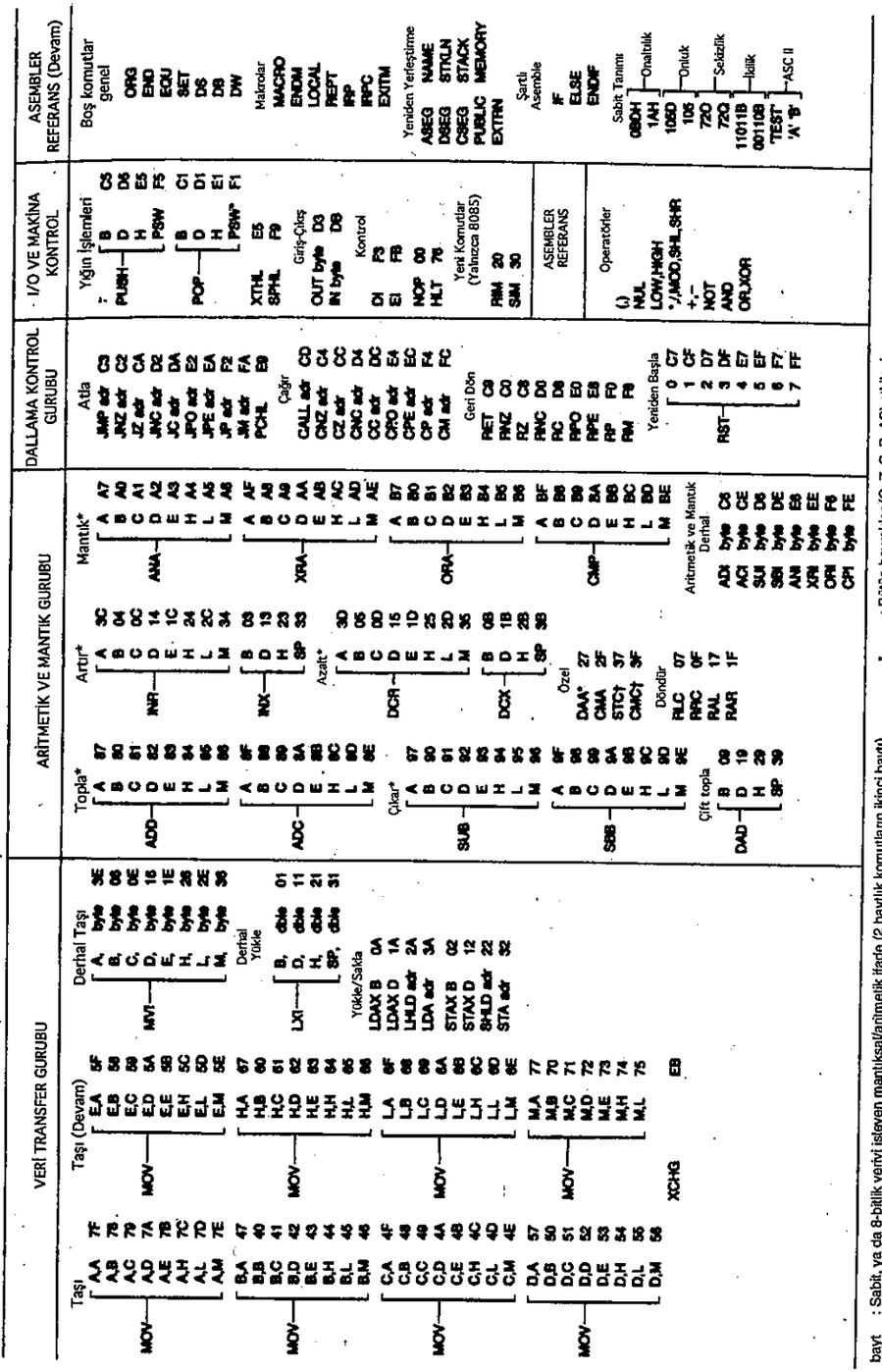
AÇIKLAMALAR: PORT 0'DAN 10 VE ÖRNEĞİ ALIP ONLARI 2050 H ADRESİNDEN BAŞLAYARAK RAM'A SAKLAYAN PROGRAM. BU PROGRAM 8080 A, 8085 A VEYA Z-80 İŞLEMCİ SİSTEMLERİNDE ÇALIŞTIRILMAK İÇİN YAZILMIŞTIR.

KULLANILAN KAYDEDİCİLER: A, C, H, L,

ÇAĞRILAN YORDAMLAR: YOK

Şekil 8-20 Şekil 8-18 deki akıştablosunun çeviri dili programı

TABLO 8.1 8080A/8085A KOMUT SETİ (INTELCORPORATION)



bayt : Sabit, ya da 8-bitlik veriyi işleyen mantıksal/aritmetik ifade (2 baytlık komutların ikinci baytı)
dble : Sabit, ya da 16-bitlik veriyi işleyen mantıksal/aritmetik ifade (3 baytlık komutların ikinci ve üçüncü baytları).
adr : 16-bitlik adres (3 baytlık komutların ikinci ve üçüncü baytları).

* : Bütün bayraklar (C, Z, S, P, AC) etkilenir.
** : ELDE (CARRY) dışındaki bütün bayraklar etkilenir (INX ve DCX hiç bir bayrağı etkilemez).
† : Yalnızca ELDE etkilenir.

Bütün belleniler Intel Corporation, tarafından 1976 da copyright altına alınmıştır.

Bundan sonra yaklaşılması gereken problem, 10 tane örneğin ne zaman alındığını belirlemektir. Bir yöntem, bir kaydediciyi başlangıçta 0 yapmak ve her örnekten sonra kaydediciyi 1 artırmaktır. Kaydedicideki sayıyı 10 (0AH) ile karşılaştırarak 0 sonucunu algıladığınızda 10 tane örnek almış olduğunuzu söyleyebilirsiniz. Daha kolay bir yöntem, başlangıçta kaydediciye 10 (0AH) koymak ve her örnek saklandıktan sonra kaydediciyi 1 azaltmaktır. Eğer azaltma sonucu kaydedicide 0 kalırsa, DCR komutu sıfır bayrağını doğrudan set etmektedir, böylece karşılaştırma komutuna gerek kalmaz.

Bu program ikinci yöntemi kullanmaktadır. Programın başında C kaydedici, MVI C,0AH komutuyla bir sayıcı olarak hazırlanır. Bu komut C kaydedicisine derhal 0AH yazar. Her örnek saklandıktan sonra, DCR C komutu C kaydedicisindeki sayıyı 1 azaltır. Eğer azaltma sonucu C kaydedicisinde 0 kalırsa, bu takdirde sıfır bayrak flip-flopu 1 e set olur.

JNZ komutu sıfır bayrak flip-flopunu inceler. Eğer bayrak set edilmemişse, bu komut, program sayıcıyı bu komutun sonunda içerilen adrese atlatacaktır. Programı yazarken bu adresin ne olacağını bilemeyebileceğiniz için,bu adresi *"etiket" (label)* denilen sembolik bir adresle gösterirsiniz. Bir etiket için ismi siz seçersiniz, fakat op kodlara benzer harf kodları kullanmaktan kaçınmalısınız. Çoğu çeviriciler etiketlerin bir harfle başlamasını ve 5 veya 6 karakterden fazla olmamasını isterler. Program çevrilirken yani makina diline dönüştürülürken, etiketler adreslerle değiştirilir.

Etiket alanındaki etiket, atlanacak yeri belirtmekte kullanılır. Şekil 8-20 deki program için, eğer azaltma sonucunda C kaydedicisi sıfır değilse, program yürütme IN 00H komutuna atlar. Program çalıştırıldığında, C kaydedicisini azaltma, 0 sonucunu verinceye kadar program bu döngü içinde döner durur. Bir 0 sonucu algılandığında, program HLT komutuna devam eder ve durur.

Bu programı anlayıp anlamadığınızı belirlemek için döngü etiketinin yanlışlıkla MVI C, 0AH komutunun önüne konduğunda ne olacağını bulmaya çalışın. Diğer bir deyişle, eğer JNZ komutu, program sayıcıyı MVI C, 0AH komutunun adresine atlatırsa ne olur?

Bunun cevabı programın hiçbir zaman durmayacağı şeklinde olacaktır, çünkü C kaydedicisi hiç bir zaman 0 olmayacaktır. Döngü içinde her seferinde MVI C, 0AH komutuyla C kaydedicisine tekrar 10 (0AH) yüklenecektir. Program veri örneğinden sonra HL yi artırmaya devam edecek ve bütün bellek port 00H'den alınan örneklerle dolacaktır. Bu tür hatalarla oldukça sık karşılaşılır.

Şekil 8-20, ayrıca her komutun kodlarının, ardarda gelen bellek yerlerine nasıl girildiğini göstermektedir. Bu durumda, başlangıç adresi 2000H seçilmiştir. LXI *H*, 2050H komutundaki baytların sırasına dikkat edin. Ayrıca, program kodlandığında, JNZ LOOP komutundaki LOOP etiketinin gerekli atlama adresinin 2005H ile değiştirildiğine dikkat edin. Atlama adresinin baytları, tabii ki, önce alt bayt sonra üst bayt olmak üzere yüklenmektedir. Gösterilen özel programlama kağıdı, programın bütün parçalarının düzenli olmasına ve hataların azalmasına yardımcı olmaktadır.

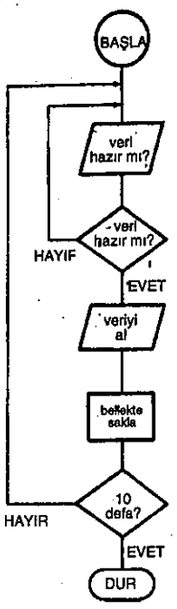
ÖRNEK 2: VERİ HAZIR DARBESİ İLE VERİYİ ALIP SAKLAYAN PROGRAM : Şekil 8-20'deki program çalıştırıldığında, makina port 0'dan 10 veri örneğini bütün hızıyla alır, belleğe saklar ve işletmen bundan sonra ne yapacağını söyleyinceye kadar beklemede kalır. Yukarıdaki programa basit bir ilave, makina 10 veri örneğinin her birini port 0'dan almadan önce, makinanın port 1'in en küçük değerlikli bitinden bir darbe gelmesini beklemesini sağlar.

Şekil 8-21, bu programın akıştablosunu ve çeviri dili programını göstermektedir. HL kaydedicisindeki bellek gösterici ve C kaydedicisindeki örnek sayıcı ilklendirildikten sonra, port 1 den veri alınır. ANI 01H komutu bu veriyi 01H ile VE ler. Eğer port 1 deki verinin en küçük değerlikli biti 0 ise, VE leme sonucu akümülatörde 0 olur. JZ komutu, programı LOOP a geri atlatacaktır. Eğer port 1'deki verinin en küçük değerlikli biti 1 ise, ANI 01H komutu sonunda akümülatörde 01H olur. Bu, 0 olmadığına göre, program IN 00H komutuna devam eder. Port 0 dan alınan veri baytı bellekte saklanır, bellek gösterici *HL* bir artırılır, sayıcı kaydedici C bir azaltılır ve 10 örneğin saklanıp saklanmadığı kontrol edilir. Eğer henüz 10 tane olmamışsa port 1 in en küçük değerlikli bitinde bir pozitif darbe olup olmadığına bakmak üzere, program LOOP a geri atlar. Port 0 da verinin hazır olduğunu gösteren bu darbe varsa, yeni bir örnek alınır.

Port 1 deki veride, veri hazır darbesinin varlığını anlamanın diğer bir yolu, akümülatörün en küçük bitini elde flip-flopuna almak için RRC veya RAR komutunu kullanmaktır. Eğer port 1 in en küçük değerlikli bitinde bir 1 varsa, sağa döndürmeden sonra elde bayrağı flip-flopu set olacaktır. Elde bayrağı set oluncaya kadar programı döngü içinde tutan bir JNC komutu kullanılabilir. Bu, basit bir şekilde, programcı sayısı kadar program yazmanın değişik yolları olduğunu göstermektedir.

ALTYORDAMLARIN KULLANILIŞI

ÖRNEK 3: 1ms GECİKTİRME ALTYORDAMI OLAN PROGRAM Şekil 8-20'deki temel veriyi-al-ve-sakla programının biraz değiştirilmiş şekli altyordamların kullanımını göstermektedir. Son örnekte yapıldığı gibi, port 1in en küçük değerlikli bitinde bir veri hazır darbesi aramak yerine, 1ms aralıklarla 10 örnek almak istediğinizi kabul edelim. Şekil 8-22a, bunu yapacak bir programın akıştablosunu göstermektedir. 1 ms-lik geciktirme kutusunun kenarlarındaki çift çizgiye dikkat edin. Bu, programın geciktirme kısmının altyordam olarak yapıldığını belirtir.



(a)

Programcı					SAYFA
Programın Adı					İŞ NUMARASI
ADR	Komut	Etiket	Bellenir	İşlenen	Açıklamalar
2000	21		LXI	H2050H	H+L'yi birinci depolama adresini gösterecek şekilde ayarla
2001	50				
2002	20				
2003	0E		MVI	C,0AH	; Sayıcı kaydediciyi 10'a ayarla
2004	0A				
2005	DB	LOOP:	IN	01	; Veri hazır sinyalini al
2006	01				
2007	E6		ANI	01H	; En küçük değerlikli bit 1 için kontrol et
2008	01				
2009	CA		JZ	LOOP	; En küçük değerlikli bitte bir 1 için bekle
200A	05				
200B	20				
200C	DB		IN	00	; Veriyi Port 0'dan al
200D	00				; H+L Kaydedici ile gösterilen bellek yerine sakla
200E	77		MOV	M,A	
200F	23		INX	H	; Bir sonraki adresi gös. için H+L'yi bir artır
2010	0D		DCR	C	; Sayıcıyı bir azalt
2011	C2		JNZ	LOOP	; 10 veri alınmamışsa Loop'a atla
2012	05				
2013	20				
2014	76		HLT		BİTTİYSE OLUR

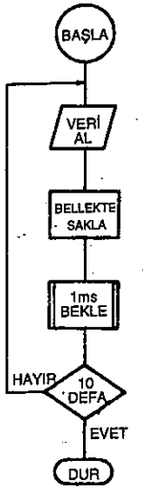
AÇIKLAMALAR: Bu program, PORT I 'in LSB'sinde "Veri Hazır" darbesi olduğunda PORT 0'dan bir veri örneği alır. Tolam 10 örnek alınır. Örnekler, bellekte 2050H adresinden itibaren saklanır.
Kullanılan kaydediciler : A, C, H, L
Çağrılan yordamlar: Yok

(b)

Şekil 8-21 veri hazır darbesi var olduğunda 10 örnek alıp belleğe saklayan program. *(a)* Akıştablosu. *(b)* Çeviri dili program.

Şekil 8-22b, bu örneğin çeviri dili programını göstermektedir. C kaydedicisi 0AH ile yüklenir, *HL* bellek gösterici, veri saklamanın başlangıç adresine ayarlanır ve yığın gösterici kaydedici, yığının başlangıç adresine ayarlanır. Ne zaman bir program bir altyordam kullanacaksa, yığın gösterici ilklendirilmektedir, çünkü çağırma komutları geri dönüş adreslerini yığına saklar. Yığına genellikle var olan RAM'ın üst kısımlarında bir adres atanır, çünkü adres ve veriler itildikçe, yığın bu adresten aşağıya doğru büyür.

İlklendirme basamaklarından sonra, port 0 dan bir örnek alınıp belleğe saklanmaktadır. *HL* bellek gösterici kaydedici, bir sonraki saklama yerini göstermek üzere 1 artırılır. Sayıcı olarak kullanılan C kaydedicisi 1 azaltılır ve 1 ms lik geciktirme altyordamı DELAY çağrılır. CALL komutu, geri dönüş adresini yığına iter, yığın göstericiyi 2 azaltır ve program sayıcıyı DELAY etiketi



PROGRAMCI					SAYFA
PROGRAMIN ADI					İŞ NUMARASI
ADRES	KOMUT	ETİKET	BELENİR	İŞLENER	AÇIKLAMALAR
200 0	21		LXI	H,2050H	H+L'Yİ BİRİNCİ DEPOLAMA
200 1	50				ADRESİYLE YÜKLE
200 2	20				
200 3	0E		MVI	C, 0AH	ÖRNEK SAYICIYI 10 (ON)'A AYARLA
200 4	0A				
200 5	31		LXI	SP, 2062H	YIĞIN SAYICIYI 2062'YE AYARLA
200 6	62				
200 7	20				
200 8	DB	LOOP:	IN	00	VERİ ÖRNEĞİNİ PORT 0'DAN AL
200 9	0				
200 A	77		MOV	M,A	ÖRNEĞİ BELLEĞE SAKLA
200 B	23		INX	H	H+L'Yİ BİR SONRAKİ ADRESİ GÖS. İÇİN ARTIR
200 C	0D		DCR	C	SAYICIYI BİR AZALT
200 D	CD		CALL	DELAY	1 mS BEKLE
200 E	25				
200 F	20				
201 0	C2		JNR	LOOP	SAYICI SIFIR DEĞİLSE BİR BAŞKA
201 1	8				ÖRNEK AL
201 2	20				
201 3	76		HLT		DUR
*******	DELAY YORDAMI		BU YORDAM 1MS	GECİKMESİ YARAT	GİRİŞLER: YOK / ÇIKIŞLAR : yok — KULLANILAN KAYDEDİCİLER A,B,S,P
202 5	F5	DELAY:	PUSH	PSW	AKÜMÜLATÖR İÇERİĞİNİ VE BAYRAKLARI SAKLA
202 6	C5		PUSH	B	B+C KAYDEDİCİ İÇERİKLERİNİ SAKLA
202 7	6		MVI	B, 8H	GECİKME SAYICI KAYDEDİCİSİNİ YÜKLE
202 8	81				
202 9	5	CNTDN:	DCR	B	GECİKME SAYICISINI AZALT
202 A	C2		JNZ	CNTDN	B= 0 OLUNCAYA KADAR DEVAM ET
202 B	29				
202 C	20				
202 D	C1		POP	B	B+C KAYDEDİCİ İÇERİKLERİNİ YENİDEN YÜKLE
202 E	F1		POP	PSW	AKÜMÜLATÖR VE BAYRAKLARI YENİDEN
202 F	C9		RET		YÜKLE ANA PROGRAMA DÖN

AÇIKLAMALAR: BU PROGRAM YAKLAŞIK 1 ms'de 10 VERİYİ PORT 0'DAN ALIR VE HER ÖRNEĞİ 2050H'DAN BAŞLAYAN BELLEK YERLERİNE SIRASIYLA DEPOLAR.
KULLANILAN KAYDEDİCİLER : A,C, H,L,SP
ÇAĞRILAN YORDAM : DELAY

Şekil 8-22 1 ms aralıklarla 10 veri örneği alıp bellekte saklayan program. *(a)* Akıştablosu. *(b)* Çeviri dili program.

ile temsil edilen altyordam adresi ile yükler. Bundan sonra, program yürütme, altyordam adresindeki ilk komutla devam eder.

Herhangi bir altyordamdaki ilk adım, altyordam tarafından kaybedilebilecek veya üzerine yazılabilecek kaydedicilerin içeriklerini ve bayrakları saklamak olmalıdır. PUSH PSW, akümülatör içeriğini ve bayrakları yığında saklar ve yığın göstericiyi 2 azaltır. PUSH *B* komutu, *B* ve *C* kaydedicilerini yığında saklar ve yığın göstericiyi 2 azaltır. Altyordamda kullanılan bütün kaydedicileri yığına itmek iyi bir fikirdir.

Altyordamın sonundaki POP *B*, *B* ve *C* kaydedicilerine yığında saklanmış verilerini geri yükler, POP PSW akümülatör ve bayrakları gerisin geriye yükleyerek ana programa dönmeye hazırlanılır. Veriler yığına itildikleri sıranın tersine yığından geri alınır. Örneğin, bir altyordamın başındaki PUSH PSW, PUSH *B* ve PUSH *D*, altyordamın sonunda POP D, POP B ve POP PSW gerektirir.

Geciktirme yordamın kalbi MVI B, 81H; DCR B; ve JNZ CNTDN komutlarıdır. MVI B, 81H komutu, *B* kaydedicisine 81H yükler. DCR *B*, sayıyı 1 azaltır ve JNZ, sayının 0 olup olmadığını kontrol eder. Eğer sayı 0 değilse, program küçük döngü üzerindeki CNTDN ye atlar ve *B* kaydedicisi tekrar 1 azaltılır. Gecikme, makinanın program döngüsünü 81H, yani 129, defa yürütmesi için geçen zamanla oluşur. Gelecek kısımda, 8080A veya 8085A için gecikme sabitlerinin nasıl hesaplanacağını göreceğiz.

Gecikme döngü sayıcısı 0 a azaltıldığında, program *B* yi POP eder ve yığın göstericiyi 2 artırır. Sonra POP PSW yapılır ve yığın gösterici tekrar 2 artırılır. Şimdi yığın gösterici kaydedici, geri dönüş adresinin saklandığı bellek yerini göstermektedir. RET komutu, bu adresi program sayıcıya yükler ve yığın göstericiyi 2 artırarak başlangıçtaki değerine ulaştırır. PUSH işlemi sayısı kadar POP işlemi sayısı olması çok önemlidir, aksi halde yığın dengesiz hale gelir.

Ana program dizisine JNZ LOOP komutuyla devam edilir. Bu komut, DCR C komutunun sıfır bayrağını set edip etmediğini kontrol etmektedir. Eğer bayraklar altyordamdaki PUSH PSW ve POP PSW komutlarıyla konumlanmış olsalardı, bu komut doğru bayrakları incelemiyecekti. JNZ LOOP komutu, programın geri dönerek 1 ms aralıklı toplam 10 örnekten bir sonrakini almasına sebep olur.

Deneyimli bir programcı, programla ilgili olarak herhalde en az iki gözlemde bulunacaktı. Birincisi DCR C komutu, bayrakları saklamak gereğinden kaçınmak için CALL DELAY komutundan sonraya konulabilirdi. İkincisi, gecikme döngüsünün üç aktif komutunun (MVI B, 81H; DCR *B*; ve JNZ CNTDN), altyordam kullanmaktan tamamen kaçınmak üzere ana program dizisinin bir parçası olarak yazılabileceğidir. Bu küçük program için, bu yo-

rumlar çok doğrudur. Ancak; uzun uygulama programlarında, 1 ms'lik gecikmeyi altyordam olarak yazmak, bu yordamın programın herhangi bir yerinden çağrılabileceği demektir. Programda sadece bir defa yazılması gerekir. Çarpma veya kod çevrimi gibi bir programda defalarca çağrılabilecek uzun yordamlar için bu büyük bir avantajdır. Bir altyordamın programın herhangi bir yerinde kullanılabilmesini garanti etmek için, en emin yol, altyordamın, kullandığı kaydedicileri ve PSW yu yığına ittirmesidir.

Çoğu programcılar, buldukları veya yazdıkları faydalı altyordamların bir dosyasını tutar. Bunları temelde modül yani altyordamların birbirine eklenmesiyle oluşan uygulama programlarını yazmakta kullanırlar. Her altyordamın, gerçekleştirdiği işlevleri tanıtan bir başlığı, kullandığı kaydediciler ve çağrılan diğer altyordamları bildiren açıklamaları olmalıdır.

Geciktirme altyordamları, birçok programın önemli bir parçası olduğundan, şimdi istenilen gecikme miktarı için sabitin nasıl hesaplandığını göstereceğiz.

PROGRAM YÜRÜTME ZAMANI VE GECİKTİRME SABİTLERİ

GECİKTİRME SABİTLERİNİN HESAPLANMASI: Bir mikroişlemcinin kalp atışları, onun saatidir. Her komutu yürütmek için belirli sayıda kalp atışı (saat çevrimi) gerekir. Eğer her komutun kaç saat çevrimi olduğunu ve bir saat çevriminin süresini biliyorsanız, her komutu yürütmek için gerekli zamanı hesaplayabilirsiniz. Bir saat çevrimine aynı zamanda bir durum denir. Tablo 8-2, 8080A ve 8085A komutlarının herbiri için saat çevrimi yani durum sayısını göstermektedir.

8080A da çok kullanılan bir saat frekansı 2MHz dir. Bu frekans için, bir çevrimin süresi 1/(2MHz), yani 500 ns'dir. Durum sayısı, bir durumun süresiyle çarpıldığında ilgili komutun yürütülmesi için toplam süreyi verir. Örneğin, Tablo 8-2, 8080A için CALL komutunun 17 durum gerektirdiğini belirtmektedir. 17 durum defa 0,5 µs/durum, CALL komutu için 8,5 µs verir.

Şimdi; bir komutun yürütme süresinin hesaplanması açıklanmış olduğuna göre, 1ms'lik geciktirme döngüsünün zamanlaması anlatılabilir. Şekil 8-23 ilgili komutları ve herbirinin durum sayısını göstermektedir. Defalarca tekrarlanan DCR B ve JNZ komutları dışındaki komutların toplam durum sayısı 59 dur. Her durum için 0,5 µs den bu 29,5 µs eder. 1000 µs nin geri kalanı DCR B ve JNZ defalarca tekrarlanarak tamamlanır. CNTDN döngüsü 1000 µs-29,5 µs yani 970,5 µs olmalıdır. Her CNTDN döngüsü 15 durum, yani 7,5 µs'dir. 970,5 µs'yi 7,5 µs'ye bölerek döngünün 129 defa tekrarlanması gerektiği bulunur. Bu, onaltılıkta 81 olduğundan, başlangıçta B kaydedicisine 81H yüklenmektedir.

Komut		Kod	Baytlar	T Durumları 8085A	T Durumları 8080A	Makina Çevrimi	Komut		Kod	Baytlar	T Durumları 8085A	T Durumları 8080A	Makina Çevrimi
ACI	DATA	CE data	2	7	7	F R	LXI	RP,DATA16	00RP 0001 data16	3	10	10	F R R
ADC	REG	1000 1SSS	1	4	4	F	MOV	REG,REG	01DD DSSS	1	4	5	F*
ADC	M	8E	1	7	7	F R	MOV	M,REG	0111 0SSS	1	7	7	F W
ADD	REG	1000 0SSS	1	4	4	F	MOV	REG,M	01DD D110	1	7	7	F R
ADD	M	86	1	7	7	F R	MVI	REG,DATA	00DD D110 data	2	7	7	F R
ADI	DATA	C6 data	2	7	7	F R	MVI	M,DATA	36 data	2	10	10	F R W
ANA	REG	1010 0SSS	1	4	4	F	NOP		00	1	4	4	F
ANA	M	A6	1	7	7	F R	ORA	REG	1011 0SSS	1	4	4	F
ANI	DATA	E6 data	2	7	7	F R	ORA	M	B6	1	7	7	F R
CALL	LABEL	CD addr	3	18	17	S R R W W*	ORI	DATA	F6 data	2	7	7	F R
CC	LABEL	DC addr	3	9/18	11/17	S R -/S R R W W*	OUT	PORT	D3 data	2	10	10	F R O
CM	LABEL	FC addr	3	9/18	11/17	S R -/S R R W W*	PCHL		E9	1	6	5	S*
CMA		2F	1	4	4	F	POP	RP	11RP 0001	1	10	10	F R R
CMC		3F	1	4	4	F	PUSH	RP	11RP 0101	1	12	11	S W W*
CMP	REG	1011 1SSS	1	4	4	F	RAL		17	1	4	4	F
CMP	M	BE	1	7	7	F R	RAR		1F	1	4	4	F
CNC	LABEL	D4 addr	3	9/18	11/17	S R -/S R R W W*	RC		D8	1	6/12	5/11	S/S R R*
CNZ	LABEL	C4 addr	3	9/18	11/17	S R -/S R R W W*	RET		C9	1	10	10	F R R
CP	LABEL	F4 addr	3	9/18	11/17	S R -/S R R W W*	RIM (8085A only)		20	1	4	—	F
CPE	LABEL	EC addr	3	9/18	11/17	S R -/S R R W W*	RLC		07	1	4	4	F
CPI	DATA	FE data	2	7	7	F R	RM		F8	1	6/12	5/11	S/S R R*
CPO	LABEL	E4 addr	3	9/18	11/17	S R -/S R R W W*	RNC		D0	1	6/12	5/11	S/S R R*
CZ	LABEL	CC addr	3	9/18	11/17	S R -/S R R W W*	RNZ		C0	1	6/12	5/11	S/S R R*
DAA		27	1	4	4	F	RP		F0	1	6/12	5/11	S/S R R*
DAD	RP	00RP 1001	1	10	10	F B B	RPE		E8	1	6/12	5/11	S/S R R*
DCR	REG	00SS S101	1	4	5	F*	RPO		E0	1	6/12	5/11	S/S R R*
DCR	M	35	1	10	10	F R W	RRC		0F	1	4	4	F
DCX	RP	00RP 1011	1	6	5	S*	RST	N	11XX X111	1	12	11	S W W*
DI		F3	1	4	4	F	RZ		C8	1	6/12	5/11	S/S R R*
EI		FB	1	4	4	F	SBB	REG	1001 1SSS	1	4	7	F
HLT		76	1	5	7	F B	SBB	M	9E	1	7	7	F R
IN	PORT	DB data	2	10	10	F R I	SBI	DATA	DE data	2	7	7	F R
INR	REG	00SS S100	1	4	5	F*	SHLD	ADDR	22 addr	3	16	16	F R R W W
INR	M	34	1	10	10	F R W	SIM (8085A only)		30	1	4	—	F
INX	RP	00RP 0011	1	6	5	S*	SPHL		F9	1	6	5	S*
JC	LABEL	DA addr	3	7/10	10	F R/F R R†	STA	ADDR	32 addr	3	13	13	F R R W
JM	LABEL	FA addr	3	7/10	10	F R/F R R†	STAX	RP	000X 0010	1	7	7	F W
JMP	LABEL	C3 addr	3	10	10	F R R	STC		37	1	4	4	F
JNC	LABEL	D2 addr	3	7/10	10	F R/F R R†	SUB	REG	1001 0SSS	1	4	4	F
JNZ	LABEL	C2 addr	3	7/10	10	F R/F R R†	SUB	M	96	1	7	7	F R
JP	LABEL	F2 addr	3	7/10	10	F R/F R R†	SUI	DATA	D6 data	2	7	7	F R
JPE	LABEL	EA addr	3	7/10	10	F R/F R R†	XCHG		EB	1	4	4	F
JPO	LABEL	E2 addr	3	7/10	10	F R/F R R†	XRA	REG	1010 1SSS	1	4	4	F
JZ	LABEL	CA addr	3	7/10	10	F R/F R R†	XRA	M	AE	1	7	7	F R
LDA	ADDR	3A addr	3	13	13	F R R R	XRI	DATA	EE data	2	7	7	F R
LDAX	RP	000X 1010	1	7	7	F R	XTHL		E3	1	16	18	F R R W W
LHLD	ADDR	2A addr	3	16	16	F R R R							

Makina Çevrim Tipleri:
F 4 Saat periyodu komut getirme
S 6 Saat periyodu komut getirme
R Bellek okuma
I/O Okuma
W Bellek yazma

O I/O yazma
B Yol başta
X Değişken veya isteğe bağlı ikilik sayı
DDD Hedef kaydediciyi belirten ikilik sayılar
B= 000, C= 001, D= 010

SSS Kaynak kaydediciyi ikilik sayılar
E= 011, H= 100, L= 101, A= 111
RP Kaydedici çifti
BC= 00, HL= 10, DE= 01, SP= 11

*8080A'da beş-saat periyodu komut getirme
†Şartın yerine getirilmesine bakmaksızın 8080A ile daha uzun makina çevrim sırası. Bütün bellekler 1976'da Intel Corporation tarafından Copyright'a alınacaktır.

TABLO 8-2 8080A/8085A KOMUT SET

Örnekler arasındaki gerçek süre 1 ms den biraz daha uzun olacaktır, çünkü örnekler arısında yürütülen başka komutlar vardır. Daha doğru zamanlama için bu komutlara ait süreler de hesaplara alınmalıdır ve gecikme döngü sayıcısı daha küçük bir sayı ile yüklenmelidir. Farklı bir saat frekansı kullanılırsa, o saatin periyodu kullanılmalıdır.

8080A ve 8085A için bir çok komutun durum sayısının aynı olmadığına dikkat edin. Ayrıca, 8085A da JNZ gibi komutlar için 7/10 şeklinde iki sayı verilmiştir. Bu, atlama olmazsa 7 saat çevriminin, atlama olursa 10 saat çevriminin gerektiğini söylemektedir.

```
PROGRAMIN ADI                    8080A DURUMLARI        8085A DURUMLARI

DELAY:  PUSH  PSW                    11                     12
        PUSH  B                      11                     12
        MVI   B, ____H                7   } 29               7   } 31

CNTDN:  DCR   B                       5                      4
        JNZ   CNTDN                  10   } 15             10   } 14


        POP   B                      10                     10
        POP   PSW                    10   } 30              10   } 30
        RET                          10                     10
                                     59                     61
                                                  CNTDN DÖNGÜSÜ
                                                  DIŞINDA TOPLAM
```

Şekil 8-23 Her komut için durum sayısını gösteren geciktirme altyordamı

16-BİTLİK KAYDEDİCİ KULLANAN GECİKME DÖNGÜLERİ: Şekil 8-24'de gösterildiği gibi bir 16-bitlik kaydedici yükleyerek ve aşağıya doğru sayarak daha uzun gecikmeler yaratılabilir. Ancak DCX komutu sıfır bayrağını etkilememektedir. Bu nedenle, genişletilmiş kaydedicinin 0 a ulaştığını belirtmek için başka bir yol bulunmalıdır. Şekil 8-24 te gösterildiği gibi, bu genelde, 16-bitlik kaydedicinin bir baytını akümülatöre taşıyarak ve diğer baytla VEYA layarak yapılmaktadır. Eğer her iki baytta 0 ise, ORA komutu sıfır bayrağını set eder.

```
        LXI     D, (16-BİTLİK SAYI)
LOOP    DCX     D
        MOV     A,D
        ORA     E
        JNZ     LOOP
```

Şekil 8-24 16-bitlik geciktirme döngüsü

DAHA UZUN GECİKMELER İÇİN İÇİÇE GECİKME DÖNGÜLERİ

Daha uzun gecikmeler üretmek için diğer bir yöntem, içiçe gecikme döngüleri kullanmaktadır. Örneğin, 10 ms lik bir gecikme üretmek için, Şekil 8-22 deki 1 ms'lik döngü yordamı, bunu 10 defa tekrarlayan bir başka döngü içinde kullanılabilir. Şekil 8-25, bunu yapan bir programı göstermektedir. B kaydedici 0AH ile yüklenip aşağı sayılmaktadır.

```
DELAY   PUSH    PSW
        PUSH    B
        MVI     C,0AH
LOAD1   MVI     B,81H
CNTDN   DCR     B
        JNZ     CNTDN
        DCR     C
        JNZ     LOAD1
        POP     B
        POP     PSW
        RET
```

Şekil 8-25 Daha uzun gecikmeler için içiçe gecikme döngüleri

BİR PROGRAMI ELDE ÇEVİRME

Bir programı elde çevirmek, her etikete, op koda ve işlenene, bellek adreslerini ve makina kodlarını atamaktan ibarettir. Buna daha önce kodlama demiştik. Programın bellekteki başlama adresi programcı tarafından seçilir. Şekil 8-22, bellekte 2000H adresinden başlayarak çalışacak şekilde elde çevrilmiş son örnek programı göstermektedir. Programın elde çevrilmesi sırasında, programın bütün parçalarını düzenlemek için program çevirme kâğıdı kullanılmıştır.

İlk iki sütun onaltılık (hex) koddadır. Bunlar, gerçek adresleri ve bu programı çalıştırmak için bu adreslere yüklenecek ikilik kodları temsil etmektedir. Çok kullanılan bir şekil 65536 baytlık belleği, herbiri 256 bayt olan 256 sayfaya bölmektir. Böylece, sayfa adresi 16-bitlik adresin üst baytı ve satır adresi de alt baytı olur.

İki baytlık işleneni olan bütün 8080A/8085A komutlarında belleğe önce en küçük değerlikli baytın yüklendiğini tekrar hatırlatalım. Örneğin, LXI *H*, 2050H komutu, belleğe 21, 50, 20; JNZ LOOP komutu C2, 08, 20 ve CALL DELAY komutu CD, 25, 20 sırasında yüklenmiştir.

Uzun programların elde çevrilmesi en iyi halde bile zaman alıcı ve ince bir iştir. Eğer çevirmeyi elde yapmanız gerekiyorsa, size işinizi kolaylaştıracak bazı ipuçları verelim. Birincisi, herhangi bir kodlama yapmadan önce, çeviri dili programınızda herhangi bir komutu unutmadığınızdan emin olmak için, programınızı kontrol edin, tekrar kontrol edin (ve sonra bir daha kontrol edin!). Aksi halde, eğer bir komut unutursanız, yani araya bir komut sokmanız gerekirse, programın geri kalan kısmındaki bütün adresleri değiştirmeniz gerekecektir. Ayrıca bütün atlama ve çağırma komutlarında içerilen adreslerin de değiştirilmesi gerekecektir. Bu karmaşada hata yapma olasılığı çok yüksektir.

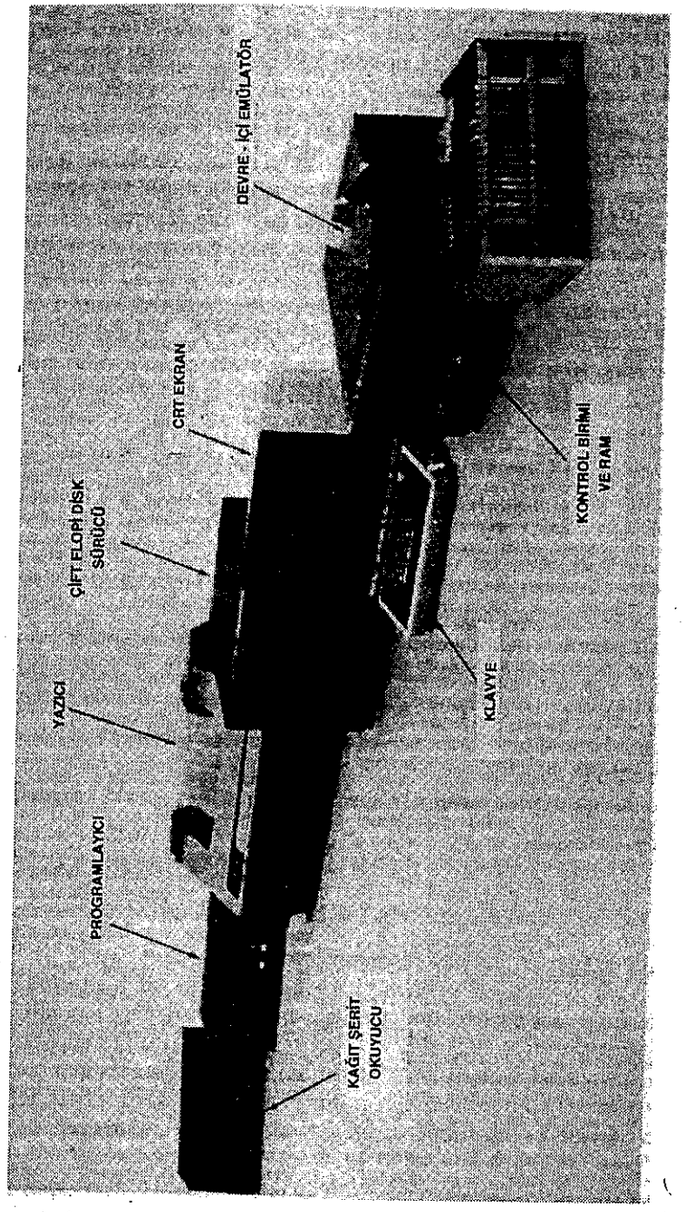
İkincisi, unutulmuş bir komutu araya sokma problemini en aza indirgemek için, her 16 program komutundan sonra araya birkaç NOP komutu konabilir. NOP komutları programın akışını etkilemez. Fazladan kullanılan bellek, komut eklemede sağlanan kolaylık için ödenen çok ufak bir fiyattır. Eğer bellek miktarı çok sınırlıysa, her program kısmının hataları ayıklandıtan sonra NOP komutları o kısımdan kaldırılabilir. NOP komutları ayrıca ana program ile altyordamlar arasında da bırakılmalıdır, böylece altyordamı çağırmalardaki adresleri değiştirmeye gerek kalmadan ana program genişletilebilir. 0 rakamını O harfinden ayırtetmek için çoğunlukla çizgili sıfır (Ø) kullanılır.

Birçok hata yapmadan uzun programları elde çevirmek hemen hemen imkânsız olduğu için, bu işi sizin için yapmak üzere, bilgisayar ve mikrobilgisayar-tabanlı geliştirme sistemleri tasarlanmıştır. Bir sonraki kısmın konusu bunlardır.

MİKROİŞLEMCİLİ GELİŞTİRME SİSTEMLERİ

Fiziksel olarak, bir geliştirme sistemi çoğunlukla 32K ve daha fazla RAM'ı, bir daktilo tipi klavyesi, bir video (CRT) ekranı, bir yazıcısı ve bir veya daha fazla flopi disk sürücüsünden oluşan bir mikroişlemci sistemidir. Ayrıca bir kağıt-şerit okuyucu ve bir PROM programlayıcı da bulunabilir. Şekil 8-26, Intel geliştirme sistemini göstermektedir. İyi bir geliştirme sistemi birçok işlevi yerine getirebilir. Şimdilik metin editörü ve çevirmenlik işlevlerini inceleyeceğiz. Bu işlevler, çeviri dili program yazmanın tatsızlığının çoğunu giderirler.

METİN EDİTÖRLERİ: Kafanıza sıkıca yerleştirmeniz gereken ilk nokta, hem bir metin editörünün hem de bir çeviricinin her birinin binlerce bayttan oluşan programlar olduğudur. Çoğunlukla, gerekli oluncaya kadar yu-



Şekil 8-26 İntel mikroişlemcili geliştirme sistemi (Intel Corporation)

muşak diskler üzerinde saklı tutulur. Metin editör programının ana işlevi, çevirici programın makina diline doğru şekilde tercüme edilebilmesi için, çeviri dili programı tam doğru formatta yazmanıza yardımcı olmaktır.

Geliştirme sistemiyle bir çeviri dili program yaratmak istediğinizde, önce metin editör programı RAM a yüklersiniz. Sonra editör programını çalıştırınız. Bu program çalıştırıldığında, klavyeyi kulanarak bir çeviri dili programını yazmanıza imkan verir. Program, RAM da editörün çalışma alanında saklanır. CRT ekranında görüntülenen 24 ile 40 arasındaki program satırı yazdıklarınızı görebilmenizi sağlar. Çeviri dili program, açıklamalarıyla birlikte (adres ve kodlar hariç) aynen Şekil 8-22 de gösterildiği gibi daktilo edilir. Bir tuşa dokunmakla, program ortalarında, hatalar düzeltilebilir veya satırlar eklenebilir yahut silinebilir.

ÇEVİRİCİ YÖNLENDİRİCİLERİ VEYA SÖZDE KOMUTLAR

Programı çeviriciye tam hazır hale getirmek için, program çevirici için yönlendiriciler içermelidir. Örneğin, çeviriciye hangi adresten itibaren programı çevirmeye başlayacağı söylenmelidir. Bu çeviri yönlendiricileri son programın bir parçası haline gelmezler, çünkü bunlar op kod değildir. Bu nedenle, bunlar sözde komutlar veya sahte komutlar diye de bilinirler.

Tablo 8-1 deki çevirici başvuru sütunu bir 8080A/8085A çeviricisinde kullanılan sözde komutları göstermektedir. Bu komutlar, her çevirici program için değişiktir, fakat aşağıda anlatılan yedi genel komutun, çoğu çeviricide karşılık gelen bir dengi vardır.

Orijin komutu ORG'un arkasında bir adres vardır. Bu komut çeviriciye hangi adresten itibaren programı çevirmeye başlayacağını söyler. ORG, programın başında veya programın sırasındaki adresten başka bir adresle çevirmeye devam etmesini istediğiniz herhangi bir yerine konur. Örneğin ORG 0100H, bu komutu izleyen programı, çeviricinin bellekte 0100H adresinden itibaren başlayarak çevirmesini söyler.

END komutu, çeviriciye programın sonunu belirtir. Eğer bir END satırı verilmezse, çevirici bütün belleği çevirmeye devam eder.

Eşit komutu EQU, programda kullanılan sembolleri tanımlar. Örneğin, eğer 736AH i programda defalarca kullanacaksanız, sayıya JOE gibi sembolik bir isim verebilirsiniz. Böylece LXI *D*, 736AH yazmak yerine LXI *D*, JOE yazabilirsiniz. Bu sayılardaki basamakların karışması gibi hataları azaltır. Çeviriciye her JOE gördüğünde yerine 736AH koyması gerektiğinin söylenmesi gerekir. Bir EQU cümlesi bunu yapar. Doğru format JOE EQU 736AH şeklindedir. Eşitleme cümleleri genelde programın baş tarafına konur, ancak mutlaka sembolik değerin ilk kullanılışından önce konulmalıdırlar.

SET komutu EQU ya benzer, ancak SET için, bir sembolik tanımı, programda daha sonra bir başka SET cümlesiyle değiştirilebilir. Bir EQU değeri bütün program boyunca sabittir. Örneğin, programın başında JOE SET 736AH yazabilir ve 500 satır sonra JOE nun tanımını JOE SET 4253H ile değiştirebilirsiniz.

Tablo 8-1 de yukarıdakilerden sonraki üç söz de komut *DS (define storage) saklama yeri tanımla; DB (define byte) bayt tanımla;* ve *DW (define word) kelime tanımladır.* DS yönlendiricisi, çeviriciye veri saklamak üzere bir bellek bloğunu rezerve etmesini (ayırmasını) söyler. Örneğin, BUFFER: DS 96, çeviriciye programı çevirirken veri saklamak üzere bellekte 96 bayt ayırmasını söyler.

Bir DB cümlesi, işlenen alanında içerilen verileri, artarda gelen bellek yerlerine saklar. Örneğin, DB 7AH cümlesi, bir önceki komuttan hemen sonra belleğe 7AH saklar. DB JOHN cümlesi, JOHN ın ASCII karakter dizisi 4A, 4F, 48 ve 4E yi ardarda gelen dört bellek yerinde saklar.

Bir DW komutu, işlenen alanındaki 16-bitlik veriyi ardarda gelen iki bellek yerine saklar. Önce en küçük değerlikli bayt, sonra en büyük değerlikli bayt saklanır.

MACROLAR: Uzun programlarda, sık sık bir grup komutun birçok defalar tekrar edilmesi gerekmektedir. Bunu çözmenin bir yolu, komutları bir altyordam olarak yazmak ve gerektikçe altyordamı çağırmaktır. Eğer istenilen komut grubu çok kısa ise ve/veya zamanlama nedenleriyle komut grubunun ana programla birlikte kodlanması gerekiyorsa, bu durumda bir MACRO kullanılır.

Bir MACRO, programın başında yazılan ve bir isim verilen komutlar grubudur. Çevirici, programda bu ismi her okuyuşunda, o MACRO da sıralanmış komutları alıp kodlar. Diğer bir deyişle, bir MACRO tanımlayabilmeniz, bir komutlar gurubunu tekrar tekrar daktilo etmenize gerek bırakmaz. Şekil 8-27*a*, LOAD1 isimli bir MACRO'nun, akümülatör ve *B* kaydedicisini iki ayrı bellek yerinden yüklemek üzere nasıl yazılabileceğini göstermektedir. Burada *A1*, MACRO nun her kullanılışında HL ye yüklenecek parametreyi (değişkeni) temsil eder. MACRO nun her kullanılışında, bu parametre farklı olabilir. Şekil 8-27*b*, bu MACRO nun bir programda nasıl çağrılacağını göstermektedir. Bir gerçek programda HL de bulunması istenen adres *A1*'in yerine konacaktır.


```
LOAD1   MACRO       A1
        LXI         H,A1
        MOV         A,M
        INX         H
        MOV         B,M
        INX         H
        ENDM
                (a)
        LOAD1   (b)     (MACRO nun bu kullanılışı
                        için istenilen gerçek adres)
```

Şekil 8-27 MACRO örneği.

ÇEVİRİCİYİ KULLANMA: Çeviri dili program editörün işalanında tamamlandığında, iyice kontrol edilir ve sonra bir flopi disk veya bant üzerinde bir dosyaya yazılır. Bundan sonra, çevirici program RAM a yüklenir. Bu program çalıştırıldığında, disk veya banttan çeviri dili programı okur ve makina diline tercüme eder. Genellikle, bir çeviricinin çeviriyi yapabilmesi için çeviri dili yani kaynak programı iki defa okuması gerekir. Birinci geçişte, çevirici programı okur ve bir sembol tablosu yaratır. Bu, çeviricinin, başlangıç adresinden itibaren sayarak etiketlere adres ataması demektir.

Çeviricinin ikinci geçişi, gerçekte, programın makina yani "amaç" koduna tercümesini yapar. Elde edilen makina kodu, bir flopi disk üzerinde bir dosyaya yüklenebilir veya çalıştırılmak üzere bir RAM a saklanabilir. Amaç kodu, bir, PROM a programlanabilir. Ayrıca, tamamen çevrilmiş program, Şekil 8-22 deki gibi bir formatta, bir printerden alınabilir.

Çoğu çeviriciler nizami olmayan op kodlar gibi hataları, yazılı çıktıda belirtirler. Eğer bir hata görülürse, kolayca düzeltilebilir. Editör programı ve çeviri dili kaynak dosya RAM a tekrar yüklenir. Programın hatayı içeren kısmı ekrana getirilir ve hata düzeltilir. Düzeltilen program, disk üzerinde bir kaynak dosyaya yazılır. Çevirici program tekrar yüklendikten sonra, kaynak dosya çevirici tarafından okunur ve çevrilir. Bu döngüye çevirici program hiçbir hata göstermeyinceye kadar devam edilir. Bu arada, hatasız bir yazılı çıktının programın çalışacağı demek olmadığını belirtelim. Hatasız çıktı sadece nizami op kodlar, adresler v.b. kullanılmış demektir.

Bazı geliştirme sistemleri, BASIC FORTRAN veya PASCAL gibi yüksekseviye dillerini, makina diline tercüme eden derleyici ve yorumcu programlara sahiptir. Bu diller, daha fazla İngilizce cümlelere benzeyen deyimler kullanırlar. Her BASIC, FORTRAN veya PASCAL deyimi, birçok çeviri dili komut deyimini temsil edebilir. Bölüm 14 te bu dillerle programlamayı anlatacağız.

ÇEVİRİ DİLİ PROGRAMLARDA ARIZA BULMA (HATA AYIKLAMA)

Bir program RAM a yüklenip ilk defa çalıştırıldığında, çoğunlukla doğru çalışmaz. Sistemli bir yaklaşım, rasgele didiklemeden daha çabuk problemi bulacaktır. İzleyebileceğiniz basamaklar şunlardır:

1. Algoritmanızı kontrol edin. İstediğinizi yapması için gerekli komutları içerdiğinizden emin olmak üzere aklınızdan program basamakları üzerinden geçin. Bir hesaplamanın sonucunu saklamayı hatırladınız mı? Bellek göstericisini veya yığın göstericisini ilklendirmeyi hatırladınız mı?
2. Doğru komutları yazıp yazmadığınızı kontrol edin. Çok yapılan bir hata, bir komutta işlenenlerin karıştırılmasıdır. Örneğin, MOV *M,A* yazmak niyetiyle MOV *A,M* yazılmış olabilir.
3. Her komutun kodlarının doğru olup olmadığını kontrol edin. Eğer komut MOV *B,C* ise, MOV *C,B*nin kodunu yazmadığınızdan emin olunuz. MVI A, 7AH gibi komutlarda, hazır veri 7A nın MVI komutunun kodundan hemen sonraki bayta kodlanmış olduğundan emin olunuz. JNZ 2050H gibi 3-baytlık komutlarda, atlama adresinin iki baytının da girilmiş ve doğru sırada (önce alt bayt) olduğundan emin olunuz.
4. Eğer bu noktaya kadar herşey tamam görünüyorsa, bundan sonra yapabileceğiniz mümkün bir işlem, programı tek-adımlamaktır. Çoğu mikroişlemcili geliştirme kartları tek bir komut yürütmenize, durmanıza ve bellek ve kaydedicilerin içeriklerini incelemenize müsade etmektedir. Bu, her komuttan sonra sonuçların doğru olup olmadığını kontrol etmenize imkân verir. Eğer sonuçlar doğruysa, bir sonraki komutu yürütün ve bellek ve kaydedicilerin içeriklerini tekrar kontrol edin. Eğer sonuçlar doğru değilse, istenilen sonucun niçin üretilmediğini anlamak için kodunuzu, komutlarınızı ve düşüncelerinizi inceleyin. Bir geciktirme döngüsü gibi bir sayım halkasını tek adımlamak için, geçici olarak sayımı 1 veya 2 ye azaltın, böylece döngüden çıkmak bütün gününüzü almayacaktır.
5. Uzun programlarda hata ayıklamak için, bir durdurma noktası (breakpoint) kullanışlı bir araçtır. Çoğu mikroişlemcili geliştirme kartları, bütün kaydedicilerin durumlarını saklayacak ve kontrolu monitör programına geçirecek bir komutu araya sokmanıza izin verir. Bundan sonra, monitör komutları, bellek yerlerini ve kaydedicileri incelemenize imkân verir. Örneğin, Intel SDK-85 kartında, RST 1 komutu bunu yapar. RST 1 in kodu CFH tır. Eğer bir programın ilk 100 baytı çalışıyor, fakat bundan sonra problemler çıkıyor gibi görünüyorsa, problemli kısmın başında bir komut kodunun yerine CFH kodu geçici olarak konulabilir. Çalıştırıldığında, program bu konulan koda kadar gelecek ve kontrolu monitöre verecektir. Şimdi, kaydedicilerin ve ilgili olabilecek bellek yerlerinin içeriklerini inceleyebilirsiniz. Bunlar doğruysa, orijinal komutla CFH'yi değiştirir ve CFH durdurma noktasını programda daha aşağı bir yere taşırsınız. Bu, dikkatle yapılırsa, programı bir durdurma noktasına kadar çalıştırır ve problemi bulmak üzere buradan itibaren tek adımlayabilirsiniz.
6. Bir programın hatalarını ayıklamada ana noktalar, sistematik olmak ve elinizdeki sistemin bütün özellik ve güçlerinden yararlanmaktır.

Bu bölümün büyük bir kısmı, bilgisayarcıların "software" dediği yazılımlar üzerineydi. Yazılım; komut kümelerini, programları ve programlamayı içine alır. Bölüm 9'da daha çok mikroişlemcilerin donanımıyla (hardware) uğraşacağız. Donanım; entegre devreleri, devre bağlantılarını ve zamanlama parametrelerini içine alır.

GÖZDEN GEÇİRME SORULARI VE PROBLEMLER

1. Basit bir bilgisayarın blok diyagramını yolları da göstererek çiziniz.
2. Bir minibilgisayar veya mikrobilgisayar sisteminde belleğin I/O portlarının ve CPU nun işlevlerini anlatınız.
3. Bir IN komutunu getirirken ve yürütürken yollar üzerindeki sinyal akışını anlatınız.
4. Bir 8080A/8085A sisteminin kaydedici ve bellek içerikleri Şekil 8-28 de gösterildiği gibi verilmişse, aşağıda sıralanan komutların bunların içeriklerini nasıl etkileyeceğini gösterin. Tek bir harf altında olmayan komutlar ardışıl olarak düşünülmemiştir.

KAYDEDICILER

A	73		
B	4A	5F	C
D	3C	A7	E
H	20	20	L

BELLEK

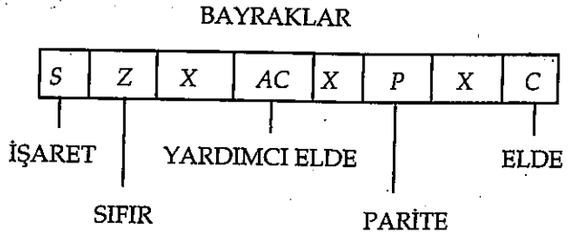
2020	3E
2021	7B
2022	2C
2023	F2

ŞEKİL 8-28

a.	INR	C	*h.*	CMP	C
b.	MOV	A,D	*i.*	ANI	0F0H
c.	MVI	B,7EH		RRC	
d.	LXI	H,2022H		RRC	
	MOV	A,M		RRC	
e.	MOV	A,L		RRC	
	ADD	D	*j.*	LDA	2023H
	INX	H		SUB	D
	MOV	M,A		DCR	B
f.	RAL			MOV	C,A
g.	DAD	B		ORA	0F0H

5. Şekil 8-28'deki kaydedici ve bellek içeriklerini ve Şekil 8-29 daki bayrak kaydedicisini kullanarak, aşağıdaki komutların herbiri yürütüldükten sonra bayrakların durumunu gösterin. Komutun başlangıcında bütün bayrakların reset olduğunu kabul edin.

a.	MOV	A,D
b.	MVI	A, 59H
	ADD	E
c.	SUB	E



6. 11110000 ile 10101010'un VE lenmesinin, VEYA lanmasının ve özel VEYA lanmasının sonucunu gösterin.

7. Aşağıdaki komutlar ve kodlamadaki hataları bulup düzeltiniz.

a.	3E	MVI	A,7FH
b.	21	LXI	H,27H
	27		
c.	77	MOV	A,M
d.	C6	ADI	B
e.	C3	JMP	307AH
	30		
	7A		

8. Port 01H den 100 veri örneği alan, herbirine 7 ekleyen ve sonuçları bellekte 2050H adresinden başlayarak saklayan bir programın akış tablosunu çizin.

9. Problem 8 de çizdiğiniz akıştablosunun çeviri dili programını yazın.

10. *a.* Şekil 8-21 deki programın, port 01 in en küçük değerlikli bitindeki H dan L ye (yüksekten alçağa) giden bir darbeyi algılamak için nasıl değiştirebileceğini gösterin.

b. Programın, port 01 in bit 0, yerine bit 6 sındaki bir yüksek darbeyi algılamak için nasıl değiştirilebileceğini gösterin.

c. Programın tekrar bir yükseği aramak üzere geri dönmeden önce, port 01 in en küçük değerlikli bitindeki darbenin alçağa gitmesini bekleyecek şekilde nasıl değiştirilebileceğini gösterin.

11. Aşağıdaki komut dizisinin yığın gösterici, yığın ve program sayıcı üzerindeki etkilerini gösterin.

2000	21C220	LXI	SP,20C2H
2003	CD5020	CALL	2050H
2006	77	MOV	M,A

12. Bir altyordamın sonundaki RET komutu, program sayıcıyı hangi adrese geri döndüreceğini nasıl belirler?

13. Bir altyordamda PUSH ve POP komutları niçin kullanılır?

14. Port 7 nin en küçük değerlikli bitinden 500 Hz lik bir kare-dalga darbe treni elde etmek için, Şekil 8-22*b* deki 1ms geciktirme yordamını kullanarak, bir 8085A çeviri dili program yazınız. (İpucu: Her döngüde geciktirme yordamını iki defa kullanın)

15. *a.* Şekil 8-22b deki DELAY geciktirme altyordamı, 3,072 MHz lik saati olan bir 8085A işlemcide çalıştırılırsa, gecikme süresinin ne olacağını hesaplayın.

b. 3,072 MHz saatli bir 8085A kullanarak, 1 ms lik bir gecikme elde etmek için gerekli geciktirme sabitinin değerini hesaplayın.

16. Veri örnekleri arasında 100 ms lik bir gecikme oluşturmak için 1 ms lik DELAY altyordamının nasıl kullanılabileceğini gösteren bir yordam yazın. (İpucu: Örnekler arasında altyordamı 100 defa kullanın).

17. Çarpma ardarda toplamayla yapılabilir. Örneğin, 3x5 işlemi, akümülatörü sıfırlayıp 3 ü 5 defa toplamakla gerçekleştirilebilir. Bu çarpma yöntemi için bir akış tablosu çizin. *B* kaydedicisindeki 4-bitliğe kadar bir ikilik sayı ile *C* kaydedicisindeki 4-bitliğe kadar bir ikilik sayıyı çarpan bir 8085A çeviri dili programı yazınız.

18. İki 8-bitlik ikilik sayıyı ardarda-toplama yöntemiyle çarpmak için genişletilmiş *H* ve *D* kaydedicileri ile DAD ve DCR komutlarını kullanan bir çeviri dili program yazınız. Çarpılan *E* kaydedicisinde ve çarpan *C* kaydedicisindedir.

19. Aşağıdakileri tanımlayınız:

    a. Editör

    b. Çeviri

    c. MACRO

20. Çalışmayan bir çeviri dili programın hatalarını ayıklamak için yapacağınız mantıklı işlemler dizisini anlatın.

REFERANSLAR/BİBLİYOĞRAFYA

Intel MCS-80/85 Users' Manual, Intel Corporation, Santa Clara, C.A, Ekim 1979.

Intel 8080/8085 Assembly Language Programming Manual, Intel Corporation, Santa Clara, CA, 1977-1978.

Leventhal, Lance A.: 8080/8085A Assembly Language Programming, Osborne and Associates, Inc., Berkeley, CA, 1978.

9.BÖLÜM

8080A/8085A SİSTEM DONANIMI VE ZAMANLAMASI

Bölüm 8, size 8080A/8085A mikroişlemcileri için çeviri dili programlar yazmayı tanıtmıştı. Bu bölüm, 8080A/8085A mikroişlemci sistemlerinin donanım bağlantıları, sinyalleri ve zamanlama ilişkileri üzerinde durmaktadır. Her ne kadar 8080A, yeni tasarımlarda kullanılmıyorsa da, var olan birçok cihazda bulunmaktadır. Bu nedenle, tamir yönünden bilinmesinde yarar vardır. Ayrıca 8080A yı anlamak 8085A yı anlamayı çok kolaylaştırmaktadır.

Bu bölümün yaklaşımı şu şekildedir: Önce, 8080A-tabanlı bir sistemde, adres, veri ve kontrol yolu sinyallerinin nasıl üretildiğine ve bir sistem oluşturmak üzere 8080A nın RAM, ROM ve port aygıtlarına nasıl bağlandığına genel bir şekilde bakılmaktadır. Sonra, 8085A-tabanlı bir sistem aynı şekilde ele alınmaktadır. Bundan sonra, bu iki sistem örnek alınarak adres kodunu çözme anlatılmaktadır. 8085A'nın zamanlama parametreleri üzerinde bir miktar durulduktan sonra, 8080A ve 8085A'nın özel bekleme, tutma ve durma durumları incelenmektedir. Bunu, her işlemcinin, kesme girişlerini nasıl kodlandığını açıklayan bir kısım izlemektedir. 8080A ve 8085A'nın donanımları üzerinde anlatılanlar tamamlandıktan sonra, mikroişlemcilerde arıza bulmak için sistematik bir yöntem verilmektedir. Bu bölüm, *Z-80* mikroişlemci ile 8048 tek-yonga mikroişlemci ailesini tanıtarak sona erer. Bu bölümün temel amaçlarından biri, belli bir mikroişlemci hakkında ihtiyacınız olan donanım bilgisini edinmek için üretici firmaların veri kitaplarını okuyup anlamanıza yardımcı olmaktır.

Bu bölümü bitirdikten sonra şunları yapabileceksiniz:

1. 8080A, 8224 ve 8228 CPU grubunun işlevlerini anlatabileceksiniz.
2. Durum, komut çevrimi ve makina çevrimini tanımlayabilieceksiniz.
3. 8085A nın alt 8 adres bitini veri yolu üzerine nasıl çoğulladığını anlatabileceksiniz.
4. Basit giriş ve çıkış işlemleri için 8155 ve 8355 gibi programlanabilir port birimleri ilklendirebileceksiniz.
5. Adres kod çözücüsünün her çıkışının yetkilendirdiği adres aralığını söyleyebilecek ve bir mikroişlemci sistemi için bir bellek haritası çizebileceksiniz.
6. Bellek-haritalı *I/O* ve doğrudan *I/O* yu tanımlayabileceksiniz.
7. 8080A/8085A' nın bekleme, tutma ve durma durumlarını açıklayabileceksiniz.

8. Bir 8080A ve bir 8085A sisteminde kesmelere nasıl hizmet verildiğini açıklayabileceksiniz.
9. Arızalı bir mikroişlemci sisteminin sistemli, bir şekilde arızasını bulmadaki ana basamakları sırayla söyleyebileceksiniz.
10. *Z-80* ve 8048 mikroişlemcilerinin ana özelliklerini anlatabileceksiniz.

BİR 8080A MİKROİŞLEMCİ SİSTEMİ

CPU GRUBU

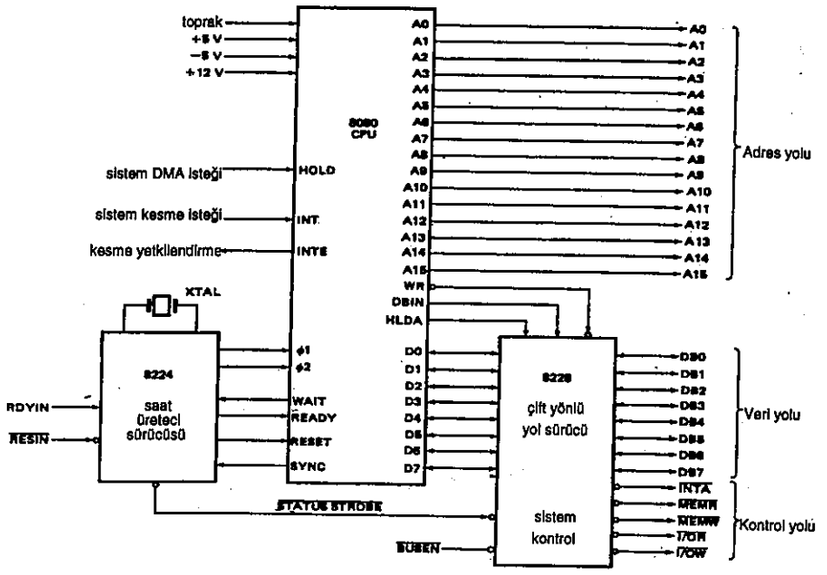
Şekil 9.1, işlevsel bir CPU bloğu oluşturmak üzere bir 8080A CPU, bir 8224 saat üreteci ve bir 8228 sistem kontrolör ve yol sürücünün nasıl bağlandığını göstermektedir. 8080A CPU, iki fazlı saat gerektirir. Bu saat, bir kristal osilatör ve bazı entegre devreler kullanılarak üretilebilir, ancak genel yöntem bir tek 8224 saat üreteci entegresi kullanmaktır. 8224, saat ve 8080A bloğunun gerektirdiği diğer kontrol sinyallerini üretmek için sadece bir harici kristal gerektirir. Bu saatlerin ürettiği sinyalin gerilim salınımı +0,6 V tan +11V kadardır.

8228 in iki ana işlevi vardır. Birincisi, veri hatlarını sürme yeteneğini artıran bir tampon gibi davranır. Yol sürücü çift yönlüdür, çünkü veri yolu üzerinde veri gidiyor veya geliyor olabilir. İkincisi, 8228, uygun kontrol yolu sinyallerini üretir: *belleği oku (memory read: MEMR), belleğe yaz (memory write : MEMW), giriş-çıkışı oku (input-output read : I/OR), giriş-çıkışa yaz (input-output write : I/OW)* ve *kesme alındı (interrupt acknowledge: INTA)*. **8080A**, her makina çevriminin başında, bu sinyalleri kodlanmış şekilde veri yolu üzerinden gönderir. 8224 ten gelen *durum yükleme (status strobe: STSTB)*, bu kodlanmış şekli 8228 e kilitler. 8228, istenen kontrol yolu sinyallerini üretmek üzere bu kilitlenmiş sinyallerin kodunu çözer.

Şimdi, bu temel CPU grubu tarafından üretilen sinyaller dizisini gösteren zamanlama diyagramının nasıl okunacağını açıklayacağız.

KOMUT ÇEVRİMLERİ, MAKİNA ÇEVRİMLERİ VE DURUMLAR

Şekil 9.2, bir 8080A mikroişlemcisinin bir komut çevriminin zamanlama diyagramını göstermektedir. Bir komut çevrimi, bir bilgisayar veya mikroişlemcinin bir komutun bütün gerekli parçalarını alıp getirmesi ve o komutu yürütmesi için gerekli zamandır. Bir 8080A için, her komut çevrimi bir ile beş arasında makina çevriminden oluşur. Bunlar, Şekil 9.2 de, *M1, M2* ve *M3* olarak etiketlendirilmiştir. Bellekten bir komut baytı getirmek, makina çevrim işleminin bir örneğidir. Bellek veya I/O portlarıyla ilgili her işlem bir makina

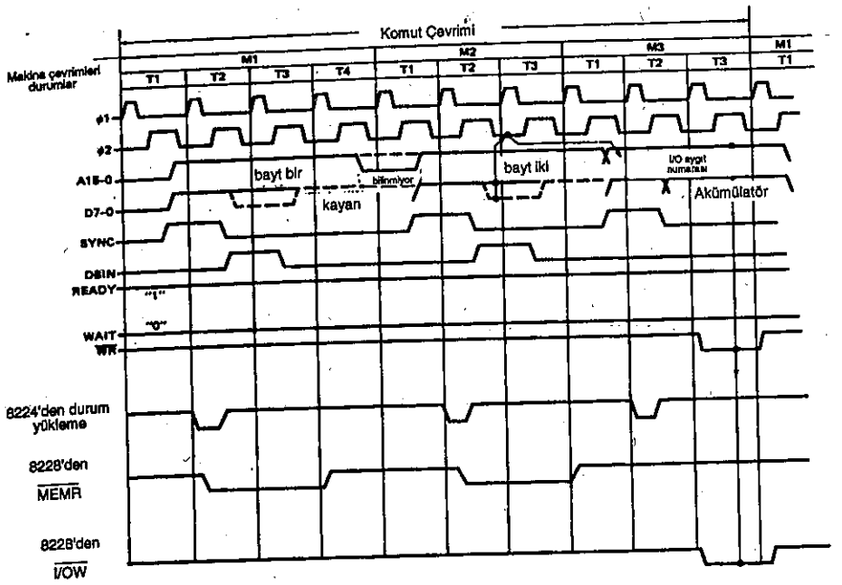


Şekil 9.1 8080A, 8224 ve 8228 ile birlikte 8080A CPU grubunun şeması. (Intel Corporation.)

çevrimi gerektirir. Makina çevriminin diğer örnekleri, belleği oku, belleğe yaz, I/O yu oku, I/O ya yaz ve kesme alındır. Makina çevrimleri de durumlara bölünmüşlerdir. 8080A için bir makina çevrimi üç, dört veya beş durumdan oluşur. Daha sonra anlatılacak olan bekleme, tutma ve durma durumları dışında, bir 8080 durumu, bir $\phi 1$ saat darbesinin pozitif kenarı ile bir sonraki darbenin pozitif kenarı arasındaki süre olarak tanımlanmıştır. Tablo 8.2, her 8080A komutunun durum sayısını göstermektedir.

Şekil 9.2, 8080A nın OUT (port) komutu için makina çevrimlerini, durumları ve yol bilgilerini göstermektedir. Şekil 9.2 nin üst iki satırı, komut çevrimleri, makina çevrimleri ve durumların bütün zamanlama dalgaşekillerini üretmek için 8080A nın gerektirdiği $\phi 1$ ve $\phi 2$ saatlerini göstermektedir. Komut çevrimi, üç makina çevrimi gerektirmektedir -*M1*, *M2* ve *M3*. M1, OUT komutunu bellekten getirir; *M2* bir sonraki bellek yerinden port adresini getirir ve *M3* portu adresler ve akümülatörün içeriğini çıkışa verir. Komut getirme, daima bir komut çevriminin birinci makina çevrimidir.

Komutu getirme makina çevriminin ilk durumu *T1* sırasında, komutun adresi, adres yolu üzerine konur. Bu, Şekil 9.2 de *A15*-0 etiketli dalgaşekli ile



Şekil 9.2 Bir OUT (port) komutu yürüten makina çevrimleri ve durumlarını gösteren 8080A zamanlama dalgaşekilleri.

gösterilmiştir. Bu dalgaşeklinin üste çıktığı nokta, bu noktada bütün 16 adres hattının üste çıktığı anlamında değildir. Bunun anlamı, o anda adres yolunda geçerli bir adres olduğudur.

Bir makina çevriminin ilk durumu esnasında, CPU, veri yolu üzerine ayrıca durum kelimesini (status word) koyar. Bu, Şekil 9.2 de üste giden D7-0 dalgaşekliyle gösterilmiştir. Durum kelimesi, makina çevrimi sırasında gerçekleştirilecek işlem türünü belirtir. 10 mümkün işlem ve bunların durum baytlarının hex kodu şunlardır: komut getirme, A5H; belleği okuma, 82H; belleğe yazma, 00H; yığını okuma 04H; girişi okuma, 42H; çıkışa yazma, 10H; kesme alındı, 23H; durma alındı 8AH; ve dururken kesme alındı, 2BH.

Şekil 9.2 ye bakın ve OUT komutunun *M1* çevriminin, *T1*, *T2*, *T3* ve *T4* olmak üzere dört durumu olduğuna dikkat edin. İlk durum *T1* sırasında, adres taşıtı (*A15* - 0) üzerinde, bir bellek adresi, ve veri yolu (*D7* - 0) üzerinde durum bilgisi gönderilir. 8080A dan gelen bir SYNC sinyali, durum kelimesinin veri taşıtı üzerinde olduğunu belirtir. 8224, STSTB darbesini üretmek için SYNC sinyalini ve $\phi 2$ saatini kullanır. Durum yükleme darbesi, durum kelimesini 8228'e yükler.

M1'in T2 durumunda, ϕ2' nin yükselen kenarı, 8080A yı, durum kelimesini veri yolundan kaldırmaya yönlendirir (kesikli çizgi alçağa gider-*D7 - 0*). Bundan sonra, 8080A, veri yolunun giriş modunda olduğunu belirtmek üzere $\overline{DBIN}$ sinyalini gönderir. 8228, $\overline{MEMR}$ sinyalini üretmek üzere durum kelimesinin kodunu çözer. $\overline{MEMR}$, adreslenen belleği yetkilendirmek için kullanılır. Belleğin, çıkışı yetkilendirme erişim zamanından sonra, bellekteki bayt Şekil 9.2 de "bayt 1" diye etiketlendirilen anda veri taşıtında (*D7 - 0*) gözükür. 8080A için, bellekten gelen veri, *M1* in *T3* durumunda, ϕ2 nin yükselen kenarından en az 150 ns önce geçerli olmalıdır. *T3* de, ϕ2 nin yükselen kenarı $\overline{DBIN}$ i reset eder. Bir 30 ns kadar sonra, 8228, $\overline{MEMR}$ yi tekrar yükseğe çıkarır. Bu, bellek çıkışını tekrar kayan hale getirir (Şekil 9.2 de "kayan" diye isimlendirilen kesikli çizgi- D7-0).

M1'in *T4* durumu, komutun kodunu çözmek gibi işlemcinin dahili işlemleri için kullanılır. OUT komutunun, saklanmış olan port adresini getirmek için bir başka bellek okuma işlemi gerektiğini belirtmek üzere *T4* sırasında kodu çözülür. Belleği tekrar okuma işlemi, Şekil 9.2 de, *M2* makina çevrimi olarak gösterilmiştir. *M2* çevrimindeki işlemler, *M1* çevriminin ilk üç durumu ile aynıdır. Bu çevrim için *T4* durumu gerekmemektedir, çünkü bellekten komut yerine veri (port adresi) okunmuştur. Bu verinin kodunun çözülmesine gerek yoktur.

M2 çevriminin sonunda, komut ve port adresi CPU'dadır. *M3* çevrimi sırasında komut yürütülür. *T1* de, port adresi, adres yolunun alt 8 bitine ve durum kelimesi de veri yoluna konur. 8080A durum kelimesinin veri yolunda olduğunu göstermek üzere bir SYNC darbesi gönderir. 8224, veri kelimesini 8228 e kilitleyen $\overline{STSTB}$ üretmek için bu SYNC darbesini kullanır.

M3 'ün *T2* 'sinde, *ϕ2*' nin yükselen kenarı, veri yolundan durum kelimesini temizler. Bundan sonra 8080A, akümülatörün içeriğini veri yoluna koyar. (Daha önceki komut çevrimi sırasında, porta çıkış vermek istediğiniz veri akümülatöre yüklenmiştir.) *T3* 'e *ϕ1* saatinin yükselen kenarı, 8080A'yı $\overline{WR}$ darbesini vermek üzere tetikler. 8228, bu $\overline{WR}$ darbesi ve kilitlenmiş olan durum kelimesini kullanarak, adreslenmiş çıkış portunu yetkilendirmek için kullanılabilecek $\overline{I/OW}$ darbesini üretir.

Diğer komutların da benzer makina çevrimleri ve durumları vardır. Bütün 8080A komutları, bellekten komut getirme çevrimi M1 ile başlar.

Şekil 9.2 'deki gibi bir takım dalgaşekillerinde, sinyal dizilerinin analizini yapmak için, bir dik çizginin dalgaşekilleri üzerinde soldan sağa doğru yavaşça kaydığını düşünmek genellikle yardımcı olur. Biraz önce anlatılan OUT komutu üzerinde dikkatlice çalışırsanız, 8080A veri kitabındaki diğer komut dizilerini anlamakta hiç zorluk çekmeyeceksiniz.

8080A SİSTEM ŞEMASI

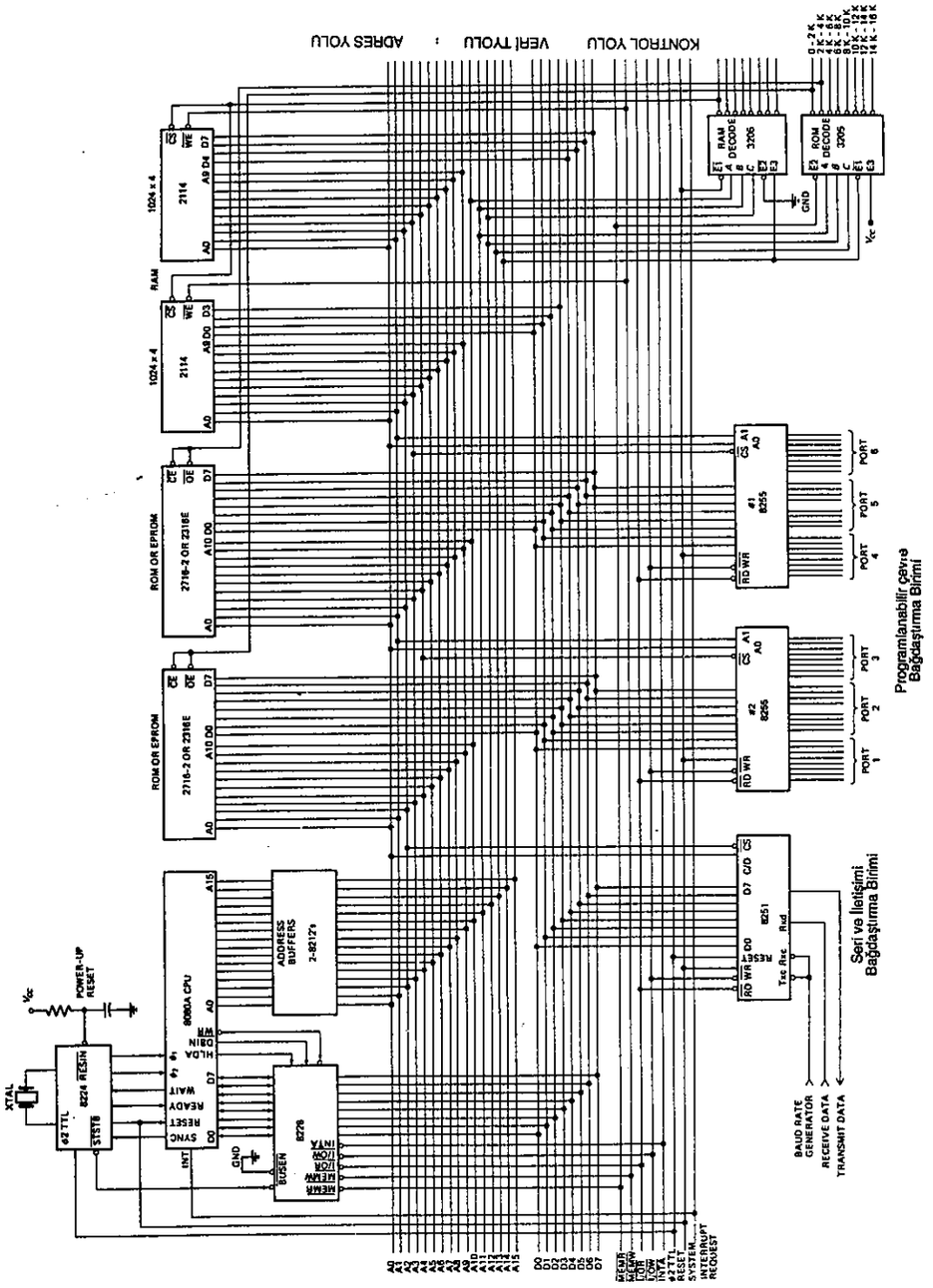
Şekil 9.3 bir 8080A tabanlı mikrobilgisayar sisteminin şemasını göstermektedir. Bu kısımda, sistemi oluşturmak üzere temel CPU grubuna, ROM, RAM, I/O portları ve diğer aygıtların nasıl eklendiğini açıklayacağız.

AÇILIŞTA RESET ETME DEVRESİ Şekil 9.3 'te, 8224 'ün $\overline{RESIN}$ girişinde V_{cc} 'ye bağlı dirençle toprağa bağlı kapasitöre dikkat edin. Güç ilk açıldığında, kapasitör yüklü değildir. Kapasitör, direnç üzerinden şarj oluncaya kadar $\overline{RESIN}$ alçak tutulur. Bu 8224 ün, 8080A ve diğer sistem çevre birimlerine bir RESET sinyali üretmesine neden olur. Reset olduğunda, 8080A program sayıcısına 0000H adresini yükler ve bu adresten komut getirme makina çevrimini gerçekleştirir. Bu nedenle, 0000H adresine, programcının resetten sonra yürütülmesini istediği ilk komutun konulması gerekmektedir.

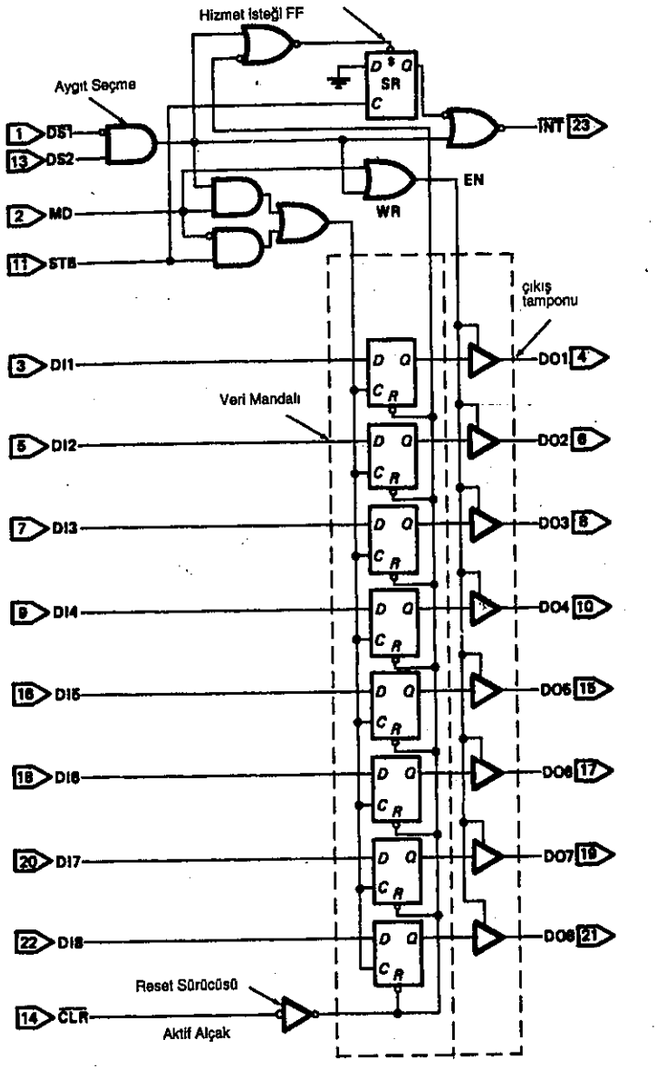
ADRES TAMPONLARI 8080A nın adres çıkışları sadece yaklaşık 100 pf lık kapasitörle bir standart TTL girişi sürebildiklerinden, genellikle adres hattı tamponları kullanılmaktadır. Belleğin erişim zamanını uzatmamaları için tamponların kısa yayılma gecikmeleri olmalıdır. Çok kullanılan adres hattı tamponları 74LS244 sekizli ters çevirmeyen sürücü ve 8212 8-bitlik giriş-çıkış portudur. Her ikisinin de, daha sonra anlatılacak, *doğrudan bellek erişimi (direct memory access: DMA)* için gerekli olan üç-durumlu çıkışları vardır.

8212 çeşitli kullanışları olan bir entegredir. Şekil 9.4 'te gösterildiği gibi, 8212, her birinin çıkışında üç-durumlu tampon olan sekiz D mandalı içerir. Ayrıca, aygıtın, iki aygıt seçme (device select) girişi $\overline{DS1}$ ve *DS2*, bir mod girişi *MD*, bir yükleme (strobe) girişi *STB* ve bir temizleme (clear) girişi $\overline{CLR}$ vardır. Bir mandal için, yükleme girişi yüksek olduğu sürece çıkışın girişi izlediğini hatırlayın. Bu nedenle, tampon olarak kullanmak için, mod girişi (*MD*) ve $\overline{DS1}$ alçağa bağlanır, *STB*, *DS2* ve $\overline{CLR}$ girişleri +5 V 'a bağlanır. Doğruluk tablosunun anlamanıza yardımcı olması için aygıtın mantık şemasını kullanın. 8212, 10 standart TTL girişini sürebilir. Bu aygıtın diğer kullanılış şekilleri daha sonra gösterilecektir.

ROMLAR Şekil 9.3'teki mikrobilgisayar sisteminde gösterilen 2716 EPROM, Bölüm 3 te anlatılan 2048 x 8'lik aygıttır. $\overline{CS}$ girişi yüksek olduğunda, yongayı alçak-güç bekleme moduna koyar ve $\overline{OE}$ girişi yüksek olduğunda, çıkışı kayan hale getirir. ROM'u okuyabilmek için her iki girişin de alçak olması gerekir. Elemandaki 2048 baytı adreslemek için 11 adres hattı gerekmektedir.



Şekil 9.3 RAM, ROM ve I/O portlarıyla birlikte 8080A-tabanlı mikrobilgisayar sistemi. *(Intel Corporation.)*



STB	MD	(DS1-DS2)	Veri Çıkışı Eşittir
0	0	0	3-durumlu
1	0	0	3-durumlu
0	1	0	veri kilitleme
1	1	0	veri kilitleme
0	0	1	veri kilitleme
1	0	1	veri girişi
0	1	1	veri girişi
1	1	1	veri girişi

CLR	DS1-DS2)	STB	*SR	INT
0	0	0	1	1
0	1	0	1	0
1	1	↓	0	0
1	0	0	1	1
1	0	↑	1	0

* Dahili SR flip-flop

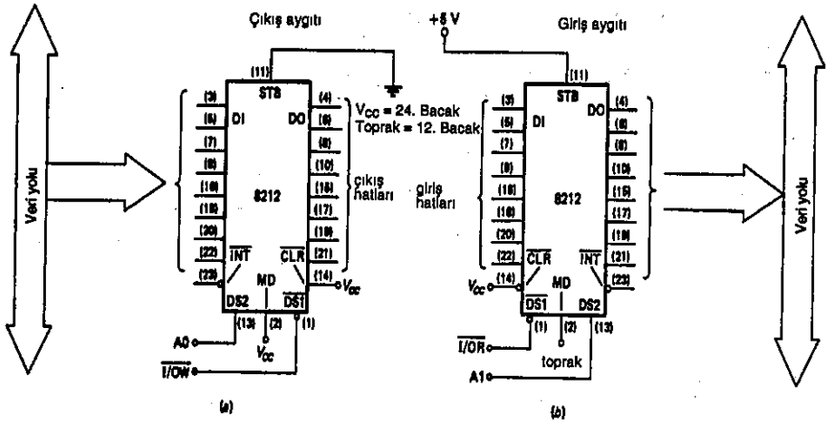
$\overline{CLR}$ – Veri mandalını reset eder.
SR flip-flop'u set eder.
(Çıkış tamponuna etkisi yoktur.)

Şekil 9.4 Doğruluk tablosu ile birlikte 8212 sekizli mandal ve I/O portu. *(Intel Corporation).*

RAM Bir sistemde, geçici olarak veri saklamak ve yığın olarak kullanmak için genellike bir miktar RAM içerilir. Şekil 9.3 'teki sistem 4K lık (1024 x 4) statik RAM olan Intel 2114 lerden iki tane kullanmaktadır. x4 formatının avantajı, bayt saklamak için sadece iki tane RAM gerekmesidir. x1 RAM' lar da bayt saklamak için 8 tane entegre gerekecektir. İki 2114, biri üst 4 biti (nibıl), diğeri de alt 4 biti saklayacak şekilde bağlanmıştır. 10 adres hattı, saklanmış 1024 nibıldan birini seçmek üzere doğrudan her iki 2114 'e de bağlıdır. Her bir 2114 ün yazmayı yetkilendirme (write enable: $\overline{WE}$) girişi, 8228'in $\overline{MEMW}$ hattına bağlıdır. Her iki 2114 ün yonga-seçme $\overline{CS}$ girişleri birlikte bağlanmıştır, çünkü bir tam bayt elde etmek için her ikisi de seçilmelidir.

PARALEL I/O PORTLARI *I/O* portları, donanım olarak çeşitli şekillerde gerçekleştirilebilir. Basit bir giriş ve çıkış portu 8212 ile yapılabilir. 8212, her biri 10 standart TTL girişi sürebilen üç-durum tampon çıkışlı bir sekizli mandaldır. Şekil 9.4, 8212' nin mantık şemasını ve Şekil 8.5 de bir giriş ve bir çıkış portu olarak 8212'nin bağlanışını göstermektedir. Bir çıkış portu için, *MD* yükseğe ve *STB* alçağa bağlanır. $\overline{DS1}$, 8228 in $\overline{I/OW}$ çıkışına ve DS2 de A0 adres hattına bağlanır. Ne zaman *A0* yüksekse ve $\overline{I/OW}$ alçalırsa, 8212 nin sekiz veri girişinde bulunan veriler çıkışına aktarılır. $\overline{I/OW}$ yükselince, veriler çıkışlara kilitlenir. İstenirse *MD* 'yi alçağa çekerek çıkışlar kayan hale getirilebilir.

Şekil 9.5b'de gösterildiği gibi, 8212 giriş portu olarak da kullanılabilir. $\overline{DS1}$, 8228 'in $\overline{I/OR}$' ına ve *DS2* bir adres hattına bağlanır. *STB* yükseğe bağlanır. Aygıta $\overline{I/OR}$ alçak ve *DS2* yükseğe geldiği anlar dışında çıkış tamponlarının kayan olması için *MD* alçağa bağlanır.



Şekil 9.5 I/O portu olarak kullanılan 8212. ***(a)*** **Çıkış portu.** ***(b)*** **Giriş portu.**

8255 Giriş ve çıkış portları oluşturmanın diğer bir yolu, 8255A gibi bir *programlanabilir çevre bağdaştırma entegresi (programmable peripheral interface IC)* kullanmaktır. Bu entegrenin, giriş ve çıkış olarak programlanabilen 8-bitlik üç portu vardır.

Güç verildiğinde, 8255A reset olur. 8255A'nın bütün port hatları ve veri yol hatları yüksek-Z durumundadır. Entegre giriş ve çıkış portları olarak kullanılmadan önce ilklendirilmelidir. Bu, 8255A'ya bir kontrol kelimesi gönderen programdaki bir seri komutla yapılır. Bunun gibi programlanabilir paralel portların nasıl ilklendirildiğini daha sonra anlatacağız.

SERI I/O PORTLARI : 8251A USART Bir mikroişlemci sistemi ile bir teletayp, MODEM veya CRT ekranı arasındaki iletişim için, veriler paralelden seriye veya seriden paralel şekle çevrilmelidir. Bölüm 4'te, bir üniversal senkron-asenkron alıcı-vericinin, yani USART'ın, bu dönüşümü nasıl gerçekleştirebileceğini göstermiştik. UART veya USART'ın, temelde paralel-giriş seri-çıkışlı kaydırma kaydedici ve bazı kontrol devreleriyle birlikte paketlenmiş bir seri-giriş paralel-çıkışlı kaydırma kaydedicisinden ibaret olduğunu unutmayın.

8251A, bir programlanabilir USART tır. Yonga, hemen hemen her seri formatta veri alabilir veya gönderebilir. Çoğu programlanabilir entegrelerde olduğu gibi, cihaz programın başında ilklendirilmelidir. 8251A yı ilklendirmek, ona 2 kontrol baytı göndermekten ibarettir. Bu elemanın nasıl ilklendirildiğini Bölüm 11'de göstereceğiz.

ADRES ÇÖZÜCÜLER Şekil 9.3 teki 8080A sistemi, iki 3205 i adres çözücü olarak kullanmaktadır. Bir mikroişlemci bir programı yürütürken, komutları ve verileri almak için belleği adresler. Mikroişlemci, dış dünyaya veri göndermek ve dış dünyadan veri almak için portları adresler. Adres çözücü devrenin işlevi, gönderilen adresi, o adres için istenilen entegreyi seçmek için gerekli sinyale çevirmektir. Bu kod çözme sadece istenilen cihazı seçmemeli, ayrıca, aynı anda başka hiç bir cihazın seçilmemesini de garanti etmelidir. Adres çözümleme tekniklerini genel olarak daha sonra anlatacağız ve bir sistemin şemasından her cihazın adresini belirlemek için basit bir teknik göstereceğiz. Bundan sonra, Şekil 9.3 'teki 8080A sisteminin özel adres çözücüsünü inceleyeceğiz.

BİR 8085A MİKROİŞLEMCİ SİSTEMİ YENİ BİR İŞLEMCİYİ ÖĞRENMEK

Bu kitap size, piyasada bulunan bütün mikroişlemcileri anlatamaz; ancak karşılaştığınızda, yeni bir mikroişlemciyi öğrenmenin bir yöntemini göstermeye çalışacaktır. Bu bölümün gerisinde ve bundan sonraki bölümde, çok kullanılan mikroişlemcilerin bir kaçı ile çalışmanıza imkan vereceğiz.

Yöntem şu şekilde özetlenebilir. İlk olarak, CPU mimarisi için aygıtın kılavuzuna bakın. Kaydedici sayısına, veri yolunun genişliğine, adres yolundaki bit sayısına ve gerekli güç kaynaklarına dikkat edin. Hangi giriş ve çıkış kontrol sinyallerinin ve hangi kesme girişlerinin var olduğuna bir ilk bakış olmak üzere bacak tanımlarını okuyun.

İkinci olarak, komut kümesini bulun ve CPU 'nun ne çeşit komutları yapabildiğini bulmak üzere bunu okuyun. Doğrudan I/O komutları mı var? Hangi koşullar altında atlayabilir veya dallanabilir? Hangi aritmetik işlemleri gerçekleştirebilir? Kesmelere nasıl davranılmaktadır?

Üçüncü olarak, cihazın kullanıldığı küçük bir sistem için verilmiş bir şema olup olmadığını araştırın. Bu şemadan, bu CPU nun diğer cihazlarla nasıl bağlandığı ve bu sistemin gördüğünüz diğer sistemlerle nasıl karşılaştırılabileceği hakkında bir fikir elde edebilirsiniz.

Dördüncü olarak, kontrol sinyallerini, zamanlama dalga şekillerinin ve a.a. karakteristiklerinin ayrıntılı tanımlarını inceleyin. Aygıtın üzerindeki her bacağın işlevini anlamaya çalışın.

Bu dört basamak, gerçekte belli bir işlemci ile çalışırken üzerinden defalarca geçeceğiniz bir döngüyü oluşturmaktadır.

BLOK DİYAGRAMI VE BACAK TANIMI

8080A hakkındaki bazı eleştiriler, üç güç kaynağı gerektirdiği, çalışan bir CPU oluşturmak için üç entegreye gerek duyduğu ve tek bir kesme girişi olduğu idi. Intel tasarımcıları bu eleştirilere, bütün bu problemleri ve başkalarını da çözen 8085A ile cevap verdiler.

Şekil 8.3b, 8085A'nın bir işlevsel blok diyagramıdır. 8085A gerçekte, 8224, 8228 ve 8080A CPU grubunun bütün işlevlerini ve hatta daha fazlasını bir tek paketle gerçekleştirmektedir. 8085A, tek bir 5V'luk kaynakla çalışır. 8085A nın gerektirdiği saatler, *X1* ve *X2* bacakları arasına bir harici kristal bağlanarak dahili olarak üretilmektedir.

Dahili kaydedici yapısı, ALU, bayraklar, v.b. 8080A'dakilerle aynıdır, fakat buradan itibaren 8085A farklıdır. Bacak yetmezliğinden dolayı, alt 8 adres biti, veri yolu AD0 - AD7 ye çoğullanarak verilir. Bir adres mandalını yetkilendirme çıkış sinyali (ALE), dışardaki cihaza veri yolunda bir adres olduğunu söyler. ALE sinyali, alt 8 biti bir harici mandala kilitlemek için kullanılır.

8080A'nın INT girişine benzer kesme girişi INTR'in yanısıra, 8085A'nın *RST 5.5*, *RST* 6.5, *RST* 7.5 ve TRAP şeklinde isimlendirilen dört vektörlenmiş kesmesi vardır.

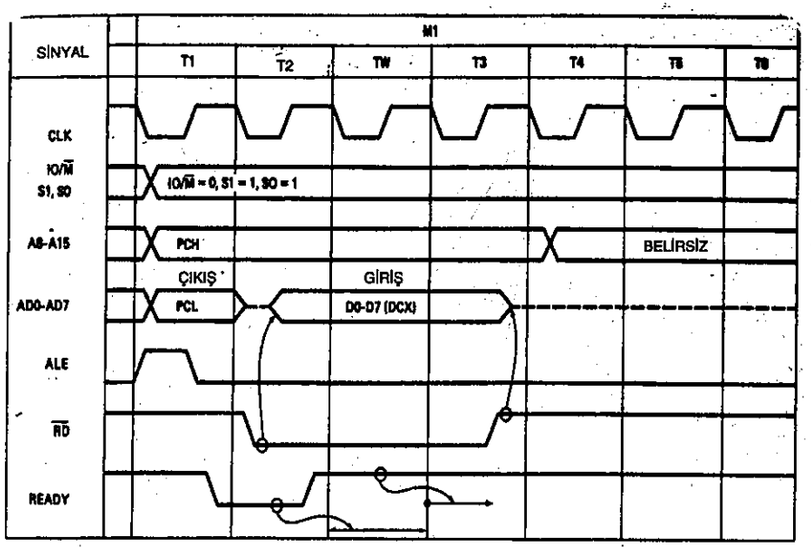
8085A'nın bir *seri veri giriş (SID)* ve bir *seri veri çıkışı (SOD)* bacağı vardır. Bir teletaypla olduğu gibi basit seri haberleşme şekilleri için, bu ikisi harici UART'ı gereksiz kılar.

8085A'nın kontrol sinyalleri oldukça basittir. Bir $\overline{RD}$ çıkış, adreslenen bellek veya I/O portunun okunacağını belirtir. Bir $\overline{WR}$, adreslenen bellek yerine veya I/O portuna yazılacağını belirtir. Bir diğer çıkış $IO/\overline{M}$, yazma veya okumanın belleğe mi yoksa I/O portuna mı olacağını belirtir. $IO/\overline{M}$ yüksekse okuma veya yazma I/O portundan olur. Bir bellek okuması sırasında $IO/\overline{M}$ alçaktır.

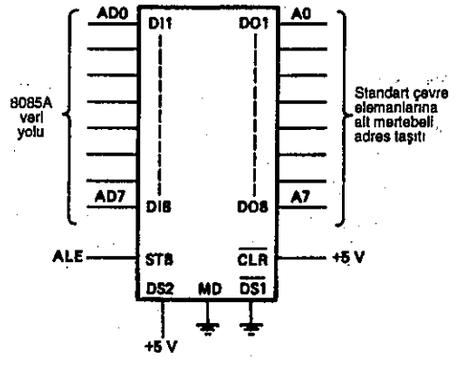
Veri yolu durum çıkışları *S1* ve *S0*, yürütülmekte olan çevrim tipine ait kodlanmış bilgi içerir. *S1S0* in 00 olması durmayı; 01, bir yazma; 10, bir okuma; ve 11, bir komut getirmeyi belirtir. Diğer 8085A bacakları daha sonra anlatılacaktır.

KOMUT ÇEVRİMLERİ, MAKİNA ÇEVRİMLERİ VE DURUMLAR

8080A' da olduğu gibi, 8085A 'nın her komut çevrimi 1 ile 5 makina çevriminden oluşur. Her makina çevrimi de 3 ile 6 saat çevriminden, yani durumdan oluşur. Şekil 9.6, bir 8085A "OUT port" komutunun zamanlamasını göstermektedir. Görüldüğü gibi, bir durum, saatin bir düşen kenarından diğerine kadar ölçülür. *T1* sırasında, durum bilgisi *S1* ve *S0* çıkışlarına, üst-sıralı adres bitleri *A8-A15* e ve alt-sıralı adres bitleri *AD0-AD7* ye gönderilir. Veri yolunun adresin alt 8 bitini içerdiğini göstermek üzere bir adres mandalı yetkilendirme sinyali ALE gönderilir. Bu sinyalin düşen kenarı, alt sekiz adres bitini bir harici mandal veya çevre aygıtına kilitlemek için kullanılır. Şekil 9.7, adres bitlerini ALE' yi kullanarak veri yolundan 8212' ye yüklemek için 8212 'nin çoklayıcı (demultipleksör) olarak nasıl kullanılacağını göstermektedir. 8212, genellikle adres hattı tamponu olarak gerektiğinden, fazladan bir donanım eklenmemiş olur. Sonuçta elde edilen adres, Şekil 9.3 teki 8080A'daki gibi çoğullanmamış sistemle aynıdır.



Şekil 9.6 8085A komut getirme makina çevriminin zamanlama diyagramı.



Şekil 9.7 8085A'nın alt-sıralı adresini veri yolundan çoklamak için adres mandalı olarak kullanılan 8212

Şekil 9.6'daki 8085A nın makina çevriminde *T2* 'nin başlamasından sonra $\overline{RD}$ alçağa gider. Bu sinyal, belleği, op kod baytını veri yolu üzerine koymak üzere yetkilendirir. $\overline{RD}$ dalgaşeklinin alçak seviyesinden *AD0-AD7* dalga şekline eğri ok, $\overline{RD}$ alçağa gitmesinin bellekten bir bayt çıkışı verilmesine sebeb olduğunu gösterir. $\overline{RD}$ yüksek seviyesinden *AD0-AD7* dalgaşekline giden ok (*T3* te), $\overline{RD}$ yüksek gitmesinin, belleğin baytı, veri yolundan kaldırmasına sebeb olduğunu (bellek çıkışlarını kayan hale getirdiğini.) belirtir. *T4* durumu komutun kodunu çözmek için kullanılır.

İkinci makina çevrimi sırasında, komut yürütüldüğünde çıkış verilecek portun adresini almak üzere bir başka bellek adresi ve $\overline{RD}$ darbesi gönderilir. *IO/*$\overline{M}$ alçak kalır.

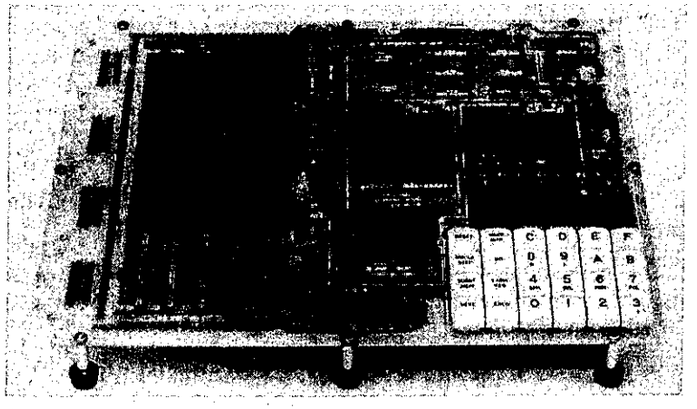
Üçüncü makina çevrimi sırasında, *OUT* komutu yürütülür. Port adresi, adres yolu üzerine konur (*AD0-AD7*). *IO/*$\overline{M}$, bir I/O işleminin yapıldığını göstermek üzere yükselir. Alt 8 adres biti *AD0-AD7* üzerinden kaldırıldıktan sonra, 8085A, akümülatördeki veri baytını veri yoluna koyar ve bir $\overline{WR}$ sinyali gönderir. $\overline{WR}$ sinyali, çıkış verisini port cihazına kilitler.

MİKROBİLGİSAYAR GELİŞTİRME KARTLARI : SDK-85

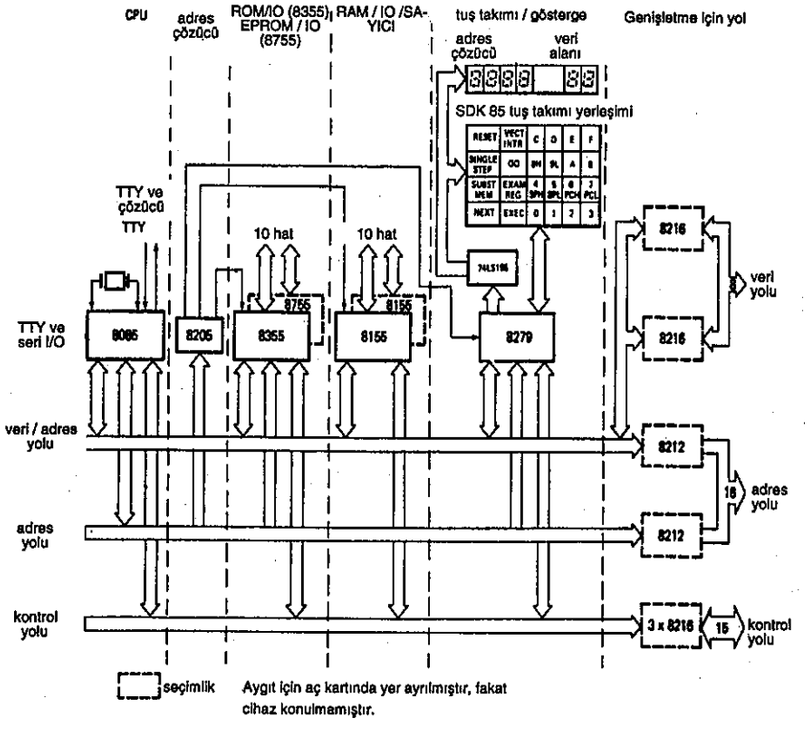
Yeni bir mikroişlemci tabanlı ürün için temel örnek geliştirilirken, temel CPU, RAM, ROM ve I/O portlarını birbirine bağlayarak en baştan başlamak çoğunlukla uygun değildir. Bu nedenle, üreticiler, bu elemanları hazır bağlanmış ve test edilmiş olarak içeren geliştirme kartları veya kolayca monte edilebilecek kitler satmaktadırlar. Bu kitlerden biri Şekil 9.8' de gösterilen Intel SDK-85'tir.

Bu kit, bir 8085A CPU, 8155 RAM ve I/O, 8355 önceden programlanmış monitör ROM, 8279 klavye-gösterge bağdaştırma entegresi ve kod çözücüleri içerir. Ayrıca kartın üzerinde bir 24-tuşlu onaltılık klavye ve adres ve veri yolu içeriklerini onaltılık olarak görüntülemek üzere 7-parçalı göstergeler vardır. Bir başka 8155; bir başka 8355 veya 8755A; iki 8212 adres tamponu; ve dört 8216 veri ve kontrol yolu tamponları için kart üzerinde soketler tasarlanmıştır. Sistemi genişletmek için bu entegrelerin soketlere takılması yeterlidir. Kartın, üniversal PC kartına benzeyen, büyük ve boş bir kısmı da daha fazla bellek ve bağdaştırma devreleri için kullanılabilir.

ROM'daki monitör programı, bir teletayp veya onaltılık klavyeden komut veya veri girmenize imkan verir. Bir kaydedicinin veya bellek yerinin içeriğini inceleyebilirsiniz, programları yürütebilirsiniz ve bir programı her komutunu tek tek yürüterek tek-adımlayabilirsiniz. Bölüm 12'de, SDK-85'in, mikroişlemci-tabanlı bir terazinin temel örneği için nasıl kullanıldığını açıklayacağız.



(a)



Şekil 9.8 SDK-85 Mikrobilgisayar geliştirme kartı. *(a)* Fotoğraf. *(b)* İşlevsel blok diyagramı. *(Intel Corporation.)*

SDK-85 İÇİN 8085A SİSTEM ŞEMASI

8085A, CPU grubunun bütün parçalarının yanısıra bir seri port içerdiğinden, sistemi tamamlamak için bütün gerekli olanlar RAM, ROM ve paralel I/O portlarıdır. Basit bir sistem için gerekli entegre sayısını en aza indirgemek için, Intel, RAM la I/O yu veya ROM la I/O portlarını tek bir yonga üzerinde birleştiren parça aileleri tasarlamıştır. 8155/8156, 256 baytlık statik RAM; iki 8-bitlik paralel I/O portu, bir 6-bitlik I/O portu ve bir 14-bitlik sayıcı-zamanlayıcı içerir. 8355 aygıtı, 2048 baytlık ROM ve iki 8-bitlik paralel I/O portu içerir. 8755 aygıtı ise iki port ve 2048 baytlık EPROM'a sahiptir.

Şekil 9.8 'de gösterilen SDK-85 mikrobilgisayar geliştirme kartının, CPU, RAM, ROM ve port kısımlarının şeması Şekil 9.9'da gösterilmiştir. Bu kartın klavye ve gösterge kısmının şeması Bölüm 11 de gösterilmekte ve anlatılmaktadır. Şekil 9.9'daki şema, Şekil 9.3 teki 8080A sistemiyle karşılaştırılabilecek bir küçük sistem oluşturmak üzere iki 8755, iki 8155 ve bir 8085A'nın nasıl bağlanacağını göstermektedir. Bir 8205, adres hatlarındaki yonga seçme sinyallerinin kodlarını çözer. Hem 8155'ler hem de 8755'ler, *AD0-AD7* girişlerinde 8-bitlik dahili mandallar içerir. CPU dan gelen ALE, alt 8-adres bitini bu mandallara kilitler, böylece harici mandallara gerek kalmaz.

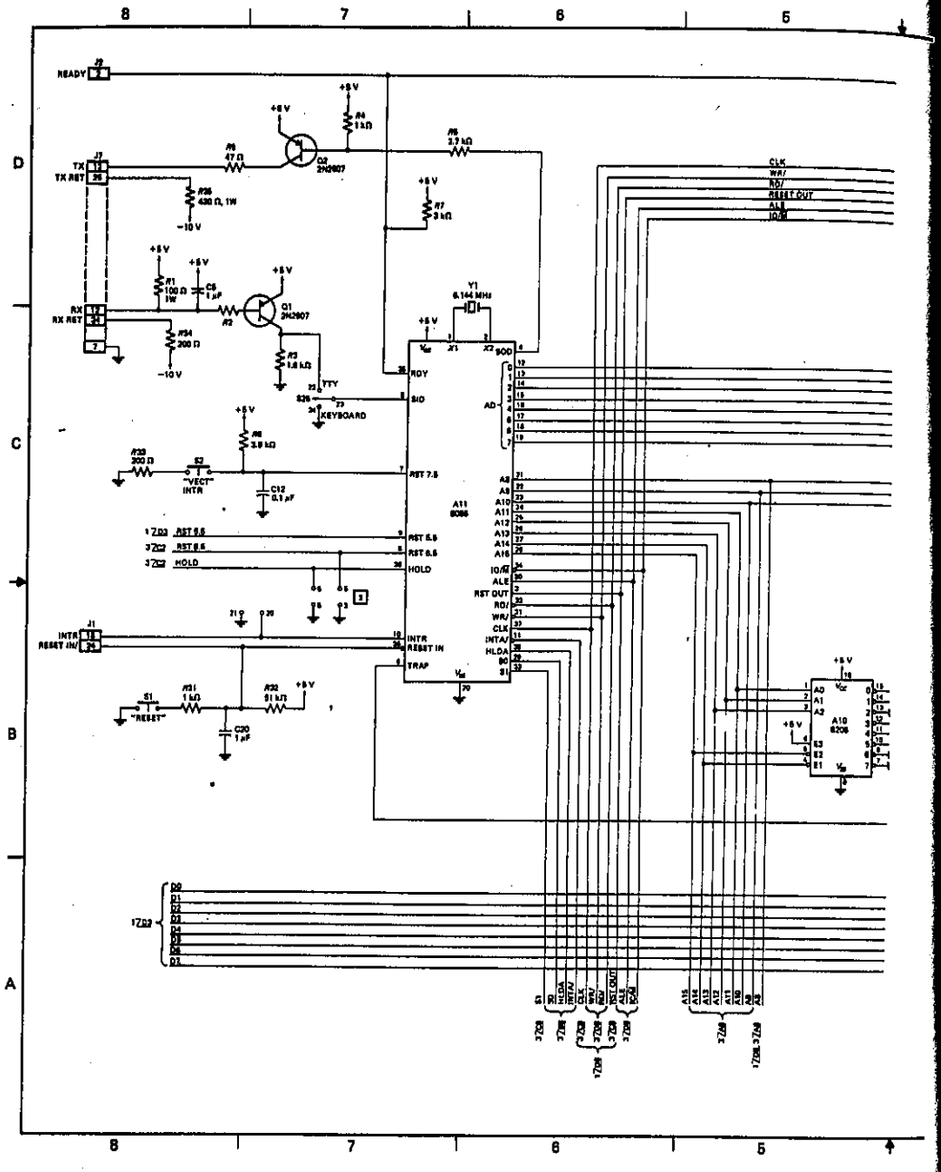
Bu aygıtları ayrıntılarıyla anlatmadan önce, Şekil 9.9'daki sistem şemasının bazı önemli özelliklerine işaret edelim. Önce, şemanın dışındaki *A* ile *D* arasındaki büyük harflere ve 1 ile 8 arasındaki sayılara dikkat edin. Bunlara zon koordinatları denir. Bir haritada istediğiniz yeri bulmanıza yardımcı olan sisteme benzer şekilde, bunlar şemadaki parçaları veya bağlantıları bulmanıza yardımcı olurlar. Örneğin, Şekil 9.9 daki *A15* ünitesi (bir 8755) zon *C3* tedir. Zon koordinatları ayrıca sinyal hatlarının bağlantı noktalarını tanımak için kullanılır. Örneğin, Şekil 9.9' da *A7* zonundaki veri yol hatları, yanlarında 1ZD3 atamasını taşımaktadır. Bunun anlamı, bu hatların sayfa 1 zon D3 teki karşılık gelen hatlara bağlandığı demektir.

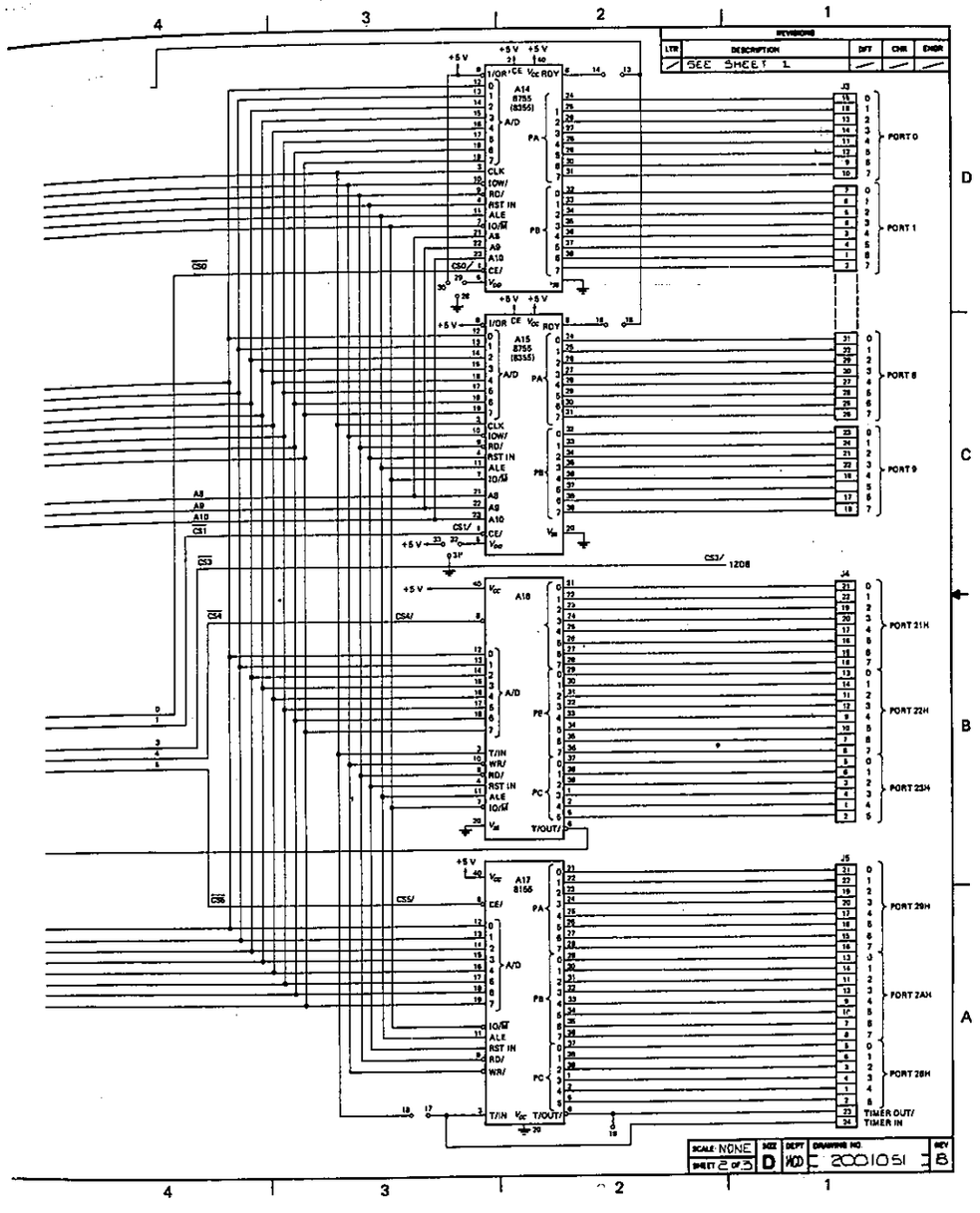
Diğer bir özellik de, Şekil 9.9' da *A1*, *B1*, *C1* ve *D1* zonlarındaki *J3*, *J4* ve *J5* etiketli birbirine bağlı kutulardır. Bu kutular yassı kabloların takılabileceği bağlantı uçlarıdır. Küçük kutulardaki sayılar bağlantı bacak numaralarıdır.

Şekil 9.9' daki bir diğer standart şematik özellik de, *A3* zonunda 18 ve 17 etiketli bağlantılara benzer küçük bağlantılardır. Bunlar, kullanıcı tarafından yapılabilecek ara bağlantı ve tel-sargılı bağlantılardır. Örneğin 18 noktasını 17 noktasına bağlamak, sistem saatini 8155 zamanlayıcı girişine bağlar.

8755A/8355

8755, 2K baytlık EPROM, iki programlanabilir paralel port ve iki veri yönü kaydedicisini içerir. $\overline{CE}$ alçak ve $IO/\overline{M}$ girişi alçak ise EPROM a ulaşılır. *AD0-AD7* üzerinden *A0-A7* gönderilir ve ALE nin düşen kenarı ile kilitlenir. *A8-A10* ayrıca girer.





Şekil 9.9 8085A mikrobilgisayar sistemi şeması. (Intel Corporation.)

8755A'da *I/O* portları veri ve yönü kaydedicileri, $\overline{CE}$ alçak, *IO/M* yüksek veya $\overline{RD}$ veya $\overline{IOW}$ alçak iken alt iki adres biti *A1* ve *A0* tarafından adreslenir. Şekil 9.10, 8755A daki iki port ve iki veri yönü kaydedicinin dahili adreslerini göstermektedir.

AD_1	AD_0	SEÇİM
0	0	**PORT *A***
0	1	**PORT *B***
1	0	**PORT *A*** VERİ YÖNÜ KAYDEDİCİSİ
1	1	**PORT *B*** VERİ YÖNÜ KAYDEDİCİSİ

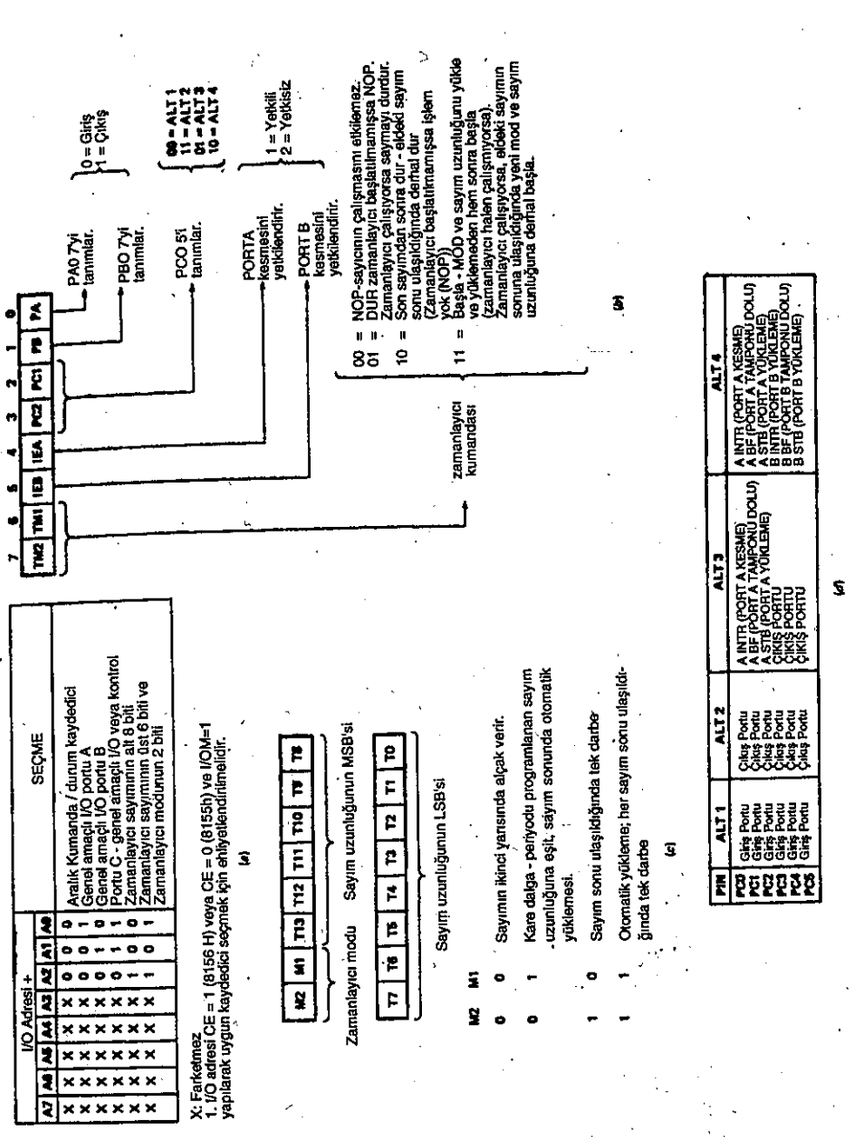
Şekil 9.10 Portlar ve veri yönü kaydedicileri için 8755A nın dahili adresleri. (Intel Corporation.)

Bir resetten sonra, 16 port hattının tamamı giriş olarak programlanır. Belli bir uygulamada, eğer bu hatlardan bazılarının çıkış olması isteniyorsa, bunlar çıkış olarak ilklendirilmelidir. 16 hattın herbiri tek tek giriş veya çıkış hattı olarak programlanabilir. Bir port hattını çıkışa programlamak için, veri yönü kaydedicisinde karşılık gelen bite bir 1 gönderilir. Örneğin, PA0 hattını bir çıkış olarak programlamak için, port *A* veri yönü kaydedicinin bit 0 ına bir 1 yazılmalıdır. Port *A* nın tamamını çıkış, port *B7-4* ü giriş ve port *B3-0*'ı çıkış yapmak için, sistem başlangıç adresi 00H olan bir 8755A da aşağıdaki çeviri dili komut dizisi kullanılacaktır.

```
MVI A, 11111111B ; hepsi çıkış için veri yönü baytı
OUT 02           ; veri yönü kaydedici A (DDRA) ya gönder
MVI A, 00001111B ; B7-4 giriş ve B3-0 çıkış için veri yönü baytı
OUT 03           ; DDRB ye gönder
```

Bunun gibi programlanabilir bir cihazın avantajı, çeşitli giriş ve çıkış hattı yapılarının oluşturulabilmesidir. Bazı uygulamaların gerektirdiği gibi, hatlar, program akışı içinde giriş ve çıkış olarak ileri geri değiştirilebilirler.

8155'in *I/O* portlarına ve dahili kaydedicilerine $\overline{CE}$ alçak, $\overline{RD}$ veya $\overline{WR}$ alçak ve *IO/M* yüksek iken alt adres bitleri üzerinden erişilebilir. Şekil 9.11a, 8155'teki her kaydedici ve portun dahili adreslerini göstermektedir. Her birinin sistem adresini elde etmek için, verilen dahili adresi, aygıtın sistem tabanına yani başlangıç adresine eklemeniz yeterlidir. Örneğin, Şekil 9.9 da A16 olarak etiketlenmiş 8155'in, 20H adresinden başlamak üzere, kodu çözümlenmiştir.



Şekil 9.11 8155. *(a)* Dahili adresler. *(b)* Kumanda kelimesi formatı. *(c)* Zamanlayıcı modları. *(d)* Port C alternatifleri. (Intel Corp.)

Bu nedenle, bu cihaz için, komut/durum kaydedici, 20H adresine; port *A*, 21 H adresine; port *B*, 22H adresine; port *C*, 23H adresine; zamanlaycı alt kaydedici, 24H adresine ve zamanlayıcı üst ve mod kaydedici, 25H adresine sahiptir.

Bir resetten sonra, bütün port hatları girişe ayarlanmıştır. Portları çıkışlar olarak ilklendirmek için, komut kaydedicisine bir komut kelimesi gönderilmelidir. Şekil 9.11b, 8155 komut kelimesinin formatını göstermektedir. Bu komut kelimesinin bit 0' ında bulunan bir 1, port *A* nın 8 hattının tamamını çıkış olarak tanımlar. Bit 0 daki bir 0, port *A* nın hatlarını giriş olarak tanımlar. Port hatlarının herbirinin, 8755 'te olduğu gibi, ayrı ayrı giriş ve çıkışa programlanamadığına dikkat edin. Benzer şekilde bit 1, port *B'* nin bütün 8 hattını giriş veya çıkış olarak tanımlar. Şekil 9.11c de gösterildiği gibi, port *C* için komut kelimesinin bit 2 ve bit 3' ü ile belirlenen dört seçenek vardır. Birinci seçenek, port *C* nin 6 bitinin tamamını giriş, ikinci seçenek de port *C* nin 6 bitinin tamamını çıkış yapar. Üçüncü ve dördüncü seçenekler ve komut kelimesinin bit 4 ve bit 5 inin kullanılışı, Bölüm 11'deki onaylı I/O kısmında anlatılmaktadır. Komut kelimesinin bit 6 ve bit 7 si zamanlayıcıyı başlatmak ve durdurmak için kullanılır.

8155'teki zamanlayıcı, bir 14-bitlik önceden ayarlanabilir aşağı sayıcıdır. Sistem saati veya başka bir harici darbe kaynağı, cihazın zamanlayıcı girişi (timer-in) (TI) ya bağlanır. Başlangıç değeri olan bir 14-bitlik ikilik sayım, iki iç sayım uzunluğu kaydedicisine yüklenir. Alt sıralı sayım uzunluğu kaydedicisi dahili 04H adresinde ve üst sıralı kaydedici 05H adresindedir. 05H adresindeki üst kaydedicinin en büyük değerlikli iki biti sayımın bir parçası değildir. Bu bitler, *zamanlayıcı çıkışı (timer-out) (TO)* için, Şekil 9.11c'de gösterilen dört çıkış modundan birini tanımlar. Bu dört mod şunlardır: 00, *TO* sayımın ikinci yarısında alçak olur; 01, *TO* sayımın ikinci yarısında alçak olur ve sayıcı 0 a ulaştığında sayım yeniden yüklenir (karedalga çıkış); 10, son sayım 0 a ulaştığında *TO* dan bir tek çıkış darbesi verilir; 11, sayım sonunda tek darbe verilir ve sayım yeniden yüklenir.

Sürekli kare dalga veya sürekli darbe modunda, zamanlayıcı çıkışının frekansı, zamanlayıcı giriş frekansının programlanan sayıma bölünmesine eşittir. Örneğin, 3,072 MHz lik girişten 1 KHz lik kare dalga çıkışı elde etmek için, zamanlayıcıya 3072_{10}, yani 0C00H, yüklenmeli ve sürekli kare dalga modu için mod bitleri 01'e set edilmelidir. Zamanlayıcı üst, alt ve mod kaydedicilerini yüklemek için kullanılacak çeviri dili program dizisi aşağıda gösterilmiştir. Bu dizi, ayrıca, zamanlayıcıyı başlatan ve port *A*'yı girişe, port *B'*yi çıkışa ve port *C'* yi çıkışa ayarlayan komut kelimesini de göndermektedir. Bu örnekteki cihazın taban adresi 20H'tir.


```
MVI A, 00H          ; zamanlayıcı sayımının alt baytı
OUT 24H             ; zamanlayıcı alt kaydedicisine gönder
MVI A, 01001100     ; kare dalga modu ve zamanlayıcının
OUT 25H             ; üst 6 bitini üst kaydediciye gönder
MVI A, 11001110     ; zamanlayıcıyı başlatan, port kesmelerini
                    ; yetkisiz kılan, port C = çıkış,
                    ; port B = çıkış, port A = giriş olarak
                    ; programlayan komut kelimesi
OUT 20H             ; komut kaydedicisine gönder.
```

Daha sonra göstereceğimiz gibi, bu zamanlayıcı, program zamanlamadaki gecikme döngülerinin yerine almak üzere mikroişlemci üzerindeki kesme girişi ile birlikte kullanılabilir. Eğer zamanlayıcıyı kullanmıyorsanız, sadece portları istenilen şekilde ilklendirecek komut kelimesini göndermeniz gerekecektir.

ADRES ÇÖZME

16 adres hattı olan bir mikroişlemci sistemi 2^{16}, yani 65536, yeri doğrudan adresleyebilir. Onaltılık olarak bu aralık 0000 ile FFFF arasındadır. Adres hatlarını, 16-bitlik bir sayıcının çıkışları olarak düşünürseniz, bu takdirde, her bitin yani adres hattının ikilik ağırlığı Şekil 9.12 deki kod çözücü çalışma sayfasında gösterildiği gibi olur. Adres çözmenin nasıl olduğunu göstermek için bunu ve Şekil 3.48 deki şemayı kullanacağız.

Şekil 3.48 deki devre, sekiz tane 2732 EPROM u bağlı olarak göstermektedir. Her aygıt 4096 veri baytı içermektedir. 4096 sayısı 2^{12} ye eşit olduğundan, aygıttaki 4096 bayttan birini adreslemek için 12 adres hattı (*A0* ile A11 arası) gerekmektedir. Bu hesabı anlamıyorsanız, bir sayıcının 0' dan 4095_{10} e, yani 0000H'dan 0FFFH'ye, kadar 4096 durumu sayabilmesi için kaç tane biti olması gerektiğini düşünün.

Şekil 3.48 deki her 2732'nin, o cihazda adreslenen baytın veriyoluna konulması için, bir $\overline{CS}$ girişi vardır. Buradaki önemli nokta, herhangi bir anda sadece bir aygıtın $\overline{CS}$ girişinin alçak olmasını sağlamaktır. Bu, 74LS138 ile gerçekleştirilir. Eğer, 74LS138, $\overline{G2A}$ ve $\overline{G2B}$ girişlerini alçak ve *G1* girişini yüksek yaparak yetkilendirilirse, bu takdirde 74LS138 in çıkışlarından sadece biri alçak olacaktır. Alçak olacak çıkış, *C, B* ve *A* seçme girişlerine uygulanan 3-bitlik adresle belirlenir. Örneğin, *CBA* = 000 ise, *Y0* çıkışı alçak ve bütün diğer

çıkışlar yüksek olur. Eğer CBA = 010 ise, *Y2* çıkışı alçak, diğerleri yüksektir. Eğer CBA = 111 ise sadece *Y7* çıkışı alçaktır. Şekil 3.48 'de, sistem adres hatları A14, A13 ve A12, 74LS138 in CBA girişlerine bağlanmıştır. Bunu bildiğinize göre, sistemdeki her aygıtın adresini belirlemek için Şekil 9.12'deki çalışma sayfasını kullanabilirsiniz.

		HEX BASAMAK				HEX BASAMAK				HEX BASAMAK				HEX BASAMAK				HEX DENGİ
		A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	ADRES
blok 1	başlangıç	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	= 0000
	bitiş	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	= 0FFF
blok 2	başlangıç	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	= 1000
	bitiş	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	= 1FFF
blok 3	başlangıç	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	= 2000
	bitiş	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	= 2FFF
blok 4	başlangıç	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	= 3000
	bitiş	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	= 3FFF
blok 5	başlangıç	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	= 4000
	bitiş	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	= 4FFF
blok 6	başlangıç	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	= 5000
	bitiş	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	= 5FFF
blok 7	başlangıç	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	= 6000
	bitiş	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	= 6FFF
blok 8	başlangıç	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	= 7000
	bitiş	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	= 7FFF

kod çözücü adres girişleri

Şekil 9.12 Şekil 3.48 'deki şemadaki sekiz 2732 EPROM un adreslerini çözmeyi gösteren kod çözücü çalışma sayfası.

Şekil 9.12'deki çalışma sayfası, bağlı olan adres hatlarının altında bütün mümkün CBA giriş birleşimlerini göstermektedir. Adres hattı *A15*, 0 olarak gösterilmiştir. Gerçekte, *A15* bu sistemde 'farketmez' dir, çünkü kod çözücüye veya aygıtlara bağlanmamıştır. Ancak, en basit adresleri vermesi için, *A15* 'i 0 seçtik. Çalışma sayfasından, Şekil 3.48' deki ROM 0 ın, eğer *A15*, *A14*, *A13* ve *A12* 'nin hepsi de alçaksa, yetkilendirildiğini görebilirsiniz. ROM 0 daki ilk yeri adreslemek için *A0* ile *A11* arası da alçak olmalıdır. Bu nedenle, ROM 0'ın başlangıç adresi 0000H' dir. ROM 0 ın bitiş adresi, *A0* ile *A11* arasının yüksek olduğu adrestir. *A12* ile *A15* arası hâlâ alçak olduğundan, ROM 0 ın bitiş adresi 0FFFH dir. ROM 0'ın adres aralığı 0000H den 0FFFH ye kadardır (4K-baytlık bir blok) denir.

Eğer $A15 = 0, A14 = 0, A13 = 0$ ve $A12 = 1$ ise ROM 1 yetkilendirilmektedir. ROM 1' in başlangıç adresi için *A0* ve *A11* arası alçaktır. Bu nedenle ROM 1 in başlangıç adresi 1000H' dir. ROM 1' in bitiş adresi *A0* ile *A11* arası yüksekken olan adres olup, 1FFFH' ye eşittir. Şekil 9.12 deki çalışma sayfasını kullanarak, diğer altı elemanın aralıklarının 2000H'den 2FFFH'ye, 3000H'den 3FFFH'ye, 4000H'den 4FFFH'ye, 5000H'den 5FFFH'ye, 6000H'den 6FFFH'ye

ve 7000H' den 7FFFH ye olduğunu görebilirsiniz. Bazıları, *A12*, *A13* ve *A14* ü 4K lık bellek blokları sayıyor diye düşünmeyi tercih ederler.

BİR 8080A SİSTEMİNDE ADRES ÇÖZME

Şekil 9.3 teki 8080A sisteminde adreslemenin analizini yapmak için her cihazda saklanan kelimelerin büyüklüğünü ve sayısını belirleyerek işe başlayın. Bunu yapmanın kolay bir yolu, her aygıta bağlı adres ve veri hatlarının sayısını bir kenara yazmaktır. Örneğin, 2716 ların her birine sekiz veri hattı bağlıdır. O halde, bunlar 8-bitlik kelime (bayt) çıkışı vermektedirler. 2716 ların 0 dan *A10* a kadar 11 tane adres hattı vardır. Bu nedenle her biri 2^{11}, yani 2048_{10} kelime içerir.

Şekil 9.3'teki her RAM ın dört veri hattı vardır, o halde her biri 4-bitlik kelime (nibıl) çıkışı verir. Her birinin 10 adres hattı (*A0-A9*) vardır, o halde herbiri 2^{10}, yani 1024_{10}, kelime içerir.

Şimdi Şekil 9.12 dekine benzer başlıkları olan bir çalışma sayfası hazırlayın ve bunu her kod çözücüye hangi adres hatlarının bağlanacağını belirlemek için kullanın. 3205 ROM kod çözücüsü ile başlayacağız.

3205, daha önce anlattığımız 74LS138 e benzer şekilde işlev görür, fakat yetki girişlerinde farklı etiketler vardır. E1 ve $\overline{E2}$ aktif alçak, E3 ise aktif yüksektir. ROM kod çözücüsünün E3 ü V_{cc} ye bağlanarak sürekli yüksek tutulmuştur. $\overline{E2}$ ise sistem MEMR sinyaline bağlıdır. Böylece ROM kod çözücü, sadece bir bellek okuma işlemi için yetkilendirilir. Bu, ROM a zarar verebilecek ve kaza ile yapılmaya çalışılacak bir yazma işlemini engeller. E1, sistem adres hattı *A14* e bağlıdır, bu nedenle çalışma sayfasında bu bitin altına bir 0 yazın. Bu sistemde *A15* hiçbir yere bağlı değildir, bu nedenle *A15* bir 'farketmez' dir. Çalışma sayfasında bu bitin altına bir 0 yazın. *A13*, *A12* ve *A11* sırasıyla kod çözücünün *C*, *B* ve *A* girişlerine bağlanmıştır. Çalışma sayfasında, uygun bitlerin altına bu adres hatlarının bütün mümkün birleşimlerini yazın. Her kod çözücü çıkışı için adres aralığı, daha önceki örnekte yapıldığı gibi, doğrudan bu sayfa üzerinden okunabilir. Birinci ROM un başlangıç adresi *A11* ile *A15* arası 0 ve *A0* ile *A10* arası 0 iken elde edilir. Bu blok *A11* ile *A15* arası 0 ve A0 ile A10 arasının hepsi 1 olduğunda biter. Hex olarak bu aralığın karşılığı 0000H ile 07FFH dir. İkinci ROM kod çözücüsü çıkışı, $A15=0$, $A14 = 0, A13 = 0, A12 = 0$ ve $A11 = 1$ iken yetkilendirilir. Bu ikinci çıkışı yetkilendirecek başlangıç adresi 0800H dir. Bu çıkışın aralığı 0800H ile 0FFFH arası olur. Diğer çıkışların aralıkları 1000H'den 17FFH ye, 2800H den 2FFFH ye, 3000H den 37FFH ye ve 3800H den 3FFFH ye kadardır.

Şekil 9.3'teki RAM kod çözücüsünün $\overline{E1}$ yetki girişi sistem reset hattına bağlıdır. Bu bir reset işlemi sırasında RAM'ın yetkilendirilmemesi için-yapılmıştır. $\overline{E2}$ toprağa bağlanarak sürekli yetkili kılınmıştır. E3 yetki girişi *A14* e bağlanmıştır. E3, aktif yüksek yetki girişi olduğundan, bu kod çözücüyü yetkilendirmek için A14 ün yüksek olması gerekir. Şekil 9.12'ye benzer bir çalışma sayfasında *A14'* ün altına bir 1 yazın. RAM kod çözücüsünün *C*, *B* ve *A* girişlerine, sırasıyla, *A12*, A11 ve *A10* sistem adres hatları bağlanmıştır. Şekil 9.12'dekine benzer şekilde, çalışma sayfası üzerinde, sırasıyla, bu üçünün bütün mümkün birleşimlerini yazın. *A13* ve *A15* 'farketmez' dir; bu nedenle çalışma sayfasında bu sütunlara 0 yazılabilir. İlk RAM blogu 4000H adresinde başlayacak ve 43FFH adresinde bitecektir. Diğer RAM kod çözücüsü çıkışlarının aralıkları 4400H'den 47FFH' ye, 4800H' den 4BFFH'ye, 4C00H 'den 4FFFH' ye, 5000H 'den 53FFH 'ye, 5400H 'den 57FFH' ye, 5800H' den 5BFFH' ye ve 5C00H'den 5FFFH 'ye kadardır.

PORT KODU ÇÖZME: BELLEK HARİTALANMIŞ I/O VE DOĞRUDAN I/O

BELLEK HARİTALANMIŞ I/O Bir bellek haritalanmış I/O sisteminde, port aygıtlarının adreslerinin, daha önceki ROM ve RAM adreslerinin kod çözülmesinde yapıldığı gibi, kodları çözülür. RAM kod çözücüsünün kullanılmayan çıkışları, port entegrelerini yetkilendirmek için kullanılabilir. Örneğin, Şekil 9.3'teki 8205 RAM kod çözücüsünün 05, 06 ve 07 çıkışları üç port entegresinin $\overline{CS}$ girişlerine bağlanabilir. Bu takdirde bu portların 16-bitlik tam adresleri 5400H, 5800H ve 5C00H olacaktır. Bu yaklaşımın avantajı, bellekle ilgili her komut, porta veri gönderebilir veya porttan veri okuyabilir. Örneğin, HL yazmacı 5400H'ye ilklendirilmişse, MOV D,M komutu, veriyi doğrudan porttan okuyup *D* kaydedicisine koyar. CPU, porta herhangi bir bellek yeriymiş gibi davranır. Bellek haritalanmış I/O'nun dezavantajları: her porta yazma veya porttan okuma tam 16-bitlik adres gerektirmektedir ve portlar sistem adres alanının bir kısmını kullanmaktadır.

DOĞRUDAN I/O 8080A ve 8085A gibi bazı mikroişlemciler, giriş portları ve çıkış portları için farklı adres alanları ayırırlar. Bu, doğrudan I/O olarak bilinir. Bu adreslere, 8080A ve 8085A'da 'IN port' ve 'OUT port' komutlarıyla erişilir. Bir 8080A veya 8085A bir 'IN port' veya 'OUT port' komutu yürüttüğünde, alt sekiz adres hattına bir 8-bitlik port adresi gönderir. (Not : Bu 8-bitlik port adresi, aynı zamanda üst sekiz adres hattına da kopyalanmıştır. Örneğin OUT 29H komutu, adres taşıtında 2929H oluşturur.) $\overline{I/OR}$ ve $\overline{I/OW}$ gibi kontrol sinyalleri, taşıttaki adresin bir port için olduğunu, bir bellek yeri için olmadığını, belirtir. Bu kontrol sinyalleri, istenilen port aygıtını açmak için kullanılmaktadır. 8-bitlik bir adres gönderildiği için, 256 taneye kadar giriş portu ve 256 taneye kadar çıkış portu $\overline{CS}$ sinyali, 3205 gibi elemanlar kullanarak, kodu çözülebilir. Giriş portu kod çözücüleri $\overline{I/OR}$ ile ve çıkış portu kod çözücüleri $\overline{I/OW}$ ile yetkilendirilir.

Doğrudan I/O 'nun avantajı, her IN veya OUT komutunun sadece bir tek baytlık adres gerektirmesi ve portlar için bellek alanının kullanılmamasıdır. Dezavantajları ise, daha fazla kontrol sinyalinin gerekmesi ve porta erişimde çok fazla komut kullanılamamasıdır.

Sadece birkaç portun seçilmesi gereken Şekil 9.3' teki gibi küçük sistemlerde, adres hatları, port aygıtlarını yetkilendirmek için doğrudan kullanılırlar. Bu tekniğe lineer seçme denir. Doğrudan I/O adresleri 8 bit genişliğinde olduğundan, 8 taneye kadar giriş ve 8 taneye kadar çıkış portu seçilebilir.

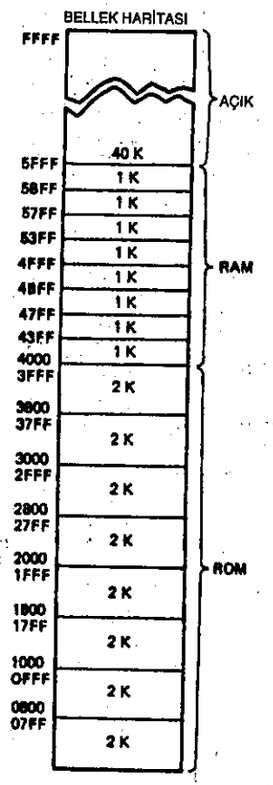
Örneğin, Şekil 9.3'teki sistemde, *A2*, 8251 port aygıtının $\overline{CS}$ girişine bağlanmıştır. $\overline{I/OR}$, 8251'in $\overline{RD}$ girişine ve $\overline{I/OW}$ 'de $\overline{WR}$ girişine bağlanmıştır. Eğer *A2* alçak ve $\overline{I/OR}$ alçak ise aygıt okuma için seçilir. Eğer *A2* alçak ve $\overline{I/OW}$ alçak ise cihaz yazma işlemi için seçilir. *A0*, aygıtın CONTROL /$\overline{DATA}$ (C/$\overline{D}$) girişine bağlanmıştır. Eğer C/$\overline{D}$ yüksek ise, aygıta gönderilen bir kelime aygıtın dahili kontrol kaydedicine gider. Eğer C/$\overline{D}$ alçak ise, gönderilen bayt porttan dışarı çıkar. Şekil 9.3'teki 8251A için, kontrol kaydedicisini adreslemek için adres taşıtının alt 8 bitinde gerekli kelime XXX110X1 ve seri giriş-çıkış portu için XXX110X0'dır. 8255A ların hiçbirinin 8251A ile aynı zamanda seçilmemesi için A3 ve A4 bitleri 1 olmalıdır. Eğer farketmez bitlerinin hepsi 1 yapılırsa, bu takdirde kontrol port adresi FBH ve seri çıkış port adresi FAH olur.

Şekil 9.3' te 1 numaralı 8255 in $\overline{CS}$ ı, sistem adres hattı *A3*'e bağlanmıştır. Eğer *A3* alçak ve $\overline{RD}$ veya $\overline{WR}$ alçak ise cihaz yetkilendirilir. *A0* ile *A1*, aygıttaki üç porttan birini veya kontrol kaydedicisini seçmekte kullanılmaktadır. Cihazın işgal ettiği dört adresten birincisi XXX10100 dır. Farketmezler 1 seçilirse, adres F4H olur. Entegredeki diğer adresler F5H, F6H ve F7H'dir.

Şekil 9.3'teki 2 numaralı 8255'in $\overline{CS}$ ı, sistem adres hattı *A4*'e bağlanmıştır. Bu programın işgal ettiği dört adresten birincisi XXX01100, yani farketmezler 1 seçilirse ECH; diğer üç adres de EDH, EEH ve EFH dir.

BELLEK HARİTALARI

Önceki kısımlarda, mikrobilgisayar sistemlerinde adres çözme yöntemleri anlatılmış ve belli bir sistemde her aygıtın adres aralığını belirlemek için bir çalışma sayfası kullanılmıştı. Bir sistemdeki aygıtların adres aralıkları genellikle bir *bellek haritası (memory map)* olarak özetlenir. Bu, sistemin bellek kullanışının bir bakışta görülmesini sağlar. Şekil 9.13, Şekil 9.3' teki sistemin bir bellek haritasını göstermektedir.



Şekil 9.13 Şekil 9.3'teki 8080A sisteminin bellek haritası..

BİR 8085A SİSTEMİNDE ADRES ÇÖZME

BELLEK Adres çözmede daha fazla alıştırma yapmak için, şimdi de Şekil 9.9 da gösterilen SDK-85 sisteminde kod çözmeyi inceleyeceğiz. Başlangıç olarak, daha önceki kısımlarda yapıldığı gibi bir adres çalışma sayfası çizin. Her biri belirlendikçe değerini yerine yazın.

Şekil 9.9'daki adres çözücü için, E3 yetki girişi, +5V'a bağlanarak sürekli yetkilendirilmiştir. $\overline{E2}$ yetki girişi, adres hattı *A15* e ve $\overline{E1}$ yetki girişi *A14* 'e bağlanmıştır. Elemanın yetkilendirmek için her ikisinin de alçak olması gerektiğinden *A15* ve *A14* alçak olmalıdır. Adres çalışma sayfanızda bu iki biti alçak olarak işaretleyin. Sistem adres hatları *A13*, *A12* ve *A11*, sırasıyla 8205'in seçme girişleri *A2, A1 ve A0* a bağlanmıştır. Adres çalışma sayfanızda bu bitlerin altına 000' dan 111' e kadar ikilik kodları yazın. *A10*'dan *A0*'a kadar olan hatlar, örneğin 8355 ROM'daki baytları seçmek için kullanıldığından, çalışma sayfanızda *A10* ile *A0* arasındaki bitlerin altına 0 yazın. Çalışma sayfanızı Şekil 9.14'teki ile karşılaştırın. Çalışma sayfanızdan, adres hatlarının hepsi 0 olduğunda (0000H) , 8205' in 0 çıkışının seçildiğini görebilirsiniz. Bu 0 çıkışı, monitör program ROM'unu içeren *A14* numaralı entegre 8355'i seçer, o halde monitör programının başlangıç adresi, 0000H'dir.

HEX				HEX				HEX				HEX				HEX DENGİ ADRESİ	Bloğun Başlangıcı
A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	= 0000	1
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	= 0800	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	= 1000	3
0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	= 1800	4
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	= 2000	5
0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	= 2800	6
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	= 3000	7
0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	= 3800	8

kod çözücü adres girişleri

Şekil 9.14 SDK-85 için adres çözme çalışma sayfası.

Eğer *A13* = 0, *A12* = 0 ve *A11* = 1 ise 8205 in 1 çıkışı seçilir. Bu birleşimi oluşturan ilk adres 0800H dir. 8205'in 1 çıkışı *A15* entegresi olan ikinci 8355 ROM 'u seçer. O halde bu ROM un başlangıç adresi 0800H'dir. 8205'in 0 çıkışı ile 1 çıkış için başlangıç adresleri arasındaki fark 800H, yani 2048_{10} 'dir. Buna diğer bir bakış şekli, *A13*, *A12* ve *A11* i 2048 (2K) baytlık adres sahası blokları sayıyor diye düşünmektir.

Şekil 9.9 'daki her 8205 çıkışı için başlangıç ve bitiş adreslerini belirlemek üzere çalışma sayfanızı kullanın. 0 için 0000H' den 07FFH' ye; 1 için 0800H 'den 0FFFH' ye; 2 için 1000H' den 17FFH' ye; 3 için 1800H'den 1FFFH'ye; 4 için 2000H 'den 27FFH 'ye; 5 için 2800H'den 2FFFH'ye; 6 için 3000H 'den 37FFH 'ye ve 7 için 3800H'den 3FFFH'ye bulacaksınız.

3 çıkışı, Bölüm 11'de anlatılan 8279 klavye-gösterge bağdaştırma aygıtına gitmektedir.

4 çıkışı, bir 8155 olan A16 entegresini yetkilendirmektedir. O halde, bu aygıttaki 256 baytlık RAM ın başlangıç adresi 2000H dir. 4 çıkışı için yukarıda gösterilen aralığa göre, 2000H ile 27FFH arasındaki 2048-bitlik aralıkta bulunan herhangi bir adres aygıtı yetkilendirecektir. 8155, sadece 256 baytlık bir RAM içerdiğinden, sadece 2000H ile 20FFH arasındaki adresleri gerektirir. Ancak, 2100H ile 27FFH arasındaki herhangi bir adres de aygıtı seçecek ve temel 256 baytlık alanda bir adrese geri katlanacaktır. 2100H ile 27FFH arasındaki adres alanına çoğunlukla geri katlama (foldback) denir. Adres çalışma sayfanıza bakarsanız, bu geri katlamayı anlatmanın diğer bir yolu; *A13*, *A12* ve *A11* hatları *A16* entegresi 8155 i seçer; *A7* ile *A0* arası RAM daki 256 bayttan birini seçer; ve *A10*, *A9* ve *A8*, aygıta bağlantıları olmadığı için, 'farketmez' dirler şeklindedir.

8205 in 5 çıkışı, başlangıç adresi 2800H' de olan *A17* entegresi 8155' i seçer. 2800H ile 2FFFH arasındaki herhangi bir adres aygıtı seçecektir. Bu aygıt için 2900H ile 2FFFH arası geri katlama bölgesidir.

SDK-85 SİSTEMİNDE PORT ADRESLEME Bir 8080A veya 8085A, bir 'OUT port' veya 'IN port' komutu yürütünce, 8-bitlik port adresi, alt 8 adres hattına gönderilir. Aynı adres üst 8 adres hattına da kopyalanır. Şekil 9.9'daki gibi, 8085A port kodu çözmeyi izlerken bunu hatırlamalısınız, çünkü portları seçerken üst adres hatlarının kodu çözülmektedir. Üst adres hatlarını kullanmak daha kolaydır, çünkü bunlar, *A7* ile *A0* arasındakiler gibi çoğullanmamıştır.

SDK-85 gibi bir sistemde, port adreslerini belirlemek için evvelce yaptığınız gibi bir adres çalışma sayfası kullanabilirsiniz. Çalışma sayfasından, port aygıtına $\overline{CS}$ sinyalini üretmek için *A15* ile *A8* arasında bulunması gereken adresi belirleyebilirsiniz. Daha önce belirtildiği gibi, bu adresin aynısı *A7* ile *A0* arasındaki port adresi olmalıdır. Örneğin, Şekil 9.9'da, *A15* ile *A8* arasında hep 0 varsa, üst 8355 seçilir. Bu entegredeki portların başlangıç, yani taban, adresi 00H olur. Şekil 9.10 da gösterildiği gibi, 8355'in *A1* ve *A0* tarafından seçilecek dört dahili adresi vardır. Eğer *A15* ile *A8* arası 08H' ye eşitse, Şekil 9.9 daki *A15* entegresi olan ikinci 8355 seçilir. Bu nedenle, bu aygıtın taban adresi 08H'dir. *A1* ve *A0* tarafından seçilen dahili adres de eklenirse, port *A* için 08H adresi, port *B* için 09H, DDRA için 0AH ve DDRB için 0BH adresleri bulunur.

Eğer *A15* ile *A8* arası 20H içeriyorsa, *A16* entegresi olan 8155 seçilecektir. O halde bu aygıtın taban adresi 20H dir. 8155 hakkındaki önceki kısımda açık- landığı gibi, aygıttaki altı adres şöyle olacaktır: kumanda/durum kaydedicisi için 20H; port *A* için 21H; port *B* için 22H; port *C* için 23H; zamanlayıcı alt kaydedici için 24H; ve zamanlayıcı üst ve mod yazmacı için 25H. *A15* ile *A8* arası 28H içeriyorsa, *A17* entegresi olan 8155 seçilir. O halde bu aygıtın taban adresi 28H dir.

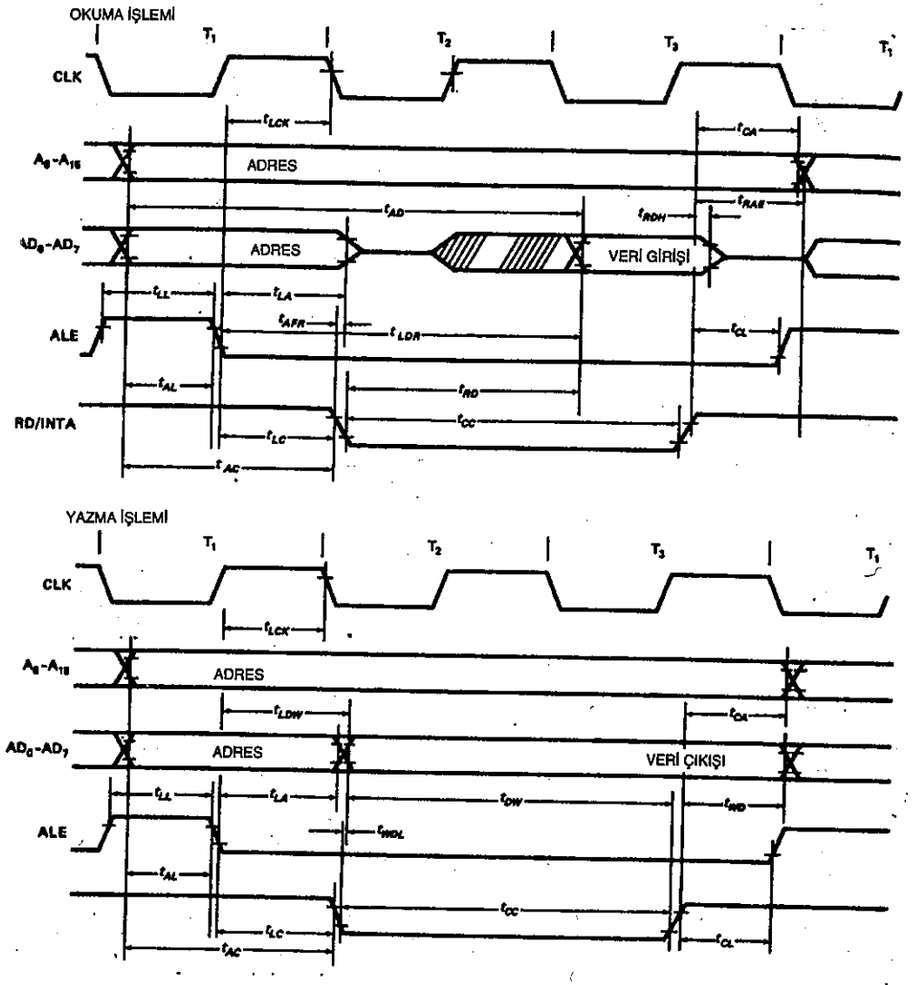
Mikrobilgisayar sistemlerinin arızasını bulurken, adres çözmenin çok iyi anlaşılmış olması çok önemlidir. Bir bellek aygıtı veya port, adres çözme devresiyle uygun şekilde yetkilendirilmiyorsa, bu bellek yongayı veya port doğru veriyi okuyamaz.

MİKROBİLGİSAYAR ZAMANLAMA PARAMETRELERİ

Çoğu mikrobilgisayar sisteminde zamanlama ilişkileri çok kritiktir. Şekil 9.15, 8085A için okuma çevrimi dalga şekillerini ve ilgili zamanlama değerlerini göstermektedir. Önce, saat çevrim periyodu T_{CYC} nin maksimum ve minimum sürelerine dikkat edin. 8085A-2 için olan 200 ns lik minimum T_{CYC}, aygıtın 5 MHz'den daha büyük dahili saat frekansı ile çalışamıyacağını belirtmektedir. 8085A' nın *X1* ve *X2* girişleri arasına bir kristal bağlayarak üretilen saat frekansı dahili olarak 2 ye bölünmektedir. Bu nedenle, 10 MHz lik bir kristal, 5MHz'lik bir dahili saat üretir. 2000 ns'lik maksimum saat çevrim periyodu, 500 KHz 'den daha küçük bir saat frekansı kullanılırsa, 8085' nin içindeki dinamik devrelerin, saat darbeleri arasında, saklanmış verileri kaybetmeye başlayacağını belirtmektedir.

Eğer 8085A'ya nizami çalışma aralığında bir saat yani kristal sağlanırsa, ALE nin genişliği gibi bir çok zamanlama değeri dahili olarak işlemci tarafından halledilecektir. Burada bizi en fazla ilgilendiren zamanlama değerleri, 8085A ile sistemdeki diğer aygıtlar arasındaki ilişkilerdeki zamanlama değerleridir. Örneğin, bir okuma işlemi sırasında, 8085A bir adres, bir ALE sinyali ve bir $\overline{RD}$ sinyali gönderir. Adresin gönderilmesinden T_{AD} kadarlık bir süre içinde ve $\overline{RD}$ alçağa düştükten sonra T_{RD} kadarlık bir süre içinde, adreslenen bellek yongası veya port, 8085A 'ya yol hattı üzerinden geçerli veri göndermelidir. Bir 3-MHz'lik 8085 için maksimum T_{AD} , 575 ns ve bir 5-MHz'lik 8085A -2 için 350 ns'dir. Başarılı bir çalışma için, adreslenen aygıtın adres erişim zamanı ile aygıtı seçmede kullanılan adres çözücünün yayılma gecikmesi toplamı bu T_{AD} den daha küçük olmalıdır. Şekil 9.9'daki gibi bir sistem için, bu, şu şekilde elde edilir. 8205 kod çözücünün maksimum yayılma gecikmesi 18 ns'dir. 8755A nın maksimum adres erişim zamanı 450 ns ve 8755A-2' ninki 330 ns'dir. Kod çözücü gecikmesi bunlara ilave edilirse, 8755A için 468 ns'lik ve 8755A-2 için 348 ns lik toplam erişim süresi elde edilir. Bu değerleri yukarıdaki T_{AD} değerleri ile karşılaştırdığınızda, 8755A'nın sadece 3-MHz'lik 8085A ile ve 8755A-2' nin hem 8085A hem de 8085A-2 ile doğru olarak çalışabileceğini görebilirsiniz. Bu, verilen bir sistemin doğru olarak çalışması için doğru sonek numaralı parçanın kullanılması gerektiğini bir defa daha göstermektedir. Örneğin, belli bir sistemde, doğru çalışacağını görmek üzere zamanlamayı kontrol etmeden arızalı bir 8755A-2' yi bir 8755A ile değiştiremezsiniz.

Buna benzer bir durumda kontrol edilmesi gereken diğer bir zamanlama parametresi T_{RD} 'dir. 8085A için maksimum T_{RD}, 300 ns ve 8085A-2 için 150 ns dir. Karşılık gelen $\overline{RD}$ erişim zamanları 8755A için 170 ns ve 8755A-2 için 140 ns dir. Bu, yine, daha hızlı 8755-2'nin 8085A-2 ile kullanılması gerektiğini göstermektedir.



Şekil 9.15 8085A zamanlama dalgaşekilleri ve ilgili zamanlama değerleri (bir sonraki sayfa). (Intel Corp.)

SEMBOL	PARAMETRE	8085A*		(ÖN BİLGİ)		BİRİM
		MİN.	MAX.	MİN.	MAX.	
t_{CYC}	CLK Çevrim Periyodu	320	2000	200	2000	ns
t_1	CLK Alçak Süresi (Standart CLK Yükleme)	80		40		ns
	CLK Yüksek Süresi (Standart CLK Yükleme)	120		70		ns
t_R	CLK Yükselme ve Düşme Zamanı		30		30	ns
t_1, t_2	Yükselen X1'den yükselen CLK 'ye	30	120	30	100	ns
t_{HCR}	Yükselen X1'den düşen CLK'ye	30	150	30	110	ns
t_{HCF}	Geçerli A8—15'den Kontrolun Yükselen Kenarında.	270		115		ns
t_{AC}	Geçerli A0—7'den Kontrolun Yükselen Kenarına	240		115		ns
t_{ACL}	Geçerli A0—15'den Geçerli Veri Girişine		575		350	ns
t_{AD}	READ (INTA) 'ın Ön Kenarından Sonra Adres Kayma		0		0	ns
t_{AFR}	ALE'nin Düşen Kenarından Önce Geçerli A8—15 +	115		50		ns
t_{AL}	ALE'nin Düşen Kenarından Önce Geçerli A0—7	90		50		ns
t_{ALL}	Geçerli Adresten Geçerli READY'ye		220		100	ns
t_{ARY}	Kontroldan Sonra Geçerli Adres (A8—15)	120		60		ns
t_{CA}	Kontrol Alçak Genişliği (RD, WR, INTA) ALE Kenarı	400		230		ns
t_{CC}	Kontrolun Düşen Kenarından ALE'nin Yükselen Kenarına					
t_{CL}	Kontrolun Düşen Kenarından ALE'nin Yükselen Kenarına	50		25		ns
	Geçerli Veriden WRİTE 'ın Arka Kenarına	420		230		ns
t_{DW}	HLDA'dan Yol Yetkilendirmeye		210		150	ns
t_{HABE}	HLDA'dan Sonra Yol Kayması		210		150	ns
t_{HABF}	Geçerli HLDA'dan CLK'nın Düşen Kenarına	110		40		ns
t_{HACK}	HOLD Tutma Süresi	0		0		ns
t_{HDH}	CLK'nın Düşen Kenarından HOLD Kurma Süresi	170		120		ns
t_{HDS}	INTR Tutma Süresi	0		0		ns
t_{INH}	CLK'nın Düşen Kenarından, INTR- RST ve TRAP	160		150		ns
t_{INS}	Kurma Süresi	100		50		ns
t_{LA}	ALE'den Sonra Adres Tutma Süresi	130		60		ns
t_{LC}	ALE'nin Düşen Kenarından Kontrolun Ön Kenarına	100		50		ns
t_{LCK}	CLK Yüksek Sırasında ALE Alçak		460		270	ns
t_{LDR}	Okuma Sırasında ALE'den Geçerli Veriye		200		120	ns
t_{LDW}	Yazma Sırasında ALE'den Geçerli Veriye	140		60		ns
t_{LL}	ALE Genişliği		110		30	ns
t_{LRY}	ALE'den Kararlı READY ve	150		90		ns
t_{RAE}	READ'in Düşen Kenarından Adresin Yeniden Yetkilendirilmesine		300		150	ns
t_{RD}	READ (veya INTA) dan Geçerli Veriye	400		220		ns
t_{RV}	Kontrolun Arka Kenarından, Bir Sonraki Kontrolun Ön Kenarına	0		0		ns
t_{RDH}	READ INTA 'Dan Sonra Veri Tutma Süresi	0		0		ns
t_{RYH}	READY Tutma Süresi	110		100		ns
t_{RYS}	CLK'nın Ön Kenarına READY Kurma Süresi	100		60		ns
t_{WD}	WRİTE'ın Arka Kenarından Sonra Geçerli Veri		40		20	ns
t_{WDL}	WRİTE'ın Ön Kenarından Geçerli Veriye					

Notlar: * Test Koşulları: tcyc = 320 ns (8085 A) / 200 ns (8085A—2) ; CL = 150 pF
+ A8—A15 adres özellikleri IO/M, S0 ve S1 için geçerlidir. Ancak, OF çevriminin T4–T6 sırasında IO/M S0 ve S1 kararlı iken, A8—A15 tanımsızdır.

Şekil 9.15 'in devamı

Özet olarak, adreslenen bellek aygıtının veya portun erişim süresi, işlemci tarafından gerekli tepki süresinden (T_{AD}, v.b.) daha az olmalıdır. Bu kuralın zedelenmesi, bazen doğru çalışan bazen çalışmayan sistemlerin en çok karşılaşılan sebebidir.

BEKLEME, TUTMA VE DURMA DURUMLARI

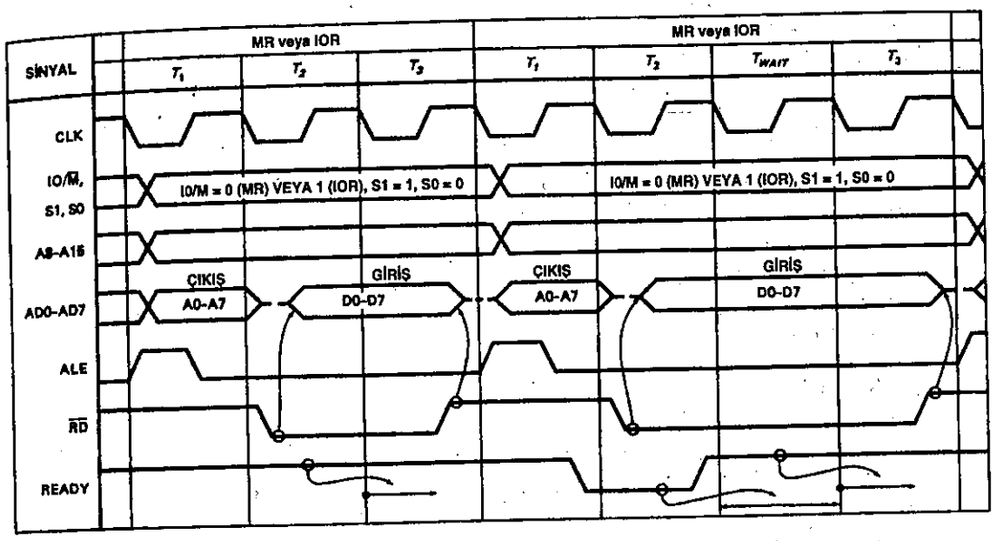
Şimdi 8080A ve 8085A'nın birkaç bacağının daha işlevlerini açıklayacağız.

BEKLEME DURUMLARI

Eğer bir 8080A veya 8085A CPU'nun *READY (hazır)* girişi doğru anda alçak yapılırsa, son makina çevriminin *T2*'sinden sonra, CPU bir bekleme durumu T_W 'ya girer. Şekil 9.16, bekleme durumlu ve bekleme durumsuz bir 8085A okuma makina çevriminin zamanlama dalgaşekillerini göstermektedir. Şekil 9.16'da altta bulunan dalgaşekli, READY girişinin etkisini göstermektedir. Eğer READY, CPU tarafından *T2* sırasında kontrol edildiğinde, yüksek ise, CPU doğrudan makina çevriminin *T3* durumuna ilerler. CPU tarafından *T2*'de kontrol edildiğinde, eğer READY alçak ise, bir bekleme durumu T_W eklenir. Eğer bekleme durumunda iken READY yükselirse, bekleme durumundan sonra, CPU, makina çevriminin *T3*'ü ile devam eder. Eğer READY hâlâ alçaksa, bir başka bekleme durumu eklenir. READY alçak olduğu sürece bekleme durumları eklenmeye devam edilir. Bekleme durumunda olduğunu belirtmek üzere, 8080A, *WAIT (bekleme)* çıkışını yükseltecektir. 8085A'nın WAIT çıkışı yoktur.

Şekil 9.16, bekleme durumu sırasında, adres yolunun, veri yolunun ve kontrol yolunun içeriklerinin sabit tutulduğunu göstermektedir. O halde, bekleme durumu, adreslenen bir bellek veya porta, veri yoluna geçerli veriyi koymak için ilave saat çevrim süresi sağlar. Bu özellik, daha uzun erişim zamanları olan daha ucuz bellek aygıtlarının kullanılmasına imkan verir. 74LS121 gibi bir tekkararlı multivibratör, daha yavaş aygıtın her adreslenişinde READY yi belli süreyle alçağa çekmek üzere 8080A'nın $\overline{MEMR}$ veya $\overline{MEMW}$, veya 8085A'nın $\overline{RD}$ veya $\overline{WR}$ sinyali ile tetiklenebilir. Tekkararlı devre, adreslenen aygıtın $\overline{CS}$ girişine gönderilen sinyalle yetkilendirilebilir. Bu, her okuma ve yazma işlemi sırasında bir bekleme durumunun eklenmesini engeller. 8755A gibi bazı aygıtlar, bir bekleme durumu istemek için dahili devre içerirler. Şekil 9.9 da, *D2* ve *C2* zonlarında, 8755A'nın açık-kanallı RDY çıkışlarının, 8085A nın READY girişine bağlanabileceğine dikkat edin. Eğer bu bağlantı yapılırsa, 8755A, her okunuluşunda, 8085A nın bir bekleme durumu eklemesine sebep olacaktır.

Şekil 9.15b'deki 8085A'nın zamanlama parametrelerine ve Şekil 9.15a' daki dalgaşekillerine bakın. Tanınması için, READY girişi, *T2* de CLK nın yükselen kenarından T_{RYS} kadar önce alçak yapılmalıdır. Bekleme durumundan çıkmak için, READY, T_W 'daki CLK nın yükselen kenarından T_{RYS} kadar önce yüksek yapılmalıdır.



Şekil 9.16 8085A'da bekleme durumlu ve bekleme durumsuz okuma makina çevrimi zamanlama dalgaşekilleri. (Intel Corporation.)

TUTMA DURUMLARI VE DOĞRUDAN BELLEK ERİŞİMİ

Bir 8080A veya 8085A'nın HOLD (tutma) girişini yükseğe çıkarmak, bir tutma durumu isteğinde bulunur. Tutma durumunda, adres, veri ve kontrol yolları, bir harici aygıt veya sistem tarafından kullanılabilmeleri için kayan hale getirilirler. Tutma istek sinyali, işlemci o andaki makina çevrimi için yolları kullanmayı bitirince HLDA çıkışında bir tutma alındı sinyali gönderecek şekilde senkronize edilmiştir .Bundan sonra işlemci yolları kayan hale getirir. Bu, makina çevriminde T3 'ün sonuna doğrudur. Eğer işlemci, dört, beş veya altı durum gerektiren bir makina çevrimi yapıyorsa, bu durumlar yollar kayan hale getirildikten sonra dahili olarak tamamlanır. Bundan sonra işlemci

oturup bekler. Dahili kaydediciler tazelenmeye devam edilir. HOLD alçak yapıldığında program bir sonraki makina çevriminin *T1*'i ile başlar.

Tutma durumunun bir uygulaması, doğrudan bellek erişimi (direct memory access: DMA) için kullanılmasıdır. DMA, bir harici sistem, 8080A veya 8085A nın alıp bellekte saklayabileceğinden daha hızlı veri baytları grubu gönderiyorsa kullanılır. 8257 programlanabilir DMA kontrolörü gibi bir aygıt, bir veri bloğu hazır olduğunda 8080A veya 8085A'ya bir HOLD istek sinyali gönderir. O andaki makina çevriminde yollarla işi bittiğinde, 8080A veya 8085A bir HLDA sinyali gönderir ve adres ve veri yollarını kayan hale getirir. Şekil 9.3'teki 8080A sisteminde, HLDA nın veri yolunu kayan hale getirmesi için HLDA çıkışı 8228'e bağlanmıştır. DMA kontrolöründen, 8228'in $\overline{\text{BUSEN}}$ girişine gönderilen bir sinyal 8228'in kontrol yolu çıkışlarını kayan hale getirir. Bir 8085A da bu dahili olarak yapılmaktadır. DMA kontrolörü, her üç yolu da kontrolu altında tutar ve belleğe bir veri bloğu yazmak için uygun adres ve kontrol sinyallerini sağlar. Veri aktarımı tamamlandığında, DMA kontrolörü HOLD girişini alçağa düşürür ve 8080A veya 8085A bir sonraki makina çevrimi ile devam eder.

DURMA DURUMU

Bir durma komutu HLT yürütüldüğünde, bir 8080A veya 8085A, bir sonraki makina çevriminin *T2*'sinden sonra durma durumuna (halt state) girer. Durma durumunda, işlemci durdurulur; adres ve veri yolları kayan hale getirilir. Durma durumundan çıkmak için üç yol vardır. Birincisi, 8085A' nın RESET IN girişindeki bir alçak, bütün sistemi reset eder ve program sayıcıya 0 yükler. 8080A için, 8224 'ün RESIN girişindeki bir alçak, sistemi reset etmek ve program sayıcıya 0 yüklemek üzere 8080A nın RESET girişine (ve diğer sistem elemanlarına) bir yüksek gönderir. Bir resetten sonra, 8080A/8085A program yürütmesi derhal 0000H adresinden başlar.

Durma durumundan çıkmanın ikinci yolu, HOLD girişini yüksek yapmaktır. Böylece işlemci bekleme durumuna girer, fakat HOLD girişi tekrar alçaldığında, CPU durma durumuna tekrar geri döner.

Durma durumundan çıkmanın üçüncü yolu, 8080A veya 8085A' nın kesme girişine bir kesme sinyali göndermektir. Bu yöntem, kesmeler, programda bir kesmeyi yetkilendir (enable interrupt: EI) komutu ile yetkilendirilmişse çalışır. Bir sonraki kısım kesmeleri ve onların kullanılışlarını anlatmaktadır.

Şimdilik sizin için önemli olan, bir programın sonunda HLT komutu kullandığınızda, CPU'nun programı yürüttükten sonra durduğunu hatırlamanızdır. Bundan sonra CPU'ya başka birşey yaptırmanın tek yolu, bir RESET, bir HOLD veya bir INT girişiyledir.

KESMELER (INTERRUPTS)

Bir kesme, mikrobilgisayar tabanlı sistemlerde, bazen veya rasgele olan giriş sinyalleriyle uğraşmak için bir yoldur. Örneğin, bir mikrobilgisayar sisteminin büyük bir kağıt üretme makinasını kontrol etmede kullanıldığını varsayın. Mikrobilgisayarın görevlerinden bir tanesi de, buhar kazanının basınç emniyet sınırını aşmamasını garanti etmektir. Eğer kazan basıncı emniyet sınırını aşarsa, mikrobilgisayar yakıtı derhal kesmeli ve basınç düşürme vanasını açmalıdır. İdeal olarak, mikrobilgisayar yaptığı işi kesmeli ve kazan patlamadan önce bu problemi çözmelidir.

8080A KESMELERİ

Böyle bir probleme hizmet vermenin en çabuk yolu bir kesme iledir. Örneğin, kazan basınç dönüştürücüsünün çıkış sinyali 8080A'nın kesme girişine (INT) bağlanabilir. INT girişi, programda bir EI komutuyla önceden yetkilendirilir. 8080A bir INT sinyali alınca, yürütülmekte olan komutu tamamlar ve sonra bir M1 bellek getirme makina çevrimi yapar. *T1* sırasında çıkış verilen durum kelimesi, bir kesme işleminin yer alacağını belirtir. *T1* sırasında DBIN çıkışı verilerek veri yolunun giriş modunda olduğu belirtilir. 8228, kesme alındı sinyali (interrupt acknowledge) INTA üretmek üzere durum kelimesini kullanır. Bu noktada, esas olarak, CPU, "Beni kestiğinize göre, şimdi ne yapacağımı söyleyin" demektedir. Veri yolu üzerine bir komut yüklenmesini beklemektedir. Bu harici donanımla yapılmalıdır.

8080A tasarımcıları, komut kümesine yeniden başlama (restart: RST) komutlarını ekleyerek bu işi oldukça kolaylaştırmışlardır. RST komutları, 1-baytlık koşulsuz çağırma komutlarıdır. Bir 3-bitlik adres, komutun *D3*, *D4* ve D5 bitlerinde içerilir. Sekiz mümkün yeniden başlama adresi vardır: RST0'dan RST7'ye. RST komutu, harici donanım tarafından veri yoluna yüklendiğinde, 8080A komutun kodunu çözer. RST komutu temelde bir çağırma komutu olduğu için, CPU, programa dönüş adresini yığında saklar. Bundan sonra, 0 ile 7 arasındaki RST numarasını, altyordam başlangıç adresinin *A3*, *A4* ve *A5* adres bitleri olarak kullanır. Diğer adres bitlerinin 0 olduğu varsayılır. Program sayıcı bu adresle yüklenir ve program yürütme buradan başlar. Örneğin, RST 7 komutu, geri dönüş adresini yığında saklar ve program sayıcıyı 0038H adresine atlatır. Sekiz RST adresi hex olarak 00, 08, 10, 18, 20, 28, 30 ve 38'dir.

Seçilen RST adresinden başlayarak, programcı, yakıtı kapatmak ve emniyet vanasını açmak için bir dizi komut yazar. Kesmenin programın neresinde olacağı bilinmediğinden, kesmeye hizmet veren altyordam, büyük bir ihtimalle bütün kaydedicileri yığına iterek başlayacaktır. Kazan kurtarıldıktan sonra, veriler yığından kaydedicilere atılarak ana programın yürütülmesine

kalınan yerden devam edilir. 8080A bir kesmeyi tanıyınca, kesme girişindeki bir sinyalin tekrar tekrar işlemciyi kesip durmasını engellemek için kesme girişini yetkisiz kılar. Kesmeye hizmet verildikten sonra, kesme girişini tekrar yetkilendirmek için bir EI komutu kullanılmalıdır. İşlemci bundan sonra yeni bir kesme sinyaline tepki gösterebilir.

Bir kesme komutu veri taşıtına basit bir 8212 devresiyle yüklenebilir. RST komutu, 8212'nin girişlerine tel bağlantılarıyla yapılabilir ve 8228'den gelen $\overline{\text{INTA}}$ sinyali, RST komutunu veri yoluna yüklemek için kullanılır.

8228 kullanan bir sistemde sadece bir kesme gerekiyorsa bu, $\overline{\text{INTA}}$ çıkışını 1kΩ luk bir dirençle +12V a bağlayarak gerçekleştirilebilir. Bu şekilde bağlandığında, 8080A bir INT sinyali aldığında, 8228 otomatik olarak RST7 komutunu veri yoluna yükler.

8085A KESMELERİ

8085A'nın tasarımcıları, çoklu-kesme yeteneğine olan ihtiyacın farkına varmışlardır. Bu nedenle, 8085A nın *INTR, RST 5.5, RST 6.5, RST 7.5* ve *TRAP* olmak üzere beş tane kesmesi vardır.

INTR girişi, 8080A nın INT girişine çok benzer şekilde çalışır. Bir resetten sonra INTR girişi yetkisiz kılınır. INTR girişini yetkilendirmek için bir EI komutunun yürütülmesi gerekir. Her komut çevriminin sonunda, 8085A, kesmelerinin yetkilendirilip yetkilendirilmediğini ve bir INTR kesmesi istenip istenmediğini kontrol eder. Eğer bu iki koşul yerine getirilmişse, 8085A kesmeleri yetkisiz kılar, bir kesme alındı sinyali $\overline{\text{INTA}}$ gönderir ve veri yolunu giriş moduna koyar. Dış donanım, daha önce 8080A için anlatıldığı gibi, veri taşıtına bir RST veya CALL komutu koymalıdır. RST veya CALL komutuna cevap olarak, 8085A geri dönüş adresini yığına iter, RST veya CALL adresine gider ve bu adresten kesme hizmet altyordamını yürütmeye başlar. Kesme hizmet altyordamının sonunda, bir sonraki kesmenin tanınması için kesmeleri yeniden yetkilendirmek üzere bir EI komutu kullanılır. Bundan sonra, bir RET komutu, saklanmış geri dönüş adresini yığından çekerek yürütmeyi ana programa verir.

8085A'nın *RST 5.5, RST 6.5, RST 7.5* ve *TRAP* kesme girişleri bir RST komutu için gerekli harici donanım ihtiyacını ortadan kaldırır. Bu dört girişten herhangi birine bir kesme sinyali uygulanırsa ve bu girişler yetkilendirilmişse ve maskelenmemişse; 8085A, şu şekilde davranır. Aynı kesme sinyalinin kesmeye devam etmesini engellemek için kesmeler yetkisiz kılınır. Geri dönüş adresi yığına itilir. Program sayıcı, o kesme girişine ait özel RST adresi ile yüklenir ve yürütme bu yüklenen adresten devam eder. Bu kesmelere ait dört adres; *TRAP* için 24H; *RST 5.5* için 2CH; *RST 6.5* için 34H ve *RST 7.5* için 3CH

dir. Her kesmeye hizmet verecek altyordamın başlangıcı, bellekte karşılık gelen adrese konur. Hizmet yordamlarında, TRAP dışında, kesmelerin tanınabilmesi için kesmeler EI komutuyla yetkilendirilmelidir. Bu çoğunlukla yordamın sonuna doğru yapılır. Yordamın sonundaki bir RET komutu, geri dönüş adresini yığından çekerek yürütmeyi ana programa geri verir.

8085A'nın beş kesme girişinin sabit bir hizmet önceliği vardır. Bu, iki kesme sinyali aynı zamanda gelirse, en öncelikli kesmeye hizmet verilir, demektir. Girişlerin öncelikleri yukarıdan aşağı doğru olmak üzere *TRAP, RST 7.5, RST 6.5, RST 5.5* ve *INTR* şeklindedir.

MASKELEME VE YETKİLENDİRME

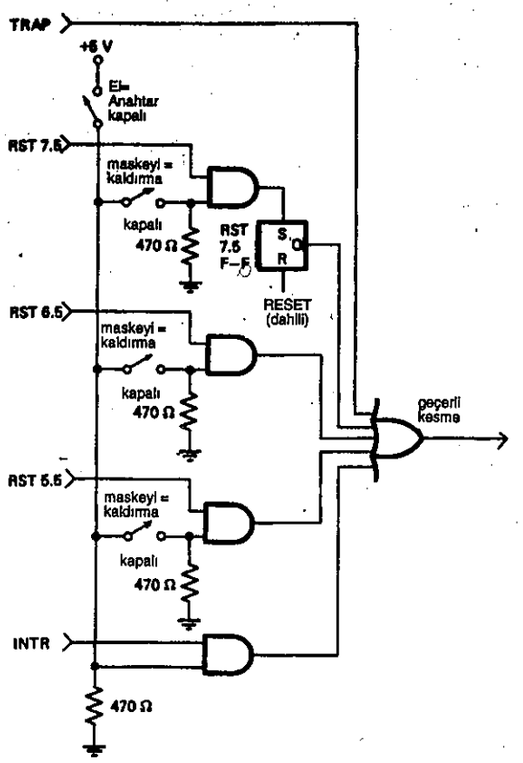
TRAP kesme girişi sadece en yüksek önceliğe sahip değildir, ayrıca hiçbir şekilde yetkisiz kılınamaz veya maskelenemez. 8085A, TRAP'taki bir kesme isteğini her zaman tanır. Bu nedenle, bu giriş genellikle güç kesilmesi durumunda verileri çabucak saklamak için veya erimeye başlayan bir atomik reaktör gibi acil durumlara cevap vermek üzere kullanılır.

INTR girişi, bir resetten sonra, bir kesmeyi yetkisiz kıl (disable interrupt: DI) komutundan sonra veya kesme isteğinden sonra yetkisiz kılınır, EI komutuyla yetkilendirilir.

Bir resetten sonra, *RST 7.5, RST 6.5* ve *RST 5.5* kesme girişleri hem yetkisiz kılınır hem de maskelenir. Bu girişler EI komutuyla yetkilendirilir ve *kesme maskesini set et (set interrupt mask: SIM)* komutuyla istenilenlerin maskesi kaldırılabilir. Şekil 9.17, beş 8085A kesme girişinin herbirinin işlemci için geçerli bir kesme oluşturması için neyin gerektiğini canlandırmanıza yardımcı olacak bir basit mantık diyagramını göstermektedir. Örneğin, maskeleme seçime bağlı olarak *RST 7.5* ve *RST 5.5* u yetkisiz kılmadan *RST 6.5* u yetkisiz kılabilir.

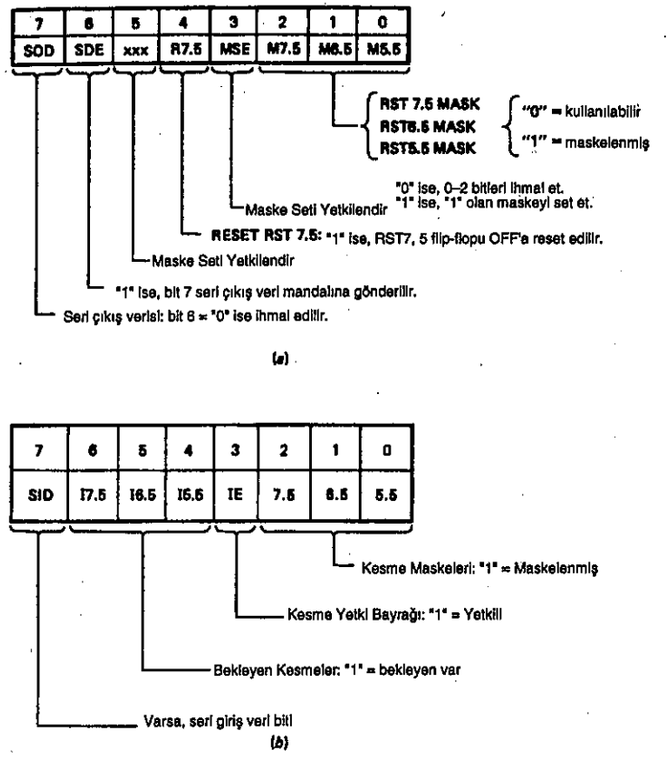
Şekil 9.18a, SIM komutuyla birlikte kullanılan kelime için bit formatını göstermektedir. Eğer bu kelimenin bit 3 ü 1 ise, 0 ile 2 arasındaki bitler, seçime bağlı olarak sırasıyla *RST 5.5, RST 6.5* ve *RST 7.5* girişlerini maskeleyecek veya maskesini kaldıracaktır. Bir bitteki bir 0, O kesmenin maskesini kaldırır.

Eğer bit 3 bir 0 ise, 0 ile 2 arasındaki bitler ihmal edilir. Örneğin *RST 7.5* ve *RST 5.5*'un maskesini kaldırmak fakat *RST 6.5* u maskeli bırakmak için bit3=1, bit 2 = 0, bit 1 = 1 ve bit 0 = 0 olmalıdır. SIM kelimesinin üst 4 biti şimdilik 0 olabilir. Böylece SIM kelimesi 0AH olur. Kesmelerin maskelerini kaldırmak için, önce bir MVI A, 0AH komutuyla bu kelime akümülatöre yüklenir. Sonra SIM komutu yürütülür. Bundan sonra, daha önce yapılmamışsa, girişleri yetkilendirmek üzere EI komutu yürütülebilir.



Şekil 9.17 8085A kesme yetkilendirme ve maske kaldırma.

Eğer akümülatördeki *SIM* kelimesinin bit 6 sı bir 1 ise, bu takdirde bit 7 de bulunan veri, SIM komutu yürütüldüğünde, 8085A nın *SOD (seri veri çıkışı)* hattına gönderilir. Bu özellik, 8085A'nın, bir UART donanımı yerine bir program yordamı kullanarak, bir teletayp veya yazıcıya kolayca seri veri göndermesine imkan verir.



Şekil 9.18 8085A SIM ve RIM komutlarının bit formatı. (a) SIM. (b) RIM.

Şekil 9.18b, 8085A *oku-kes-maskele (read-interrupt-mask: RIM)* komutunun kelime formatını göstermektedir. RIM komutu yürütüldükten sonra, bu kelime akümülatördedir. 0,1 ve 2 bitleri, *RST 5.5, RST 6.5* ve *RST 7.5*'un halen maskeli mi maskesiz mi olduğunu belirtir. Bit 3, 8085A kesmelerinin yetkili mi yetkisiz mi olduğunu söyler. Bu bitin 1 olması kesmelerin yetkili olduğu anlamına gelir. 4,5 ve 6 bitleri *RST 5.5, RST 6.5* ve *RST 7.5*'dan bir kesme isteği olup olmadığını belirtir. Bu, kritik bir süreç sırasında programın kesmeleri yetkisiz kılmasına, fakat belli bir kesme isteği olup olmadığını anlamak için bir

RIM komutuyla 4,5 ve 6 bitlerini hâlâ kontrol edebilmesine imkan verir. Bir RIM komutundan sonra, akümülatörün bit 7'si, 8085A SID (seri veri girişi) bacağında bulunan veriyi içerir. Bu özelik, 8085A'nın, bir UART donanımı yerine bir program yordamı kullanarak bir teletayp veya terminalden seri veri okumasına imkan tanır.

8085A GEÇİŞ DURUM DİYAGRAMI

Şekil 9.19, 8085A geçiş durum diyagramını göstermektedir. Bu, bir komut çevrimi sırasında, 8085A'nın ne zaman bir durma durumuna gireceğini, bir bekleme durumu ekleyeceğini, bir HOLD girişine cevap vereceğini veya bir kesme girişine cevap vereceğini göstermenin kısa bir yoludur.

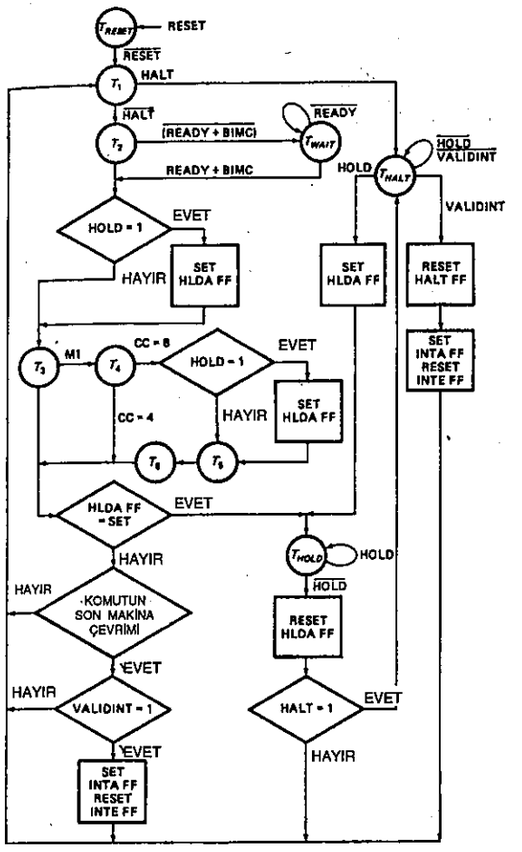
Eğer $\overline{\text{RESET IN}}$ girişi aktifse, 8085A, adres yolu kayan durumda olmak üzere, resette kalır. Eğer $\overline{\text{RESET IN}}$ aktif değilse veya önceki makina çevrimi bitmişse, CPU yeni bir makina çevriminin *T1*'ine girer. Eğer bir önceki komut HLT komutu idiyse, CPU doğrudan durma durumuna gider. Durma durumundan çıkmanın üç yolu bir resetle (gösterilmemiştir), geçerli bir kesmeyle ve bir HOLD isteği iledir. Tutmada çıkışın sadece geçici olduğuna dikkat edin. HOLD aktif olmaktan çıkar çıkmaz, CPU durma durumuna geri döner.

Eğer bir durma durumuna girilmemiş ise, bu takdirde CPU, makina çevriminin *T2*'sine devam eder. Burada, CPU, READY girişini kontrol eder. Eğer READY hattı aktif değilse *(L)*, READY yükselinceye kadar CPU bekleme durumları eklemeye devam eder.

HOLD girişi çeşitli noktalarda kontrol edilir. Eğer üzerinde bir tutma isteği varsa, tutma alındı flip-flopu set edilir. Ancak, CPU, makina çevriminin taşıt kullanımının sonuna kadar tutma durumuna girmeyecektir.

8085A nın, komut çevriminin sonuna kadar geçerli bir kesme isteğinin var olup olmadığını kontrol etmediğine dikkat edin. Bu, bir sonraki komutun adresinin yığına itilebilmesi için gereklidir.

Özetlersek: Bir durma durumuna *T1* sırasında girilmektedir, bir bekleme durumuna *T2* den sonra girilmektedir, bir tutma durumuna makina çevrimi tamamlandıktan sonra girilmektedir ve bir kesmeye, komut çevrimi tamamlandıktan sonra cevap verilmektedir.



NOT: SEMBOL TANIMLARI

- T_X = CPU durumu T_X. Bütün CPU durum geçişleri saatin (CLK) düşen kenarında olur.
- X = Çeşitli alternatif yollardan hangisinin izleneceğini belirleyen bir karar (x)
- X = X hareketini yap.
- → = Olaylar dizisini belirten akış hattı
- X → = Eğer koşul X doğruysa dizisini belirten akış hattı

CC = O andaki makina çevriminde bulunan saat çevrimi sayısı.

BIMC = "BUS IDLE MACHINE CYCLE" = Sistem taşıtını kullanmayan makina çevrimi

VALIDINT = "VALID INTERRUPT" = Geçerli Kesme = Yetkilendirilmiş ve maskesi kalkmış beklemekte olan bir kenar (maskeleme sadece RST 5.5, 6.5 ve 7.5 girişleriyle ilgilidir).

HLDA FF = Dahili tutma alındı flip-flopu. 8085A sistem yollarının, HLDA FF SET olduktan bir saat çevrimi sonra üç-durumlu, yapıldıklarına dikkat edin.

Şekil 9.19 8085A geçiş durum diyagramı.

BİR 8080A VEYA 8085A MİKROBİLGİSAYAR SİSTEMİNİN ARIZASINI BULMA

Bölüm 13, mikrobilgisayar tabanlı cihazların geliştirilmesi ve arızalarının giderilmesi için kullanılan teknikleri ve cihazları anlatmaktadır. Burada, size, daha önce çalışmış olan arızalı bir 8080A veya 8085A mikrobilgisayar sisteminin arızasını bulmak için Bölüm 8 deki programlama bilgisini ve bu bölümdeki donanım bilgisini nasıl kullanacağınızı göstereceğiz.

Bir mikrobilgisayar veya mikrobilgisayar tabanlı bir cihazın arızasını ararken, bir sıralı yaklaşım genelde rasgele araştırmak, dürtmek ve ümit etmekten daha etkilidir. Büyük bir arıza için de olsa, ufak bir hatalı çalışma için de olsa, en iyisi aşağıda anlatılan arıza bulma yordamını başından sonuna kadar uygulamaktır. Alıştırma olması için, bu yordamı kullandığınız mikrobilgisayarda deneyin.

İLK KONTROLLAR VE SİNYAL YOKLAMASI

1. Belirtiyi tanımlayın. Bulduğunuz veya müşterinin anlattığı belirtileri listeleyin. Belirtileri size bir başkası anlatıyorsa, problemin sadece bir işletmen hatası olmadığından emin olmak üzere bu belirtileri kendiniz kontrol edin veya anlatan şahsa gösterin.
2. Aşağıda önerilenler de dahil olmak üzere, gözle ve dokunarak mikrobilgisayarınızın dikkatli bir muayenesini yapın.
   a. Aşırı sıcaklık belirtilerini kontrol edin.
   b. Bütün entegrelerin yerlerine sıkıca oturduklarını ve entegrelerin bacaklarından herhangi birisinin eğilmemiş olduğunu kontrol edin. Eğilmiş bir bacak bazan bir süre temas sağlar, fakat ısınma, soğuma ve titreşme sonradan temas sağlamayabilir. Ayrıca, ucuz entegre devre soketleri zamanla oksitlenebilir ve temas yapmayabilir.
   c. Baskılı devre kartı kenar konnektörlerinin temiz ve yerlerine tam oturmuş olduğunu kontrol edin. Kenar konnektörlerinin parmakları üzerinde bulunabilecek bir ince tabaka kir veya yağ temiz bir silgi ile temizlenebilir.
   d. Yassı-kablo konnektörlerini kontrol edin. Eğer bunlar çok fazla oynatılmışlarsa, hala güvenilir temas sağlamıyor olabilirler. Bir kabloyu kımıldatmak, makinanın çalışmasını değiştiriyorsa, kabloyu değiştirmeyi deneyin.

3. Güç kaynaklarını kontrol edin. Güç kaynakları problemlerin sık karşılaşılan nedenleridir. Alçak gerilim veya aşırı dalgacıklar, bir entegrenin bozuk olduğunu düşünmenize sebeb olabilir.
   a. İşletme kılavuzundan mikrobilgisayarınızın güç kaynağı özelliklerini belirleyin. Sonra güç kaynağı gerilimlerini ölçün ve belirtilen değerler arasında olduklarını kontrol edin.
   b. Güç kaynaklarının üzerinde aşırı gürültü olmadığından emin olmak için güç kaynaklarını ayrıca bir osiloskopla kontrol edin.
4. Saat sinyallerini kontrol edin. Bir osiloskop kullanarak, saat sinyallerinin var olduğunu ve doğru olduğunu kontrol edin.
5. CPU giriş kontrol sinyallerini kontrol edin.
   a. Şimdiye kadar olan herşey doğru gözüküyorsa fakat birimin hala problemleri varsa, RESET, READY, HOLD, INT, $\overline{\text{NMI}}$, v.b. CPU giriş kontrol sinyallerinin normal çalışma için doğru seviyede olduklarını kontrol edin. (Doğru seviyeler için CPU veri kitabına veya kılavuzuna başvurun.) Sık karşılaşılan bir problem, başka bir donanım probleminden dolayı işlemcinin bekleme, tutma, reset veya kesme durumunda takılıp kalmasıdır.
   b. Eğer bu sinyallerden birisi hatalı seviyede ise, şemadan o girişe ne bağlandığını bulun ve problemi izleyin.
6. CPU çıkış sinyallerini kontrol edin.
   a. Giriş sinyalleri doğru ise, SYCN, ALE, DBIN, $\overline{\text{STATUS STROBE}}$, $\overline{\text{MEMR}}$, $\overline{\text{RD}}$, v.b. gibi aktif CPU veya CPU-grubu çıkış sinyallerinin varlığını kontrol edin. Bu sinyallerin yokluğu büyük bir ihtimalle bozuk bir CPU veya CPU-grubu entegreyi gösterir. Olmayan sinyalin kaynağı olan entegreyi değiştirmeyi deneyin.
   b. Ayrıca, alt adres hatlarında ve veri hatlarında darbelerin varlığını kontrol edin. Normal osiloskoplarda bu darbeler rasgele gözükecektir, ancak bu en azından CPU 'nun adresleri gönderdiğinin ve veri taşıtının aktif olduğunun hızlı bir kontrolu imkanını vermektedir.
   c. Ayrıca, bellek kod çözücülerin, yonga-seçme ($\overline{\text{CS}}$) sinyallerini üretip üretmediğini belirlemek için bir osiloskop kullanın. Eğer bir kod çözücü $\overline{\text{CS}}$ sinyallerini üretmiyorsa, kod çözücü arızalı olabilir veya yetkilendirilmemektedir.

KOMUT YÜRÜTÜLMESİ SIRASINDA CPU NUN ÇALIŞMASININ ADIM ADIM KONTROLU

Eğer CPU veya CPU-grubu entegrelerden biri bozuk değilse, bu takdirde doğru komutların bellekten doğru sırada CPU ya gelip gelmediğini bulmalısınız.

1. Bir mantık analizörü, ayrıntılı CPU çalışmasını kontrol için iyi bir araçtır.

   a. Sinyal analizörünün sekiz sinyal girişini veri yoluna ve kalan sinyal girişlerini adres yoluna bağlayın.

   b. Analizöre, bir 8080A için $\overline{MEMR}$ veya bir 8085A için $\overline{RD}$ sinyali saat olarak bağlanabilir.

   c. Analizör, bellekten okunacak ilk komut kelimesi ve bu kelimenin adresiyle tetiklenecek şekilde ayarlanmalıdır.

   d. Analizörün izi, tetikleme kelimesinden sonra ilk 256 kadar adres ve veri kelimesini göstermelidir.

   e. Okunan kelimeler yanlışsa, problemin bir çok mümkün kaynağı vardır.
   (1) Değişmesi gereken fakat değişmeyen bir adres veya veri hattı V_{cc} veya toprağa kısa devre olabilir.
   (2) Bir adres veya veri yolu tampon entegresi bozuk olabilir veya yetkilendirilmemektedir.
   (3) İki adres veya veri hattı birbirine kısa devre olmuş olabilir. Mantık analizörü izinde bu deseni arayın.

2. Tek-adımlama, ayrıntılı CPU çalışmasını kontrol etmek için diğer bir yoldur. Eğer bir mantık analizörünüz yoksa, 8080A, 8085A ve Z80 işlemcileri için, aynı bilgileri görmek üzere bir donanımla gerçekleştirilen tek- adımlama yöntemi kullanılabilir.

   a. Açıklama. Önceki kısımdan, 8224 ün READY girişine veya 8085A'nın READY girişine bir alçak uygulamanın, 8080A veya 8085A yı *T2* durumunun sonunda bir bekleme durumuna girmesine sebeb olduğunu hatırlayın. READY alçak tutulduğu sürece, işlemci bekleme durumları eklemeye devam eder. Bekleme durumu sırasında, işlemde olan makina çevrimi için olan adres, veri ve kontrol sinyalleri yollar üzerinde tutulur. İşlemci bekleme durumunda gerekli olduğu sürece tutulabilir. Bütün yol hatlarındaki seviyelerin doğruluğunu kontrol etmek için bir osiloskop hatta bir mantık probu kullanılabilir.

Sinyaller bir makina çevrimi için kontrol edildikten sonra, READY hattı, 8080A veya 8085A'nın sadece o makina çevrimini bitirmesine ve bir sonraki makina çevrimine başlamasına yetecek kadar bir süre yüksek yapılır. READY tekrar alçak yapıldığından, işlemci, bu makina çevriminin *T2*'sinin sonunda başka bir bekleme durumuna girer. Bu makina çevrimi için de taşıt sinyalleri tekrar kontrol edilebilir. Taşıt sinyalleri doğruysa, READY hattına uygulanan kısa bir darbe, işlemciyi bir sonraki makina çevriminin *T2*'sine götürür.

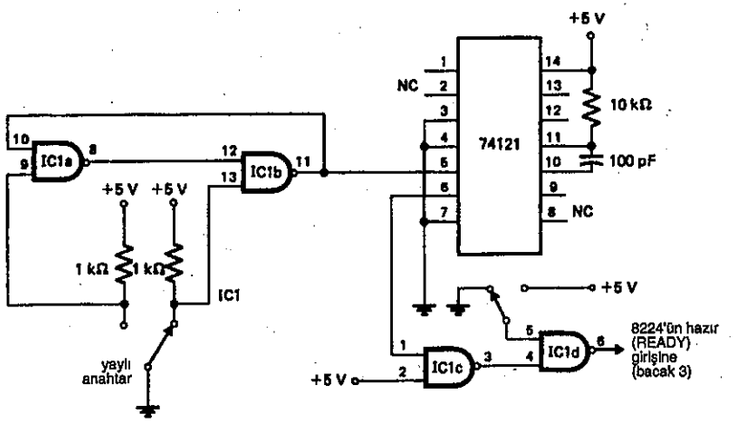
Bu teknik, bir programın çalışmasını her makina çevriminde kontrol etmek için, tek-adımlama ile programı çalıştırmak üzere bekleme durumlarını kullanır. Bu teknik, genellikle, zamanlama problemlerini açığa çıkarmaz. Ancak, bir programın, niçin belli bir noktada durduğunu veya IOR sinyali gönderilmediği için bir portu okumadığını belirlemek için bu, mükemmel bir yoldur. Örneğin, 8080A OUT komutunun üç makina çevrimi için yollarda beklediğiniz sinyalleri görmek üzere Şekil 9.2'ye bakın.

b. Donanımın kuruluşu. Şekil 9.20, Şekil 9.3'teki gibi bir 8080A sisteminiveya Şekil 9.9'daki gibi bir 8085A sistemini tek-adımlatmada kullanılabilecek basit bir devreyi göstermektedir. İki 7400 VE DEĞİL kapısı, yaylı basma düğmesi anahtarın ON durumunda titreşimi gidermektedir. Anahtardan gelen bir sinyalle tetiklendiğinde, 74121 tekkararlı multivibratör 500 ns lik bir darbe üretmektedir.

(1) Bir 8080A için, 74121'in çıkış darbesi, 8080A yı bir makina çevrimindeki bekleme durumundan bir sonrakindeki bekleme durumuna tekadımlatmak için 8224'ün READY girişine uygulanabilir.

(2) Bir 8085A sistemi için, 74121'in tek-adımlatma çıkışı, doğrudan READY girişine bağlanır. Kalan iki VE DEĞİL kapısı, tek-adım darbeleri için anahtar olarak kullanılır. Eğer anahtarın kontrol girişi alçak ise, READY yüksek olacaktır ve işlemci normal çalışacaktır.

c. Tek-adımlatma yaklaşımının önemli bir noktası, sinyalleri doğru akım gibi izlemenize müsade etmesidir. Sürücü transistörlü LED'ler adres ve veri hatlarına bağlanabilir. Böylece, programda makina çevriminden makina çevrimine adım adım ilerlerken yollardaki seviyeleri kolayca görebilirsiniz.



Şekil 9.20 Bir 8080A veya 8085A sistemini, bekleme durumları ekleyerek tek-adımlatmak için devre.

ADRES VE VERİ YOL TAMPONLARINI TEST ETME

Adres yolu ve veri yolu tamponları kullanan mikrobilgisayar sistemlerinde, bu tamponlar sık karşılaşılan arıza nedenidir. Bir tamponla ilgili problemi bulmada çok kullanılan iki yöntem aşağıda verilmiştir.

1. Bir mantık analizörü veya donanım türü tek-adımlama yöntemi ile bir osiloskop veya bir voltmetre kullanarak, her tamponun girişlerinin ve çıkışlarının uygun seviyede olup olmadıklarını kontrol edebilirsiniz.

   a. Eğer bir sinyal tamponun giriş bacağına geliyorsa fakat çıkış bacağında görülmüyorsa:
   (1) Önce tamponun yetkilendirildiğini kontrol edin. Eğer yetkilendirilmiyorsa, yetki ile ilgili problemi bulun.
   (2) Tampon yetkilendiriliyorsa, bu takdirde arızalı olabilir. Değiştirmeyi deneyin ve çıkışı tekrar kontrol edin.
   (3) Şüpheli tamponun çıkış seviyesi hala doğru değilse, büyük bir ihtimalle o yol üstündeki başka bir aygıtın dahili olarak kısa devre olmuş giriş veya çıkışı vardır. Kısa devre olan aygıtı bulmak için, arızalı hatta bağlı olan girişleri veya çıkışları bulunan aygıtları bulmak üzere devre şemasını kullanın. Hat doğru mantık seviyeye ulaşıncaya kadar bu entegreleri birer birer çıkarın veya bacaklarını kaldırın. Bu durumda, en son kaldırılan entegre arızalıdır ve değiştirilmelidir.

   (4) Maalesef, bir yol hattında birden fazla giriş veya çıkışın kısa devre olması pek nadir değildir. Bu nedenle, arızalı entegreyi değiştirdikten sonra, hiçbirinin arızalı olmadığından emin olmak için diğer entegreleri teker teker yerlerine takın.

   b. Eğer bir tamponun giriş sinyali doğru değilse:
   (1) Tamponu çıkarın. Eğer, şimdi sinyal giriş hattı doğru değere sahipse, büyük bir ihtimalle tampon girişi kısa devredir ve tampon değiştirilmelidir.
   (2) Tampon çıkarıldıktan sonra, giriş sinyal hattı hala doğru değilse, büyük bir ihtimalle o hat üzerindeki başka bir aygıt kısa devredir. Bu hat üzerindeki diğer aygıtları (veya bacakları), sinyal seviyesi doğru oluncaya kadar, yukarıda (3) te anlatıldığı gibi, birer birer çıkarın. Arızalı entegreyi değiştirin. Ayrıca, yukarıda (4)'te anlatılana benzer bir yordam kullanarak diğer entegreleri yerine takın ve doğru çalıştıklarını kontrol edin.
   (3) Hattaki bütün aygıtları çıkarmak problemi düzeltmezse, sinyal kaynağı arızalı olabilir.

   c. Birkaç ucuz tampon veya kod çözücü entegreyi değiştirmenin maliyetinin, arızalı birini bulmak üzere lehimleri sökmek ve bacakları kaldırmak için gerekli işçiliğin maliyetinden çok daha az olabileceğini unutmayın.

2. Eğer Hewlett-Packard 546A gibi bir darbe probunuz veya Hewlett-Packard 547A (Şekil 1.5 e bakın) gibi bir akım izleme probunuz varsa, kısa devre olmuş bir entegre genellikle çok çabuk bulunabilir. 546A, kısa devre olan hatta akım darbesi göndermek için kullanılır. Daha sonra akım probu, Şekil 1.5 de gösterildiği gibi, kısa devre olan hat boyunca hareket ettirilir. Prob, kısa devre girişli entegreyi geçince, prob ucundaki ışık söner.

RAM'I TEST ETME

Eğer mikrobilgisayar, ROM'dan komutları ve verileri doğru olarak okuyorsa, fakat hâlâ bütün işlevlerini doğru olarak yapmıyorsa, test etmek için kalan kısımlar RAM ve I/O portlarıdır. RAM ı test etmek için, mikrobilgisayarınızın yeteneklerine göre aşağıdaki iki basamaktan birini kullanın.

1. Monitör programını kullanmak. Eğer sistemin bir monitör programı varsa, bu takdirde bu program, RAM a yazmak ve yazmanın başarılı olduğunu görmek için geri okumakta kullanılabilir. Bir RAM bloğunu test ederken, önce her adrese 1 ler (FF) yazıp okuyan, sonra 0 lar yazıp okuyan

ve son olarak da alterne eden 1'ler ve 0'lar (55 veya AA) yazıp okuyan bir yordam kullanın.

2. Bir test PROM u kullanmak. Eğer sistemin monitör programı yoksa, bu takdirde sistem ROM'u, yukarıdaki madde 1 de anlatılana benzer bir yordam içeren ve sistemdeki bütün RAM ları test edip arıza varsa bildiren bir test PROM'uyla değiştirilebilir.

I/O PORTLARINI TEST ETME

Eğer RAM doğru çalışıyorsa, bu takdirde I/O portlarını çalıştırmak ve bunlardaki bir problemi yakalamak için program yordamları kullanılabilir.

1. Örneğin, bir giriş portunu kontrol etmek için, o portu ilklendirmek ve o porttan tekrar tekrar okumak için bir yordam yazılır.
2. Portun adreslendiğini (yetkilendirildiğini) ve aygıt yetkilendirildiğinde port aygıtının girişlerindeki verinin, sistem veri yoluna ulaştığını kontrol etmek için bir standart osiloskop kullanılabilir.

ZILOG Z80

Intel tasarımcıları, *8080A*'nın gelişmişi olarak *8085A* üzerinde çalışmaktayken, Intel de *8080A* üzerinde çalışmış olan tasarımcıların bazıları *Zilog*, Inc., e geçtiler ve *Z80*' i geliştirdiler. *Z80/Z80A*, *8080A* nın büyütülmüşü olarak kabul edilir, çünkü *Z80/Z80A*, benzer bir mimari yapı kullanmaktadır ve *8080A* için hazırlanan makina kodu programlar bir *Z80*'de çalışmaktadır. Z80, Radio-Shack TRS-80 ev bilgisayarlarında kullanılan *CPU*'dur.

Z80/Z80A BLOK DİYAGRAMI, KAYDEDİCİLERİ VE BACAK TANIMLARI

Zilog Z80/Z80A'nın tam veri sayfaları ve ürün özellikleri Ek C'de bulunmaktadır. Z80 in aşağıdaki anlatımında bunlar kullanılmaktadır. Bu arada, Z80A nın Z80 ile aynı olduğunu, sadece *Z80A*'nın 4 MHz e kadar bir giriş saatiyle, Z80 in ise sadece 2,5 MHz e kadar bir saatle çalıştığını belirtelim.

Önce, 1. veri sayfasında, Z80'in sadece tek bir 5-V'luk kaynak ve bir tek fazlı TTL seviyesi saat gerektirdiğine dikkat edin. CPU blok diyagramı, 8080A daki fonksiyonların aynısını göstermektedir. Z80A'nın bir 8-bitlik veri yolu ve bir 16-bitlik adres yolu vardır. Kaydediciler yönünden Z80 çok daha zengindir. 8080A nın *A*, *PSW*, *B*, *C*, *D*, *E*, *H* ve *L* kaydedicileri aynen bulunmaktadır. 16-bitlik program sayıcı ve yığın göstericiye ek olarak, Z80A' nın *IX* ve

IY olmak üzere iki 16-bitlik indis kaydedicisi vardır. Ayrıca, bir 8-bitlik kesme vektör kaydedicisi ve bir 8-bitlik bellek tazeleme kaydedicisi de bulunmaktadır.

2. veri sayfası, Z80'in bacak tanımlarını göstermektedir. Bir 16-bitlik üç-durumlu adres yolu ve bir 8-bitlik üç-durumlu veri yolu, 8080A'da olduğu gibi, gösterilmiştir. Kontrol yolu sinyallerinin dahili olarak kodları çözülmektedir, bu nedenle 8228 gibi bir harici aygıta gerek yoktur. $\overline{M1}$, $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$ ve $\overline{WR}$ fonksiyonları, tanımlarından oldukça açıktır, fakat $\overline{RFSH}$ bir miktar açıklama gerektirebilir. Dinamik MOS bellekler, statik MOS belleklerden daha ucuz olduğu için, tasarımcılar Z80'de dinamik bellekleri tazelemek için devre koymuşlardır. Bu, harici tazeleme kontrolör gereksinimini ortadan kaldırmaktadır. Dinamik RAM'ların, her satırını yaklaşık 2 ms'de bir adresleyerek, tazelendiğini Bölüm 5'ten hatırlayın. Getirilen bir komutun kodu çözülürken, tazeleme kaydedicisinin içeriği adres yoluna gönderilmektedir. Bir $\overline{RFSH}$ ve bir $\overline{MREQ}$ sinyali, bu adresin alt 7 bitini sistemdeki bütün dinamik belleklere göndermek için kullanılır. Bu, adreslenen satırları tazeler. Bundan sonra, bir sonraki çevrimde, bir sonraki satırı tazelemek için kaydedicinin içeriği bir artırılır. Bütün bunlar kullanıcı için saydamdır. Diğer bir deyişle, tazeleme, işlemci, adres ve veri yollarını başka amaçlarla kullanmadığı sırada gerçekleştirilmektedir.

$\overline{HALT}$, CPU nun durma durumunda olduğunu belirtir. $\overline{WAIT}$ girişi 8080A'daki READY girişi ile aynı işlevi görür. $\overline{WAIT}$ alçak olduğu sürece bekleme durumları eklenir. $\overline{INT}$ kesme girişi, 8080A'daki $\overline{INT}$ girişi ile aynı işlevi görür. $\overline{NMI}$ maskelenemeyen kesme girişi, 8085A'nın TRAP girişine benzer işlev görür. Bu giriş maskelenemez ve bu girişe uygulanacak bir alçak, program yürütülmesini 0066H adresine atlatır.

$\overline{BUSRQ}$, 8080A'nın HOLD girişine karşılık gelir. Bu girişe uygulanan bir alçak, o andaki makina çevriminin sonunda adres, veri ve kontrol yollarını kayan hale getirir. Bundan sonra, yollar bir başka işlemci veya DMA kontrolör tarafından kullanılabilir. $\overline{BUSAK}$ *(bus acknowledge = taşıt alındısı)*, yolların kayan durumda olduğunu belirtir.

Z80A KOMUT KÜMESİ

Z80A'nın 158 komutu vardır. 8080A'nın 78 komutu da bunların arasında içerilmiştir, fakat Z80A bunlar için farklı bellenir kullanmaktadır (genellikle bellenirlerin yayın hakları saklıdır). Bu nedenle 8080A komutları hemen tanınamayabilir. Yeni komutların çoğu op kodları için bellekte iki bayt gerektirirler. Örneğin, CPIR komutu, hex olarak iki baytlık EDB1 op koduna sahiptir.

8080A komutlarına ek olarak, Z80A'nın, 8080A'nın bir çok komutuna denk olan tek komutları vardır. Örneğin, bir tek LDIR komutu belleğin bir ye-

rindeki bir veri baytı bloğunu başka bir yerine taşır. Ek C'deki veri sayfası Z80A komut kümesinin tamamını sembolik şekilde göstermektedir. Komutların veri sayfalarında sık kullanılan sembolik şekillerini öğrenmeniz için bu iyi bir fırsattır. Z80A komutlarının bütün ayrıntılarının tam bir açıklaması için Zilog'un Z80 Assembly Language Programming Manual'ına (Z80 Çeviri Dili Programlama Kılavuzuna) bakın. 4. veri sayfasının üst kısmı, kullanılan sembollerin bir anahtarını vermektedir. Bunlardan açıklanması gerekebilecek ikisi "( )" ve "e" dir. Bir kaydedici çifti isminin etrafındaki parantezler, o kaydedici çifti tarafından gösterilen bellek yeri veya I/O portunun içeriğini kastetmektedir. Örneğin, *(HL)*, "*H* ve *L* kaydedicilerindeki 16-bitlik sayının gösterdiği bellek yerinin içeriği" diye okunur. "e" ise program sayıcıya veya indis kaydedicilerinden birine eklenebilen +128 ile -128 arasında bir kaydırmayı temsil eder. "e" 'nin değeri, komutun ikinci veya üçüncü baytı olarak konur. "e", program göreceli atlama komutlarında ve indislenmiş adreslemede kullanılır. Bunlar daha sonra anlatılmaktadır.

4. veri sayfasındaki ilk komut *'LD r,s'* dir. Bu gerçekte birçok MOV tipi komutu temsil eder. Bu komut, bir kaynak kaydedici veya bellek yerinin içeriğini (s), belirtilen kaydediciye (r) yüklemesini söyler. Kaynak; bir başka kaydedici, bir derhal sayı veya HL tarafından gösterilen bir bellek yeri olabilir. Kaynak, ayrıca, 16-bitlik indis kaydedici ile komutun üçüncü baytındaki kaydırmanın toplamının gösterdiği bellek yeri de olabilir. Bu indislenmiş adreslemenin bir örneğidir. Eğer *IX* kaydedici 25AFH içeriyorsa, bu takdirde LD C, *(IX + 18H)* komutu 25AFH ile 18H yi toplar ve bellek yeri 25C7H nin içeriğini C kaydedicisine kopyalar. *IY* kaydedicisi de bu indislenmiş adresleme için kullanılabilir.

Burada bütün Z80A komutlarını anlatması çok fazla yer alacaktır, fakat bu işlemcinin ne kadar sofistike (incelikli) olduğunu göstermek için bir kaç tanesini anlatacağız. Blok taşıma komutu LDIR iyi bir örnektir. Bu komut, bir veri bloğunu belleğin bir yerinden bir başka yerine taşır. Bu komuttan önce kaynağın başlangıç adresi *HL* kaydedicisine, hedefin başlangıç adresi *DE* kaydedicisine, ve taşınacak bayt sayısı *BC* kaydedicisine yüklenir. LDIR komutu yürütülürken, *HL* tarafından gösterilen bellek yerinin içeriği, *DE* tarafından gösterilen bellek yerine taşınır. Bir sonraki kaynak yerini ve hedef yerini göstermek üzere *DE* ve *HL* bir artırılır. Bayt sayıcı 1 azaltılır. Bir LDIR komutu, bütün baytlar taşınıp BC = 0 oluncaya kadar işleme devam eder.

CPIR komutu, akümülatörde içerilen bir bayta eşit bir baytı bulmak üzere bir bellek bloğunda arar. Başlangıç adresi HL ye yüklenir. Aranacak bellek yeri sayısı *BC*'ye yüklenir. CPIR komutu yürütüldüğünde, bir eşleşme bulununcaya veya *BC* = 0 oluncaya kadar aramaya devam eder.

Z80A komut kümesinin önemli bir özelliği, sadece akümülatörün değil, kaydedicilerden herhangi birisinin veya bir bellek yerinin sağa veya sola

döndürülebilmesidir. RLD ve RRD komutları ile nibıllar bile döndürülebilmektedir.

Bit set ve reset komutları, bir kaydedici veya bellek yerinde istenilen bir bitin, diğer bitleri değiştirmeden, set veya reset edilmesine imkân tanır.

Z80A nın IN ve OUT komutlarının 8080A nınkilere göre daha fazla çeşitlemeleri vardır. Bir IN komutu bir baytı akümülatöre getirebildiği gibi, bir başka kaydediciye veya bellek yerine de getirebilir (koyabilir). INIR komutu, bir veri bloğunu HL tarafından gösterilen ardışıl bellek yerlerine koyacaktır.

Atlama komutları da genişletilmiştir. Z80A'da, 8080A'nın bütün bayrakları vardır ve fazladan, parite bayrağı aritmetik işlemlerde taşmayı göstermek için kullanılır. Akümülatördeki sonuç +127'den büyük veya -128'den küçük ise taşma bayrağı set edilir. *'JP nn'* ve *'JP cc, nn'* komutları 8080A'nın bütün koşullu ve koşulsuz atlama komutlarına karşılık gelmektedir. Bunlar 2. ve 3. baytların- da atlama adresini içeren 3-baytlık komutlardır. Çoğu atlamalar bulunulan yere göre ± 128 bayttan daha yakın olduğundan, 2-baytlık bir göreceli atlama bellekte 1 baytlık bir kazanç sağlayacaktır. *'JR e'* komutu, program yürütmesini, program sayıcı adresini "e" kadar kaydırarak, koşulsuz atlatan bir komuttur. Örneğin, JR komutunun 0250H adresinde ve 05H'lık kaydırmanın da 0251H adresinde olduğunu varsayın. CPU, JR komutunu getirir, program sayıcıyı bir artırır ve kaydırmayı getirir. Bundan sonra otomatik olarak program sayıcıyı 0252H'ye artırır ve program 0257H'ye atlar ve buradan devam eder. Bazı çevirici programlar, JR komutunun adresine dayalı bir "e" girmenize imkan vermektedirler.

Zamanlama döngülerinde faydalı diğer bir enteresan komut 'DJNZ e' dir. *B* kaydedicisi bir azaltılır; eğer 0 değilse, program yürütme e kaydırması kadar atlatılır. Bu, Z80A komut kümesinin tanıtımını tamamlamaktadır. Şimdi Z80A'nın zamanlaması üzerinde kısaca duracağız.

Z80A ZAMANLAMASI VE SİSTEM BAĞLANTILARI

8080A ve 8085A'da olduğu gibi, her Z80 komut çevrimi birkaç makina çevriminden oluşur. Her makina çevrimi üç ile altı durumdan, yani saat çevriminden oluşur. 4 MHz saat girişli bir Z80A için, her durumun süresi 0,25µs dir.

Ek C'deki veri sayfaları, Z80 ve Z80A için ayrıntılı a.a. karakteristiklerini ve zamanlama diyagramlarını göstermektedir.

Bir Z80A, standart RAM ve ROM parçalarını kullanarak kolayca bir mikroişlemci sistemi haline getirilebilir. İki 8-bitlik programlanabilir port içeren Z80-PIO ile paralel I/O portları eklenebilir. Z80A-SIO programlanabilir USART gibi aygıtlarla seri I/O portları eklenebilir.

TEK-YONGA MİKROİŞLEMCİLER

Mikroişlemcilerin gelişmesinde, CPU ve yeterli RAM, ROM ve I/O portunu, küçük bir sistem oluşturmak üzere, bir tek entegre paketine koymak şeklinde güçlü bir eğilim görülmüştür. Bunun iyi bir örneği Intel 8048 ailesidir.

INTEL 8048 AİLESİ

MİMARİ YAPI Şekil 9.21, 8048/8049'un iç yapısının bir blok diyagramını göstermektedir. 8048 ile 8049 birbirlerinin aynıdır, sadece 8048 de 1K lık ROM ve 64 baytlık RAM vardır, 8049'da ise 2K lık ROM ve 128 baytlık RAM vardır. ROM program belleği, bir resetin CPU'yu ROM'daki programın başlangıcına gönderebilmesi için, 0000H ile 3FFFH arasına haritalanmıştır.

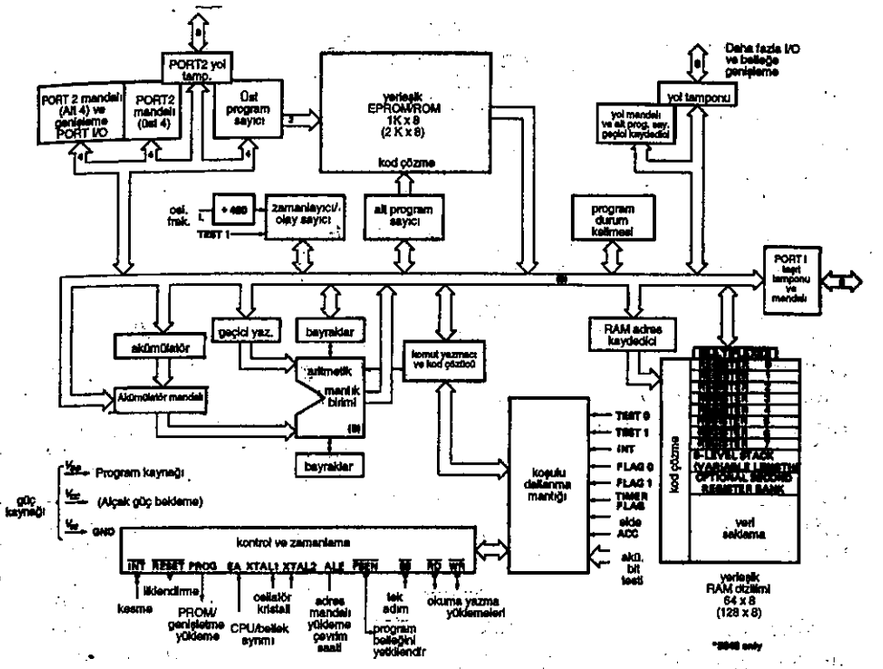
Veri belleği RAM'ın bazı kısımları yaz-boz kaydedicileri olarak kullanılmaktadır. İki tane 8-baytlık kaydedici bankası vardır; kaydedici bankası 0 (RB0) ve seçimlik ikinci kaydedici bankası RB1. Ayrıca veri RAM ında 8-baytlık bir yığın alanı da vardır.

8048'in giriş veya çıkış olarak kullanılabilen 27 hattı vardır. 8-bitlik üç tane portu ve üç test girişi bulunmaktadır. Port 1 ve 2'nin her bacağı giriş veya çıkışa programlanabilir. BUS, dahili yolun bir uzantısı olan çift-yönlü bir porttur. Adresleri, verileri ve komutları aktaran tek bir dahili yol olduğuna dikkat edin. TEST0 TEST 1 ve $\overline{\text{INT}}$ girişleri, koşullu atlama komutlarıyla test edilebilir.

Dahili saat mantığı, 8048'in saatini üretmek için sadece bir harici kristal gerektirir. Yonga üzerinde bir 8-bitlik programlanabilir zamanlayıcı da bulunmaktadır.

KOMUT KÜMESİ 8048 ailesinin komut kümesi 8080A'nınki ile hiçbir şekilde uyuşmamaktadır. Farklı bellenirler ve op kodlar kullanılmıştır. 8048'in 90 komutu Tablo 9.1 de gösterilmiştir. Bu komutları baştan sona okuyup 8080A ve Z80A komutlarıyla karşılaştırın.

SWAP A komutu akümülatördeki nibılların yerlerini değiştirir. Çok fazla atlama ve dallanma komutları çeşitlemesi vardır. Kaydediciyi bir azalt ve 0 değilse belirtilen adrese atla demek olan *'DJNZ R*, adres' komutuna dikkat edin. Bu, bir komut halinde birleştirilmiş iki komutun bir örneğidir. Diğer bir örnek, 8080A daki POP, PSW ve RET komutlarının görevini gören, geri dön ve *durumu eski haline getir (return and restore status: RETR)* komutudur.



Şekil 9.21 8048 blok diyagramı. (Intel Corporation.)

Ayrıca zamanlayıcı komutlarına dikkat edin. Sayıcı olarak kullanıldığında, bir MOV T,A komutu sayıcıyı 0 a set edebilir. STRT CNT komutu zamanlayıcı girişini TEST 1 girişine bağlar. Sayıcı, bundan sonra, TEST 1 girişinden giren her darbe için bir artar. Sayım, MOV A, T komutuyla her an okunabilir. STRT T komutu, zamanlayıcıyı, sistem saatinin daha küçük bir değere bölünmüşüne bağlar. 6-MHz lik bir kristal için, zamanlayıcı her 80 µs 'de bir artar.

Tek bir SEL RB0 veya SEL RB1 komutu, çalışan kaydedici bankasını, RB0 veya RB1, değiştirebilir. 8048, 4K' ya kadar program belleği adresleyebilir. Bu, her biri 2K olan iki bankaya bölünmüştür. SEL MB0 ve SEL MB1 komutları istenilen bellek bankasını seçer.

8048 AİLESİ SİSTEMLERİ Çok küçük sistemler için bir 8048 veya 8048 bütün gerekli fonksiyonları içerebilir. 8048'in farklı seçenekleri olan değişik modelleri vardır. Örneğin 8748' in, 8048'deki maske programlanmış ROM yerine, 1K 'lık EPROM' u vardır. 8035 in RAM ı vardır, fakat dahili program belleği yoktur. 8021 ise 8048 komutlarının bir altkümesini gerçekleştiren 28-bacaklı ucuz ve basitleştirilmiş bir cinsidir. 8022 modeli bir 8-bitlik A/D çevirici içerir.

8031/8051/8071, 8048 ailesinin bir geliştirilmiş altailesidir. 8051 in 4-KB lik ROM'u, 128 baytlık RAM ı, dört 8-bitlik portu, iki 16-bitlik zamanlayıcı - olay sayıcısı ve bir tam UARTı vardır. Komut kümesi, çarpma, bölme, çıkarma ve karşılaştırma komutlarını içermek üzere genişletilmiştir.

TABLO 9.1

BELLENİR	TANIM	BAYT	ÇEVRİM
	Akümülatör		
ADD A, R	Kaydediciyi A ya ekle	1	1
ADD A, @ R	Bellekteki veriyi A ya ekle	1	1
ADD A, # veri	Derhal A ya ekle	2	2
ADDC A, R	Elde ile kaydediciyi A ya ekle	1	1
ADDC A, @ R	Elde ile bellekteki veriyi A ya ekle	1	1
ADDC A, # veri	Elde ile veriyi derhal A ya ekle	2	2
ANL A, R	Kaydediciyi A ya VE le	1	1
ANL A, @ R	Bellekteki veriyi A ya VE le	1	1
ANL A, # veri	Derhal A ya VE le	2	2
ORL A , R	Kaydediciyi A ya VEYA la	1	1
ORL A, @ R	Bellekteki veriyi A ya VEYA la	1	1
ORL A, # veri	Derhal A ya VEYA la	2	2
XRL A, R	Kaydediciyi A ya XOR la	1	1
XRL , @ R	Bellekteki veriyi A ya XOR la	1	1
XRL A, # veri	Derhal A ya XOR la	2	2
INC A	A yı bir artır	1	1
DEC A	A yı bir azalt	1	1
CLR A	A yı temizle	1	1
CPL A	A nın tersini al	1	1
DA A	A yı onluğa ayarla	1	1
SWAP A	A nın nibıllarını yer değiştir	1	1
RL A	A yı sóla çevir	1	1
RLC A	A yı elde üzerinden sola çevir	1	1
RR A	A yı sağa çevir	1	1
RRC A	A yı elde üzerinden sağa çevir	1	1

	Giriş / Çıkış		
IN A, P	Port girişini A-ya al	1	2
OUTL P, A	A yı port çıkışına ver	1	2
ANL P, # veri	Derhal porta VE le	2	2
ORL P, # veri	Derhal porta VEYA la	2	2
INS A, BUS	BUS u A ya al	1	2
OUTL BUS, A	A yı BUS a yükle	1	2
ANL BUS, # veri	Derhal BUS a VE le	2	2
ORL BUS, # veri	Derhal BUS a VEYA la	2	2
MOVD A, P	Genişletici portu A ya al	1	2
MOVD P, A	A yı genişletici porta yükle	1	2
ANLD P, A	A yı genişletici porta VE le	1	2
ORLD P, A	A yı genişletici porta VEYA la	1	2
	Kaydediciler		
INC R	Kaydediciyi bir artır	1	1
INC @ R	Bellekteki veriyi bir artır	1	1
DEC R	Kaydediciyi bir azalt	1	1
	Dallanma		
JMP adres	Koşulsuz atla	2	2
JMPP @ A	Dolaylı atla	1	2
DJNZ R, adres	Kaydediciyi bir azalt ve atla	2	2
JC adres	Elde = 1 ise atla	2	2
JNC adres	Elde = 0 ise atla	2	2
JZ adres	A sıfırsa atla	2	2
JNZ adres	A sıfır değilse atla	2	2
JT0 adres	T0 = 1 ise atla	2	2
JNT0 adres	T0 = 0 ise atla	2	2
JT1 adres	T1 = 1 ise atla	2	2
JNT1 adres	T1 = 0 ise atla	2	2

JF0 adres	F0 = 1 ise atla	2	2
JF1 adres	F1 = 1 ise atla	2	2
JTF adres	Zamanlayıcı bayrağı set ise atla	2	2
JN1 adres	INT = 0 ise atla	2	2
JBb adres	Akümülatör bitine göre atla	2	2
	Altyordam		
CALL adres	Altyordama atla	2	2
RETR	Geri dön	1	2
RETR	Geri dön ve durumu eski haline getir	1	2
	Bayraklar		
CLR C	Eldeyi temizle	1	1
CPL C	Eldenin tersini al	1	1
CLR F0	Bayrak 0 ı temizle	1	1
CPL F0	Bayrak 0 ın tersini al	1	1
CLR F1	Bayrak 1 i temizle	1	1
CPL F1	Bayrak 1 in tersini al	1	1
	Veri taşıma		
MOV A, R	Kaydediciyi A ya al	1	1
MOV A, @ R	Bellekteki veriyi A ya al	1	1
MOV A, # veri	Derhal A ya al	2	2
MOV R, A	A yı kaydediciye al	1	1
MOV @ R, A	A yı belleğe al	1	1
MOV R, # veri	Derhal kaydediciye al	2	2
MOV @ R, # veri	Derhal belleğe al	2	2
MOV A, PSW	PSW yu A ya al	1	1
MOV PSW, A	A yı PSW ya al	1	1
XCH A,R	A ile kaydediciyi değiştir	1	1
XCH A, @ R	A ile bellekteki veriyi değiştir	1	1

XCHD A, @ R	A ile kaydedicinin nibılını değiştir	1	
MOVX A, @ R	Harici bellekteki veriyi A ya al	1	2
MOVX @ R, A	A yı harici belleğe al	1	2
MOVP A, @ A	Hazırdaki sayfadan A ya al	1	2
MOVP3 A, @	Sayfa 3 ten A ya al	1	2
	Zamanlayıcı - Sayıcı		
MOV A, T	Zamanlayıcı sayıcıyı oku	1	1
MOV T, A	Zamanlayıcı-sayıcıyı yükle	1	1
STRT T	Zamanlayıcıyı başlat	1	1
STRT CNT	Sayıcıyı başlat	1	1
STOP TCNT	Zamanlayıcı- sayıcıyı durdur	1	1
EN TCNT1	Zamanlayıcı-sayıcı kesmesini yetkilendir	1	1
DIS TCNT1	Zamanlayıcı-sayıcı kesmesini yetkisiz kıl	1	1
	Kontrol		
EN 1	Harici kesmeyi yetkilendir	1	1
DIS 1	Harici kesmeyi yetkisiz kıl	1	1
SEL RB0	Kaydedici bankası 0 ı seç	1	1
SEL RB1	Kaydedici bankası 1 i seç	1	1
SEL MB0	Bellek bankası 0 ı seç	1	1
SEL MB1	Bellek bankası 1 i seç	1	1
ENT 0 CLK	T0 daki saat çıkışını yetkilendir	1	1
NOP	İşlem Yok	1	1

8048, 4 K'lık harici program belleği için genişletilebilir. Bir adresin sekiz alt biti BUS 'a gönderilir ve ALE sinyali ile bir 8212 ye kilitlenir. Dört üst adres biti, port 2 nin alt yarısına gönderilir. Harici program belleğini kullanmak için, işlemcinin $\overline{PSEN}$ çıkışı belleğin $\overline{CS}$ girişine bağlanır. 8048 ailesi aygıtlar, 8155 ve 8755 gibi 8085A çevre birimleri ile kolayca bağdaşırlar.

GÖZDEN GEÇİRME SORULARI VE PROBLEMLER

1. 8080A CPU grubundaki 8224 ve 8228'in işlevlerini anlatın.
2. Niçin bir sistemdeki ana program *ROM*'u 0000H adresinden başlamalıdır?
3. *(a)* Komut çevrimi, *(b)* makina çevrimi ve *(c)* durum'u tanımlayın.
4. 8080A için, bir komut getirme (*M1*) makina çevrimi sırasındaki olaylar dizisini anlatın. Şekil 9.2'yi referans olarak kullanın.
5. Bir mikrobilgisayar sisteminde, adres tamponlarına genellikle niçin gerek duyulur?
6. *8251 A* bir programlanabilir *USART*'tır. Bir mikroişlemci sisteminde bir *USART*'ın işlevi nedir?
7. Bir mikrobilgisayar sisteminde adres çözücünün amacı nedir?
8. Bir *8085A* için bir komut getirme (*M1*) makine çevrimi sırasındaki olaylar dizisini anlatın.
9. a. *8085A ALE* sinyalinin amacı nedir?
   b. Bu sinyalin hangi kenarı aktif kenardır?
10. Bir komut getirme sırasında, *8085A* nın *IO/$\overline{M}$* çıkışı hangi durumdadır?
11. Bir Intel *8755A* ve bir *Intel 8155* te bulunan fonksiyonları kısaca anlatın.
12. 08H taban adresindeki bir *8755* i aşağıdaki şekilde ilklendirecek çeviri dili komutlarını gösterin: *Port A7*'den *A4* 'e çıkış olarak, *port A3*'ten *A0*'a giriş olarak, *port B* çıkış olarak.
13. 40H taban adresindeki bir 8155 i aşağıdaki şekilde ilklendirecek çeviri dili programı yazın: zamanlayıcı, 309A H sayımı ile yüklenecek; zamanlayıcı modu, sürekli kare dalgaya ayarlanacak; *port A çıkışa, port B girişe, port C* çıkışa programlanacak ve zamanlayıcı başlatılacak.
14. Bir 3205 kod çözücünün üç adres girişine, sistem adres taşıtırın *A12, A13* ve *A14* hatları bağlanmıştır. 3205'in *E1* ı *A15*'e, *E2*ı MEMR a ve *E3* ü +5V'a bağlanmıştır. Kod çözücünün çıkışları hangi sekiz ROM adres bloklarını seçecektir? Niçin MEMR, bir ROM kod çözücüde yetkilendirmelerden birisi olarak kullanılmıştır?

15. Problem 14'teki ROM için bir bellek haritası çizin.
16. *8000H* adresinden başlayan sekiz 1 KB lık *RAM* lardan birini seçmek için bir başka 3205'in nasıl bağlanabileceğini gösteren bir devre çiziniz.
17. Bellek-haritalanmış*I/O* ile doğrudan *I/O*'yu tanımlayın.
18. Şekil 9.3'teki sistemde, *8080A*, F4H adresindeki bir *I/O* portunu nasıl seçer?
19. Şekil 9.9'daki *8205*'in, port 08H adreslendiğinde nasıl *A15* entegresi *8755* i seçtiğini açıklayın.
20. Niçin *8080A* ve *8085A*'nın minimum çalışma frekansları 500 KHz dir?
21. Şekil 9.15'teki *8085A* zamanlama diyagramına ve parametrelere bakarak aşağıdakileri belirleyin.

    a. *ALE*'nin düşmesi ile *RD*'ın düşmesi arasındaki zaman
    b. *RD* yükseldikten sonra belleğin verileri kararlı tutması gereken süre
    c. Saatin alçak olması gereken minimum süre
    d. *ALE* alçaldıktan sonra, alt 8 adres bitinin veri yolunda kalması gereken süre.
22. 2732, adres erişim zamanı 450 ns ve yonga-seçme erişim zamanı 450 ns olan standart bir 4K x 8 *EPROM* dur. Maksimum 18 ns T_{PD} si olan bir 8205 kod çözücü kullanan bir 5 MHz'lik *8085A* sisteminde, bu aygıt doğru çalışır mı? Bir 3 MHz'lik *8085A* sisteminde bu aygıt doğru çalışır mı?
23. *a.* Bir *8080A* veya *8085A*, bekleme durumuna nasıl sokulur?
    b. 8080A veya *8085A*, makina çevriminin hangi noktasında bekleme durumuna girer?
    c. Bekleme durumu sırasında yollardan hangi bilgi vardır?
    d. Bir bekleme durumunun uzunluğu nedir?
    e. Kaç tane bekleme durumu konulabilir?
24. Tutma durumu sırasında *8080A* ve *8085A* yollarındaki durumu anlatın. Tutma durumunun bir kullanılışını söyleyin.
25. Durma durumu sırasında *8080A* ve *8085A* hangi işlemleri gerçekleştirir? *8080A* veya *8085A*'nın durma durumundan çıkarılabilmesi için üç yol söyleyin.
26. a. *Kesmeler* ne için kullanılır?
    b. *RST* komutunun, bir kesmeye hizmet vermek için nasıl kullanıldığını anlatın.
    c. Bir kesme altyordamı niçin bütün kaydedicileri *PUSH* ve *POP* etmelidir?
27. Bir *8085A*'nın bir *RST 6.5* kesme sinyali aldıktan sonra gerçekleştireceği başlıca işlemleri anlatın. Kesmelerin yetkili olduğunu ve maskelenmediğini varsayın.

28. Azalan öncelik sırasına göre beş *8085A* kesmesini sıralayın. *TRAP* kesme girişi ne için kullanılabilir?

29. *8085* kesmelerini yetkilendirmek ve sadece *RST 6.5*'un maskesini kaldırmak için gerekli çeviri dili komutları yazın.

30. Bir *8085A* aşağıdakileri bir komut çevriminin hangi anında yapar? (Şekil 9.19'daki geçiş durum diyagramına bakın.)

    a. Durma durumuna girmek
    b. Bekleme durumuna girmek
    c. Tutma durumuna girmek
    d. Geçerli bir kesme isteğine cevap vermek

31. Daha önce çalışmış olan bir mikrobilgisayar sisteminde arızayı bulmak için sırasıyla hangi işlemleri yapacağınızı anlatın.

32. Arıza bulmada bir *8080A* veya *8085A* mikrobilgisayar sistemini tek-adım- latmak için bekleme durumlarının nasıl kullanılacağını açıklayın.

33. *8080A*'ya göre *Z80A* mikroişlemci yapısının ve komut kümesinin nasıl bir gelişme gösterdiğini anlatın.

34. *Z80A*'nın LD HL, 0082H komutu, *8080A* nın hangi komutuna karşılık gelir?

35. Aşağıdaki *Z80A* komutlarının işlevlerini açıklayın:
    JR e; LDDR; LD B, IX +e; EXX.

36. 8048 gibi bir tek-yonga mikrobilgisayar, hangi uygulamalarda daha çok kullanılabilir?

REFERANSLAR / BİBLİYOĞRAFYA

Intel MCS-80/85 *User's Manual,* Intel Corporation, Santa Clara, CA, Ekim 1979.

Intel SDK-85 *System Design Kit User's Manual,* Intel Corporation, Santa Clara, CA, 1978.

Intel *8080A/8085A Assembly Language Programming Manual,* Intel Corporation. Santa Clara, CA, 1977-1978.

Intel *Components Data Catalog,* Intel Corporation, Santa Clara, CA, 1982.

Intel *MCS-48 Family of Single Chip Microcomputers User's Manual,* Intel Corporation. Santa Clara, CA, 1978.

Osborne, Adam: *An Introduction to Microcomputers,* cilt 2: Some Real Products, Adam Osborne and Associates, Inc., Berkeley, CA, 1976.

Z80/Z80A CPU Technical Manual, Zilog Inc., Cupertino, CA, 1977.

BÖLÜM 10

MC 6800 MİKROİŞLEMCİSİ VE MİKROİŞLEMCİ GELİŞİMİ

Başka bir grup işlemcileri anlatmadan önce, birkaç tarihi noktayı belirtelim. Aralık 1971'de Intel, ilk 8-bitlik paralel işlemci 8008'i tanıttı. Bu birinci nesil işlemci PMOS ile yapılmıştı ve çalışan bir sistem oluşturmak için birçok harici parça gerektiriyordu Aralık 1973'te, Intel, *NMOS ile yapılan* ve *CPU* grubu oluşturmak üzere sadece iki ek yonga gerektiren ikinci nesil mikroişlemci 8080'i piyasaya sürdü. Yaklaşık aynı zamanda *Motorola da yine bir ikinci nesil mikroişlemci olan MC6800*'ü tanıttı. MC6800 farklı bir mimari yapıdadır ve farklı bir komut kümesi kullanmaktadır, bu nedenle de 8080 ile hiçbir şekilde uyuşmaz. Yıllarca 8080A ve MC6800, sırasıyla en iyi satan birinci ve ikinci işlemcilerdi. Her biri de bir dizi gelişkin türler doğurmuştur.

Bölüm 9'da *8080A dalındaki gelişmeyi gösterdik.* Bu gelişme, ikinci nesil *8080A ile başlamış, 8085A* ve *Z80A* ile devam etmiş ve *8048* ile bitmiştir.

Bu bölümde, taban olarak MC6800'ü iyice açıklamakta ve sonra 6502'yi anlatmaktayız. MC6801 ile MC6809 da içerilmektedir. 16-bitlik üçüncü nesil işlemcilere de kısaca değinilmektedir.

Bu bölümün sonunda, şunları yapabileceksiniz:

1. *MC6800* mikroişlemcinin mimarisini anlatabileceksiniz.
2. *6800*'ün yedi adresleme modunun çalışmasını listeleyip anlatabileceksiniz.
3. Bir *6800* mikrobilgisayar için basit çeviri dili programlar yazabileceksiniz.
4. *6800* programlarının veya yordamlarının yürütme sürelerini hesaplayabileceksiniz.
5. Zamanlama dalga şekillerinden ve tablolardan, *6800*'ün *saat* ve *kontrol yolu zamanlama parametrelerini* belirleyebileceksiniz.
6. Bir *6800* için *reset* sürecini anlatabileceksiniz.
7. Bir *6800*'ün, *NMI* girişindeki veya *IRQ* girişindeki bir sinyale olan cevabını açıklayabileceksiniz.
8. Bir *6800* tabanlı mikrobilgisayar sisteminde adres çözmeyi belirleyebileceksiniz.
9. Basit bir giriş ve çıkış portu olarak *MC6821 PIA* yı ilklendirebileceksiniz.

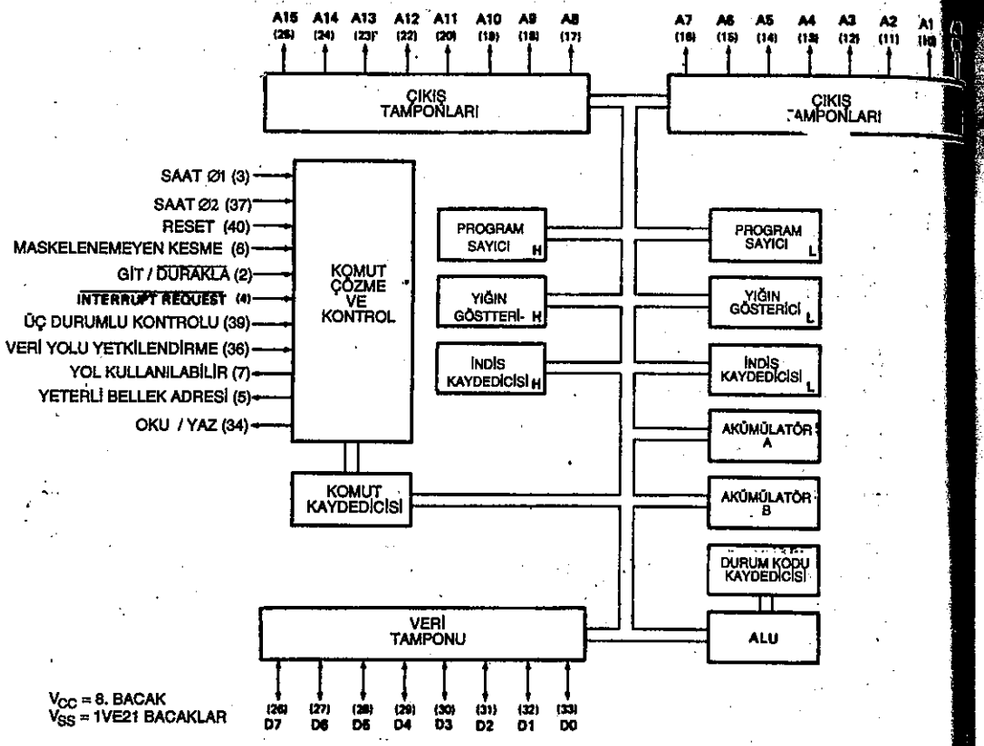
10. *Bozuk bir 6800 mikrobilgisayar sisteminin arızasını* bulmak için mantıksal adımlar dizisini sıralayabileceksiniz.

11. *MCS6502 mikroişlemcisinin mimarisini* ve dolaylı adresleme modundaki çalışmasını anlatabileceksiniz.

MC6800 MİKROİŞLEMCİSİ

MC6800 CPU Grubu

Şekil 10-1, *MC6800 CPU'nun bir blok diyagramıdır. MC6800*'ün bir 8-bitlik üç-durumlu veri yolu ve 16-bitlik üç-durumlu adres yolu vardır. Her hat bir standart TTL girişini sürmek üzere tamponlanmıştır. *MC6800*'ün ayrıca bir 16-bitlik program sayıcı kaydedici, 16-bitlik yığın gösterici kaydedici ve 16-bitlik indis kaydedicisi vardır. İndis kaydedicisi, Z80A'daki *IX* ve *IY* indis kaydedicilerinin kullanıldığı şekilde kullanılır.



Şekil 10.1 Motorola MC6800 mikroişlemcisinin blok diyagramı.

Her ne kadar iki akümülatörü varsa da, MC6800'ün 8080A ve Z80A daki gibi zengin bir yaz-boz kaydedici dizisi yoktur. Bu iki grup arasındaki en önemli farklardan biridir. 8080A grubu işlemciler ise belleğe yöneliktir. Bunlar, RAM yerlerini yaz-boz olarak kullanır ve bütün I/O portları bellek-haritalanmıştır. Göreceğiniz gibi, MC6800 ün, 8080A'nın IN veya OUT doğrudan I/O komutuyla aynı anlamda hiç bir komutu yoktur. ALU 'ya bağlı durum kodu kaydedicisi (condition code register), MC6800 ün bayraklarını içerir. Bunlar 6800 'ün komut kümesi kısmında anlatılmaktadır. Durum kodu kaydedicisi, 8080A'nın bayrak kaydedicisine karşılık gelir.

MC6800, tek bir +5V luk kaynakla çalışır. İki-fazlı bir saat gerektirdiğinden, MC6875 iki-fazlı saat üreteci sürücüsü gibi bir harici saat üreteci genellikle kullanılır. MC6875, iki saati üretmek için Ø1 frekansının 4 katı olan bir harici kristal kullanır. Ayrıca, 6875, sinyallerin 6800 e girebilmesi için açılışta RESET gibi diğer sinyalleri saatlerle senkronize eden devrelere sahiptir.

6800 'ün doğrudan I/O 'su olmadığından, kontrol sinyalleri daha basittir. Bu nedenle, 8080A CPU grubundaki 8228 in bir karşılığı olan bir dengine gerek yoktur.

MC6800 SİNYAL TANIMLARI

Okuma-yazma ($R/\overline{W}$) çıkışındaki bir yüksek, CPU nun bir bellek yerini veya I/O portunu okumakta olduğunu belirtir. Bu çıkıştaki bir alçak bir yazma işlemini belirtir. Geçerli bellek adresi (VMA) çıkışı, adından da anlaşıldığı üzere, adres yolu üzerindeki adresin geçerli olduğunu belirtir. Veri yolu yetkilendirme (DBE) normal olarak yüksek tutulur. DBE alçağa indirildiğinde, veri yolu ve $R/\overline{W}$ çıkışı kayan hale gelir. Üç-durumlu kontrolu (three-state control: TSC) normalde alçak tutulur. TSC yüksek yapıldığında, adres yolu kayan hale gelir. DBE yi alçak TSC'yi yüksek yapmak, DMA işlemi için taşıtları kayan hale getirmenin bir yoludur. DMA işlemleri için kullanılmadığında, DBE, genellikle Ø2 saat darbesine bağlanır. Bu, veri yolunu, sadece bir okuma veya yazma yapılırken yetkilendirilir.

Bir DMA işlemi için yolları kayan hale getirmenin diğer bir yolu da $\overline{HALT}$ girişine bir alçak uygulamaktır. 6800'ün durma durumu, 8080A'nın tutma durumuna denktir. $\overline{HALT}$ girişi alçak yapıldığında, CPU o anda yürütmekte olduğu komutu bitirir ve yollar ile $R/\overline{W}$ çıkışını kayan hale getirir. Yolların kayan halde olduğunu belirtmek için BA bacağından bir yol kullanılabilir (bus available = BA) sinyali gönderilir.

6800'ün dört doğrudan kesme yöntemi vardır (üçü donanımla, biri yazılımla): reset, maskelemeyen kesme, kesme isteği ve yazılımla kesme. $\overline{RESET}$ alçak yapıldığında, CPU otomatik olarak program sayıcıyı FFFE'ye atlatır. CPU, burada saklanmış olan adresin yüksek mertebeli adres baytını okur.

Bundan sonra, program sayıcıyı bir artırarak FFFF 'de saklanan alçak mertebeli adres baytını okur. CPU, bu adresi program sayıcıya yükler ve bir sonraki komutunu bu adresten alıp getirir. Diğer bir deyişle, FFFE ve FFFF adresleri, daima bir resetten sonra istenilen programın başlangıç adresini saklamak için kullanılır. 16-bitlik bir adres saklandığı için, programın başlangıcı belleğin herhangi bir yerinde olabilir. 0000 a reset eden 8080A'nın tersine, 6800, 0000 ve bunu izleyen adreslerin RAM için kullanılmasına imkan verir. Daha sonra açıklayacağımız gibi, bunun bazı programlama avantajları vardır.

Maskelenemeyen kesme girişi, maskelenebilir kesme isteği girişi ve yazılımla kesme komutu da daha sonra anlatılacaktır.

MC6800 KOMUT KÜMESİ

Tablo 10-1, 6800'ün komut kümesinin tamamını göstermektedir. Tablo 10-1 de, tablo da kullanılan sembolleri açıklayan noktalara bakın. OP sembolü, işlem kodu (operation code) demektir ve her komut için hex olarak verilir. "~" sembolü altındaki sütun her komutun saat çevrimi sayısını gösterir. "#" sembolü altındaki sütun her komutun gerektirdiği bayt sayısını gösterir. Tablo 10-1 ayrıca her 6800 komutuyla etkilenen bayrakları, her komut için kullanılabilecek adresleme modlarını ve her komutun işlevinin bir sembolik gösterilişini göstermektedir.

BAYRAKLAR

6800'ün altı durum bayrağı: yarım-elde (half carry: *H*), kesme maskesi (interrupt mask: *I*), negatif (*N*), sıfır (*Z*), taşma (*V*) ve elde (carry: *C*) dir.

Yarım-elde bayrağı H, eğer bit 3 ten bir elde üretilmişse set edilir. Bu bayrak, 8080A nın yardımcı elde bayrağına karşılık gelir ve BCD aritmetik işlemlerinde kullanılır.

I bayrağı, 1 ise kesmelerin yetkisiz kılındığını, 0 ise kesmelerin yetkili olduğunu belirtir.

N bayrağı, işaret biti veya negatif bayrağıdır. Bu bitin 1 olması bir işlemin sonucunun negatif olduğunu belirtir. Bu bayrak, sonucun bit 7 sinin bir kopyasıdır.

Sıfır bayrağı Z, eğer işlem sonucunda bir akümülatördeki veya bellek yerindeki sonuç 0 ise set olur.

Taşma bayrağı V, işaretli bir toplama veya çıkarmanın sonucunun -128 ile +127 arasındaki nizami aralığı aştığını belirtir. Örneğin, eğer bit 7 işaret biti olarak kullanılıyorsa ve 96_{10} yani 01100000_2 sayısı 48_{10} yani 00110000_2 ye eklenirse, sonuç 10010000_2, yani 142_{10}, dir. Bu da izin verilen maksimum 127 den büyüktür. Bit 7 deki 1, hatalı olarak sonucun negatif olduğunu belirtmektedir.

Buna taşma hatası adı verilir ve 6800 böyle bir hata yapıldığını belirtmek üzere V bayrağını set eder.

Elde bayrağı C, bit 7 den bir elde çıkarsa set edilir. Kaydedici veya bellek içerikleri elde bayrağı flip-flopu üzerinden döndürülebilir. Ayrıca, 6800 ün *eldeyi temizlemek için (clear carry: CLC)* ve *eldeyi set etmek için (set carry: SEC)* komutları bulunmaktadır.

TAP komutu, kumanda kodu kaydedicinin tamamını, *A* akümülatöründeki bir baytla yükler ve *TPA*, kumanda kaydedici baytını, *A* akümülatörüne kopyalar.

ADRESLEME MODLARI

Bir adresleme modu, bir komutun üzerinde işlem yapacağı işleneni nasıl veya nereden alacağını gösterme şeklini belirtir. 6800'ün yedi adresleme modu vardır: derhal, doğrudan, indislenmiş, genişletilmiş, anlaşılır, akümülatör ve göreceli. Bu modlar anlatıldıkça, bir çoğu ile 8080A ve Z80A anlatılırken karşılaşmış olduğunuzu göreceksiniz. 8080A ve Z80A'nın komut kümelerinin düzenleniş şekli dolayısıyla, bunları anlatırken adresleme modlarını fazla vurgulamaya gerek olmamıştı. Ancak Tablo 10-1'deki ilk 6800 komutu ADDA ya bakarsanız, bu tek komutun dört adresleme modu olduğunu görürsünüz. Açık olması için, komut kümesine bir bütün olarak yaklaşmadan önce yedi modun tamamını anlatacağız.

Derhal adresleme, işlenenin, komutun ikinci veya ikinci ve üçüncü baytlarında içerildiğini belirtir. Örneğin, ADDA #65, 65_{10} sayısını *A* akümülatörünün içeriği ile topla demektir. # işareti çeviriciye, adresleme modunun derhal olduğunu söyler. Bir MC6800 çevirici için # den sonraki $ işareti sayının hex (onaltılık), bir @ işareti sekizlik, bir % işareti ikilik ve kesme (') işareti bir ASCII karakteri olduğunu belirtir. Örneğin, ADDB #$3A, 3AH sayısını B akümülatörüne ekler. ADDA # komutu, 8080A daki ADI komutuna karşılık gelir.

Doğrudan adresleme, işlenenin 8–bitlik adresinin komutun ikinci baytında içerildiğini belirtir. Bu mod için, işlenenin, 0000 ile 00FF arasındaki en alt 256 bellek yerinde saklanması gerekir. Doğrudan adreslemeye bazen taban sayfa adreslemesi denir, çünkü işlenenin, belleğin 256-baytlık ilk sayfasında olması gerekmektedir. 6800 en alt 256 adresi doğrudan adresleme için boş bırakmak üzere, yeniden başlama ve kesme adreslerini saklamak için FFF8 ile FFFF arasındaki adres yerlerini kullanır. Bu modun bir örneği olarak, ADDA $09 komutu, 09H bellek yerinin içeriğini A akümülatörünün içeriğine ekler. Sonuç A akümülatöründe bırakılır. Bu modun avantajı, komut için sadece 2 bellek baytının gerekmesidir. 8080A daki IN ve OUT komutları esas olarak doğrudan adresleme modunu kullanan yegane 8080A komutlarıdır.

Tablo 10-1

İşlem	Bellenir	Deshol OP	Deshol ~	Deshol #	Direk OP	Direk ~	Direk #	İndisli OP	İndisli ~	İndisli #	Genişletilmiş OP	Genişletilmiş ~	Genişletilmiş #	Göreceli OP	Göreceli ~	Göreceli #	BOOLE/ARİTMETİK İŞLEM (Bütün kaydedici etiketler içeriği belirtir.)	5 H	4 I	3 N	2 Z	1 V	0 C
Topla	ADDA	9B	2	2	9B	3	2	AB	5	2	BB	4	3				A + M → A	↕	·	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	5	2	FB	4	3				B + M → B	↕	·	↕	↕	↕	↕
Akümülatörle topla	ABA													1B	2	1	A + B → A	↕	·	↕	↕	↕	↕
Elde ile topla	ADCA	99	2	2	99	3	2	A9	5	2	B9	4	3				A + M + C → A	↕	·	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	5	2	F9	4	3				B + M + C → B	↕	·	↕	↕	↕	↕
AND	ANDA	94	2	2	94	3	2	A4	5	2	B4	4	3				A · M → A	·	·	↕	↕	R	·
	ANDB	C4	2	2	D4	3	2	E4	5	2	F4	4	3				B · M → B	·	·	↕	↕	R	·
Bit testi	BITA	95	2	2	95	3	2	A5	5	2	B5	4	3				A · M	·	·	↕	↕	R	·
	BITB	C5	2	2	D5	3	2	E5	5	2	F5	4	3				B · M	·	·	↕	↕	R	·
Temizle	CLR							6F	7	2	7F	6	3				00 → M	·	·	R	S	R	R
	CLRA													4F	2	1	00 → A	·	·	R	S	R	R
	CLRB													5F	2	1	00 → B	·	·	R	S	R	R
Karşılaştırma	CMPA	91	2	2	91	3	2	A1	5	2	B1	4	3				A − M	↕	↕	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	5	2	F1	4	3				B − M	↕	↕	↕	↕	↕	↕
Akümülatörleri karşılaştırma	CBA													11	2	1	A − B	↕	↕	↕	↕	↕	↕
1'in tümleyeni	COM							63	7	2	73	6	3				$\overline{M} \rightarrow M$	·	·	↕	↕	R	S
	COMA													43	2	1	$\overline{A} \rightarrow A$	·	·	↕	↕	R	S
	COMB													53	2	1	$\overline{B} \rightarrow B$	·	·	↕	↕	R	S
2'nin tümleyeni	NEG							60	7	2	70	6	3				00 − M → M	·	·	↕	↕	①	②
Negatifini al	NEGA													40	2	1	00 − A → A	·	·	↕	↕	①	②
	NEGB													50	2	1	00 − B → B	·	·	↕	↕	①	②
Ondalığa ayarla	DAA													19	2	1	Converts binary add. of BCD characters to BCD format	·	·	↕	↕	↕	③
Azalt	DEC							6A	7	2	7A	6	3				M − 1 → M	·	·	↕	↕	4	·
	DECA													4A	2	1	A − 1 → A	·	·	↕	↕	4	·
	DECB													5A	2	1	B − 1 → B	·	·	↕	↕	4	·
Özel VEYA	EORA	88	2	2	98	3	2	A8	5	2	B8	4	3				A ⊕ M → A	·	·	↕	↕	R	·
	EORB	C8	2	2	D8	3	2	E8	5	2	F8	4	3				B ⊕ M → B	·	·	↕	↕	R	·
Arttır	INC							6C	7	2	7C	6	3				M + 1 → M	·	·	↕	↕	⑥	·
	INCA													4C	2	1	A + 1 → A	·	·	↕	↕	⑥	·
	INCB													5C	2	1	B + 1 → B	·	·	↕	↕	⑥	·
Akümülatörü yükle	LDAA	96	2	2	96	3	2	A6	5	2	B6	4	3				M → A	·	·	↕	↕	R	·
	LDAB	C6	2	2	D6	3	2	E6	5	2	F6	4	3				M → B	·	·	↕	↕	R	·

Tablo 10-1 (Devam ediyor)

VEYA	ORAA	8A	2	2	9A	3	2	AA	5	2	BA	4	3				A + M → A	·	·	↕	↕	R	·
	ORAB	CA	2	2	DA	3	2	EA	5	2	FA	4	3				B + M → B	·	·	↕	↕	R	·
Veriyi ittir	PSHA													36	4	1	$A \rightarrow M_{SP}$, SP − 1 → SP	·	·	·	·	·	·
	PSHB													37	4	1	$B \rightarrow M_{SP}$, SP − 1 → SP	·	·	·	·	·	·
Veriyi çek	PULA													32	4	1	SP + 1 → SP, $M_{SP} \rightarrow A$	·	·	·	·	·	·
	PULB													33	4	1	SP + 1 → SP, $M_{SP} \rightarrow B$	·	·	·	·	·	·
Sola döndür	ROL							69	7	2	79	6	3				M, A, B C b7 ← b0	·	·	↕	↕	⑥	↕
	ROLA													49	2	1		·	·	↕	↕	⑥	↕
	ROLB													59	2	1		·	·	↕	↕	⑥	↕
Sağa döndür	ROR							66	7	2	76	6	3				M, A, B C b7 → b0	·	·	↕	↕	⑥	↕
	RORA													46	2	1		·	·	↕	↕	⑥	↕
	RORB													56	2	1		·	·	↕	↕	⑥	↕
Sola döndür aritmetik	ASL							68	7	2	78	6	3				M, A, B C b7 b0 ← 0	·	·	↕	↕	⑥	↕
	ASLA													48	2	1		·	·	↕	↕	⑥	↕
	ASLB													58	2	1		·	·	↕	↕	⑥	↕
Sağa döndür aritmetik	ASR							67	7	2	77	6	3				M, A, B b7 b0 C	·	·	↕	↕	⑥	↕
	ASRA													47	2	1		·	·	↕	↕	⑥	↕
	ASRB													57	2	1		·	·	↕	↕	⑥	↕
sağa döndür, mantıksal	LSR							64	7	2	74	6	3				M, A, B 0 → b7 b0 C	·	·	R	↕	⑥	↕
	LSRA													44	2	1		·	·	R	↕	⑥	↕
	LSRB													54	2	1		·	·	R	↕	⑥	↕
Akümülütörü sakla	STAA				97	4	2	A7	6	2	B7	5	3				A → M	·	·	↕	↕	R	·
	STAB				D7	4	2	E7	6	2	F7	5	3				B → M	·	·	↕	↕	R	·
Çıkar	SUBA	90	2	2	90	3	2	A0	5	2	B0	4	3				A − M → A	·	·	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	5	2	F0	4	3				B − M → B	·	·	↕	↕	↕	↕
Akümülatörleri çıkar	SBA													10	2	1	A − B → A	·	·	↕	↕	↕	↕
Elde ile birlikte çıkar	SBCA	82	2	2	92	3	2	A2	5	2	B2	4	3				A − M − C → A	·	·	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	5	2	F2	4	3				B − M − C → B	·	·	↕	↕	↕	↕
Akümülatörleri aktar	TAB													16	2	1	A → B	·	·	↕	↕	R	·
	TBA													17	2	1	B → A	·	·	↕	↕	R	·
Test, sıfır veya eksi	TST							6D	7	2	7D	6	3				M − 00	·	·	↕	↕	R	R
	TSTA													4D	2	1	A − 00	·	·	↕	↕	R	R
	TSTB													5D	2	1	B − 00	·	·	↕	↕	R	R
																		H	I	N	Z	V	C

SEMBOLLERİN AÇIKLAMALARI				DURUM KODU SEMBOLÜ	
OP	İşlem kodu (Hex)+	+	Boole özl VEYA	H	Bit 3'ten yarım elde
~	MPU çevrim sayısı	⊕	Boole dışlayıcı VEYA	I	Kesme maskesi
#	Program bayt sayısı	M̄	M'nin tersini al	N	Negatif (İşaret biti)
+	Aritmetik artı	→	Aktar	Z	Sıfır (bayt)
−	Aritmetik eksi	0	Bit = Sıfır	V	Taşma, 2'ye tümleyen
.	Boole VE	00	Bayt = Sıfır	C	Bit 7'den elde
M_{SP}	Yığın gösterici tarafından gösterilen bellek yerinin içeriği				
				R	Sürekli reset
				S	Sürekli set
				↕	Test; doğruysa set et,aksi halde temizle
				•	Etkilenmez

Not: Akümülatör adresleme modu komutlar, anlaşılır adresleme sütununa konulmuştur.

İndis Yazmacı ve Yığın İşlemleri Komutları

GÖSTERİCİ İŞLEMİ	BELLENİR	Adresleme Modu: Derhal OP	Derhal ~	Derhal #	Doğrudan OP	Doğrudan ~	Doğrudan #	İndislenmiş OP	İndislenmiş ~	İndislenmiş #	Genişletilmiş OP	Genişletilmiş ~	Genişletilmiş #	Anlaşılır OP	Anlaşılır ~	Anlaşılır #	BOOLE ARİTMETİK İŞLEM	Durum Kodu Kaydedicisi 5 H	4 I	3 N	2 Z	1 V	0 C
İndis yazmacını karşılaştır	CPX	8C	3	3	9C	4	2	AC	6	2	BC	5	3				$X_H - M, X_L - (M+1)$	•	•	⑦	↕	⑧	•
İndis yazmacını bir azalt	DEX													09	4	1	$X - 1 \rightarrow X$	•	•	•	↕	•	•
Yığın göstergesini bir artır	DES													34	4	1	$SP - 1 \rightarrow SP$	•	•	•	•	•	•
İndis yazmacını bir artır	INX													08	4	1	$X + 1 \rightarrow X$	•	•	•	↕	•	•
Yığın göstergesini bir artır	INS													31	4	1	$SP + 1 \rightarrow SP$	•	•	•	•	•	•
İndis yazmacını yükle	LDX	CE	3	3	DE	4	2	EE	6	2	FE	5	3				$M \rightarrow X_H, (M+1) \rightarrow X_L$	•	•	⑨	↕	R	•
Yığın göstericiyi yükle	LDS	8E	3	3	9E	4	2	AE	6	2	BE	5	3				$M \rightarrow SP_H, (M+1) \rightarrow SP_L$	•	•	⑨	↕	R	•
İndis yazmacını sakla	STX				DF	5	2	EF	7	2	FF	6	3				$X_H \rightarrow M, X_L \rightarrow (M+1)$	•	•	⑨	↕	R	•
Yığın göstericiyi sakla	STS				9F	5	2	AF	7	2	BF	6	3				$SP_H \rightarrow M, SP_L \rightarrow (M+1)$	•	•	⑨	↕	R	•
İndis Yığın Göster	TXS													35	4	1	$X - 1 \rightarrow SP$	•	•	•	•	•	•
Yığın göster İndis yazmacı	TSX													30	4	1	$SP + 1 \rightarrow X$	•	•	•	•	•	•

İŞLEME	BELLEME	Adresleme Modu: Göreceli OP	Göreceli ~	Göreceli #	İndislenmiş OP	İndislenmiş ~	İndislenmiş #	Genişletilmiş OP	Genişletilmiş ~	Genişletilmiş #	Anlaşılır OP	Anlaşılır ~	Anlaşılır #	DALLANMA TESTİ	Durum Kodu Kaydedicisi* 5 H	4 I	3 N	2 Z	1 V	0 C
Koşulsuz dallan	BRA	20	4	2										None	•	•	•	•	•	•
Elde reset dallan	BCC	24	4	2										C = 0	•	•	•	•	•	•
Elde set ise dallan	BCS	25	4	2										C = 1	•	•	•	•	•	•
= sıfır ise dallan	BEQ	27	4	2										Z = 1	•	•	•	•	•	•
≥ sıfır ise dallan	BGE	2C	4	2										N⊕V = 0	•	•	•	•	•	•
> sıfır ise dallan	BGT	2E	4	2										Z + (N⊕V) = 0	•	•	•	•	•	•
Dahna yüksekse dal-	BHI	22	4	2										C + Z = 0	•	•	•	•	•	•
≤ sıfır ise dallan	BLE	2F	4	2										Z + (M⊕V) = 1	•	•	•	•	•	•
Daha alçak veya aynı ise dallan	BLS	23	4	2										C + Z = 1	•	•	•	•	•	•
< sıfır ise dallan	BLT	2D	4	2										N⊕V = 1	•	•	•	•	•	•
Eksi ise dallan	BMI	2B	4	2										N = 1	•	•	•	•	•	•
Sıfır değil ise dallan	BNE	26	4	2										Z = 0	•	•	•	•	•	•
Taşma resetse dallan	BVC	28	4	2										V = 0	•	•	•	•	•	•
Taşma set ise dallan	BVS	29	4	2										V = 1	•	•	•	•	•	•
Artı ise dallan	BPL	2A	4	2										N = 0	•	•	•	•	•	•
Alt yordama dallan	BSR	8D	8	2										Özel işlemlere bakın	•	•	•	•	•	•
Atla	JMP				6E	4	2	7E	3	3					•	•	•	•	•	•
Alt yordama atla	JSR				AD	8	2	BD	9	3					•	•	•	•	•	•
İşlem yok	NOP										01	2	1	Sadece program sayıcıyı iletir	•	•	•	•	•	•
Kesmeden geri dön	RTI										3B	10	1		⑩					
Alt yordamdan geri dön	RTS										39	5	1		•	•	•	•	•	•
Yazılım kesmesi	SWI										3F	12	1	Özel İşlemlere bakın	•	•	•	•	•	•
Kesmeyi bekle	WAI										3E	9	1		•	⑪	•	•	•	•

*WAI, VMA olacak tutulurken, adres yolunu, RIW ve veri yolunu üç-durumlu modda

TABLO 10.1 (Devamı)
DURUM KODU KAYDEDİCİ İLE İLGİLİ KOMUTLAR

İŞLEM	BELLENİR	Anlaşılır OP	Anlaşılır ~	Anlaşılır #	BOOLE İŞLEMİ	Durum Kodu Kaydedicisi H	I	N	Z	V	C
Eldeyi Temizleme	CLC	0C	2	1	0 → C						R
Kesme maskesini temizle	CLI	0E	2	1	0 → I		R				
Taşmayı temizle	CLV	0A	2	1	0 → V					R	
Eldeyi set et	SEC	0D	2	1	1 → C						S
Kesme maskesini set et	SEI	0F	2	1	1 → I		S				
Eldeyi set et	SEV	0B	2	1	1 → V					S	
Akümülatör A → CCR	TAP	06	2	1	A → CCR	(5)					
CCR → Akümülatör (A)	TPA	07	2	1	CCR → A						

Durum Kodu Kaydedici Notları (Test doğru ise bit set edilir, aksi halde temizlenir):

1 (BIR V) Test: Sonuç = 1000000?
2 (BIR C) Test: Sonuç = 0000000?
3 (BIR C) Test: En önemli BCD karakterin onluk değeri 9'dan büyük mü? (Önceden set edilmişse, temizlenmez)
4 (BIR V) Test: Yürütmeden önce işlenen=10000000
5 (BIR V) Test: Kaydırma olduktan sonra N⊕ C'nin sonucuna eşitle
6 (BIR V) Test: En önemli baytın işaret bili = 1?
7 (BIR N) Test: En önemli bayttaki çıkarılmasından 2'ye tümleyen taşma
8 (BIR V) Test: Sonuç 0'dan az? (Bit 15 = 1)
9 (BIR N) Durum kodu kaydedicisinin yığından yükle (Özel işlemlere bakın)
10 (Alt) Kesme olunca set olur. Daha önce set edilmişse, bekleme durumundan çıkmak için bir maskelenmeyen kesme gerekir.
11 (BIR I) A akümülatörünün içeriğine göre set edilir.
12 (Alt)

KAYNAK

İndislenmiş adresleme modu, işlenenini bir etkin adresten alır. Bu adres, komutun ikinci baytındaki kaydırma sayısını, 16-bitlik indis kaydedicisinin içeriğine ekleyerek hesaplanır. İşlenen alanındaki bir *X*, indislenmiş mod kullanıldığını belirtir. Örneğin, indis kaydedicisi 4321 H taban adresini içeriyorsa, ADDA $30, *X* komutu, yürütüldüğünde, önce işleneni alacağı etkin adresi hesaplar. Bunu, belirtilen 30H kaydırmasını, taban adresi 4321 H'ye eklemekle 4351 H etkin adresini bularak yapar. (Not: İndis kaydedicisinin içeriği değiştirilemez. İndis kaydedicisi hâlâ taban adresi 4321 H yi içermektedir.) Bundan sonra, 6800, bellekteki 4351 H etkin adresindeki baytı getirir ve bu baytı *A* akümülatörünün içeriğine ekler. Toplama sonucu, *A* akümülatöründe bırakılır. Belleğin etkin adresindeki bayt değişmez. Kaydırma 8-bitlik bir miktar olduğundan, 0 ile 255 arasında bir değer olabilir. 6800 için bu sayı sadece pozitif olabilir. 8080A nın indislenmiş adresleme modu yoktur, fakat Z80A hem *IX* hem de IY kaydedicileriyle indislenmiş adresleme yapabilir.

Genişletilmiş adresleme, işlenenin adresini saklamak için komutun ikinci ve üçüncü baytlarını kullanır. 8080A ve Z80A işlemcilerdekinin tersine, ikinci bayt, yüksek mertebeli adres baytını içerir. Komutun üçüncü baytı, alçak mertebeli adres baytını içerir. Örneğin, ADDA $073A komutu, bellekte ardarda gelen üç yere BB, 07, 3A olarak yazılır. Bu komut, bellek yeri 073A'nın içeriğini *A* akümülatörüne ekler. Birçok 8080A ve Z80A komutu bu modu kullanır. 8080A'nın JMP, CALL, LDA ve STA komutları bunun örnekleridir.

Anlaşılır adresleme modu, işlenir adresini komutun op kodunda içerir. Örneğin indis kaydedicisini bir azalt (DEX) komutu sadece bir bayt gerektirir çünkü indis kaydedicisiyle olan ilişki 09 op kodu içine kodlanmıştır. 8080A MOV komutları, bu adresleme modunun örnekleridir.

Akümülatör adreslemesi, anlaşılır adreslemenin özel bir durumudur. İşlenen, akümülatörlerden biridir ve bu, komutun op kodunda belirtilir. Akümülatör *A*'yı temizleyen CLRA komutu buna bir örnektir. Bu modu kullanan komutlar sadece 1 bayt gerektirir. DAA, CMA ve döndürme komutları, 8080A için bu modun örnekleridir.

Göreceli adresleme, 6800 adresleme modlarının sonucudur. Göreceli adresleme de, komutun ikinci baytında içerilen bir kaydırma, program sayıcıya eklenir. Bir sonraki komut, bu sonuç adresten getirilir. Bu mod, atlama ve dallanma komutları için kullanılır. 6800 için kaydırma ikinci baytta bir işaretli 2'ye tümleyen sayı olarak saklanır. Kaydırma, -128 ile +127 arasında olabilir. Program sayıcı her getirmeden sonra otomatik olarak arttığından, kaydırma, kaydırma baytının saklandığı adresten sonraki adrese eklenir. Bu nedenle komut adresinden itibaren ölçüldüğünde, göreceli adresleme aralığı -126 'dan +129 'a kadar olur.

Kaydırma, program sayıcıya eklendiğinde alt 8 bitten çıkan bir elde otomatik olarak üst adres baytına eklenir. Daha sonra kaydırmaların nasıl hesaplandığını göstereceğiz.

KOMUT KÜMESİ AÇIKLAMALARI

Çoğu üreticiler, mikroişlemci komutlarının işlevlerini tanımlamak için bir sembolik kısa yazılış kullanırlar. Örneğin, Tablo 10-1 deki ADDA komutu, $A+M \rightarrow A$ şeklinde gösterilir. Bu, bir derhal bayt veya bir adreslenen bellek yerinin içeriğinin *A* akümülatörü içeriğine eklendiği ve toplamın *A* akümülatöründe bırakıldığı demektir. Gördüğünüz gibi sembolik şekil çok daha kısadır ve sözlü tanıma göre belki daha açıktır. Bayrak harflerinin altındaki semboller, bir bayrağın o komut tarafından etkilenip etkilenmediğini belirtir. Dik bir ok o bayrağın etkilendiğini belirtir, bir nokta o bayrağın etkilenmediğini belirtir. *R*, bayrağın her zaman reset olduğunu, *S* bayrağın her zaman set olduğunu ve daire içine alınmış bir sayı tablonun sonundaki bir nota bakmanızı söyler. Bu sembolik gösterilişlere alışmanız çok önemlidir; çünkü bunlar bir bakışta bir komutun etkisini görmenize imkân verir.

Tablo 10-1'deki 6800 komutlarının çoğu isimden,veya sembolik ifadesinden oldukça açıktır. Daha az açık olan bazı komutları, izleyen kısımlarda açıklayıp örnekleyeceğiz.

AKÜMÜLATÖR VE BELLEK KOMUTLARI

Birbirine benzeyen iki komut ANDA ve BITA belirtilen bellek yerinin içeriğini *A* akümülatörleriyle VE'ler. Ancak ANDA komutları VE'lemenin sonucunu *A* akümülatörüne koyar. BITA komutları bayrakları etkiler, fakat bellek yerinin içeriğini veya *A* akümülatörünü etkilemez. Böylece BITA komutu test edilen sayıları değiştirmeden test etme imkânı sağlar.

COM komutları işlenenin bütün bitlerini ters çevirerek 1'e tümleyeni oluşturur. NEG komutları 2'ye tümleyeni oluşturur.

6800 komutlarından çok kullanılan ikisi LDAA ve LDAB 'dir. Bu komutlar, bir derhal baytı veya bir bellek yerinin içeriğini belirlenen akümülatöre kopyalar. Bu komutlar için dört adresleme modu bulunmaktadır. Örneğin LDAA #$7A, hex sayı 7A'yı *A* akümülatörüne derhal kopyalar. LDAB $37 doğrudan bellek adresi 37H yi *B* akümülatörüne kopyalar. LDAA $4FC0, belleğin 4FC0H genişletilmiş adresinin içeriğini *A* akümülatörüne kopyalar. LDAB $53, *X*, indis kaydedicisine 53H ekleyerek hesaplanan etkin adresteki belleğin içeriğini *B* 'ye kopyalar. Bir 6800 sisteminde bütün portlar bellek yerleri gibi adreslendiğinden, LDAA ve LDAB komutları aynı zamanda porttan giriş komutları olarak işlev görürler.

6800, altyordamlara giderken geri dönüş adreslerini ve kaydedicileri saklamak için RAM da bir yığın kullanır. PSHA komutu, *A* akümülatörünün içeriğini, yığın gösterici tarafından gösterilen bellek yerine kopyalar. Bundan sonra yığın gösterici otomatik olarak 1 azaltılır. 6800 ün önce bir baytı ittirdiğine ve sonra yığın göstericiyi azalttığına dikkat edin. Halbuki 8080A önce yığın göstericiyi azaltmakta sonra baytı ittirmektedir. Programların hatasını ayıklarken bunu unutmamak çok önemlidir. Bir altyordamın sonunda akümülatöre verileri geri koymak için PUL komutları kullanılır. PULA, yığın göstericiyi 1 artırır ve sonra gösterilen bellek yerinin içeriğini *A*'ya kopyalar. Yığında saklanan veri etkilenmez.

6800 döndürme ve kaydırma komutları, *A* akümülatörünün, *B* akümülatörünün veya bir bellek yerinin içeriğini kaydırabilir. ROL komutu elde üzerinden sola döndürür ve ROR komutu elde üzerinden sağa döndürür. ASL komutu en önemsiz bit konumuna bir 0 kaydırır ve en büyük değerlikli biti eldeye kaydırır. LSR komutu MSB konumuna bir 0 koyar ve LSB 'yi eldeye kaydırır. ASR komutu LSB eldeye gitmek üzere baytı sağa kaydırır ancak MSB aynı kalır. Bu komutlar için Tablo 10-1'deki diyagramlara bakın.

Diğer çok kullanılan bir 6800 komut çifti STAA ve STAB dir. Bu komutlar belirtilen akümülatörü bir doğrudan genişletilmiş veya indislenmiş bellek adresine kopyalar. Akümülatörün içeriği değişmez. Bir 6800 sisteminde portlar bellek yerleri gibi adreslendiğinden, bu komutlar porta çıkış verme komutları olarak işlev görür.

TST komutları, bir akümülatör veya bellek yeri içeriğinin 0 veya negatif olduğunu belirlemek için kullanılır. TST komutları, test edilen baytı değiştirmeden bayrakları etkiler.

İNDİS KAYDEDİCİ VE YIĞIN KOMUTLARI

6800'ün indis kaydedicisinin ve yığın gösterici kaydedicisinin her ikisinin de 16 bitlik olduğunu unutmayın.

LDX komutu, indislenmiş adresleme modu komutlarıyla kullanılmak üzere indis kaydedicisine taban adresini yüklemek için kullanılır. Bu komut ayrıca indis kaydedicisi sayıcı olarak kullanıldığında başlangıç değerini ilklendirmek için kullanılır. LDX 2 derhal baytı veya ardarda gelen iki bellek yerinin içeriklerini indis kaydedicisine yükler. Örneğin, LDX #$075A indis kaydedicisine 075AH yükler. LDX $3A7B, indis kaydedicisinin üst 8 bitine 3A7BH adresinin içeriğini ve indis kaydedicisinin alt 8 bitine 3A7CH adresinin içeriğini kopyalar.

DEX komutu, indis kaydedicisini 1 azaltır ve INX komutu, indis kaydedicisini 1 artırır. Eğer işlem sonucunda indis kaydedicisinin 16 biti de 0 olursa

her iki komut da sıfır bayrağını set eder. Bu, indis kaydedicisinin uzun gecikme döngülerinde kullanılmasını kolaylaştırır.

STX, indis kaydedicisini iki ardarda bellek yerine kopyalar. Örneğin STX $3A7B indis kaydedicisinin üst 8 bitini 3A7BH adresine ve alt 8 bitini 3A7CH adresine kopyalar. Bir alt yordam sırasında indis kaydedicisini ittirmek için hiçbir 6800 komutu olmadığından, bu, altyordamın başlangıcında bir STX komutuyla ve sonunda bir LDX komutuyla gerçekleştirilir. Örneğin, STX $3A7B, indis kaydedicisini 3A7BH ve 3A7CH da saklar. LDX $3A7B indis kaydedicisinin eski değerlerini 3A7BH ve 3A7CH adreslerinden alıp koyar.

CPX komutu, 2 derhal baytı veya iki ardarda bellek yerinin içeriklerini indis kaydedicisinin içerikleriyle karşılaştırır. Bellekteki ilk bayt X ın yüksek mertebeli baytıyla ve bellekteki bir sonraki bayt X'in alçak mertebeli baytıyla karşılaştırılır. Bir karşılaştırma çıkarma ile yapılır, ancak ne indis kaydedicisinin ne de belleğin içerikleri etkilenir. Eğer indis kaydedicisi karşılaştırılan kelimeye eşitse bu takdirde sıfır bayrağı set olur. Eğer sonucun bit 15'i 1 ise, negatif bayrağı set olur. Eğer çıkarma 16 bit için nizami 2 'ye tümleyen aralığının dışında bir sonuç üretmişse taşma bayrağı set edilir.

LDS komutu, yığın göstericiyi ilklendirmek için kullanılır. Örneğin LDS #$00FF, yığın gösterciyi 00FFH derhal sayısına ilklendirir. LDS $0A00 yığın göstericinin yüksek baytına 0A00H'den ve alçak baytına 0A01H 'den yükleme yapar.

TSX komutu indis kaydedicisine yığın göstericinin içeriğinin 1 fazlasını yükler. Yığın gösterici değişmez.

TXS komutu yığın göstericiyi, indis kaydedicisinin içeriğinin 1 eksiği ile yükler. İndis kaydedicisinin içeriği değişmez.

ATLAMA VE DALLANMA KOMUTLARI Atlama ve dallanma komutları çoğunlukla birbirinin yerine kullanılırlar. Ancak 6800'de bunlar arasında fark vardır. Dallanma komutları sadece göreceli adresleme kullanır, böylece bunlar program yürütmesini -126 dan +129 adrese kadar değiştirebilir. Atlama komutları program yürütülmesini belleğin herhangi bir yerine atlatabilir, çünkü bunlar indislenmiş veya genişletilmiş adresleme kullanır. Koşulsuz atlama (JMP), koşulsuz dallanma (BRA) ve bütün koşullu dallanma komutları program yürütülmesini belirtilen adrese taşırlar.

JSR (altyordama atla) ve *BSR (altyordama dallan)*, 8080A/8085A'nın çağırma komutlarına benzer. Her ikisi de geri dönüş adresini yığında saklar ve program sayıcıyı altyordamın başlangıç adresiyle yükler. 6800'ün çok çeşitli koşullu dallanma komutları olmasına rağmen 8080A/8085A'nın koşullu çağırma komutları veya parite bayrağı yoktur.

RTI ve RTS komutları geri dönüş adresini yığından çeker ve program yürütmesini o adrese geri verir. RTI ayrıca, 6800 bir kesmeyi tanıdığında yığına itilmiş olan durum kod kaydedicisi, akümülatörler ve indis kaydedicisinin eski değerlerini yerine koyar.

SWI komutu, yazılım kesmesidir. Yürütüldüğünde bu komut bütün kaydedicileri ve program sayıcının içeriğini yığına iter. Bundan sonra CPU istenilen kesme hizmet yordamının başlangıç adresini almak üzere FFFA ve FFFB adresine gider.

Kesmeyi bekle (WAI) komutu kaydedici içeriklerini ve program sayıcıyı yığına iter. Bundan sonra CPU, adres yolunu, veri yolunu ve R/$\overline{W}$ kontrol hattını kayan hale getirir. VMA çıkışı alçaktır. İşlemci, bir kesmeyi-bekleme durumundadır. İşlemci, bir maskelenmeyen kesme ($\overline{NMI}$) veya bir maskelenebilir kesme ($\overline{IRQ}$) olmasını başka hiçbir şey yapmadan beklemektedir. Bu kesmeler ve RESET'ten başka işlemciyi kesmeyi-bekleme durumundan çıkarabilen yol yoktur. $\overline{IRQ}$, bir kesme maskesini temizle (CLI) komutuyla maskesi kaldırılmadıkça bir çıkışa sebep olamaz. Motorola, $\overline{IRQ}$ kesmesinin maskesini kaldırmak ve kesmeyi-bekleme durumuna girmek için bir NOP-CLI-WAI komut dizisi önermektedir. Bütün kaydediciler vb. halen yığına itilmiş olduklarından WAI komutu, normal yönteme göre kesmeye daha çabuk cevap verilmesine imkân verir. Kesildiğinde, işlemci doğrudan kesme hizmet yordamına atlar. Ancak işlemci beklerken hiçbir şey yapmaz. Bu bölümde daha sonraki bir kısım 6800 donanım ve yazılım kesmelerinin etkilerini özetlemektedir.

DURUM KODU KAYDEDİCİSİ İLE İLGİLİ KOMUTLAR

Bu komutların çoğu isimleriyle açıkça tanımlanmışlardır. SEI (kesme maskesini set et) komutunun $\overline{IRQ}$ kesmesini yetkisiz kıldığına ve CLI komutunun kesmeyi yetkilendirdiğine dikkat edin. TAP ve TPA komutları *A* akümülatörüyle durum kodu kaydedicisinin içeriklerini yer değiştirir.

6800 komut kümesi hakkında anlatacaklarımız burada bitmektedir. Bu komutların nasıl kullanıldığı hakkında size bir fikir verebilmek için şimdi bazı örnek programlar gösterilecektir.

PROGRAM ÖRNEKLERİ

ÖRNEK 1: İlk programlama örneği, Şekil 8-21a'daki akış tablosunun 6800 için nasıl programlandığını göstermektedir. Program, bir veri hazır sinyali varolduğunda, bir veri örneği alır. Veri örneği bellekte saklanmaktadır. Bu süreç, 10 örnek alınıp saklanıncaya kadar tekrarlanır.

Şekil 10-2 bu akış tablosu için 6800 programını göstermektedir. Bu özel örnek, adres 8006H'deki portun LSB sindeki veri hazır darbesini kontrol etmek için yazılmıştır. Program, veriyi 8004H adresindeki porttan alır. Veri örnekleri 00A0H adresinden başlamak üzere RAM'da saklanmaktadır.

Şekil 10-2'deki programda bir sayının önündeki $ işareti çeviriciye sayının onaltılık (hex) olduğunu belirtir. Bir # işareti derhal modu adreslemenin kullanıldığını gösterir. Bu arada bu programın yedi mümkün adresleme modundan altısını kullandığını belirtelim.

LDX #$00A0, 10 örneğin saklanma yerinin başlangıç adresini indis kaydedicisine yükler. LDAB #$0A örnek sayıcı olarak hazırlamak üzere *B* akümülatörüne 10 yükler. LDAA $8006, 8006H adresindeki veri hazır sinyal portundan bir bayt alarak A akümülatörüne yükler. BITA #$01 bayrakları set etmek üzere akümülatördeki sayı ile 01H sayısını VE ler. ANDA #$01 komutu da kullanılabilirdi, ancak BITA#$01, akümülatörün içeriğini değiştirmeden bayrakları set etme avantajına sahiptir. Her iki komut için de 8006H portundaki baytın LSB'si 1 olmadıkça sıfır bayrağı yüksek kalacaktır. BEQ LOOP komutu, veri hazır sinyali bu LSB'de gözükünceye kadar programı geriye dallandırmaya devam eder.

Veri hazır darbesi gözükünce, LDAA $8004 komutu, 8004H adresindeki veri portundan örneği alır, *A* akümülatörüne koyar. STAA $00, X komutu veri baytlarını belleğe taşımak için indislenmiş adresleme kullanır. $00'lık birkaydırma belirtildiğinden ilk örnek indis kaydedicisine başlangıçta yüklenmiş 00A0H adresine saklanır. Burada *X* indislenmiş adresleme kullanıldığını söyler.

INX komutu, bir sonraki saklama adresini göstermek üzere indis kaydedicisini bir artırır. DECB, *B* akümülatöründeki örnek sayısı sayıcısını bir azaltır. BNE, eğer sayıcı 0 olmamışsa bir başka veri hazır darbesi beklemek üzere programı geri gönderir.

PROGRAM İSMİ: PORT 8004H'DEN ON ÖRNEK

```
;PORT ADRESİ 8004 H'DEN 10 VERİ ÖRNEĞİ ALMAK İÇİN PROGRAM.
;HER ÖRNEK, 8006 H'DAKİ PORTUN LSB SİNDE BİR "VERİ HAZIR" DARBESİ
;OLDUĞUNDA
; ALINACAKTIR. ÖRNEKLER, 00A0H ADRESİNDEN BAŞLAMAK ÜZERE BELLEKTE
; SAKLANACAKTIR.
;KULLANILAN BELLEK : 00A0 - 00A9 H; 8004 H, 8006 H VE PROGRAM İÇİN 1,5 H BAYT.
;ÇAĞRILAN YORDAMLAR : YOK
ADRES KOMUT ETİKET OP KOD İŞLENEN   AÇIKLAMALAR
0000   CE          LDX    #$00A0  ; İNDİS KAYDEDİCİSİNİ, SAKLAMA YERİNİN
 01    00                          ; BAŞLANGIÇ ADRESİYLE YÜKLE
 02    A0                          ; (DERHAL)
 03    C6          LDAB   #$0A    ; B AKÜMÜLATÖRÜNDEKİ ÖRNEK
                                   ; SAYICIYI
 04    0A                          ; İLKLENDİR (DERHAL)
 05    B6    LOOP: LDAA   $8006   ; 8006H ADRESİNDEKİ PORTTAN VERİ
 06    80                          ; HAZIR SİNYALİNİ AL
 07    06                          ; (GENİŞLETİLMİŞ)
 08    85          BITA   #$01    ; LSB NİN 1 OLUP OLMADIĞINI TEST ET
 09    01                          ; (DERHAL)
 0A    27          BEQ    LOOP    ; 8006H DEN GELEN LSB 1 E EŞİT OLUNCA
 0B    F9                          ; YA KADAR TEKRARLA (GÖRECELİ)
 0C    B6          LDAA   $8004   ; VERİ HAZIR-VERİYİ 8004H PORTUNDAN
 0D    80                          ; AL (GENİŞLETİLMİŞ)
 0E    04
 0F    A7          STAA   $00, X  ; VERİYİ, BELLEKTE İNDİS KAYDEDİCİ
                                   ; TARAFINDAN
0010   00                          ; GÖSTERİLEN ADRESE SAKLA
                                   ; (İNDİSLENMİŞ)
 11    08          INX            ; İNDİS KAYDEDİCİSİNİ BİR ARTIR
                                   ; (ANLAŞILIR)
 12    5A          DECB           ; B SAYICISINI BİR AZALT(AKÜMÜLATÖR)
 13    26          BNE    LOOP    ; 10 ÖRNEK ALININCAYA KADAR
                                   ; DÖNGÜYÜ
 14    F0                          ; TEKRARLA (GÖRECELİ)
```

Şekil 10-2 Bir veri sinyali varolduğunda 10 veri örneği alan 6800 programı.

Şekil 10-2, ayrıca programın çevrilmiş şeklini yani makina kodunu göstermektedir. Her komutun adresleme modunu, Tablo 10-1 ve makina kodunu kullanarak kontrol edebilirsiniz. Belleğe önce adreslerin yüksek mertebeli baytlarının yazıldığına dikkat edin. Bu, 8080A/8085A ve Z80A'da yapılanın tersidir.

Göreceli adresleme kullanan BEQ ve BNE komutları için kaydırma miktarının hesaplanması gerekir. Eğer program bir çeviriciyle kodlanmışsa bu, çevirici tarafından otomatik olarak yapılır. Elde yapılan çevirilerde kaydırma şu şekilde hesaplanır. Program sayıcı, kaydırmayı getirdikten sonra otomatik olarak bir artar, böylece BEQ komutu için kaydırma 010C adresinden itibaren ölçülür. 0105 adresine dallanma istenmektedir. Fark, -7'lik bir kaydırmadır. BEQ komutu için bu kaydırma ikiye tümleyen şeklinde 11111001 yani $F9_{16}$ olarak gösterilir. MSB konumundaki 1 kaydırmanın negatif olduğunu belirtir. Bu, ikiye tümleyen şeklinde 11110000 yani $F0_{16}$ olarak kodlanır.

ÖRNEK 2: Bu program Şekil 8-22 de gösterildiği gibi sabit zaman aralıklarında 10 veri örneği alır. Şekil 8-22 deki programda aralık 1ms idi, fakat burada daha uzun gecikmelerin nasıl elde edileceğini de göstereceğiz.

Şekil 10-3 8004H adresindeki porttan 10 örnek alan ve bunları 00A0H adresinden başlayarak bellekte saklayan 6800 çeviri dili programı göstermektedir. Ayrıca 1 ile 65535 ms ye kadar olan aralıklarda örnekleme imkânı veren üç altyordam da gösterilmiştir.

Program yığın göstericiyi RAM daki bir adrese ilklendirerek başlar. Bu, BSR, JSR veya kesmeler kullanıldığında gereklidir çünkü bütün bunlar geri dönüş adresini, vb. saklamak için yığını kullanırlar. Baytlar itildikçe yığın, bellekte aşağı doğru büyüdüğü için yığın gösterici kullanılabilir. RAM'daki en yüksek adrese ilklendirilebilir.

Bundan sonra indis kaydedicisi ve örnek sayıcı ilklendirilir. Bir veri örneği 8004H adresindeki porttan alınır ve indis kaydedicisi tarafından gösterilen bellek adresine saklanır. Bu tamamlandığında gecikme yordamlarından birisi çağrılır. Eğer yordam 129 bayt içinde bir yerdeyse bir BSR kullanılabilir. Eğer değilse, bu takdirde bir JSR kullanılmalıdır. Eğer geciktirme yordamı programda sadece bir defa kullanılacaksa, bu takdirde doğrudan ana program içine yazılabilir. Ancak geciktirme döngüsünü bir altyordam olarak yazmak, bir defa yazıldıktan sonra programın herhangi bir yerinde kullanılmasına imkân verir.

PROGRAM İSMİ: GECİKTİRMELİ VERİ ÖRNEKLERİ:

```
;  PORT ADRESİ 8004H'DEN 10 VERİ ÖRNEĞİ ALMAK VE BUNLARI 0A00H
;  ADRESİNDEN
;  BAŞLAMAK ÜZERE BELLEKTE SAKLAMAK İÇİN BİR MC6800 PROGRAMI.
;  FARKLI GECİKME YORDAMLARI KULLANARAK, ÖRNEKLER 1 MS'DEN
;  65535 MS'YE
;  KADAR ARALIKLARDA ALINABİLİR
;  KULLANILAN BELLEK:  00F0-00FFH (YIĞIN); 00A0-00A9H (VERİ SAKLAMA) ;
;  8004H (GİRİŞ PORTU)
;  ÇAĞRILAN YORDAMLAR: WAITMS, WAITMR, VEYA WAITLT.

ETİKET   OP KOD   İŞLENEN          AÇIKLAMALAR
         LDS      #$00FF       ;  YIĞIN GÖSTERİCİYİ İLKLENDİR
         LDX      #$00A0       ;  İNDİS KAYDEDİCİSİNİ İLKLENDİR
         LDAB     #$0A         ;  ÖRNEK SAYICIYI AYARLA
GET DAT: LDAA     $8004        ;  8004H ADRESİNDEKİ PORTTAN VERİYİ AL
         STAA     $00,X        ;  VERİ ÖRNEĞİNİ BELLEKTE SAKLA
         BSR      WAIT–        ;  GECİKME ALTYORDAMINA DALLAN
         INX                   ;  İNDİS KAYDEDİCİSİNİ BİR ARTIR
         DECB                  ;  SAYICI BİR AZALT VE
         BNE      GET DAT      ;  10 ÖRNEK ALININCAYA KADAR DÖNGÜ-
                               ;  DE KAL
WAITMS:  LDAA     #165 (DECIMAL) ;  1 MS LİK GECİKME SABİTİNİ YÜKLE
LOOP:    DECA                 ;  A YI BİR AZALT VE
         BNE      LOOP         ;  SIFIR OLUNCAYA KADAR DÖNGÜDE KAL
         RTS                   ;  ALTYORDAMDAN GERİ DÖN
WAITMR:  PSHB                  ;  B AKÜMÜLATÖRÜNÜ YIĞINDA SAKLA
         LDAB     # NUMBER     ;  255 MS'YE KADAR GECİKME
                               ;  (MS CİNSİNDEN)
LOOP1:   LDAA     #165         ;  1 MS LİK GECİKME SABİTİNİ YÜKLE
                               ;  (ONLUK 165)
LOOP2:   DECA                  ;  A'YI BİR AZALT VE
         BNE      LOOP2        ;  SIFIR OLUNCAYA KADAR DÖNGÜDE KAL
         DECB                  ;  B'Yİ AZALT VE
         BNE      LOOP1        ;  SIFIR OLUNCAYA KADAR DÖNGÜDE KAL
         PULB                  ;  ÖNCEKİ B NIN İÇERİĞİNİ YERİNE KOY
         RTS                   ;  ALTYORDAMDAN GERİ DÖN
WAITLT:  STX      $0AF0        ;  İNDİS KAYDEDİCİSİNİ SAKLA
         LDX      # NUMBER     ;  65535 MS'YE KADAR GECİKME
                               ;  (MS CİNSİNDEN)
LOOP1:   LDAA     #165         ;  1 MS'LİK GECİKME SABİTİNİ YÜKLE
LOOP2:   DECA                  ;  A'YI BİR AZALT VE
         BNE      LOOP2        ;  SIFIR OLUNCAYA KADAR DÖNGÜDE KAL
         DEX                   ;  İNDİS KAYDEDİCİSİNİ BİR AZALT VE
         BNE      LOOP1        ;  SIFIR OLUNCAYA KADAR DÖNGÜDE KAL
         LDX      $00B0        ;  İNDİS KAYDEDİCİSİNİN ESKİ DEĞERİNİ
                               ;  YERİNE KOY
         RTS                   ;  ALTYORDAMDAN GERİ DÖN
```

Şekil 10-3 1 den 65535 ms'ye kadar aralıklarda 10 veri örneği alan 6800 programı.

WAITMS yordamı 165_{10} sayısını A akümülatörüne yükler ve 0'a kadar azaltır. Her komutun çevrim sayısı Tablo 10-1'de verilmişitir. Eğer 6800 1 MHz lik bir saat kullanırsa her komutun mikrosaniyeler cinsinden süresi çevrim sayısına eşit olur. Bunu bildiğinize göre komut sürelerini toplayarak WAITMS yordamının süresini hesaplayabilirsiniz.

Daha fazla beklemek için WAITMR gecikme yordamı, 255 ms'ye kadar bir gecikme elde etmek üzere içiçe döngüler kullanılır. Yordamın başlangıcında, B akümülatörünün içeriğini yığına ittirmek gerekmektedir, çünkü B akümülatörü hem ana programda hem de altyordamda kullanılmaktadır. Yordam, istenildiği kadar sayıda 1 ms lik yordamı yükler ve sayar. İstenilen milisaniye sayısı B kaydedicisine yüklenir ve her 1 ms lik döngü yürütüldüğünde azaltılarak 0 oluncaya kadar işlemler tekrarlanır.

65535 ms ye kadar gecikmeye imkân veren bir üçüncü yordam WAITLT, benzer bir yöntem kullanır. 16-bitlik indis kaydedici, milisaniye sayıcısı olarak kullanılmaktadır. STX $00B0 komutu, indis kaydedicisinin içeriğini bellekte saklayarak korur. İstenilen gecikme (milisaniyeler cinsinden), LDX komutuyla indis kaydedicisine yüklenmektedir. LOOP2, 1 ms'lik yordamdır ve LOOP1 indis kaydedicisindeki sayıyı azaltarak sayar. Ana programa geri dönmeden önce, LDX $00B0 komutuyla indis kaydedicisine eski değeri geri konur.

Bu yordamlar oldukça basittir, fakat bunlar çalışıyorken işlemcinin başka bir iş yapamaması gibi bir dezavantajları vardır. Çoğu uygulamalarda bu, işlemci zamanının boşa harcanmasıdır. Bu nedenle, anlatmış olduğumuz 8155 gibi harici zamanlayıcılar, gecikme yordamları yerine kullanılabilir.

Gecikme yordamından geri döndükten sonra, program, bellekteki bir sonraki saklama yerini göstermek üzere indis kaydedicisini bir artırır. Bundan sonra, örnek sayıcısı bir azaltılır. Program, 10 örnek alınıp saklanıncaya kadar GETDAT'a geri dönmeye devam eder.

ÖRNEK 3: Çoklu Kesinlikli Toplama ve Çıkarma

Bölüm 12 de daha çok 6800 programlama gösterilmektedir, ancak burada iki kısa örnekle doğrudan ve indislenmiş adreslemeyi açıklığa kavuşturacağız.

6800'ün akümülatörleri ve veri taşıtı 8 bit genişliğindedir ve 6800 aritmetik komutları 8-bitlik miktarlarla çalışır. Bu, toplanacak veya çıkarılacak sayıların aralığını -128 ile +127 arasına sınırlar. Büyük sayılar, en küçük değerlikli bayttan başlamak üzere, her seferinde bir baytlık işlem yaparak, toplanır veya çıkarılabilir. Bir baytlık işlemden çıkan bir elde veya ödünç, bir sonraki bayt işleminde dikkate alınır. 2-baytlık sayılarla olan işlemlere çift kesinlikli (double precision) ve 2'den fazla baytlı sayılarla olan işlemlere çoklu kesinlikli (multiple precision) denir.

Şekil 10-4*a*, doğrudan adreslemeyi kullanarak, 2-baytlık sayıları toplayan kısa bir yordamı göstermektedir. Toplanacak sayılardan birisi, bellekte 0040H ve 0041H adreslerinde; diğeri ise 0050H ve 0051H adreslerinde saklanır ve toplam 0060H ve 0061H adreslerine saklanmaktadır. En küçük değerlikli bayt, daha alttaki adrestedir.

Sayının birinin en küçük değerlikli baytı, *A* akümülatörüne ve en büyük değerlikli baytı *B* akümülatörüne taşınır. ADDA $50 komutu bellek yeri 0050 nin içeriğini *A* akümülatörüyle toplar ve sonucu A akümülatöründe bırakır. Bu komut doğrudan adresleme kullanır, bu nedenle sadece bir 1-baytlık adres verilmiştir. Doğrudan adresleme otomatik olarak üst adres baytının 0 olduğunu varsayar. ADCB $51 komutu, 0051 adresinde saklı en büyük değerlikli baytı ve eldeyi, *B* akümülatörüne ekler. Yine doğrudan adresleme kullanılmıştır, bu nedenle komut için bellekte sadece 2 bayt gerekmektedir. ADCB komutunun yürütülmesinden sonra, A akümülatörü sonucun en küçük değerlikli baytını, *B* akümülatörü de sonucun en büyük değerlikli baytını içermektedir. STAA $60 ve STAB $61, bu sonuçları uygun bellek yerlerine taşımak için doğrudan adresleme kullanmaktadır. Doğrudan adreslemenin avantajı, sadece 2 bayt gerektirmesidir. Genişletilmiş adresleme ise her komut için 3 bayt gerektirir. Doğrudan adreslemenin dezavantajı ise, bunun sadece 00H ile FFH arasındaki yerler için kullanılabilmesidir. Bu yordam, ADDA yi SUBA veya ADCB yi SUBB ile değiştirerek çıkarma için kullanılabilir. Her baytın iki BCD basamağı gösterdiği sıkıştırılmış BCD sayılar, ADDA ve ADCB komutlarından sonra birer DAA komutu konularak doğru olarak toplanabilir.

İndislenmiş adresleme de bir bellek referansı için 2 bayt kullanmaktadır. Şekil 10-4b, çoklu kesinlikli sayıları toplamak için indislenmiş adreslemenin nasıl kullanıldığının iyi bir örneğini göstermektedir.

Önce elde temizlenir ve toplanacak bayt sayısı *B* akümülatörüne yüklenir. Toplanacak sayılardan birisi, en büyük değerlikli baytı 00A0H adresinden başlamak üzere bellekte saklıdır. Toplanacak diğer sayı, en küçük değerlikli baytı 00B0H adresinden başlamak üzere bellekte saklıdır. Sonuç, 00C0H adresinden başlamak üzere belleğe saklanmaktadır.

```
İKİ 2-BAYTLIK SAYIYI TOPLAMAK İÇİN YORDAM. TOPLANACAK SAYILARDAN
BİRİ 0040H VE 004IH ADRESLERİNDE VE DİĞERİ 0050H VE 0051H ADRESLERİNDE
SAKLIDIR. SONUÇ, 0060H VE 006IH ADRESLERİNE SAKLANIR. BÜTÜN
DURUMLARDA, EN KÜÇÜK DEĞERLİKLİ BAYT ÖNCE SAKLANMAKTADIR
(DAHA KÜÇÜK ADRESTEDİR)

ETİKET OP KOD    İŞLENEN      AÇIKLAMALAR
        LDAA     $40      ;  BELLEKTEN 16 BİTLİK SAYIYI
        LDAB     $41      ;  A VE B AKÜMÜLATÖRLERİNE YÜKLE
        ADDA     $50      ;  EN KÜÇÜK DEĞERLİKLİ BAYTLARI TOPLA
        ADCB     $51      ;  EN BÜYÜK DEĞERLİKLİ BAYTLARI TOPLA
        STAA     $60      ;  SONUCU BELLEKTE SAKLA
        STAB     $61
        ROLA              ;  ELDEYİ A NIN LSB 'İNE AL
        ANDA     # $01     ;  LSB DIŞINDAKİ BİTLERİ MASKELE
        STAA     $62      ;  SON ELDEYİ SAKLA
```

(a)

```
; İKİ ÇOK BAYTLI SAYIYI TOPLAMAK İÇİN YORDAM, SAYILARDAN BİRİ
; BELLEKTE 00A0H
; ADRESİNDEN BAŞLAMAK ÜZERE DİĞER SAYI, 00B0H ADRESİNDEN BAŞLAMAK
; ÜZERE SAKLIDIR. SONUÇ 00C0H ADRESİNDEN BAŞLAMAK ÜZERE
; SAKLANMAKTADIR.

ETİKET   OP KOD    İŞLENEN      AÇIKLAMALAR
         CLC                ;  ELDEYİ TEMİZLE
         LDAB      #N       ;  BAYT SAYISINI SAYICI OLARAK B'YE YÜKLE
         LDX       #$00A0   ;  TABAN ADRESİNİ İNDİS KAYDEDİCİSİNE YÜKLE
LOOP:    LDAA      $00,X    ;  İLK BAYTI A AKÜMÜLATÖRÜNE YÜKLE
         ADCA      $10,X    ;  İLK BAYTLARI TOPLA
         STAA      $20,X    ;  SONUCU SAKLA
         INX                ;  İNDİS KAYDEDİCİSİNİ BİR ARTIR
         DECB               ;  BAYT SAYICIYI BİR AZALT
         BNE       LOOP     ;  BÜTÜN BAYTLAR TOPLANINCAYA KADAR
                            ;  DÖNGÜDE KAL
         ROLA               ;  ELDEYİ A'NIN LSB 'SİNE AL
         ANDA      #$01     ;  LSB DIŞINDAKİ BÜTÜN BİTLERİ MASKELE
         STAA      $20,X    ;  SON ELDEYİ SAKLA
```

(b)

Şekil 10-4 Çok baytlı sayıları toplamak için 6800 yordamları. *(a)* Doğrudan adreslemeyi kullanan çift kesinlikli toplama. *(b)* Çoklu kesinlikli toplama.

LDX #$00A0, taban adresini indis kaydedicisine yükler. LDAA $00,X komutu, 00A0H adresinden *A* akümülatörüne bir bayt yükler. ADCA $10,X komutu, 00B0H adresindeki sayıyı elde ile birlikte *A* akümülatörüne ekler. STAA $20,X, sonucu 00C0H bellek adresine yükler. INX komutu indis kaydedicisini bir artırır. Böylece, program geri döndüğünde, LDAA, ADCA ve STAA komut-

ları, işlenenlerini bir sonraki adreslerden alırlar. O halde, basit bir INX komutu, üçü için de işlenenin kaynağını bir anda değiştirmektedir. DECB komutu, bayt sayıcıyı bir azaltır ve henüz 0 olmamışsa, BNE, bir başka bayt toplamı için programı geri döndürür. Toplanacak ikinci baytlar 00A1H ve 00B1H adreslerinden gelmekte ve sonuç 00C1H adresinde saklanmaktadır.

BASİT BİR MC6800 SİSTEMİ

GENEL BAKIŞ

Şekil 10-5, ROM, RAM ve I/O portlarıyla basit bir MC6800 sisteminin şemasını göstermektedir. Bu devre, Şekil 12-12'de gösterilen Motorola MEK 6800D2 geliştirme kitinin mikroişlemci kontrollu EPROM programlayıcısının temeli olarak kullanılmaktadır.

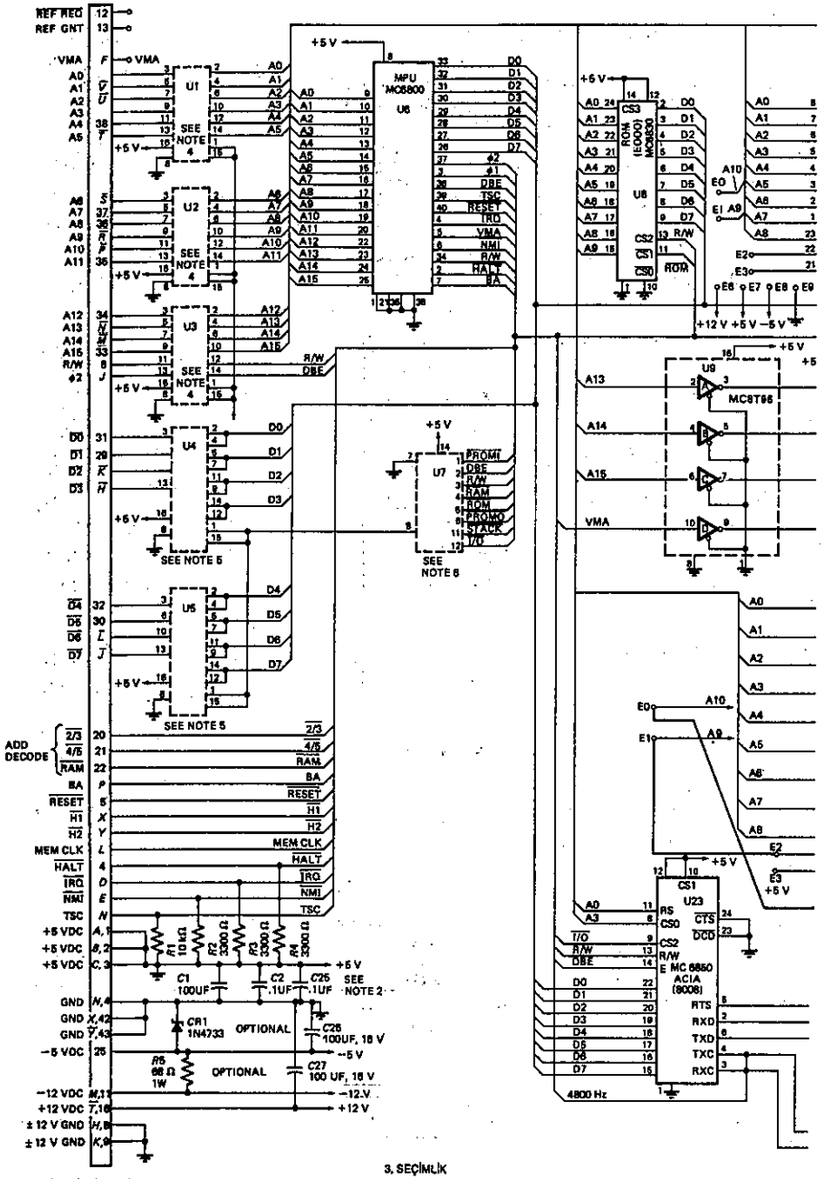
Şekil 10-5 'te, yollar entegrelerden çıktıktan sonra bunların tek bir hatta sıkıştırıldığına dikkat edin. Bu, şemalardaki sıkışıklığı azaltır, fakat bazı sinyal yollarını izlemeyi biraz zorlaştırabilir. Böyle bir şemaya yaklaşırken, CPU gibi bildiğiniz bir parça ile başlayın ve buradan itibaren devrenin gerisine doğru devam edin.

MC6800 CPU, şemanın sol üst köşesindedir. Adres ve veri yolları buradan ROM, RAM ve I/O portlarına gider. Yollar ayrıca, sistem genişletme için sürüşü artıran bazı seçimlik tamponlara da gider.

Bundan sonra, CPU'nun saat kaynağını inceleyin. MC6800 için gerekli olan iki-fazlı saat, bir MC6871B olan *U15* tarafından üretilmektedir. MC6871B, hem kristal hem de osilatör devresini içeren bir hibrit pakettir. Bu özel kart için, seçilen saat frekansı 614,4 KHz dir. MC6800 ün 2MHz'e kadar saatle çalışan tipleri vardır. Ancak, UART için 4800 Hz lik saatin kolayca türetilebilmesi için 614,4 KHz lik saat seçilmiştir. 614,4 KHz lik Ø1 ve Ø2 çıkışlarından başka, MC6871B nin bir de saat frekansının iki katı olan 1228800 Hz lik çıkışı vardır, MC14040 ikilik bölücü, istenilen 4800 Hz i vermek üzere bunu 256 'ya böler. $\overline{\text{HOLD1}}$ ve $\overline{\text{HOLD2}}$, 6871B ye girişlerdir. Bunlar, saat çevrimini uzatmada kullanılırlar ve daha sonra anlatılacaklardır.

ROM

Bundan sonra, ROM ve bunun bellek alanında haritalanmasını incelemek en iyisidir. Bölüm 9'daki gibi, izlemenizi kolaylaştırmak için bir adres çözme çalışma sayfası hazırlayın. U8 ROMu bir MC6830 1Kx8 maske-programlanmış ROM dur. Bu yonga, kullanıcının RAM programlar yazmasını, programları çalıştırmasını v.b. sağlayan sistem monitör yordamlarını içerir. Şemada gösterildiği gibi, ROM un başlangıç adresi E000H dir. Bu ROM, CS2 yüksek



1. AKSİ BELİRTİLMEDİKÇE: DİRENÇ DEĞERLERİ, OHM; KAPASİTANS DEĞERİ, MİKROFARAD CİNSİNDENDİR. DİRENÇLER 1/4 WATT, ± % 10 LUKTUR.
2. HER ÜÇ PAKET İÇİN KAPASİTÖR.

3. SEÇİMLİK

MC68708	MCM7641	MCM68317
E1 TO E2	E0 TO E9	E0 TO E4
E3 TO E9	E4 TO E7	E1 TO E2
E4 TO E8	E6 TO E7	E5 TO E9
E5 TO E8		

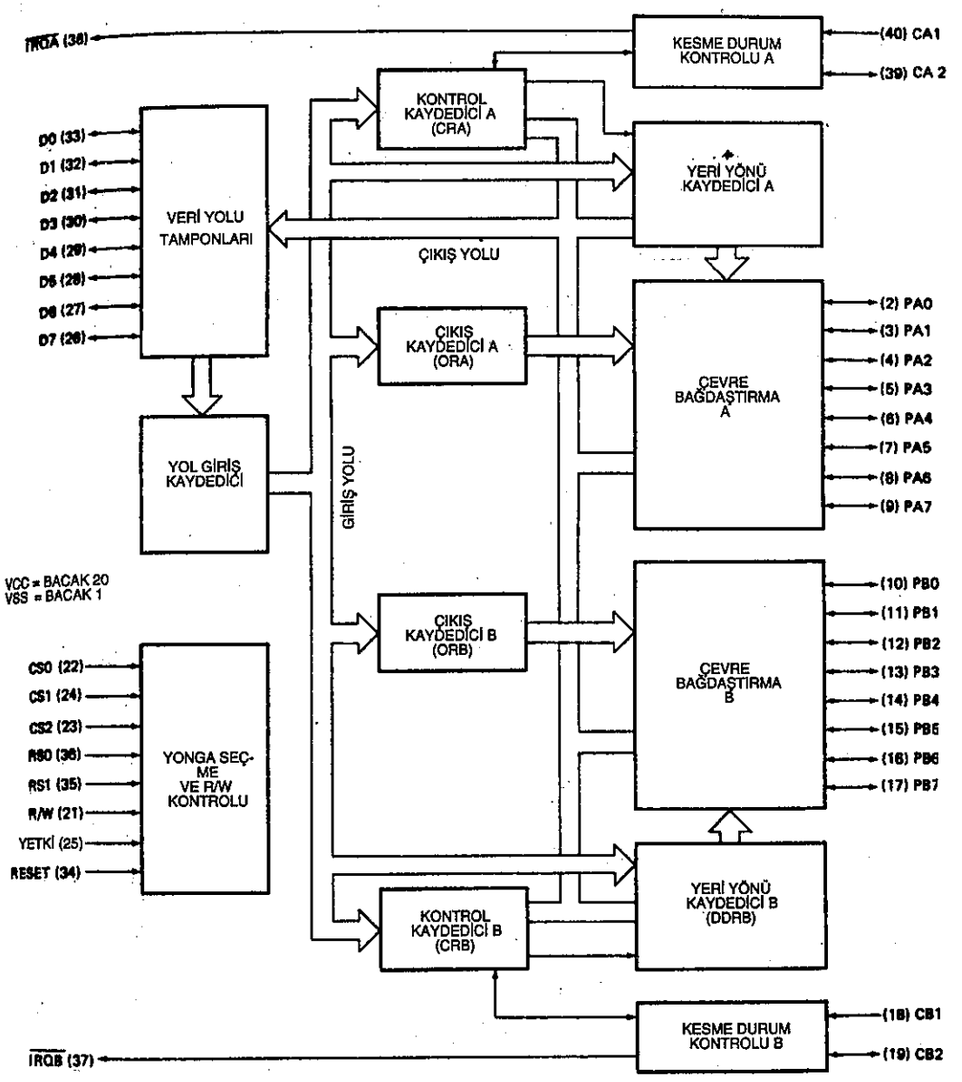
4. MC6879 OPTIONAL
5. MC6728 OPTIONAL
6. MC7406P OPTIONAL

Şekil 10-5 Basit bir MC6800 tabanlı sistemin şeması (Motorola)

ve CS1 alçak ise seçilir. CS1 sinyali, üç-hattan-sekiz hatta kod çözücü olarak kullanılan 74155 tarafından üretilmektedir. Diğer bir deyişle, eğer 74155, yükleme girişlerine geçerli bir bellek adres sinyali uygulayarak yetkili hale getirilmişse, *A*, *B* ve *D* girişlerince adreslenen bir çıkış alçak olacaktır. Bu, daha önce anlattığımız 8205'e benzemektedir. *A13*, *A14* ve *A15* adres bitleri MC8T96 tamponlarıyla ters çevrilmiş ve 74155'in *A*, *B* ve *D* girişlerine bağlanmıştır. Bu nedenle, *A15*, *A14* ve *A13* ün hepsi 1 ise ve VMA yüksek ise, 74155'in bacak 9 undaki Y0 çıkışı alçak olacak ve U8 ROM'unu seçecektir. *A15*, *A14* ve *A13* yüksek olduğunda bu E000 adresine karşılık gelir. Gerçekte *A12* nin kodu çözülmediğinden FXXX da bu ROM u seçer. Bu önemlidir, çünkü CPU, kesme ve yeniden başlama yordamlarının adreslerini almak için FFF8 den FFFF'e kadar olan adresleri gönderir. Diğer iki ROM, sırasıyla C000H ve 6000H adreslerinden başlamak üzere 74155'ten gelen $\overline{\text{PROM 0}}$ ve $\overline{\text{PROM 1}}$ tarafından seçilir.

RAM

RAM entegreleri de 74155'ten gelen çıkışlarla seçilmektedir. U13, monitör programı tarafından bir yaz-boz ve yığın olarak kullanılan bir MC6810 128X8 RAM dır. 74155 in *Y2* çıkışı, A000H adresinden başlayan bu RAM'ı seçer. Dört tane daha MC6810 (*U14*, *U16*, *U18* ve *U19*), 0000H adresinden başlayan 512 baytlık bir RAM oluşturur. 74155 kod çözücünün *Y7* çıkışı, *A15*=0 *A14*=0 ve *A13*=0 olduğunda bu dört RAM'a bir CS sinyali gönderir. *A12*, *A11* ve *A10*, RAM lara bağlı olmadıklarından, bunlar farketmezdir. *A7*, *A8* ve *A9* adres hatları dördünden hangisinin seçildiğini belirler. RAM, 00'dan FF'e kadar olan adres alanına konulmuştur, böylece doğrudan adresleme modu kullanılabilir. 74155'in iki çıkışı da harici ROM veya RAM veya I/O portlarını seçmek için kullanılabilir. $\overline{2/3}$ hattı 2000H ile 3FFFH arasındaki 8000-baytlık bloğu ve $\overline{4/15}$ da 4000H ile 5FFFH arasındaki 8000-baytlık bloğu seçer. I/O çıkışı, 8000H adresinden başlamak üzere I/O portlarını seçer.



Şekil 10-6 MC6821 PIA'nın blok diyagramı. (Motorola)

PARALEL I/O PORTLARI

Paralel I/O portları, bu devreye MC6821 çevre bağdaştırma adaptörleri ile konulmuştur. Bu aygıt Bölüm 9'da anlatılan 8255 ve 8155'e çok benzer. Kullanılmadan önce programda ilklendirilmesi gerekmektedir. 6821'in bir blok diyagramı Şekil 10-6'da gösterilmiştir. Aygıtın her yarısı, bir 8-bitlik giriş-çıkış portu ve iki kontrol hattı içerir. Her hattın yönü, veri yönü kaydedicisine yüklenen bir kelimeyle programlanır. Veri yönü kaydedicisinin bir bit konumundaki bir 1, karşılık gelen hattı çıkış olarak ve bir 0 da giriş olarak ayarlar.

İki veri yönü kaydedicisine ek olarak, *MC6821'in iki kontrol kaydedici, CRA* ve *CRB*, vardır. Bunların da programlanması gerekir. Kontrol kelimesi formatı Şekil 10-7a 'da gösterilmiştir. İlklendirme yordamını anlatmadan önce, 6821'in geri kalan bacaklarının işlevlerini açıklamamız gerekir.

CA1 ve *CB1, programlanabilir kesme girişleridir. CA2* ve *CB2, giriş veya çıkış kontrol sinyali olarak programlanabilir.* Yetkilendirildiğinde, *CA1* veya *CB1* girişindeki bir kesme, *CPU* nun $\overline{IRQ}$ veya $\overline{NMI}$ girişine bir $\overline{IRQA}$ veya $\overline{IRQB}$ gönderir.

Alçak yapıldığında, $\overline{RESET}$ girişi, bütün dahili kaydedici bitlerini 0'a reset eder, böylece bütün port hatları giriş olarak programlanır. Bu giriş genellikle bir güç açma veya elle resete bağlanır. Şekil 10-5 te elle resetteki çapraz-kupleli VE DEĞİL kapısı kontak zıplamasını gidericiye dikkat edin.

Yetkilendirme girişi *E*, sadece bir çevrimin uygun kısmında aygıtın aktif olmasını garanti eder. Bu giriş, DBE denilen bir gecikmeli *Ø2* saat sinyaline bağlanmıştır.

CS0, CS1 ve *CS2 elemanı seçmek üzere adres hatlarına* veya *kod çözücü çıkışlarına bağlanan yonga seçme girişleridir.* RS0 ve RS1 girişleri, sırasıyla *A0* ve *A1* adres hatlarına bağlanmıştır. Her MC6821 dört adres kullanır. Örneğin, Şekil 10-5'teki U20, 8004H ile 8007H arasındaki adresleri kullanmaktadır. Bu iki RS girişi, kontrol kaydedicisine yüklenen bir bitle birlikte, hangi dahili kaydedicinin adreslendiğini belirler. Şekil 10-7*b*, kontrol kaydedicinin bit 2 'si ve RS0 ile RS1 girişleri tarafından belirlenen dahili adreslemeyi göstermektedir.

Şimdi 6821'ın basit giriş ve çıkış portları olarak ilklendirilmesini inceleyeceğiz. Bölüm 11'de, 6821'in, *CA* ve *CB* bacaklarını kullanarak giriş ve çıkış el sıkışması için nasıl ilklendirileceğini açıklamaktayız.

Herhangi bir programlanabilir port aygıtını ilklendirmek için, önce aygıttaki her veri yönü kaydedici, kontrol kaydedici ve port kaydedici için sistem adresini belirleyin. Böylece kontrol kelimelerini hangi adrese göndereceğinizi öğrenmiş olursunuz. Bunu, Şekil 10-7b'de verilenlere benzer dahili adresleri, sistem adres çözümü ile belirlenen aygıtın taban adresine ekleyerek yapın. Şekil 10-5'teki *U20 PIA*'nın taban adresi 8004H dir. Şekil 10-7*c*, bu

6821'in her kaydedicisinin sistem adresini göstermektedir. Eğer kontrol kaydedici *A*'nın bit 2 si 0 ise, 8004H adresine gönderilen bir bayt, veri yönü kaydedici *A* (DDRA)'ya gider. Eğer bu bit 1 ise, bu takdirde 8004H adresine gönderilen bir bayt, port *A* kaydedicisine gider ve port *A*'dan çıkış verilir. Benzer şekilde, kontrol kaydedici *B* 'nin bit 2'si 0 ise, 8006H adresine gönderilen bir bayt, veri yönü kaydedici *B* (DDRB) ye gider. Eğer bu bit 1 ise, aynı adrese gönderilen bir bayt, çıkış portu *B* ye gider.

	7	6	5	4	3	2	1	0
CRA	IRQA1	IRQA2	CA2 kontrol			DDRA erişim	CA1 kontrol	
	7	6	5	4	3	2	1	0
CRB	IRQB1	IRQB2	CB2 kontrol			DDRB erişim	CB1 kontrol	

(a)

RS1	RS0	Kontrol Kaydedici Biti CRA2	Kontrol Kaydedici Biti CRB2	Seçilen Yer
0	0	1	X	Çevre Kaydedici A
0	0	0	X	Veri Yönü Kaydedici A
0	1	X	X	Kontrol Kaydedici A
1	0	X	1	Çevre Kaydedici B
1	0	X	0	Veri Yönü Kaydedici B
1	1	X	X	Kontrol Kaydedici B

X=Farketmez

(b)

8004 — DDR A, EĞER CRA BIT 2=0 İSE
8004 — PORT A, EĞER CRA BIT 2=1 İSE
8005 — KONTROL KAYDEDİCİ A
8006 — DDR B, EĞER CRB BIT 2=0 İSE
8006 — PORT B, EĞER CRB BIT 2=1 İSE
8007 — KONTROL KAYDEDİCİ B

(c)

Şekil 10-7 MC6821 PIA kontrol kelimesi ve dahili adresleme formatı. *(a)* Kontrol kelimesi formatı. *(b)*. Dahili adresleme formatı. *(c)* Şekil 10-5'teki U20 6821 için sistem adresleri.

Bir resetten sonra, veri yönü kaydedicilerinin ve kontrol kaydedicilerinin bütün bitleri 0 dır. Bu nedenle, bütün portlar girişe ayarlanmıştır. Bir port hattını çıkış olarak ilklendirmek için, veri yönü kaydedicinin o hatta karşılık gelen bitine bir 1 gönderilmelidir. Veri kontrol kelimesi, veri kontrol kaydedicisine gönderildikten sonra, bit 2'si 1 olan bir kontrol kelimesi, kontrol kaydedicisine gönderilmelidir. Port kaydedicisine erişim sağlamak için gönderilen bu kontrol kelimesinden sonra, veri porta gönderilebilir. Şekil 10-8, bir 6821'in port *A* sının tamamını çıkışa, port B'nin üst dört hattını girişe ve alt dört hattını çıkışa ilklendiren bir çeviri dili yordamı göstermektedir. Aygıtın taban adresi 8004H dir.

Veri yönü kaydedicilerine erişim sağlamak için CLR komutlarıyla, iki kontrol kaydedicisine önce 0 yüklenir. Bu sadece, bir reset yapılmamışsa gereklidir. Bir FFH baytı *A* akümülatörüne yüklenir ve 6821'in *A* veri yönü kaydedicisine gönderilir. Bu, port *A* 'nın bütün hatlarını çıkış yapar. Sonra bit 2'si 1 olan bir bayt (04H), *A* akümülatörüne yüklenir ve 6821 in *A* kontrol kaydedicisine gönderilir. Bu, erişimi DDRA'dan *port A*'ya değiştirir, böylece porta veri yazılabilir. Bu işlem *port B* için de tekrarlanır, ancak giriş hatlarına karşılık gelen veri yönü kaydedici bitlerine 0 gönderilir. LDAA $50 komutu, 0050H adresinden bir veri baytını kopyalar sonra, STAA $8004, bu veri baytını *port A* ya gönderir. LDAB $8006, *port B* den bir veri baytı okuyup *B* akümülatörüne koyar. Bir port hattı, çıkış olarak ayarlanmış bile olsa, hâlâ okunabilir.

Süreç şöyle özetlenebilir: İlk olarak, reset edin veya bit 2 si 0 olan kontrol kelimeleri gönderin. İkinci olarak, her çıkış hattı için ilgili biti 1 olan veri yönü kelimelerini, veri yönü kaydedicilerine gönderin. Üçüncü olarak, bit 2'si 1 olan yeni kontrol kelimeleri gönderin, böylece port kaydedicileri okunabilir veya yazılabilir.

```
CLR     $8005     ;  DDR'LERE ERİŞMEK İÇİN KONTROL
CLR     $8007     ;  KAYDEDİCİLERİN BİT 2 SİNİ SIFIRLA.
LDAA    #$0FF     ;  HEPSİNİN ÇIKIŞ OLMASI İÇİN VERİ YÖNÜ KELİMESİ,
STAA    $8004     ;  PORT A VERİ YÖNÜ KAYDEDİCİSİNE GÖNDER.
LDAA    #$04      ;  PORT A ERİŞİM KONTROL KELİMESİ,
STAA    $8005     ;  KONTROL A 'YA GÖNDER.
LDAA    #$0F      ;  ÜST 4 HAT GİRİŞ, ALT 4 HAT ÇIKIŞ İÇİN DDR
                  ;  KONTROL KELİMESİ,
STAA    $8006     ;  PORT B DDR' YE GÖNDER
LDAA    #$04      ;  PORT A ERİŞİM İÇİN KONTROL KELİMESİ,
STAA    $8007     ;  KONTROL B 'YE GÖNDER
LDAA    $50       ;  GÖNDERİLECEK VERİ BAYTINI AL
STAA    $8004     ;  PORT A 'YA GÖNDER.
LDAB    $8006     ;  PORT B 'Yİ AKÜMÜLATÖR B 'YE OKU.
```

Şekil 10-8 8004H adresindeki MC6821 PIA yı ilklendiren çeviri dili yordamı.

Şekil 10-5'teki devrenin anlatacağımız son kısmı, MC6850 asenkron iletişim bağdaştırma adaptörü, yani ACIA'dır. Bu eleman, Bölüm 9'da anlatılan 8251'in asenkron moduna benzer şekilde çalışan bir programlanabilir UART tır. Kullanılmadan önce ilklendirilmelidir.

Şekil 11-30, MC6850 ACIA nın bir blok diyagramını göstermektedir. Aygıt, istenilen baud oranının 1, 16 veya 64 katı frekansta bir harici saat gerektirmektedir. Şekil 10-5'teki sistem için, bir 4800 Hz lik saat kullanılmıştır. Bu, istenilen 300 baudluk oranın 16 katıdır.

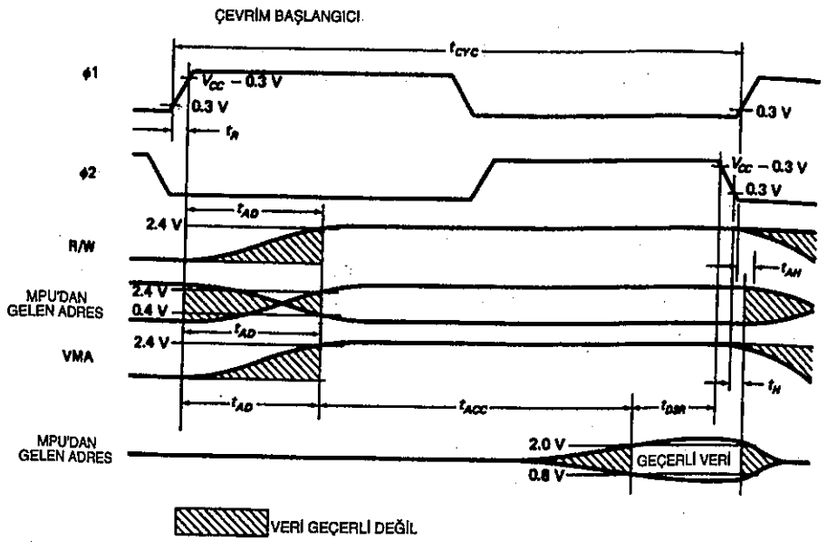
Üç yonga seçme girişi (CS0, CS1 ve CS2), elemanı seçer. Yetki girişi, Ø2 saatinin geciktirilmiş bir şekline bağlanmıştır. Bir kaydedici seçme girişi, veri kaydedicilerini veya *control/durum kaydedicilerini* seçmek için kullanılır. Bir girişteki bir yüksek, veri gönderme/alma kaydedicilerini seçer. Bu giriş, çoğunlukla *A0* adres hattına bağlanır. Şekil 10-5'teki sistemde, MC6850'nin adresleri, kontrol/durum kaydedicisi için 8008H ve veri port kaydedicisi için 8009H'dir.

6800 SİSTEM ZAMANLAMASI

KOMUT ÇEVRİMLERİ VE MAKİNA ÇEVRİMLERİ : Daha önce belirtildiği gibi, MC6800, bir iki-fazlı ve üstüste binmeyen bir saat gerektirmektedir. 6800 için, bir makina çevrimi, her zaman *Ø1*'in yükselen kenarından ölçülen bir saat periyoduna eşittir. Bir komut çevrimi, iki ile sekiz arasında makina çevriminden oluşur (Tablo 10-1). Makina çevrimi sayısı, saat çevrimi sayısına eşit olduğundan, makina çevrimi sayısı ile saat periyodunu çarparak, herhangi bir komutun yürütme süresini hesaplamak oldukça kolaydır.

OKUMA ÇEVRİMİ : 6800 'ün sadece üç temel tip makina çevrimi vardır: Bir okuma çevrimi, bir yazma çevrimi ve bir dahili çalışma çevrimi. Dahili çalışma çevrimi sırasında, yollar kullanılmazlar. Sinyaller yönünden, sadece okuma ve yazma çevrimleri bizi ilgilendirir. Şekil 10-9, bir MC6800 okuma çevrimi için basit zamanlama dalgaşekillerini ve değerlerini göstermektedir. CPU, *Ø1*' in yükselen kenarından t_{AD} kadar bir süre içinde, bir okuma sinyali, bir adres ve bir geçerli bellek adresi sinyali gönderir. Belleğin, geçerli veri ile cevap vermesi için t_{ACC} kadar süresi vardır ve veriyi t_{DSR} kadar sabit tutmalıdır. *Ø2* saat sinyalinin düşen kenarı işlemin bittiğini belirtir ve bir sonraki *Ø1*'in yükselen kenarı, bir sonraki çevrimi başlatır.

Eğer belleğin erişim zamanı, CPU tarafından müsade edilen t_{ACC} 'den daha uzunsa, varolan t_{ACC} yi artırmak üzere saatler uzatılabilir. Bu, MC6871 B nin $\overline{HOLD2}$ girişini alçak yapmakla gerçekleştirilebilir. HOLD2 yüksek yapılıncaya kadar, *Ø1* saati alçak ve *Ø2* saati yüksek tutulur. Saatler, maksimum 5 µs ye kadar uzatılabilir, çünkü bu süre içinde, dahili dinamik kaydediciler tazelenmemektedir. $\overline{HOLD2}$ giriş sinyali, *Ø1* yüksekten alçağa düşecek şekilde senkronize edilmelidir. MC6871'in $\overline{HOLD1}$ girişi, *Ø1* yüksek ve *Ø2* alçak olmak üzere saatleri uzatmak için kullanılır.



(a)

Okuma - Yazma Zamanlaması

Parametre	Sembol	Min	Tipik	Mak	Birim
Adres Gecikmesi	t_{AD}	—	220	300	ns
Çevre Okuma Erişim Süresi $t_{ACC} = t_{CYC} - (t_{AD} + t_{DSR})$	t_{ACC}	—	—	540	ns
Kurulu veri Süresi (Okuma)	t_{DSR}	100	—	—	ns
Giriş Verisi Tutma Süresi	t_H	10	—	—	ns
Çıkış Verisi Tutma Süresi	t_H	10	25	—	ns
Adres Tutma Süresi (Adres, R/W, VMA)	t_{AH}	50	75	—	ns

(b)

Şekil 10-9 MC6800 okuma çevrimi zamanlaması. (Motorola)

YAZMA ÇEVRİMİ: MC6800'ün yazma çevrimi, okuma çevrimine çok benzer. Ø1 yükseldikten sonra, t_{AD} kadar bir süre içinde CPU, bir yazma sinyali, adresler ve bir *VMA* sinyali gönderir. CPU, bir veri yolunu yetkilendirme (DBE) sinyali alınca, veriyi veri yoluna yükler. DBE, genellikle sadece bir geciktirilmiş Ø2 saatidir. Bu nedenle, veri, yüksek Ø2 'nin son kısmında geçerli hale gelecektir.

6800 ÜN KESMELERİ

Şekil 10-10, WAI ve SWI komutlarının yanısıra *6800* ün donanım girişleri $\overline{RESET}$ ($\overline{RESTART}$), $\overline{IRQ}$ ve $\overline{NMI}$ in etkilerini özetlemektedir. Burada önemli olan, bütün bunları ezberlemeniz değil, gerek duyduğunuzda 6800 ün cevabını belirlemenizde yardımcı olması için Şekil 10-10 'u nasıl kullanacağınızı öğrenmenizdir.

$\overline{\text{RESTART}}$

Şekil 10-10 'un sağ tarafında gösterildiği gibi, 6800 'ün $\overline{\text{RESET}}$ ($\overline{RESTART}$) girişindeki bir alçak, birimin FFFEH adresine gitmesine ve orada saklanmış olan yeniden başlama adresinin en büyük değerlikli baytını alıp getirmesine sebep olur. Bundan sonra 6800 program sayıcısını FFFFH a artırır ve orada saklanmış olan yeniden başlama adresinin en küçük değerlikli baytını alıp getirir. Kesme maskesini (durum kodu kaydedicinin bit 4 ü) set ettikten sonra, 6800, yeniden başlama adresini program sayıcısına yükler ve bir sonraki komutunu bu adresten getirir. 6800, yeniden başlama vektörünü (adresini) almak için, daima FFFEH ve FFFFH adreslerine gittiğinden, programcı, $\overline{\text{RESTART}}$ yordamının başlangıç adreslerinin burada saklanmış olduğunu garanti etmelidir.

$\overline{\text{NMI}}$

6800'ün *maskelenemeyen donanım kesmesi* ($\overline{NMI}$) girişindeki bir alçak, aygıtın, o anda yürütmekte olduğu komutu bitirmesine ve geri dönüş adresini, indis kaydedicisini, akümülatörleri ve durum kodu kaydedicisini yığına itmesine neden olur. Şekil 10-10'da $\overline{\text{RESTART}}$ ın yanındaki düşey okla gösterildiği gibi, 6800, hizmet yordamının başlangıç adresinin en büyük değerlikli baytını almak üzere FFFCH adresine gider. Aygıt bundan sonra da, hizmet yordamının başlangıç adresinin en küçük değerlikli baytını almak üzere FFFDH adresine gider. Diğer donanım kesmelerini yetkisiz kılmak için kesme maskesini set ettikten sonra, 6800, FFFCH ve FFFDH adreslerinden almış olduğu kesme vektörünü (adresini) program sayıcısına yükler. Aygıt, bir sonraki komutunu bu adresten getirir. Bu nedenle, programcı, $\overline{\text{NMI}}$ kesmesinin hız-

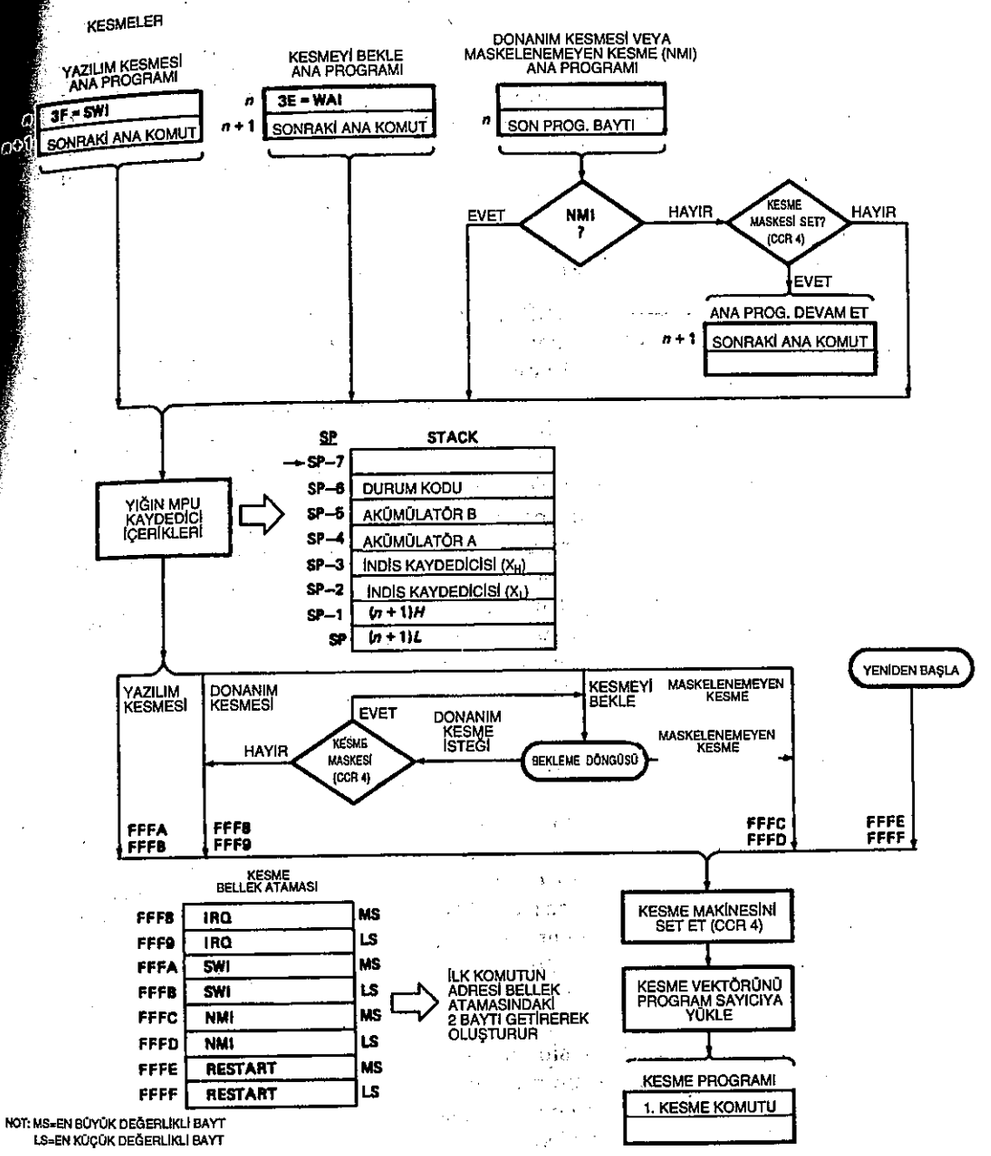
met yordamının başlangıç adresini FFFCH ve FFFDH 'ye yüklemelidir. Hizmet yordamının sonunda bir kesmeden geri dön komutu (RTI), bütün kaydedicileri yığından çeker ve yürütmeyi ana programdaki bir sonraki komuta verir. Eğer ana programda başka donanım kesmelerinin tanınması gerekiyorsa, RTI komutundan önce, bu kesmenin maskesini kaldırmak için bir CLI komutu kullanılmalıdır.

INTERRUPT REQUEST

6800'ün kesme isteği INTERRUPT REQUEST ($\overline{ZRQ}$) girişindeki bir alçak sinyalin, kesme maskesi (durum kodu yazmacının bit 4 ü) bir CLI komutuyla temizlenmemişse, hiçbir etkisi olmayacaktır. Eğer kesme maskesi temizlenmemişse, bir $\overline{IRQ}$ girişi, 6800 işlemcinin o anda yürütmekte olduğu komutu bitirdikten sonra, geri dönüş adresini, indis kaydedicisini, akümülatörleri ve durum kodu kaydedicisini yığına itmesine neden olur. Aygıt, bundan sonra, kesme vektörünün (adresinin) en büyük değerlikli baytını almak için FFF8H adresine ve en küçük değerlikli baytını almak için FFF9H adresine gider. Daha sonra, 6800, $\overline{IRQ}$ sinyalinin hemen bir başka kesmeye sebep olmaması için kesme maskesini set eder. Bunun arkasından, aygıt, FFF8H ve FFF9H den aldığı hizmet yordamının başlangıç adresini (kesme vektörü) program sayıcısına yükler ve bir sonraki komutunu bu adresten getirir. Hizmet yordamının sonunda, bir CLI komutu, kesme maskesini kaldırır. Bir RTI komutu da, bütün kaydedicileri yığından çeker ve yürütmeyi ana programa geri verir.

YAZILIM KESMESİ (SOFTWARE INTERRUPT: SWI)

6800 bu komutu yürüttüğünde, geri dönüş adresini, indis kaydedicisini, akümülatörleri ve durum kodu kaydedicisini yığına iter. Sonra, kesme vektörünün en büyük değerlikli baytını almak için FFFAH ve en küçük değerlikli baytı için FFFBH adreslerine gider. Daha sonra, 6800, kesme maskesini set eder ve kesme hizmet yordamının başlangıç adresini program sayıcıya yükler. İşlemci, hizmet yordamını yürütür. Bazı 6800 sistem monitör programları bu komutun bir program durma noktası olarak kullanılmasına imkân verir. Monitör programı, 6800, sizin programınızda bir SWI komutu yürüttüğünde, aygıtın sizin programınızı kesip, bütün kaydedicileri ve program sayıcıyı yığına saklamasını ve kontrolu monitör programına vermesini sağlayacak şekilde yazılmıştır. Böylece, monitör programının kaydedicileri inceleme işlevini kullanarak, o noktada kaydedicilerin, v.b. lerinin içeriklerinin doğru olup olmadığını kontrol edebilirsiniz. Eğer kaydedicilerin v.b. lerinin içerikleri doğruysa, SWI durma noktasını, programınızda daha ileri bir noktaya kaydırıp, programı tekrar çalıştırarak, bu noktada kaydedicilerin v.b. lerinin durumlarını inceleyebilirsiniz.



Şekil 10-10 MC6800'ünRESTART, NMI, IRQ girişlerinin ve SWI ve WAI komutlarının etkileri.

KESME İÇİN BEKLE (WAIT FOR INTERRUPT: WAI)

6800, kesme için bekle komutu WAI yi yürüttüğünde, geri dönüş adresini, indis kaydedicisini, akümülatörleri ve durum kodu kaydedicisini yığına iter. Aygıt bundan sonra, Şekil 10-10'un ortasında gösterildiği gibi bir kesmeyi bekleme döngüsüne girer. Bu döngüden çıkış sadece bir donanım $\overline{NMI}$ giriş sinyali ile veya bir donanım $\overline{IRQ}$ giriş sinyali ile olabilir. Eğer bir $\overline{NMI}$ girişi gelirse, işlemci FFFCH ve FFFDH den $\overline{NMI}$ hizmet yordamının başlangıç adresini alır, kesmeyi maskeler ve $\overline{NMI}$ hizmet yordamına gider. Eğer bir $\overline{IRQ}$ girişi gelirse, daha önceden bir CLI komutuyla kesme maskesi kaldırılmamışsa, 6800, kesmeyi bekleme döngüsünden çıkamaz.

BİR 6800 TABANLI MİKROBİLGİSAYAR SİSTEMİNDE ARIZA BULMA

Daha önce doğru olarak çalışmış bozuk bir 6800 tabanlı mikrobilgisayar sistemi için, aşağıdaki basamaklar dizisi arızayı bulmada yardımcı olacaktır.

1. Bir osiloskopla, bütün güç kaynaklarının gerilimlerinin doğru olduğunu ve aşırı dalgacık veya gürültüleri bulunmadığını kontrol edin. Aşırı güç kaynağı dalgacıkları, bir mikrobilgisayar sistemine çok garip şeyler yaptırabilir. Sistem bazen çalışıp, bazen çalışmayabilir.
2. *CPU* çevresinde bir sinyal yoklaması yapın.

   a. CPU 'nun nabzı olan *Ø1* ve *Ø2* saatleri var mı ve doğru mu?

   b. DBE sinyali var mı? Bu sinyal *Ø2* 'yi geciktirerek elde edildiğinden, osiloskopta bakıldığında *Ø2* 'ye benzemelidir.

   c. $\overline{RESET}$ yüksek olmalıdır.

   d. $\overline{NMI}$ ve $\overline{IRQ}$ yüksek olmalıdır.

   e. $\overline{HALT}$ yüksek olmalıdır.

   f. TSC alçak olmalıdır.

   g. VMA darbeler göstermelidir.

   h. Bir osiloskopla, veri yolunda ve adres taşıtında bir hareketlilik olduğunu kontrol edin. Tam olarak ne olup bittiğini söylemeyebilirsiniz; fakat adres hatlarındaki darbeler en azından işlemcinin adresleri gönderdiğini göstermektedir.

   i. Eğer saat sinyalleri yoksa, saat üreteci entegresinden şüphelenin. Saatler varsa, fakat DBE yoksa, bu takdirde *Ø2* den DBE yi üretmek için kullanılan kapıları kontrol edin. Eğer RESET girişinin zıplamasını gideren kapılar bozuksa, $\overline{RESET}$ alçakta takılıp kalabilir. Bu, 6800 'ü reset durumunda tutacaktır ve hiçbir işlem yapılmayacaktır. $\overline{NMI}$ sinyalini sağlayan kapıların arızalanması da $\overline{NMI}$'ı alçağa çekmişse, 6800'ün sürekli kesmede kalmasına sebep olabilir.
3. Eğer 1. ve 2. basamaklar problemi açığa çıkarmıyorsa, RESET düğmesini bir bantla yapıştırarak basılmış tutun. Bu, 6800 ün $\overline{RESET}$ girişine sürekli bir alçak verir ve CPU 'yu bir reset durumunun başında tutar. Bu durumda adres taşıtı FFFEH içerecektir, BA alçak olmalıdır, VMA alçak olmalıdır ve R/$\overline{W}$ yüksek olmalıdır. Bu sinyaller bir osiloskopla kontrol edilebilir. Çıkış tamponlarının girişlerindeki ve çıkışlarındaki sinyallerde kontrol edilebilir. RESET düğmesini bırakın ve yeniden başlama yordamını içeren ROM'un $\overline{CS}$ girişine osiloskopun probunu bağlayın. RESET düğmesi tekrar tekrar basıldığında, bu $\overline{CS}$ girişinde darbeler görülmelidir. Burada darbeler görülmüyorsa, adres çözücüden şüphelenin. Eğer darbeler varsa fakat monitör programı yürütülmüyorsa, monitör ROM'u veya CPU'dan şüphelenin.
4. Eğer monitör programı doğru olarak yürütülüyorsa, RAM'a veri yazmak ve geri okuyup doğruluğunu karşılaştırmak üzere kısa yordamlar kullanabilirsiniz. Ayrıca tekrar tekrar bir porta yazmak veya porttan okumak için test yordamları da kullanılabilir. Böylece, sinyallere standart bir osiloskopla bakabilirsiniz.
5. Daha sonra problemler için, işlemci çalışırken, adres, veri ve kontrol yollarındaki sinyal akışını izlemek üzere bir mantık analizörü kullanılabilir.

MC6800 ÜN SOYUNDAN GELENLER

Bu kısımda, Intel'in 8080A'dan 8085A ve 8048 ailesine olan gelişmesine benzer şekilde Motorola'nın MC6800'ünün gelişmesini kısaca anlatacağız.

MC6802

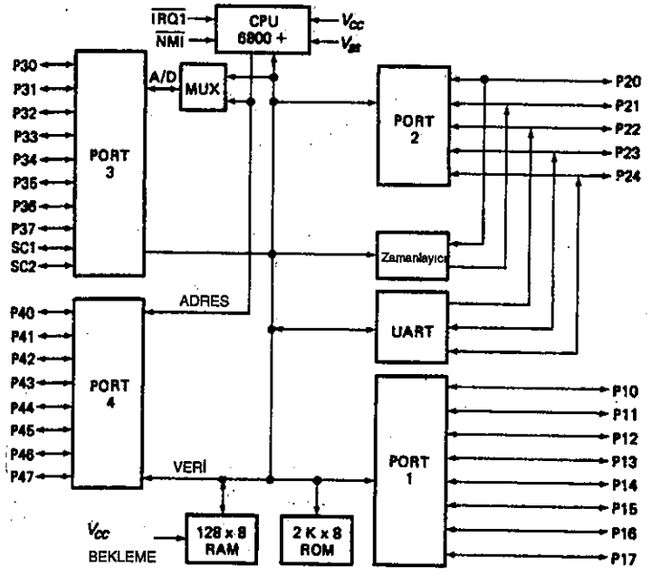
MC6800' ün gelişmesindeki ilk adım MC6802 dir. Yonga MC6800 ile aynı komut kümesini kullanır, fakat entegre bir saatle 128 baytlık RAM içermektedir. RAM adresleri 0000 ile 007F arasındadır, böylece burada saklanan veriler için doğrudan adresleme kullanılabilir. Bu RAM kolayca yaz-boz kaydedicileri olarak kullanılabileceğinden, 6800'ün kaydedici kıtlığı problemi çözülmüş olmaktadır.

MC6802 ile beraber giden bir entegre MC6846'dır. Bu entegre, 2K'lık ROM, 10 I/O hattı ve bir 16-bitlik zamanlayıcı içermektedir. İki aygıt birlikte küçük bir sistem oluşturmaktadır.

MC6801

MC6801, tek bir entegreye daha da fazla fonksiyon sıkıştırmaktadır. Bu entegre, 2K 'lık maske programlanabilir ROM, 128 baytlık RAM, 31 I/O hattı, bir 16-bitlik zamanlayıcı ve bir tam UART içerir. Şekil 10-11, MC6801'in bir işlevsel blok diyagramını göstermektedir.

Entegrenin, I/O hatlarının nasıl kullanılacağı ile seçim yapmak üzere üç çalışma modu vardır. Dahili RAM ve ROM 'un yeterli olduğu uygulamalarda, bütün hatlar I/O için kullanılabilir. Buna tek-yonga modu denir. Diğer bir mod, genişletilmiş çoğullamasız mod olup, port 3'ü veri yolu ve port 4 'ü alt 8 adres biti olarak kullanır. Bu mod 6801 in çevre entegrelerle iletişimde bulunmasına veya 256 tane daha bellek yerine imkan verir.



Şekil 10-11 MC6801 'in işlevsel blok diyagramı. (Motorola)

Üçüncü mod, genişletilmiş çoğullamalı mod olup, port 4 'ü üst adres bitleri için kullanır. Alt adres bitleri ve veri yolu port 3'te çoğullanır. Port 1 ve port 2 hâlâ I/O için kullanılabilir. Bu mod MC6801 65K bellek ve çevre birimleri kullanmasına imkan verir.

6800'ün komut kümesine ek olarak, MC6801'in 16-bitlik toplama, 16-bitlik çıkarma ve bir 16-bitlik sonuç vermek üzere 8-bit X 8-bit 'lik çarpma komutları vardır.

MC6809

MC6809 bir yarı 16-bitlik işlemcidir. 6800'ün bütün komutlarını ve 8-bitlik A ve B akümülatörlerini tek bir 16-bitlik D kaydedici olarak kullanan ek bir komut grubunu yürütmektedir. Bu, 16-bitlik aritmetik işlemlerin daha kolayca gerçekleştirilmesine imkân verir. MC6809 un iki 16-bitlik indis kaydedici, iki 16-bitlik yığın göstericisi, bir 16-bitlik program sayıcısı, bir 8-bitlik durum kodu kaydedicisi ve bir doğrudan sayfa kaydedicisi vardır.

MC6809'daki bir ana gelişme adresleme modlarının sayılarındaki artıştır. Doğrudan sayfa kaydedici, adresi kaydedicide içerilen sayfa içinde doğrudan adresleme kullanılmasına imkan verir.

Ayrıca MC6809'a ilave edilmiş bulunan birçok dolaylı adresleme türü vardır. Dolaylı adresleme, bellekten bir sonraki komut yerine bir sonraki komutun adresini getirir. Dolaylı adreslemenin bir örneği, 6800'ün kesme vektörü adresleridir. Bir NMI sinyali gelmesi durumunda, CPU, kesme hizmet yordamının başlangıç adresini almak üzere FFFC ve FFFD adreslerine gider. Diğer bir deyişle, yürütülecek bir sonraki komutun etkin adresi FFFC veya FFFD'de içerilmektedir. Buna mutlak dolaylı adresleme denir, çünkü NMI yordam adresini almak için CPU daima mutlak adreslere gitmektedir.

Genişletilmiş dolaylı adresleme, etkin adresin saklandığı bellek yerini göstermek üzere komutun ikinci ve üçüncü baytlarını kullanır. CPU etkin adresi getirir ve sonra bir sonraki komutu getirmek üzere bunu adres taşıtı üzerine koyar.

İndislenmiş dolaylı adresleme, indis kaydedicisine komutun ikinci baytında içerilen kaydırmayı ekler. Bu sonuç, etkin adresin saklandığı bellek yerini gösterir.

Göreceli dolaylı adresleme, komutun ikinci ve üçüncü baytlarında içerilen kaydırmayı program sayıcıya ekler. Sonuç, etkin adresin saklandığı bellek yerini gösterir. MC6809'un standart göreceli adreslemesi, 6800'ün 1-baytlık kaydırmasına karşın, 2-baytlık kaydırma kabul edebilir.

Bu adresleme modları, özellikle büyük veri tablolarıyla uğraşırken faydalıdır. Bu kitapta bunların uygulamalarını göstermeye yer yoktur. Ancak bunlarla ilgili kavramları anlatmakta yarar vardır.

MOS TECHNOLOGY 6500 MİKROİŞLEMCİ AİLESİ

Motorola tarafından MC6800 tanıtımından hemen sonra, MOS Technology, Inc., MC6800 geliştirilmişi olarak 6500 ailesini tanıtmıştır. 6500 ailesi daha sonra Synertek ve Rockwell International tarafından da üretilmeye başlanmıştır. Ailenin halen paket büyüklüğü, adresleme kapasitesi ve dahili saat mantığının varlığı veya yokluğu yönünden birbirinden farklı 10 çeşidi bulunmaktadır. Burada, bu ailenin çok yüksek bir satış miktarına ulaşan bir üyesi olan 6502 anlatılacaktır. 6502, kavram olarak MC6800'e benzer şekilde tasarlanmış olmasına rağmen, Z80A'nın 8080A/8085A ile uyuştuğu gibi, MC6800 ile uyuşmamaktadır. 6502, Apple II bilgisayarlarında kullanılan CPU dur.

MİMARİ YAPI VE KONTROL SİNYALLERİ

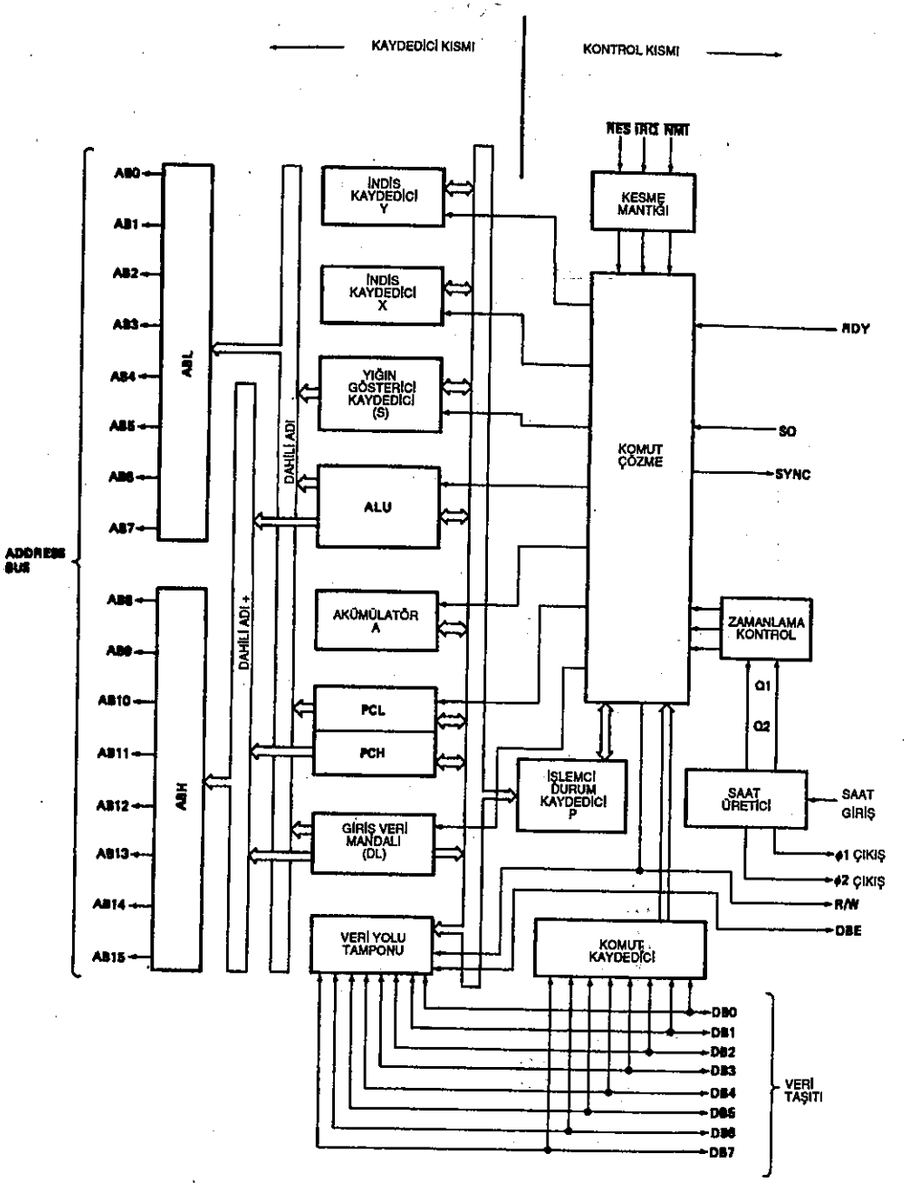
MİMARİ YAPI: Şekil 10-12, 6502'nin mimari yapısını göstermektedir. Bir dahili saat üreteci, tek bir TTL-seviyesi saat girişinden gerekli iki-fazlı saatleri üretir. Bu nedenle, 6871B gibi bir harici saat üreteci gerekmemektedir.

6502 nin sadece bir tane 8-bitlik akümülatörü vardır. Aygıtın bir 8-bitlik yığın gösterici kaydedicisi, iki 8-bitlik indis kaydedicisi X ve Y ve bir 8-bitlik durum kaydedicisi bulunmaktadır. Adres ve veri yolları, bir TTL yükü sürmek üzere tamponlanmışlardır.

KONTROL SİNYALLERİ: 6502'nin popüler olmasının bir sebebi kontrol sinyallerinin basitliğidir. Basit sistemler için bu kontrol sinyalleri yeterli olmaktadır.

Bir okuma-yazma çıkışı, işlemcinin bir bellek okuma veya yazma yaptığını belirtir. 6800 gibi, 6502' nin de doğrudan I/O komutları yoktur, bu nedenle bütün I/O lar bellek-haritalanmıştır.

$\overline{\text{RESET}}$, $\overline{\text{IRQ}}$ ve $\overline{\text{NMI}}$ girişlerinin çalışması 6800'dekilerle aynıdır. Eğer $\overline{\text{RESET}}$ sekiz veya daha fazla çevrim alçak tutulursa, istenilen yeniden başlama yordamının bellekteki başlangıç adresini almak üzere CPU, FFFC ve FFFD adreslerine gider. $\overline{\text{NMI}}$, maskelenmeyen kesme girişidir. $\overline{\text{NMI}}$ alçak yapıldığında, işlemci o anda yürütmekte olduğu komutu tamamlar ve sonra program sayıcı ile durum kaydedicisinin içeriklerini yığına iter. Bundan sonra, FFFA ve FFFB adreslerinden hizmet yordamının başlangıç adreslerini alıp getirir. $\overline{\text{IRQ}}$ girişi bir maskelenebilir kesmedir. Eğer bu kesme yetkilendirilirse ve $\overline{\text{IRQ}}$ girişi alçak tutulursa, CPU program sayıcı ve durum kaydedicisini yığına iter. Sonra, hizmet yordamının başlangıç adresini almak üzere FFFE ve FFFF adreslerine dallanır.



Şekil 10-12 MOS tekniği ile yapılmış MCS6502 'nin iç mimarisi

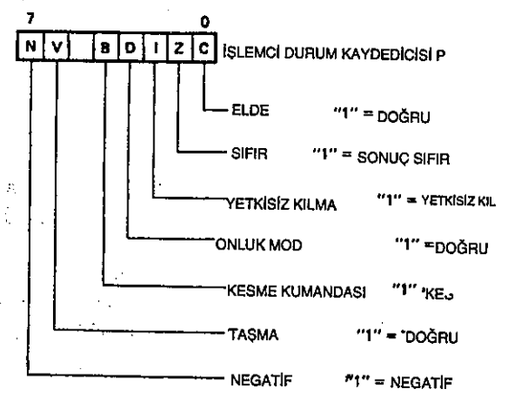
6502'nin saati uzatma yeteneği ve durma durumu yoktur. Gerçekte, yolların kayan hale getirildiği hiçbir durum yoktur. Eğer DMA işlemleri gerekiyorsa harici üç-durumlu tamponlar kullanılmalıdır. Ancak, 6502, 8080A dakine benzer bir manevrayla, bekleme durumları ekleyebilir. 6502 'nin hazır girişi (RDY), bir okuma çevrimi sırasında alçağa çekilirse, CPU bir bekleme durumuna girer. Eğer bir yazma çevrimi yürütülmekte ise, bu tamamlanır ve işlemci bekleme durumuna girmeden önce bir okuma çevrimine girilir. İstenildiği kadar çok bekleme durumu eklenebilir. Bekleme durumları, Bölüm 9 'da anlatıldığı gibi, yavaş bellekler için süreleri uzatmakta veya işlemciyi tek-adımlatmak için kullanılabilir.

SYNC çıkışı, CPU bir komut getirme işlemi yaparken pozitif bir darbe gönderir. Set-taşma girişi (SO), alçak yapıldığında taşma bayrağını set eder.

6502 NİN BAYRAKLARI

Şekil 10-13, *6502 nin durum kaydedicisinin formatını göstermektedir.* Elde, sıfır taşma ve negatif (işaret) bayrakları MC6800'dekilerle aynıdır. Diğer üçünü ise anlatmamız gerekir.

Eğer kesmeyi yetkisiz kılma bayrağı *I* yüksekse, $\overline{IRQ}$ kesme girişi maskelenir. Bu bayrak program komutlarıyla set veya reset edilebilir. Bu bayrak ayrıca bir reset veya bir kesme cevabı sırasında CPU tarafından da set edilir ve işlemcinin bir $\overline{IRQ}$ girişine cevap verebilmesi için temizlenmesi gerekir.



Şekil 10-13 MC6502'nin durum kaydedici formatı

Onluk-modu bayrağı *D*, eğer set edilmişse, her toplama veya çıkarma işleminin sonucu üzerinde otomatik olarak onluğa ayarlama yapar. Bu, sonucu, doğru BCD şekline sokar. Onluk-modunu set et komutu SED, bu bayrağı set eder ve "onluk-modunu temizle" komutu CLD bu bayrağı reset eder.

Kesme-kumanda bayrağı *B*, 6502 yazılım kesme komutunun (BRK) yürütülmesiyle set olur. Bu bayrak, işlemci tarafından kesmenin yazılımdan mı yoksa donanımdan mı geldiğini belirlemek için kullanılır. Kesme-kumanda bayrağını doğrudan set veya reset eden komutlar yoktur.

KOMUT KÜMESİ VE ADRESLEME MODLARI

Tablo 10-2, 6502'nin 56 komutunu adresleme modları, bayt sayısı, saat çevrimi sayısı ve op kodlarıyla birlikte göstermektedir. Komutların çoğunun işlevi, sembolik gösterilişinden oldukça açıktır. Tablonun altındaki anahtarda, mantıksal VE, VEYA ve özel VEYA için kullanılan sembollere dikkat edin. "Etkin adresteki bellek" etkin adresteki bellek yerinin içeriği demektir ve MS, yığın gösterici tarafından belirtilen bellek yerinin içeriği demektir.

Tablo 10-2 de, çarpıcı bir nokta, 6502'nin 13 adresleme modunun bulunmasıdır. Ancak bunlardan bazıları sadece bir veya iki komut için kullanılmaktadır. Modların çoğu ile MC6800 ve MC6809'u anlatırken karşılaşmıştınız, fakat burada 6502 ile ilgili olarak tekrar kısaca anlatacağız.

DERHAL ADRESLEME: İşlenen, komutun ikinci baytında içerilir. Örneğin, AND $7E, akümülatörleri 7EH sayısıyla mantıksal olarak VE ler ve sonucu akümülatörde bırakır.

MUTLAK ADRESLEME: Bu, 6800'ün genişletilmiş adreslemesine karşılık gelir. Bu modda, komutun 2. ve 3. baytları, işlenenin saklandığı 16-bitlik bellek adresini içerir. Örneğin, AND 3F08 komutu, akümülatörün içeriğini, 3F08H bellek yerinin içeriği ile mantıksal olarak VE ler. Gerçekte komut bellekte alt mertebeli adres baytı önce olmak üzere 2D, 08, 3F olarak saklanmaktadır.

SIFIR-SAYFASI ADRESLEME: Bu, 6800'ün doğrudan adresleme moduna karşılık gelir. Buna bazen taban-sayfası adresleme de denir. 6502, belleği herbiri 256 baytlık sayfalara böler. Bir adresin üst 8 biti sayfa numarasını, alt 8 biti de sayfa içindeki adresi veriyor diye düşünülebilir. Sıfır-sayfası, 00H den FFH ye kadar olan ilk 256 bellek yerini belirtir. Sıfır-sayfası adresleme de, işlenenin 8-bitlik adresi, komutun ikinci baytında içerilir. Örneğin, AND 3A komutu, akümülatörün içeriği ile 003A bellek yerinin içeriğini mantıksal VE ler.

AKÜMÜLATÖR ADRESLEME: Bu, akümülatörün işleneni içerdiği "anlaşılır adreslemenin" özel bir durumudur. Örneğin, *ROL, akümülatörü elde üzerinden sola döndürür.*

ANLAŞILIR ADRESLEME: Bu adreslemede, işlenenin kaynağı komutun op kodu içinde bulunmaktadır. Örneğin, INX komutu, X indis yazmacını 1 artırır.

GÖRECELİ ADRESLEME: Bu, 6800'de anlatılanla aynıdır. Komutun ikinci baytında içerilen bir 2'ye tümleyen kaydırma miktarı program sayıcıya eklenir. Bu modu sadece dallanma komutları kullanmaktadır. Dallanma, bu komutun saklandığı adresten -126 dan +129'a kadar olabilir. Örneğin, BEQ 09H komutu 0072 adresinde bulunuyorsa, yürütüldüğünde, 007D adresine bir dallanma yapar.

MUTLAK X VE MUTLAK Y: Bunlar indislenmiş adresleme modlarıdır. İşlenen, X veya Y indis kaydedicinin içeriğini, ikinci ve üçüncü baytlarda içerilen sayı ile toplayarak hesaplanan adresten getirilir. Eklenen sayı 0 ile 255 (0 ile FFH) arasında olabilir. Eğer X indis kaydedicisi 93H içeriyorsa, AND 00F2, X komutu, işleneni 0185H adresinden alıp getirecektir. Eğer, indis kaydedicisinin içerilen adresle toplanması alt 8 bitten bir elde üretirse, bu elde, program sayıcıda doğru 16-bitlik adresi vermek üzere üst 8 bite eklenecektir . Buna, sayfa sınırını geçme (page boundary crossing) denir.

SIFIR-SAYFASI X VE SIFIR SAYFASI Y: Bunlar da indislenmiş adresleme modlarıdır. Komutun ikinci baytındaki bir 8-bitlik sıfır-sayfası adresi, etkin adresi elde etmek üzere indis kaydedicisiyle toplanır. Örneğin, X kaydedicisi 33H içeriyorsa, AND 0A, X komutu, bellek yeri 3D'nin içeriğini akümülatörle VE leyecektir. Sonuç akümülatörde bırakılacaktır. Bu modu kullanırken dikkatli olmalıdır, çünkü sıfır-sayfası adresle indisin toplamı FFH dan büyükse bir elde üretilmemektedir. Eğer FFH aşılırsa, program sıfır-sayfası kısmını esas alacaktır. Buna sayfa sınırı hatası (page boundary error) denir.

DOLAYLI ADRESLEME: Bu mod 6809 için anlatılmıştır. Bu modda, komutun son iki baytı etkin adresin adresini içerir. Örneğin, JMP FF08 komutu, yürütülmesi gereken bir sonraki komutun saklandığı adresi getirmek üzere program sayıcıyı FF08 ve FF09 adresine gönderir.

İNDİSLENMİŞ DOLAYLI ADRESLEME (IND, X) : Komutun ikinci baytındaki bir sıfır-sayfası adresi, X indis kaydedicisine eldesiz eklenir. Toplam, bir sıfır sayfası adresidir. CPU, etkin adresi elde etmek üzere, bu ve bir sonraki sıfır-sayfası adresine gider. Örneğin, X kaydedicisi 33 H içeriyorsa, AND (IND 0A, X) komutu, akümülatörle VE lenecek baytın adresini almak üzere 3DH ve 3EH adreslerine gidecektir.

DOLAYLI İNDİSLENMİŞ ADRESLEME (IND, Y) : Bu modda, komutun ikinci baytı bir sıfır-sayfası adres içerir. Bu ve bir sonraki sıfır-sayfası adresden, CPU, saklanmış bir adresi alıp getirir. Etkin adresi elde etmek için, Y indis kaydedicisinin içeriği, bu getirilen adrese eklenir. Buna bazen sondan-indislenmiş adresleme de denir, çünkü dolaylı adres getirilmesinden sonra Y indisi

eklenmektedir. IND,X için anlatılan süreç önden-indislenmiştir, çünkü dolaylı adresi getirmeden önce indis toplanmıştır.

Bütün bu adresleme modlarının uygulama örnekleri için, MOS Technology MCS6500 Programming Manual isimli kitaba bakın.

BASİT BİR 6502 SİSTEMİ

Bu kitapta anlattığımız diğer mikroişlemcilerde olduğu gibi, 6502'de bir sistem oluşturmak üzere kendisine bağlanabilecek bir çevre entegreleri ailesine sahiptir.

6522, daha önce anlatılan MC6821 PIA ya çok benzeyen bir çevre bağdaştırma adaptörüdür. Aygıtın, iki 8-bitlik I/O portu, dört kontrol hattı ve iki programlanabilir zamanlayıcısı vardır.

Diğer bir faydalı çevre entegresi 6530'dur. Bu aygıt, 1K baytlık ROM, 64 baytlık RAM, bir programlanabilir aralık zamanlayıcı ve iki I/O portu içerir. Bir sistem kurmak için standart RAM ve ROM lar da 6502 ile kullanılabilir.

6502 ÇEVRİMLERİ, ZAMANLAMASI VE BORUHATTI

6800'de olduğu gibi, 6502 'nin bir makina çevrimi, Ø1 saatinin bir periyoduna eşittir. 6502A, 2 MHz e kadar olan giriş saatiyle çalışır. Bu, 500 ns'lik bir makina çevrim süresi verir. Tablo 10-2 'de, her komut için verilen makina çevrimi sayısını kullanarak, komutların yürütme sürelerini hesaplayabilirsiniz.

Burada önemli bir nokta, bir işlemcinin hızını sadece giriş saat frekansına bakarak belirleyemeyeceğinizdir. Şekil 10-14, bu kitapta anlatılan dört işlemcinin halen varolan en hızlı tiplerinin denk komutlarının yürütme sürelerini karşılaştırmaktadır. 2-MHz lik giriş saati olan R6502A ve MC68B00, komutu 4-MHz'lik bir Z80A veya 5-MHz'lik bir 8085A dan daha çabuk yürütmektedir.

Saat girişi	İşlemci	OP Kodu	İşlenen	Çevrimler	Süre
2MHz	R6502A	ADC	Derhal	2	1 µs
2MHz	68B00	ADCA	Derhal	2	1 µs
5MHz	8085A	ACI	Derhal	7	1,4 µs
4MHz	Z80A	ADC	A, Derhal	7	1,75 µs

Şekil 10-14 Dört işlemcinin yürütme zamanlarının karşılaştırılması

R6502A ve MC68B00, boruhattı (pipelining) denilen bir teknikten büyük ölçüde yararlandıkları için daha düşük saat frekanslarında daha kısa komut çevrim sürelerine sahip olabilmektedirler. Boruhattı, son getirilen baytı açıklarken, bir sonraki baytı bellekten getirmek demektir. Bunun nasıl yapıldığını bir örnekle gösterelim. Tablo 10-2 'ye göre, sıfır-sayfası adresleme ile bir ADC komutunu yürütmek üç saat çevrimi alır. Şekil 10-15, bu yürütme sırasındaki olayları ve zamanlamaları göstermektedir. Tek başına bakıldığında, komut dört adım gerektirmektedir. Ancak, birinci ve dördüncü basamaklar komşu komutlarla üstüste binmekte olduğundan, komutu yürütmek için etkin olarak sadece üç basamak gerekmektedir.

Çevrim	İşlemci Hareketi
1	Önceki komutu bitir, OP kodu getir
2	OP kodu yorumla, sıfır-sayfası etkin adresi getir
3	Satır-sayfası etkin adresten sayıyı getir
4	Getirilen sayıyı akümülatöre ekle, sonraki OP kodu getir

Şekil 10-15 Sıfır-sayfa adreslemeli ADC komutu için mikroişlemci işlemleri.

Adil olmak için, bu noktada, bir işlemcinin bir programı yürütmedeki göreceli hızının, tek bir komutun yürütme süresine bakmakla doğru olarak belirlenemeyeceğini söylemeliyiz. Mikroişlemcilerin genel hızlarını, "değerlendirme programları" veya "denek taşı programları" (benchmark programs) denilen test programları karşılaştırır.

Başarı, büyük ölçüde kullanılan denektaşı programına bağlıdır. Örneğin, parite kontrolunu kullanan bir denektaşı, büyük bir ihtimalle bir 8085A veya Z80A da daha hızlı çalışacaktır, çünkü bunların, 6502 ve 6800 de olmayan, parite bayrakları vardır. Ancak, bir bellek tamponundaki veri üzerinde çalışmayı ve veriyi tekrar tampona koymayı içeren bir denektaşı, 6800 ve 6502 de daha hızlı olabilir, çünkü bu işlemcilerde doğrudan ve indislenmiş adresleme mümkündür.

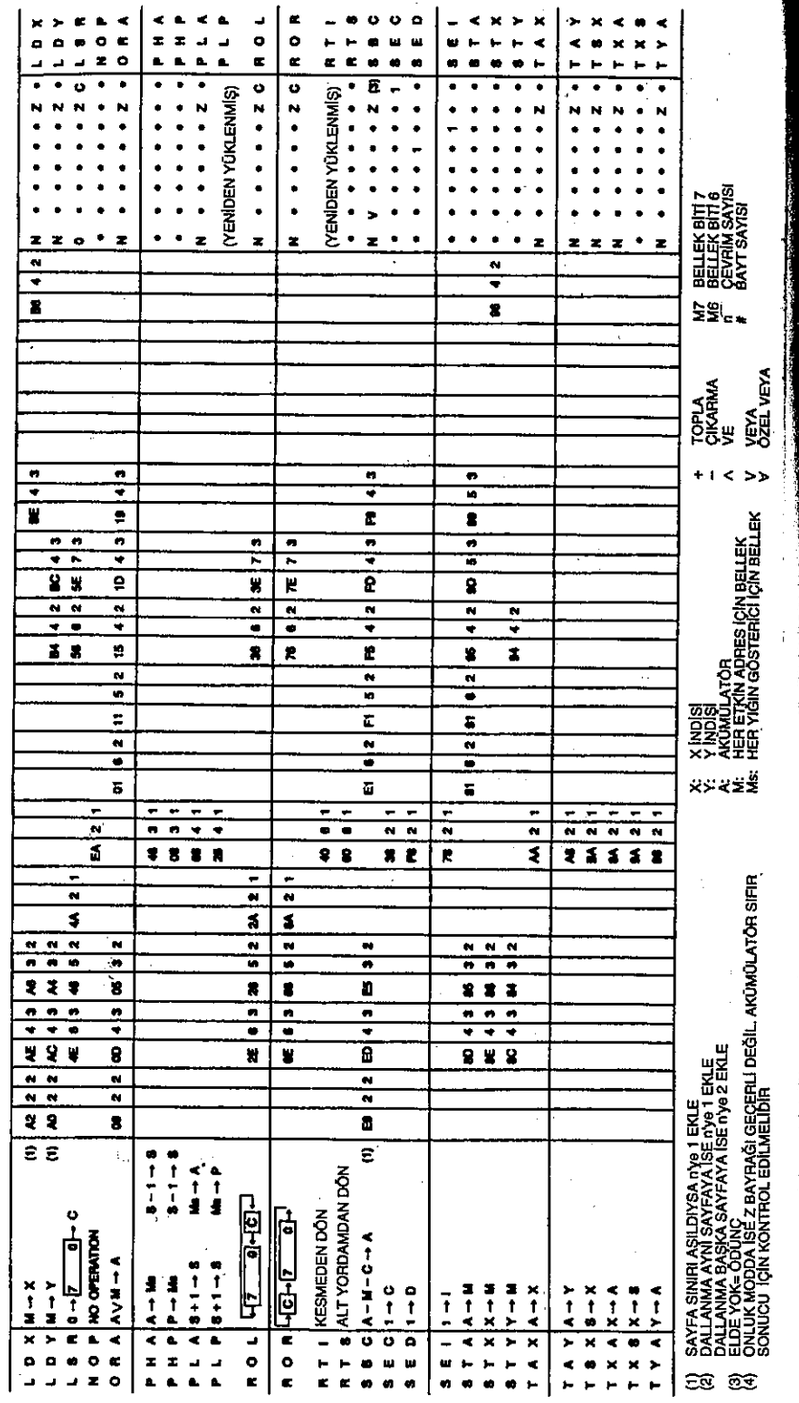
PİYASADA BULUNAN MİKROİŞLEMCİLERİN ÖZETİ VE KARŞILAŞTIRILMASI

Piyasada bulunan mikroişlemciler, dört ana kategoride toplanabilir: Genel amaçlı CPU lar, tek-yonga mikrobilgisayarlar, süper işlemciler ve bit dilimli işlemciler.

TABLO 10-2 R6502 Komut Seti

Komut	İşlem	Derhal		Mutlak		Sayfa sıfır		Akümülatör		Anlaşılır		İndis X		İndis Y		Sayfa X		Mutlak X		Mutlak Y		Göreceli		Dolaylı		Sayfa Sıfır Y		İşlemci Durum Kodu		Bellenir
		OP	#	OP	#	OP	#	OP	#	OP	#	OP	#	OP	#	OP	#	OP	#	OP	#	OP	#	OP	#	OP	#	N V - B D I Z C		
ADC	A+M+C→A	(1)	2	4	3	5	2					61	6	71	5	75	4	70	4	79	4							N V • • • • Z C		ADC
AND	A∧M→A	(1)	2	4	3	5	2					21	6	31	5	35	4	30	4	39	4							N • • • • • Z •		AND
ASL	C←[7 0]←0			6	3	6	2	0A	1							16	6	1E	7	3E	7							N • • • • • Z C		ASL
BCC	C=0 ISE DALLAN																					90	2					• • • • • • •		BCC
BCS	C=1 ISE DALLAN																					B0	2					• • • • • • •		BCS
BEQ	Z=1 ISE DALLAN																					F0	2					• • • • • • •		BEQ
BIT	A∧M			2C	4	24	3																					M7 M6 • • • Z •		BIT
BMI	N=1 ISE DALLAN																					30	2					• • • • • • •		BMI
BNE	Z=0 ISE DALLAN																					D0	2					• • • • • • •		BNE
BPL	N=0 ISE DALLAN																					10	2					• • • • • • •		BPL
BRK	DURMA									00	7																	• • 1 • • • •		BRK
BVC	V=0 ISE DALLAN																					50	2					• • • • • • •		BVC
BVS	V=1 ISE DALLAN																					70	2					• • • • • • •		BVS
CLC	0→C									18	2																	• • • • • • 0 •		CLC
CLD	0→D									D8	2																	• • • • 0 • • •		CLD
CLI	0→I									58	2																	• • • 0 • • • •		CLI
CLV	0→V									B8	2																	• • • • • 0 • •		CLV
CMP	A−M			CD	4	C5	3					C1	6	D1	5	D5	4	D0	4	D9	4							N • • • • Z C		CMP
CPX	X−M			EC	4	E4	3																					N • • • • Z C		CPX
CPY	Y−M			CC	4	C4	3																					N • • • • Z C		CPY
DEC	M−1→M			CE	6	C6	5																					N • • • • Z •		DEC
DEX	X−1→X									CA	2																	N • • • • Z •		DEX
DEY	Y−1→Y									88	2																	N • • • • Z •		DEY
EOR	A∨M→A			4D	4	45	3					41	6	51	5	55	4	50	4	59	4							N • • • • Z •		EOR
INC	M+1→M	(1)	2	EE	6	E6	5									F6	6	FE	7									N • • • • Z •		INC
INX	X+1→X									E8	2																	N • • • • Z •		INX
INY	Y+1→Y									C8	2																	N • • • • Z •		INY
JMP	YENİ YERE ATLA			4C	3	3																		6C	5	3		• • • • • • •		JMP
JSR	ALT YORDAMA ATLA			20	6	3																						• • • • • • •		JSR
LDA	M→A	(1)	2	AD	4	A5	3					A1	6	B1	5	B5	4	B0	4	B9	4							N • • • • Z •		LDA

TABLO 10-2 (R6502 Komut Seti (Devamı)



GENEL AMAÇLI MİKROİŞLEMCİLER

Genel amaçlı mikroişlemcilerin örnekleri Intel 8085A, zilog Z80A, Motorola MC6800 ve MOS Technology 6502'dir. Bu aygıtlar, herhangi bir büyüklükte bir mikrobilgisayar sistemi kurmak için harici ROM, RAM, ve I/O port entegreleri kullanırlar.

TEK-YONGA MİKROBİLGİSAYARLAR

Intel 8048 ailesi ve Motorola 6801 gibi yonga, tek bir entegre üzerinde ROM, RAM, saat mantık devresi ve I/O portları içermektedirler. Bunlara harici ROM ve RAM eklenebilir fakat çoğu uygulamalarda yonga üzerindeki ROM ve RAM yeterli olmaktadır.

SÜPER İŞLEMCİLER

Mikroişlemci gelişmelerinin bir yönü de, bir mikrobilgisayarın bütün CPU işlevlerini gerçekleştiren aygıtlara doğru olmuştur. Minibilgisayarların tipik olarak 16 veya 32 bit genişliğinde veri yolları vardır. Bunların adres yolları da çok geniştir, böylece megabayt boyutlarında bellek adresleyebilirler. Mikrobilgisayarlar, kelimelerle, baytlarla, nibillarla, bitlerle ve karakter dizileriyle çalışabilme yeteneğine sahiptirler. Komut çevrim süreleri tipik olarak 1 µs civarındadır.

Minibilgisayar tipi CPU'ların örnekleri Intel iAPX 86/10 ile iAPX 88/10, Motorola MC68000 ve Texas Instruments TMS9900'dür. Aşağıda herbirinin kısa bir tanımını yapacağız.

iAPX 86/10 VE iAPX 88/10: Çoğunlukla 8086 denilen *iAPX 86/10, 8080A* nın 10 MHz lik ve 16 bitlik işlemciye mantıksal uzantısıdır. Bu aygıt, 40 bacaklı paketinin içinde sadece bir CPU içerir. Bir CPU grubu oluşturmak için harici bir 8284 saat üreteci, harici bir 8288 yol kontrolü ve harici adres yol çoğullayıcıları gerekir. 16 veri ve 20 adres hattı bir 20-bitlik yola çoğullanır. 20 adres hattı, 1.048.576 bayta kadar belleğe erişebilmesini mümkün kılar.

Şekil 10-16 da gösterildiği gibi, iAPX 86/10 yol *bağdaştırma birimi (BIU)* ve *yürütme birimi (EU)* olmak üzere iki bağımsız işlevsel kısma bölünmüştür. BIU, EU için yollarındaki bütün veri ve adres aktarımlarını yapar. Yürütmeyi hızlandırmak için, BIU, altı taneye kadar bellek baytını zamanından önce bellekten getirir ve EU için bunları QUEUE (kuyruk) kaydedicilerinde tutar. EU bir komutun kodunu çözerken veya dahili bir komutu yürütürken BIU 'da komutları getirebilir. Ayrıca, EU, bir sonraki komutunu, bellekten almaya göre QUEUE'dan daha hızlı alabilir. Atlama komutları dışında, bu önceden getirip kuyrukta bekletme yöntemi işlemeyi büyük ölçüde hızlandırır.

BIU, 20-bitlik bir adres gönderir ve böylece 1.048.576 yeri adresleyebilir. Ancak, herhangi bir anda, BIU adres alanı içindeki 64 K baytlık segmanların (segment) sadece dört tanesiyle çalışır. Bu dört segmanın herbiri için taban adresinin üst 16 biti, BIU nun segman kaydedicisinde bulunur. Alt 4 bitin 0 olduğu varsayılır. Segmanlar üstüste binebilir. Kod segmanı, komutlar için, yığın segmanı, yığınlar için, veri ve ekstra segmanları veri için kullanılır. Bu segmanlı bellek yaklaşımının avantajı, bellek alanının çeşitli bölgelerinde birçok programın bulunabilmesidir. İşlemci, bir programdan diğerine, dört segman kaydedicisini içeriğini değiştirerek kolayca geçebilir.

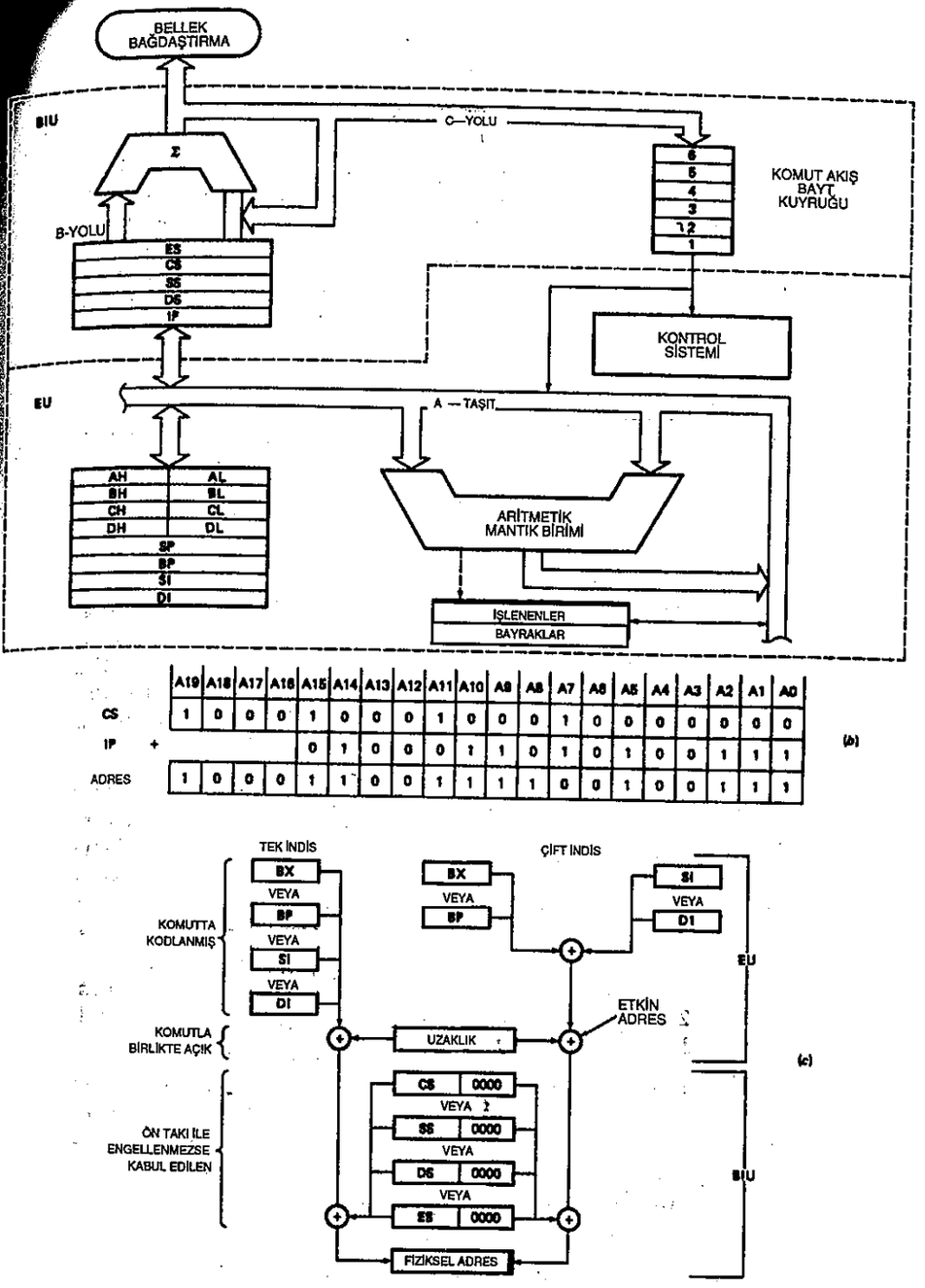
BIU' da ayrıca program sayıcı olarak çalışan komut gösterici kaydedicisi (IP) bulunmaktadır. Bir komutun getireceği yerin adresini hesaplamak için, BIU, kod segman kaydedicisinin içeriğini buna ekler. Şekil 10-16b'de gösterildiği gibi, sonuç 20 bitlik bir adrestir. O halde, komut gösterici, kod segmanı içindeki kaymayı belirtmektedir.

iAPX 86/10'un yürütme birimi (EU), bir aritmetik mantik birimi, dokuz bayrak flip-flopu ve sekiz 16-bitlik kaydedici içerir. Şekil 10-16 'da gösterildiği gibi, *A, B, C* ve *D* kaydedicileri 8 veya 16 bitlik kaydediciler olarak kullanılabilir. *AL* kaydedici, 8085A'nın akümülatörüne; *BL* ve BH, 8085A'nın *HL*; CH ve CL, 8085A 'nın *BC*; ve *DH* ve *DL*, 8085A'nın *DE* kaydedicilerine karşılık gelir. Bu benzer kaydedici yapısı, 8085A programlarının bir iAPX 86/10 'da çalıştırılmak üzere tercüme edilmesini kolaylaştırır.

EU 'daki diğer dört kaydedici, *yığın gösterici (SP), taban gösterici (Base pointer = BP), kaynak indisi (source index = SI)* ve *hedef indisi (destination index=DI)* kaydedicilerdir. Bu indis kaydedicileri, aygıta zengin bir adresleme modu dizisi vermektedir. Şekil 10-16*c*, bir adresin hesaplanma yollarını göstermektedir. *EU* önce bir etkin adres EA'yı hesaplar. Bunu yapmanın bir yolu, *BX, BP, SI* veya *DI*' nın içeriğini komutta içerilen kaydırmaya eklemektir. Diğer bir yol, *BX* veya *BP* nin içeriğine, *SI* veya *DI*'nın içeriği ile komutta içerilen kaydırmayı eklemektir. *Hesaplanan etkin adres (EA) BIU*'ya iletilir. Burada, *EA*, 20-bitlik adresi üretmek için segman kaydedicisinin kaydırılmış içeriğine eklenir.

Bilinen mikroişlemci komutlarına ek olarak, iAPX 86/10 komutlarında birçok çarpma ve bölme şekilleri vardır. İndislenmiş ve dolayli adresleme modları bulunmaktadır ve *iAPX*, bitlerle, baytlarla ASCII karakter dizileriyle, kelimelerle veya bloklarla çalışabilir. Eleman, ASCII karakteri şeklindeki sayılar üzerinde doğrudan aritmetik işlemleri yapabilir.

iAPX 88/10, iAPX 86/10 ile aynı komut kümesine benzer dahili mimari yapıya sahiptir. Ancak, bu aygıt bir 8-bitlik programlanabilir çevre aygıtlarıyla bağdaştırmayı kolaylaştırır. iAPX 88/10, IBM Personel Computer (PC) de kullanılan CPU'dur.



Şekil 10-16*(a)* Intel 8086'nın işlevsel blok diyagramı. *(b)* Kod segman kaydedicisini ve komut göstericisini kullanarak 8086 da adres hesaplama. *(c)*Bellek adresi hesaplaması.

MC68000 Motorola MC68000 de bir 16-bitlik işlemcidir. Eleman 64 bacakla bir pakete konulmuştur, bu nedenle 23 bitlik adres yolu ve 16-bitlik veri yolu çoğullanmamıştır. Bu 23 adres hattı ile 1 tane çoğullanmış hat, 68000'in 16 Megabayta kadar adresleme yapabilmesine imkan vermektedir.

Kaydediciye-yönelik işlemcileri örnek alarak, Motorola 68000'e sekiz tane 32-bitlik genel amaçlı kaydedici, yedi tane 32-bitlik adres kaydedici ve iki 32-bitlik yığın göstericisi koymuştur. Ayrıca, bir 32-bitlik program sayıcı ve bir 16-bitlik durum kaydedicisi bulunmaktadır. 68000, d.a.'dan 12MHz'e kadar herhangi bir frekansta bir tek-fazlı saat kullanır.

68000'in sadece 56 temel komut tipi vardır, fakat çoğu temel komutta kullanılabilen 14 adresleme modu bulunmaktadır. Bu nedenle, toplam mümkün işlemler sayısı büyüktür. 68000, ayrıca hata ayıklamada yardımcı olacak yazılım ve donanım özellikleri içerir.

TEXAS INSTRUMENTS TMS 9900 AİLESİ

TMS 9900 ailesi mikroişlemciler hem NMOS hem de I^2L tipleri içerir. CPU lar, bir program sayıcı, çalışma alanı kaydedicisi ve durum kaydedicisi içerir. Bu ailenin en dikkate değer özelliği, çoğunun bir dahili akümülatörünün veya veri kaydedicilerinin olmayışıdır. 256-baytlık bir RAM bölgesi onaltı tane 16-bitlik genel amaçlı kaydedici olarak kullanılmak üzere bir kenara ayrılır. Genel amaçlı demek, herhangi bir tanesinin bir akümülatör olarak, bir sayıcı olarak veya veri saklamak için kullanılabileceği demektir. Bu kaydedici bankasının başlangıç bellek adresi, CPU'daki çalışma alanı kaydedicisinde tutulur. Bütün kaydediciler için belleği kullanan bu *sisteme bellekten-belleğe mimari yapı (memory-to-memory architecture)* denir. Bu mimari yapının bir avantajı bir kesme hizmet yordamının, sadece çalışma alanı kaydedicisine yeni bir değer koyarak yeni bir kaydedici bankasına erişebilmesidir. Bu tekniğe *"bağlam değiştirme" (context switching)* denir. Bu teknik, bir grup kaydediciyi yığına ittirmekten daha hızlıdır. Ancak, çoğu komut zamanları daha uzundur, çünkü her işlem bir bellekten okuma ve bir belleğe yazma gerektirir. Bu, aynı yonga üzerindeki kaydedicilere ulaşmaya göre daha uzun zaman alır. Bu ailenin 12MHz lik bir üyesi olan TMS 9995, bu problemi çözmek için, üzerinde 256 baytlık RAM içerir.

BİT-DİLİMLİ İŞLEMCİLER

Daha önce anlatılmamış olan diğer bir işlemci tipi, bit-dilimli tiptir. Genel amaçlı mikroişlemcilerin yeterince hızlı olmadığı veya istenilen bir komutu bulunmadığı uygulamalarda, bir CPU 'yu özel olarak yapmak için bit dilimleri kullanılır.

Başlangıç bloğu, yani dilimi, bir 4 veya 8 bitlik ALU ile belki bazı kaydediciler ve çoğullama devresidir. Temel bit dilimleri, daha geniş kelimelerle çalışabilmeleri için, paralel olarak bağlanabilirler. Faydalı işlemler yapabilmek için, bit dilimine gerçekleştireceği her küçük adımın söylenmesi gerekir. Bu basit adımlara mikrokomutlar denir. Tasarımcı, gerçekte her komutu, o komut için bir dizi mikrokomutlar belirleyerek yaratır. Bir mikrokomutlar dizisi, bir ROM da saklanabilir. Bir komut çözücü, her komut için istenilen diziyi seçer. Bir mikrokomut dizisi, 8080A gibi bir işlemcinin bir makina dili komutuna karşılık gelir. Çoğu uygulama, bit dilimlerinden yapılmış bir CPU gerektirmediği için şanslı sayılırız. Bit-dilimli işlemcilerin örnekleri olarak Motorola 10800 ECL serisi ve Advanced Micro Devices 2900 Schottky TTL serisi verilebilir.

GÖZDEN GEÇİRME SORULARI VE PROBLEMLER

1. *MC6800* mimari yapısının bir blok diyagramını çiziniz. *6800*, dahili yazboz kaydedicilerinin yerine ne kullanılır?
2. *6800 CPU*'nun, $\overline{RESET}$ girişindeki bir alçağa olan cevabını anlatın. Bu, *8080A*'nın cevabından nasıl farklıdır?
3. $\overline{NMI}$ girişindeki bir alçağa, *6800*'ün cevabını anlatın.
4. *6800*'ün taşma bayrağı ile hangi durum belirtilir?
5. Aşağıdaki *6800* adresleme modlarını tanımlayın:

   a. Derhal — *b.* Doğrudan
   c. İndislenmiş — *d.* Genişletilmiş
   e. Anlaşılır — *f.* Akümülatör
   g. Göreceli

6. Eğer indis yazmacı *227AH* içeriyorsa, *ADDB $3B, X* komutu için işleneni getirmek üzere *CPU* hangi adrese gidecektir?
7. Eğer BRA $32 komutu 07A5H ve 07A6H bellek yerlerinde saklıysa, *CPU*, bir sonraki komutu getirmek üzere hangi adrese gidecektir? Not: Elde çevrim yapıldığını varsayın, böylece kaydırma, kaydırmanın saklandığı yerden sonraki adresten itibaren ölçülecektir.
8. Tablo 10-1 i kullanarak, aşağıdaki *6800* komutlarıyla gerçekleştirilen işlemleri anlatın:

   a. ADD #$2A — *f.* STAB $0076
   b. BITA$09 — *g.* WAI
   c. LDX $345E — *h.* RTI

d. SWI
e. ROR $27,X
i. TST $31,X
j. BGE Ş10

9. *6800*'ün bekleme durumunu tanımlayın. *8080*'in bekleme durumundan nasıl farklıdır?

10. Şekil 10-3'teki *WAITMS* yordamını, 0,3 ms'lik gecikme verecek şekilde yeniden yazın.

11. İki çokbaytlı sayıyı çıkarmak için kısa bir *6800* çeviri dili yordam yazınız.

12. İki 8-bitlik sayıyı ardarda toplama yöntemiyle çarpan bir *6800* çeviri dili yordam yazınız.

13. *0-9* ve *A-F* arasındaki bir *ASCII* karakteri, dengi olan 4-bitlik onaltılığa çeviren bir *6800* çeviri dili yordam yazınız. *ASCII* kodları için Tablo 3-2 ye bakınız. *ASCII* kodunun *A* akümülatöründe olduğunu ve sonucun, *B* akümülatörünün alt 4 bitinde saklanacağını varsayın.

14. Bir 6800 durma durdurma, yollarının durumunu anlatın.

15. Şekil 10-5'teki sistemde, *74155*'in $\overline{2/3}$ çıkışı, bellekteki hangi adres bloğunu yetkilendirmektedir? Şekil 10-5'teki sistem için bir bellek haritası çizin.

16. *ACIA* ve *PIA*'yı tanımlayın.

17. Şekil 10-4*a* ve *b*' deki *6800* programlarının yürütme sürelerini hesaplayın. 1 MHz'lik bir saatin kullanıldığını varsayın. Şekil 10-4*b*'de *N*=5 olsun.

18. *a.8010H* taban adresli bir *MC6821* i aşağıdaki şekilde ilklendirmek için gerekli çeviri dili komutları yazın: *Port A7-A2* çıkış; port *A1-A0*, giriş; ve port *B7-B0*, çıkış olacak. Bir $\overline{RESET}$'ın gelmiş olduğunu varsayın.

    b. Port A 'nın bit 7 sine bağlanmış bir *LED*' i 1 s yakan ve 1 s söndüren bir çeviri dili program yazın.

19. *a.* Şekil 10-5'teki sisteme bir başka *6821 PIA*'yı başlangıç adresini *8040H* yapmak için, nasıl ekleyeceğinizi gösterin.

    b. Şekil 10-5'teki sisteme bir *2732 EPROM*'u başlangıç adresi *4000H* olacak şekilde nasıl bağlayacağınızı gösterin.

20. Şekil 10-5 'teki gibi bir *6800* tabanlı mikrobilgisayar sistemi için, aşağıdaki problemlerin kaynağını belirlemek için atacağınız adımları anlatın:

    a. Güç kaynağı tamam, saat var, işlemci çalışmıyor, adres yolu *FFFEH* gösteriyor.

b. Güç kaynağı tamam, saat var, resetten sonra veri yolunda hareketlilik var, sistem monitör programını yürütmüyor.

c. Monitör programı yürütülmekte, fakat *RAM*'a veri yazamıyorsunuz.

21. *MC6800*'e göre *MC6801*'in üstünlüklerini anlatın.

22. *a.* Dolaylı adresleme, *(b)* genişletilmiş indislenmiş adresleme, *(c)* indislenmiş dolaylı adresleme ve *(d)* göreceli dolaylı adreslemeyi tanımlayın.

23. Şekil 10-2 'deki programı *6502A*'yı 2 MHz lik bir saatle çalıştırmak için, yeniden yazın.

24. Sıfır-sayfası adreslemeyi ve boru hattını tanımlayın.

25. 6502 için, indislenmiş dolaylı adresleme ile dolaylı indislenmiş adresleme arasındaki fark nedir?

26. *8086, Z8000* ve *MC68000*, minibilgisayarlara nasıl benzer?

REFERANSLAR/BİBLİYOĞRAFYA

MEK6800D2 Manual, Motorola Semiconductor Products. Inc., Austin, TX. 1976.

Osborne, Adam: *An Introduction to Microcomputers*, cilt 2: Some Real Products, Adam Osborne and Associates. Inc., Berkeley, CA, 1976.

M6800 Microcomputer System Design Data, Motorola Semiconductor Products. Inc., Phoenix, AZ, 1976.

MC6800 Programming Reference Manual, M68PRM(D), Motorola Semiconductor Products, Inc., Phoenix, AZ, 1976.

The 8086 Family User's Manual, Intel Corporation, Santa Clara, CA, Ekim 1979.

Microprocessor Application Manual, Motorola Semiconductor Products, Inc., (Katkıda bulunanlar: Alvin W. Moore, Vinay Khanna, Gary G. Sawyer, Thomas C. Daly, Karl E. Fronheiser, John M. DeLaune, Mark E. Eidson. James F. Vittera), McGraw-Hill, New York, 1975.

Zilog Z8000 Advance Specification, Zilog, Inc., Cupertino, CA Nisan 1978.

Motorola Semiconductor Data Sheets: MC6800, MC68A00, MC68B00, MC6801 MC6802, MC6809 (E), MC68A09 (E), MC68B09 (E), MC68000 (L4), (L6). & (L), Motorola Semiconductor Products, Inc., Austin, TX. 1978, 1980.

Rockwell R6500 Microcomputer System Data Sheet, Doküman no. 29000 D39 değişiklik 2 Ağustos 1978, R650X and R651X, parçaları Rockwell International Corporation, Anaheim, CA- 1978.

Simpson, William D., : *9900 Family Systems Design and Data Book*, Texas Instruments, Inc., Houston, 1978.

BÖLÜM 11

ARABİRİMLER ÜZERİNDEN GİRİŞ VE ÇIKIŞLARIN BAĞLANTILARINI KURMA

Mikroişlemci tabanlı aletlerin geliştirilmesinde çoğu iş programlama ve *I/O* arabirimlerinin bağlanması ile ilgilidir. Bu bölümde, mikroişlemcileri gerçek dünyaya arabirimlerle bağlamak için kullanılan bazı çok önemli teknikleri anlatacağız.

Bu bölümün sonunda siz

1. Tarama ve kesme tipinde *I/O* işlemlerini tanımlayabilecek,
2. Bir *8080A/8085A* veya *6800* için kesme servis yordamlarını yazabilecek,
3. Bir onaylı giriş veya çıkış veri transferi sırasındaki sinyallerinin sırasını tanımlayabilecek,
4. Onaylı giriş veya çıkış veri transferi için bir programlanabilir paralel port entegre devresini ilk kullanıma hazırlayabilecek,
5. Bir klavye kodlayıcının algılama, zıplama ve kodlama işlemlerini tanımlayabilecek,
6. Bir *klavyeyi tarayacak* veya kodunu çözecek programın bir *akış diyagramını* çizebilecek,
7. Sekiz basamaklı yedi-parçalı bir göstergeyi çoklamak için bir akış diyagramını çizebilecek,
8. Bir *D/A* veya *A/D* çeviricinin bir mikroişlemciye nasıl bağlandığını gösteren bir devreyi çizebilecek,
9. Çoklanmamış yada çoklanmış bir *A/D* çevirici çıkışından verilerin nasıl girileceğini tanımlayabilecek,
10. *S-100*, çoklu taşıt, ve *IEEE-488*'i tanımlayabilecek,
11. *RS-232-C*, boşluk, işaret, 20mA akım döngüsü, *RS-422*, *RS-423*, *MODEM* ve frekans kaymalı anahtarlamayı tanımlayabilecek,
12. Asenkron seri veri iletimi için *82514A* veya *MC6850* gibi bir programlanabilir seri port entegre devresini ilk kullanıma hazırlayabilecek,
13. *BISYNC* veya *SDLC* formatını kullanarak sayısal verinin seri olarak nasıl gönderildiğini tanımlayabilecek,
14. Alfanümerik karakterlerin bir *CRT* üzerinde nasıl gösterildiğni açıklayabileceksiniz.

BASİT, TARAMALI VE KESMELİ I/O

Basit Giriş ve Çıkış

Bir mikroişlemci, birçok uygulama için veriye ihtiyacı olduğunda veriyi yalnızca bir paralel porttan okur ve gönderilecek veri hazır olduğunda bu veriyi paralel porta gönderir. Basit bir çıkış örneği, bir port hattına bağlanmış bir lambayı her 30 saniyede açmak ve kapamak olabilir.

Taramalı I/O

Taramalı I/O modunda, harici bir cihazdan gelen sinyal hazır hattı mikroişlemci tarafından bir port hattı üzerinden üstüste okunur. Bu sinyal hazır hattı onaylandığında, mikroişlemci harici cihaz tarafından bir porta gönderilen bir baytlık veriyi okur veya bir baytlık veriyi harici devreye gönderir. Şekil 8.21 ve 10.2'deki örnekler verilerin taramalı girişinin örnekleridir. Hazır sinyali ASCII kodlanmış klavyeden gelen tuşbasımlı strob olabilir. Strob onaylandığında basılan tuşun geçerli ASCII kodu kodlayıcının paralel çıkışlarındadır ve mikroişlemci tarafından okunabilir.

Taramalı I/O'nun dezavantajı, bilgisayarın hazır sinyalini taraması sırasında başka bir işi yapamamasıdır. Mikrobilgisayarın birçok iş yapması gereken sistemlerde tarama bir zaman kaybıdır. Bu yüzden bu sistemlerde kesme-sürmeli I/O kullanılır.

KESMELİ I/O

Hemen bütün mikrobilgisayarlar bir yada daha fazla kesme girişine sahiptir.Yetkilendirilmiş bir kesme girişi, bir dış sinyalin mikrobilgisayara yürütmekte olduğu deyimi bitirmesini ve kesme tarafından belirtilen ihtiyaca cevap verecek bir altyordama atlamasını söylemesine imkan verir. Kesme servis altyordamının sonunda bir dönüş deyimi, CPU'yu kesme gelmeden önce çalıştırdığı programa geri gönderir.

Kesmelerin birçok önemli fonksiyonları ve avantajları vardır. Mikroişlemciyi, tuşbasma strobu gibi bir sinyalin mevcut olup olmadığını anlaması için bir girişi devamlı tarama zahmetinden kurtarır. Bir mikrobilgisayarın çok ani cevap gerektiren durumlara cevap verebilmesine imkan sağlar. Mikrobilgisayarı aynı zamanda zamanlama ve gecikme döngüsü zahmetlerinden kurtarırlar.

Takip eden bölümlerde kesmeler kullanıldığı zamanki donanım ve yazılımda gözönüne alınması gerekenleri tartışacağız. Bunu takiben hem 8085A ve hem de 6800'de kesme kullanma örnekleri gösterilecektir.

DONANIMDA GÖZÖNÜNE ALINACAKLAR: Belirli bir mikrobilgisayar ile kesme kullanırken gözönüne alınacak önemli donanım noktaları aşağıdaki gibidir:

1. Mikrobilgisayar kaç tane kesme girişine sahiptir?

2. Bu girişlerin öncelikleri var mı?

3. Bu girişler sinyalleri onaylamak için aktif yüksek, aktif alçak ya da kenar aktif sinyaller mi gerektiriyor?

4. Giriş sinyalleri ne kadar süre için onaylanmalıdır?

5. Yeniden başlama sinyali sokmak için harici donanım gerekli mi, yoksa bu kesme isteğine cevap verdiği zaman CPU tarafından otomatik olarak mı yapılıyor?

YAZILIMDA GÖZÖNÜNE ALINACAKLAR

1. Kesmeleri yetkilendirmek ve / veya maskelerini kaldırmak için hangi deyimler gerekmektedir?

2. Yığın gösterici ilk gösterime nasıl hazırlanmıştır?

3. Kesmeye cevap verdiği zaman CPU, ana program kaydedici içeriklerini, bayrakları yığının üzerine iterek otomatik olarak saklıyor mu? Veya kesme servis yordamı bu işlemleri yapmak için başlangıcında itme deyimlerine mi sahip olmalı?

4. Ana program içinde kesme nerede gelirse gelsin servis yordamı için gerekli veriye nasıl erişilir?

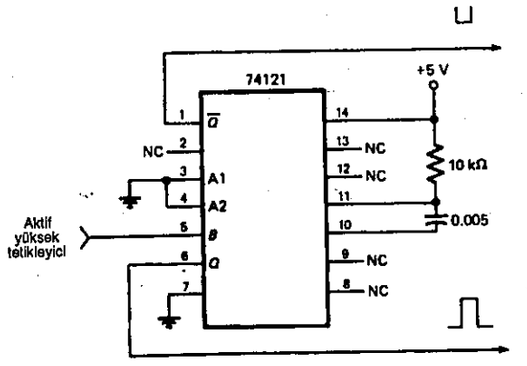
5. Ana program kaydedicilerini yeniden yüklemek, kesmeleri yetkilendirmek ve kesme servis yordamı tamamlandığında ana programa dönmek için neler gereklidir?

BİR 8085A KESME ÖRNEĞİ

Bölüm 9'da tartıştığımız gibi *8085A beş kesme girişine sahiptir. TRAP, RST 7.5, RST 6.5, RST 5.5* ve *INTR*. İki kesme aynı anda meydana geldiğinde, *TRAP* en yüksek önceliğe ve *INTR* en düşük önceliğe sahiptir. *TRAP* girişi alçaktan-yükseğe giden bir kenarı olan ve tanınıncaya kadar yüksekte kalan bir sinyal gerektirir. *RST 7.5* yalnızca yükselen bir kenar gerektirir; çünkü bu kenar bir dahili *RST 7.5* flip-flop'unu set eder. *RST 6.5, RST 5.5* ve *INTR* tanınıncaya kadar yüksek seviyede kalan bir sinyal gerektirir. Bu örnek için *RST 6.5* kesmesini kullanacağız.

Bir 8085A, RST 6.5 girişinde bir yüksek giriş aldığında kesmeleri otomatik olarak yetkisiz kılar, program sayıcısı içeriğini (dönüş adresi) yığının üstüne iter, ve bir sonraki deyimi almak için 0034H adresine vektörler. Böylece bir yeniden başlama deyimini sokmak için harici bir donanıma gerek yoktur. Fakat bir kesmenin olması için *RST 6.5* girişinin yüksek olması gerçeğinden dolayı, eğer harici bir cihazdan gelen kesme sinyali uzun süre yüksek kalırsa, *8085A RST 6.5* girişindeki ilk yüksek seviyesi algılandığında kesme isteğine cevap verir, ve daha sonra harici cihazdan gelen kesme sinyali ile hemen yeniden kesmeye maruz kalır. Bu problem Şekil 11.1'de gösterilen darbe daraltıcı devresi ile çözülebilir. 74121 *B* girişine harici bir cihazdan gelen aktif yüksek sinyal uygulandığında *Q* çıkışı, *R* ve *C* değerleri tarafından belirlenen bir süre için yüksek darbe üretir. Bu süre şekilde gösterilen değerler için 20μ saniyedir. 74121'in *Q* çıkışı 8085A'nın *RST 6.5* girişine bağlanır. Bu *RST 6.5*'un tanınabilecek kadar uzun süre yüksek kalmasını, fakat birçok kesme meydana gelmesini önleyecek kadar kısa süre yüksek kalmasını garanti eder.

Bir resetten sonra 8085A RST 6.5 hem maskelenir hem yetkisizleştirilir. Bir kesme isteğine cevap vermeden önce Bölüm 9'da anlatıldığı gibi SIM deyimi ile maskesi kaldırılmalı ve EI deyimi ile yetkilendirilmelidir.



Şekil 11.1 Kesme Girişi için Darbe Daraltıcı

Şekil 11.2, göstericileri ilk kullanıma hazırlayan ve bir sistemin ana programını benzeten bir 8085A çevirici dil programını göstermektedir. Aynı zamanda port 29H'den bir baytlık veriyi okuyan, veriyi bellekteki bir tablo içine depolayan, ve bir sonraki kesmeyi beklemek için ana programa dönen bir servis yordamı da gösterilmektedir. Programlar bir SDK-85 mikrobilgisayar kartı üzerinde çalışacak şekilde yazılmıştır, fakat 8085A tabanlı herhangi bir mikrobilgisayar üzerinde çalışacak şekilde değiştirilebilirler.

Yığın gösterici 20C2H'ye ayarladıktan sonra, şekil 11.2a'daki benzetilmiş anaprogram, *HL* kaydedicisini, veri baytlarını bellek içinde depolanacağı tablonun başlangıcını gösterecek şekilde ayarlar.

Normal bir ana program *HL*'yi birçok iş için kullanabileceğinden, gösterici bellek içinde bozulmayacağı ve alt yordamın gerektiğinde erişebileceği bir yere saklanmalıdır. SHLD 20B0H deyimi *L*'nin içeriğini bellek adresi 20B0H'ye ve *H*'nin içeriğini 20B1H adresine depolar. Daha sonra 20B0H ve 20B1H gösterici deposu olarak ayarlanır. Gerektiği zaman LHLD 20B0H deyimi gösterici değerini HL'ye yüklemekte kullanılır. MVI A, 00H deyimi yalnızca *RST 6.5*'un maskesini kaldırmak için gerekli SIM kelimesini akümülatöre yükler. Daha sonra *SIM, RST 6.5*'un maskesini kaldırır ve EI deyimi ile kesmeler yetkilendirilir. Ana program JMP WAIT deyimi tarafından benzetilir. Bu deyim bir kesme oluşuncaya kadar defalarca yürütülür.

PROGRAM BAŞLIĞI - 8085A - RST 6.5 KESMESİ GELİNCE VERİ OKU

```
ADRES  DEYİM   ETİKET  İŞKODU  İŞLENEN   AÇIKLAMALAR
2000   31C220          LXI      SP,20C2H  ; YIĞIN GÖSTERİCİYİ AYARLA
2003   218020          LXI      H, 2080H  ; TABLO ADRESİNİN
                                          ; BAŞLANGICINI YÜKLE
2006   22B020          SHLD     20B0H     ; TABLO ADRESİNİ SAKLA
2009   3E0A            MVI      A,0DH     ; RST 6.5'İN MASKESİNİ
                                          ; KALDIRMAK İÇİN KOD
200B   30              SIM                ; MASKEYİ AYARLA
200C   FB              EI                 ; KESMELERİ YETKİLENDİR
200D   C30D20  WAIT     JMP      WAIT      ; KESMELERİ BEKLE
                                          ; (Ana programı benzetir)

20C8   C32020          JMP      RDATA     ; KESME SERVİS YORDAMININ
                                          ; ANA KISMINA ATLA
```

(a)

PROGRAM BAŞLIĞI - 8085A - RST 6.5 KESMESİ GELİNCE VERİ OKU

```
ADRES  DEYİM   ETİKET  İŞKODU  İŞLENEN   ; AÇIKLAMALAR
2020   F5      RDATA   PUSH     PSW       ; A ve PSW'yi SAKLA
2021   E5              PUSH     H         ; H VE L'Yİ SAKLA
2022   2AB020          LHLD     20B0H     ; GÖSTERİCİYİ, GÖSTERİCİ
                                          ; DEPOSUNDAN AL
2025   DB29            IN       29H       ; VERİYİ OKU
2027   77              MOV      M,A       ; TABLOYU DEPOLA
2028   23              INX      H         ; BİR SONRAKİ TABLO
                                          ; YERİNİ GÖSTER
2029   22B020          SHLD     20B0H     ; GÖSTERİCİYİ SAKLA
202C   E1              POP      H         ; H VE L'Yİ YENİDEN YÜKLE
202D   F1              POP      PSW       ; A VE BAYRAKLARI
                                          ; YENİDEN YÜKLE
202E   FB              EI                 ; KESMELERİ YETKİLENDİR
202F   C9              RET                ; ANA PROGRAMA GERİ DÖN
```

(b)

Şekil 11.2 *a)* İlk kullanıma hazırlama ve kesmeleri beklemek için bir 8085A çevirici dil programı *b)* Kesme geldiğinde bir veri baytını okuyan kesme servis yordamı

Bir RST 6.5 kesmesi meydana geldiğinde 8085A kesmeleri yetkisizleştirir, dönüş adresini (200DH) yığının üstüne iter, ve bir sonraki deyimi almak için 0034H'ye vektörler. Bu adreste SDK-85 monitor program ROM'u JMP 20C8H deyimine sahiptir. 20C8H adresi RAM içinde olduğundan, bu adrese kesme servis yordamının asıl başlangıcına atlama yapacak bir deyim yazabilirsiniz. Şekil 11.2'deki program için atlamalar, 0034H'den 20C8H'ye ve daha sonra kesme servis yordamının ana kısmının başladığı 2020H'ye yapılmaktadır.

Şekil 11.2b'deki servis yordamı ilk olarak yordam içinde kullanılan kaydedicileri yığının üstüne itmelidir, böylece bu kaydediciler ana programa dönüşten önce yeniden depolanabilirler. Daha sonra LHLD 20B0H bellek göstericisini, 20B0H ve 20B1H'den *HL* içine geri yükler. Veri 29H portundan okunur ve tabloya depolanır. *HL* göstericisi tablo içindeki bir sonraki yeri göstermek için bir arttırılır. Yeni değer, SHLD 20B0H ile gösterici deposuna saklanır. POP H ve POP PSW deyimleri *HL*, *A* ve bayrakların kesme gelmeden önceki değerlerini geri yükler. 8085A, *RST 6.5* kesmesine cevap verdiği zaman kesmeleri yetkisizleştirdiği için 8085A'nın bundan sonra gelecek kesmelere cevap vermesini istiyorsanız, kesmeleri *EI* deyimi ile yetkilendirmelisiniz. Dönüş deyimi dönüş adresini yığın içinden alacak ve yürütmeyi ana

programdaki bir sonraki deyime getirecektir. Bu durumda, kesme gelmeden önce JMP WAIT deyimi 200DH yüklediğinden dönüş 200DH'ye olacaktır. Daha sonra 8085A yeni bir kesme gelinceye kadar üstüste bu deyimi yürütecektir.

BİR 6800 KESME ÖRNEĞİ

Bölüm 10'da tartışıldığı gibi *6800 iki kesme girişine sahiptir, $\overline{NMI}$* ve $\overline{IRQ}$. *$\overline{NMI}$ yüksekten-alçağa bir kenar gerektirir* ve $\overline{IRQ}$ bir alçak seviye gerektirir. Şekil 11.1'dekine benzer bir darbe daraltıcı, 6800'ün aynı isteme sinyali tarafından defalarca kesilmemesini garanti etmek için kullanılabilir. 6800 bir $\overline{IRQ}$ girişi alınca program sayıcısını, dizin kaydedicisini, akümülatörü ve durum kodu kaydedicisini yığının üstüne iter. Daha sonra CPU, kesme servis yordamının adresini almak üzere FFF8H ve FFF9H'ye gider. Kesme maskesini kesmeleri yetkisizleştirecek şekilde ayarladıktan sonra, CPU FFF8H ve FFF9H'den aldığı adrese gider ve yürütmeye oradan devam eder. Bir resetten sonra 6800 $\overline{IRQ}$ girişi yetkisizleştirilir. 6800'ün bir $\overline{IRQ}$ girişine cevap vermesi için bir kesme maskesini temizle (CLI) deyimi ile yetkilendirilmelidir.

Şekil 11.3, bir kesme için hazırlamak, ana programı benzetmek ve bir $\overline{IRQ}$ girişine cevap vermek için 6800 çevirici dil programlarını göstermektedir. Servis yordamı 8004H portundan bir baytlık veri alır ve bunu 0080H adresinden başlayan bir bellek tablosuna depolar.

Şekil 11.3'teki program yığın göstericiyi ayarladıktan sonra, dizin kaydedicisini verilerin depolanacağı tablonun birinci adresini gösterecek şekilde ayarlar. Normal bir anahat programı dizin yazmacı içindeki bu göstericinin üzerine başka birşey yazabileceğinden, bu gösterici, servis yordamı tarafından gerektiği zamanlarda geri alınabilmesi için bellekte bir yere saklanmalıdır. STX $00E0 deyimi dizin kaydedicisinin içeriğini 00E0H ve 00E1H adreslerine depolar. Bir LDX $00E0 deyimi gerektiği zamanlarda göstericiyi dizin kaydedicisine geri alır. CLI deyimi, durum kodu kaydedicisindeki bit 4'ü resetler. Bu kesmeyi yetkilendirir. JMP WAIT deyimi yeni bir kesme gelinceye kadar defalarca yürütülür. Bu anahat programını benzetmektedir.

Daha öncede belirtildiği gibi, bir $\overline{IRQ}$ girişi 6800'ün FFF8H ve FFF9H adreslerine gitmesine sebep olur. Bu programın çalışması için geliştirildiği MEK6800D2 mikrobilgisayar geliştirme kartında bu adresler monitor ROM'un içindedir. JBUG monitor programı bu yerlerde E014H adresine sahiptir, böylece 6800 E014H'ye gider. E014H'de monitor programı RAM içindeki A000H adresine atlama deyimine sahiptir. A000H adresine asıl kesme servis yordamının başlangıcına atlayacak bir deyim koymalısınız. Bu size karışık gelebilir, fakat sizin gerçekten hatırlamanız gereken tek şey A000H'ye bir atlama deyimi koymaktır.

Şekil 11.3b'deki servis yordamının ana kısmı ilk olarak LDX $00E0 deyimi ile gösterici deposundan dizin kaydedicisine alınır. Sonra 8004H portundan veri okunur ve dizin kaydedicisi tarafından gösterilen bellek yerine depolanır. Sonra tablodaki bir sonraki yeri göstermek için dizin kaydedici bir arttırılır. Arttırılmış gösterici STX $00E0 deyimi ile gösterici deposuna depolanır. CLI deyimi kesmeyi yeniden yetkilendirir.

Sonra kesmeden-dönüş (RTI) deyimi, şart kodu kaydedicisinin, akümülatörün, ve dizin kaydedicisinin değerlerini yığından alır. Bu deyim, kaydedicilerin kesme gelmeden önceki değerlerini kaydedicilere geri yükler. RTI aynı zamanda dönüş adresini yığından alır ve yürütmeyi anahat programına döndürür. Şekil 11.3'teki programlar için yürütme 000AH adresindeki JMP WAIT deyimine, yani benzetilmiş anahat programına geri döner.

PROGRAM BAŞLIĞI - 6800 IRQ KESMESİNDE VERİ OKUMA

ADRES	DEYİM	ETİKET	İŞKODU	İŞLENEN	AÇIKLAMALAR
0000	8E00FF		LDS	#$00FF	; YIĞIN GÖSTERİCİYİ AYARLA
0003	CE0080		LDX	#$0080	; TABLO GÖSTERİCİYİ ; AYARLA
0006	FF00E0		STX	$00E0	; GÖSTERİCİ DEPOSUNA SAKLA
0009	0E		CLI		; KESMEYİ YETKİLENDİR
000A	7E000A	WAIT	JMP	WAIT	; KESME İÇİN BEKLE ; (Anahat programını benzetir)
A000	7E0010		JMP	RDATA	; KESME SERVİS YORDAMININ ; BAŞLANGIÇ ADRESİNE ; ATLAMAYI YÜKLE

(a)

PROGRAM BAŞLIĞI - 6800 IRQ KESMESİNDEN VERİ OKUMA

ADRES	DEYİM	ETİKET	İŞKODU	İŞLENEN	AÇIKLAMALAR
0010	CE00E0	RDATA	LDX	$00E0	; TABLO GÖSTERİCİYİ AL
0013	B68004		LDAA	$8004	; PORTTAN VERİYİ OKU
0016	A700		STAA	$00,X	; VERİYİ TABLOYA DEPOLA
0018	08		INX		; TABLODAKİ BİR SONRAKİ ; YERİ GÖSTER
0019	FF00E0		STX	$00E0	; BİR SONRAKİ OKUMA İÇİN ; GÖSTERİCİYİ SAKLA
001C	0E		CLI		; KESMEYİ YETKİLENDİR
001D	3B		RTI		; ANA PROGRAMA GERİ DÖN

(b)

Şekil 11.3 Veri okumak için MC6800 kesme servis yordamı *a)* İlk kullanıma hazırla ve kesme için bekle *b)* Kesme servis yordamı.

ZAMANLAMA İÇİN KESMELER: Bir kesmenin, verileri okuma yada göndermek için taramalı metod yerine nasıl geçtiğini gösterdik. Bir kesme girişi aynı zamanda zamanlama için gecikme döngülerinin yerine de geçebilir. Bölüm 9'da tartışılan 8155 gibi bir programlanabilir zamanlayıcı her 1msaniyede kesme girişine darbe gönderecek şekilde programlanabilir. Bu kesme işlemciye, bir baytlık bir veriyi okumasını veya göndermesini söyler. Servis yordamı, baytı girmeden veya göndermeden önce 10 kesme veya 100 kesme sayacak şekilde yazılabilir. Bu, hassas ve programlanabilir bir zamanlayıcı verir. Bir bellek yerindeki sayıyı her 1000 kesmeden sonra arttıran bir servis yordamı yazılabilir. Bellek yerindeki sayı, saniyeleri sayan bir sayıcı olacaktır. Servis yordamı başka bir bellek yerindeki sayıyı, her 60 saniyede arttırarak dakika sayıcısı, ve başka bir bellek yerindeki sayıyı da her 60 dakikada artırarak saat sayıcısı olacak şekilde genişletilebilir. Böylece bu kesme ve servis yordamı anahat programı içindeki herhangi bir süreci zamanlamak için bir gerçek-zaman saati üretir. Anahat programı gerektikçe saniye, dakika ve saat sayıcılarını bellekten okur.

ÇOKLU KESMELER: Bazı uygulamalar işlemcinin birçok sinyalden biri tarafından kesilmesini gerektirebilir. Bazı uygulamalar ilk önce en üst öncelikli kesme sinyaline servis verilmesini isteyebilir. Örneğin, bir atomik reaktör kritik duruma geliyorsa bu reaktöre taşmakta olan bir sabun makinesinden önce servis verilmelidir.

8259 programlanabilir kesme kontrolör birimi (PICU) gibi bir entegre devre, sekiz taneye kadar olan kesmelere önceliklerine göre servis verilmelerine imkân sağlar. Bir cihazdan gelen kesme sinyalinin önceliği 8259'a bağlandığı kesme istek girişi tarafından belirlenir. IR0 en yüksek önceliğe ve IR7 en düşük normal önceliğe sahiptir.

8259 sekiz IR girişinden birinden kesme isteği aldığında, 8080A veya 8085A'ya bir INT sinyali gönderir. 8080A veya 8085A bir INTA sinyali ile cevap verir. 8259 bunu sistem veri yoluna bir standart çağırma (onaltılık CD) yüklemek için kullanır. 8080A veya 8085A çağrı deyimi kodunu çözer, ve iki tane daha INTA darbesi gönderir. İkinci INTA darbesi, 8259'u istenen altyordam adresinin düşük baytını veri yoluna koyması için uyarır. 8080A veya 8085A'dan gelen üçüncü INTA darbesi, adresin yüksek baytını veri yoluna yükler. (Sekiz altyordam adresi 8259'un ilk kullanıma hazırlanması sırasında depolanmıştır). 8259 adresi çıkışa verdikten sonra 8080A veya 8085A çağırma deyimini yürütür. Dönüş adresini yığına depolar ve program sayıcısını 8259'dan aldığı adresle yükler. 8259, standart bir çağırma deyimi ve iki adres baytı sağladığından, programı bellek içindeki herhangi bir adrese vektörleyebilir. RST deyimlerinin, programı 00H ile 38H arasındaki sekiz adresten yalnızca birine vektörleyebildiğini hatırlayınız.

Bir kesmeye servis verilirken ve düşük-öncelikli bir kesme sinyali geldiğinde, daha yüksek-öncelikli kesme servisi tamamlanıncaya kadar bu istek ihmal edilir. Kesme servis altyordamlarındaki EI deyimi 8080A veya 8085A kesme girişini yeniden yetkilendirir, böylece bir sonraki kesme sinyali işlemciyi kesebilir ve ilk servis tamamlandıktan sonra servis verilebilir. 8259, programı ikinci servis yordamı için bir başka adrese vektörler.

Bir kesmeye cevap verilirken daha yüksek- öncelikli bir kesme sinyali gelirse iki durum ortaya çıkabilir. Eğer çalışan yordam başladığında bir EI deyimi yoksa, yeni gelen kesme bir EI deyimi gelene kadar ihmal edilir. Eğer EI varsa, yüksek-öncelikli kesme düşük-öncelikli kesme yordamını durdurur. 8259 bir çağırma deyimi gönderir ve adresi yüksek-öncelikli yordama vektörler. Yüksek-öncelikli servis tamamlandıktan sonra, program yürütülmesi düşük-öncelikli yordama geri döner. Düşük-öncelikli yordam da tamamlandıktan sonra yürütme ana programa döner. Bu aslında iç içe geçmiş altyordam işlemidir.

BİR MİKROBİLGİSAYAR İLE ONAYLI GİRİŞ VE ÇIKIŞ

Verinin bir sistemden başkasına düzenli olarak transferi üç yolla yapılabilir: basit bir strob I/O, tek katlı onaylı I/O, ve iki kere onaylı I/O.

BASİT STROB I/O

Şekil 11. 4a, bir çevrebirimden mikrobilgisayara verinin *basit strob I/O tekniği ile giriş-çıkışını* göstermektedir. Kodlanmış klavye gibi bir çevrebirim bir baytlık veriyi paralel porta verir. Daha sonra geçerli verinin çıkış hatlarında olduğunu belirten bir strob sinyali gönderir. Strob sinyali CPU üzerindeki bir kesme girişine bağlanabilir, böylece veri kesmeye bağlı olarak okunabilir. Strob sinyali aynı zamanda bir port hattına bağlanabilir, ve hat CPU tarafından onaylanıncaya kadar taranır. Şekil 8.21 ve 10.2'deki programlar bu işlemlerin nasıl yapıldığını göstermektedir. Basit strob metodu, çevrebirim çıkış verilerinin düşük hızda olması durumunda çok iyi çalışır. Fakat çok basit sistemler dışındaki bütün sistemlerde veri transferi onay tabanlı olarak yapılır. Bunun anlamı çevrebirimin, alıcıdan ilk baytın okunduğunu belirten bir alındı onay sinyalini geri almadan ikinci baytı çıkışa koymaması demektir.

TEK -ONAYLI I/O

Şekil 11. 4b, bir çevrebirimden bir mikrobilgisayara olan tek-onaylı veri transferi için sinyal sırasını göstermektedir. Çevrebirim bir baytlık veriyi

çıkışa verir ve geçerli verinin hazır olduğunu belirtmek için bir strob gönderir. CPU bir kesme sırasında ya da tarama tabanlı olarak strobu algılar ve veriyi okur. Daha sonra çevrebirime bir sonraki veri baytını göndermesi için bir alındı sinyali gönderir. Çevrebirim, verileri sistemin kabul edebileceği hızdan daha hızlı bir şekilde gönderemez.

ÇİFT-ONAYLI I/O

Veri transferi sırasında daha fazla haberleşmeyi gerektiren sistemlerde *çift onay* kullanılır. Şekil 11. 4c, bir çevrebirimden bir mikrobilgisayara olan veri transferi için çift-onaylı sinyal sırasını göstermektedir. Çevrebirim mikrobilgisayara "Hazır mısın?" sorusunu sormak üzere strobu alt seviyede gönderir. CPU alındı çıkışını yüksek olarak göndererek "Evet" cevabını verir. Daha sonra çevrebirim bir baytlık veriyi çıkışa koyar ve mikrobilgisayara "senin için geçerli bir veri" sözünü söylemek için strob sinyalini yükseğe çeker. CPU veriyi okur ve alındı sinyalini alçağa düşürür. Alındı sinyalinin alçak seviyeye gitmesi çevrebirime "Veriyi aldım, teşekkür ederim, ve bir sonraki baytı göndermek için sizin isteğinizi bekliyorum" der.

ONAY SİNYALLERİNİN ÜRETİLMESİ:

PROGRAMLANABİLİR PORTLU ENTEGRE DEVRELERİ

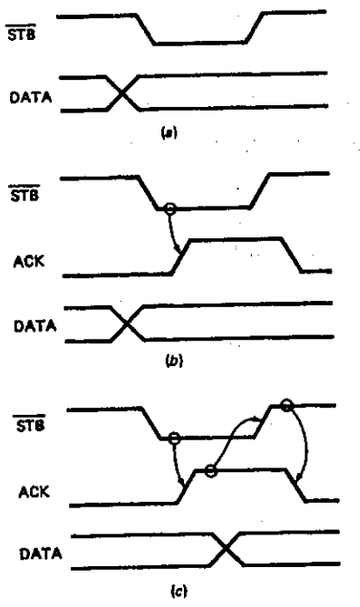
Yukarıdaki örneklerin hepsi verilerin çevrebirimden mikrobilgisayara gönderilmesi durumu içindi. Fakat, bir mikrobilgisayardan yazıcı gibi bir çevrebirime çıkış için de benzeri sinyal sırası çizilebilir. Mikrobilgisayardan gelen onay sinyalleri, yazılımla, donanımla, veya ikisinin bir kombinasyonu ile üretilebilir. CPU'yu onay sinyallerini üretme zahmetinden kurtarmak için, Intel 8155 ve Motorala MC6821 gibi programlanabilir paralel port entegre devreleri, bu sinyalleri otomatik olarak üretmek için programlanabilirler. Aşağıdaki alt bölümler bu cihazların ilk kullanıma nasıl hazırladıklarını ve onaylı I/O işlemleri için nasıl kullanıldıklarını göstermektedir.

8155'in İLK KULLANIMA HAZIRLANMASI Bölüm 9'da tartışıldığı gibi 8155, 256 baytlık RAM, iki 8-bitlik programlanabilir port, bir 6-bitlik programlanabilir port, ve bir 14-bitlik programlanabilir zamanlayıcı içermektedir. Bölüm 9'daki tartışma basit giriş ve çıkış işlemleri için üç portun nasıl ilk kullanıma hazırlandığını göstermektedir (Eğer bu işlemi şu anda hatırlamıyorsanız daha ileri gitmeden önce Bölüm 9'a dönüp bir hatırlama yapınız.) Şimdi Port A ve port B üzerinde onaylı giriş ve çıkış için, Port C'nin onaylı sinyallerini üretmek üzere nasıl programlanacağını göstereceğiz. Şekil 9.11*d*'de gösterildiği gibi 8155'in C portu dört alternatiften birine ayarlanabilir. Alternatif 1, port C'nin 6 hattının hepsini giriş olarak programlar. Alternatif 2, Port C'nin 6

hattının hepsini çıkış olarak programlar. Alternatif 3, port C'nin 0'dan 2'ye kadar olan hatlarını port A'nın onay sinyalleri için ve port C'nin 3'ten 5'e kadar olan hatlarını çıkış olarak programlar. Alternatif 4, port C'nin 0'dan 2'ye kadar olan hatlarını port A'nın onay sinyalleri için ve port C'nin 3'den 5'e kadar olan hatlarını port B'nin onay sinyalleri için programlar.

8155 ONAYLI GİRİŞ ÖRNEĞİ: TEYP OKUYUCU

Şekil 11.5*a*, 8155'in onaylı giriş yapmak için yüksek-hızlı teyp okuyucu gibi bir çevrebirime nasıl bağlandığını göstermektedir. 8155'in A portu giriş portu olarak kullanıma hazırlanır, port C alternatif 3 veya alternatif 4 olarak ayarlanır, ve port A kesmesi yetkilendirilir. 15 H veya 19 H'lik bir komut kelimesi bunu yapabilir. Şekil 11.5*b*, onaylı giriş veri transferi yapılırken meydana gelen sinyal sırasını göstermektedir.

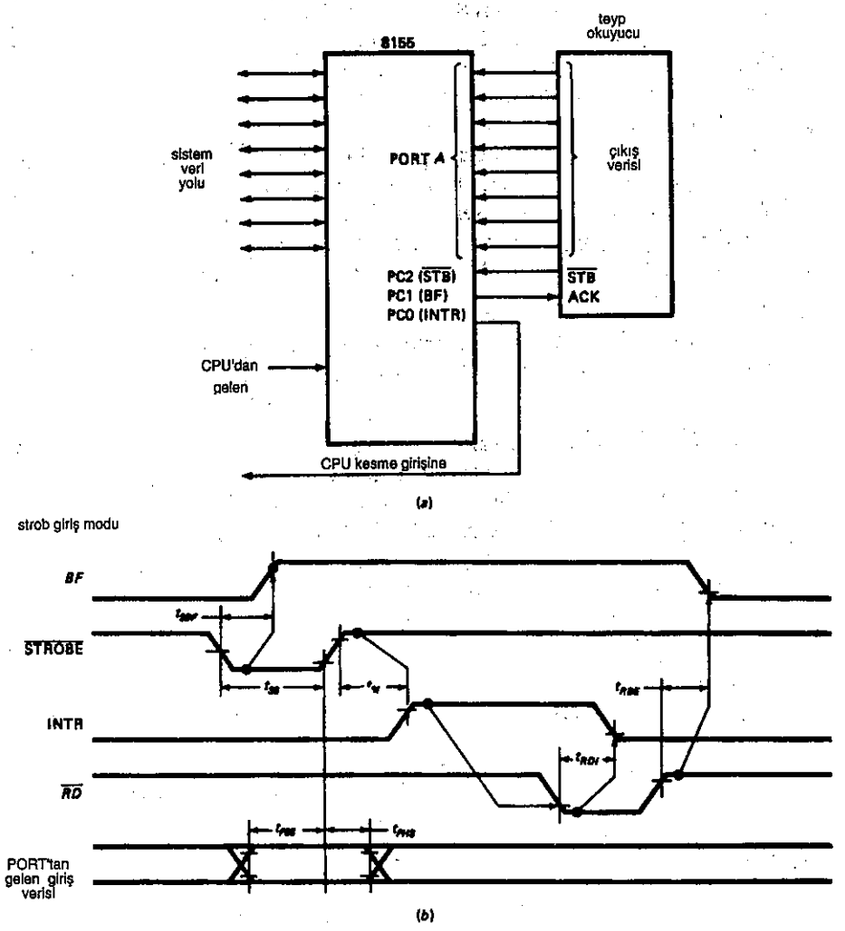


Şekil 11.4 Onaylı veri transferi *a)* Basit strob *b)* Tek seviyeli *c)* Çift seviyeli

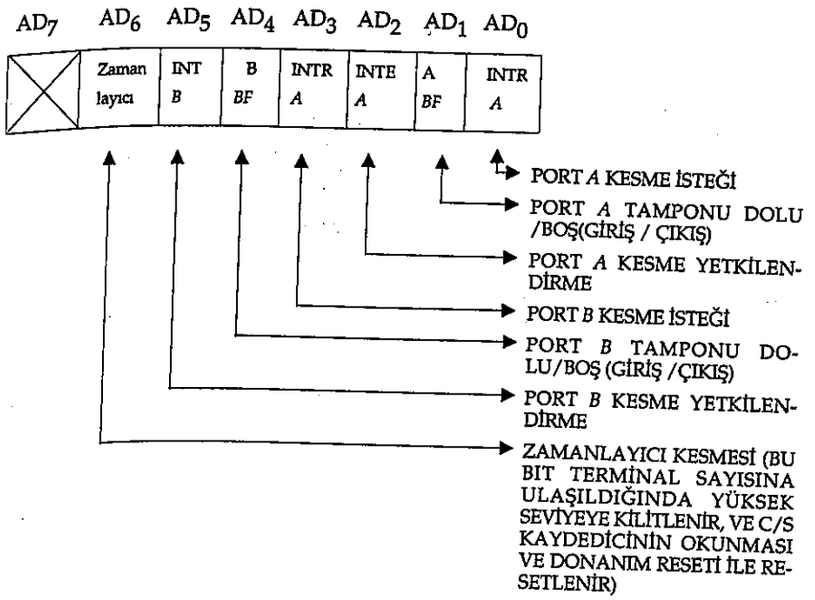
İlk olarak teyp okuyucu, 8155'e göndermek için bir baytlık veriye sahip olduğunu belirten bir alçak strob sinyali ($\overline{STB}$) gönderir. *PC2* girişinde ($\overline{STB}$) bir alçak seviye algılayan 8155, otomatik olarak *PC1, tampon dolu* (BF), çıkışını yükseğe çeker. Bu, 8155'in veri almaya hazır olduğunu belirtir. Eğer veri halen gönderilmediyse, teyp okuyucu bir baytlık veri gönderir ve $\overline{STB}$ çıkışını yeniden yüksek seviyeye çeker. $\overline{STB}$'nin yükseğe gidişi veri baytını *port A* tamponuna kilitler. $\overline{STB}$'nin yükseğe gidişi aynı zamanda 8155'in PC0'dan bir kesme sinyali göndermesine sebep olur. Bu sinyal CPU üstündeki bir kesme girişine bağlıdır. Eğer kesmeler yetkilendirilmişse, CPU 8155'den gelen veriyi, okuyan bir kesme servis yordamına vektörler. *Port A*'yı okumak için CPU elemanı adresler ve bir $\overline{RD}$ sinyali gönderir. $\overline{RD}$'nin alçak seviyeye gidişi 8155'in *port C* hat 0' daki INTR çıkış sinyalini resetlemesine sebep olur.

CPU, $\overline{RD}$ sinyalini yeniden yükseğe çektiği zaman, 8155 *port C* hat 1'deki BF sinyalini yeniden alçak seviyeye düşürür. BF'nin alçak seviyeye gidişi teyp okuyucuya yeni bir karakter göndermek için aynı sırayı tekrarlayabileceğini belirtir.

CPU, 8155 kontrol sinyallerinin herhangi bir andaki durumunu 8155 statü kaydedicisini okuyarak anlayabilir. Statü kaydedicisi, kontrol kaydedicisi ile aynı adrese sahiptir. Şekil 11.6, 8155 statü kelimesinin bit formatını göstermektedir. Çevrebirime kesme tabanlı olarak servis vermek yerine CPU, çevrebirimin karakter gönderdiğini ya da karakter almaya hazır olduğunu statü kaydedicisinin 0 bitini tarayarak anlayabilir.



Şekil 11.5 8155 kullanarak onaylı veri transferi *a)* Yüksek-hızlı teyp okuyucuya bağlantı *b)* onaylı veri girişi için sinyal sırası



Şekil 11.6 8155 Statü Kaydedicinin Bit Formatı

8155 ONAYLI ÇIKIŞ ÖRNEĞİ: KONUŞMA SENTEZLEYİCİ

8155, onaylı yada stroblu çıkış veri transferi içinde kullanılabilir. Şekil 11.7, 8155'in programlarınıza konuşma kabiliyeti vermek için bir Votrax SC-01 konuşma sentezleyici entegre devresine nasıl bağlanabileceğini göstermektedir.

SC-01 konuşma üretmek için fonemleri kullanır. Fonemler, kelimelerin parçalarıdır. Fonemleri bağlayarak kelimeleri üretebilirsiniz. SC-01'e gönderilen 6-bitlik ikilik kod, 64 fonemden hangisinin çıkışa gönderileceğini belirler. İlave 2 bit seslendirilen fonemin pitchini belirler. Kelimeleri veya cümleleri üretmek için mikrobilgisayar uygun zamanlama ile bir seri baytı SC-01'e göndermelidir. Fonemler ses vermek için 47 ile 250 ms süre gerektirirler. SC-01 01/R çıkışı, eleman bir sonraki fonemi almaya hazır olduğunda yükseğe gider. SC-01'den gelen bu isteğe taramalı veya kesme tabanlı olarak servis verilebilir. Fakat, istekler arasındaki uzun sürelerden dolayı, kesme tabanlı servis daha mantıklıdır.

Bu işlem için 8155 aşağıdaki şekilde ilk kullanıma hazırlanır. Komut kelimesi bit 0'daki 1 *port A*'yı çıkış olarak ayarlar, *port C* komut kelimesinin 2 ve 3 nolu bitleri 10 veya 01 yapılarak alternatif 3 veya 4 olarak ilk kullanıma hazırlanır ve port A kesme çıkışı, komut kelimesindeki bit 4'ü 1 yaparak yetkilendirilir. Komut kelimesindeki diğer bitler port B ve zamanlayıcının kullanımı ile belirlenir.

Şekil 11.7b, bir 8155'ten SC-01'e onay çıkış veri transferi için zamanlama diyagramını göstermektedir. Güç kaynağı açıldığında SC-01'in alındı / istek ($A/\overline{R}$) çıkışı yüksek olacaktır. TTL seviyelerine kaydıktan sonra bu sinyal 8155'in $\overline{STB}$ girişine bağlanır. Bu girişteki yüksek seviye, 8155'in CPU'ya bir kesme sinyali göndermesine sebep olur. CPU, kesmeye servis verirken *port A*'ya bir baytlık veri yazar. 8155 *port A*'ya yazmak *port C* bit 0'dan gelen kesme çıkışını resetler. CPU'dan gelen $\overline{WR}$ sinyali yüksek olunca, geçerli verinin *port A*'da mevcut olduğunu çevrebirime anlatmak için 8155 tampon -dolu hattını (BF) otomatik olarak yükseğe çeker.

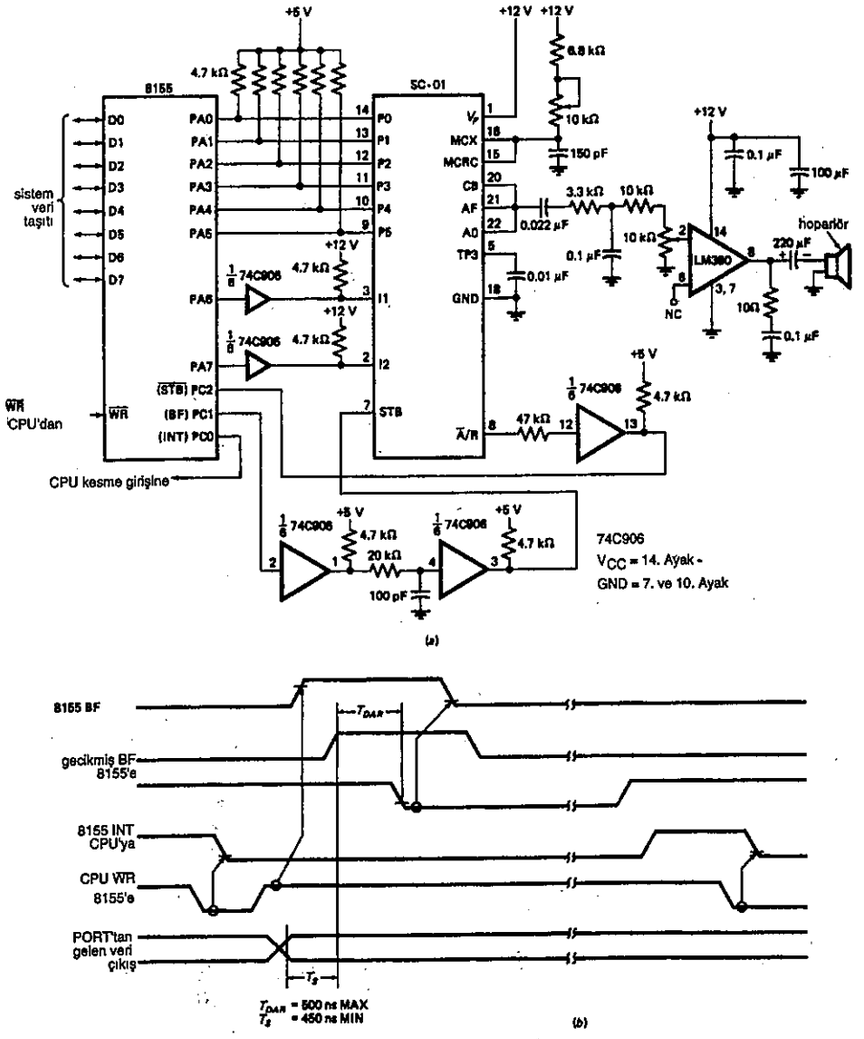
Fakat, *SC-01* bir $\overline{STB}$ sinyali almadan önce verinin, veri girişlerinde en azından 500 ns kalmasını gerektirir. İki 74C906 tamponu strobu gerektiği şekilde geciktirmek için kullanılır. SC-01, ikinci 74C906 tamponundan bir $\overline{STB}$ girişi aldığında, fonemin alındığını belirtmek için $A/\overline{R}$ çıkışı hattını alçağa çeker. $A/\overline{R}$'nın alçak seviyeye gidişi 8155'in BF sinyalini yeniden alçağa düşürmesine sebep olur.

BF sinyalinin düşük seviyeye gidişi, normal olarak bir çevrebirime herhangi bir zamanda bilgisayardan bir baytlık başka bir veri isteyebileceği sinyalidir. Fakat buradaki örnekte, *BF* sinyalinin düşük seviyeye gidişinden çok ikinci 74C906 tamponunun çıkışının düşük seviyeye gidişi, *SC-01*'e bir sonraki fonemi isteyebileceğini anlatır. 8155'in $\overline{STB}$ girişine gelen bu sinyal 8155'in CPU'ya bir kesme sinyali göndermesine sebep olur. Bu durumda bir kesme servis yordamı bir sonraki fonemi 8155'in A portuna yazar ve işlem tarif edildiği gibi devam eder.

SC-01 her kelime için birkaç fonem ister. Örneğin, onaltılık 9F, 42, 63, 58, 5D, 3E, 3E, AA, 7B, 5F, 6A, 3E, 3E, 59, 63, 4C, 58, 6C, 3E, 3E, 3E dizisini göndermek "self-test complete" deyimini üretir. Kendi deyimlerinizi kurmak istiyorsanız, 64 fonemin hepsinin onaltılık kodları bir tablo halinde EK.*D*'de verilmiştir.

MC6821'i ONAYLI VERİ I/O İÇİN KULLANMAK : İLK KULLANIMA HAZIRLAMA

Onaylı giriş ve çıkış için yaygın olarak kullanılan bir başka programlanabilir port MC6821'dir. 6821 iki 8-bitlik porta ve dört kesme girişi veya kontrol hattına sahiptir.



Şekil 11.7 8155 ile kullanılan Votrax SC-01 konuşma sentezleyicisi

***a)* Devre bağlantıları *b)* Zamanlama diyagramları**

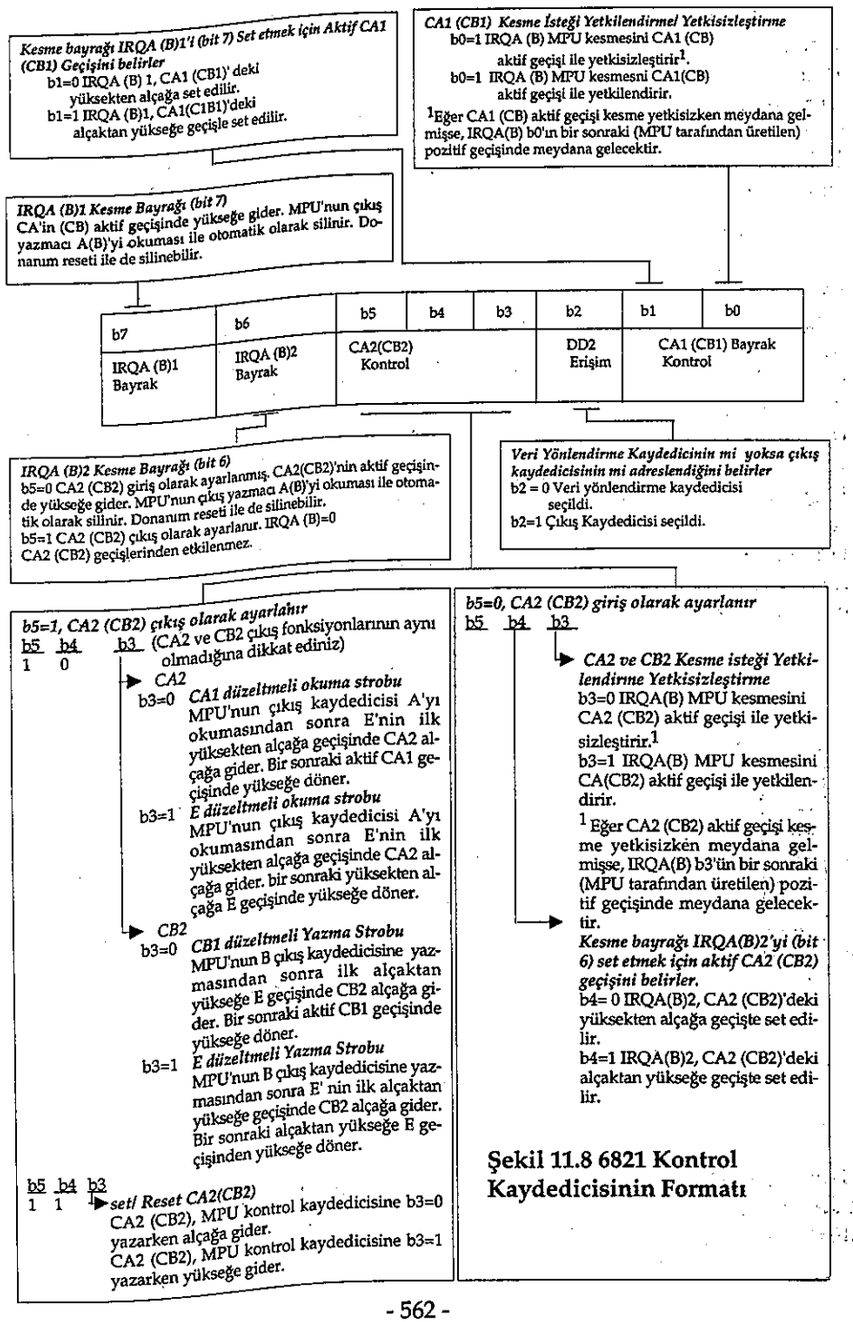
Bölüm 10'dan 6821 üzerindeki bir portu giriş veya çıkış olarak ayarlamanın üç-adımlık bir işlem olduğunu hatırlayınız. İlk olarak entegreyi resetleyiniz ya da her kontrol kaydedicisine bit 2'nin 0 olduğu kontrol kelimesini gönderiniz. Bu veri yönlendirme kaydedicilerine erişimi sağlar. İkinci olarak

her *veri yönlendirme kaydedicisine (DDR)* bir veri yönlendirme kelimesi gönderiniz. DDR içindeki herhangi bir bite 1 göndermek, bite karşı gelen port hattını çıkış yapar. DDR içindeki herhangi bir bite 0 göndermek, o bite karşı gelen port hattını giriş yapar. Üçüncü olarak bit 2'nin 1 olduğu kontrol kelimesi her kontrol kaydedicisine gönderilir böylece CPU iki portada erişebilir.

6821 üzerindeki dört kesme girişi ve kontrol hatları *CA1, CA2, CB1* ve *CB2* olarak adlandırılır. *CA1* ve *CB1* kesme girişleri olarak programlanabilir. *CA2* ve *CB2*, kesme girişleri veya kontrol sinyal çıkışları olarak programlanabilir. Şekil 11.8, 6821 kontrol kaydedicisinin formatını göstermektedir. *A* kaydedicisini kontrol etmek için gönderilen kontrol kelimesi *CA* hatlarının fonksiyonlarını belirler. B kaydedicisini kontrol etmek için gönderilen kontrol kelimesi de *CB* hatlarının fonksiyonlarını belirler. Basitlik için örnek olarak *CA* hatlarını ve *A* kontrol *kaydedicisini (CRA)* kullanacağız. Cihazın *B* kısmı da hemen hemen aynıdır.

CRA bit 0, ise, *CA* hattı kesme girişi olarak yetkilendirilir. *CRA* bit 1, *CA1*'e gelen yükselen kenar sinyalinin mi yoksa düşen kenar sinyalinin mi bir kesme üreteceğini belirler. *CA1* girişi bir aktif kenar sinyali aldığında 6821 iki şey yapar. İlk olarak *A* kontrol kaydedici içindeki bit 7'yi kesme bayrağı olarak ayarlar. İkinci olarak 6821 $\overline{\text{IRQA}}$ çıkışına alçak bir sinyal gönderir. Bu genellikle *CPU*'nun bir kesme girişine bağlıdır.

CA2'de bir kesme girişi olarak programlanabilir. Eğer kontrol kelimesinin 5 nolu biti 0 ise, bit 3'teki bir 1 *CA2*'yi bir kesme girişi olarak yetkilendirir ve bit 4, *CA2* giriş sinyalinin hangi kenarının bir kesmeye sebep olacağına karar verir. *CA2* girişi bir aktif sinyal aldığında 6821 iki şey yapar. İlk olarak kontrol kaydedicisinin 6 nolu bitini kesme bayrağı olarak ayarlar. İkinci olarak, $\overline{\text{IRQA}}$ çıkışına düşük seviye gönderir. Hem *CA1* hem de *CA2* kesme girişleri olarak kullanıldığında, *CPU*'nun *CA1* girişinin mi yoksa *CA2* girişinin mi kesmeye sebep olduğunu nasıl bildiği sorusu ortaya çıkar. Bunun cevabı kesme servis yordamının 6821 kontrol/statü kaydedicisini okuması ve bit 6 ile bit 7'yi kontrol etmesi gerektiğidir. Bit 7'deki bir *1 CA1*'in kesme ürettiğini belirtir, ve bit 6'daki bir *1 CA2*'nin kesme ürettiğini belirtir. Bu durumda 6821 iki kesme sinyalini *CPU* üzerindeki bir kesme girişine yönlendirmiştir. *CB1* ve *CB2*'de kesme girişleri olarak programlanabilirler. 6821 $\overline{\text{IRQA}}$ ve $\overline{\text{IRQB}}$ çıkışları açık-akaçlı olduklarından aynı *CPU* kesme girişine bağlanabilirler. Bu durumda 6821 dört-girişli kesme hunisi gibi davranır.

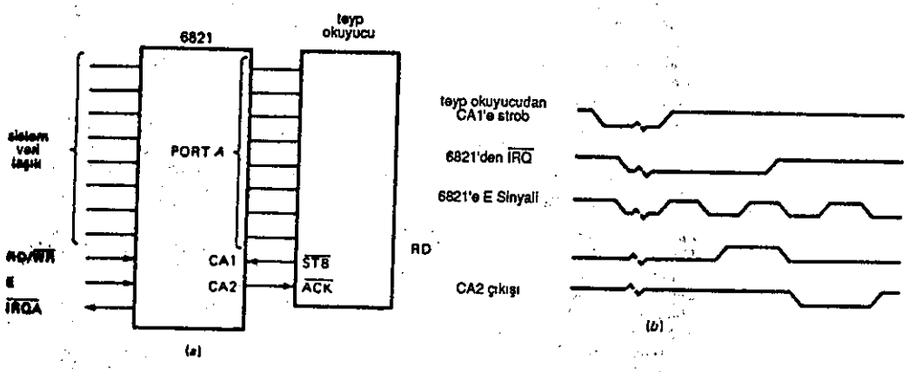


Şekil 11.8 6821 Kontrol Kaydedicisinin Formatı

CA2 hattı, kontrol kelimesindeki bit 5'i 1 yaparak çıkış olarak programlanabilir. Eğer bit 4'te 1 ise, kontrol kelimesi kontrol kaydedicisine gönderildiği zaman bit 3'te olan mantık seviyesi *CA2* hattında çıkış olarak gözükür. Bu durumda *CA2* hattı bir tek-bitlik çıkış portu olarak çalışır. Eğer kontrol kelimesinde bit 5 1 ve bit 4 0 ise, *CA2* daha önceki bölümde 8155 için anlatıldığı gibi bir otomatik onay sinyali olarak çalışır. *CA2* yalnızca bir okuma strobu (giriş) ve *CB2* ise yalnızca bir yazma strobu (çıkış) olarak çalışır.

6821 ONAYLI GİRİŞ ÖRNEĞİ: TEYP OKUYUCU

Şekil 11.9a, onaylı (stroblu) giriş işlemi için bir 6821'in yüksek-hızlı teyp okuyucuya nasıl bağlandığını göstermektedir. Şekil 11.9b bir veri transferi sırasında oluşan sinyal sırasını göstermektedir. 6821 *CRA*, *CA1* yetkilendirilmiş ve düşen kenarda aktif, ve *CA2*, E düzeltmeli okuma strobunun çıkışı olacak şekilde programlanır. Teyp okuyucu *CA1* girişine bir $\overline{STB}$ sinyali ve port *A*'ya bir baytlık veri gönderir. Daha sonra 6821, CPU'ya bir kesme isteği ($\overline{IRQA}$) gönderir. *CPU*, 6821'in A portundan veri okuyan bir kesme servis yordamını yerine getirir. Yüksek seviyeye giden *RD* sinyali, 6821 'den gelen $\overline{IRQA}$ sinyalinin yükseğe dönmesine sebep olur. *CA2* çıkışı, okumadan sonraki E sinyalinin düşen kenarını takiben düşük seviyeye gider. Programlandığı gibi *CA2* sinyali, *E* sinyalinin bir sonraki yüksekten-düşüğe kenarından hemen sonra yüksek seviyeye döner. *CA2*'nin yükseğe dönüşü teyp okuyucuya bir sonraki veri baytını gönderebileceği sinyalidir.



Şekil 11.9 6821 ile onaylı veri transferi *a)* Giriş veri transferi için 6821'e bağlanmış bir teyp okuyucu *b)* Sinyal sırası: *CA1* kesme girişi, *CA2* *E* düzeltmeli okuma strobu

6821 ONAYLI ÇIKIŞI

Port B'den çıkış verisi transferi için kontrol kaydedici *B* benzer sinyaller sırası üretmek üzere programlanabilir. Şekil 11.10, 6821'in 8004 H'den 8007H'ye kadar olan adreslerde aşağıdaki şekilde ilk kullanıma hazırlanması için gerekli deyimleri göstermektedir. *Port A* stroblu giriş, *CA1* yükselen kenarda aktif, ve *CA2 E* düzeltmeli okuma strobu; *Port B* stroblu çıkış, *CB1* düşen kenarda aktif, ve *CB2 CB1* düzeltmeli yazma strobu olarak ayarlanır.

```
LDAA   #$00        ; DDRA ERİŞİMİ İÇİN KONTROL KELİMESİ
STAA   $8005       ; DDRA'YA ERİŞME
STAA   $8007       ; DDRB'YE ERİŞME
STAA   $8004       ; DDRA'YA 00H, HEPSİ GİRİŞ
LDAA   $00101111   ; PORT A İÇİN KONTROL KELİMESİ
STAA   $8005       ; KONTROL KAYDEDİCİYİ A'YA GÖNDER
LDAA   #$FF        ; PORT B İÇİN DDR KELİMESİ-HEPSİ ÇIKIŞ
STAA   $8006       ; DDRB'YE GÖNDER
LDAA   #00100101   ; PORT B İÇİN KONTROL KELİMESİ
STAA   $8007       ; KONTROL KAYDEDİCİYİ B'YE GÖNDER
```

Şekil 11.10 6821 *A portunu* stroblu giriş, *B portunu* stroblu çıkış olarak kullanıma hazırlayan MC6800 yordamı

KLAVYELERİ MİKROBİLGİSAYARLARA ARABİRİMLERLE BAĞLAMA

Programların, verilerin ve komutların mikrobilgisayarlara girilmesinin yaygın bir metodu klavye kullanmaktır. Klavye, şekil 9.8a'da SDK-85 üzerinde gösterilen basit bir onaltılık tip ya da daktilo tip klavye olabilir. Her iki durumda da klavye, Bölüm 4'te tartışıldığı gibi satır ve sütunlar şeklinde düzenlenmiş anahtarlar matrisidir. Klavyeden anlamlı bir veri almak üç işin yerine getirilmesini gerektirir.

1. Kapanan bir tuşu algılama
2. Titreşimini önleme
3. Kodlama

Hatırlayacağınız gibi, bir tuşa basılması bir satır ile sütunu kısa devre yapar. Bir tuşun kapandığını algılamanın genel yöntemi; her seferinde bir satıra düşük sinyal koymak ve normalde yüksek olan sütunların düşük satır ile kısa devre yapılarak düşüğe çekilip çekilmediğini taramaktır. Bir tuşun kapandığını algılamanın daha hızlı bir metodu; bütün satırlara aynı zamanda düşük seviye koymak ve sütunları düşük seviye için taramaktır. Eğer kapalı bir tuş bulunursa, tuşun yerleştirildiği satırı bulmak için satırlar sırayla taranır. Satır bulunduğunda titreşim önlemenin başlaması için satır tarama durdurulur.

Titreşim giderme, 10 ile 20 saniye bekledikten sonra tuşun hâlâ basılı olup olmadığını kontrol etmekle yapılır. Eğer tuş hâlâ basılı ise, tuşa basma geçerli kabul edilir.

Satır ve sütun kodunun onaltılık yada ASCII gibi daha standart bir koda çevrilmesi bir ROM ya da program yordamı ile yapılabilir.

Bir klavye bir bilgisayara arabirimle bağlandığında, harici donanım veya dahili program yordamlarıyla üç iş yapılabilir.

DONANIM YAKLAŞIMI

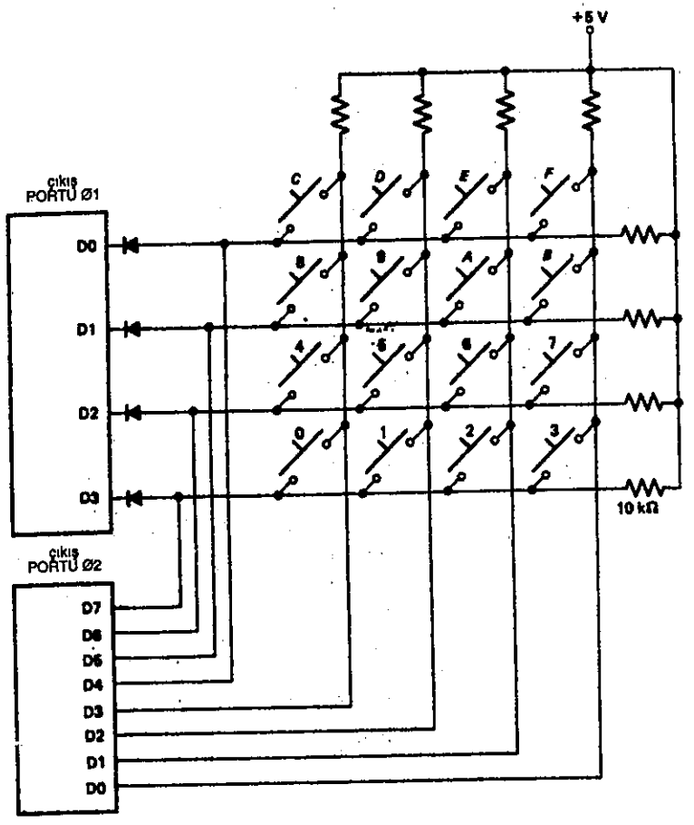
Donanım yaklaşımında, klavye matrisinin taranması, titreşimin giderilmesi ve kodlama tamamen harici cihaz tarafından yapılır. Harici klavye kodlayıcı bir veri hazır sinyali ya da geçerli tuş basılma sinyali gönderdiği zaman mikroişlemci kodlanmış karakteri alır. Bir tam *MOS LSI* klavye kodlayıcı örneği General Instruments *AY-5-2376*'dır. Bu cihaz daktilo tipi bir klavyeyi arabirimle bağlamak için gerekli bütün devrelere sahiptir. Bir tarama osilatörü, titreşim giderici zamanlayıcısı, ve kodlama *ROM*'u içerir. Karakterler, bit 8'in tek yada çift eşlik biti olarak seçilebildiği *ASCII* karakterler şeklinde gönderilir. Sekiz paralel çıkış hattında geçerli bir karakter olduğunda *AY-5-2736* bir pozitif tuş basıldı darbesi gönderir. Bir *CPU*, bu tuşbasıldı darbesini algılayabilmek için tarama ya da kesme modunu kullanır ve karakteri alır.

AY-5-2736'nın bir başka özelliği *iki-tuşun üstüste yuvarlanmasıdır*. Eğer iki tuş birbirine çok yakın zamanlarda basılırsa, ilk basılanın kodu gönderilir. Bu tuş bırakıldığında ikincisinin kodu gönderilir. Bu özellik olmadan iki tuşa basılması yanlış ya da uygun olmayan bir çıkış kodu verecektir.

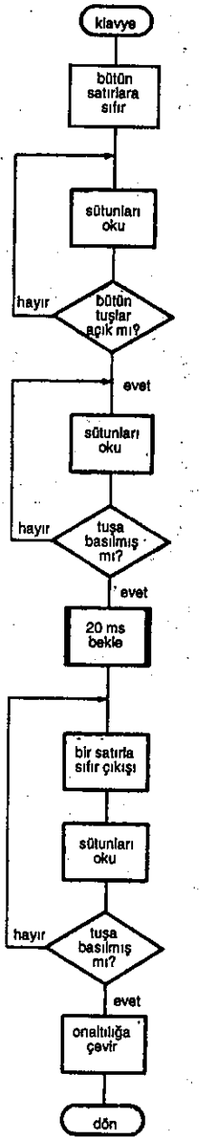
Donanım yaklaşımının avantajı, işlemcinin bu üç işi yapmak için meşgul edilmesidir. Fakat çoğu uygulamalar için işlemcinin birçok boş zamanı olacaktır. Bu durumlarda, donanım kodlayıcı *PC* kartlarının yer kaplamalarından ve maliyetlerinden kurtulmak için yazılım yaklaşımı kullanılır.

YAZILIM YAKLAŞIMI

Yazılım yaklaşımı temelde üç işi program yordamları ile yapmaktan oluşur. Şekil 11.11'de gösterildiği gibi, klavye matrisinin satır hatları bir çıkış portuna ve sütun hatları bir giriş portuna bağlanırlar. Şekil 11.12 bu klavyeyi tarayacak bir programın basit bir akış diyagramını göstermektedir.



Şekil 11.11 Bir klavye matrisinin I/O portlarına bağlanması



Şekil 11.12 Klavye tarama yordamı için akış diyagramı

Dört satır hattının herbirine sıfır çıkışı yapılır. Giriş portundaki dört sütun hattı okunur ve 0 için kontrol edilir. 0, tuşun basıldığını belirtir. Program bir tuşun basıldığını kabul etmeden önce, bütün tuşları açık buluncaya kadar döner. Diğer bir deyişle, son basılan tuş serbest bırakılmadan yeni bir tuşa basılamaz. Bu iki tuşun üstüste yuvarlanması yerine kullanılır. Klavye tamamen açık olarak bulunduğunda, program basılmış bir tuş bulmak için döner. Eğer basılı bir tuş bulunursa, basılan tuşun titremeyi durdurmasını beklemek için program 20m saniyelik bir gecikme yordamını çağırır.

Bir sonraki problem hangi tuşun basıldığını bulmaktır. Bu, birinci satıra 0 diğer satırlara 1 çıkışı yaparak gerçekleştirilir. Daha sonra sütunlar okunur ve 0 için kontrol edilir. Eğer 0 bulunmazsa, basılan tuş bu satırda değildir, bu yüzden bir sonraki satıra 0 uygulanır ve sütunlar yeniden kontrol edilir. Eğer bir sütunda 0 bulunursa, bu durumda giriş portundan okunacak 8-bitlik kelime, basılan tuşun 4-bitlik satır kodunu ve 4-bitlik sütun kodunu içerir. Bu 8-bitlik kod, bir başvuru tablosu kullanarak istenen 4-bitlik onaltılık koda çevrilir.

Şekil 11.13, bu klavye tarama yordamı için bir 8080A/8085A çevirici dil programını göstermektedir. Programın çoğunda akış diyagramı aynen takip edilir, fakat birkaç noktada açıklamaya ihtiyaç vardır. *XRA A* deyimi hem akümülatörü hem de elde bayrağı flip-flopunu sıfırlamak için kullanışlı bir deyimdir. Bunu, akümülatörü kendisi ile *XOR*'layarak ($A \oplus A = 0$) yapar. *8080A/8085A* eldeyi sıfırlamak için özel bir deyime sahip değildir, fakat *ORA A* akümülatör içeriğini değiştirmeden eldeyi sıfırlar. Eldedeki 0'ı ardışıl satırlara döndürmek için *RAL* deyiminin nasıl kullanıldığına dikkat ediniz.

20 ms gecikme yordamı genişletilmiş *BC* kaydedicisini kullanır ve yüklenmiş bir sayıyı 0'a azaltır. *DCX* deyimi bayrakları etkilemediğinden, 0 testi *B* ile *C*'nin VEYA lanması ile yapılır. *B* ve *C*'nin her ikiside tamamen 0 ise sonuç 0 olur ve sıfır bayrağı set edilir.


```
; KYBRD
; BU YORDAM 16 ANAHTARLI TUŞ TAKIMINI TARAR VE KODUNU ÇÖZER
; SATIR HATLARI ÇIKIŞ PORTU 01, D0-D3'E BAĞLANMIŞTIR. SATIR VE
; SÜTUN GİRİŞLERİ PORT 02, D0-D7'DE GİRİLİR
; KULLANILAN KAYDEDİCİLER: HL, BC, A
; ÇAĞRILAN YORDAMLAR : WT20MS

KYBRD:  PUSH   H        ; KULLANILAN KAYDEDİCİLERİN
        PUSH   B        ; İÇERİKLERİNİ SAKLA
        XRA    A        ; AKÜMÜLATÖR VE ELDEYİ SIFIRLA
        OUT    01       ; KLAVYENİN BÜTÜN SATIRLARINA
                        ; SIFIR GÖNDER
KYOPN:  IN     02       ; SÜTUNLARI OKU
        CPI    0FH      ; SÜTUNDAKİ SIFIRLARI ARA
        JNZ    KYOPN    ; BİR TUŞ KAPALIYSA, HEPSİ AÇIK
                        ; OLUNCAYA KADAR DÖN
KYCLOS: IN     02       ; BÜTÜN TUŞLAR AÇIKSA, KAPANAN
        CPI    0FH      ; TUŞ İÇİN KONTROL ET
        JZ     KYCLOS   ; TUŞ BASILINCAYA KADAR DÖN
        CALL   WT20MS   ; 20MS GECİKME
        MVI    A, FF    ; AKÜMÜLATÖRÜN HEPSİNİ BİR YAP
        ORA    A        ; ELDEYİ SIFIRLA
NXTROW: RAL             ; SIFIRI BİRİNCİ SATIR POZİSYONUNA DÖNDÜR
        MOV    B,A      ; SATIR MASKESİNİ SAKLA
        OUT    01       ; SATIRLARA ÇIKIŞ
        IN     02       ; SATIR VE SÜTÜNLARI OKU
        MOV    C,A      ; SATIR VE SÜTUN KODUNU SAKLA
        ANI    0FH      ; SATIR KODUNU MASKELE
        CPI    0FH      ; SÜTUNLARI BİR DÜŞÜK SEVİYE İÇİN KONTROL ET
        JNZ    DECODE   ;EĞER DÜŞÜK SEVİYE VARSA SATIR BULUNDU
                        ; C SATIR VE SÜTUN KODUNA SAHİP
        STC
        MOV    A,B      ; SATIR MASKESİNİ YENİDEN YÜKLE
        JMP    NXTROW   ; BİR SONRAKİ SATIRI DENE
DECODE: LXI    H,TBLTOP ; ÇEVİRME TABLOSUNUN BAŞLANGIÇ
                        ; ADRESİNİ YÜKLE
        MVI    B,0FH    ; KARAKTER SAYICIYI AYARLA
        MOV    A,C      ; SATIR VE SÜTUN KODUNU AKÜMÜLATÖRE TAŞI
NXTBYT: CMP    M        ; TABLO ELEMANI İLE EŞİTLİK İÇİN KONTROL ET
        JZ     DONE     ; EŞİTLİK BULUNDUYSA, TAMAM
        INX    H        ; HL'Yİ BİR SONRAKİ TABLO ELEMANINI
                        ; GÖSTERMEK İÇİN ARTTIR
        DCR    B        ; KARAKTER SAYICIYI AZALT
        JP     NXTBYT   ; KARAKTER SAYICI HÂLÂ POZİTİF
                        ; EŞİTLİK İÇİN BİR SONRAKİ BAYTI DENE
ERROR:  STC             ; KARAKTERİN BULUNMADIĞINI BELİRTMEK İÇİN
        JMP    EXIT     ; ELDEYİ SET ET VE ÇIK
DONE:   MOVE   A,B      ; ONALTILIK KODU AKÜMÜLATÖRE TAŞI
        ORA    A        ; ELDEYİ SIFIRLA
        POP    B        ; KAYDEDİCİLERİ YENİDEN YÜKLE
```

```
        POP     H
        RET                     ;GERİ DÖN
WT20MS: LXI     B,09C4H         ; 16-BİT SAYICIYI YÜKLE
LOOP:   DCX     B               ; AZALT
        MOV     A,B
        ORA     C               ; SIFIR İÇİN KONTROL ET
        JNZ     LOOP            ; B VE C İÇİNDE SIFIR OLUNCAYA KADAR DÖN
        RET                     ; GERİ DÖN
TBLTOP: F = 1110 1110           7 = 1011 1110
        E = 1110 1101           6 = 1011 1101
        D = 1110 1011           5 = 1011 1011
        C = 1110 0111           4 = 1011 0111
        B = 1101 1110           3 = 0111 1110
        A = 1101 1101           2 = 0111 1101
        9 = 1101 1011           1 = 0111 1011
        8 = 1101 0111           0 = 0111 0111
```

Şekil 11.13 Klavye tarama yordamı için 8085A çevirici dil programı

Bu programın belkide en ilginç kısmı 8-bitlik sütun ve satır kodunu 4-bitlik onaltılık koda çeviren yordamdır. Sütun ve satırların 16 muhtemel kombinasyonu Şekil 11.13'te gösterildiği gibi tablo şeklinde 16 peşpeşe bellek adresine yerleştirilir. *F* tuşunun 8-bitlik kodu ilk adrese, *E* tuşunun kodu bir sonraki adrese ve diğer 14 tuşun kodlarıda takip eden adreslere yerleştirilmiştir. *B* kaydedicisine bellekteki tablonun ilk girişinin onaltılık koduna karşı gelen 0F yüklenir. 8-bitlik sütun-satır kodu *C*den akümülatöre taşınır. Daha sonra bu HL tarafından gösterilen bellekteki ilk tablo adresinin değeri ile karşılaştırılır. Eğer iki kod eşitse, sıfır bayrağı set edilir ve yordamdan çıkılır.

Eğer akümülatör içindeki sütun-satır kodu tablo değeri ile aynı değilse, HL bir arttırılır, *B* bir azaltılır ve bir sonraki tablo değeri akümülatörle karşılaştırılır. Program bir eşitlik bulununcaya kadar tablo boyunca hareket eder. Bir eşitlik bulunduğunda, *B* basılan tuş için doğru onaltılık kodu içerecektir. Eğer tablo içinde eşitlik bulunmazsa, hata olduğunu belirtmek için elde set edilir ve tuş tarama yordamından çıkılır.

Bellek içindeki değerlerin ilk değer 0F'ye karşı gelecek şekilde konulduğuna dikkat ediniz. Bu B'nın 0'a doğru azaltılması ve hata bulunması için işaret bayrağının kullanılması için yapılmıştır. Eğer *B* negatif bir sayıya azalır ve eşitlik bulunmazsa, bu bir hata olduğunu belirtir. *B* pozitif olduğu süre içinde JP deyimi programı, tabloda bir eşitlik araması için geri döndürür.

Bir klavyeyi, bir mikroişlemciye arabirimlerle bağlamak için gösterilen yazılım yaklaşımı kolaylıkla daktilo-tipi klavyelerle çalışabilecek şekilde genişletilebilir; fakat onaltılık tip basitliği nedeniyle burada gösterilmiştir. Bu metodun dezavantajları işlemci zamanını meşgul etmesidir.

YEDİ-PARÇALI GÖSTERGELERİN MİKROBİLGİSAYARLARA ARABİRİMLERLE BAĞLANMASI

Mikroişlemci tabanlı aletler, verileri, deyimleri ve adresleri göstermek için genellikle yedi-parçalı göstergeler kullanır. Verimlilik için göstergeler genellikle çoklanırlar. Çoklama yazılım yordamlarıyla ya da donanım elemanlarıyla yapılabilir.

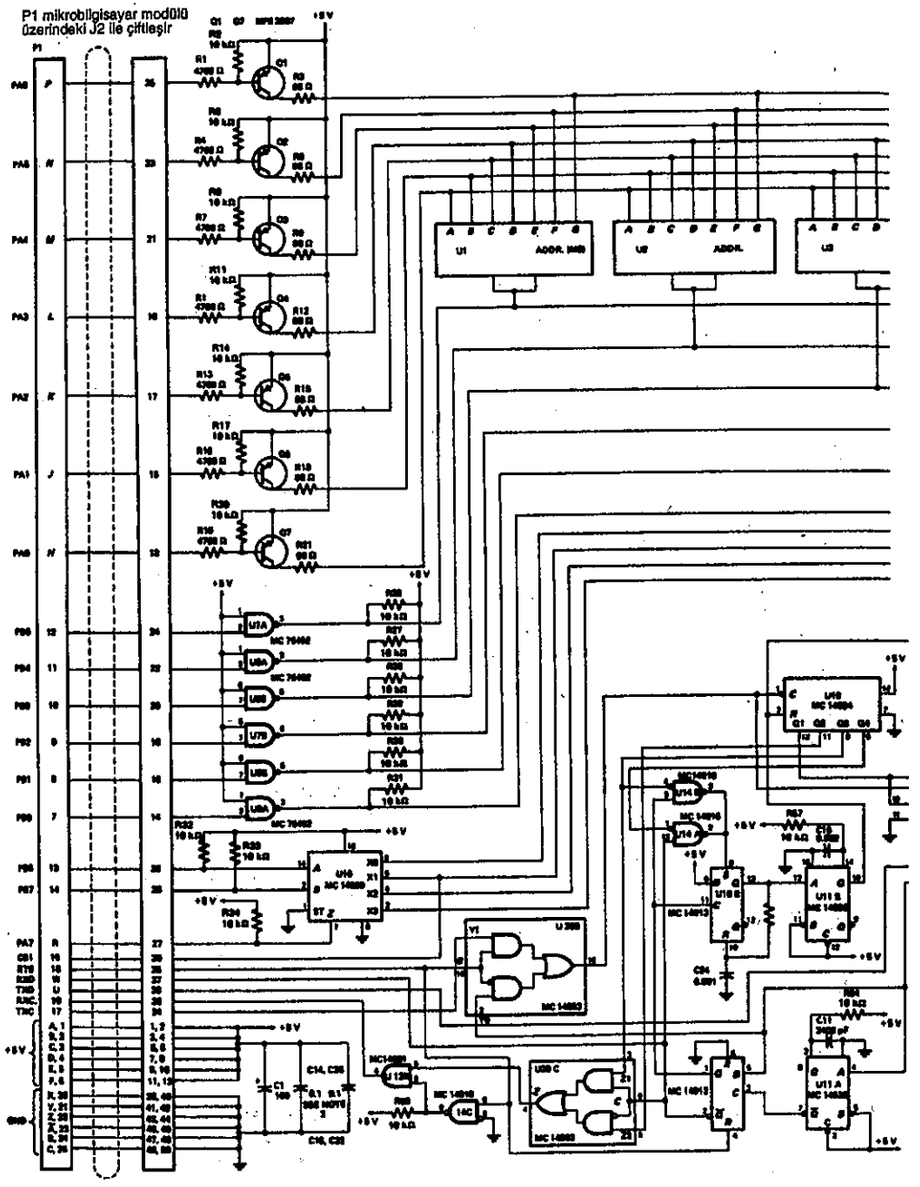
YEDİ-PARÇALI GÖSTERGELERİN YAZILIMLA ÇOKLANMASI

Şekil 11.14 Motorola *MEK 6800D2* mikroişlemci prototip geliştirme kiti için klavye-gösterge modülünün şemasını göstermektedir. Kit, iki baskılı devre kart modülünden oluşmaktadır. İlk kart, şeması Şekil 10.5'te gösterilen *MC6800* tabanlı *CPU* kartıdır. İkinci kart ise şeması burada gösterilen klavye-gösterge modülüdür. Bu kit Bölüm 12'de bir *EPROM* programlayıcı prototipinin temeli olarak kullanılacaktır.

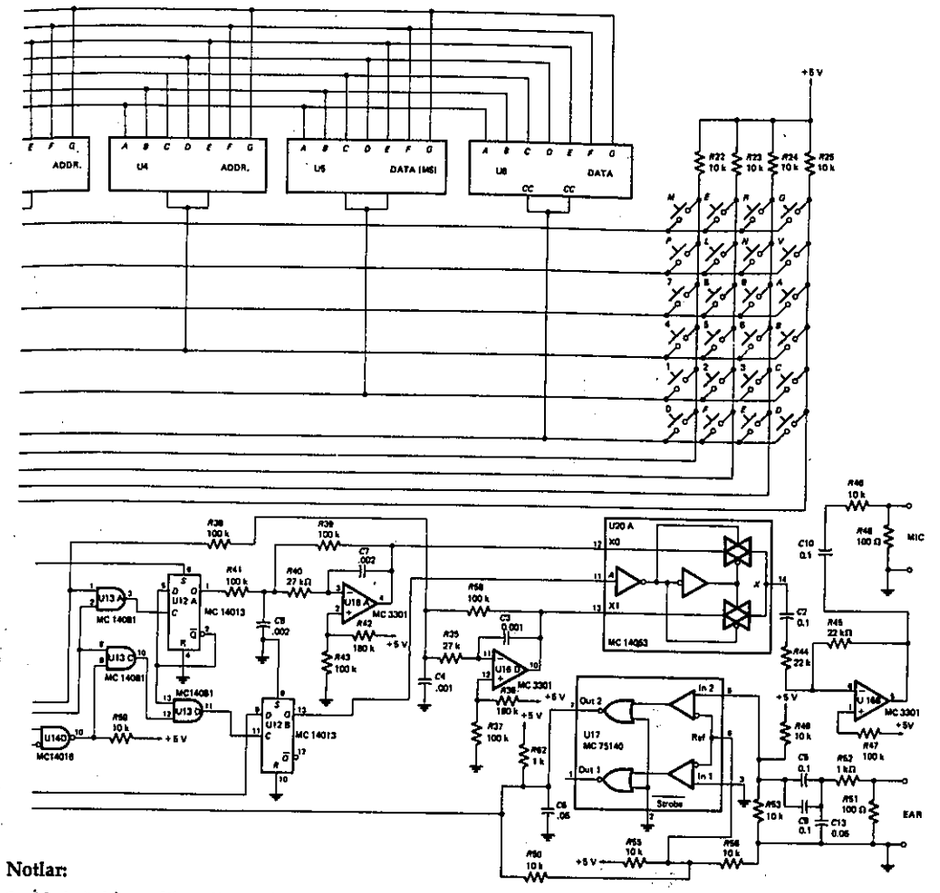
Şekil 11.14'teki şemanın sağ alt köşe kısmı Kansas City standart kaset-teyp arabiriminin devresini göstermektedir. Bu programların pahalı olmayan teyp kasetlerine depolanmasına imkan sağlar (Kansas City formatının tarumı için Bölüm 5'e bakınız.)

Şekil 11.14'teki şemanın geri kalanı klavye ve gösterge devrelerini göstermektedir. Klavye devrelerini geçici olarak ihmal ediniz.

U1-U6, yedi-parçalı, ortak-katodlu *LED* göstergelerdir. *Q1-Q7*, parçaları süren transistörlerdir. Transistörlerin herhangi birinin bazındaki düşük seviye o transistörü iletime geçirir ve parçaya akım gönderir. Her parça sürücü bir *MC6821* çevresel arabirim adaptörünün *(PIA)* bir çıkış hattına bağlıdır.

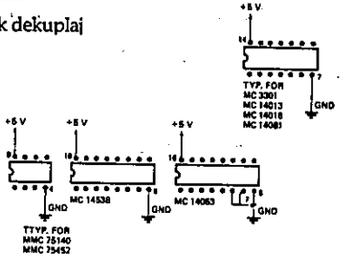


Şekil 11.14 MEK 6800D2 Klavye-gösterge Modülü Şeması (Motorola)



Notlar:

1. Aksi söylenmedikçe direnç değerleri ohm, kondansatör değerleri mikrofarad ve dirençler 1/4 Wattlıktır.
2. *PW* planında her 3 entegre devresi için 0.1μF'lık dekuplaj kondansatörü
3. *Q1-Q7* : *MPS2907*
4. *U1-U6* : Litronix *DL704* ya da eşdeğeri

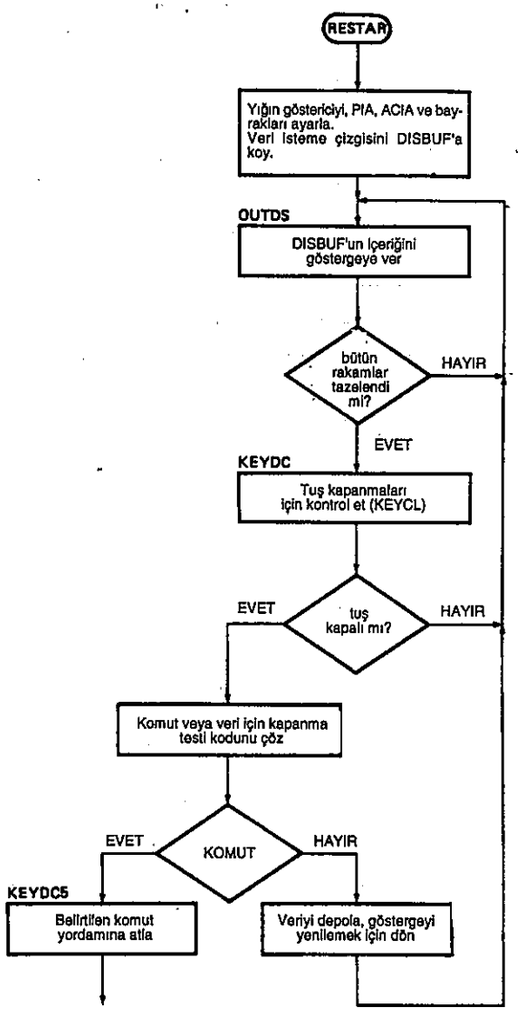


U7-U9 rakam sürücüleri olarak kullanılan *MC75452* yüksek-akım sürücüleridir. Evirici olduklarından, birinin girişindeki yüksek seviye çıkışını alçak yapacaktır. Bu toprağa olan akım yolunu tamamlar ve rakamı aydınlatır. Her rakam-sürücü girişi aynı zamanda bir *6821 PIA*'nın bir çıkış hattına bağlıdır.

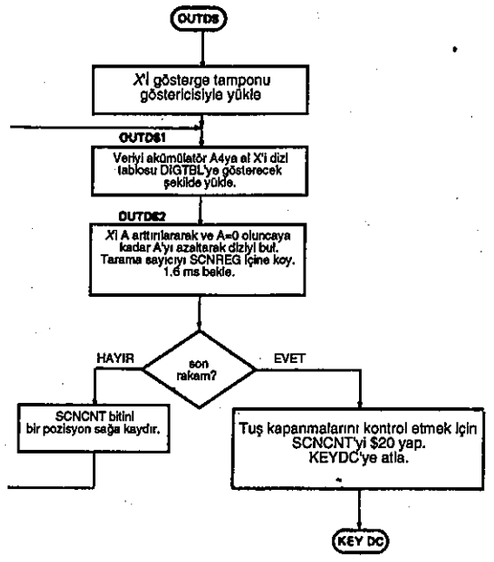
Gösterge çoklama yordamı genelde, şu şekilde çalışır. *U1* rakamı için yedi-parçalı kodu *PA0-PA4* hatları üzerinde gönderilir. Daha sonra, PB0-PB4 alçak seviyede tutulurken *PB5* üzerinden bir yüksek seviye gönderilir. Bu *U7A*'yı iletime geçirir ve bu rakamı ışıklandırmak için *U1 LED* katodlarını alçağa çeker. Yaklaşık 1ms sonra *U2* rakamının parçalı kodu gönderilir ve *U2* rakamı katodları 1ms için alçağa stroblanır. Bu dizi bütün rakamlar tazelenene kadar tekrarlanır. Bütün rakamlar tazelendikten sonra, program basamak 1'i tazelemek üzere geri dönebilir. Fakat, Şekil 11.14'te gösterildiği gibi bir klavyede kullanılıyorsa rakam-strob hatları göstergeyi tazelemek için geri dönmeden önce klavyeyi taramak için kullanılabilirler. Bu MEK6800D2'de kullanılan tekniktir. Şekil 11.15 kullandığı monitor programı için bir genel akış diyagramını göstermektedir.

DISBUF, bellek içinde gösterilecek karakterlerin depolandığı bir tampon depolama yeridir. Program DISBUF'tan bir karakter alır ve onu gösterir. Altı gösterge rakamının hepsi tazelenene kadar karakterler bellekten alınır ve gösterilir. Daha sonra program parçaları siler ve kapalı bir tuşu kontrol etmek için rakam-strob/klavye satır tarama hatlarının hepsine alçak seviye gönderir. Eğer kapalı bir tuş algılanmazsa, program göstergeleri tazelemek için geri döner. Eğer kapalı bir tuş bulunursa, gösterge silik olarak bırakılır ve tuşun titremesi giderilerek kodlanır. Sütun verisi için gerekli giriş sayısını azaltmak için dört sütun hattı bir *MC14539* dört-hattan-bir-hatta çoklayıcısı ile bir hatta çoklanır.

MEK6800D2, Bölüm 12'de bir *EPROM* programlayıcı temel örneğinde kullanıldığından, gösterge yordamı *OUTDS*'nin detaylarını burada tartışmak iyi bir başlangıç için faydalı olacaktır. Şekil 11.16 bir akış diyagramını, ve şekil 11.17 bu akış diyagramı için 6800 çevirici dil programını göstermektedir.



Şekil 11.15 MEK6800D2 klavye-gösterge tarama yordamı için akış diyagramı (Motorola)



Şekil 11.16 MEK6800D2 gösterge yordamı OUTDS için akış diyagramı (Motorola)

Bir başkası tarafından yazılan bir programı anlamak istemek yaygındır. Bunu zorlaştırmak için birçok programda açıklamaları olmayan ve bir anda görülemeyen hileler kullanılır. Şekil 11.17'deki programa benzeyen profesyonelce yazılmış programlar birçok etiket ve sembolik adresler içerirler.

Bu şekildeki bir programa yaklaşmanın ilk adımı bu sembolik adreslerin tanımlarını bulmaktır. Bunlar genellikle programın başında eşitlemeler şeklindedir. Çoğu programcılar sembolik adresleri kısa komutlarla yapmaya çalışırlar. Örneğin, DISBUF altı gösterge rakamının depolandığı tamponun başlangıç adresini belirtir. Eğer etiketler tanımlama olmadan programın sonunda mutlak adreslerine eşitlenmişse, kendiniz bir tanımlar listesi yapmayı deneyiniz. Bir örnek Şekil 11.17'deki programın üst kısmında gösterilmiştir.

Gösterilen *OUTDS* programı, *MEK6800D2*' nin *JBUG* monitor programında bulunan asıl yordama benzemektedir, yalnızca açıklamalar anlaşılabilirlik için genişletilmiştir. Yordam şu şekilde çalışır. *DISBUF*'un başlangıç adresi dizin kaydedicisine yüklenir ve;


```
; SCNREG-  GÖSTERGE KATODLARINA GİDEN ÇIKIŞ PORTLARI ADRESİNİN SEMBOLİK
;          ADI
; SCNCNT-  RAKAM SÜRÜCÜ MASKESİNİN DEPOLANDIĞI ADRES. BU ADRES BAŞLAN-
;          GIÇTA EN SOLDAKİ GÖSTERGEYİ IŞIKLANDIRAN 20H İLE YÜKLENİR.
; DISREG-  GÖSTERGE PARÇALARINA GİDEN ÇIKIŞ PORTLARI ADRESİNİN SEMBOLİK
;          ADI
; XDSBUF-  DİZİN KAYDEDİCİ İÇERİKLERİNİN GEÇİCİ OLARAK DEPOLANDIĞI
;          BELLEK YERİNİN BAŞLANGIÇ ADRESİ
; DIGTBL-  ONALTILIK KARAKTERLERİN HEPSİ İÇİN YEDİ PARÇALI KODLARIN DEPO-
;          LANDIĞI BELLEK ADRESİ
; DISBUF-  GÖSTERİLECEK KARAKTERLERİN DEPOLANDIĞI BELLEK ADRESİ

OUTDS:  LDX   #DISBUF    ; KARAKTER TAMPONUNUN BAŞLANGIÇ AD
                         ; RESİNİ DİZİN KAYDEDİCİSİNE YÜKLE
OUTDS1: LDAA  0, X       ; GÖSTERİLECEK İLK KARAKTERİ A'YA YÜKLE
        INCA             ; KARAKTER DEĞERİNİ ARTTIR
        INX              ; DİZİN KAYDEDİCİSİ GÖSTERİCİYİ ARTTIR
        STX   XDSBUF     ; GÖSTERİCİYİ BELLEKTE SAKLA
        LDX   #DIGTBLI-1 ; YEDİ-PARÇALI KOD TABLOSUNUN BAŞLAN;
                         ; GIÇ ADRESİ-1'İ YÜKLE
OUTDS2: INX              ; DİZİN KAYDEDİCİSİ GÖSTERİCİYİ ARTTIR
        DECA             ; A'YI AZALT
        BNE   OUTDS2     ; SIFIRA EŞİT DEĞİLSE OUTDS2'YE ATLA
        CLR   SCNREG     ; GÖSTERGEYİ SIFIRLA (BÜTÜN RAKAM KATOD;
                         ; LARINA BİR GÖNDER)
        LDAA  0,X        ; X İLE GÖSTERİLEN TABLO ADRESİNDEN YEDİ-;
                         ; PARÇALI KODU AL
        STAA  DISREG     ; KODU, PARÇA SÜRÜCÜLERİNE GÖNDER
        LDAA  SCNCNT     ; RAKAM STROB MASKESİNİ YÜKLE
        STAA  SCNREG     ; MASKEYİ RAKAM SÜRÜCÜLERİNE GÖNDER
        LDX   #$4D       ; 1MS GECİKME İÇİN SABİTİ YÜKLE
        BSR   DLY1       ; 1MS GECİKTİR
        LDX   XDSBUF     ; GİRİLEN KARAKTERİN GÖSTERİCİNİ DISBUF'A
                         ; YENİDEN YÜKLE
        CPX   #DISBUF+6  ; DISBUF İÇİNDEKİ BÜTÜN KARAKTERLER;
                         ; GÖSTERİLDİ Mİ?
        BEQ   OUTDS3     ; EVET İSE BASILAN TUŞU KONTROL İÇİN ÇIK
        LSR   SCNCNT     ; HAYIR, RAKAM MASKESİNİ BİR SONRAKİ RA-
                         ; KAMI GÖSTERMEK İÇİN DÖNDÜR
        BRA   OUTDS1     ; BİR SONRAKİ RAKAMIN KODUNU ALMAK VE;
                         ; GÖSTERMEK İÇİN GERİ DÖN
```

Şekil 11.17 OUTDS gösterge yordamı için MC6800 çevirici dil programı (Motorola).

gösterilecek ilk karakter tamponun ilk adresinden A akümülatörüne yüklenir. Daha sonra karakter değeri ve dizin kaydedicisi arttırılır. (Bunun sebebi daha sonra açıklanacaktır). STX XDSBUF dizin kaydedicisinin içeriğini XDSBUF etiketli bellek adresinden başlayarak belleğe depolar. Daha sonra yedi-parçalı kodları içeren tablonun başlangıç adresinin 1 eksiği dizin kaydedicisine yüklenir. OUTDS2 döngüsü gösterilecek karakteri 0'a azaltır ve dizin kaydedicisini arttırır. Bu, dizin kaydedicisini, karakterin tablo içindeki doğru yedi-parçalı kodunu gösterecek şekilde ayarlar.

CLR SCNREG bütün rakam sürücülerine 0 göndererek göstergeyi siler. Yordam içindeki OUTDS2 bir karakterin doğru yedi-parçalı kodunu şu şekilde üretir. Yedi -parçalı kodlar bellekteki bir tablo içinde depolanmıştır. 0 için kod ilk adreste, 1 için kod bir sonraki adreste, 2 için kod bir sonraki ve diğerleri takip eden adreslerde depolanmıştır. LDX # DIGTBL-1 başlangıç adresi -1'i dizin yazmacına yükler, ve INX bunu tablo içindeki ilk elemanı (0 karakteri için kod) girmek için arttırılır. Fakat *A* içindeki karakter 0 ise, döngü içindeki DECA deyimi bunu 0 yerine FF'ye azaltacaktır. Yedi-parçalı tablosunun doğru karakterlerle sıralanması için bu döngüden önce *A* akümülatörünün içindeki karakterin değeri arttırılmalıdır. Bu, döngünün OUTDS1 bölümü içindeki INC A deyimi ile yapılır. Daha sonra dizin kaydedicisi ile gösterilen tablo adresindeki yedi-parçalı kod akümülatöre alınır.

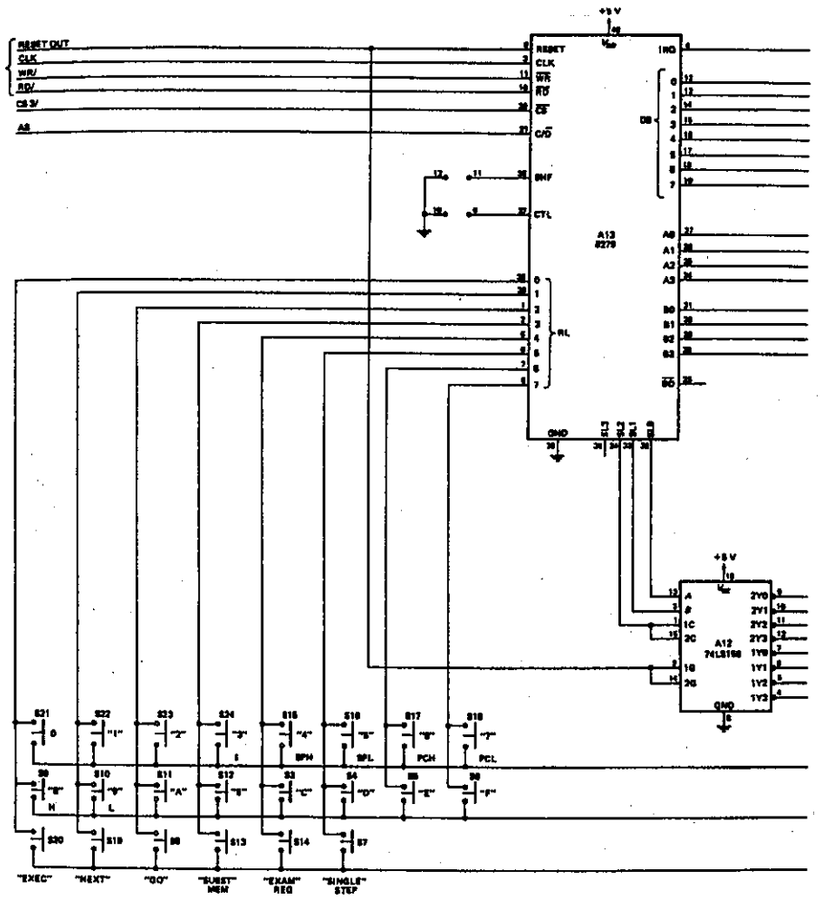
STAA DISREG deyimi, kodu parça sürücülerine gönderir. Fakat rakam sürücüleri iletimde olmadığı için henüz hiçbir rakam aydınlatılmaz. LDAA SCNCNT, SCNCNT etiketli bir bellek yerindeki rakam-strob maskesini akümülatöre yükler. SCNCNT değeri başlangıçta altı göstergenin en sonundakini aydınlatan 20_{16} sayısına ayarlanır. STAA DISREG deyimi rakam-strob maskesini rakam sürücülerine gönderir. Bu stroblanmış rakamı aydınlatır.

1ms gecikmeden sonra gösterilen karakterin DISBUF içindeki gösterici adresi yeniden dizin kaydedicisine yüklenir. X içindeki adres, bütün karakterlerin gösterilip gösterilmediğini görmek için DISBUF'un başlangıç adresi artı 6 ile karşılaştırılır. Eğer hepsi gösterilmişse BEQ, programı OUTDS3 aracığılıyla tuş-kapalı yordamına çıkarır. Eğer karakterler gösterilmemişse SCNCNT içinde depolanmış rakam-strob maskesi içindeki 1, bir sağdaki rakam pozisyonuna kaydırılır. Daha sonra, bir sonraki rakamı göstermek için BRA OUTDS1 programı geri gönderir.

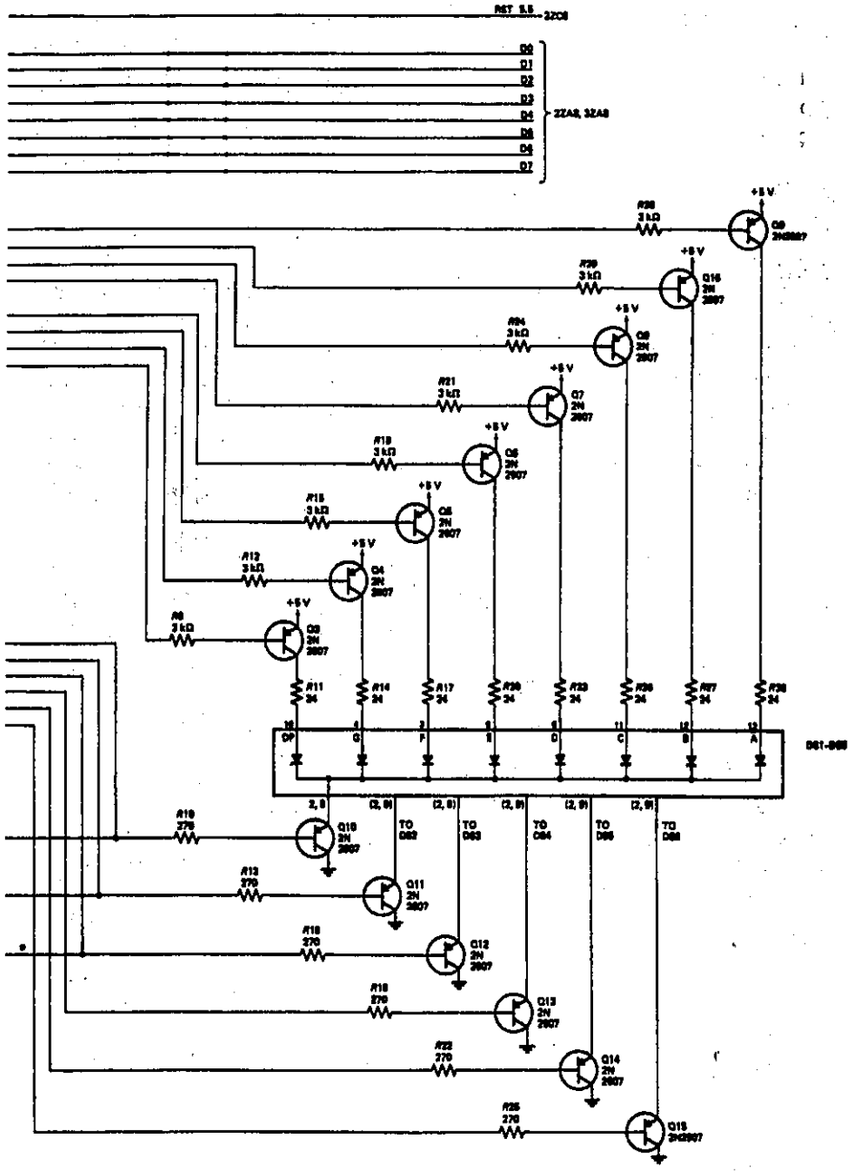
Bu şekildeki bir programdaki her adımı iyice anlayıncaya kadar programı çalışmak iyi bir fikirdir. Bu, genellikle mikro-işlemci tabanlı aletlerde hata giderirken çok gereklidir.

YEDİ -PARÇALI GÖSTERGELERİ ÇOKLAMAK İÇİN DONANIM YAKLAŞIMI

Göstergeleri ve klavyeleri yazılım metodu ile taramanın dezavantajı, bu metodun birçok işlemci zamanını kullanmasıdır. Bir program çalışırken göstergenin silinmesi gerekebilir. Bu tür problemleri olmayan bir başka yaklaşım Intel 8279 gibi bir programlanabilir klavye-gösterge arabirim entegre devresi kullanmaktır. Şekil 11.18, bir 8279'un Intel SDK-85 mikroişlemci temel örnek kartı üzerindeki klavye ve yedi-parçalı göstergelere nasıl bağlandığını göstermektedir.



Şekil 11.18 Intel SDK-85 mikrobilgisayar geliştirme kartının 8279 klavye-gösterge bölümü (Intel Corporation)



Şekil 11.18 (Devam ediyor)

Cihaz temelde şu şekilde çalışır: 8279 ilk kullanıma hazırlandıktan sonra, gösterilecek karakterlerin parça kodları cihaz içindeki bir 16-baytlık kaydediciye yüklenir. Daha sonra 8279 her rakam için parça kodunu ve rakam strobunu sırayla gönderir. Cihaz yüklendikten sonra göstergelerin tazelenmesi için işlemcinin dikkat etmesine gerek yoktur. Şekil 11.18'de 74LS156, 3-hattan-8 hattın-birine-alçak kod çözücü olarak kullanılır. Bu, 8279'un rakam-seçme çıkışları tarafından adreslenen rakamı seçer.

Rakam-strob sürücüleri girişlerindeki alçak seviyelerin dizisi aynı zamanda klavyeyi taramak için kullanılır. Klavyenin sütunları, 8279'un R0-R7 geri dönüş girişleri tarafından okunur. Bir tuşun kapalı olduğu bulunursa, dahili bir titreşim giderme devresi 10 ms bekler ve anahtarın hâlâ kapalı olup olmadığını kontrol eder. Eğer kapalı ise basılan tuşun kodu dahili bir *ilk giren ilk çıkar (FIFO)* belleğe depolanır. *FIFO* bellekte bir karakter olduğunu belirtmek için IRQ çıkışında bir yüksek seviye gönderilir. Genellikle IRQ mikroişlemci üzerindeki RST 5.5 gibi bir kesme girişine bağlanır. Kesmeler yetkilendirilmişse ve bir tuşa basılmışsa işlemci FIFO bellekteki karakteri okuyan bir klavye servis yordamına vektörler. 8279, FIFO bellek taşmaya sebep olmadan sekiz tane okunabilecek karakteri depolayabilir.

Bir 8279 kullanıldığında mikroişlemci cihazı ilk kullanıma hazırlamalı, gösterilecek karakterleri dahili kaydediciye göndermeli, ve bir IRQ ile uyarıldığında klavye karakterlerini okumalıdır.

8279'UN İLK KULLANIMA HAZIRLANMASI

SDK-85, Bölüm 12'de bir temel örnek örneğinde kullanıldığından 8279'un ilk kullanıma hazırlanmasını burada tartışacağız.

İlk olarak Şekil 11.18'deki 8279'un giriş ve çıkışlarına daha yakından bakınız. DB0-DB7 çift yönlü bir veri yoludur. $\overline{\text{Reset}}$, saat, yazma ve $\overline{\text{RD}}$ girişleri karşı gelen sistem hatlarına bağlanmıştır. $\overline{\text{CS}}$ bir yonga seçme girişidir. C/$\overline{\text{D}}$ yüksek olduğunda bir kontrol kelimesinin yazılmakta veya statü kelimesinin okunmakta olduğunu belirtir. C/$\overline{\text{D}}$ alçak ise veri yazılmakta veya okunmaktadır. SDK-85 içinde 8279, kontrol yazma-statü okuma için 1900H adresine ve veri okuma-yazma için 1800H adresine bellek haritalanmıştır. İki 4-bitlik çıkış (*A0-A3* ve *B0-B3*), parçalar için BCD karakterleri göndermekte bağımsız olarak veya gösterildiği gibi yedi-parçalı göndermekte beraber kullanılabilirler. Dört rakam tarama hattın *(SL0-SL3)* kodu 16 taneye kadar olan rakamı taramak için çözülebilir.

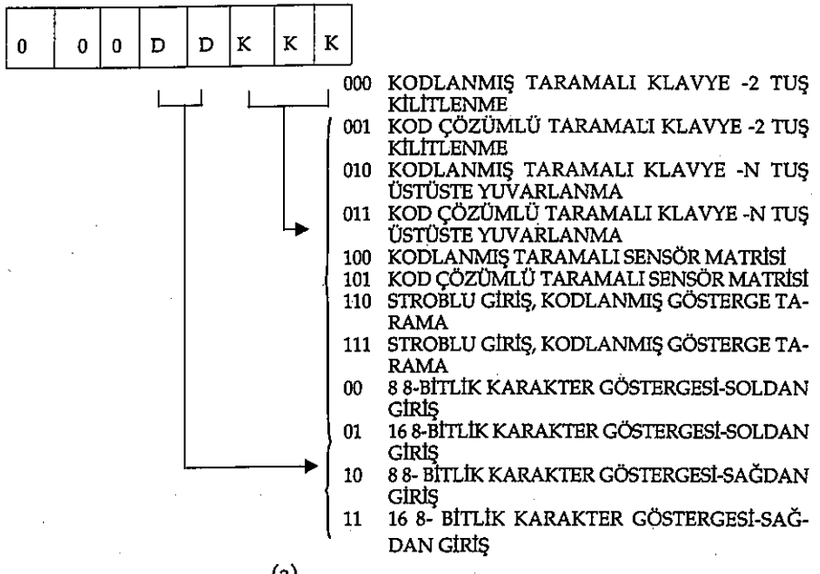
Kodu çözülmüş tarama hatları aynı zamanda klavye satırlarını taramak içinde kullanılabilir. Daha çok hat taranarak daktilo-tipi klavyelerede servis verilebilir. Klavyenin sütunları, 8279'un geri dönüş girişlerine bağlanmıştır. Bu girişler kapanan bir tuş tarafından alçağa çekilene kadar yüksek seviyede

tutulan dahili yukarı çekme dirençlerine sahiptir. Eğer klavye kayma (shift) ve kontrol (control) tuşlarına sahipse, bunlar 8279'un kayma ve kontrol girişlerine bağlanırlar ve böylece bunların etkileri çıkış karakterine dahil edilir. Şekil 11.18'deki tuşlar, bir tuşun kod çıkışı onaltılık 00_{16}-15_{16} aralığında olacak şekilde bağlanmıştır.

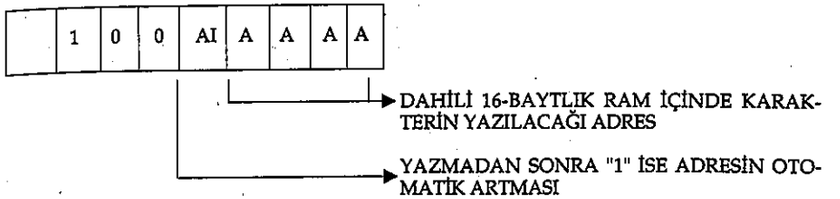
8279'u kullanıma hazırlamak için birçok kontrol kelimesi gönderilmelidir. Şekil 11.18'deki devre için 1900H adresine gönderilirler. Gönderilen ilk kontrol kelimesi klavye-gösterge modunu ayarlar. Şekil 11.19 *a*, bu kelimenin formatını göstermektedir.

Sağdan giriş birçok elektronik hesap makinesi tarafından kullanılan şekildir. İlk karakter en sağdaki basamağa girer. Bir başka karakter girildiğinde ilk karakter bir pozisyon sola kaydırılır. Her yeni karakter girişinde önceki karakterler yeni karaktere yer açmak için topluca sola kaydırılır. Soldan giriş ise karakterler yazıldıkça onların soldan ardarda girişine sebep olur.

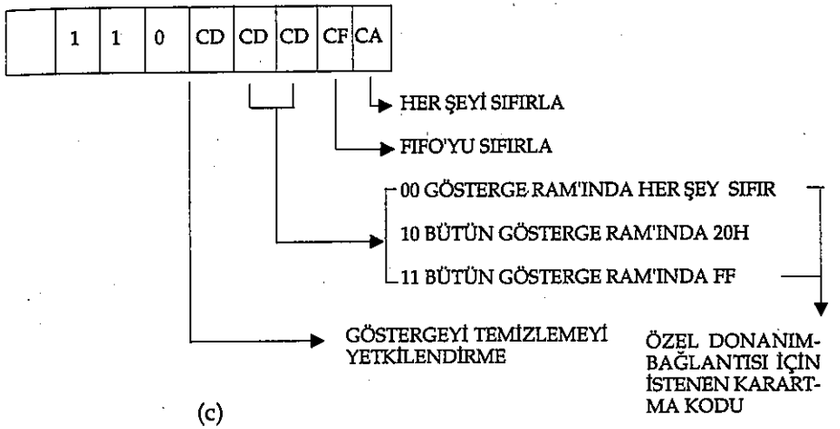
Kodlanmış ve *kodu çözülmüş* klavye modları, dört tarama hattı çıkışları SL0-SL3'ü belirtirler.Kodlanmış modda bu çıkışlar üzerindeki sayı 0'dan 15'e doğru döner. 16 alçak sinyalden birini veya 8 alçak sinyalden birini elde etmek için bir harici kod çözücü kullanılır. Kod çözülmüş modda herhangi bir zamanda, dört SL hattından biri alçak seviyede olur. Küçük klavyeler ve dört yada daha az göstergeler için harici kod çözücü kullanılmasına gerek yoktur.



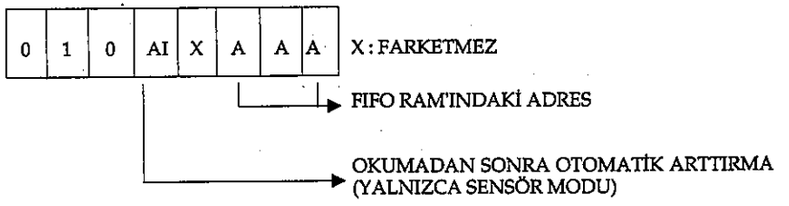
(a)



(b)



(c)



Şekil 11.19 *Intel 8279* Komut Kelimeleri için Klavye-Gösterge Mod Ayar Formatı
***a)* Klavye-gösterge mod ayar formatı**
***b)* Gösterge *RAM*'ına yazma kelimesi**
***c)* Göstergeyi silme ve karartma karakterini ayarlama kelimesi**
***d)* *FIFO RAM*'ını Okuma**

Iki-tuş kilitlenmenin anlamı, eğer iki tuş yaklaşık aynı zamanda basılmışsa daha sonra serbest bırakılan tuşun girilmesidir. *N-tuş üstüste yuvarlanmada* ise, yaklaşık aynı zamanda basılan iki veya daha fazla tuşun hepsi basıldıkları sıraya göre girilecektir.

Muhtemel komut kelimelerinin hepsi burada tartışılamaz, fakat gözönüne alınması gereken önemli bir kelime, dahili 16-baytlık gösterge RAM'ı içinde yazılacak adresi gösteren kelimedir. Bu kelimenin formatı 100AI-*AAAA*'dır, burada *AI* otomatik artışı belirtir (Bakınız Şekil 11.19b). Dört A dahili gösterge *RAM*'ı içindeki 16 yerden birinin 4-bitlik adresini temsil eder. Bir karakteri göstergeye yazmak için önce bir kontrol kelimesi ve daha sonra gösterilecek karakter gönderilir. Bir başka faydalı kontrol kelimesi göstergeyi karartmak için kullanılan karakteri ayarlar. Kontrol kelimesi Şekil 11.19c'de gösterilmiştir. Tamamen 1 ya da FF_{16} çıkışı *SDK-85* göstergesini karartacağından, bunu yapmak için örneğin, CC_{16} kontrol kelimesi gönderilir.

Bir örnek 8279'un ilk kullanıma hazırlanması işlemini açıklığa kavuşturacaktır. Şekil 11.20, 8279'u *SDK-85* kartında kullanmak için kullanıma hazırlanması ile ilgili kısa bir program göstermektedir. Klavye-gösterge komut kelimesi kontrol port adresi 1900H'ye gönderilir. Daha sonra dahili göstericiyi, dahili *RAM*'ın ilk adresine taşıyacak komut kelimesi gönderilir. SDK-85 karartma için tamamen 1'ler gerektirir ve bunun için komut kodu CC_{16}'dır 4 sayısı için gerekli yedi-parçalı kod 99_{16}'dur, ve okuma-yazma port adresi 1800H'ye gönderilerek RAM adresine yazılır. Bu, en soldaki basamakta bir 4 gösterecektir. Bir yazmadan sonra dahili gösterici, dahili RAM içindeki adresi göstermek için arttırılır.

Tuşa basılması sonucu gelen kesmeden sonra *FIFO RAM*'ını okumak için, *FIFO RAM*'ına erişilmelidir. Bunun için *FIFO RAM*'ını gösteren dahili gösterici için bir kontrol kelimesi gönderir. Şekil 11.19'da bu kontrol kelimesinin formatını göstermektedir. 40H'lik bir kontrol kelimesi bunu yapar. Bu kelime kontrol adresine gönderildikten sonra, Şekil 11.20'de gösterildiği gibi veri adresinden bir okuma, *FIFO* bellekteki karakteri okuyacaktır. Gösterge *RAM*'ına yeniden yazabilmek için bir başka gösterge-*RAM*'ına -yaz kontrol kelimesi, kontrol adresine gönderilmelidir.

8279 yaklaşımının ana noktası *CPU*'yu, göstergeyi tazeleme ve klavyeyi tarama işlerinden kurtarmasıdır. Diğer komut kelimeleri ve çalışma modları için *8279* veri kağıtlarına danışınız.

LİKİD KRİSTAL GÖSTERGELERE ARABİRİMLERLE BAĞLANMA

Likid Kristal göstergeler (LCD), bir likid kristal akışkanı iki cam plaka arasına sıkıştırarak elde edilir. Şeffaf bir elektriksel iletken film ya da arka yüzey arasında bir elektrik alanı yaratıldığında, ışığın likid kristal içinden iletimi değiştirilir.

Iki genel LCD tipi vardır: Dinamik yayılma ve *alan etkili.* Dinamik yayılma tipi alanın olduğu yerde molekülleri karıştırır. Bu, bölgeye çarpan ışığı yayar ve karanlık arka planda asitle yakılmış camdan görünen ışığa benzer bir ışık karakteri üretir. Alan etkili tipler, elektrik alanının olduğu yerlerde ışığı yutmak için polarizasyon kullanır. Bu, parlak arka plan üzerinde karanlık karakterler üretir.

Çoğu *LCD'ler çok yavaş açılır* ve *kapanırlar*, böylece kolaylıkla çoklanamazlar. Bu yüzden her parça bağımsız olarak sürülmelidir. LCD'ler düzgün bir dc gerilim uygulandığında çabucak bozulurlar, ve bu yüzden genellikle bir 30-125 Hz'lik ac kare-dalga sürme kullanılır. Kare-dalga sürme ya her parça sürücüye ilave edilir ya da ortak arka yüzeye uygulanır.

RCA CD4054, BCD'den *yedi-parçalıya kod çözmeyi yapar* ve bir LCD basamak için ac sürücüyü sağlar. Intersel ICM 7211 ikilikten onaltılığa kod çözmeyi ve dört LCD rakam için parça sürücüyü sağlar. Kilitler, rakamların mikroişlemciden birer birer yüklenmesini mümkün kılar. 7211 aynı zamanda LCD arka yüzeyi için gerekli ac çıkışa sahiptir.

Çoklanmış *BCD* çıkış, *A/D* çeviricileri ve sayıcılarla kullanılmak üzere tasarlanmasına rağmen 7211 bir çıkış portuna bağlanabilir. Kısa bir program yordamı rakam strobunu ve her rakamın BCD rakam kodunu 7211'e gönderebilir.

```
*İLK KULLANIMA HAZIRLAMA
MVI     A,00H       ;KLAVYE/GÖSTERGE
                    ;MODUNU GÖNDER
STA     1900H       ;SOLDAN GİRİŞ İÇİN, 8
                    ;BASAMAK, KODLANMIŞ 2 TUŞ
                    ;KİLİTLENME
MVI     A,90H       ;DAHİLİ GÖSTERİCİYİ 8279 KARAKTER
                    ;RAMI İÇİNDEKİ
STA     1900H       ;İLK ADRESE
                    ;AYARLAMAK VE OTOMATİK
                    ;ARTIŞ İÇİN KOMUT
                    ;KELİMESİNİ GÖNDER
MVI     A,0CCH      ;GÖSTERGEYİ SİLMEK İÇİN
STA     1900H       ;KULLANILAN KARAKTERİN
                    ;KOMUT KELİMESİNİ GÖNDER
                    ;(SDK-85 İÇİN HEPSİ 1)
```

```
*GÖSTERİLECEK İLK KARAKTERİ GÖNDER
MVI   A,99H      ; EN SOLDAKİ BASAMAKTA
                 ; GÖSTERİLECEK 4
STA   1800H      ; SAYISININ 7-PARÇALI
                 ; KODUNU
                 ; GÖNDER
*FIFO RAM'I OKU
MVI   A,40H      ; FIFO ERİŞİM İÇİN KONTROL
                 ; KELİMESİ
STA   1900H      ; KONTROL ADRESİNE
LDA   1800H      ; FIFO'YU OKU
```

Şekil 11.20 8279'u ilk kullanıma hazırlamak için çevirici dil programı

GÜÇ DEVRELERİ

ZARARLI OLMAYAN İLETİME GEÇME

Mikroişlemciler sıksık akkor lambaları, ısıtıcılar ve motorlar gibi yüksek güçlü cihazları kontrol etmek için kullanılır. Bir felaketi önlemek için bu aletlerin bağlanmaları çok dikkatli yapılmalıdır. Belki de en önemli faktör her devre iletime geçtiği anda zararlı olmayacak bir duruma ayarlamaktır.

Örneğin, Bölüm 2'de tanımlanan yarı iletken rölelerine benzer bir röleye olan girişin bir 8255A üzerindeki bir çıkış hattına bağlandığını varsayalım. Rölenin çıkışı bir sirene bağlıdır, ve rölenin girişindeki mantıksal yüksek onu iletime geçirir. İşlemci ilk olarak çalıştırıldığında, 8255'in bütün port hatları yüksek empedanslı yüzer durumdadır. Bu yüzden yarı iletken röleye olan giriş tahmin edilemez. Bir yüksek duruma kayabilir ve sireni çalıştırabilir. Problem, 8255 ilk kullanıma hazırlanırken katı-hal röle girişini alçakta tutan bir direnç kullanmakla çözülür. Her cihaz için güvenli durum belirlenmelidir.

GÜÇ SÜRÜCÜSÜ

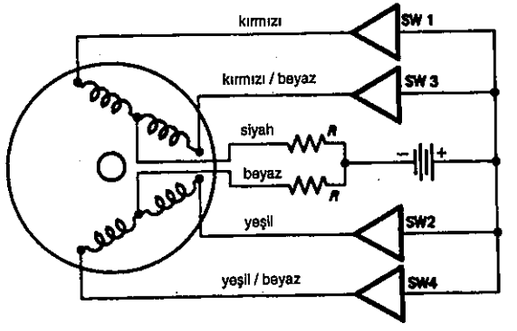
Bu devrelerin gerçek bağlanması, çoğunlukla çıkış portu hatlarının akım sürüşlerini büyük oranda arttırma ve eğer devre ac ile çalışıyorsa onu yalıtma problemidir. Bu fonksiyonlar, Darlington transistörleri, mekanik röleler, veya katı-hal röleleri ile yerine getirilebilir. Mekanik rölelerde ters-kutuplanmış koruyucu diyotları ilave etmeyi unutmayınız.

ADIM MOTORLARI

Robotları ve benzeri makineleri yapmakta kullanılan faydalı bir motor tipi *adım motorudur*. Birçok motorun yaptığı gibi sürekli olarak düzgün bir

şekilde dönmek yerine, darbe geldiğinde sabit bir pozisyondan bir sonraki pozisyona döner ya da adım atar. Yaygın adım büyüklükleri 0.9 ile 30° arasındadır. Adım motorları sayısal motorlar gibi düşünülebilirler.

Bir adım motoru bir pozisyondan bir sonraki pozisyona motor içindeki alanların uygun birleşimi ile adım atar. Şekil 11.21 bir Superior Electric SLO-SYN motorun şemasını ve adımlama sırasını göstermektedir. Eğer bobinler SWI ve SW2 kapalı iken enerjilendirilirlerse, SW1 ve SW4 kapalı durumuna geçmek motor şaftının saat yönünde 1.8°lik bir adım hareket etmesine sebep olur. SW3 ve SW4 kapalı durumuna geçmek robot kolunuzun saat yönünde yeniden 1.8° dönmesine sebep olur. 2 ve 3 anahtarlarına değiştirme bir başka 1.8° ve yeniden 1 ve 2 anahtarlarına dönüş bir başka 1.8°lik saat yönünde dönmeye sebep olacaktır. Bu sıra istenildiği kadar adım üretmek için tekrarlanabilir. Eğer anahtarlama adım sırası ters çevrilirse, şekil 11.21*b*'deki oklarlada belirtildiği gibi, motor saat yönünün tersi yönde döner. Eğer bütün anahtarlar açılırsa, motor şaftı motor içindeki bir mıknatıs tarafından son pozisyonda tutulur. Fakat, bu pozisyonda alan akımları, iletimde iken tutulduğu kadar sabit bir şekilde tutulamaz. Şekil 11.21*a*'daki anahtarlar *JFET* veya *iki kutuplu güç transistörleri* olabilir.



	Anahtar				
Adım	SW4	SW3	SW2	SW1	Saat ibresi yönü
1	0	0	1	1	
2	1	0	0	1	
3	1	1	0	0	
4	0	1	1	0	
1	0	0	1	1	

Saat ibresi yönü tersi

1 = Anahtar Kapalı

(b)

Şekil 11.21 Superior Electric *SLO-SYN* adım motoru *a)* sürücü şeması *b)* Adım sırası

Şekil 11.21*b*'deki adım sonrasına dikkatli bir bakış ilginç bir diziyi ortaya çıkarır. Pozisyon 1'den pozisyon 2'ye hareket etmek için, 1'ler ve iki 0'lar dizisi bir pozisyon sağa döner. Anahtar 4'teki 1 etrafta dönerek anahtar 1'e gelir. Her adım için dizi bir pozisyon sağa döner. Dönüşün yönünü değiştirmek için dizi sola doğru kaydırılır.

Bu sürüş sırası anahtarları, başlangıçta 0011 ile yüklenmiş bir 4-bitlik halka sayıcının çıkışlarına bağlayarak üretilebilir. Halka sayıcı her adım için bir defa darbelenir. Halka sayıcı için bir sola-kayma-sağa kayma kaydedicisi kullanılırsa motor her iki yönde adım atabilir. Bir başka metot da dört anahtarı bir mikroişlemcinin dört çıkış port hattına bağlamaktır. Bu durumda adım sırası basit bir programla elde edilebilir. Eğer anahtarlar dört alçak port hattına bağlanırsa 00110011 dizisi akümülatöre yüklenir. Bunu sağa döndürmek ve çıkışa vermek ise saat yönünün tersi yönde bir adım üretir. Motor her saniyede yalnızca 500 adım gidebildiğinden adımlar arasında kısa bir gecikme bulunmalıdır.

Robotlarda kullanılmalarına ilave olarak adım motorları, bir flopy disk sürücü ünitesindeki kafaların pozisyonlarını ayarlamakta da kullanılır. Aynı zamanda bilgisayar kontrolu altında grafik çizen *x-y* çizicilerinde de kullanılır. Bir adım motoru ve bir kayış *x* yönündeki kalemin pozisyonunu kontrol eder ve bir başka adım motoru ve kayışta y yönündeki kalemin pozisyonunu kontrol eder. Kısacası adım motorları hareketin artışlar halinde gerektiği hemen her yerde faydalıdır.

D/A ÇEVİRİCİLERİ ARABİRİMLERLE BAĞLAMA VE UYGULAMALARI

BİR D/A ÇEVİRİCİYİ BİR MİKROİŞLEMCİYE BAĞLAMA

Çoğu durumlarda bir D/A çeviricinin paralel/girişleri doğrudan bir mikroişlemci çıkış portunun paralel hatlarına bağlanabilir. Porta yazılan bir kelime *D/A* çevirici tarafından orantılı gerilim veya akıma çevrilir.

Fakat, D/A çeviricinin giriş hatları (bitler) portun çıkış hatlarından fazla ise bir problem ortaya çıkar. Örneğin, 12-bitlik bir *D/A* çevirici 8-bitlik bir işlemcinin çıkışına bağlanacaksa, *D/A* çevirici iki çıkış portüna bağlanmalıdır. İşlemci aynı zamanda iki porta çıkış veremeyeceğinden, veri D/A çeviricinin bazı bitlerine diğerlerinden daha önce gelir. Bunun anlamı verinin ikinci kısmını almadan önce *D/A* çıkışı yanlış bir değerden geçecektir. Problem 12 harici kilit kullanarak çözülür. Veri iki porta yazılır ve daha sonra 12 bitin hepsini aynı zamanda *D/A* çeviriciye aktarmak için kilitler stroblanır. Bazı pahalı *D/A* çeviricilerin içinde bu amaç için konulmuş kilitler bulunmaktadır.

BAZI MİKROİŞLEMCİ VE D/A UYGULAMALARI

D/A çeviricilerin bir mikroişlemcinin çıkışında kullanıldıkları uygulamaların sayısı çok büyüktür. Sayısal olarak programlanabilen gerilim veya akım çıkışı istenen hemen her durumda *D/A* çevirici kullanılır.

PROGRAMLANABİLİR GÜÇ KAYNAĞI: *Bir uygulama programlanabilir güç kaynağıdır.* D/A çeviricinin çıkışı güç kaynağının referans girişine bağlanır. D/A çeviricinin çıkışı değiştikçe, güç kaynağının çıkışı da orantılı olarak değişir.

MOTOR-HIZ KONTROLU: *Küçük bir dc motorun hızı, üzerinden akan akım ile orantılıdır. D/A* çeviricinin çıkışı kuvvetli bir akıma çevrilebilir ve motorun hızını kontrol etmek için kullanılabilir.

KONUŞMA SENTEZİ: *D/A çeviricilerin mikroişlemcilerle birlikte ilginç bir uygulaması konuşma* sentezidir. Konuşan hesap makinaları ve sayısal voltmetreler, görmeyenler ve diğerleri için büyük bir yardımdır. Konuşmayı üretmenin bir yolu her kelime için sayısal veriyi ROM içine depolamaktır. Bir işlemci istenen kelimenin verisini okur ve bunu bazı karmaşık filtrelerden geçirir. Filtreleme aslında insanın konuşma yolunu benzetir.

Sayısal filtrelemenin sonucu, analog ses dalga şekillerini üretmek için bir D/A çeviriciden geçirilir. Bunlar kuvvetlendirilir ve bir hoparlörü sürmek için kullanılır. Sözlük içindeki kelime sayısı kullanılan ROM miktarı ile belirlenir. Örneğin, Telesensor Company tarafından yapılan bir hesap makinesinin 64 kelimelik bir sözlüğü vardır. Texas Instruments tarafından üretilen TMC 0280 entegre devresi sinyal üreticiler, sayısal filtreler ve bir D/A çevirici içerir. Konuşan bir heceleme oyunu yapmak için eklenmesi gerekenler yalnızca ROM ve bir mikroişlemcidir.

A/D ÇEVİRİCİLERİ ARABİRİMLE BAĞLAMA VE UYGULAMALAR

A/D ÇEVİRİCİLERİ ARABİRİMLE MİKRO İŞLEMCİLERE BAĞLAMA

Birçok A/D çeviricide her bitlik çözünürlük için bir hatta paralel çıkışlar vardır. Çıkış hatları üzerindeki veri geçerli olduğunda *çevirici bir çevirme sonu (EOC)* sinyali gönderir. Bu tip çeviriciler, paralel çıkışları bir mikroişlemcinin bir veya daha fazla giriş portuna bağlayarak kolaylıkla bağlanabilir.

Verinin geçerli olduğunu belirten *EOC* sinyali işlemcinin bir kesme girişine bağlanabilir, ve veriyi içeri almak için bir kesme servis yordamı kullanılabilir. Bir alternatifte *EOC* sinyal hattını bir giriş port hattına bağlamak ve hattı bir program döngüsü ile taramaktır. *EOC* sinyali algılandığında veri okunur. Bu tip çeviricinin paralel çıkış kodu genellikle ikilik veya 2'nin tümleyeni ikiliktir.

ÇOKLANMIŞ ÇIKIŞLI A/D ÇEVİRİCİLER

Sayısal voltmetrelerde ve sayısal panel metrelerde kullanılmak üzere tasarlanan A/D çeviriciler genellikle çoklanmış BCD çıkışa sahiptir. Çoklanmış BCD formatı göstergelerle kolaylıkla bağlanabildikleri için kullanılır. Bu tür çeviriciler, DVM'ler için büyük miktarlarda üretildiklerinden oldukça ucuzdurlar ve bu yüzden diğer uygulamalarda da kullanılır.

Bu çoklanmış BCD tip çeviriciyi bir mikroişlemciye bağlamak için, dört BCD hattı ve basamak strob hatlarının herbiri giriş port adreslerine bağlanır. İlk basamak-strob hattı taranır; doğru olduğu bulunduğunda bu basamak için veri çekilir ve depolanır. Bölüm 12'de bu giriş işleminin bir 8080A/8085A çevirici dil örneği gösterilmiştir. Bu tür verinin nasıl çekileceğini bilmek çok kullanışlıdır, çünkü birçok frekans sayıcı, sayısal voltmetre ve diğer test aletlerinin arka panellerinde çoklanmış BCD çıkışları mevcuttur. Bölüm 12'deki terazi örneği örneği bu tip A/D çeviricinin arabirimlerle bağlanmasının bir örneğidir.

MİKROİŞLEMCİLER VE SAR A/D ÇEVİRİCİLER

Orta hızlı uygulamalarda, bir mikroişlemci ve bir program yordamı ardışık yaklaştırma tipindeki çeviriciler içindeki ardışık yaklaştırma kaydedicinin *(SAR)* yerine kullanılabilir (Gerekirse Bölüm 6'yı yeniden gözden geçiriniz). Bir sayısaldan analoga çevirici bir çıkış portuna bağlanır. D/A çeviricinin çıkışı bir karşılaştırıcının referans girişine bağlanır. Karşılaştırıcının çıkışı da bir giriş port hattına bağlanır. Bir program yordamı her biti sırasıyla D/A çeviriciye gönderir. Karşılaştırıcı çıkışı yüksek ise bit tutulur ve bir sonraki denenir. Karşılaştırıcı çıkışı alçak ise bit resetlenir ve bir sonraki bit denenir. Bu tekniğin mikroişlemciyi meşgul etme dezavantajı vardır.

Bir çok mikroişlemci uygulaması bir A/D çevirici gerektirdiğinden Intel bu ihtiyaca *8022*'yi üreterek cevap verdi. *8022* bir entegre devre paketi içinde bir *8048 ailesi mikroişlemcisi* ve bir *tam 8-bitlik ardışık yaklaştırmalı çevirici* içerir. Entegre aynı zamanda *2 kilobaytlık ROM, 64 baytlık RAM, 28 I/O hattı* ve bir *8-bitlik zamanlayıcı* içerir.

A/D çevirici kart üzerindeki bir SAR kullanır, böylece yalnızca bir çevirimi başlatmak ve sonucu belleğe depolamak için mikroişlemci zamanı gerekir. Çevirme zamanı yaklaşık 50μsaniyedir. İki girişten biri sayısallaştırma için programla seçilebilir.

VERİ TOPLAMA SİSTEMLERİ

Bazı mikroişlemci uygulamaları birçok kaynaktan gelen analog verinin sayısallaştırılıp depolanmasını gerektirir. Her kanalda ayrı bir A/D çevirici kullanılabilir, fakat bu pahalıdır ve bütün A/D çeviricileri diğerleri ile kalibrasyon halinde tutmak zordur. Bir başka yaklaşım bir A/D çevirici kullanmak ve MC14051 veya CD4051 gibi bir analog çoklayıcı ile analog sinyalleri birer birer seçmektir. *4051, çıkışıle sekiz ikilik adreslenmiş giriş arasında düşük-dirençli* bir yol yaratır. Bu cihazın Bölüm 3'te osiloskop çoklayıcı için de kullanıldığını hatırlayabilirsiniz.

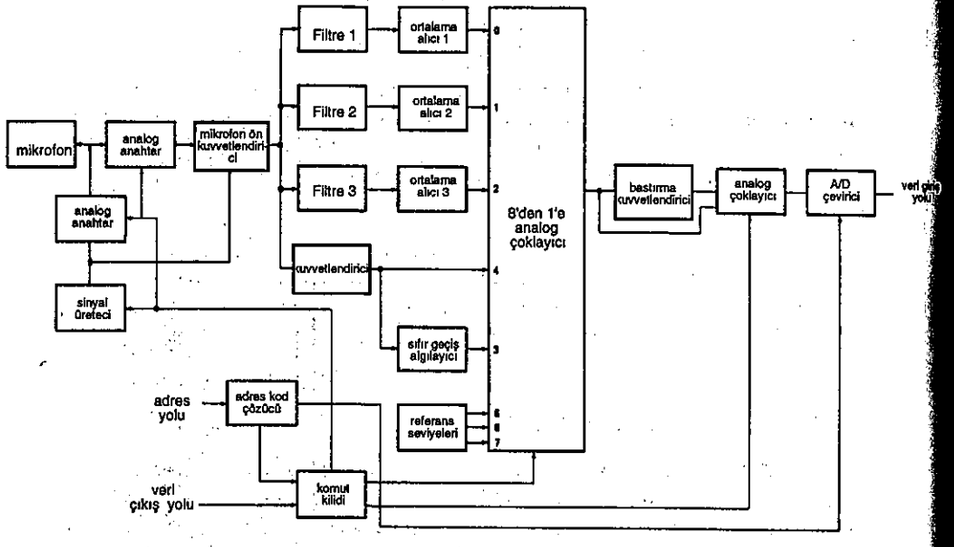
MİKROİŞLEMCİLER VE A/D ÇEVİRİCİ UYGULAMALARI

A/D çeviricilerin mikroişlemcilerle olan uygulamalarının sayısı nerdeyse sınırsızdır. Çoğu sıcaklık algılayıcıları, basınç dönüştürücüleri, sıvı akış ölçerleri ve foto algılayıcılar düşük bir akım veya gerilim çıkışına sahiptir. Bu çıkış mikroişlemcide kullanılmak için *kuvvetlendirilmeli* ve *sayısal şekle çevrilmelidir*. Bölüm 12, bir mikroişlemci-tabanlı sayısal terazi yapmak için bir A/D çevirici ile yük-hücresi basınç dönüştürücü uygulamasını göstermektedir.

A/D çeviriciler ile mikroişlemcilerin bir diğer ilginç uygulaması bir konuşmayı tanıma ünitesinin bir parçası olmasıdır. Bu üniteler veri ya da deyimlerin mikroişlemcilere direk olarak sesle iletimini sağlar. Konuşulan kelimeleri tanımak için birçok teknik kullanılmaktadır. Hiçbiri yüzde yüz doğru çalışmaz, fakat dinleyen ve konuşan bilgisayarların günleri çok uzakta değildir.

Heuristic Inc.'nin, Speech Lab Model 50 ürünü bir S-100 taşıt-uyumlu PC kartı tarafından içerilen bir konuşmayı-sayısala çevirme modülüdür. Mikroişlemcilerle birlikte kullanmak için amaçlanmıştır. Programlar sağlandığında, bir mikroişlemcinin 64 kelimeyi yüzde 90 hassasiyetten daha iyi bir hassasiyetle tanımasını öğretmeye izin verir. Şekil 11.22 Model 50'nin bir blok diyagramını göstermektedir.

Mikrofon hem sinyalleme için küçük bir hoparlör ve hem de alınan konuşma için bir mikrofon olarak kullanılır. *Mikrofon çıkışı bir önkuvvetlendirici tarafından kuvvetlendirilir* ve *80 dB/ decade etekli üç band geçiren filtrenin girişlerine* uygulanır. Üç filtre bir ortalama insan sesinin üç rezonant yada formant frekansında merkezlenmiştir. Bu filtreler 150-900, 900-2200 ve 2200-5000 Hz frekans araliklarına ayarlanmıştır. Ortalama alıcılar, filtre çıkışlarını her bant içindeki sinyalin genliğine orantılı bir gerilime çevirir.



Şekil 11.22 Heuristic Speech Lab 50 ses tanıma ünitesinin blok diyagramı

Sinyalin filtrelenmemiş bir kısmı yeniden kuvvetlendirilir ve bir sıfır geçiş algılayıcıya uygulanır. Sıfır geçiş algılayıcının darbe katarı çıkışı gerilime çevrilir ve girişteki baskın frekansı bulmak için kullanılır. Şekil 11.22'de gösterildiği gibi ortalama alıcıların çıkışları, kuvvetlendirilmiş sinyal, sıfır geçiş algılayıcı çıkışı ve üç referans geriliminin hepsi sekiz-girişli bir analog çoklayıcıya gönderilir. 4051 çoklayıcının 3-bitlik adres girişi 6-bitlik A/D çeviriciye gönderilecek gerilimi seçer.

Cihaz şu şekilde çalışır: İlk olarak, üniteyi eğitmek için bir mikroişlemci programı kullanılır. Bunu yapmak için operatör istenen ilk kelimeyi söyler. Program kelimeyi her 10 m saniyede 1.5 saniye uzunluğunda örnekler. Konuşulan kelimede düzenli aralıklardaki onaltı örnek incelenir. Bu 16 örnek için üç frekans band çıkışı ve sıfır geçiş çıkışı bilgileri sayısal olarak bellekte depolanır. Bu istenen kelimenin depolanmış bir şablonu yaratır. Benzer şekilde istenen sözlükteki her kelime için bir şablon depolanır. Her sözlük kelimesi için 64 baytlık bellek gerekmektedir.

Ünitenin çalışması istendiğinde konuşulan kelime benzer şekilde örneklenir ve dört parametre depolanır. Konuşulan kelimelerin değerleri bellekte depolanmış şablonlarla karşılaştırılır ve en iyi eşleşme doğru kelime olarak kabul edilir. Her kelime işlemciyi ışıkları açmak, kapıyı açmak ya da bir içeceği karıştırmak gibi istenen bir işi yapmak için yönlendirir.

ORTAK YOL VE VERİ HABERLEŞMESİ STANDARTLARI

Amerika Birleşik Devletlerindeki 120V, 60 Hz Standart ac güç birçok üreticiden alınan aletlerin uyumluluğu düşünmeden fişe takılmasını sağlar.* Bir mikroişlemci sisteminden bir diğerine ya da yazıcı gibi bir çevrebirim ünitesine haberleşme için ve tellerin ve yolların bağlanması için bir standartın olması istenmektedir.

Bir mikrobilgisayar içindeki veya başka bir mikrobilgisayar sistemi ile olan haberleşmede genellikle paralel yollar kullanılır. *Uzun mesafeli* haberleşmeler için veya *CRT terminaller, yazıcılar,* ve *uzaktan yazıcılarla* yapılan haberleşmeler için veri, çoğunlukla seri şekilde gönderilir.

PARALEL YOL STANDARTLARI

S-100 YOLU S-100 yol formatı MITS Inc. tarafından, kendilerinin 8080 tabanlı Altair mikrobilgisayarında kullanmak için geliştirilmiştir. Temel Altair ünitesi bir güç kaynağı, bir CPU baskılı-devre kartı, ve diğer PC kartlarının eklenebilmesi için yuvalar ve büyük bir kasadan oluşmuştur. Eklenecek PC kartları , RAM kartları, ROM kartları, bir konuşma tanıma kartı, bir flopy disk

*Türkiye'de 220 V, 50H'dir.

kontrol edici kartı, veya diğer muhtemel kartlardan biri olabilir. Her kart 100 bacaklı bir kenar konnektörüne girebilir, böylece *S-100* adı ortaya çıkmıştır. Konnektör genellikle kartın her iki tarafında 50 bacak olacak şekilde çift taraflıdır.

Bu standart bir çok hobi kartlarında ve sistemlerinde kullanılmıştır. Tablo 11.1 S-100 yolu için kullanılan bacak tanımlarını göstermektedir. Her sinyalin daha detaylı tanımı için MITS Altair 8800B elkitabına danışınız.

Yol üzerinde gönderilen güç-kaynağı gerilimlerinin regülesiz olduğuna dikkat ediniz. Yola bağlanan her kart ya da sistemin kendi gerilim regülatör entegre devrelerine sahip olacakları kabul edilmiştir. Aynı zamanda, bazı sistemlerde gerektiği gibi veri-çıkış hatlarının veri-giriş hatlarından ayrıldığına dikkat ediniz. D0 ve D1 hatlarının birleştirilmesini gerektiren RAM kartları gibi kartlarda bu iş, kart üzerindeki üç-durum tamponları ile yapılır. S-100 formatında kontrol ve kesme sinyal bacaklarının fazlalığı vardır bu yüzden 8080'den başka işlemcilerle de kullanılırlar.

Bu yolun tasarımındaki bazı eleştiriler; komşu bacaklardaki saat hatlarından kontrol hatlarına olan karışma ve komşu +8 ve -16V bacakları arasında kısa devre olmasıdır.

INTEL ÇOKLU YOL: Geniş kabul gören bir endüstriyel yol standartı Intel çoklu yollardır. Mikrobilgisayar sistemi içindeki RAM, ROM veya I/O kartlarını bağlamak için kullanılan bu yol bir 86-bacaklı kenar konnektörüne ve isteğe bağlı olan bir 60-bacaklı kenar konnektörüne sahiptir. Tablo 11.2'de gösterildiği gibi 86-bacaklı konnektör güç bölümlerine, bir 20-bitlik adres yoluna, bir 16-bitlik iki yönlü veri yoluna, ve geniş bir kontrol bölümüne sahiptir. Bir kısa komuttan sonraki bir kesimi bu bacaktaki sinyalin aktif alçak olduğunu belirtir.

Çoklu yolun en önemli özelliği, birçok CPU kartının bir çokişlemcili sistem oluşturmak için bir yola aynı zamanda bağlanabilmesidir. Her CPU kartı, sistem bellek kartına, bir diğer CPU kartına, veya bir yazıcı veya sabit disk depolama ünitesine erişim ihtiyacı hissedene kadar kendi kartı üzerindeki bulunan ROM, RAM ve I/O portlarını kullanarak özel olarak ayrılmış işleri yerine getirebilir. Bir CPU kartı yolu kullanmak istediğinde, kendi *yol istek sinyalini (BREQ/)* ve *yol öncelik çıkış sinyalini (BPRO/)* onaylar. Bir paralel ya da seri öncelik kodlayıcı planı hangi CPU kartının yolun kontroluna sahip olduğunu belirler. Yol kontrolunun bir CPU'dan diğerine düzenli geçişi Intel 8218 veya 8288 gibi bir yol kontrol edici entegre devresi tarafından yapılır.

Tablo 1.1

S-100 Paralel Taşıt Bacak Tanımları

BACAK NUM.	SEMBOL	İSİM	FONKSİYON
1	+8V	+8VOLT	Yol üstündeki regülesiz gerilim, *PC* kartlara sağlanır ve 5 volta regüle edilir.
2	+18V	+18Volt	Pozitif önregüle edilmiş gerilim
3	XRDY	Harici hazır	*CPU* kartın hazır devresine harici hazır girişi
4	VI0	Vektörlü kesme hattı 0	
5	VI1	Vektörlü kesme hattı 1	
6	VI2	Vektörlü kesme hattı 2	
7	VI3	Vektörlü kesme hattı 3	
8	VI4	Vektörlü kesme hattı 4	
9	VI5	Vektörlü kesme hattı 5	
10	VI6	Vektörlü kesme hattı 6	
11	VI7	Vektörlü kesme hattı 7	
12	XRDY2	Harici hazır 2	XRDY benzer ikinci harici hattı
13-17	Tanımlanacak		
18	STAT DSB	Statü yetkisizleştirme	Sekiz statü hattı için tamponların üç -durumlu olmasını sağlar
19	C/C DSB	komut/kontrol yetkisizleştirme	Altı çıkış komut/kontrol hattı için tamponların üç durumlu olmasını sağlar
20	UNPROT	Korumayı kaldır	Verilen bir bellek kart üzerindeki bellek koruma flip flopuna giriş
21	SS	Tek adım	Makinenin tek adımı yerine getirme süreci içinde olduğunu belirtir (yani gösterge/kontrol üzerindeki SS Flip-flop set edilir)
22	ADD DSB	Adres yetkisizleştirme	Onaltı adres hattı için tamponların üç-durumlu olmasını sağlar
23	DO DSB	Veri çıkışı yetkisizleştirme	Sekiz veri çıkış hattı için tamponların üç-durumlu olmasını sağlar
24	02	Faz 2 saati	
25	01	Faz 1 saati	
26	PHLDA	Hold alındı	*HOLD* sinyaline cevap olarak gözüken işlemci komut/kontrol çıkış sinyali; veri ve adres yollarının yüksek-empedans durumuna gideceğini ve işlemcinin şu andaki makine döngüsünü tamamladıktan sonra *HOLD* durumuna gireceğini belirtir.
27	PWAI T	Bekle	*READY* sinyalinin alçağa gidişine cevap olarak gözüken işlemci Komut/Kontrol çıkış sinyali; işlemcinin *READY* sinyali yeniden yüksek oluncaya kadar 0,5 µ saniyelik bir seri bekleme durumlarına gireceğini belirtir.
28	PINTE	Kesme yetkilendirme	İşlemci komut/kontrol çıkış sinyali; *CPU* dahili kesme flip-flopu içeriği tarafından belirlendiği gibi kesmelerin yetkilendirildiğini belirtir. Flip-flop set edildiğinde (kesme yetkilendirme deyimi) kesmeler CPU tarafından kabul edilir, flip-flop resetlendiğinde (kesme yetkisizleştirme deyimi), kesmeler bastırılır.
29	A5	Adres hattı 5	
30	A4	Adres hattı 4	

31	A3	Adres hattı 3	
32	A15	Adres hattı 15	
33	A12	Adres hattı 12	
34	A9	Adres hattı 9	
35	DO1	Veri çıkış hattı 1	
36	DO0	Veri çıkış hattı 0	(En küçük değerlikli bit)
37	A10	Adres hattı 10	
38	DO4	Veri çıkış hattı 4	
39	DO5	Veri çıkış hattı 5	
40	DO6	Veri çıkış hattı 6	
41	D12	Veri çıkış hattı 2	
42	D13	Veri çıkış hattı 3	
43	D17	Veri çıkış hattı 7	(En büyük değerlikli bit)
44	SM1	Makine döngüsü 1	İşlemcinin bir deyimin birinci baytı için getirme döngüsünde olduğunu belirten statü çıkış sinyali.
45	SOUT	Çıkış	Adres yolunun bir çıkış cihazının adresini içerdiğini ve *PWR* aktif olduğunda veri taşıtının çıkış verisini içereceğini belirten statü çıkış sinyali
46	SINP	Giriş	Adres yolunun bir giriş cihazının adresini içerdiğini ve *PDBIN* aktif olduğunda giriş verisinin veri taşıtı üzerine yerleştirileceğini belirten statü çıkış sinyali.
47	SMEMR	Bellek okuma	Veri yolunun bellekteki veriyi okumak için kullanılacağını belirten statü çıkış sinyali.
48	SHLTA	Durma	Bir *HALT* deyiminin alındığını belirten statü çıkış sinyali
49	CLOCK	Saat	Faz 2 saatinin evrilmiş çıkışı
50	GND	Toprak	
51	+8V	+8Volt	5V regülatörlere regülesiz giriş
52	-18V	-18Volt	Negatif önregüle edilmiş gerilim
53	SSWI	Hissetme anahtarı girişi	Hissetme anahtarlarından bir giriş veri transferi olacağını belirtir. Bu sinyal gösterge/ kontrol mantığı tarafından hissetme anahtarları sürücülerini yetkilendirmekte, gösterge /Kontrol kart sürücüleri veri girişini *(FD10-FD17)* yetkilendirmekte, ve *CPU* kartı veri giriş sürücülerini *(D10-D17)* yetkisizleştirmekte kullanılır.
54	EXT CLR	Harici silme	*I/O* cihazları için silme sinyali (ön panel anahtarı toprağa kapalı)
55	*RTC	Gerçek-zaman saati	60Hz'lik sinyal gerçek-zaman saati/vektörlü kesme kartı tarafından zamanlama referansı olarak kullanılır.
56	*STSTB	Statü strob	8224 saat üreteci tarafından sağlanan çıkış strob sinyali. Temel amacı makine döngüsü içinde statünün ayarlaması için 8212 statü kilidini stroblamaktadır. Bu sinyal gösterge / kontrol mantığıncada kullanılır.
57	*DIGI	Veri giriş kapısı 1	Hangi veri giriş sürücülerinin *CPU* kartının iki yönlü veri yolu üzerinde kontrolu olduğunu belirleyen gösterge / kontrol mantığından gelen çıkış sinyali. *DIGI* yüksek ise *CPU* sürücüleri kontrola sahiptir; eğer alçak ise gösterge/kontrol mantık sürücüleri kontrola sahiptir
58	*FRDY	Ön panel hazır	Ön panelin *CPU*'ya olan *READY* hatlarını kontrol etmesini sağlayan gösterge/ kontrol mantığından gelen çıkış sinyali
59-67	Tanımlanacak		
68	MWRITE	Belleğe yazma	Veri çıkış yolundaki mevcut verinin adres yolu üzerindeki bellek adresine yazılacağını belirtir
69	PS	Koruma statüsü	Şu anda adreslenen bellek kartı üzerindeki bellek koruma flip-flopunun statüsünü belirtir
70	PROT	Koruma	Şu anda adreslenen kart üzerindeki bellek koruma flip-flopuna giriş
71	RUN	Çalışma	64/*RUN* flip-flopunun resetlendiğini belirtir, yani makine *RUN* modundadır.
72	PRDY	İşlemci hazır	*CPU* kartı *WAIT* devresine bellek ve *I/O* girişi
73	PINT	Kesme isteği	Şu andaki deyimin sonunda ya da durdurulmuşken

			işlemci bu hat üstünde bir kesme isteği olduğunu anlar. Eğer işlemci *HOLD* durumundaysa veya kesme yetkilendirme flip-flopu resetse, isteğe cevap vermez.
74	PHOLD	Tutma	İşlemcinin *HOLD* durumuna girmesini isteyen işlemci komut/kontrol giriş sinyali; işlemcinin şu andaki makine döngüsü için adres ve veri yolarının kullanımını bitirir bitirmez bu yolların kontrolunun bir harici cihaz tarafından alınmasına izin verir.
75	PRESET	Reset	İşlemci komut/kontrol girişi; aktif olduğunda, program sayıcının içeriği silinir ve deyim kaydedici sıfıra ayarlanır
76	PSYNC	Sync	İşlemci komut/kontrol çıkışı her makine döngüsünün başlangıcını belirten bir sinyal sağlar
77	PWR	Yazma	Belleğe yazma veya *I/O* çıkış kontrolu için kullanılan işlemci komut/kontrol çıkışı. *PWR* aktifken veri yolu üzerindeki veri kararlıdır.
78	PDBIN	Veri yolu girişi	Harici devrelere, veri yolunun giriş modunda olduğunu belirten işlemci komut/kontrol çıkışı
79	A0	Adres hattı 0	
80	A1	Adres hattı 1	
81	A2	Adres hattı 2	
82	A6	Adres hattı 6	
83	A7	Adres hattı 7	
84	A8	Adres hattı 8	
85	A13	Adres hattı 13	
86	A14	Adres hattı 14	
87	A11	Adres hattı 11	
88	DO2	Veri çıkışı hattı 2	
89	DO3	Veri çıkışı hattı 3	
90	DO7	Veri çıkışı hattı 7	
91	DI4	Veri çıkışı hattı 4	
92	DI5	Veri çıkışı hattı 5	
93	DI6	Veri çıkışı hattı 6	
94	DI1	Veri çıkışı hattı 1	
95	DI0	Veri çıkışı hattı 0	(En küçük değerlikli bit)
96	SINTA	Kesme alındı	Kesme isteği sinyalinin alındığını belirten statü çıkış sinyali
97	SWO	Yazma-çıkış	Şu andaki makine döngüsü işleminin bir belleğe yazma veya çıkış fonksiyonu olduğunu belirten statü çıkış sinyali
98	SSTACK	Yığın	Adres yolunun, yığın göstericiden aşağı itilen yığın adresini tuttuğunu belirten statü çıkış sinyali
99	POC	Güç açıldığında sil	
100	GND	Toprak	

* 8800b için yeni taşıt sinyali
KAYNAK: MITS-Altair

HPIB VEYA IEEE-48: Hewlett-Packard Arabirim yolu (HPIB) bilgisayar, sayısal voltmetre, ve sinyal üreticileri gibi aletleri bağlamakta kullanılır. Çoğu kısmı Hewlett-Packard tarafından geliştirilmiş ve daha sonra IEEE tarafından standart olarak kabul edilmiştir. Gerçekte Hewlett-Packard bu standartta kullanılan karmaşık onay metodunun patentini elinde tutmaktadır. Aynı zamanda genel amaçlı *arabirim yolu (GPIB)* olarak da bilinir.

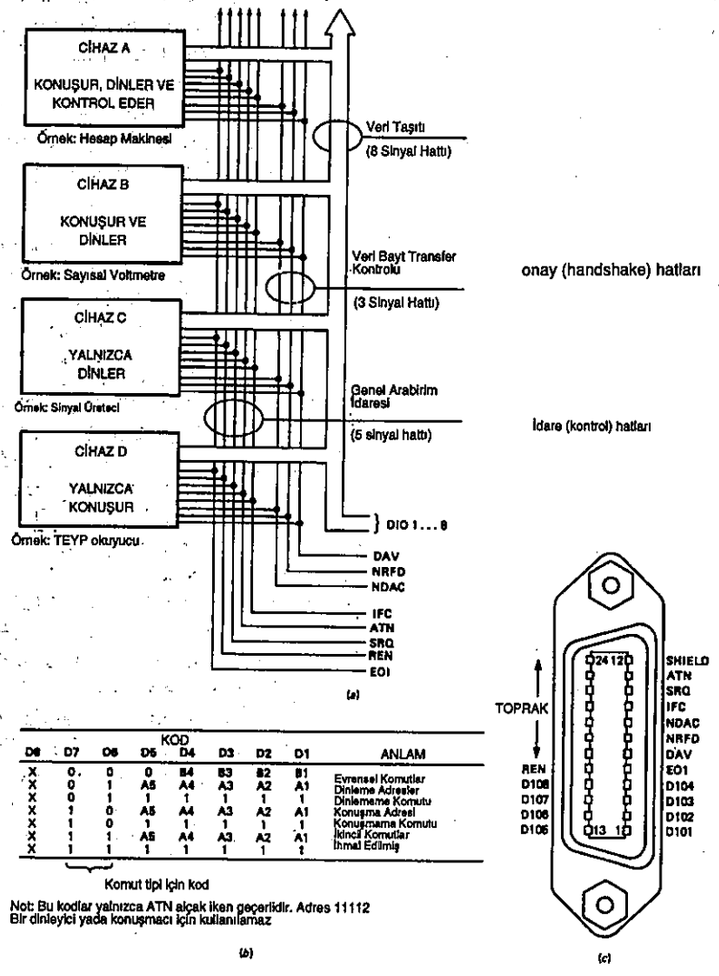
Standart, HPIB'ye bağlanabilecek üç tip birim tanımlar. Birincisi diğer aletlerden veri alabilen bir dinleyicidir. Dinleyici örnekleri: bir yazıcı, gösterge cihazı programlanabilir güç kaynağı, veya programlanabilir sinyal üreticidir. İkincisi diğer aletlere veri gönderebilen bir konuşmacıdır. Konuşmacı örnekleri: teyp okuyucular, sayısal voltmetreler, ve frekans sayıcılardır. Bir cihaz hem konuşmacı hem de bir dinleyici olabilir. Örneğin fonksiyonu program kontrolu ile seçilebilen programlanabilir bir DVM bir konuşmacı ve dinleyicidir. Üçüncüsü, yol üzerinde kimin konuştuğunu ve kimin dinlediğini belirleyen bir kontrol edicidir.

Şekil 11.23*a*'da gösterildiği gibi, GPIB sekiz *ikiyönlü veri hattına* sahiptir. Bu hatlar 8 taneden 15 taneye kadar olan aletler arasında verileri, adresleri, komutları ve statü baytlarını transfer eder. Şekil 11.23*b*, bir kontrol edicinin konuşmacı ve dinleyicilere gönderebileceği komut-adres kodları kombinasyonunun formatını göstermektedir. Bu kelimelerin 8. biti farketmezdir, bit 7 ve 6 hangi komutun gönderildiğini belirtir, ve bit 5'ten bit 1'e olan bitler komutun gönderildiği konuşmacı ya da dinleyicinin adresini verir. Örneğin 04 adresindeki bir cihazı bir konuşmacı olarak ayarlamak için kontrol edici *dikkat hattını (ATN)* alçak yapar ve X1000100'lık bir komut-adres baytını gönderir. Her dinleyici, kendisine kontrol edici tarafından $X01A_5A_4A_3A_2A_1$ içeren bir bayt gönderildiği zaman yetkilendirilir. Veri transferi tamamlandığında kontrol edicinin *dinlememe komutu* X011111'i göndermesiyle dinleyiciler kapatılır, ve bir konuşmacı, kontrol edicinin X1011111 *konuşmama komutunu* göndermesiyle kapatılır. 7,6 ve 5. bitlerin hepsinin 0 olduğu evrensel komutlar bütün aletlere gidecektir. Bu kelimenin alt 4 biti 16 evrensel komuttan birini belirtir.

TABLO 11.2
İNTEL ÇOKLU TAŞIT BACAK TANIMLARI

Eleman Tarafı				Devre Tarafı		
	BACAK	KISA KOMUT	TANIM	BACAK	KISA KOMUT	TANIM
Güç Kaynakları	1	GND	GND Sinyali	2	GND	GND sinyali.
	3	+5V	+5V dc	4	+5V	+5V dc
	5	+5V	+5Vdc	6	+5V	+5V dc
	7	+12V	+12Vdc	8	+12V	+12Vdc
	9	-5V	-5Vdc	10	-5V	-5V dc
	11	GND	GND sinyali	12	GND	GND sinyali
Yol kontrolları	13	BLCK/	Yol saat	14	INIT/	Kullanıma hazırla
	15	BPRN/	Yol giriş önceliği	16	BPRO/	Yol çıkış önceliği
	17	BUSY/	Yol meşgul	18	BREQ/	Yol isteme
	19	MRDC/	Bellek okuma komutu	20	MWTC	Belleğe yazma komutu
	21	IORC/	I/O okuma komutu	22	IOWC/	I/O yazma komutu
	23	XACK/	XFER alındı sinyali	24	INH1/	RAM yetkisizleştirme
Yol kontrolları ve adresler	25		Ayrılmış	26	INH2/	PROM veya Rom'u yetkisizleştirme
	27	BHEN/	Yüksek bayt yetkilendirme	28	AD10/	
	29	CBRQ/	Ortak yol isteği	30	AD11/	
	31	CCLK/	Sabit saat	32	AD12/	Yol
	33	INTA/	Kesme alındı sinyali	34	AD13/	
kesmeler	35	INT6/	Paralel	36	INT7/	Paralel kesme
	37	INT4/	Kesme	38	INT5/	istekleri
	39	INT2/	İstekleri	40	INT3/	
	41	INT0/		42	INT1/	
Adres	43	ADRE/		44	ADRF/	
	45	ADRC/		46	ADRD/	
	47	ADRA/		48	ADRB/	
	49	ADR8/		50	ADR9	
	51	ADR6/	Adres yolu	52	ADR7/	Adres yolu
	53	ADR4/		54	ADR5/	
	55	ADR2/		56	ADR3/	
	57	ADR0/		58	ADR1/	
Veri	59	DATE/		60	DATF/	
	61	DATC		62	DATD/	
	63	DATA/		64	DATB/	
	65	DAT8/		66	DAT9/	
	67	DAT6/	Veri yolu	68	DAT7/	Veri yolu
	69	DAT4/		70	DAT5/	
	71	DAT2/		72	DAT3/	
	73	DAT0/		74	DAT1/	

	75	GND	GND sinyali	76	GND	GND sinyali
	77		Ayrılmış	78		Ayrılmış
Güç	79	-12V	-12Vdc	80	-12V	-12V dc
kay-	81	+5V	+5V dc	82	+5V	+5V dc
nakları	83	+5V	+5V dc	84	+5V	+5V dc
	85	GND	GND sinyali	86	GND	GND sinyali



KOD								
D8	D7	D6	D5	D4	D3	D2	D1	ANLAM
X	0	0	0	B4	B3	B2	B1	Evrensel Komutlar
X	0	1	A5	A4	A3	A2	A1	Dinleme Adresler
X	0	1	1	1	1	1	1	Dinlememe Komutu
X	1	0	A5	A4	A3	A2	A1	Konuşma Adresl
X	1	0	1	1	1	1	1	Konuşmama Komutu
X	1	1	A5	A4	A3	A2	A1	İkinci Komutlar
X	1	1	1	1	1	1	1	İmal Edilmiş

Komut tipi için kod

Not: Bu kodlar yalnızca ATN alçak iken geçerlidir. Adres 11112 Bir dinleyici yada konuşmacı için kullanılamaz

(b)

(c)

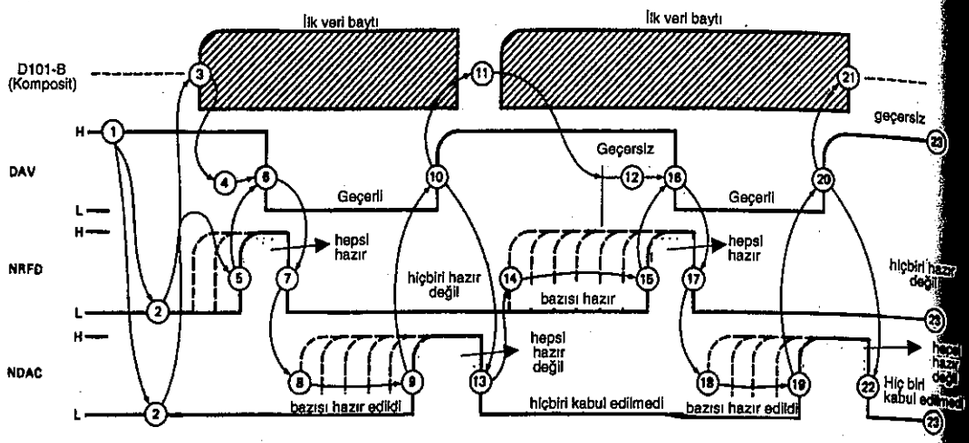
Şekil 11.23 *a)* Hewlett-Packard arabirim yolu *b)* GPIB adres-komut baytı formatları *c)* GIPB konnektör

GPIB aşağıdaki şekilde çalışan beş *idare hattına* da sahiptir. *Arabirim silme hattı (IFC)* kontrolör tarafından onaylandığında bütün cihazları bilinen yada başlangıç durumuna getirir. Bu sistem resetine benzemektedir. *Dikkat (ATN)* onaylandığında (alçak), kontrolörün veri yoluna ya bir evrensel komut ya da dinle gibi bir adres-komut koymakta olduğunu belirtir.

ATN onaylanmadığında veri hatları bir konuşmacıdan gelen bir veri yada statü baytı içerir. *Servis isteği (SRQ)* bir kesmeye benzer. Bir cihaz tarafından kontrolöre dikkat isteğini belirtmek için gönderilir. Örneğin bir cihaz bir seri ölçümleri tamamlandığında ve tamponu verilerle dolu olduğunda bir SRQ gönderebilir. Kontrolör tarafından onaylandığında *uzaktan yetkilendirme (REN)* bir aleti, aletin ön panel kontrolu (yerel) yerine sistem kontrolörü (uzaktan) yetkilendirir. *Tanıtmanın sonu (EOI)* genellikle bir konuşmacı tarafından sistem kontrolörüne bir veri bloğunun transferinin tamamlandığını belirtmek için gönderilir.

Veri yolu ve idare yoluna ilaveten, GPIB veri yolu üzerindeki veri transferini koordine etmek için üç el sıkışma hattına sahiptir: *Veri geçerli (DAV). veri için hazır değil (NRFD)*, ve *veri kabul edilmedi (NDAC).* GPIB onaylı veri transfer metodunun temel özelliği çok değişik cevap hızlarına sahip cihazların bir sistemde birlikte bağlanmasına izin vermesidir.

Şekil 11.24 bir konuşmacıdan birçok dinleyiciye veri transferi için onay hatları üzerindeki sinyallerin sırasını gösterir. DAV, NFRD ve NDAC açık kollektör aktif alçak sinyallerdir. NFRD ve NDAC açık kollektör, olduklarından, herhangi bir dinleyici NFRD'yi alçağa çekerek veri için hazır olmadığını belirtir veya NDAC'yi alçağa çekerek verinin kabul edilmediğini belirtir. Örneğin NFRD hattı adreslenen bütün dinleyiciler onu serbest bırakıncaya kadar hazır olduğunu belirtmek için yükseğe gitmez. Bütün dinleyiciler hazır olduklarını belirttiklerinde (Şekil 11.24'de 5) konuşmacı, geçerli verinin yolda olduğunu belirtmek için DAV sinyalini onaylar. Adreslenen dinleyicilerin hepsi NFRD hattını alçak yapar ve veriyi kabul etmeye başlar. Her dinleyici veriyi kabul etikçe NDAC hattını tutmayı bırakır. En yavaş dinleyici veriyi kabul ettiği zaman NDAC hattı yükseğe gider. Konuşmacı bunu hisseder ve kendi DAV hattını yükseğe çekerek bir sonraki veri baytını göndermeye hazırlanır. DAV yükseğe gidince, bütün adreslenen dinleyiciler NDAC'yi yeniden alçak yapar. İşlem tekrarlanır. Bu durumda en yavaş dinleyici verinin gönderilme hızını belirler. GPIB için belirtilen en yüksek veri hızı 1 mega bayt/sn'dir, fakat tipik veri hızı 250kbayt/sn veya daha azdır.



Şekil 11.24 Bir konuşmacıdan birçok dinleyiciye bir baytlık veri transferi için GPIB sinyal sırası

Muhtemel bir dizi örneği bu hatların hepsinin nasıl birlikte çalıştıklarını gösterecektir. Güç kaynağı açıldığında kontrolör yolun kontrolunu ele alır ve taşıt üstündeki bütün aletleri bilinen bir duruma ayarlamak için bir IFC sinyali gönderir. Belki de bir konuşmama ve dinlememe sinyalide gönderebilir. Daha sonra kontrolör yol ve aletleri istendiği şekilde kullanır. Kontrolör periyodik olarak bir servis isteği için SRQ hattını kontrol eder. SRQ hattı alçak ise, kontrolör servis isteyen cihazı (cihazları) buluncaya kadar yol üstündeki her cihazı sırayla (seri) veya bir defada hepsini (paralel) tarar. Örneğin DVM gibi bir konuşmacı, bir çevirmeyi bitirdiğini ve grafik kaydedici gibi bir dinleyiciye gönderilecek bazı verileri olduğunu belirtiyor olabilir. Kontrolör SRQ'nun kaynağını belirlediği zaman, ATN'yi alçak yapar ve her dinleyiciye veriyi almaları için dinleyici adres komutlarını gönderir. Daha sonra kontrolör istenen konuşmacıya bir konuşma adres komutu gönderir ve ATN'yi yeniden yükseğe çeker. Bu durumda konuşmacı, biraz önce tanımlanan NFRD-DAV-NDAC onay sırasını kullanarak veriyi dinleyicilere gönderir. Veri transferi tamamlandığında konuşmacı kontrolöre bir EOI sinyali gönderir. Daha sonra kontrolör, kontrolu yeniden ele alır. Bir konuşmama adres komutu ve dinlememe adres komutu gönderir. Yol, şimdi diğer veri transferleri için hazır durumdadır.

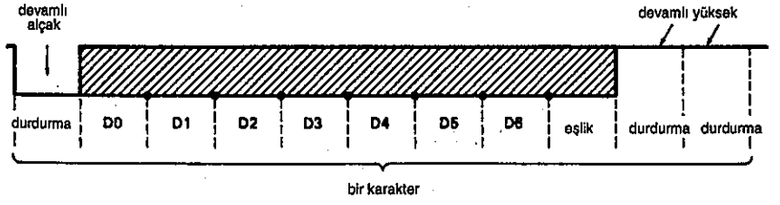
Bir standart 8080A veya 8085A veri yolu, Intel 8291 GPIB konuşmacı-dinleyici ile HPIB'ye arabirimle bağlanabilir. Intel 8292 bir GPIB kontrolörüdür. 6800 tip işlemcileri HPIB'ye bağlamak için MC68488 genel-amaçlı arabirim adaptörü kullanılır. Bir dinleyici veya konuşmacı olarak hizmet verebilir.

IEEE-488 yolunun önemi, bir bilgisayarın veya mikrobilgisayarın birçok test aletine bağlanarak bir entegre test sistemi oluşturmasına imkan sağlamasıdır.

SERİ VERİ GİRİŞİ VE ÇIKIŞI

Sayısal verileri uzak mesafelere göndermek için gerekli bağlantı tellerinin sayısını azaltmak için genellikle veri seri şekilde gönderilir. Diğer bir deyişle, bir kelimenin bir biti bir diğerinden sonra tek bir tel üzerinde gönderilir. Veri kelimeleri *senkron* veya *asenkron* olarak gönderilir.

Asenkron formatta bir veri kelimesinin bitleri ya da karakter sabit hızla gönderilir, fakat karakterler üstüste çakışmadıkları sürece herhangi bir hızda gelebilirler. Hiç karakter gönderilmediği durumda, yüksek ya da *işaretli tutulur*. Şekil 11.25 asenkron seri verinin formatını göstermektedir (İlk tartışmalar için Bölüm 4'e bakınız). Bir karakterin başlangıcı devamlı alçak olan bir başlangıç biti ile belirtilir. Başlangıç bitinden sonra veri bitleri ilk olarak en küçük değerlikli bit olmak üzere sırayla gönderilir. Değişik sistemler 5,6,7 veya 8 veri biti kullanır. Bir eşlik biti veri bitlerini takip edebilir. Karakterin sonu, devamlı yüksek olan bir ya da daha fazla durdurma bitleri ile belirtilir. Değişik sistemler $1^{1/2}$ veya 2 durdurma biti kullanır.



Şekil 11.25 Asenkron seri verinin formatı

Bitlerin gönderilme hızı *baud hızı* olarak adlandırılır. Baud hızı 1'in 1 bit zamanına bölümüne eşittir. Örneğin bit zamanı 3.33msn ise baud hızı 1/3.33msn = 300Bd olur. Yaygın hızlar 75, 110, 150, 300, 600, 1200, 2400, 4800, 9600 ve 19200 Bd'dir.

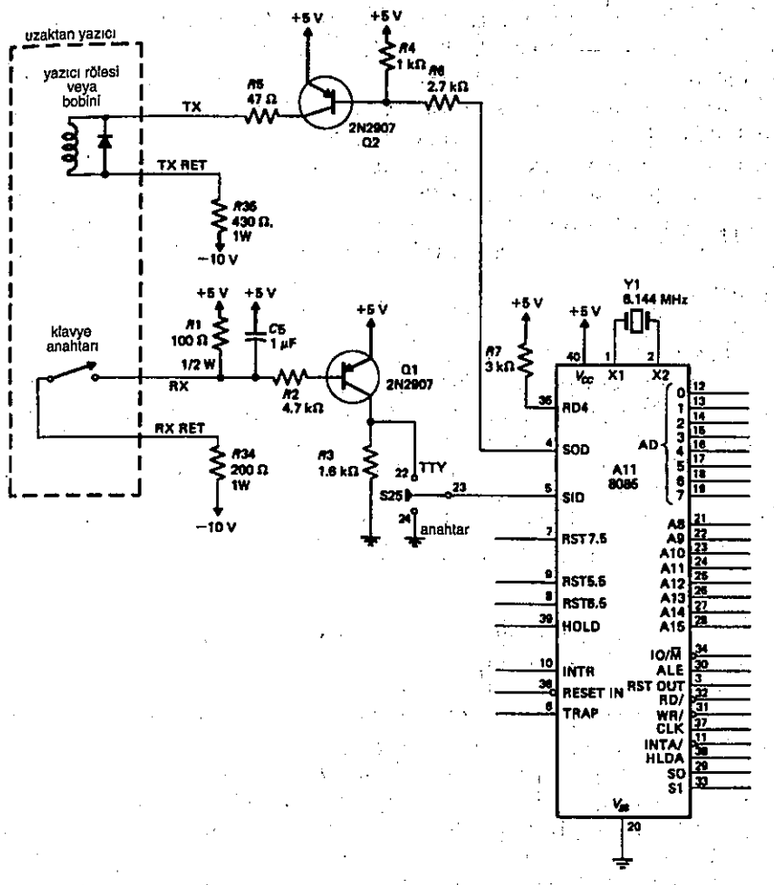
Mikrobilgisayarlar paralel şekildeki veri ile çalışır. Bu yüzden seri hatlarda bağlanabilmek için veri seri şekle ve seri şekilden paralele çevrilmelidir. Bir paralel-giriş, seri çıkış (PISO) kayma kaydedicisi ve seri-giriş, paralel-çıkış (SIPO) kayma kaydedici bu işleri yapabilir. Bu amaç için birlikte paketlenirler ve *evrensel asenkron alıcı-verici (UART)* olarak adlandırılırlar. Bir UART'nın bir örneği Bölüm 4'te tanımlanan AY-5-1013'tür. Bu cihaz, veri bitleri sayısı, eşlik, ve durdurma bitleri için donanım programlanabilir. İstenilen baud hızının 16 katında bir verici-alıcı saati gerektirir. Alıcı veri çıkışları üç-durum kapasitesine sahiptirler, böylece bir mikrobilgisayar veri yoluna doğrudan bağlanabilir ve bir $\overline{CS}$ sinyali tarafından yetkilendirilirler. Bu şekildeki bir cihaz, verici ve alıcı parametrelerinin her zaman aynı olduğu özel olarak ayrılmış sistemlerde çok iyi çalışır. Fakat genel amaçlı bir mikrobilgisayarda, değişik baud hızları, değişik veri bitleri sayısı, değişik durdurma bitleri sayısı, ve değişik eşlik bitleri için yazılım programlanabilir bir UART'a ihtiyaç vardır. Genellikle bu UART'ın, modülatör-demodülatörlerle (MODEM), CRT terminaller ve yazıcılarla onaylı veri transferi kapasitesine sahip olması gereklidir.

Çoğu mikroişlemci üreticileri, onay kabiliyetine sahip yazılım-programlanabilir UART'lar üretirler. En yaygın olan iki tanesi Intel 8251/8251A ve Motorola MC6850'dir. Daha ileride bu cihazların seri verileri göndermek ve almak için kullanıma nasıl hazırlandıklarını göreceğiz. Fakat ilk olarak, yaygın olarak kullanılan sinyalleri ve onay standartlarını tanımlamak gereklidir. Seri 1'ler ve 0'lar verisi , akım var ve akım yok olarak, iki gerilim seviyesi olarak, veya iki ses tonu olarak gönderilebilir.

20 ve 60mA'lik AKIM DÖNGÜLERİ: UZAKTAN YAZICI (TELETAYP)

Uzaktan yazıcılar, seri 1'ler ve 0'ları genellikle sabit bir akım veya akım yok olarak gönderir ve alır. Bir 1 veya *işaret*, 20 veya 60mA'lik bir nominal akımdır, ve bir 0 veya *boşlukta* akım yoktur. Uzaktan yazıcı veriyi 300 Bd veya daha az bir hızda gönderir. 110Bd'lik hız çok yaygındır. Şekil 11.26, SDK-85 mikrobilgisayar kartı üzerinde 8085 SID ve SOD portlarının TTL seviyelerini 20mA'lik akım döngüsü seviyelerine arabirimle bağlamak için kullanılan devreyi göstermektedir. Uzaktan yazıcıya gelen akım var veya akım yok dizisi istenilen karakterleri basmak için dahili basma çubuklarını ayarlar.Şekil 11.26'da gösterildiği gibi, veri göndermek için uzaktan yazıcı her veri bitini kapalı veya açık bir anahtar olarak temsil eder. Eğer uzaktan yazıcı anahtarı kapatılırsa, +5Volttan R1 ile uzaktan yazıcı anahtarı üzerinden ve RX RET ve R34

üzerinden negatif kaynağa akım akar. Bu akım akışı *Q1*'i iletime geçirir. *Q1* üzerinden akan akım *R3* üzerinde bir gerilim üretir ve 8085A SID portuna bir yüksek seviye gönderir.Hiç karakter gönderilmediğinde, anahtar kapalıdır böylece sabit bir yüksek ya da işaret gönderilir. Bir bit zamanı sırasında uzaktan yazıcı anahtarı açık ise akım akmaz, *Q1* iletimde olmaz ve SID alçak seviye alır. Şekil 11.26'da *R1* yanındaki 1µF kondansatör *C5*'in amacı uzaktan yazıcı içindeki döner anahtar tarafından üretilen gürültüyü filtrelemektir.



Şekil 11.26 SDK-85'in 20mA akım döngüsü uzaktan yazıcı arabirimle bağlanması (Intel corporation, eklemelerle birlikte)

Bir uzaktan yazıcıyı Şekil 11.26'da gösterildiği gibi bir bilgisayara iki bağımsız akım döngüsü ile bağlamak *tam-dubleks mod* olarak adlandırılır. *Tam-dubleksin* anlamı gönderilen ve alınan veriler için ayrı hatların kullanılmasıdır. Böylece veri aynı zamanda gönderilebilir ve alınabilir. Şekil 11.26'da bir uzaktan yazıcı klavyesinden bir tuşa basıldığında bu tuşun karakteri bilgisayara gönderilir. Daha sonra bilgisayar bu karakteri uzaktan yazıcıya geri gönderir *(ekolar)*. Bu metodun bir avantajı, karakteri basılı gördüğünüzde bilgisayarın basılan karakteri aldığını bilmenizdir.

Uzaktan yazıcı yarım-dubleks modunda da bağlanabilir, bu modda klavye anahtarı ve yazıcı, elektromıknatıs bir döngü içinde bir akım kaynağı ile bağlanırlar. Şekil 11.26'ya bakarsak, R35 ve R1 devreden çıkarılırsa ve TX RET hattı Rx ile birleştirilirse, devre bir yarım-dubleks hat gibi çalışır. Not:20mA'lik işaret akımının akması için SOD çıkışı alçak olmalıdır. Bir tuşa basıldığında, anahtar açma ve kapamalar yazıcı elektromıknatısı üzerinden akan akımı keser. Böylece bir karakter basıldığı anda yazılır. Bilgisayardan bir eko gelmesine gerek yoktur.

MODEMLER VE RS-232C

MODEMLER: 1960'larda bilgisayar terminallerinin ortak kullanımı genişlemeye başlayınca bu terminalleri uzaktaki bilgisayarla kolayca bağlamanın yollarını bulmak gerekti. Halen kurulu oldukları için standart telefon hatları mantıklı bir seçimdi. Standart telefon hatlarındaki bir problem bu hatların gerilim-yada akım darbesi tipinde sinyalleri göndermekte kullanılamamasıdır. Standart telefon hatlarının band genişliği yaklaşık 3kHz'dir. Darbeleri alçak-geçiren bu filtreden göndermek, bu darbeleri tanınmayacak şekilde bozar. Bu yüzden, sayısal veriler telefon hatları üzerinde üç sinüs dalga şeklinden biriyle gönderilirler. Bu dalga şekilleri: genlik modülasyonu, faz kaymalı modülasyonu, veya frekans kaymalı anahtarlamadır *(FSK). Genlik modülasyonunda* bir sinüs dalga taşıyıcının varlığı bir 1'i belirtir. Taşıyıcı olmaması bir 0'ı belirtir. *Faz kaymalı modülasyonda,* sabit frekanslı taşıyıcının fazındaki 180° kayma 0'dan 1'e veya 1'den 0'a olan değişmeyi belirtir. *Frekans kaymalı anahtaramada* sayısal 1 bir frekanstaki sinüs dalgası ile ve sayısal 0 başka bir frekanstaki sinüs dalgası ile belirtilir. (Şekil 5.13'e bakınız). Her iki frekans telefon hatlarının bant geçiren aralığı içindedir.

Bir *modülatör-demodülatör (MODEM)* telefon hatları ile uyumlu sinyalleri gönderir ve alır. Örneğin FSK kullanılıyorsa, MODEM'in modülatör bölümü mantık seviye sinyallerini uygun sinüs dalgası çıkış frekanslarına çevirir. MODEM'in demodülatör kısmı ise alınan iki sinüs dalga sinyalini sayısal 1'ler ve 0'lara çevirir.

Motorola MC6860 600 Bd'ye kadar olan FSK verilerini gönderebilen ve alabilen bir MODEM için gerekli devrelerin büyük kısmını içerir.

MODEM'ler dört farklı frekans kullanarak, bir anda her iki yönde de haberleşebilir (tam-dubleks modu). Bir MODEM *kaynak modu* veya *cevap modundan* birinde çalışabilir. Kaynak modunda 1270 Hz'lik frekansı bir işaret veya 1 için, ve 1070Hz'i bir boşluk veya 0 için kullanır. Cevap modundaki bir MODEM, kaynak modundaki MODEMden gelen 1270Hz ve 1070Hz'lik sinyalleri kabul eder ve işaret için 2225 Hz'lik sinüs dalga ve boşluk için 2025Hz'lik sinüs dalga ile geri haberleşir. Bu sistemde, bilgisayar veriyi almaya hazır olmadan terminalin veri göndermemesini sağlamak için onay sinyalleri de gereklidir. MODEM ler arasındaki onay bir yüksek seviye için frekansın varlığı veya yokluğu ile yapılır. Örneğin bir kaynak-modu MODEM, cevap-modu MODEM'den 2225Hz'lik sinyali alıncaya kadar veriyi göndermez.

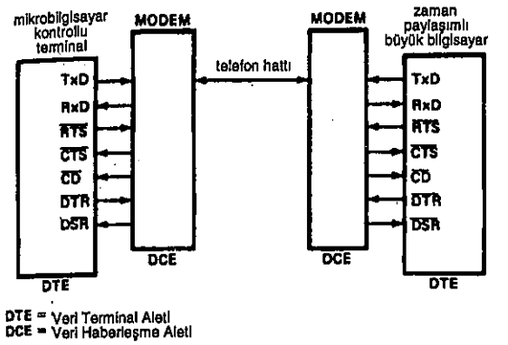
RS-232C: Seri verileri uzak mesafelere göndermekte kullanılan MODEM'ler ve mikrodalga hatları gibi aletler, *veri haberleşme aletleri (DCE)* olarak adlandırılırlar. Seri verileri gönderen ve alan terminaller ve bilgisayarlar ise *veri terminal aletleri (DTE)* olarak adlandırılırlar. DTE ve DCE arasındaki sinyal ve onay ihtiyaçlarına cevap olarak Electronic Industries Association (EIA), EIA standardı RS-232C'yi yayımladı. Bu standart, sinyaller için gerilim seviyelerini, onaylarını, ve 25 bacaklı bir konnektörü tanımlamaktadır. Şekil 11.27, 25 bacağın hepsinin isimlerini ve fonksiyonlarını göstermektedir. Birçok basit sistemlerde, yalnızca 1'den 8'e kadar olan bacaklar ve 20 nolu bacak kullanılmaktadır. Şekil 11.28 bu sinyalleri kullanarak bir CRT terminalin ve bir bilgisayarın MODEM lere nasıl bağlandığını göstermektedir. Her sinyalin akış yönünü belirten ok işaretlerine dikkat ediniz.

BACAK NUMARASI	ORTAK İSİM	RS-232-C İSİM	TANIM
1		AA	Koruma toprağı
2	TxD	BA	Gönderilen veri
3	RxD	BB	Alınan veri
4	$\overline{RTS}$	CA	Gönderme için istek
5	$\overline{CTS}$	CB	Gönderme için serbest
6	$\overline{DSR}$	CC	Veri kümesi hazır
7	GND	AB	Sinyal toprağı (Ortak dönüş)
8	$\overline{CD}$	CF	Alınan hat sinyal algılayıcı
9		-	(Veri kümesi testi için ayrılmış)
10		-	(Veri kümesi testi için ayrılmış)
11			Tanımlanmamış
12		SCF	İkincil alınan hat sinyal algılayıcısı
13		SCB	İkincil gönderme için serbest
14		SBA	İkincil gönderilen veri
15		DB	İletim sinyalelemanı zamanlaması (DCE kaynağı)
16		SBB	İkincil alınan veri
17		DD	Alıcı sinyal elemanı zamanlaması (DCE kaynağı)
18			Tanımlanmamış
19		SCA	İkincil gönderme için istek
20	$\overline{DTR}$	CD	Veri terminali hazır
21		CG	Sinyal kalitesi algılayıcı
22		CE	Halka gösterge
23		CH/CI	Veri sinyali hızı seçicisi (DTE/DCE kaynak)
24		DA	İletilen sinyal elemanı zamanlaması (DTE kaynağı)
25			Tanımlanmamış

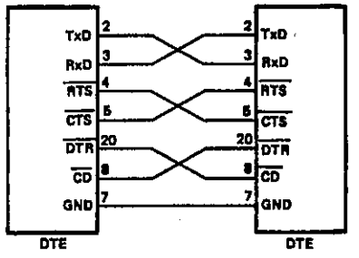
Şekil 11.27 RS-232C Sinyal adları ve bacak numaraları

Terminal ve MODEM arasındaki onay temelde şu şekilde çalışır. Terminal gücü açıldıktan ve terminal kendi test yordamlarından geçtikten sonra MODEM'e bir $\overline{DTR}$ gönderir. MODEM çalışmakta olduğunu bir $\overline{DSR}$ göndererek belirtir. Bu durumda bilgisayar tarafındaki MODEM aranır. Cevap-modundaki bu MODEM bir 2225 Hz taşıyıcı frekansla cevap verir. Terminale bağlı MODEM bu taşıyıcıyı aldığını, *taşıyıcı algılandı (CD)* sinyali ile terminale belirtir. Bu durumda terminal, MODEM'e *gönderme isteği* ($\overline{RTS}$) sinyalini gönderir. Uygun bir zamanlama aralığından sonra MODEM *gönderme için serbest* ($\overline{CTS}$) sinyali ile cevap verir. Daha sonra terminal TxD çıkışında seri veriyi gönderir. Veri almak için benzeri bir onay yapılır.

Değişik tipte RS-232 aletinin birlikte bağlandığında meydana gelen bir problem, Şekil 11.28'deki MODEM'leri kaldırıp terminalleri doğrudan bilgisayara bağlamakla gösterilebilir. Görebileceğiniz gibi her iki ünitede bacak 2'de veri göndermeyi denemektedir. Eğer doğrudan bağlanırlarsa bu iki çıkış birbiriyle mücadele edecektir. Çözüm, Şekil 11.29'da gösterildiği gibi iki konnektörü sinyallerin birbiri üzerinden çapraz geçeceği şekilde bir adaptör yapmaktır. Bu bağlantı *boş*MODEM olarak adlandırılır.



Şekil 11.28 MODEMler kullanarak bir terminalin bir zaman paylaşımlı bilgisayarla haberleşmesi



Şekil 11.29 İki RS-232C veri terminali tipi cihazları bağlamak için boş MODEM
***Not:* Birçok sistem bütün bu onaylama sinyallerini kullanmaz.**

Bütün RS-232 sinyalleri için gerilim seviyeleri şu şekildedir: Bir mantık yüksek veya *işaret*, yük altında -3 ile -15V arasında bir gerilimdir. Bir mantık alçak veya *boşluk*, yük altında +3 ile +15V arasında bir gerilimdir. Kullanılan tipik gerilimler +10 veya +12 Volttur. Komşu teller arasında karışmayı önlemek için yükselme zamanı ve düşme zamanı hızları 30V/µs ile sınırlandırılmalıdır. RS-232C 20000 Bd'lik en yüksek baud hızında, en uzun mesafe 50ft [15m] için tanımlanmıştır. Daha düşük baud hızları için 2000 ile 3000ft[610-915m] uzunluğundaki teller kullanılabilir.

RS-232C sinyal seviyelerini standart TTL veya CMOS ile bağlamak için sürücüler ve alıcılar kullanılmalıdır. MC1488 dört TTL'den RS-232C'ye sürücü içerir, ve MC1489 dört RS-232C'den TTL'ye alıcı içerir.

RS-422: RS-232C'nin gelişmişi, düşük-empedans fark sinyalleri kullanan EIA standardı RS-422'dir. Fark sinyalleri MC3487 gibi *fark hattı sürücüleri* tarafından yaratılır ve Bölüm 2'de anlatıldığı gibi bir bükülmüş çift tel üzerinde gönderilir. Fark sinyalleri MC3486 gibi *fark hattı alıcıları* tarafından alınır ve TTL mantık seviyelerine çevrilir. Fark hattı alıcıları bükülmüş çift teller üzerinde eşit olarak yaratılmış gürültüyü zayıflatma avantajına sahiptir. RS-422 en yüksek baud hızını 40ft [12m] bağlantı telleri için 10MBd'a veya 4000 ft [1220 m] bağlantı telleri için 100.000Bd'a yükseltir.

RS-423: RS-232C'nin bir başka gelişmişi EIA standardı RS-423'tür. Bu standart fark sürücüleri yerine düşük empedanslı, tek-sonlu sürücüler kullanarak daha uzun mesafelere daha hızlı baud hızlarında iletimi başarabilir. RS-423'ün empedansı 50Ω olarak belirtilmiştir ve bağlantılar için koaksiyel kablolar kullanılabilir. Standart mantık ailelerine bağlanmak için hat sürücü ve alıcıları gereklidir.

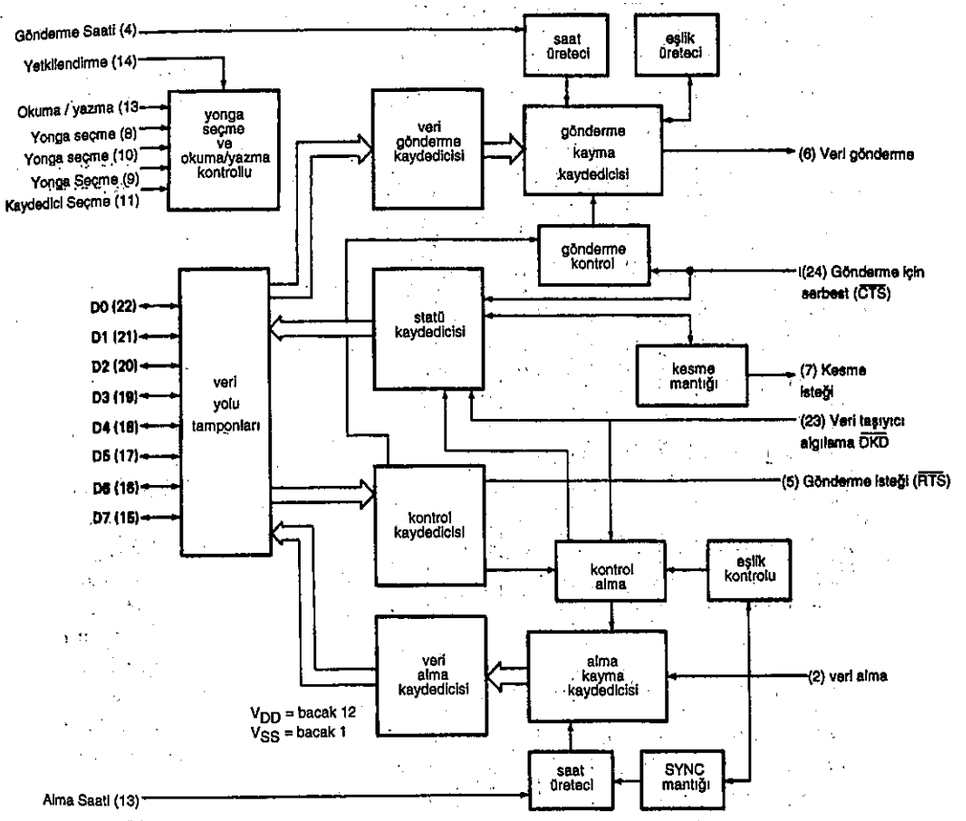
PROGRAMLANABİLİR SERİ PORT ENTEGRE DEVRELERİ:

MC6850 VE I825IA

MOTOROLA MC6850 : Motorola, bu cihazı *asenkron haberleşme arabirim adaptörü (ACIA)* olarak belirtir. Bir programlanabilir UART gibi çalışır. Şekil 11.30, 6850'nin bir blok diyagramını göstermektedir. Şekil 10.5 bu cihazın bir mikrobilgisayar sistemine nasıl bağlandığını göstermektedir. D0-D7 sistem veri taşıtına bağlanır. R/W sistemin R/W hattına ve ENABLE, DBE sinyaline bağlanır. Yonga seçme girişleri adres hatlarına veya kod çözücü çıkışlarına bağlanabilir. Kaydedici-seçme girişi genellikle sistem adres hattı A0'a bağlanır. Bu girişteki bir yüksek, eğer R/W alçak ise veri-gönderme yazmacını ve R/W yüksek ise veri-alma kaydedicisini seçer. Kaydedici-seçme girişindeki bir alçak seviye, eğer R/W alçak ise kontrol kaydedicinin, ve R/W yüksek ise statü kaydedicisini seçer. Bir sistemdeki cihazın taban adresi örneğin 8008H ise veri kaydedicinin adresi 8009H olacaktır. Kontrol ve statü kaydedicileri 8008H adresindedir.

6850, istenen baud hızının 1, 16 veya 64 katı olan gönderme ve alma saati girişleri gerektirir. VERİ GÖNDERME ve VERİ ALMA bacaklarına ilave olarak, 6850'leri MODEM'lere bağlamak için $\overline{CTS}$, $\overline{RTS}$ ve $\overline{DCD}$ (veri taşıyıcı algılama) sinyal bacaklarına sahiptir. 6850'nin kesme isteği çıkışı CPU'nun kesme isteği girişine bağlanabilir. Bu 6850'nin veri-alma yazmacında okunacak bir karaktere sahip olduğu veya veri-gönderme kaydedici boş olduğu durumlarda işlemciye kesme göndermesine imkan sağlar.

Bir 6850 kontrol yazmacına bir kontrol kelimesi gönderilerek ilk kullanıma hazırlanır. Şekil 11.31*a*, 6850 kontrol kelimesinin formatını göstermektedir. CR1 ve CR0 bitleri gönderme saati giriş frekansı ile çıkış band hızı arasındaki oranı belirtir. Gönderme saati frekansı 4800 Hz ve istenen band hızı 300 Bd ise, istenen bölme oranı 16 olacaktır. Bu durum CR1 ve CR0 bitlerini 1 programlayarak seçilebilir. Bu iki bitin 1 olduğu bir kontrol kelimesinin

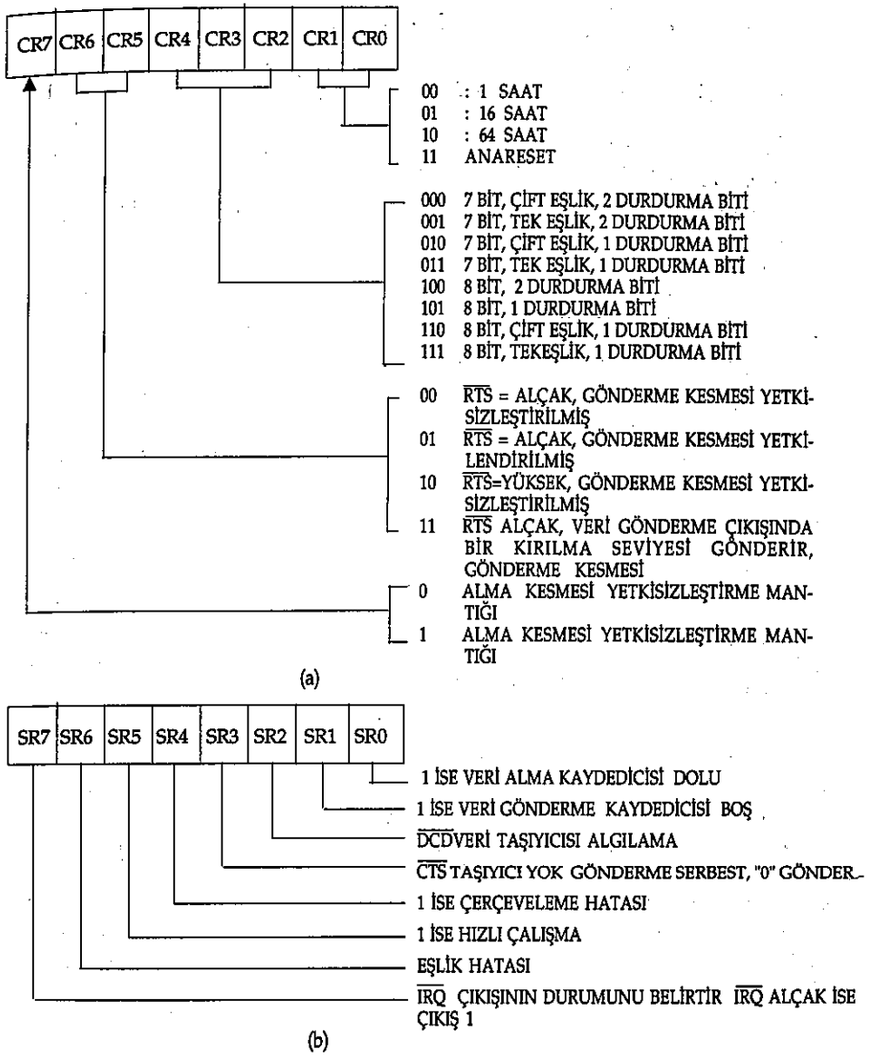


Şekil 11.30 MC6850'nin blok diyagramı (Motorola)

gönderilmesi cihazı resetler. Bu cihaz açıldığında gereklidir, çünkü 6850 bir donanım reset girişine sahip değildir. CR5, CR3 ve CR2 bitleri verinin gönderileceği formatı belirtir. CR5 biti verici kesme devresini yetkilendirir. Bu bitteki bir 1 veri-gönderme kaydedicisi boş iken 6850'nin CPU'ya bir kesme isteği göndermesine izin verir. Daha sonra bir kesme servis yordamı, gönderilecek bir sonraki karakteri bu kaydedicisine yazar. Bit CR6 alçak seviyede ise 6850'nin $\overline{RTS}$ çıkış sinyalini bir MODEM veya diğer bazı çevrebirimlere uygulamasına sebep olur. Kontrol kaydedicisinin CR7 biti alıcı kesme cevabını yetkilendirir veya yetkisizleştirir. Bit CR7 1 ise ve veri-alma kaydedicisinde okunmaya hazır bir karakter olduğunda 6850 CPU'ya bir kesme gönderir. Daha sonra bir kesme servis yordamı, karakteri CPU'ya okur.

Şekil 11.31*b*, 6850'nin statü kelimesinin formatını göstermektedir. Veri alma kaydedicisinde bir karakter olduğunu, veri-gönderme kaydedicisinin bir karakter almaya hazır olduğunu, veya bir MODEM'den taşıyıcı-algılama ve gönderme için serbest sinyalleri alındığını anlayabilmek için CPU, bu statü kelimesini okuyabilir. Statü kaydedicisini okumakla çerçeveleme hatası, hızlı çalışma hatası veya eşlik hatası gibi hata şartları belirlenebilir. Bir *çerçeveleme hatası* alınan karakterin yanlış sayıda veri bitleri veya durdurma bitlerine sahip olduğunu belirtir. Hızlı çalışma hatası, karakterlerin alındığını ve veri-alma kaydedicisine yüklendiğini fakat CPU tarafından okunmadıklarını belirtir. Bu yüzden bazı veri karakterleri kaybolur. Bir *eşlik hatası*, alınan karakterin eşliğinin karakter gönderildiği zamanki eşlikten farklı olduğunu belirtir. Statü kaydedicisinin SR7 biti 6850'nin IRQ çıkışının onaylanıp onaylanmadığını belirtir.

Karakterler 6850'den taramalı veya kesme tabanlı olarak okunabilir. Kesme tabanlı için CPU, 6850'yi kullanıma hazırlar ve 6850'den okunmaya hazır bir karakter olduğunu belirten bir kesme gelinceye kadar ana programa devam eder.Bir kesme servis yordamı karakteri okur ve kontrolu yeni bir kesme gelinceye kadar ana programa verir. Taramalı çalışma için, CPU, 6850'yi kullanıma hazırlar ve veri alma kaydedicisi dolu biti SR0 1 oluncaya kadar statü kaydedicisini üstüste okur. Daha sonra CPU karakteri okur, bu karakterin bir veri bloğunun son karakteri olup olmadığını kontrol eder, ve değilse SR0 yeniden yüksek oluncaya kadar beklemek için geri döner. Veri-alma kaydedicisinden bir karakter okumak SR0 bitini resetler. Şekil 11.32*a*, 6850'yi kullanıma hazırlamak ve yazının sonu karakteri 03H okunana kadar karakterleri taramalı olarak okumak için yazılan bir 6800 çevirici dil programını göstermektedir. Karakterler bellekte bir tablo içinde depolanmıştır.



Şekil 11.31 MC6850 Kontrol kelimesi ve statü kelimesi formatları
***a)* Kontrol kelimesi *b)* Statü kelimesi (Motorola)**

Karakterler, iletilmek için kesme veya taramalı tabanlı olarak 6850'ye gönderilebilir. Tarama tabanlı için, veri-gönderme kaydedicisindeki SR1 biti 1 oluncaya kadar statü kaydedicisi üstüste okunur. SR1=1 olunca bellekten bir

karakter okunur ve 6850'ye gönderilir. Veri gönderme kaydedicisine yazmak SR1 bitini resetler. Bir MODEM'e veri göndermek için statü kelimesi, bir karakter gönderilmeden önce, $\overline{DCD}$ biti SR2 ve $\overline{CTS}$ biti SR3'ün her ikisininde alçak olduğundan emin olmak için kontrol edilmelidir. Şekil 11.32b, 6850 veri-gönderme kaydedicisine bir karakter göndermeden önce, statü kaydedici bitleri SR3, SR2 ve SR1'i tarayan bir 6800 yordamını göstermektedir. 3 bitin istenen durumda olduğunu kontrol eden maskele ve karşılaştır dizisine dikkat ediniz.

```
        LDX     #$0100      ; BELLEK GÖSTERİCİYİ AYARLA
        LDAA    #00000011   ; ANA RESET İÇİN KONTROL KELİMESİ
        STAA    $8008       ; KONTROL KAYDEDİCİSİNE GÖNDER
        LDAA    #00000101   ; : 16 SAAT, 7 BİT, TEK EŞLİK
                            ; 2 DURDURMA BİTİ, RTS ALÇAK İÇİN
                            ; KONTROL KELİMESİ
                            ; GÖNDERME KESMESİ YETKİSİZLEŞTİRİLMİŞ
                            ; ALMA KESMESİ YETKİSİZLEŞTİRİLMİŞ
        STAA    $8008       ; KONTROL KAYDEDİCİSİNE GÖNDER
WAIT    LDAA    $8008       ; STATÜ KAYDEDİCİSİNİ OKU
        ANDA    #$01        ; RDRF HARİÇ HEPSİNİ MASKELE
        BEQ     WAIT        ; KARAKTER HAZIR OLUNCAYA KADAR BEKLE
        LDAA    $8009       ; KARAKTERİ OKU
        CMPA    #$03        ; YAZININ SONU MU?
        BEQ     EXIT        ; EVET, BİTTİ
        STAA    00,X        ; BELLEK TABLOSUNA DEPOLA
        INX                 ; GÖSTERİCİYİ ARTTIR
        BRA     WAIT        ; BİR SONRAKİ KARAKTER İÇİN BEKLE

EXIT
        MC6850'Yİ 8008H VE 8009H ADRESLERİNDE İLK KULLANIMA HAZIRLA-
        YAN VE YAZI SONU KARAKTERİ ALGILANINCAYA KADAR KARAK-
        TERLERİ TARAMALI OLARAK OKUYAN 6800 YORDAMI
```

(a)

```
WAIT    LDAA    $8008       ; STATÜ KAYDEDİCİSİNİ OKU
        ANDA    #$0E        ; SR3, SR2, SR1 HARİÇ HEPSİNİ MASKELE
        CMPA    #$02        ; TDRE=1, DCD=0, CTS=0 İÇİN KONTROL ET
        BNE     WAIT        ; HEPSİ HAZIR OLUNCAYA KADAR BEKLE
        LDAA    00,X        ; KARAKTERİ BELLEKTEN YÜKLE
        STAA    $8009       ; 6850 VERİ GÖNDERMEYE GÖNDER
        CPX     #$01FF      ; BLOK SONU MU?
        BEQ     EXIT        ; EVET, BİTTİ
        INX                 ; HAYIR, GÖSTERİCİYİ ARTTIR
        BRA     WAIT        ; BİR SONRAKİ HAZIR OLUNCAYA KADAR BEKLE
```


TDRE=1, $\overline{DCD}$=0, ve $\overline{CTS}$=0 İKEN MC6850 İLE KARAKTERLERİ TARAMALI OLARAK GÖNDERMEK İÇİN 6800 ÇEVİRİCİ DİL YORDAMI

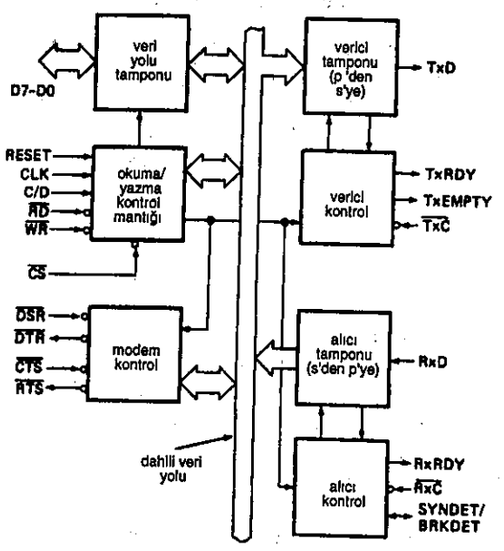
Şekil 11.32 MC6850'yi ilk kullanıma hazırlayan ve alma ve gönderme için tarayan MC6800 çevirici dil programı *a)* Alma *b)* Gönderme

8251A: EVRENSEL SENKRON-ASENKRON ALICI-VERİCİ

BACAK VE SİNYAL TANIMLARI: Çok yaygın kullanılan bir diğer programlanabilir seri port cihazı Intel 8251A'dır. Şekil 11.33, 8251A'nın blok diyagramını ve bacak tanımlarını göstermektedir. CLK, CPU saatine bağlanır. C/$\overline{D}$ sistem adres hattı *A0*'a bağlanır. *A0* alçak ise, veri-gönderme tamponuna ya da veri-alma tamponuna erişilir. A0 yüksek ise, kontrol veya statü kaydedicisine erişilir. $\overline{RD}$ ve $\overline{WR}$ girişleri her durumda hangisinin seçildiğini belirler. Örneğin, *A0*(C/$\overline{D}$) alçak iken $\overline{RD}$'nin alçak olması alma tamponunu seçer, ve $\overline{WR}$'nin alçak olması gönderme tamponunu seçer. Çoğu UART'larda olduğu gibi, istenen baud hızının 1, 16 veya 64 katı olan bir harici saat $\overline{TxC}$ ve $\overline{RxC}$ girişlerine uygulanmalıdır.

8251A *çift tamponludur*. Bunun anlamı, bir karakter dışarı kaydırılırken bir diğerinin tutma tamponuna yüklenebilmesidir. 8251A'dan gelen TxRDY çıkışı, tutma tamponunun CPU tarafından gönderilecek bir sonraki karakter için hazır olduğunu belirtir. TxEMPTY çıkış sinyali, 8251A'nın CPU'dan aldığı bütün karakterleri gönderdiğini belirtir. RxRDY, alıcı tamponunda CPU'nun okuması için bir karakterin hazır olduğunu belirtir. Doğal olarak, bu karakter okunmadan önce 8251A bir sonraki karakteri alırsa, ilk karakterin üzerine yazılmış olur.

Yonganın SYNDET/BRKDET bacağının birçok kullanımı vardır. Asenkron modda eğer alıcı hattı iki karakter zamanından daha uzun alçak seviyede kalırsa bu bacak yükseğe gider. Bu daha sonra veri akışında bir kırılmayı belirtir. Senkron mod da bu bacağın yükseğe gitmesi bir sync karakterinin bulunduğunu ve bir veri bloğunun gelmekte olduğunu belirtir.



(a)

8251'in Bacak Fonksiyonları

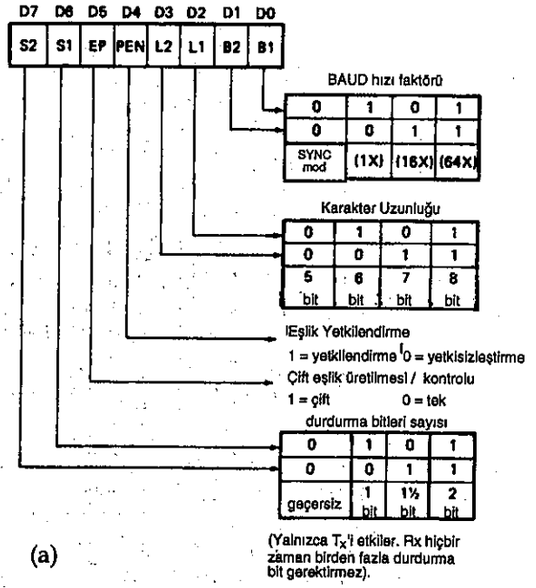
D7-D0	Veri taşıtı (8 bit)
C/D	Kontrol veya veri yazılacak yada okunacak
RD	Veri okuma komutu
WR	Veri yazma yada kontrol komutu
CS	Yonga seçme
CLK	Saat darbeyi (TTL)
RESET	Reset
TxC	Gönderici saati
TxD	Gönderici verisi
RxC	Alıcı saati
RxD	Alıcı verici
RxRDY	Verici hazır (CPU için karaktere sahip)
TxRDY	Verici hazır (CPU'dan gelen karakter içinhazır)
DSR	Veri kümesi hazır
DTR	Veri terminali hazır
SYNDET/BD	Sync algılama / kırılma algılama
RTS	Veri gönderme isteği
CTS	Veri gönderme serbest
TxEMPTY	Verici boş
V_{cc}	+5V kaynak
GND	Toprak

(b)

Şekil 11.33 ***a)*** **8251'in Blok diyagramı** ***b)*** **8251'in Bacak fonksiyonları (Intel Corporation).**

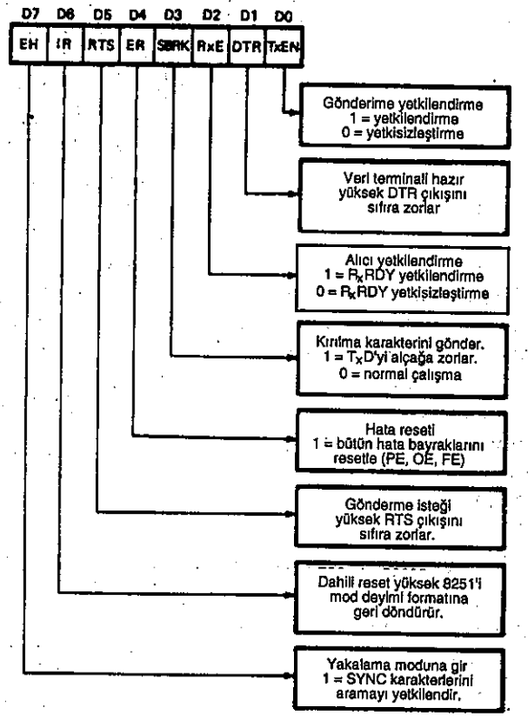
8251A'YI İLK KULLANIMA HAZIRLAMA: 8251A'yı ilk kullanıma hazırlamak içinelemanın kontrol kaydedici adresine iki kontrol kelimesi gönderilmelidir. 11.34 bu kelimelerin ve statü kelimesinin formatını göstermektedir. Bir donanım ya da yazılım resetinden sonra ilk olarak mod kelimesi gönderilmelidir. Bu kelime gönderilecek ve alınacak verinin formatını ayarlar. Mod kelimesinin *B2* ve *B1* bitleri her ikisi de 0 olduklarında senkron modu belirtir. Senkron modda, hızı TxC ve RxC giriş frekanslarına eşittir. Asenkron veri transferi için mod kelime bitleri B2 ve B1, TxC giriş frekansı ve gönderilen verinin baud hızı arasındaki oranı ayarlar. 4800 Hz'lik TxC girişi ve 16'lik baud hızı faktörü 300Bd'lik bir iletim üretir. Şekil 11.34a'da gösterildiği gibi mod kelimesindeki diğer bitler, karakter içindeki veri bitleri sayısını, eşlik olması veya olmamasını, tek ve çift eşliği, ve durdurma bitlerinin sayısını belirler.

Mod kelimesi bir 8251A'ya gönderildikten sonra, bir *komut* kelimesi gönderilir. Şekil 11.34*b* bu kelimenin formatını göstermektedir. Bu kelimenin *D0* bitindeki bir 1 verici bölümünü yetkilendirir. Yetkilendirmeden önce TxD çıkışı sürekli yüksek durumdadır. Komut kelimesinin *D2* biti RxRDY bitini yetkilendirir, böylece alıcı tamponu hazır bir karaktere sahip olunca yükseğe gider. Bu bit yetkisizken dahi karakterler alıcı tamponunun içine kaymaya devam ederler. Bu yüzden, ilk kullanıma hazırlanmadan sonra alıcı tamponundan okunan ilk karakter istenmiyen bir karakter olabilir ve kullanılmamalıdır. *D1* bitindeki bir yüksek *veri-terminali-hazır* çıkış sinyalini (DTR) onaylar, ve *A5* bitindeki bir yüksek gönderme için istek (RTS) çıkış sinyalini onaylar. Bu sinyaller cihazın hazır olduğunu ve geri göndermek istediğini belirtmek için bir MODEM'e gidebilir. *D6* bitindeki bir 1 8251A'yı resetler, böylece kontrol kaydedici adresine gönderilen bir sonraki kelime mod kelimesi gibi işlem görür. Bu özellik, bir program sırasında sistem reseti yapmadan giriş ve çıkış veri formatlarının değiştirilebilmesine imkan verir. *D3* bitinde bir 1 ile gönderilen komut kelimesi 8251A'nın TxD çıkışında sürekli alçak (kırılma) seviye olmasına sebep olur. Bazı durumlarda bu, alıcı sisteme mesajın sonu olduğunu belirtmek için kullanılır.



(a)

(Yalnızca T_x'i etkiler. Rx hiçbir zaman birden fazla durdurma bit gerektirmez).

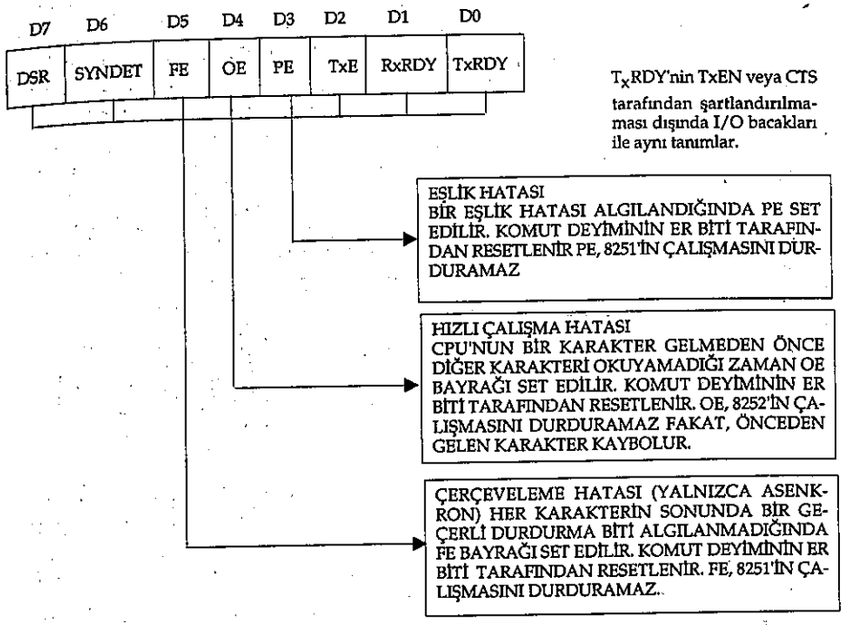


(b)

Şekil 11.34

***a)* 8251 mod sözcüğü formalı**

***b)* 8251 açıklama sözcüğün formalı**



Şekil 11.34 *c)* 8251 statü sözcüğün formatı (Intel Corporation)

KESME MODU: 8251A, bir mod kelimesi ve bir komut kelimesi gönderilerek ilk kullanıma hazırlandıktan sonra, CPU'nun vericisine bir karakter yazması ya da alıcı tamponundan bir karakter okuması için hazır olup olmadığı kontrol edilmelidir. Bu kesme veya tarama tabanlı olarak yapılabilir. 8251A'nın TxRDY çıkışı CPU'nun bir kesme çıkışına bağlandıysa, ana program yalnızca verici bir sonraki karakter için hazır olduğunda kesilebilir. Benzer şekilde eğer 8251A RxRDY çıkışlı bir CPU kesme girişine bağlıysa, ana program yalnızca alıcı tamponunun CPU tarafından okunacak bir karaktere sahip olduğunda kesilebilir. Bir kesme servis yordamı karakteri okuyabilir, belleğe depolar ve bir sonraki karakter alıcı tamponunda hazır oluncaya kadar ana programa döner.

TARAMALI MOD: Karakterleri tarama tabanlı göndermek veya almak için 8251A statü kaydedicisi okunur, ve verici yada alıcının hazır olduğunu görmek için uygun bitler kontrol edilir. Şekil 11.34*c*, 8251A statü kaydedicisinin formatını göstermektedir. Statü baytı, mod ve komut kelimelerinin gönderildiği aynı kontrol adresinden okunur. Alıcı tamponunun okunacak hazır bir

karaktere sahip olup olmadığını belirlemek için bir program döngüsü statü kaydedicisini, *D1* bitini yüksek buluncaya kadar üstüste okur.Daha sonra CPU döngüden çıkar ve karakteri alıcı tamponundan okur. Statü kaydedicisinin *D5*, *D4* ve *D3* bitleri bir çerçeveleme hatası, hızlı çalışma hatası, veya bir eşlik hatası olup olmadığını görmek için kontrol edilir. Eğer hata varsa, gönderici cihaza veriyi yeniden göndermesi için bir mesaj gönderilebilir.

Şekil 11.35*a*'da, bir 8251A'nın 50H (veri) ve 51H(Kontrol/statü) adreslerinde ilk kullanıma hazırlayan bir 8080A/8085A çevirici dil programı gösterilmektedir. 1982 öncesi bazı 8251A'nın bir küçük problemi vardır; bir donanım sistem resetine herzaman tam olarak cevap veremezler ve bu yüzden her zaman doğru olarak kullanıma hazırlanamazlar. Şekil 11.35a'daki program bir çözüm göstermektedir. Yongaya bir sahte mod kelimesi, bir veya iki boş (00H), ve bir yazılım reset komutu (40H) gönderilir. Hiçbir durumda bu fazladan deyimlerin bir zararı yoktur. Yonga istenen mod ve komut kelimelerini doğru olarak kabul eder.

Şekil 11.35*a*'da gösterilen mod kelimesi (4F): veri formatını; baud hızı faktörünü 64, 8 veri biti, eşlik yok, ve 1 durdurma biti olarak ayarlar. Komut kelimesi (27H) verici ve alıcıyı yetkilendirir, $\overline{DTR}$'yi ve $\overline{RTS}$'yi onaylar. Daha sonra, Şekil 11.35a'daki program statü kaydedicisini RxRDY biti D1'i yüksek buluncaya kadar üstüste okur ve kontrol eder. Alıcının bir karakterle hazır olduğunu bulduğunda, herhangi bir hatanın algılanıp algılanmadığını bulmak için statü kaydedicisini yeniden okur. Hata yoksa karakteri okur ve bunun mesajın son karakteri olup olmadığını kontrol eder. Okunan karakter yazı sonu karakteriyse (03H), program ana programa geri döner. Yazı sonu karakteri değilse, program bellek göstericiyi arttırır ve bir sonraki karakter hazır oluncaya kadar statü kaydedicisini taramaya geri döner.

Şekil 11.35*b*, 8251A statü kaydedicisini veri kümesi hazır sinyali alınıncaya ve verici hazır olana kadar tarayan bir çevirici dil döngüsünü göstermektedir. Bir kelime içinde 2 ya da daha fazla biti kontrol ederken, ANI ile maskeleme ve CPI ile karşılaştırmanın her ikisinin de kullanılmasının gerektiğine dikkat ediniz.


```
        MVI   A,80H     ; KONTROL ADRESİNE
        OUT   51H       ; SAHTE MOD
        MVI   A,00H     ; KONTROL ADRESİNE
        OUT   51H       ; BOŞ
        MVI   A,00H     ; KONTROL ADRESİNE
        OUT   51H       ; BOŞ
        MVI   A,40H     ; KONTROL ADRESİNE
        OUT   51H       ; RESET KOMUTU
        MVI   A,4FH     ; MOD KELİMESİ
        OUT   51H
        MVI   A,27H     ; KOMUT KELİMESİ
        OUT   51H
LOOP    IN    51H       ; OKUMA STATÜSÜ
        ANI   02H       ; RXRDY İÇİN KONTROL
        JZ    LOOP      ; HAYIR, BEKLE
        IN    51H       ; EVET, HATALAR
        ANI   38H       ; İÇİN KONTROL ET
        JNZ   ERROR     ; EVET, HATA YORDAMINA GİT
        IN    50H       ; HAYIR, KARAKTERİ OKU
        CPI   03H       ; SON KARAKTER Mİ?
        JZ    EXIT      ; EVET, DEVAM ET
        INX   H         ; HAYIR, GÖSTERİCİYİ ARTTIR
        JMP   LOOP      ; BİR SONRAKİ KARAKTERİ BEKLE
```

(a)

Şekil 11.35 8251A'yı kullanıma hazırlayıp, veri gönderen ve alan 8080A/8085A çevirici dil yordamı *a)* 8251A'yı kullanıma hazırlama ve veriyi tarama tabanlı olarak alma

```
LOOP    IN    51H       ; STATÜ KAYDEDİCİSİNİ OKU
        ANI   81H       ; DSR+TXRDY HARİÇ HEPSİNİ MASKELE
        CPI   81H       ; İKİSİDE BİRE EŞİT Mİ?
        JNZ   LOOP      ; HAYIR, BEKLE
        MOV   A, M      ; EVET, KARAKTERİ YÜKLE
        OUT   50H       ; 8251A'YA YAZ
```

(b)

Şekil 11.35 *b)* Veriyi tarama tabanlı gönder

YAZILIM KULLANARAK SERİ GİRİŞ VE ÇIKIŞI

CPU'nun, seri veriyi gönderirken veya alırken başka bir iş yapmasının beklenmediği küçük sistemlerde, bütün süreç UART yerine program yordamlarıyla yapılabilir. Veri, paralel port hatlarından veya 8085A üzerindeki 51D ve 50D gibi tek bitli portlardan girebilir ve çıkabilir. Zamanlama, gecikme döngüleriyle yapılır.

Bir karakterin gönderilişi şu şekilde yapılabilir: Port hattını çıkış olarak kullanıma hazırladıktan sonra en az 2 bit zamanı kadar yüksek gönder. Bir alçak başlangıç biti gönder ve 1 bit zamanı gecikme ver. Daha sonra ilk veri bitini pozisyonuna getirip çıkışa ver. 1 bit gecikme zamanı ver, ve bir sonraki biti çıkış port hattına gönder. Bütün veri bitleri gönderilinceye kadar bu süreci tekrarla ve en sonunda 1 veya 2 durdurma biti gönder.

Seri port hattında veri alınması benzeri bir yolla yapılabilir. Paralel port hattı, bir başlangıç biti algılanıncaya kadar defalarca taranır. 1/2 bit zamanı kadar bir gecikmeden sonra port hattının hâlâ alçak olup olmadığı kontrol edilir. Diğer bir deyişle algılanan ilk alçak seviyenin gerçekten geçerli bir başlangıç biti mi yoksa yalnızca bir gürültü darbesi mi olduğu kontrol edilir. Başlangıç biti geçerli ise, program ilk veri bit zamanının ortasına kadar 1 bit zamanı gecikme yapar. Daha sonra bit okunur. Okunan bit bir sonraki bite yer açmak için döndürülür. Bir diğer 1 bit zamanı gecikmeden sonra bir sonraki bit okunur ve döndürülür. Bu işlem bir karakterdeki bütün veri bitleri okununcaya kadar devam eder. Her veri bitini bit zamanının ortasında okumak, veri bitleri yükselen veya düşen kenarlara yakınken okunduğu zaman oluşabilecek hataları azaltır.

SENKRON SERİ VERİ İLETİMİ

Asenkron haberleşmede her karakterin başlangıcını belirtmek için bir başlangıç biti ve karakterin sonunu tanıtmak için en azından 1 durdurma biti gereklidir. Bunun anlamı 8-bitlik veri karakteri için en azından 10 bit gönderilmesidir. Diğer bir deyişle, başlangıç ve durdurma bitleri zamanın yüzde 20'sini boşuna harcarlar. Fakat, verici ve alıcı aynı zamanda başlayabilirlerse başlangıç ve durdurma bitlerine gerek kalmaz. Alıcı, otomatik olarak aldığı her 8 bitin bir veri karakteri olduğunu bilir. Verici ve alıcıyı bu şekilde birlikte kilitleme senkron haberleşmenin temelidir. *Senkron veri iletimi* için birçok format ve protokol vardır. Detaylı bilgi için bölümün sonundaki referanslara bakınız. Biz burada en yaygın iki formatı tanımlayacağız.

BISYNC: Çok kullanılan bir seri protokol IBM'in *ikilik senkron protokolu* veya BISYNC'dir. Şekil 11.36*a*, BISYNC'nin mesaj formatını göstermektedir. Veri gönderilmiyorsa hat bekleme durumundadır. Bekleme durumunda

sürekli 1'ler gönderilir. Bir mesajın başladığını işaret etmek için verici daha önceden üzerinde anlaşmaya varılmış iki veya daha fazla "sync" karakterini (örneğin ASCII 10010110) gönderir. Alıcı bunları kendi saatini vericinin saati ile senkronize etmek için kullanır. 8251A gibi bir USART, sync karakterlerini kullanır ve onları yok eder. Bir *başlık başlangıcı* (SOH) karakteri, etiketler, kontrol kodları ve adreslerden oluşan bir *başlığın* geldiğini belirtir.

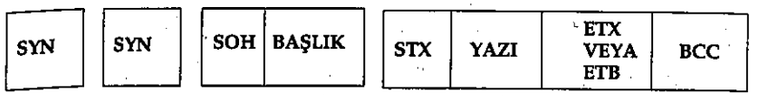
Yazı başlangıcı karakteri (STX), bir 256 baytlık veri bloğunun başlangıcını belirtir. *Yazı sonu* (ETX) karakteri veri karakter bloğunun bittiğini belirtir. Blok kontrol karakterleri verici tarafından düzenli olarak fazladan gönderilen karakterlerdir, böylece alıcı alınan veri bloğunda hata olup olmadığını kontrol eder. Intel 8251A BISYNC haberleşmeyi başarabilir.

BISYNC genellikle yarım-dubleks iletim için kullanılır. Verici bir blokluk seri gönderir ve alıcının hatayı kontrolu sırasında bekler. Eğer alıcı hata algılamazsa, *vericiye bir alındı* (ACK) sinyali gönderir. Sonra verici, bir sonraki veriyi gönderir. Verici yaklaşık yüzde 50 zamanını ACK sinyallerini beklemekte geçirir. Seri iletim için daha verimli bir protokol SDLC'dir.

SDLC : BISYNC bir *bayt-kontrollu protokoldur* (BPC). Bunun anlamı SOH, STX veya ETX gibi karakter kodları veya baytları mesajın parçalarını belirtmek için kullanılır. IBM'in *senkron veri link kontrolu* (SDLC) ise bir bit-yönlendirmeli protokoldur. Anlam mesaj içindeki bitlerin sayısı ve pozisyonu ile belirlenir. Şekil 11.36*b*, SDLC'nin mesaj formatını göstermektedir. Mesajlar alanlardan oluşan çerçeveler halinde gönderilir.

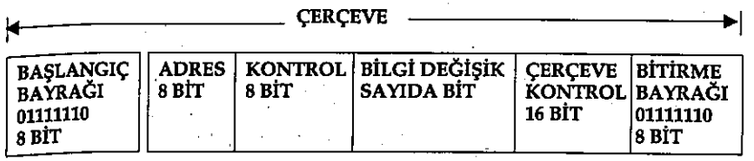
Mesajlar arasında hat sürekli 1'ler ile bekleme durumuna girer. Mesajın başlangıcı, *başlangıç bayrağı*olarak bilinen bir belirli bit dizisiyle belirtilir. Daha sonra alıcı istasyonun 8-bitlik adresi gönderilir. Sonrada 8-bitlik bir kontrol kelimesi gönderilir. Bu kontrol kelimesindeki her bitin özel bir anlamı vardır. Takip eden bilgi alanında istenilen sayıda bit olabilir.

Her çerçevede 10000 veya 20000'e kadar bit gönderilebilir. Veri bitlerinden sonra 16 bitlik dönemli, fazladan karakterler gönderilir, böylece alıcı hataları kontrol edebilir. Çerçeve, *bitirme bayrağı* dizisi 01111110 ile biter. Mesaj içindeki veri bitlerinin bitirme bayrağı olarak algılanmaması için SDLC, vericilerin peşpeşe gelen beş 1'den sonra bir 0 koymasını ister. Alıcı bu ilave 0'ları otomatik olarak çıkarır. SDLC sistemleri tam-dubleks haberleşme kullanırlar. Yalnızca hata olması durumunda alıcı, vericiye veriyi yeniden göndermesini söyler. Intel 8273A bir mikrobilgisayar-yol-uyumlu SDLC'dir.



SERİ VERİ AKIŞ YÖNÜ

(a)



(b)

Şekil 11.36 a) BISYNC mesaj formatı b) SDLC mesaj formatı

BİR CRT MONİTORUNU BİR MİKROİŞLEMCİYE BAĞLAMA

CRT'ler mikroişlemci programlarını, adresleri ve verileri göstermek için uygun bir yoldur. Bu bölümde sayıların, harflerin ve grafik göstergelerin CRT ekranında nasıl üretildiklerini anlatacağız. CRT *gösterge* ünitesi ya da bir *video monitoru* özelleşmiş bir TV olduğundan, konuya TV çalışmasının basit bir tekrarı ile başlayacağız.

TELEVİZYONUN ÇALIŞMASI

1. Bölümde anlatıldığı gibi, bir CRT ekran üzerindeki görüntüler bir elektron ışınının tüp yüzeyinin fosfor kaplı içtarafını taraması ile üretilir. Işın ekranın sağ kenarına ulaştığında karartılır ve yeniden başlamak için hızla ekranın sol tarafına döner. Eğer ışın yatay olarak ileri geri tararken yavaş yavaş ekranın üst kısmından altına doğruda kaydırılırsa tüm ekran aydınlatılmış olarak görülür. Işın ekranın alt kısmına ulaştığında karartılır ve yeniden başlamak üzere hızla ekranın üst tarafına döner. Bu görüntü raster olarak adlandırılır.

Aydınlık ve karanlık bölgelerden oluşan bir resim üretmek için bir video kuvvetlendiricisi tarama sırasında ışının şiddetini değiştirir. Belirli bir noktada ışını açmak aydınlık bir nokta üretir ve ışını kapatmak karanlık bir nokta üretir.

Yatay tarama frekansı, düşey tarama frekansı ve video bilgisi kararlı bir görüntü vermek için senkronize olmalıdır. Amerika Birleşik Devletleri'ndeki

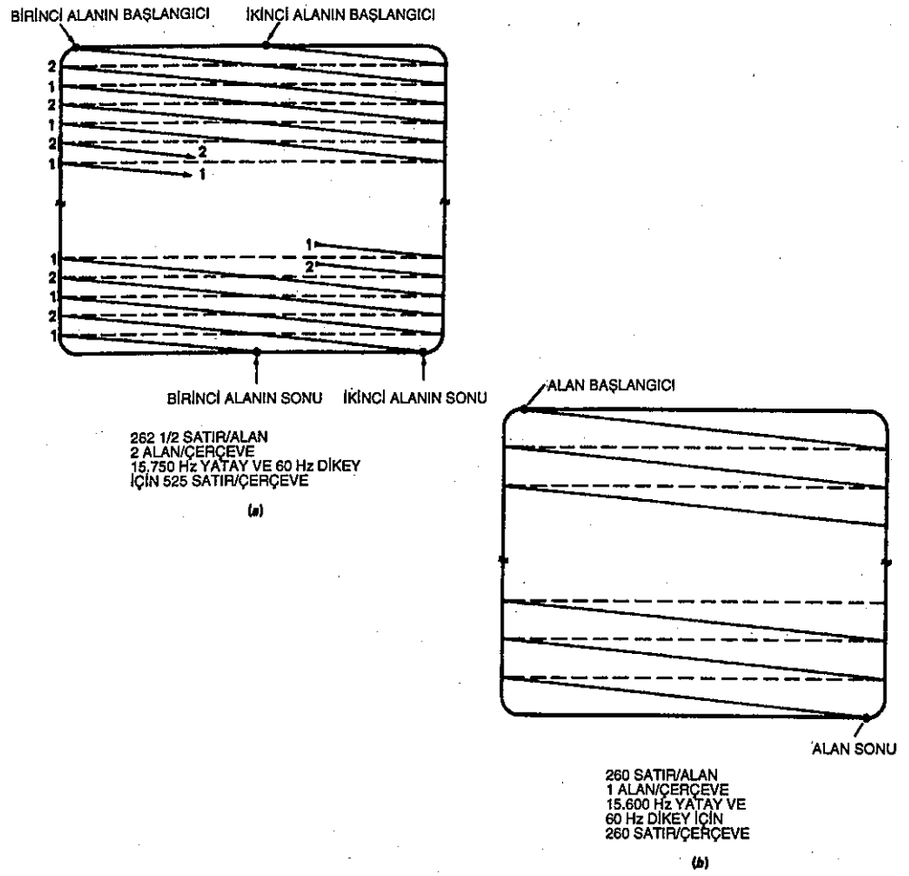
siyah-beyaz TV'ler 15,750 Hz'lik yatay tarama frekansı, ve 60 Hz'lik düşey tarama frekansı kullanır. Bunun anlamı ışının, her düşey tarama için 262.5 yatay *tarama* yapmasıdır. Işının ekranı yukarıdan aşağıya bir defa taraması alan olarak adlandırılır. Her saniyede altmış alan taranır. Daha iyi resim çözünürlüğü elde etmek için TV'ler geçmeli tarama kullanırlar. Şekil 11.37*a*'da gösterildiği gibi *geçmeli taramanın* anlamı bir alanın tarama satırlarının kaydırılması ve bir sonraki alanın satırları ile sıralı olarak geçişli olmasıdır. Her iki alandan sonra tarama dizisi tekrar eder. Böylece iki alan tambir resim ya da çerçeve oluşturur. Çerçeve hızı saniyede 30'dur.

Bilgisayar çıkışı için kullanılan CRT üniteleri , Şekil 11.37*b*'de gösterildiği gibi genellikle geçmesiz taramaya sahiptirler. 15600Hz'lik yatay tarama frekansı ve 60 Hz'lik düzey tarama frekansı çerçeve başına 260 satır verir. Bu durumda çerçeve hızı ve alan hızı, herbiri için 60 Hz'dir.

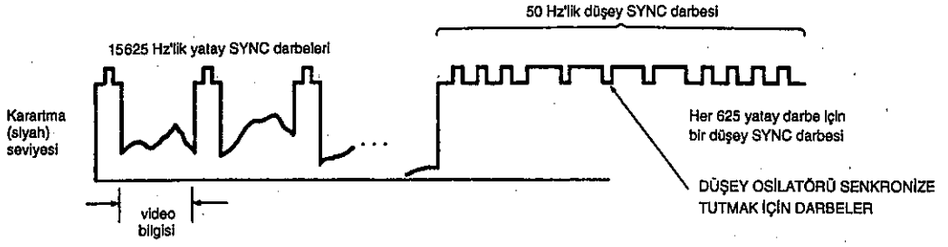
Bilgisayar çıkışlarını okumak için kullanılan CRT terminali bir video monitoru olarak adlandırılır. Bir video monitoru yalnızca bir düşey osilatöre, bir yatay osilatöre, bir video kuvvetlendiriciye, bir CRT ve gerekli bütün saptırma devrelerine sahiptir. Giriş olarak, yatay sync darbelerini, düşey sync darbelerini, geri dönüşler sırasında ışını karartan karartma darbelerini ve video şiddet bilgilerini içeren kompozit video sinyali gereklidir. Şekil 11.38, tipik bir kompozit video sinyalini göstermektedir. Bir şekil içinde göstermek zordur, fakat her 260 yatay sync darbeleri için bir düşey sync darbesi vardır. Şimdi karakterleri ekranda üretmek için kompozit video sinyalinin nasıl elde edildiğini anlatacağız.

BİR SAYFA KARAKTERİN CRT ÜZERİNDE ÜRETİLMESİ

KARAKTER ÜRETİCİLERİ: Her karakter ekran üzerinde açık ve koyu noktaların bir dizisi olarak üretilir. İstenen karakterlerin dizileri *karakter üretici* olarak adlandırılan, maskeli programlanmış bir ROM'a depolanmıştır. Yaygın bir karakter üretici Motorola MC6571'dir. Şekil 11.39'da gösterildiği gibi, MC6571 her karakter için bir 7x9 nokta matrisi kullanır. Her karakterin 7x9'luk temel matrisi, küçük harflerin azalan kuyrukları ile bir satır üzerinde uygun pozisyonlarda görülmeleri için kaydırılır. Hem büyük hem de küçük harfler kullanılıyorsa enaz 13 satır kullanılmalıdır (12 satır karakterler ve 1 satır boşluk için). MC6571 büyük ve küçük harfleri, sayıları, Yunan harflerini ve birçok sembolü gösterebilir.

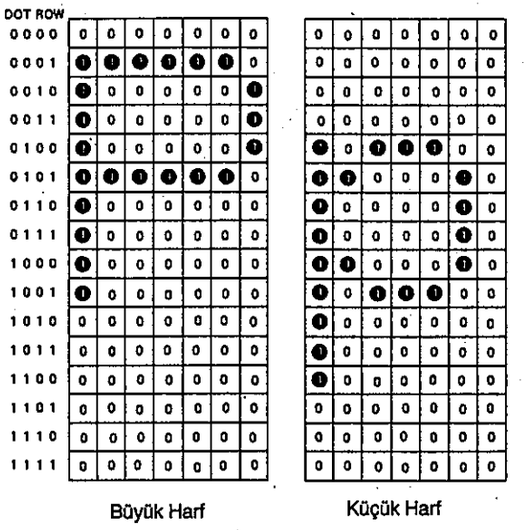


Şekil 11.37 CRT tarama dizileri *a)* Geçmeli *b)* Geçmesiz



Şekil 11.38 Kompozit video sinyali

Bir karakter üretici, örneğin MC6571, şu şekilde çalışır. Gösterilecek karakterin ASCII ya da diğer kodu karakter üreticinin veri girişlerine uygulanır. Satır adresi dört satır girişine uygulanır. Bu durumda 6571, yedi paralel hatlı çıkışından karakterin adreslenmiş satırı için nokta dizisini gönderir. 74165 gibi bir paralel-giriş, seri-çıkış kayma kaydedici, paralel nokta kodunu CRT ışınını açıp kapamak için gerekli seri forma çevirir. Bu kayma kaydedici için kullanılan saat her darbede bir nokta kaydırdığı için *nokta saati* olarak adlandırılır.



Şekil 11.39 Karakter üretici formatları : Motorola MC6571

GÖSTERGE FORMATI VE RAM: Karakterler CRT üzerinde değişik sayfa formatlarında gösterilebilir. Yaygın formatlar herbiri 32 karakterlik 16 satır ve herbiri 80 karakterlik 24 satırdır. Aşağıda anlatılacak konular için herbiri 32 karakter olan 16 satır formatı kullanılmıştır. Bu bir zamanda toplam 512 karakter gösterir.

Gösterilecek 512 karakterin ASCII kodları RAM içinde depolanır ve gerektiği zaman okunur. RAM, CPU belleğin bir parçası olabilir veya yalnızca CRT gösterge için ayrılmış bir bellek olabilir.

ZAMANLAMA : Bir satırlık karakteri gösterme işlemi şu şekilde ilerler. Satırın ilk karakterinin ASCII kodu bellekte adreslenir ve karakter üretici veri girişlerine okunur. Karakter üretici içindeki ilk nokta satırı adreslenir. Karakter üretici tarafından üretilen paralel nokta kodu, her seferinde bir nokta olarak dışarı kaydırılır. CRT ışın, ekran boyunca kayarken ilk karakterin üst satırının nokta dizisini üretir.

Video kuvvetlendiriciye verilmesi gereken bir sonraki bilgi ikinci karakterin üst satırının noktalarıdır. Bu bir sonraki karakteri adresleyip, okuyarak ve karakter üretici girişlerine uygulayarak yapılır. Kayma kaydedicisi, ikinci karakterin üst satırının nokta dizisini video kuvvetlendiriciye gönderir. Bu işlem üst satırdaki 32 karakterin hepsi için tekrarlanır. Işın üst satırdaki 32 karakterin noktalarının hepsini taradıktan sonra karakter üreticiye uygulanan nokta satırı, adres girişi, her karakter için ikinci nokta satırını elde etmek için arttırılır. 32 karakterin herbiri yeniden bellekten okunmalıdır. 6571'de karakterlerin her satırını ekranda göstermek için her karakter bellekten 13 defa okunmalıdır.

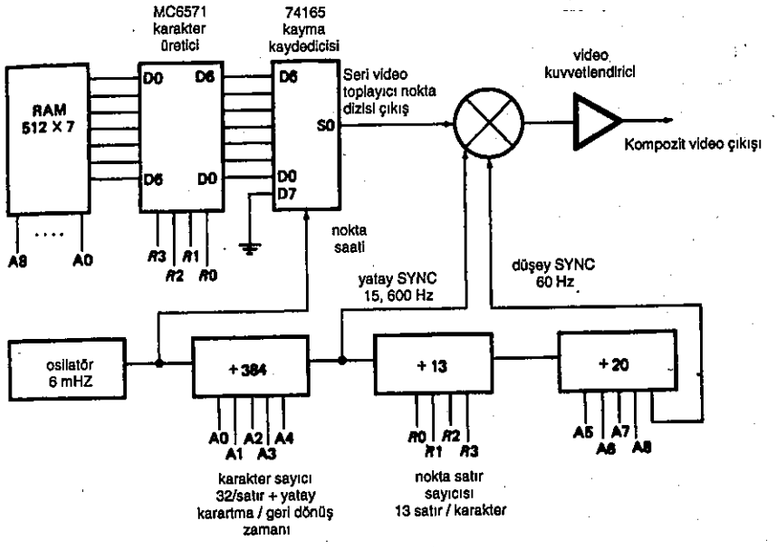
Karakterlerin birinci satırının taranması tamamlandıktan sonra, ikinci satırın ilk karakteri bellekten okunmalı ve nokta koduna çevrilmek üzere karakter üreticiye gönderilmelidir. Kayma kaydedicisi nokta kodunu video kuvvetlendiriciye seri olarak gönderir. Bu satır ve geri kalan 14 satır için aynı işlemler tekrarlanır. Herbiri 32 karakterden oluşan 16 satırın hepsi taranınca ışın karartılır ve yeniden başlamak üzere ekranın üst kısmına geri dönülür.

Şekil 11.40, bütün bu zamanlama için bir dizi sayıcının nasıl kullanıldığını bir blok diyagramla göstermektedir. En yüksek frekans, kayma kaydedicisinden noktaların çıkışını ayarlamak için kullanılan 6Mhz'lık kristal osilatördür. *A0-A4* bir sonraki karakterin adresini değiştirmekte kullanılır. Bu beş adres hattı bir satır için gerekli 32 karakteri adresleyebilir. 384 ile bölme kısmının çıkışı 15600Hz'lik yatay tarama frekansıdır. Işının ekranı her taraması için bir darbe üretir, böylece nokta satır sayıcısının saati olarak kullanılır. Her darbe, karakter üreticinin girişlerine bağlanmış dört sayıcı çıkışını arttırır, böylece bir sonraki nokta satır dizisi adreslenmiş olur.

Bir satırlık karakterlerin 13 tanesinin hepsi gönderildikten sonra, 13 ile böl sayıcısından bir darbe üretilir. Bu, karakter satır sayıcısını bellek içindeki bir sonraki karakterler satırının başlangıç adresine artırır. Karakter satır sayıcısı bir 20 ile bölmedir. Sayılardan 16 tanesi, 16 karakter satırı içindir. Diğer dört satır sayıcısı düşey geri dönüş zamanı içindir. 20 ile bölme devresinin çıkışı düşey sync darbesi için gerekli olan 60 Hz'dir.

Sayıcı dizisine ilave olarak yatay ve düşey karartma darbelerini, imleci ve göstergenin akışını üretmek için zamanlama devreleri gereklidir. Bir imleç yanıp sönen bir kutu ya da alt çizgisidir ve bir sonraki karakterin ekranda nere-

ye yazılacağını gösterir. Tam fonksiyonlu bir CRT karakter gösterge çok sayıda devre elemanı gerektirir. Fakat büyük ölçekli entegrasyon (LSI) bu devrelerden birçoğunun CRT kontrolörü olarak adlandırılan birtek entegre devre içine konulmasını mümkün kılmıştır.



Şekil 11.40 Video karakter üreticinin blok diyagramı

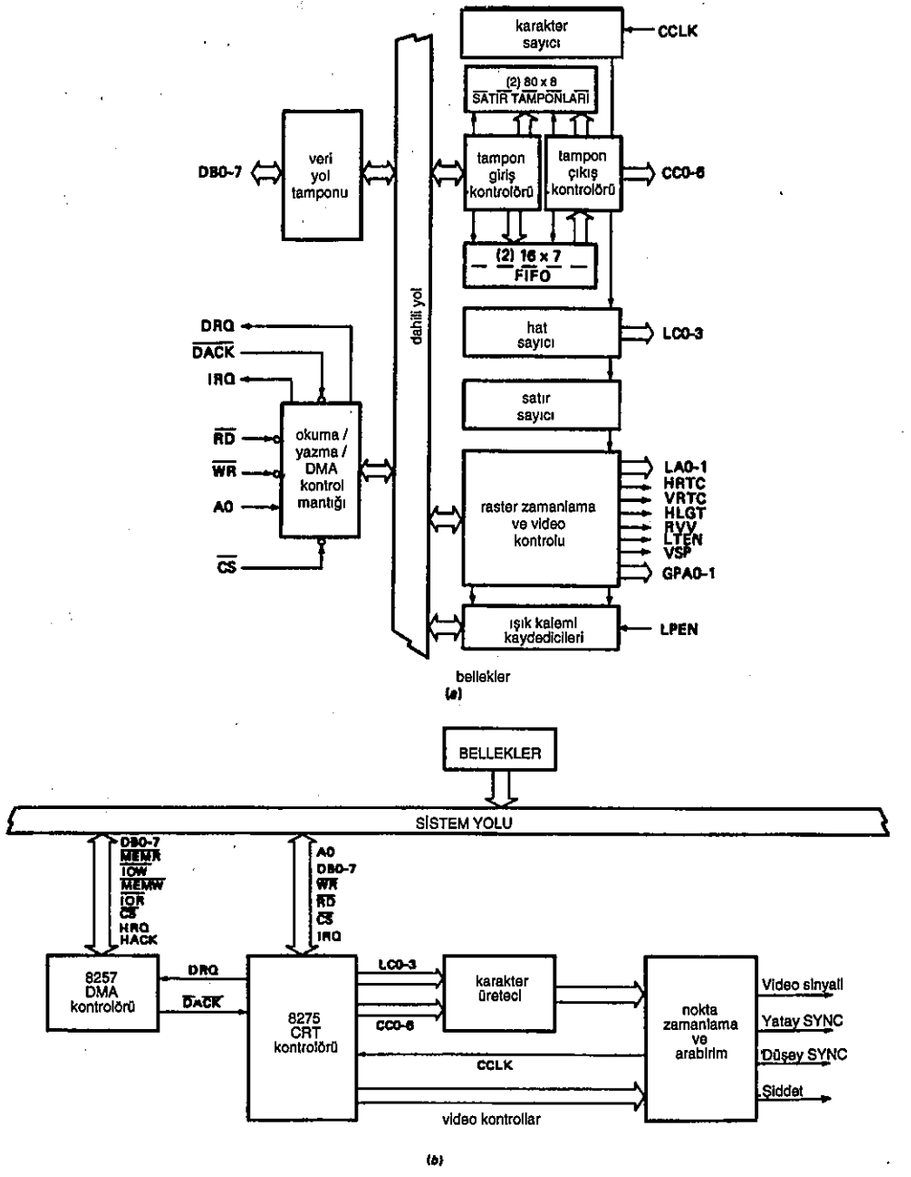
PROGRAMLANABİLİR CRT KONTROLÖRLERİ

Birçok üretici, değişik miktarlarda gerekli devreyi içeren CRT kontrolörleri üretirler. Burada bunlardan iki tanesi Intel 8275 ve Motorola MC6845 anlatılacaktır.

INTEL 8275 : Şekil 11.41*a*, Intel 8275 programlanabilir CRT kontrolörünün bir blok diyagramını , ve Şekil 11.41b bir mikroişlemci sistemine bağlanabilmesi için bir yol göstermektedir. 8275, gösterilecek karakter sayfalarını tutmak için harici RAM, bir karakter üretici, bir kayma kaydedicisi ve bir nokta saati gerektirir. Aynı zamanda, video monitor için uygun gerilim seviyelerinde kompozit video sinyalini üretecek analog arabirim devreside gereklidir. Fakat, 8275 karakter sayıcı, hat sayıcı, ve satır sayıcı içerir. Aynı zamanda, düşey ve yatay sync ve karartma için zamanlama sinyallerine de sahiptir. 8275 içindeki 80-baytlık iki satır tamponu, bir karakter satırı gösterilirken diğerinin bellekten alınmasına imkân sağlar.

8275 her satırda 1 ile 80 arası karakter, ve her çerçevede 1 ile 64 arası satır göstermek için programlanabilir. Her karakter için nokta hatları 1 ile 16 arasında programlanabilir. Her karakterdeki yatay nokta sayısı, karakter üreticiden gelen çıkış kelimesinin genişliği ve kayma kaydedicisinin uzunluğu ile belirlenir. Programlanabilir bütün özellikleri kullanıma hazırlamak oldukça karmaşıktır.

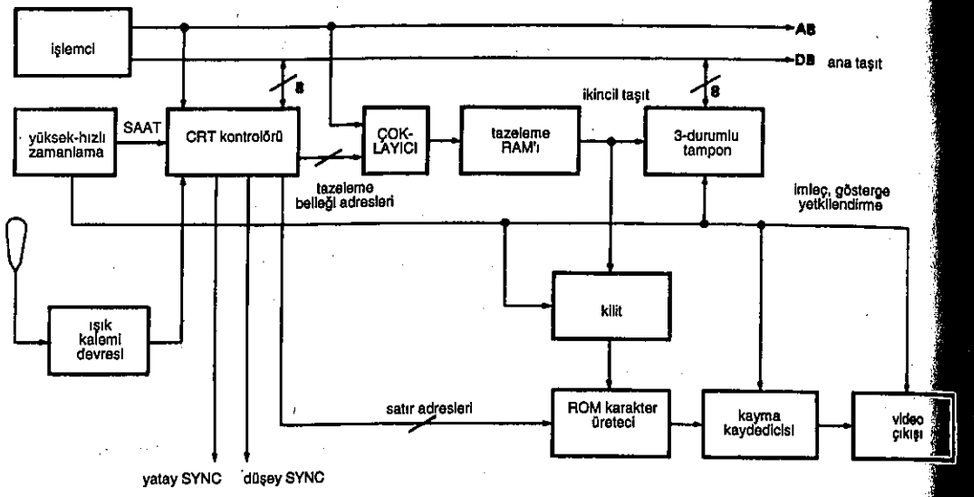
Şekil 11.41b'de gösterildiği gibi 8275, karakterleri mikroişlemcinin ana belleğinden doğrudan bellek erişimi (DMA) tabanlı çekecek şekilde ayarlanmıştır. 8275 bir yeni karakter satırı istediğinde, 8257 DMA kontrolörüne bir DMA istek sinyali gönderir. DMA kontrolörü örneğin bir 8080A'nın tutma girişine bir tutma isteği gönderir. 8080A taşıtlarını yüzer yapıp, tutmayı aldığını belirttiğinde 8257 belleği adresler ve bir sonraki karakter satırını 8275 satır tamponuna okur. DMA yaklaşımı mikrobilgisayarın zamanının çok küçük bir kısmını kullanır. Eğer klavyeden gelen bir karakter gösterilecekse, DMA kontrolörü tarafından erişilen bellek bölgesine kolaylıkla yazılabilir.



Şekil 11.41 Intel 8275 programlanabilir CRT Kontrolörü *a)* Dahili Blok Diyagramı *b)* Mikroişlemciye Sistem Bağlantısı

MOTOROLA MC6845 CRT KONTROLÖRÜ: 8275'te olduğu gibi MC6845 karakter sayıcı, satır sayıcı, hat sayıcı, yatay ve düşey sync ve karartmayı üretmek için gerekli devreleri içerir. Bir mikroişlemci kullanan tipik uygulama için gerekli harici parçalar Şekil 11.42'de gösterilmiştir. 6845'in DMA kontrolörü kullanmadığına dikkat ediniz. Gösterilecek karakteri okumak için 6845 14-bitlik tazeleme belleği adresi çıkışı verir. Çoklayıcı işlemci hatlarını tazeleme RAM'ına bağlarsa, tazeleme RAM'ına karakter yazılabilir.

İşlemci ile kontrolör arasında tazeleme RAM'ına erişme tartışmalarını önleyen bir metod kullanılırsa, tazeleme RAM'ı mikroişlemci anabelleğinin bir parçası olabilir.



Şekil 11.42 Bir mikroişlemci ile kullanılan Motorola MC6845 CRT Kontrolörü

CRT GRAFİKLERİ

Alfanümerik karakterleri göstermeye ilave olarak, CRT'ler sıksık bilgisayar tarafından üretilen grafikleri göstermekte kullanırlar. Buna örnek olarak video oyunları verilebilir.

Bir CRT üzerinde grafik üretmenin yaygın bir yolu tüm ekranı bir noktalar matrisi gibi düşünmektir. Bu noktalar bazen piksel olarak adlandırılır. Resimler, her pikselin açık ya da koyu olduğunu belirterek çizilir. Bir üretici ROM kullanmak yerine, bir hatta birbirini takip eden sekiz noktalık grupların herbirinin nokta dizisi RAM içine depolanır. Bu bayt bir kayma kaydedici içine depolanır ve her seferinde bir nokta olmak üzere video kuvvetlendiriciye yüklenir. Her sekiz nokta için enaz 1 baytlık bellek gerektiğinden, iyi çözünürlüklü grafik üretmek için büyük bellek gerekir. Renkli grafik üretmek, her nokta için renk bilgisinin depolanması gerektiğinden daha büyük bellek gerektirir.

Motorola MC6847 video gösterge üreticisi, bir renkli veya siyah-beyaz televizyon alıcısında grafik üretmek için gerekli devrelerin büyük bir kısmını sağlar. Alfanümerik modda herbiri 32 karakter olan 16 satır gösterir. Grafik modda ise 128x192 noktalı matrislere kadar dört-renkli nokta matrisleri üretir. MC6847'nin bir kullanımı mikroişlemci tabanlı video oyunlarıdır.

MİKROİŞLEMCİ ÇEVRE BİRİMLERİNİN BAĞLANMALARINDAKİ EĞİLİMLER

Çevre birimleri mikroişlemcilere bağlamaktaki eğilim daha karmaşık çevrebirim kontrolör entegre devrelerine doğrudur. Intel ve Motorola veri kitaplarına bir bakış, burada tartışabileceğimizden çok çevrebirim entegre devrelerinin olduğunu gösterir. Intel 8041 gibi bazı çevrebirim entegre devreleri, fonksiyonlarını kontrol etmek için önceden programlanmış özel mikroişlemciler içerirler.

Bir "akıllı" çevrebirim entegre devresinin avantajı ana mikroişlemciyi birçok zahmetten kurtarmak ve programlanmasını basitleştirmektir. Ana mikroişlemci programı yalnızca çevrebirim cihazlarını kullanıma hazırlamalı ve işaret edildiği zaman ya da periyodik olarak onlara cevap vermelidir.

TEKRAR SORULARI VE PROBLEMLER

1. Taramalı *I/O* ve kesme *I/O*'yu tarif ediniz. Her tipin avantajları ve dezavantajları nelerdir?

2. *a. 8085A RST 6.5* kesme girişinin maskesini kaldırmak ve yetkilendirmek için gerekli deyimleri gösteriniz.

   b. RST 6.5'un maskesini kaldırıldığını ve yetkilendirildiğini varsayarak, bir 8085A'nın şu andaki deyimini bitirdikten sonra bir *RST 6.5* girişine vereceği üç temel cevabı açıklayınız.

c. Kesme isteği darbesinin çok geniş olmaması niçin gereklidir?

3. *a.* Kesme kullanan programlarda yığın göstericiyi kullanıma hazırlamak niçin gereklidir?

   b. Bir kesme servis yordamı, kaydedicileri yığının üstüne niçin iter?

4. *a.* Bir *6800*'ün *NMI* girişindeki bir kesme sinyaline cevabını açıklayınız.

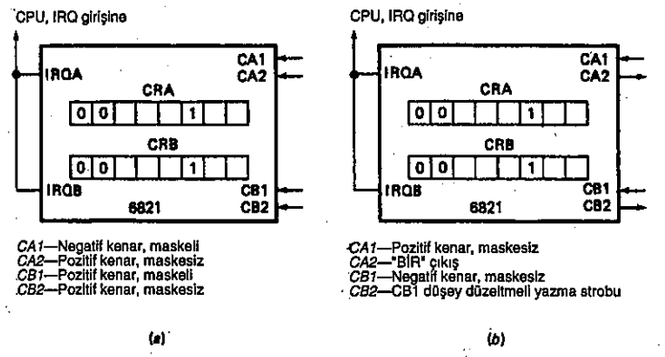
   b. Bir *6800* sisteminde resetten sonra kesmeleri yetkilendirmek için hangi deyim gereklidir?

5. *a.* Onaylı veri transferini tanımlayınız.

   b. Şekil 11.5*b*'yi kullanarak bir *8155* ile onaylı ya da stroblu giriş veri transferi sırasındaki sinyal sırasını açıklayınız.

   c. Bir 8155'i 28H ile 20H adreslerinde aşağıdaki şekilde ilk kullanıma hazırlamak için gerekli deyimleri gösteriniz: *Port A* stroblu çıkış, *port B* stroblu giriş, kesme *A* yetkilendirilmiş, kesme B yetkilendirilmiş, zamanlayıcı *NOP*.

6. Kesme kontrol hatlarını Şekil 11.43'te belirtildiği gibi programlamak için gerekli bitleri doldurunuz.



Şekil 11.43

7. Bir *6821*'i 8020H-8023H adreslerinde aşağıdaki şekilde ilk kullanıma hazırlayacak çevirici dil deyimlerini gösteriniz: *Port A* stroblu giriş, *CA1* negatif kenarda aktif ve maskesiz, *CA2 E* düzeltmeli okuma strobu; *Port B*

stroblu çıkış, *CB1* pozitif kenarda aktif ve maskesiz, *CB2*, *CB1* düzeltmeli yazma strobu.

8. Bir 6800, Şekil 11.43*a*'da gösterildiği gibi *6821*'den kesme isteği almıştır. *CPU* dört girişten hangisinin kesme istediğini nasıl anlar?

9. Bir matris klavyeden faydalı veri almak için gerekli üç temel işi tanımlayınız.

10. *a.* Bir 16-tuşlu klavyeyi taramak ve kodlamak için yazılacak programın akış diyagramını çiziniz.

    b. Akış diyagramını, program ana programa dönmeden önce tuşbasılı strobunun alçağa gitmesini bekleyecek şekilde değiştiriniz.

11. Şekil 11.13'teki programı dört satırın hepsi kontrol edildiği ve basılı tuş bulunmadığı zaman hata çıkışına atlayacak şekilde değiştirecek çevirici dil deyimlerini ekleyiniz. (İpucu: Bir satır sayıcı kullanınız).

12. Yazılım-çoklanmış ve donanım-çoklanmış göstergelerin avantaj ve dezavantajlarını sıralayınız.

13. Sekiz-basamak-yedi parçalı göstergeyi çoklayacak bir akış diyagramını çiziniz.

14. Problem 13.teki sekiz-basamaklı göstericiyi sürecek bir *8085A* veya *6800* çevirici dil yordamını yazınız. *Port 01*'i parçaları sürmek, ve *port 02*'yi basamakları sürmek için kullanınız. Ortak anod göstergeler, evirici basamak sürücüleri varsayınız. Yedi-parçalı kodlarını bellekte bir tabloda ve gösterilecek karakterleri bellekte bir başka tabloda depolayınız. Bir sonraki basamağa gitmeden önce her basamağı 1ms aydınlatınız.

15. Sıvı kristal göstergeler için gerekli sürücüleri anlatınız. *LCD*'ler niçin genellikle çoklanamazlar?

16. Adım motorunu tanımlayınız. Bir adım motorunu girilen adım sayısı kadar saat yönünde veya saat yönünün tersi yönde döndürecek bir kısa çevirici dil yordamı yazınız.

17. Bir 12-bitlik *D/A* çeviricinin 8-bitlik mikroişlemci portlarına, *D/A* çeviricinin 12 bitin hepsini bir defada alması için nasıl bağlanacağını gösteren bir devre çiziniz.

18. Dört BCD-basamaklı *D/A* çeviriciden veri girişi için bir çevirici dil yordamı yazınız. (İpucu: Bir basamak strobunun doğruluğunu kontrol ediniz ve bu basamak için veriyi okuyunuz.)

19. Bir veri toplama sisteminin çalışmasını açıklayınız.

20. *a.* S-100 yolunu, çoklu yolu ve IEEE-488'i (GPIB) anlatınız.

b. Her birinin ana kullanımı nedir?

c. Birçok *CPU* kartlı bir çok yollu sistemde yolu hangi *CPU* kartının kontrol ettiği nasıl anlaşılır?

d. GPIB veri taşıtları üzerinde bir veri baytının transferi sırasında NRFD, DAV ve NDAC üzerindeki sinyallerin sırasını anlatınız.

21. Aşağıdakileri tanımlayınız.

a) RS-232C *b)* Boşluk *c)* İşaret *d)* 20mA akım döngüsü *e)* RS-422 *f)* RS-423 *g)* MODEM *h)* Frekans kaymalı anahtarlama *i)* Tam dubleks *j)* Yarım dubleks *k)* Baud *l)* Durdurma biti *m)* Başlama biti *n)* DTE *o)* DCE

22. *a.* Aşağıdaki *RS-232C* sinyallerinin fonksiyonlarını tarif ediniz.

TxD, RxD, $\overline{\text{RTS}}$, $\overline{\text{CTS}}$, $\overline{\text{CD}}$, $\overline{\text{DTR}}$, $\overline{\text{DSR}}$

b. RS-232C mantıki yüksek ve alçak için gerilim seviyelerini söyleyiniz.
c. RS-232C sinyal kabloları için hangi tip konnektör kullanılır?
d. RS-232C için iletim uzunluğu ne kadardır?

23. *a.* Bir *MC6850* ACIA'yı aşağıdaki şekilde ilk kullanıma hazırlamak için gönderilecek kontrol kelimesini gösteriniz: 16'lık baud hızı faktörü, 7 bit, tek eşlik, 1 durdurma biti, *RST* onaylı (alçak); verici kesmesi yetkilendirilmiş, alıcı kesmesi yetkilendirilmiş

b. 6850'yi 8008H ve 8009H adreslerinde *(a)* şıkkındaki gibi ilk kullanıma hazırlayan ve bir karakterin gönderme tamponuna ne zaman gönderilebileceğini belirlemek için statü kaydedicisini tarayan çevirici dil deyimlerinin sırasını gösteriniz.

c. Yukarıdaki gibi ilk kullanıma hazırlanan *MC6850*, 1200 Bd'de veri gönderecektir, *6850*'nin verici saati girişine hangi frekansta verici saati uygulanmalıdır?

d. Hangi iki yolla *CPU, 6850* alıcı tamponunun okunmaya hazır karaktere sahip olduğunu belirleyebilir?

24. *a.* 8251A cihazını aşağıdaki gibi ilk kullanıma hazırlamak için gönderilecek mod kelimesi ve komut kelimesini gösteriniz. Baud hızı faktörü 64; her karakterde 7 bit, çift eşlik, 1 durdurma biti, verici kesmesi yetkilendirilmiş, DTR ve DSR onaylanmış (alçak), alıcı kesmesi yetkilendirilmiş, hata bayrakları resetlenmiş, kırılma karakteri yok, yakalama modu yok.

b. 8251A'yı 80H ve 81H adreslerinde *(a)* şıkkında olduğu gibi ilk kullanıma hazırlayan, ve alıcı tamponunun okunmaya hazır karakteri olup olmadığını belirlemek için statü kaydedicisini tarayan çevirici dil deyimlerini sırasıyla yazınız.

c. Okunan bir karakterin bir eşlik hatası içerdiğini nasıl anlarsınız?

d. Yukarıdaki gibi ilk kullanıma hazırlanan bir *8251A*, 2400 Bd'de veri göndermek için hangi frekanslarda verici ve alıcı saatleri gerektirir?

e. Hangi iki yolla *CPU, 8251* alıcı tamponunun okumaya hazır bir karaktere sahip olduğunu anlayabilir?

25. Bir port bitinden, bir yazılım yordamı kullanarak nasıl asenkron seri veri gönderileceğini gösteren bir akış diyagramını çiziniz.

26. Bir *CRT* üzerinde karakter veya grafik görüntüleri üretmek için kullanılan geçmeli ve geçmesiz taramayı tanımlayınız. Yatay ve düşey tarama niçin senkronize olmak zorundadır?

27. *6571* gibi bir karakter üreticinin bir *CRT* üzerinde, bir karakter için nasıl noktalar dizisi ürettiğini açıklayınız. Açıklamaya yardım için RAM, kayma kaydedici, ve zamanlama zincirini gösteren bir blok diyagram çiziniz.

28. Ana mikroişlemci belleğinden bir sonraki karakterler satırını almak için *8275 CRT* kontrolörünün *DMA*'yı nasıl kullandığını açıklayınız. Gösterilecek karakteri *8275*'e vermek için neden genellikle bir program yordamı kullanılmadığını düşünüyorsunuz? (İpucu: zaman).

REFERANSLAR/BİBLİYOGRAFYA

SC-01 Speech Synthesizer Data Sheet, Votrax, Division of Federal Screw Works, Troy, MI 1980.

Intel SDK-85 User's Manual, Intel Corporation, Santa Clara, CA, December 1977.

MEK6800D2 Evaluation Kit II Manual, Motorola Semiconductor Products. Inc..Austin, TX. 1976.

Intel Component Data Catalog 1982. Intel Corporation, Santa Clara, CA, 1982.

Lesea, Austin, and Rodnay Zaks: *Microprocessor Interfacing Techniques*, 2d ed., Sybex, Inc., Berkeley, CA, 1978.

The Complete Motorola Microcomputer Data Library, prepared by Technical Information Center, Motorola, Inc., Phoenix, AZ, 1978.

EIA Standard RS-232-C Interface between Data Terminal Equipment and Data Communication Equipment Employing Serial Binary Data Interchange, Electronic Industries Association Engineering Department, Washington, 1969.

Condensed Description of the Hewlett-Packard Interface Bus, Application Note HP-IB, Hewlett-Packard, Palo Alto, CA, 1977.

Slo-Syn Synchronous/Stepping Motors and Motor Controls, Catalog MD174-1, The Superior Electric Company, Bristol, CT, 1974.

8800 System Bus Structure, Altair 8800 Manual, MITS, Inc., Albuquerque, NM.

Speech Plus the Talking Calculator, Data Sheets, Telesensory Systems, Inc., Palo Alto, CA, 1976.

Speech Synthesizer Module TS1, Data Sheet. Telesensory System, Inc., Palo Alto, Ca, 1977.

SpeechLab, Data Sheet, Heuristics, Inc., Los Altos, Ca, 1977.

"Revised data-interface standards permit faster data rates and longer cables. New chips, and RS-232 adapters, simplify their use, " Technology, *Electronic Design,* vol. 18, September 1, 1977, pp. 138-141.

BÖLÜM 12

PARÇALARI BİR ARAYA KOYMA

Bu bölümde mikroişlemcilerin ve daha önceden tartışılan diğer elemanların mikroişlemci-tabanlı cihazların yapımında nasıl kullanıldıkları üç örnek ile gösterilecektir. Birincisi, süpermarket çıkış kasalarında kullanılan terazilere benzer 8085-A tabanlı "akıllı" bir terazidir. İkincisi 6800 tabanlı bir EPROM programlayıcıdır. Üçüncü örnekte mikro işlemciler ile endüstriyel kontrol prensiplerini göstermek için 8085-A tabanlı sıcaklık kontrol edici kullanılır. Bu örnekler anlaşılabilecek düzeyde basit fakat gerçekçi olabilecek derecede karmaşık olduklarından seçilmiş ve geliştirilmişlerdir. Örnekler, tartışılan programların sizinde kendi programlarınızı yazarken kullanabileceğiniz birçok yordamı içerecek şekilde düzenlenmiştir.

Bölümdeki her alet için olan kısım, yeni aletleri öğrenmek için gerekli mantıklı adımlar içerisinde düzenlenmiştir. Bu aletler, mikroişlemci temel örnek kartları üzerinde kullanılmıştır ve bunlar Bölüm 13 'te bir cihazın geliştirilmesi sırasında nasıl düzeltildiği hakkındaki örneklerde kullanılacaklardır.

Bölümün sonunda siz

1. Mikroişlemci tabanlı bir aletin donanım ve yazılımını öğrenmek için mantıksal adımları söyleyebilecek,
2. Monitor programını, eşitlemeleri, tampon ve modüler programları tanımlayabilecek,
3. Başvuru tablolarının bir kodu diğerine çevirirken nasıl kullanabileceğini açıklayabilecek,
4. BCD numaraları ikili eşdeğerlerine çevirmekte kullanılan bir işlemi tanımlayabilecek,
5. İkili sayıları bir mikroişlemci ile çarpmanın üç metodunu anlatabilecek,
6. Mikroişlemci tabanlı bir ***terazinin*** çalışmasını anlatabilecek,
7. Mikro işlemci tabanlı bir *EPROM* programlayıcının çalışmasını anlatabilecek,
8. Bir mikroişlemcinin aynı zamanda bir çok endüstriyel süreci nasıl gözleyebileceğini ve kontrol edebileceğini anlatabilecek duruma geleceksiniz.

MİKROİŞLEMCİ TABANLI AKILLI BİR TERAZİ

ALETİN TANIMLANMASI

Yeni bir aleti öğrenmenin ilk adımı, eğer mevcutsa üretim tanımlarını okumaktadır.

Bu terazi bir süpemarket ya da nalburiye dükkânı çıkışlarındaki kasalar vb. uygulamalarda kullanılmak amacıyla tasarlanmıştır. Tartılacak parça bir platform üzerine yerleştirilir. Ağırlık terazi tarafından belirlenir ve dört yedi-parçalı LED gösterge ve iki ilave birim göstergesi üzerinde gösterilir. Maksimum ağırlık 10.00 lb. ve hassasiyet 0.01lb'dir.

Operatör, fiyatı hesap makinesi tipindeki klavyeden girinceye kadar ağırlık göstergesi sürekli olarak yenilenir. Daha sonra dört LED gösterge üzerinde paund başına fiyat gösterilir. Tam-ölçekli ağırlıkta maksimum fiyat, paund başına 9.99 dolardır. Paund başına fiyat girilirken bir hata yapılırsa, bunu düzeltmenin iki yolu vardır. Birincisi sıfır tuşuna 4 defa basmaktır. Bu bütün rakamları paund başına fiyat göstergesinden siler ve daha sonra doğru fiyat girilebilir. Diğer bir yol da INTR tuşuna basmaktır. Bu, tartıyı yeni fiyat girilene kadar ağırlık ölçme moduna çevirir. Fiyat *(F)* tuşuna basılması fiyatın bütün rakamlarının girildiğini belirtir. Bundan sonra terazi otomatik olarak ağırlık ile paund başına fiyatı çarpar ve toplam fiyatı gösterir. 3 saniye sonra terazi, ağırlığı belirleme ve gösterme durumuna döner. Ağırlığı ölçme moduna dönüş herhangi bir zamanda INTR tuşuna basarakta elde edilebilir.

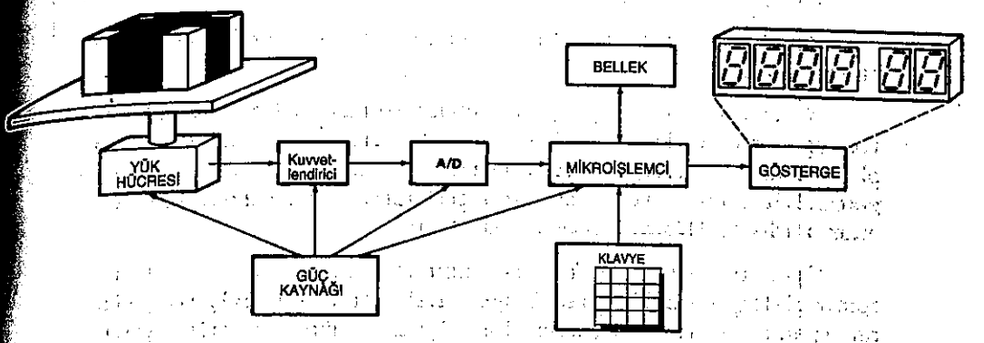
ÇALIŞMA TEORİSİ: DONANIM

Yeni bir aleti öğrenmenin bir sonraki adımı çalışma teorisinin donanım kısmını incelemektedir. Bu, şemaları, bellek haritalarını ve blok diyagramlarını içerir.

Şekil 12.1'de gösterildiği gibi terazi, ağırlığı orantılı bir elektrik sinyaline dönüştüren bir yük-hücresi ile başlar. Bir kuvvetlendirici yük-hücresi sinyalini A/D çeviricinin uygun aralığı için kuvvetlendirir. A/D çevirici kuvvetlendirici çıkışını mikroişlemci üzerinde çalıştırılabilecek bir sayısal kelimeye çevirir. Bellek mikroişlemci için yazılan programı ve ağırlığın A/D çeviriciden gelen sayısal şeklini saklar.

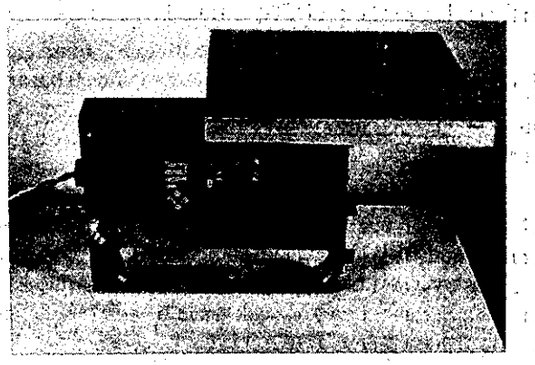
Gösterge önce ağırlığı, daha sonra paund başına fiyatı ve en sonunda toplam fiyatı gösterir. Klavye, paund başına fiyatı ve toplam fiyatı hesaplamak için gerekli komutu girmek için kullanılır.

Yük Hücresi ve Kuvvetlendiriciler: Bir yük-hücresi dönüştürücüsü, ağırlığı orantılı bir elektrik sinyaline dönüştürür. Şekil 12.2 Transducers, Inc. Şirketinin Model C46210#-10P1 yük-hücresinin bir resmini göstermektedir. Bu birimin 0 ile 10 lb (0-4.5 kg) aralığında doğru çalıştığı belirtilmiştir, fakat



Şekil 12.1 Mikroişlemci tabanlı akıllı terazinin blok diyagramı.

100lb (45.4kg)'a kadar olan ağırlıklar birimi mekanik olarak bozmadan uygulanabilir. Birimin doğruluğu yüzde ± 0.1 'den, ya da 1000'de ±1 parçadan daha iyidir. Bu 10 lb (4.5kg) bir birim için ± 0.01 lb (±0.004 kg)'a karşı gelir.



Şekil 12.2 Terazi için platformu olan yük-hücresinin fotoğrafı.

Yük hücresinin içindeki asıl sezici elemanlar, şekil 12.3'ün sol tarafında gösterildiği gibi 350 Ω direnç köprüsüne eşdeğerdir. Köprüyü sürmek için 10V uygulanır. Hücrenin üzerinde yük olmadığı zaman köprünün iki çıkışı da yaklaşık 5V olarak aynı gerilimdedirler. Uygulanan bir yük dirençlerden birinin direncini arttıracaktır. Bu da bir fark çıkışı gerilimi üretir. 10 lb [4.5 kkg]'lık bir yük için maksimum fark çıkışı gerilimi, kutuplama geriliminin her voltu

için 2 mVolttur. Gösterildiği gibi 10V kutuplama gerilimi tam ölçekte 20 mVoltluk fark çıkış gerilimi verir.

Bu küçük sinyali A/D çevirici için hatasız olarak uygun seviyelere yükseltmek için şekil 12.3'te gösterilen bir aletlendirme kuvvetlendiricisi kullanılır. *IC-1* ve *IC-2* yüksek giriş empedanslı fark-giriş, fark-çıkış kuvvetlendiricisi oluştururlar. *IC-1* ve *IC-2*'ye ortak -mod sinyali olan 5V yükseltilmez. 20 mVoltluk fark sinyali *(R3+R4+2R1)+(R3+R4)* tarafından yükseltilir. R4, 5 civarında bir kazanç vermek için ayarlanır, böylece hücre üzerindeki 10.00 lb (4.54kg) için fark çıkışı 100.0mV olur. *IC-3* standart birim kazançlı bir fark-giriş kuvvetlendiricidir. Bu *IC-1* ve *IC-2*'den gelen fark sinyalini *A/D* çevirici için gerekli tek uçlu sinyale dönüştürür.

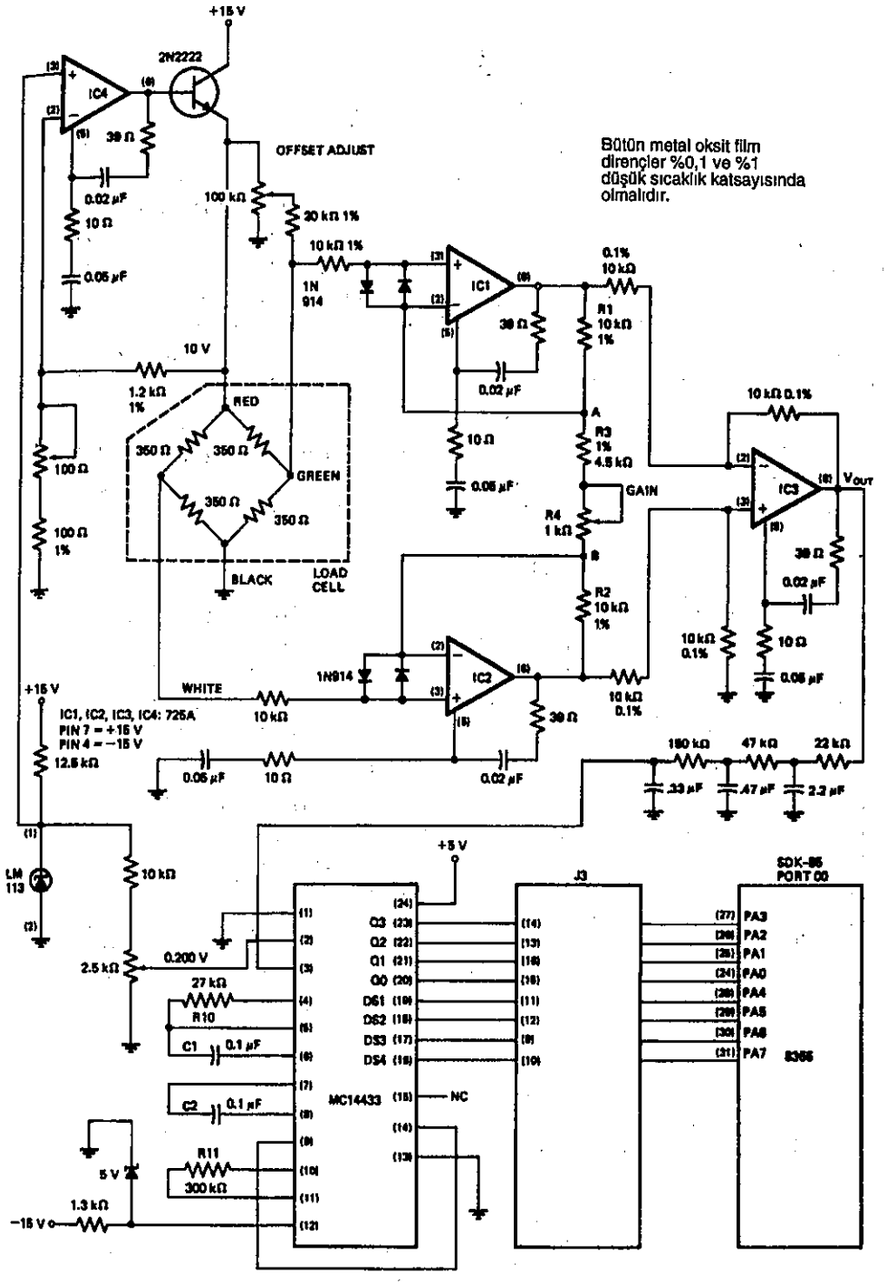
A/D çevirici: Bu terazide kullanılan *A/D* çevirici Motorola MC 14433'tür. Bu 3 1/2 rakamlı sayısal (digital) voltmetreler, vs. kullanılmak üzere amaçlanmış çift-eğimli, pahalı olmayan bir çeviricidir. Şekil 12.3'de gösterildiği gibi, tam bir *A/D* çevirici için dıştan yalnızca bir referans gerilimi ve birkaç direnç ve kondansatör gereklidir.

Çevirici kolaylıkla 2.000 V tam ölçek veya 200.0 mV tam ölçek için kullanılabilir. 2.000 V referans gerilimi ve 470 kΩ luk R10 direnci 2.000V tam ölçek verir. Şekil 12.3'te gösterildiği gibi 200.0mV referans gerilimi ve 27 kΩ luk R10 direnci 200.0 mV tam ölçek verir. Şekil 12.3'teki 1.2 Voltluk hassas, sıcaklık- kararlı LM 113 diyotu, çevirici referans gerilimini üretmek için kullanılır.

Çeviricinin çıkışı çoklanmış BCD şeklindedir. *Q0-Q3* veri çıkışlarıdır, ve *DS1-DS4* veri gözleme çıkışlarıdır. Bir veri gözleme çıkışı, 0 basamağın verisi *Q0-Q3* çıkışları üzerinde olduğu zaman yüksek duruma geçer. Çevirme hızı ve çoklama frekansı bir iç osilatör ile kontrol edilir. Osilatörün frekansı *R11* tarafından belirlenir. 300kΩ'luk bir *R11*, 66KHz'lik bir iç saat frekansı, 0.8 KHz'lik çoklama frekansı, ve her saniyede yaklaşık dört tam çevirme verir. Çevirinin doğruluğu okumada yüzde ± .05 ve saymada ± 1'dir, buda yük-hücresinin doğruluğu ile karşılaştırılabilir bir büyüklüktür. 3 1/2 rakamlı çeviricinin doğruluğunu düşünmenin bir başka yolu da 1999 tam saymada ± 2 parçadır.

SDK-85 Mikroişlemci Geliştirme Kartı

Tartı için gerekli mikroişlemci, RAM, ROM, Klavye, ve gösterge bir Intel *SDK- 85* mikroişlemci prototip kartı üzerinde bulunmaktadır. Şekil 9.9 *a* ve *b* kartın, RAM, ROM ve mikroişlemci parçalarının şemasını göstermektedir; Şekil 9.8*a* ve *b* kartın bir fotoğrafını ve blok diyagramını göstermektedir; şekil 11.18 SDK-85'in klavye ve gösterge parçalarının şemasını göstermektedir.



Şekil 12.3 Load-cell (transdüser) yükselteçli ve A/d çevirici kısmın terazi devresi

Bu şekillerden de görebileceğiniz gibi kart, 3MHz saatli bir 8085-A mikro işlemci kullanmaktadır. Kit ile birlikte gelen bir 8355, iki *I/O* portu ve bir monitor programı ile önceden programlanmış 2 Kilo baytlık ROM içermektedir. Monitor programı, geliştirme sırasında programların RAM'a girilmesine ve işletilmesine izin verir. Tartının enson temel örnek şeklinde 8355 monitoru, tüm tartı programını ve aynı iki *I/O* portunu içeren 8755 EPROM ile değiştirilmiştir.

Bir 8155 entegre 256 baytlık RAM, üç *I/O* port, ve bir 8-bitlik programlanabilir zamanlayıcı sağlar. Terazi, yığın olarak kullanmak ve veri depolamak için küçük miktarda bir RAM gerektirir. Eğer *I/O* portları ve zamanlayıcı gerekmiyorsa, terazinin en son şeklinde yalnızca RAM içeren daha ucuz bir entegre devre kullanılabilir.

8279 klavye-gösterge kontrol edici yongası işlemciyi 24 tuşlu bir klavye ve altı *adet yedi-parçalı göstergeye* bağlar. 8279 'un çalışması için Bölüm 11'e bakınız.

SDK-85 Kartı İçin Bellek Haritası

Yeni bir sistemi çalışırken bakılacak önemli bir kısımda bellek haritasıdır. Eğer harita verilmemişse, şemadan bir bellek haritası yapmak faydalı olacaktır. Bellek haritası sizin bir bakışta belleğin hangi bölgelerinin RAM, ROM, ve I/O portları için ayrıldığını görmenizi sağlar. Şekil 12.4, SDK-85 için bellek haritasını göstermektedir. 8205, (Şekil 11.18'de *A10*), adres hatlarını *CS0-CS5* altı yonga-seçme çıkışlarını üretmek için kodlar. İki tanesi, ROM içinde 0000-07FFH ve 0800-0FFFH bölgelerindeki 2 kilobaytlık blokları yetkilendirmekte kullanılır. *CS2* çıkışı açık olarak bırakılır. Bu, belleğin başka bir bloğunu ya da eklenmiş çevre entegre devresini yetkilendirmekte kullanılabilir. *CS3*, 1800H adresindeki 8279 klavye-gösterge kontrol ediciyi yetkilendirir. A8 adres hattı, komut adresi olarak 1900H'yi ve cihazın veri adresi için 1800H'yi seçen 8279 C/$\overline{D}$ girişine bağlanmıştır.

CS4, 2000H'den 27FFH'ye olan 2 kilobaytlık bellek bloğunu kodlar. Bu çıkış, 8155 olan A 16 çıkışını seçmekte kullanılır. 8155'te yalnızca 256 bayt RAM olduğundan, 2K'lık bölge doldurulmaz. Kullanılmayan bölge geri katlanmış olarak adlandırılır. Gerçek anlamda 2000H ile 27FFH içindeki herhangi bir adres *"geri katlanacak"* ve temel 256 bayt RAM içindeki bir yeri seçecektir. CS5, RAM için 2800H-2FFFH arasındaki bir başka 2Kbaytlık bloğu seçer. 3000H, veya 12228 ondalık, üzerindeki bütün bellek bölgesi, bellek genişletilmesi veya bellek-haritalanmış I/O portları için geçerlidir.

Asıl terazi programı 0000 ile 07FFH'deki bir 8755 EPROM, 1800H'deki bir 8279 klavye-gösterge kontrol edici, ve 2000H'deki bir 8155 RAM kullanır.

8080A ve 8085A gibi I/O portlarının bellekten ayrı olduğu işlemcilerde diğer önemli harita I/O port adresleridir. SDK-85 kart için I/O port adresleri Şekil 9.9 *b*'de gösterilmiştir. Terazi, *A14* 8755 'in *A portu* giriş portu olarak kullanır. Şekil 9.9*b*'de gösterildiği gibi bu portun adresi 00H'dir. 8755'de mevcut bir diğer port 01H adresindedir.

ÇALIŞMA TEORİSİ: YAZILIM

Yeni bir aletin donanım çalışmasını en azından biraz anladıktan sonra incelenecek bir sonraki kısım yazılım ya da programlamadır. Programın kendisi ile uğraşmadan önce, programın ne yaptığı konusunda size bilgi verecek bir akış diyagramına bakınız. Bazı alet kullanım kılavuzları bütün çalışma için bir akış diyagramına ve ilave olarak ana akış diyagramının her bloğunun çalışmasına ait akış diyagramlarına sahiptir.

SDK-85 Tabanlı Terazi İçin Akış Diyagramı

Şekil 12.5, SDK-85 tabanlı terazi için tam bir akış diyagramı göstermektedir. Başlangıçtan sonra, program ağırlık rakamlarını birer birer A/D çeviriciden almaktadır. Ağırlık dörtlü grup halindeki yedi-parçalı *LED*'lerde gösterilmektedir ve ilave iki yedi -parçalı *LED* göstergede *Lb* gösterilmektedir.

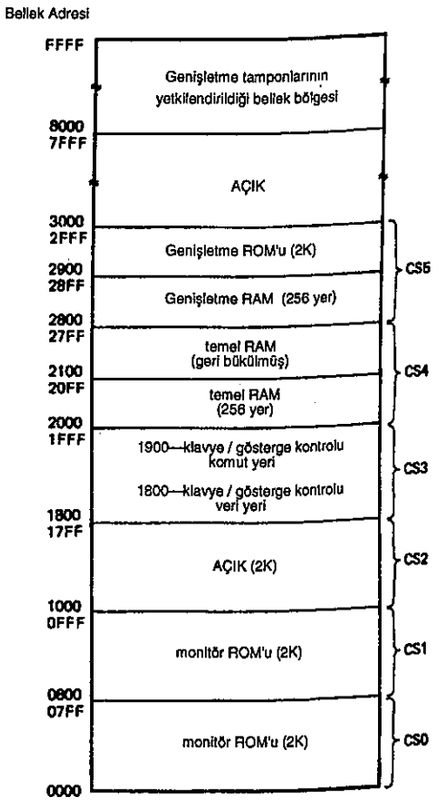
Klavye üzerindeki herhangi bir tuşun basılıp basılmadığı kontrol edilmektedir. Eğer bir tuşa basılmamışsa, program aletin geliştirilmesi sırasında *sessiz terazi* olarak bilinen kısma geri dönmektedir.

Sessiz terazi döngüsü bir tuş basılıncaya kadar tekrar edilir.

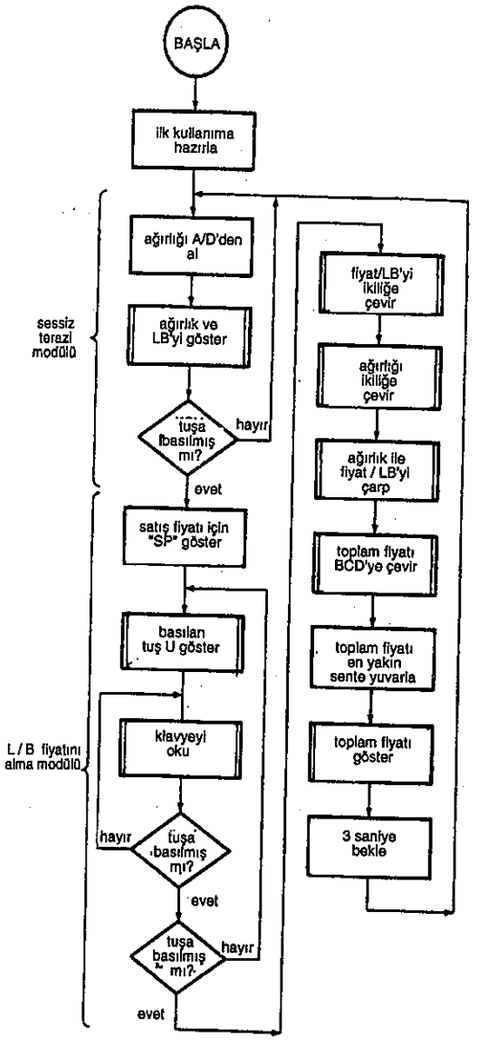
Bir tuşa basıldığında, ağırlık dört rakamlı gösterge üzerinden silinir ve basılan tuşun karakteri gösterilir. İki rakamlı gösterge *satış fiyatının (selling price)* kısaltması olan *SP*'yi okumak için değişir. Klavyeden girilen paund başına fiyata ilave rakamlar olarak dörtlü göstergede gösterilirler. Bir "F" tuşu basıldığında program, klavye okuma döngüsünden çıkar.

İkilik çarpma, BCD çarpmadan daha kolay olduğundan paund başına fiyat ve ağırlık, BCD'den ikiliğe dönüştürülür. Daha sonra ağırlık ile paund başına fiyat çarpılır ve toplam fiyat ikilik olarak bulunur. İkilik toplam fiyat BCD'ye çevrilir ve en yakın sente yuvarlanır. Toplam fiyat dört *LED* göstergede gösterilir ve fiyat kısaltması *"Pr"* iki *LED* göstergede gösterilir.

Toplam fiyatı okumak ve bir yazar kasaya girmek için yeterli bir süre olan 3 saniyeden sonra program sessiz terazi moduna döner. INTR tuşuna basmak programı herhangi bir noktadan sessiz terazi durumuna döndürür. Bu eğer paund başına fiyat girilirken bir hata yapıldıysa çok kullanışlıdır.



Şekil 12.4 SDK-85 geliştirme kartının bellek haritası .



Şekil 12.5 8085A tabanlı 10-1b terazinin akış diyagramı.

ÇEVİRİCİ DİLİNDE YAZILMIŞ PROGRAMIN ANALİZİ

Yeni bir aleti incelemenin bir sonraki adımı, çevirici dilinde yazılmış programı inceleyerek akış diyagramının bloklarının nasıl gerçekleştirildiğini görmektir. Terazi için yazılan programın tamamı Şekil 12.6'da gösterilmiştir. Program anlaşılabilir olması için yeterince basit olarak yazılmıştır, fakat gerçekçi olacak kadar da karmaşıktır. Programı tartışmaya başlamadan önce birkaç genel uyarı yapalım.

Öncelikle bir aletle ilgili program dökümünün birçok sayfadan oluşması karşısında şaşırmamak önemlidir. Bellek maliyetlerinin çok yüksek olduğu mikroişlemcilerin ilk günlerinde programcılar, programları en az hafıza kullanacak şekilde yazıyorlardı. Bayt başına bellek maliyeti düşerken program yazmadaki önem, en az hafıza kullanımından çok daha anlaşılır programlar yazmaya yöneldi.

Bu, programları, birbiriyle bağlantılı bağımsız modüller ya da altyordamlar serisi olarak yazmakla gerçekleştirildi. Bölüm 13'te gösterildiği gibi, modüler programların avantajı her bir modülün diğerleri ile bağlanmadan önce tek başına test edilip düzeltilebilmesidir. Modüller veya altyordamlar aynı zamanda bir flopy disk üzerine kaydedilebilir ve gerektiğinde yeni programları yazmakta kullanılabilirler. Bu modüllerde sembolik adresler kullanılır, böylece modüller hafızanın herhangi bir bölgesinde çalışacak şekilde çevrilebilirler. Eğer bir program burada terazi programında olduğu gibi modüler şekilde yazılmışsa, her bir modül üzerinde ayrı ayrı çalışabilirsiniz. Akış diyagramı modülleri tanımanızda size yardımcı olabilir.

EŞİTLEMELER: Program listesi bir eşitlemeler tablosu ile başlamaktadır. Bu tablo çevirici programa, program içindeki her sembolik adres için hangi mutlak adresi kullanacağını anlatır. Örneğin, BUFFR1 EQU 20A0H deyimi çevirici programa BUFFR1 sembolik adresini her okuyuşunda 20A0H adresini girmesini söyler.

SESSİZ-TERAZİ MODÜLÜ: Başlangıçtan sonra işlemci 00 portunu okur. Şekil 12.3'te gösterildiği gibi port 00, *A/D* çeviriciye, BCD çıkış hatlarına ve rakam-ışığı hatlarına bağlanmıştır. *DS1* rakam-strob hattında en büyük değerlikli rakam için 1 buluncaya kadar port 00'ı defalarca okur. Işık görüldüğü anda bu rakam için gelen veri *A/D* çeviriciden okunur *A/D* çevirici 3 1/2 rakamlı olduğundan en büyük değerlikli bit yalnızca 1 veya 0 olur. Bu 1 ya da 0 en büyük değerlikli basamağın 4-bitlik veri kelimesinin 3. bitinde çıkışa verilir. Eğer 1 okunursa gösterilir, fakat 0 okunursa boşluk bırakılır.

Programın *RST 5.5.* ve *RST 7.5* adresleri üzerinden atladığına dikkat ediniz.

Ağırlığın en büyük değerlikli basamağı için tampona bir boşluk veya bir karakter yüklendikten sonra program bir sonraki rakam ışığını beklemek için bir döngüye girer. Rakam 2 ışığı görüldüğünde bu rakam için veri okunur ve tampondaki bir sonraki adrese depolanır. Bu rakam ve geri kalan rakamlar baytın alt 4 bitinde standart BCD formatındadırlar.

Aynı şekilde, rakam 3 ve 4 verileri de ışık görüldüğü zaman okunurlar. Okunan veriler tampon içindeki takip eden iki adrese depolanır. Dört rakamın hepsi bir tampon içine okunduğu zaman, BCD karakterleri yedi-parçalı koda çeviren ve karakterleri gösterilmek üzere 8279'a gönderen bir gösterme altyordamı çağrılır.

Daha sonra *"L"* ve *"b"* karakterleri bellekteki bir başka tampon içine yüklenir. Gösterme yordamı, bu karakterleri tampondan okumak, yedi-parçalı koda çevirmek, ve yedi-parçalı kodu 8279 göstergeye göndermek üzere yeniden çağrılır. Eğer gösterme yordamı akümülatörde 00H olduğu durumda çağrılırsa HL tarafından gösterilen tampondaki karakterler dört LED göstergede gösterilirler. Gösterme yordamı çağrıldığında akümülatörde 01H varsa, karakterler iki LED göstergede gösterilirler. Gösterme altyordamı daha sonra detaylı olarak tartışılacaktır. Bir programı ilk defa okurken, en iyi yol programın ana akışını takip etmek ve yalnızca çağrılan alt yordamların tanımlarına, giriş ve çıkışlarına hızlı bir şekilde bakmaktır. Programın bütün çalışması hakkında genel bir fikir sahibi olduktan sonra alt yordamların detayları üzerinde çalışılabilir. Sessiz-terazi program modülünün son parçası klavyede bir tuşun basılıp basılmadığını kontrol eder. 8279 devamlı olarak klavyeyi kontrol eder. Bir tuşa basıldığında 8279 bir zıplama zamanı kadar bekler ve tuşun hâlâ basılı olup olmadığını kontrol eder. Eğer tuşa hâlâ basılı ise karakter iç *FIFO*'ya yüklenir ve bir kesme istek sinyali gönderilir. Bu terazide, kesme istek çıkışı 8085A'nın *RESTART 5.5* girişine bağlıdır. Eğer *RST 5.5* kesmesi maskelenmemişse ve kesme yetkilendirilmişse, bu girişteki bir sinyal dönüş adresini yığına saklar ve programı *RST 5.5*. 'un adresi 2CH'ye atlatır. Bu adreste depolanmış bir atlama deyimi programı 5.5 kesmesi hizmetini yapan *KYBINT* yordamına götürür. Bu yordam 8279 FIFO'daki karakteri okur ve onu *KYBUF* adındaki tampona saklar. Gerekli olduğu zaman karakter *KYBUF*'tan okunabilir. Reset durumunda ya da bir kesme geldiğinde 8085A, *TRAP* dışındaki bütün kesmelerin yetkilendirmelerini kaldırır. Bu yüzden işlemcinin klavyeye cevap verebilmesi için kesmelerin maskeleri kaldırılmalı ve yetkilendirilmelidir.

Ağırlık alındıktan ve gösterildikten sonra program kesmelerin maskelerini kaldırır, ve yetkilendirir ve *KYBINT*'in *KYBUF*'a bir karakter depolayıp depolamadığını kontrol eder. *KYBUF*'tan okunan karakterin en büyük değerlikli bitinin 1 olması klavyeden bir karakter girilmediğini gösterir. Eğer karakter yoksa program sessiz terazi modülünün başına geri döner. Ağırlık yeniden alınır ve gösterilir. Daha sonra *KYBUF* klavyeden girilen karakter için yeniden kontrol edilir. Bu döngü bir tuş basılıncaya kadar devam eder. *KYBUF* içinde klavyeden girilen bir karakter bulunduğunda, program sessiz terazi modülünden çıkar.


```
LOC  OBJ       SIRA                    KAYNAK DEYİMİ
                   1 ;*****************************************************************
                   2 ; HALL 8085 10 LB. TERAZİ PROGRAMI SCALE
                   3 ; D.V. HALL
                   4 ; 12/11/78
                   5 ;*****************************************************************
                   6
                   7 ;-------------------------------------------
                   8 ; 8085 TABANLI TERAZİ İÇİN EŞİTLEMELER
                   9 ;-------------------------------------------
                  10
20A0               11 BUFFR1  EQU  20A0H
20A4               12 BUFFR2  EQU  20A4H
20A6               13 BUFFR3  EQU  20A6H
20AA               14 BUFFR4  EQU  20AAH
20AC               15 SAVBP   EQU  20ACH
20AE               16 KYBUF   EQU  20AEH
                  17
                  18 ;-------------------------------------------
0000  31C220     19 START:  LXI  SP,20C2H  ; YIĞIN GÖSTERİCİYİ RAM
                                            ; İÇİNDE AYARLA
0003  320019     20         MVI  A, 00     ; KLAVYE/GÖSTERGE
                                            ; MODUNU AYARLA
0005  3E00       21         STA  1900H
0008  3ECC       22         MVI  A,00      ; BOŞLUK KARAKTERİNİ SET ET
000A  320019     23         STA  1900H     ; 8279'UN KONTROLUNA GÖNDER
000D  3E00       24         MVI  A,0CCH    ; 8155'İN KOMUT/STATÜ
000F  D320       25         OUT  20H       ; KAYDEDİCİSİNİ AYARLA
0011  21A020     26 GETWT:  LXI  H,BUFFR1  ; KARAKTER DEPOSUNUN
                                            ; BAŞLANGIÇ ADRESİNİ YÜKLE
0014  DE00       27 DS1I:   IN   00        ; PORT 00'I OKU
0016  E610       28         ANI  10H       ; STROB İÇİN KONTROL ET
0018  CA1400     29         JZ   DS1       ; STROB BULUNUNCAYA
                                            ; KADAR DÖN
001B  DB00       30         IN   00        ; EN BÜYÜK DEĞERLİKLİ
                                            ; BASAMAK VERİSİNİ OKU
001D  E60F       31         ANI  0FH       ; ÜST 4 BİTİ MASKELE
001F  FE04       32         CPI  04        ; EN BÜYÜK DEĞERLİKLİ
                                            ; BASAMAK (BİT 3) = 1 Mİ?
0021  C22F00     33         JNZ  BLANK     ; EN BÜYÜK DEĞERLİKLİ
                                            ; BASAMAK = 0 İSE BOŞLUK
0024  3E01       34         MVI  A,01      ; EN BÜYÜK DEĞERLİKLİ
                                            ; BASAMAK = 1 İSE BİR YÜKLE
0026  C33100     35         JMP  STORE
0029  00         36         NOP
002A  00         37         NOP
002B  00         38         NOP
002C  C32801     39         JMP  KYBINT    ; RST 5.5 KLAVYE KESME ADRESİ
002F  3E15       40 BLANK:  MVI  A,15H      ; BOŞLUK KARAKTERİNİ YÜKLE
0031  77         41 STORE:  MOV  M,A       ; EN BÜYÜK DEĞERLİKLİ
                                            ; KARAKTERİ DEPOLA
0032  23         42         INX  H         ; GÖSTERGEYİ BİR SONRAKİ
                                            ; ADRESE HAREKET ETTİR
0033  C34000     43         JMP  DS2       ; RST 7.5 ADRESİ ÜZERİNDEN ATLA
0036  00         44         NOP
0037  00         45         NOP
0038  00         46         NOP
0039  00         47         NOP
```

Şekil 12.6 10 Paundluk terazi için çevirici dil programı

```
ADRES OBJ      SIRA       KAYNAK DEYİMİ

003A  00       48         NOP
003B  00       49         NOP
003C  C1       50         POP   B          ; RST 7.5 KESME YIĞININI BOŞALT
003D  C30000   51         JMP   START      ; HERHANGİ BİR NOKTADAN
                                            ; SESSİZ TERAZİYE DÖNÜŞ
0040  DB00     52  DS2:   IN    00         ; PORT 0'DAN GİRİŞ
0042  E620     53         ANI   20H        ; 2. BASAMAK STROBU İÇİN KONTROL ET
0044  CA4000   54         JZ    DS2        ; STROB = 1 OLUNCAYA KADAR DÖN
0047  DB00     55         IN    00         ; 2. BASAMAK VERİSİNİ AL
0049  E60F     56         ANI   0FH        ; ÜST 4 BİTİ MASKELE
004B  77       57         MOV   M,A        ; KARAKTERİ BELLEĞE DEPOLA
004C  23       58         INX   H          ; BELLEKTEKİ BİR SONRAKİ ADRESİ
                                            ; GÖSTER
004D  DB00     59  DS3:   IN    00         ; PORT 0'DAN GİRİŞ
004F  E640     60         ANI   40H        ; 3. BASAMAK STROBU İÇİN
                                            ; KONTROL ET
0051  CA4D00   61         JZ    DS3        ; STROB= 1 OLUNCAYA KADAR DÖN
0054  DB00     62         IN    00         ; 3. BASAMAK VERİSİNİ AL
0056  E60F     63         ANI   0FH        ; ÜST 4 BİTİ MASKELE
0058  77       64         MOV   M,A        ; 3. BASAMAĞI DEPOLA
0059  23       65         INX   H
005A  DB00     66  DS4;   IN    00
005C  E680     67         ANI   80H        ; EN KÜÇÜK DEĞERLİKLİ BASAMAK
                                            ; STROBUNU KONTROL ET
005E  CA5A00   68         JZ    DS4        ; STROB= 1 OLUNCAYA KADAR DÖN
0061  DB00     69         IN    00         ; EN KÜÇÜK DEĞERLİKLİ BASAMAK
                                            ; VERİSİNİ AL
0063  E60F     70         ANI   OFH        ; ÜST 4 BİTİ MASKELE
0065  77       71         MOV   M,A        ; EN KÜÇÜK DEĞERLİKLİ BASAMAĞI
                                            ; DEPOLA
0066  3E00     72         MVI   A,00H      ; GÖSTERGEYİ 4 LED BASAMAK
                                            ; İÇİN AYARLA
0068  21A020   73         LXI   H,BUFFR1   ; GÖSTERİCİYİ AĞIRLIĞIN İLK
                                            ; KARAKTERİNE AYARLA
006B  CD3901   74         CALL  DISPLY     ; AĞIRLIĞI GÖSTER
006E  21A420   75         LXI   H,BUFFR2   ; BİRİMLER TAMPONUNUN BAŞLANGIÇ
                                            ; ADRESİNİ YÜKLE
0071  3611     76         MVI   M,11H      ; "L" KODUNU YÜKLE
0073  23       77         INX   H          ; BİR SONRAKİ ADRESİ GÖSTER
0074  360B     78         MVI   M, 0BH     ; "B" KARAKTERİNİ YÜKLE
0076  3E01     79         MVI   A,01       ; GÖSTERGEYİ BİRİM LED'LERİNE
                                            ; AYARLA
0078  21A420   80         LXI   H,BUFFR2   ; GÖSTERİCİYİ BİRİMLERİN İLK
                                            ; KARAKTERİNE AYARLA
007B  CD3901   81         CALL  DISPLY     ; "LB"Yİ GÖSTER
007E  3E08     82         MVI   A,08H      ; KESMELERİN MASKESİNİ
                                            ; KALDIRAN KARAKTERİ YÜKLE
0080  30       83         SIM              ; KESMELERİN MASKESİNİ KALDIR

0081  FB       84         EI               ; KESMELERİ YETKİLENDİR
0082  21AE20   85         LXI   H,KYBUF    ; KYBINT TAMPONUNUN ADRESİNİ
                                            ; YÜKLE
0085  7E       86         MOV   A,M        ; KYBINT TAMPONUNU OKU
0086  B7       87         ORA   A          ; KARAKTERİ DEĞİŞTİRMEDEN
                                            ; İŞARET BAYRAĞINI SET ET
0087  F28E00   88         JP    EXIT1      ; SESSİZ TERAZİDEN ÇIK VE
                                            ; KARAKTER VARSA GÖSTER
008A  FB       89         EI               ; KARAKTER YOK, KESMELERİ
                                            ; YETKİLENDİR
```

Şekil 12.6 (Devam ediyor)


```
ADRES OBJ      SIRA       KAYNAK DEYİMİ

008B  C31100   90         JMP   GETWT      ; SESSİZ TERAZİYE DÖN
008E  3680     91  EXIT1: MVI   M,80H      ; KARAKTERİN OKUNDUĞUNU
                                            ; GÖSTERMEK İÇİN
0090  FB       92         EI               ; KYBUF'UN EN BÜYÜK DEĞERLİKLİ
                                            ; BİTİNE 1 KOY
0091  57       93         MOV   D,A        ; KARAKTERİ SAKLA
0092  21AA20   94         LXI   H,BUFFR4
0095  3605     95         MVI   M,05H      ; "S"
0097  23       96         INX   H
0098  3612     97         MVI   M,12H      ; "p"
009A  3E01     98         MVI   A,01
009C  21AA20   99         LXI   H,BUFFR4   ;
009F  CD3901   100        CALL  DİSPLY     ; BİRİM ALANINDA SP. GÖSTER
00A2  7A       101        MOV   A,D        ; SAKLANAN KARAKTERİ A'YA TAŞI
00A3  110000   102        LXI   D,0000     ; D VE E'Yİ SİL
00A6  CD6F01   103 SHOWSP: CALL  INSTR      ; YENİ KARAKTERİ PAKETLENMİŞ
                                            ; BCD OLARAK D VE E'YE YÜKLE
00A9  CD7801   104        CALL  EXPND      ; GÖSTERMEK İÇİN KARAKTER/BAYT
                                            ; OLARAK GENİŞLET
00AC  21A620   105        LXI   H,BUFFR3   ; GÖSTERGE TAMPONUNUN
                                            ; BAŞLANGIÇ ADRESİNİ PAUND
              106                           ; BAŞINA FİYAT İÇİN YÜKLE
00AF  3E00     107        MVI   A,00       ; GÖSTERGEYİ 4 BASAMAK İÇİN
                                            ; AYARLA
00B1  CD3901   108        CALL  DISPLY     ; FİYAT/LB'NİN GİRİLEN
                                            ; BASAMAĞINI GÖSTER
00B4  CD9701   109        CALL  RDKY       ; BİR SONRAKİ KARAKTERİ
                                            ; KLAVYEDEN OKU
00B7  FE0A     110        CPI   0AH        ; SON KARAKTER Mİ?
00B9  DAA600   111        JC    SHOWSP     ; HAYIR - KARAKTERİ GÖSTER VE
                                            ; KLAVYEYİ OKU
00BC  CDA701   112        CALL  BCDCVT     ; FİYAT/LB DE İÇİNDE, HL İÇİNDE
                                            ; İKİLİK SAYIYA ÇEVİR
00BF  EB       113        XCHG             ; İKİLİK FİYATI DE'YE TAŞI
00C0  21AC20   114        LXI   H,SAVBP
00C3  73       115        MOV   M,E        ; İKİLİK FİYAT/LB'Yİ BELLEKTE SAKLA
00C4  23       116        INX   H
00C5  72       117        MOV   M,D
00C6  21A020   118        LXI   H,BUFFR1   ; BCD AĞIRLIĞI BCDVT İÇİN
                                            ; PAKETLEMEYE BAŞLA
00C9  7E       119        MOV   A,M        ; EN BÜYÜK DEĞERLİKLİ KARAKTERİ
                                            ; YÜKLE
00CA  FE15     120        CPI   15H        ; EN BÜYÜK DEĞERLİKLİ KARAKTERİ
                                            ; BOŞLUK MU?
00CC  C2D000   121        JNZ   SHIFT      ; HAYIR, SHIFT'E GİT
00CF  AF       122        XRA   A          ; EVET, AKÜMÜLATÖR VE ELDEYİ SİL
00D0  07       123 SHIFT: RLC              ; ÜST DÖRT BİTE GİT
00D1  07       124        RLC
00D2  07       125        RLC
00D3  07       126        RLC
00D4  23       127        INX   H          ; AĞIRLIĞIN BİR SONRAKİ BCD
                                            ; KARAKTERİNİ GÖSTER
00D5  86       128        ADD   M          ; A'NIN ALT 4 BİTİNE EKLE
00D6  57       129        MOV   D,A        ; BCDCVT İÇİN D'YE KOY
00D7  23       130        INX   H          ; AĞIRLIĞIN BİR SONRAKİ
                                            ; KARAKTERİNİ GÖSTER
00D8  7E       131        MOV   A,M        ; BCD KARAKTERİ YÜKLE
00D9  07       132        RLC              ; ÜST DÖRT BİTE KAY
00DA  07       133        RLC
```

Şekil 12.6 (Devam ediyor)

KAYNAK DEYİMİ

```
ADRES OBJ      SIRA
                 134          RLC
                 135          RLC
00DB 07          136          INX   H          ; EN KÜÇÜK DEĞERLİKLİ BCD
                              ;                ; KARAKTERİNİ GÖSTER
00DC 07          137          ADD   M          ; ALT DÖRT BİTİ EKLE
00DD 23          138          MOV   E,A        ; BCD AĞIRLIK ŞİMDİ DE İÇİNDE
00DE 86          139          CALL  BCDCVT     ; AĞIRLIĞI HL İÇİNDE İKİLİK SAYIYA
00DF 5F                       ;                ; ÇEVİR
00E0 CDA701      140          XCHG             ; İKİLİK AĞIRLIĞI DE'YE TAŞI
00E3 EB          141          LXI   H,SAVBP    ; İKİLİK FİYAT/LB'Yİ GERİ AL
00E4 21AC20      142          MOV   C,M
00E7 4E          143          INX   H
00E8 23          144          MOV   B,M
00E9 46          145          CALL  MULTO      ; AĞIRLIK İLE FİYAT/LB'Yİ ÇARP
00EA CDD001      146          ;                ;       BC X DE= DEHL
                 147          CALL  BINCVT     ; E, H, L KAYDEDİCİLERİ İÇİNDEKİ
00ED CDEE01                   ;                ; İKİLİK
                 148          ;                ; TOPLAM FİYATI HLBC İÇİNDE
                              ;                ; PAKETLİ BCD'YE ÇEVİR
                 149          MVI   A,49H      ; EN YAKIN SENTE YUVARLAMAYI
00F0 3E49                     ;                ; BAŞLAT
                 150          CMP   C          ; C>49H İSE ELDE SET
00F2 B9          151          MVI   A,00       ; AKÜMÜLATÖRÜ SİL, ELDE AYNI
00F3 3E00                     ;                ; KALSIN
                 152          ADC   B          ; ELDEYİ DİĞER BASAMAKLARA YAY
00F5 88          153          DAA              ; BCD FORMATI KORU
00F6 27          154          MOV   E,A        ; EXPND İÇİN E'YE TAŞI
00F7 5F          155          MVI   A,00
00F8 3E00        156          ADC   L          ; ELDEYİ SONRAKİ BASAMAKLARA YAY
00FA 8D          157          DAA              ; VE BCD FORMATI KORU
00FB 27          158          MOV   D,A        ; EXPND İÇİN D'YE TAŞI
00FC 57          159          CALL  EXPND      ; KARAKTER/BAYT'A GENİŞLET
00FD CD7801      160          ;                ; SONUÇ BUFFR 3'TE
                 161          LXI   H,BUFFR3   ; BUFFR3'ÜN BAŞLANGICINI YÜKLE
0100 21A620      162          MVI   A,00       ; 4 BASAMAKLI GÖSTERGE İÇİN
0103 3E00                     ;                ; AYARLA
                 163          CALL  DISPLY     ; TOPLAM FİYATI GÖSTER
0105 CD3901      164          LXI   H,BUFFR4   ; BİRİM TAMPONUNU GÖSTER
0108 21AA20      165          MVI   M,12H      ; "p"
010B 3612        166          INX   H          ; BİR SONRAKİ ADRESİ GÖSTER
010D 23          167          MVI   M, 14H     ; "R"
010E 3614        168          DCX   H          ; TAMPONUN BAŞLANGICINI GÖSTER
0110 2B          169          MVI   A,01       ; BİRİM GÖSTERGE İÇİN AYARLA
0111 3E01        170          CALL  DISPLY     ; "PR" GÖSTER
0113 CD3901      171          LXI   D,0000H    ; UZUN DÖNGÜ GECİKME SABİTİNİ
0116 110000                   ;                ; YÜKLE
0119 1B          172 LNGLP:   DCX   D          ; UZUN DÖNGÜYÜ AZALT
011A 0608        173          MVI   B,08H      ; KISA DÖNGÜ GECİKME SABİTİNİ
                              ;                ; YÜKLE
011C 05          174 SHRTLP:  DCR   B          ; KISA DÖNGÜYÜ AZALT
011D C21C01      175          JNZ   SHRTLP     ; B=0 OLUNCAYA KADAR DÖN
0120 7A          176          MOV   A,D        ; UZUN DÖNGÜ SABİTİ
0121 B3          177          ORA   E          ; SIFIR MI?
0122 C21901      178          JNZ   LNGLP      ; SIFIR OLUNCAYA KADAR DÖN (3SN)
0125 C31100      179          JMP   GETWT      ; SESSİZ TERAZİYE ATLA
```

Şekil 12.6 (Devam ediyor)

KAYNAK DEYİMİ

```
ADRES OBJ      SIRA
                 181 ;*****************************************************************
                 182 ;KYBINT - 8279'UN ÇIKIŞ KESMESİNE HİZMET VERİR
                 183 ;         - KARAKTER 8279'DAN OKUNUR VE KYBUF
                              ;           TAMPONUNA DEPOLANIR
                 184 ;GİRİŞ YOK
                 185 ;ÇIKIŞ YOK
                 186 ;ÇAĞRILAN YOK
                 187 ;BOZULAN YOK
                 188 ;*****************************************************************
                 189 ;
0128 E5          190 KYBINT: PUSH  H          ; HL'Yİ SAKLA
0129 F5          191         PUSH  PSW        ; AVE FF'LERİ SAKLA
012A 210019      192         LXI   H,1900H    ; 8279'UN KONTROL ADRESİ
012D 3640        193         MVI   M,40H      ; 8279 OKUMA KLAVYESİ İÇİN
                             ;                ; KONTROL KARAKTERİ ÇIKIŞI
012F 25          194         DCR   H          ; 8279 OKUMA ADRESİNİ GÖSTER
0130 7E          195         MOV   A,M        ; KARAKTERİ FIFO'DAN OKU
0131 E60F        196         ANI   0FH        ; ÜST DÖRT BİTİ MASKELE
0133 32AE20      197         STA   KYBUF      ; KARAKTERİ KYRUF'A DEPOLA
0136 F1          198         POP   PSW        ; AVE FF'LERİ YENİDEN YÜKLE
0137 E1          199         POP   H          ; HL'Yİ YENİDEN YÜKLE
0138 C9          200         RET
                 201
                 201 ;*****************************************************************
                 203 ;DISPLY - KARAKTERLERİ BELLEKTEKİ TAMPONDAN OKUR, + 7 PARÇALI
                 204 ;      - KODA ÇEVİRİR VE GÖSTERİLMEK ÜZERE 8279'A GÖNDERİR
                 205 ;GİRİŞLER: AKAMÜLATÖR - 00= 4 BASAMAKLI ALANDA GÖSTER
                 206 ;                      - 01= 2 BASAMAKLI ALANDA GÖSTER
                 207 ;                  HL - TAMPON BAŞLANGIÇ ADRESİ
                 208 ;ÇIKIŞLAR: YOK
                 209 ;ÇAĞRILAN: YOK
                 210 ;BOZULANLAR: A, H, L, FF'LER
                 211 ;*****************************************************************
                 212
0139 D5          213 DISPLY: PUSH  D          ; DE'Yİ SAKLA
013A C5          214         PUSH  B          ; BC'Yİ SAKLA
013B 0E04        215         MVI   C,04H      ; KARAKTER SAYICIYI 4 YAP
013D 0F          216         RRC              ; GÖSTERGE 2 BASAMAKLI ALAN İÇİN Mİ?
013E DA4601      217         JC    DISP1      ; EVET
0141 3E90        218         MVI   A,90H      ; 4 BASAMAKLI ALAN İÇİN 8279
                             ;                ; KONTROL KELİMESİ
0143 C34801      219         JMP   DISP2      ;
0146 3E94        220 DISP1:  MVI   A,94H      ; 2 BASAMAKLI ALAN İÇİN 8279
                             ;                ; KONTROL KELİMESİ
0148 320019      221 DISP2:  STA   1900H      ; KONTROL KELİMESİNİ 8279'A GÖNDER
014C 7E          222 DISP3:  MOV   A,M        ; KARAKTERİ TAMPONDAN AL
014C EB          223         XCHG             ; GÖSTERİCİYİ DE İÇİNDE SAKLA
014D 212802      224         LXI   H,DSPTB    ; 7 PARÇALI TABLONUN ÜST
                             ;                ; ADRESİNİ YÜKLE
0150 85          225         ADD   L          ; TABLO İÇİNDEKİ 7 PARÇALIYI
                             ;                ; GÖSTERMEK İÇİN
0151 6F          226         MOV   L,A        ; HL'YE BCD KARAKTERİ EKLE
0152 3E00        227         MVI   A,00       ; AKÜMÜLATÖRÜ SİL, ELDE AYNI KALSIN
0154 8C          228         ADC   H          ; ELDEYİ H'YE TAŞI
0155 67          229         MOV   H,A        ; H'Yİ YENİDEN YÜK'LE
0156 6E          230         MOV   L,M        ; 7 PARÇALI KODU L'YE TAŞI
0157 0D          231         DCR   C          ; KARAKTER SAYICIYI AZALT
0158 79          232         MOV   A,C        ;
```

Şekil 12.6 (Devam ediyor)

```
ADRES OBJ      SIRA      KAYNAK DEYİMİ

0159  FE02      233            CPI   02H       ;SOLDAN İKİNCİ KARAKTER Mİ?
015B  7D        234            MOV   A,L       ;7 PARÇALIYI AKÜMÜLATÖRE TAŞI
015C  C26101    235            JNZ   DISP4     ;İKİNCİ KARAKTER DEĞİLSE GÖSTER
015F  F608      236            ORI   08H       ;İKİNCİ KARAKTERDEN SONRA
                                               ;ONDALIK NOKTAYI MASKELE
0161  2F        237  DISP4:    CMA             ;KODUN TÜMLEYENİ
0162  320018    238            STA   1800H     ;KARAKTERİ 8279'A GÖNDER
0165  EB        239            XCHG            ;TAMPON GÖSTERİCİYİ DE'DEN
                                               ;HL'YE GERİ
0166  23        240            INX   H         ;GETİR VE ARTTIR
0167  79        241            MOV   A,C       ;KARAKTER SAYICI SIFIR MI?
0168  B7        242            ORA   A         ;BAYRAKLARI SET ET
0169  C24B01    243            JNZ   DISP3     ;SIFIR DEĞİLSE BİR SONRAKİ
                                               ;KARAKTERİ AL
016C  C1        244            POP   B
016D  DI        245            POP   D
016E  C9        246            RET
                247
                248 ;*************************************************************************
                249 ;INSTR  -  DE'NİN İÇERİĞİNİ SOLA 4 BİT KAYDIRIR VE YENİ
                250 ;            BCD KARAKTERİ E'NİN ALT 4 BİTİNE YERLEŞTİRİR
                251 ;GİRİŞLER  A : SOKULACAK BCD SAYI
                252 ;          DE - 4 PAKETLENMİŞ BCD SAYILAR
                253 ;ÇIKIŞLAR  DE - E ALT NIBIL'DA A'NIN OLDUĞU
                254 ;          4 PAKETLENMİŞ BCD SAYILAR
                255 ;ÇAĞRILAN YOK
                256 ;BOZULAN A, FF'LER
                257 ;*************************************************************************
016F  EB        258 INSRT: XCHG            ;DE'Yİ HL'YE TAŞI
0170  29        259        DAD   H         ;H VE L'Yİ SOLA 4 BİT KAYDIR
0171  29        260        DAD   H
0172  29        261        DAD   H
0173  29        262        DAD   H
0174  85        263        ADD   L         ;YENİ BCD RAKAMI GİR
0175  6F        264        MOV   L,A       ;L'YE TAŞI
0176  EB        265        XCHG            ;HL'DEN DE'YE TAŞI
0177  C9        266 RET
                267
                268 ;*************************************************************************
                269 ;EXPND  -  DE İÇİNDEKİ 4 PAKETLENMİŞ BCD KARAKTERLERİ
                270 ;          BELLEKTEKİ TAMPON İÇİNE KARAKTER BAŞINA
                271 ;          BAYT OLARAK GENİŞLETİR
                272 ;GİRİŞLER:  DE - 4 PAKETLENMİŞ BCD SAYI
                273 ;ÇIKIŞLAR:  YOK
                274 ;ÇAĞRILAN: YOK
                275 ;BOZULAN : A, HL, FF'LER
                276 ;*************************************************************************
0178  7A        277 EXPND: MOV   AD,       ;İLK 2 BCD RAKAMI AL
0179  0F        278        RRC             ;EN BÜYÜK DEĞERLİKLİ RAKAMI ALT
017A  0F        279        RRC             ;NİBILA KAYDIR
017B  0F        280        RRC
017C  0F        281        RRC
017D  E60F      282        ANI   0FH       ;ÜST DÖRT BİTİ MASKELE
017F  21A620    283        LXI   H, BUFFR3
0182  77        284        MOV   M,A       ;TAMPONA DEPOLA
0183  7A        285        MOV   A,D       ;İLK 2 BCD RAKAMI YENİDEN AL
0184  E60F      286        ANI   0FH       ;ÜST DÖRT BİTİ MASKELE
```

Şekil 12.6 (Devam ediyor)


```
ADRES OBJ      SIRA      KAYNAK DEYİMİ

0186  23        287            INX   H         ;B4FFR3 İÇİNDEKİ BİR SONRAKİ
                                               ;ADRESİ GÖSTER
0187  77        288            MOV   M,A       ;İKİNCİ EN BÜYÜK DEĞERLİKLİ
                                               ;BASAMAĞI DEPOLA
0188  7B        289            MOV   A,E       ;İKİNCİ 2 BCD RAKAMI AL
0189  0F        290            RRC             ;BİR SONRAKİ BCD RAKAMI ALT
                                               ;NİBILA KAYDIR
018A  0F        291            RRC
018B  0F        292            RRC
018C  0F        293            RRC
018D  E60F      294            ANI   0FH       ;ÜST 4 BİTİ BASKELE
018F  23        295            INX   H
0190  77        296            MOV   M,A       ;3. BCD RAKAMI DEPOLA
0191  7B        297            MOV   A,E       ;ALT BAYTI YENİDEN AL
0192  E60F      298            ANI   0FH       ;ÜST 4 BİTİ MASKELE
0194  23        299            INX   H         ;BİR SONRAKİ BUFFR3 POZİSYONU
0195  77        300            MOV   M,A       ;4. BCD RAKAMI DEPOLA
0196  C9        301            RET
                302
                303 ;*************************************************************************
                304 ;RDKY  -  KLAVYE KESME YORDAMI KYBINT İLE YÜKLENEN
                305 ;          KYBUF'U KONTROL EDER. KARAKTER
                306 ;          GİRİLMEMİŞSE GİRİLİNCEYE KADAR BEKLER.
                307 ;          GİRİLEN KARAKTER A İÇİNDE DÖNER
                308 ;GİRİŞLER:  YOK
                309 ;ÇIKIŞLAR:  A - KLAVYEDEN OKUNAN KARAKTER
                310 ;ÇAĞRILAN: YOK
                311 ;BOZULAN:  A, H, L, FF'LER
                312 ;*************************************************************************
                313 ;
0197  21AE20    314 RDKY:  LXI   H, KYBUF  ;KYBUF ADRESİNİ GÖSTER
019A  7E        315        MOV   A,M       ;KYBUF'U OKU
0198  B7        316 ORA    A               ;BAYRAKLARI SET ET, BİT = 7
                                           ;İŞARET 0
019C  F2A301    317        JP    EXIT2     ;İSE KARAKTER MEVCUT
019F  FB        318        EI              ;KARAKTER YOK-KESMELERİ YETKİLENDİR
01A0  C39701    319        JMP   RDKY      ;VE RDKY GERİ DÖN
01A3  3680      320 EXIT2: MVI   M,80H     ;KYBUF'UN EN BÜYÜK DEĞERLİKLİ
                                           ;BİTİNİ TAMPONUN BOŞ OLDUĞUNU
                321                        ;GÖSTERMEK İÇİN 1 YAP
01A5  FB        322        EI              ;KESMELERİ YETKİLENDİR
01A6  C9        323        RET
                324
                325 ;*************************************************************************
                326 ;BCDCVT  -  D VE E İÇİNDEKİ 4 PAKETLENMİŞ BCD SAYIYI HL
                327 ;          İÇİNDE 16 BİTLİK İKİLİK SAYIYA ÇEVİRİR
                328 ;GİRİŞLER:  DE - 4 PAKETLENMİŞ BCD SAYI
                329 ;ÇIKIŞLAR:  HL - 16 BİTLİK İKİLİK EŞDEĞERİ
                330 ;ÇAĞRILAN: MSBCD, LSBCD
                331 ;BOZULAN:  A, B, C, D, E, H, L, FF'LER
                332 ;*************************************************************************
                333
01A7  210000    334 BCDCVT: LXI   H,0000H  ;HL'Yİ SİL
01AA  01E803    335         LXI   B,03E8H  ;BC'YE 03E8H= 1000 ONDALIK YÜKLE
01AD  7A        336         MOV   A,D      ;BCD BİNLER VE YÜZDELERİ YÜKLE
01AE  CDC701    337         CALL  MSBCD    ;EN BÜYÜK DEĞERLİKLİ BCD
                                           ;RAKAMI ÇEVİR
01B1  016400    338         LXI   B,0064 H ;BC'YE 64 H= 100 ONDALIK YÜKLE
01B4  CDBF01    339         CALL  LSBCD    ;D'NİN EN KÜÇÜK DEĞERLİKLİ BCD
                                           ;RAKAMINI ÇEVİR
```

Şekil 12.6 (Devam ediyor)

KAYNAK DEYİMİ

```
ADRES OBJ      SIRA
01B7  7B       340          MOV   A,E       ; BCD RAKAMLARI E'DEN YÜKLE
01B8  0E0A     341          MVI   C,0AH     ; C'YE 0AH= 10 ONDALIK YÜKLE
01BA  CDC701   342          CALL  MSBCD     ; E İÇİNDEKİ ÜST BCD RAKAMI ÇEVİR
01BD  0E01     343          MVI   C,01H     ; C'YE 01H= 01 ONDALIK YÜKLE
01BF  FE00     344  LSBCD: CPI   00H       ; A SIFIRA AZALDI MI?
01C1  C8       345          RZ              ; SIFIRSA DÖN
01C2  09       346          DAD   B         ; BC'Yİ HL'YE TOPLA
01C3  3D       347          DCR   A         ; A İÇİNDEKİ BCD'Yİ AZALT
01C4  C3BF01   348          JMP   LSBCD
01C7  FE0A     349  MSBCD: CPI   0AH       ; EN BÜYÜK DEĞERLİKLİ BCD SIFIRA
                                            ; AZALTILDI MI?
01C9  D8       350          RC              ; A 10 ONDALIKTAN KÜÇÜK İSE
                                            ; ELDEYİ SET ET
01CA  09       351          DAD   B         ; BC'Yİ HL'YE TOPLA
01CB  D610     352          SUI   10H       ; EN BÜYÜK DEĞERLİKLİ BCD'DEN
                                            ; BİR ÇIKAR
01CD  C3C701   353          JMP   MSBCD
               354
               355  ;*************************************************************
               356  ;MULTO -  DE İÇİNDEKİ 16 BİTLİK İKİLİK SAYI İLE BC İÇİNDEKİ
               357  ;         16 BİTLİK İKİLİK SAYIYI ÇARPAR SONUÇ DE VE
                             HL İÇİNDEKİ 32 BİTLİK SAYI
               358  ;GİRİŞLER:  BC - 16 BİTLİK İKİLİK SAYI
               359  ;         DE - 16 BİTLİK İKİLİK SAYI
               360  ;ÇIKIŞLAR:  DEHL - 32 BİTLİK İKİLİK SAYI
               361  ;ÇAĞRILAN: YOK
               362  ;BOZULAN;  A, D, E, H, L, FF'LER
               363  ;*************************************************************
               364
01D0  210000   365  MULTO: LXI   H,0000H   ; HL'Yİ SİL
01D3  3E10     366          MVI   A,10H     ; BİT SAYICIYI 16 ONDALIĞA AYARLA
01D5  29       367  LOOP2: DAD   H         ; HL İÇİNDEKİ KISMİ ÇARPIMIN EN
               368                          ; BÜYÜK DEĞERLİKLİ BİTİNİ SOLA
                                            ; ELDEYE KAYDIR
01D6  EB       369          XCHG            ; DE VE HL'Yİ KARŞILIKLI DEĞİŞTİR
01D7  DADE01   370          JC    MULT1     ; HL'NİN KAYMASI ELDE
                                            ; ÜRETMİŞSE ATLA
01DA  29       371          DAD   H         ; DE'Yİ HL İÇİNDE SOLA KAYDIR
01DB  C3E001   372          JMP   MULT2     ; ÇARPANIN BİR SONRAKİ BİTİNİ
                                            ; KONTROL ET
01DE  29       373  MULT1: DAD   H         ; BİR SONRAKİ BİTİ KONTROL İÇİN
                                            ; DE'Yİ HL'YE SOLA KAYDIR
01DF  23       374          INX   H         ; HL'DE ÇIKAN ELDEYİ ŞİMDİ HL
                                            ; İÇİNDEKİ DE İLE TOPLA
01E0  EB       375  MULT2: XCHG            ; KİSMİ ÇARPIMI HL'YE TAŞI
01E1  D2E901   376          JNC   MULT3     ; ELDE YOK - ÇARPAN BİTİ= 0
               377                          ; BİR SONRAKİ BİTİ KONTROL İÇİN DÖN
01E4  09       378          DAD   B         ; ÇARPAN BİTİ=1, ÇARPILANI TOPLA
01E5  D2E901   379          JNC   MULT3     ; TOPLAMADAN ELDE YOK - MULT3'E ATLA
01E8  13       380          INX   D         ; ELDE - DE İÇİNDEKİ SONCUN ÜST
                                            ; KELİMESİNE TOP
01E9  3D       381  MULT3: DCR   A         ; BİT SAYICIYI AZALT
01EA  C2D501   382          JNZ   LOOP2     ; 16 BİT BİTİNCEYE KADAR DÖN
01ED  C9       383          RET             ; DÖN
               384
```

Şekil 12.6 (Devam ediyor)

KAYNAK DEYİMİ

```
ADRES OBJ      SIRA
               385  ;*************************************************************
               386  ;BINCVT -  E, H, L İÇİNDEKİ 24 BİTLİK İKİLİK SAYIYI H, L, B, C
               387  ;         İÇİNDEKİ PAKETLENMİŞ BCD EŞDEĞERİNE ÇEVİRİR
               388  ;GİRİŞLER:  E, H, L - İKİLİK SAYI
               389  ;ÇIKIŞLAR:  HLBC - PAKETLENMİŞ BCD SONUCU
               390  ;ÇAĞRILAN: CNVT1, CNVT2
               391  ;BOZULAN:  A, B, C, D, E, H, L, FF'LER
               392  ;*************************************************************
               393
01EE  1619     394  BINCVT: MVI   D,19H     ; BİT SAYICIYI ONDALIK 25'E AYARLA
01F0  CD0902   395          CALL  CNVTI     ; 2 EN KÜÇÜK DEĞERLİKLİ BCD
                                            ; SAYIYI B İÇİNDE ÜRET
01F3  48       396          MOV   C,B       ; 2 BCD SAYIYI C'YE TAŞI
01F4  1619     397          MVI   D,19H     ; BİT SAYICIYI ONDALIK 25'E AYARLA
01F6  CD0902   398          CALL  CNVTI     ; DİĞER 2 BCD SAYIYI ÜRET
01F9  C5       399          PUSH  B         ; 4 EN KÜÇÜK DEĞERLİKLİ BCD
                                            ; SAYIYI YIĞINA TAŞI
01FA  1619     400          MVI   D,19H     ; BİT SAYICIYI ONDALIK 25'E AYARLA
01FC  CD0902   401          CALL  CNVTI     ; 2 BCD SAYIYI B İÇİNDE ÜRET
01FF  48       402          MOV   C,B       ; 2 BCD SAYIYI C'YE TAŞI
0200  1619     403          MVI   D,19H     ; BİT SAYICIYI ONDALIK 25'E AYARLA
0202  CD0902   404          CALL  CNVTI     ; SON 2 BCD SAYIYI ÜRET
0205  60       405          MOV   H,B       ; EN BÜYÜK DEĞERLİKLİ BCD'Yİ
                                            ; DÖNÜŞ İÇİN YERLEŞTİR
0206  69       406          MOV   L,C       ; SONRAKİ 2 BCD'Yİ DÖNÜŞ İÇİN YERLEŞTİR.
0207  C1       407          POP   B         ; 4 BCD RAKAMI YIĞINDAN YÜKLE
0208  C9       408          RET
0209  AF       409  CNVT1: XRA   A         ; AKÜMÜLATÖR VE ELDEYİ SİL
020A  47       410          MOV   B,A       ; B'Yİ SIFIR YAP
020B  AF       411  CNVT2: XRA   D         ; AKÜMÜLATÖR VE ELDEYİ SİL
020C  15       412          DCR   D         ; BİT SAYICIYI AZALT
020D  C8       413          RZ              ; SIFIR İSE BİTTİ
020E  29       414          DAD   H         ; H'Yİ 1 BİT SOLA KAYDIR - EN BÜYÜK
                                            ; DEĞERLİKLİ BİT ELDEYE
020F  7B       415          MOV   A,E       ; ÜST SEVİYELİ BAYTI AKÜMÜLATÖRE TAŞI
0210  17       416          RAL             ; H'DEN GELEN ELDEYİ EN KÜÇÜK
                                            ; DEĞERLİKLİ BİTE DÖNDÜR
0211  5F       417          MOV   E,A       ; E'Yİ YENİDEN YÜKLE
0212  78       418          MOV   A,B       ; BCD SAYIYI A'YA TAŞI
0213  8F       419          ADC   A         ; A'YI 2 İLE ÇARP VE E'DEN GELEN İLE TOPLA
0214  27       420          DAA             ; BCD FORMATI DEVAM ETTİR
0215  47       421          MOV   B,A       ; B'Yİ SAKLA
0219  3E00     423          MVI   A,00H     ;DAA'DAN ELDE YOK - DEVAM
021B  8D       424          ADC   L         ; ELDE VARSA L'YE TOPLA
021C  6F       425          MOV   L,A       ; L'Yİ YENİDEN YÜKLE
021D  3E00     426          MVI   A,00H     ; AKÜMÜLATÖRÜ SİL, ELDE AYNI KALSIN
021F  8C       427          ADC   H         ; ELDEYİ H'YE YAY
0220  67       428          MOV   H,A       ; H'Yİ YENİDEN YÜKLE
0221  3E00     429          MVI   A,00H
0223  8B       430          ADC   E         ; ELDEYİ E'YE YAY
0224  5F       431          MOV   E,A       ; E'Yİ YENİDEN YÜKLE
0225  C30B02   432          JMP   CNVT2     ; DEVAM
               433
               434  ;*************************************************************
               435  ;DSPTB - GÖSTERGE ÇIKIŞI İÇİN BCD KARAKTERLERİ 7
               436  ;         PARÇALI KODA ÇEVİRMEK İÇİN TABLO
               437  ;*************************************************************
               438
```

Şekil 12.6

```
ADRES OBJ    SIRA                    KAYNAK DEYİMİ

0228  F3      439  DSPTB:  DB   0F3H    ;"0"
0229  60      440          DB   060H    ;"1"
022A  B5      441          DB   0B5H    ;"2"
022B  F4      442          DB   0F4H    ;"3"
022C  66      443          DB   066H    ;"4"
022D  D6      444          DB   0D6H    ;"5"
022E  D7      445          DB   0D7H    ;"6"
022F  70      446          DB   070H    ;"7"
0230  F7      447          DB   0F7H    ;"8"
0231  76      448          DB   076H    ;"9"
0232  77      449          DB   077H    ;"A"
0233  C7      450          DB   0C7H    ;"B"
0234  93      451          DB   093H    ;"C"
0235  E5      452          DB   0E5H    ;"D"
0236  97      453          DB   097H    ;"E"
0237  17      454          DB   017H    ;"F"
0238  67      455          DB   067H    ;"H"
0239  83      456          DB   083H    ;"L"
023A  37      457          DB   037H    ;"P"
023B  60      458          DB   060H    ;"T"
023C  05      459          DB   005H    ;"R"
023D  00      460          DB   000H    ;BOŞLUK
              461
              462          END

UMUMİ SEMBOLLER

HARİCİ SEMBOLLER

KULLANICI SEMBOLLERİ

BCDCVT  A  01A7   BINCVT  A  01EE   BLANK  A  002F   BUFFR1  A  20A0
CNVT1   A  0209   CNVT2   A  020B   DISP1  A  0146   DISP2   A  0148
DS1     A  0014   DS2     A  0040   DS3    A  004D   DS4     A  005A
EXPND   A  0178   GETWT   A  0011   INSRT  A  016F   KYBINT  A  0128
LSBCD   A  01BF   MSBCD   A  01C7   MULT1  A  01DE   MULT2   A  01E0
SAVBP   A  20AC   SHIFT   A  00D0   SHOWSP A  00A6   SHRTLP  A  011C
BUFFR2  A  20A4   START   A  0000   LNGLP  A  0119   DISPLY  A  0139
DISP3   A  014B   BUFFR3  A  20A6   MULTO  A  01D0   EXIT2   A  01A3
DSPTB   A  0228   DISP4   A  0161   STORE  A  0031   LOOP2   A  01D5
KYBUF   A  20AE   EXIT1   A  008E   BUFFR4 A  20AA   RDKY    A  0197
MULT3   A  01E9

ASSEMBLE TAMAM
```

Şekil 12.6 (Devam ediyor)

FİYAT/LB MODÜLÜ: Paund başına fiyatın ilk rakamı aslında sessiz terazi modülünden çıkmadan önce son döngü sırasında okunur. Bu rakam D kaydedicisine depolanır ve birim göstergesi satış fiyatının girildiğini belirten SP.. 'ye çevrilir.

Daha sonra ilk rakam akümülatöre taşınır, D ve E kaydedicileri sıfırlanır, ve SHOWSP adındaki diğer bir döngü çağrılır. Bu döngünün fonksiyonları 1) Paund başına fiyatı, D ve E kaydedicileri içinde paketlenmiş dört BCD rakam olarak yaratmak 2) Dört BCD rakamı gösterilmeleri için her biri 1 bayt olacak şekilde bir tampon içinde genişletmek 3) Bitiş tuşu basılana kadar basılan her tuşu alıp göstermek olarak sayılabilir. Karakterler birçok hesap makinesinde olduğu gibi göstergeye sağ taraftan girilir. Yeni bir karakter girildiğinde, daha önceden girilen karakterler yeni karaktere yer açmak üzere bir karakter sola kaydırılır.

Yeni bir BCD karakterine E kaydedicisinin alt 4 bitinde yer açabilmek için INSTR alt yordamı D ve E kaydedicilerinin içeriklerini 4 bit sola kaydırır. EXPND, D ve E içindeki dört paketli BCD rakamı bellekteki BUFFR3 içinde her karakter başına bir bayt olarak genişletir. Bu gösterme yordamı DISPLY tarafından istenen formattır.

Sessiz terazi modülünün son kısmında girilen karakter okunduktan, genişletildikten ve gösterildikten sonra bir sonraki karakteri KYBUF'tan okumak için bir klavye okuma yordamı olan RDKY çağrılır. RDKY bir tuş basılıncaya kadar bekler ve basılan karakteri akümülatöre getirir. Basılan tuşun geçerli bir BCD rakamı veya paund başına fiyatın girilmesinin bittiğini belirten karakter olduğunu belirlemek için girilen tuş 0AH ile karşılaştırılır. Eğer 0AH'den küçük (geçerli BCD) ise, elde bayrağı bir yapılır. Daha sonra JC deyimi programı yeni rakam için SHOWSP, INSTR, EXPND ve DISPLY yordamlarına geri döndürür. Geçerli BCD rakamdan başka herhangi bir karakter işlemcinin SHOWSP'den çıkıp bir sonraki modüle geçmesini sağlar.

Bir sonraki modüle gitmeden önce; bu noktada, her şeyin nerede olduğunu takip edebilmek için bir mini-RAM ve kaydedici haritası çizmek iyi bir düşüncedir. Şekil 12.7 böyle bir haritayı göstermektedir. Ağırlık BUFFR1 olarak adlandırılan dört bellek bölgesinde, paund başına fiyat BUFFR3 adlı dört bellek bölgesinde, ve paund başına fiyat aynı zamanda paketlenmiş şekilde D ve E yazmaçları içinde depolanmıştır.

PAUND BAŞINA FİYATI İKİLİK SAYIYA ÇEVİRME: Toplam fiyatı elde etmek için ağırlık ve paund başına fiyatı çarpmadan önce her ikisini ikilik sayıya çevirmek gerekir. Paund başına fiyat halen BCDCVT yordamında kullanılacak şekilde DE içinde olduğundan önce o çevrilir. BCDCVT alt yordamı çevirmeyi yapar ve sonucu 16 bitlik ikilik olarak HL içinde saklar. Bu sonuç SAVBP olarak adlandırılmış bir bellek tamponuna taşınır.

AĞIRLIĞI İKİLİK SAYIYA ÇEVİRME MODÜLÜ: BCDCVT'yi ağırlığı ikilik sayıya çevirmek için kullanmadan önce BUFFR1 içindeki paketlenmemiş BCD sayılar paketlenmeli ve D ve E içine taşınmalıdır. Bu bir seri basit taşıma ve sola kayma işlemi ile gerçekleştirilir. BCD rakamları BUFFR1 içine depolanırken üst 4 bitin hepsinin 0 yapılması gerçeği yapılan işi basitleştirir.

Paketleme yapıldıktan ve sonuç D ve E içine depolandıktan sonra BCDVT ağırlığı H ve L içinde 16 bitlik ikilik sayı eşdeğerine çevirmek üzere çağrılır. İkilik ağırlık XCHG deyimi ile DE'ye taşınır.

İKİLİK AĞIRLIK İLE İKİLİK PAUND BAŞINA FİYATIN ÇARPILMASI: İkilik paund başına fiyat SAVBP tamponundan B ve C kaydedicilerine taşınır. Daha sonra çarpmayı yapmak üzere MULTO altyordamı çağrılır. MULTO, B ve C içindeki 16 bitlik ikilik sayı ile D ve E içindeki 16 bitlik ikilik sayıyı çarpıp DEHL içinde 32 bitlik ikilik sonucu verir. Yordam tamsayı çarpması yapar, bir başka deyişle bütün sayılar pozitif tamsayı olarak ele alınırlar. 10.00 paundluk bir ağırlık onluk 1000 ya da $3E8_{16}$ olarak ve 5.56 dolarlık paund başına fiyat onluk 556 ve $22C_{16}$ olarak temsil edilir. Bölüm 7'de bahsedildiği gibi, programcı sonuçtaki ondalık noktanın doğru yerini belirlemelidir.

Maksimum ağırlık 10.00 paund olduğundan ve dört rakam ile gösterilebilecek maksimum toplam fiyat 99.99$ olduğundan, 10.00 paundluk bir ağırlık için maksimum paund başına fiyat 9.99 $ olur. Ağırlık ve paund başına fiyatı tamsayı gibi düşünmek çarpma sonucunu maksimum 999.000 verir. Bu rakam temsil edilmek için 20 tane ikilik bit gerektirir, bu durumda ikilik çarpmanın sonucu D kaydedicisi içindeki üst bayt tamamen 0 olur. Ağırlık ve paund başına fiyatın ikilik çarpmaları sonucu E, H ve L kaydedicileri içindeki 24 bittir.

TOPLAM FİYATI BCD'YE ÇEVİR: İkilik toplam fiyat gösterebilmek için BCD'ye çevrilmelidir. BINCVT alt yordamı E, H ve L içindeki 24 bitlik ikilik sayıyı H, L, B ve C içinde paketlenmiş sekiz BCD rakam şekline çevirir. 99.99$ maksimum fiyat bu kaydediciler içinde 00999900 haline dönüşür. Ondalık nokta açıktır ki L kaydedicisi içeriği ile B kaydedicisi içeriği arasındadır. Bu durumda, iki en küçük değerlikli BCD karakter 0 olur ve düşürülebilirler. Fakat bu her zaman olan bir durum değildir ve bu rakamlar bir üst BCD rakama yuvarlanmalıdır. Üretilen eldeler daha yüksek basamaklara taşınmalıdır.

HAFIZA ADRESİ	İÇERİK	İSİM
20A0H	AĞIRLIĞIN 1. RAKAMI	
20A1H	AĞIRLIĞIN 2. RAKAMI	BUFFR 4
20A2H	AĞIRLIĞIN 3. RAKAMI	
20A3H	AĞIRLIĞIN 4. RAKAMI	
20A4H	"L"	BUFFR 4
20A5H	"B"	
20A6H	FİYATI 1. RAKAMI	
20A7H	FİYATI 1. RAKAMI	BUFFR 4
20A8H	FİYATI 1. RAKAMI	DAHA SONRA FİYAT İÇİN KULLANILIYOR
20A9H	FİYATI 1. RAKAMI	
20AAH	"S"	BUFFR 4
20ABH	"P"	DAHA SONRA PR. İÇİN KULLANILIYOR

KAYDEDİCİLER	İÇERİK	
A	OFH	FİYAT/LB'Yİ SONLANDIRAN TUŞ
BC	XXXX	KOLAYCA TAHMİN EDİLEMEZ
DE	FİYAT/LB	4 PAKETLİ BCD RAKAM OLARAK
HL	KYBUF	

Şekil 12.7 Mini RAM ve Kaydedici Haritası.

TOPLAM FİYATI EN YAKIN SENTE YUVARLA VE GÖSTERGE İÇİN HAZIRLA

Yuvarlama, C içindeki endüşük iki BCD karakter ile akümülatör içindeki 49_{16} sayısının karşılaştırılması ile yapılır. Eğer C'nin içeriği 49_{16} 'dan daha büyük ise elde biti 1 yapılır. Üretilen herhangi bir elde B içindeki bir sonraki basamağa eklenir. Sonucu BCD formatında elde etmek için DAA işlemi yapılır. Sonuç genişletme altyordamı için hazırlanmak üzere E kaydedicisine alınır. *B* üzerinde yapılan DAA işleminden üretilen bir elde *L* kaydedicisine eklenir. Sonucu BCD formatında elde etmek için DAA işlemi uygulanır. Sonuç genişletme alt yordamı için D'ye taşınır.

GENİŞLETME, GÖSTERME, VE GECİKME: EXPND, *D* ve *E* içindeki paketlenmiş BCD sayıları BUFFR3 içinde paketlenmemiş dört BCD karakter şekline çevirir. Daha sonra toplam fiyatın bu dört rakamını dört LED göstergeye göndermek için DISPLY çağrılır.

BUFFR4 "*Pr.*" için karakterlerle yüklenir. Bu karakterleri iki rakam LED göstergeye göndermek için DISPLY yeniden çağırılır.

İç içe olan bir gecikme döngüsü sessiz-terazi modülüne geri atlamadan önce gerekli 3 saniyelik gecikmeyi elde etmek için kullanılır. DE içine yüklenen sayı LNGLP döngüsünün kaç defa döneceğini belirler, ve B'ye yüklenen sayıda SHRTLP döngüsünün kaç defa döneceğini belirler. DE'ye 0000H yüklenmesi LNGLP'nın 65536 kere dönmesini sağlar çünkü DCX 0 kontrolu yapılmadan önce DE'yi FFFFH'ye azaltır. LNGLP'nin her dönüşünde B kaydedicisi 08_{16} ile yüklenir ve SHRTLP ile 0'a doğru azaltılır. SHRTLP'den çıkılınca D ve E'nin 0'a ulaşıp ulaşmadıkları kontrol edilir. DCX deyimi herhangi bir bayrağı değiştirmez. Bu yüzden 16 bitlik bir kaydediciyi 0 ile karşılaştırmanın bir yolu 1 baytı akümülatöre taşımak ve diğer bayt ile VEYA işlemine sokmaktır. Eğer her iki bayt 0 ise sıfır bayrağı bir olur. Böylece çevirici dil programı üzerinden ilk geçişi tamamladık. Bir sonraki geçişte dört temel alt yordam DISPLY, BCDCVT, MULTO, ve BINCVT detaylı şekilde tartışılacaktır.

TEMEL ALTYORDAMLAR

DISPLY: DISPLY alt yordamının fonksiyonları bir tampon içinde depolanmış BCD veya onaltılık karakterleri yedi-parçalı koda çevirmek ve yedi parçalı kodu 8279 'un içindeki gösterge tamponuna göndermektir.

Kaydedicileri sakladıktan ve karakter sayıcısını ayarladıktan sonra, program dört rakamlı veya iki rakamlı gösterge istendiğini kontrol etmek için akümülatörü sağa eldeye doğru döndürür. Dört rakamlı alan için 8279'un kontrol port adresi 1900 H'ye 90_{16} kontrol kelimesi gönderilir. Eğer iki rakamlı alan isteniyorsa kontrol port adresi 1900H'ye 94_{16} kontrol kelimesi gönderilir. Bu kontrol kelimesi 8279'un iç gösterge RAM'ı için bir göstericiyi ayarlar.

Bir sonraki adım, ana bellek tamponundaki ilk BCD veya onaltılık karakteri yedi-parçalı koda çevirmektir. Bu çevirme için kullanılan tekniği öğrenmek önemlidir. Birçok durumda bu metod, Bölüm 11'de klavye kodu için kullanılan karşılaştırma metodundan daha hızlı ve daha verimlidir.

Programın sonunda gösterildiği gibi DSPTB adında bir tablo hazırlanır. Gösterildiği gibi istenen çıkış kodları sıraya konulur. Tablonun en üstüne 00 için istenen kod yerleştirilir, ve en yüksek kod için istenen karakter tablonun sonuna yerleştirilir.

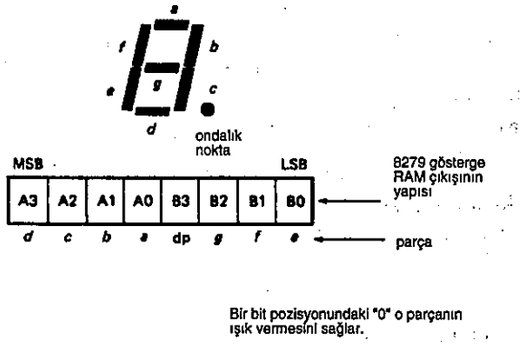
Bir çevirme yapmak için tablonun en üst adresi gösterici kaydedicisi *HL*'ye yüklenir. Daha sonra BCD veya onaltılık karakter göstericiyle toplanır. Toplamanın sonucu *HL*'ye yüklenir ve bu DSPTB içindeki istenen yedi-parçalı kodun yerini gösterir. Daha sonra MOV L,M yedi-parçalı kodu DSPTB'den geçici olarak depolamak üzere *L*'ye taşır.

DSPTB'de gösterilen yedi-parçalı kodun Tablo 3.1'de gösterilen standart yedi-parçalı kod olmadığına dikkat ediniz. Şekil 12.8'de SDK-85 için gerekli yedi-parçalı formatı gösterilmektedir. DSPTB içindeki değerler bu formatı kullanan her karakter için gerekli değerin tamlayanıdır.

Ağırlık göstergesi, paund başına fiyat, toplam fiyat ve birimlerin hepsi gösterme alanında soldan ikinci rakamdan sonra bir ondalık nokta gerektirir. Bu yedi-parçalı kod içinde ondalık nokta pozisyonuna VEYA lama ile bir 1 koymakla başarılır.

Karakter tamamlandıktan sonra karakter 8279'un yazma adresi 1800 *H*'ye gönderilir. Dört karakterin hepsi tampondan okununcaya, çevrilip 8279'a gönderilinceye kadar program DSP3'e geri döner. Birim göstergesi için yalnızca iki karakter gönderilmesi gereklidir. Fakat devamlı olarak dört karakter göndermekte zarar yoktur ve program basitleştirilir.

BCDCVT: Bu yordam *D* ve *E* içindeki dört-rakamlı BCD sayıyı *HL* içinde 16-bitlik ikilik eşdeğerine çevirir. Metod çok basittir. Her onluk yerdeki sayının ikilik eşdeğeri o yerdeki BCD rakam ile çarpılır ve sonuç *HL* içindeki toplama eklenir. Çarpma, ardışık toplamalarla yapılır. Bir örnek metodu açıklamaya yardım edecektir.



Şekil 12.8 SDK-85 Kartındaki 8279 çıkışı için Parça Formatı.

Bir BCD ya da onluk sayı olan 3421'ın ikiliğe çevrileceğini varsayalım. Bu sayı 3x1000 artı 4x100 artı 2x10 artı 1x1 olarak düşünülebilir. 1000_{10} sayısı $3E8_{16}$ sayısına, 100_{10} sayısı 64_{16} sayısına, 10_{10} sayısı $0A_{16}$ sayısına, ve 01_{10} sayısı 01_{16} sayısına eşittir. 3421_{10} sayısının ikilik eşdeğerini elde etmek için yapılması gereken bütün iş $3E8_{16}$ sayısını üç defa artı 64_{16} sayısını dört defa artı $0A_{16}$ sayısını iki defa artı 01_{16} sayısını bir defa toplamaktır. Bu çarpmanın ardışık toplama şeklini kullanır.

Her BCD rakamın gerçek çevrilmesi, iki kısa alt yordamdan biri tarafından yapılır. MSBCD üst dört bitte bulunan karakter üzerinde işlem yaparken LSBCD alt dört bitte bulunan BCD karakteri çevirir.

Bu şekilde bir yordamı takip etmekte zorluk çekiyorsanız, 3421 gibi belirli bir BCD sayı alıp yordamı adım adım bu sayı ile takip ediniz. Aşağıda en büyük değerlikli iki basamak için adımlar verilmiştir.

HL sıfırlanır ve 1000_{10} sayısının ikilik eşdeğeri *BC*'ye yüklenir. En büyük değerlikli iki BCD rakam *A*'ya taşınır ve MSBCD çağırılır. MSBCD, BC içindeki $03E8_{16}$ sayısını *HL* ile en büyük değerlikli BCD karakter, bu örnekte 3, sıfıra azalıncaya kadar defalarca toplar ve geri döner. *BC*, 0064_{16} (100_{10}) ile yüklenir ve LSBCD çağırılır. LSBCD, BC içindeki 0064_{16} sayısı *HL* ile en küçük değerlikli dört bit, bu örnekte 4, sıfıra azalıncaya kadar toplanır ve program geri döner. E içindeki diğer iki BCD karakterde aynı şekilde $0A_{16}$ ve 01_{16} kullanarak çevrilirler. Yordam daha fazla ya da daha az sayıda BCD basamakları çevirebilmek için kolaylıkla değiştirilebilir.

MULTO: MULTO, *BC* içindeki 16 bitlik ikilik sayı ile *DE* içindeki 16 bitlik ikilik sayıyı çarpar ve 32 bitlik ikilik sonucu *DE* ve *HL* içinde saklar. Bölüm 7'de bahsedildiği gibi ikilik çarpma yapmanın birçok yolu vardır. Yukarıda anlatılan BCDCVT yordamı ardışık toplamayı kullanıyordu. Şekil 12.9, Bölüm 7'de gösterilen sağa kaydır ve ekle yöntemini kullanan 8 bit x 8 bit yordamını göstermektedir. 16 bitx16bit çarpma, kalemle çarpmada sık sık yapıldığı gibi 8080A/8085A'da da sola kaydır ve ekle metodu ile daha kolay yapılır. Şekil 12.10 daha az bit kullanarak gösterilen örnek MULTO yordamının nasıl çalıştığını göstermekte yardımcı olacaktır.

Şekil 12.10'dan görüldüğü gibi kısmi çarpım sıfıra eşitlenir ve 1 bit sola kaydırılır. Çarpanın en büyük değerlikli biti kontrol edilir. Eğer bu bit 1 ise çarpılan, kısmi çarpım ile toplanır ve sonuç sola kaydırılır. Eğer en büyük değerlikli bit 0 olsaydı kısmi çarpım yalnızca sola bir bit kaydırılacaktı. Çarpanın her bir biti gözden geçirilir. Eğer 1 ise, çarpılan kaydırılmış kısmi çarpıma eklenir, 0 ise kısmi çarpıma hiçbir şey eklenmez. Şekil 12.10'daki metodun çalıştığı hemen açıkca görülemezse de bu metod çalışmaktadır.


```
;MULPY:      D İÇİNDEKİ 8 BİTLİK İKİLİK SAYI İLE C İÇİNDEKİ 8 BİTLİK İKİLİK
;            SAYIYI ÇARPAR VE 16 BİTLİK İKİLİK SONUCU B VE C İÇİNDE SAKLAR
;
;GİRİŞLER:   D - 8 BİT İKİLİK ÇARPILAN
;            C - 8 BİT İKİLİK ÇARPAN
;ÇIKIŞLAR    BC - 16 BİT ÇARPIM
;BOZULANLAR  A, B, C, E, E/FS

MULPLY:  MVI   B, 0     ;SONUCUN EN BÜYÜK DEĞERLİKLİ BAYTINI SIFIRLA
         MVI   E, 9     ;BİT SAYICIYI 9 YAP
MULP1:   MOV   A, C     ;ÇARPANI AKÜMÜLATÖRE TAŞI
         RAR            ;ÇARPANIN EN KÜÇÜK DEĞERLİKLİ BİTİNİ ELDEYE TAŞI
         MOV   C, A     ;KAYDIRILMIŞ ÇARPANI C'YE TAŞI
         DCR   E        ;BİT SAYICIYI AZALT
         RZ             ;SIFIR İSE DÖN
         MOV   A, B     ;SONUCUN EN BÜYÜK DEĞERLİKLİ BAYTINI
                        ;AKÜMÜLATÖRE TAŞI
         JNC   MULP2    ;RAR'DAN ELDE YOKSA, SONUCUN B YAZMACI
         ADD   D        ;İÇİNDEKİ YÜKSEK BAYTINA ÇARPILANI TOPLA
                        ;YÜKSEK BAYTI SAĞA KAYDIR
                        ;SONUCUN YÜKSEK BAYTINI YENİDEN YÜKLEVE
MULP2    RAR            ;BİR SONRAKİ BİTİ KONTROLA GİT
         MOV   B, A     ;BU RAR'DAN OLAN ELDE MOV A, C KOMU
                        ;TUN'DAN
         JMP   MULP1    ;SONRA RAR TARAFINDAN C'NİN EN BÜYÜK DEĞERLİKLİ
                        ;BİTİNE KAYDIRILIR BÖYLECE ÇARPIMIN ALÇAK
                        ;BAYTI C'NİN SOL UCUNDAN DIŞARI DOĞRU
                        ;KAYDIRILIR
```

Şekil 12.9 Sağa kaydır ve topla algoritması kullanarak 8 bitx8 bit ikilik çarpma yapan çevirici dil yordamı.

```
    1101      13D ÇARPILAN
  X 1011      11D ÇARAN
    0000      KISMI ÇARPIM KAYDEDİCİSİNİ SIFIRLA
   00000      SOLA KAY
    1101      ÇARPANIN EN BÜYÜK DEĞERLİKLİ BİTİ 1 OLDUĞUNDAN
   01101      ÇARPILANI TOPLA
  011010      SOLA KAY
    0000      ÇARPANIN BİR SONRAKİ RAKAMI SIFIR
  011010      OLDUĞUNDAN SIFIR EKLE
 011010       SOLA KAY
    1101      ÇARPANIN BİR SONRAKİ RAKAMI 1 OLDUĞUNDAN
 1000001      ÇARPILANI TOPLA
10000010      SOLA KAY
    1101      ÇARPANIN EN KÜÇÜK DEĞERLİKLİ BİTİ BİR OLDUĞUNDAN
10001111      ÇARPILANI TOPLA
  8  FH = 143D
```

Şekil 12.10 MULTO Algoritmasının kısa örneği.

MULTO alt yordamı 16 bitlik kaydedicilerin azlığı nedeniyle daha karmaşık gözükmektedir. Çarpılan *B* ve *C* içine alınır ve çarpanda *DE* içine yüklenir. Kısmi çarpım *HL* içinde başlar ve daha sonra çarpan *DE*'den dışarı doğru kayarken, kısmi çarpım aynı zamanda *DE* içine genişler. 32-bitlik bir sonuç *DE* ve *HL*'nin tamamını doldurur.

MULTO üzerinde çalışırken iki önemli noktayı hatırlamak gerekir. Birincisi, DAD H komutu *HL*'nin içeriğini *HL* ile toplar. Bu *HL*'nin içeriğini 2 ile çarpar ve bu işlem *HL*'nin her bitini 1 bit sola kaydırmakla aynıdır. *H*'nin en büyük değerlikli biti eldenin içine kayar. İkincisi, XCHG, JUMP, ve bütün MOV komutları bayrakları etkilemez. Bu yüzden XCHG komutundan sonra kullanılan şartlı atlama komutu daha önceki komutlarla değiştirilen bayrakları kullanır. Bu ipuçlarını, MULTO içindeki açıklamaları ve Şekil 12.10 'daki örneği çalışarak yordamı anlıyabilirsiniz. Biraz vakit alırsa umutsuzluğa kapıl-mayınız. Bir program ya da yordam üzerinde çalışmak her zaman uzun süre alabilir.

BINCVT: BINCVT, *E*, *H* ve *L* kaydedicileri içindeki 24 bitlik ikilik sayıyı, *H*,*L*, *B* ve *C* içinde paketlenmiş sekiz BCD rakam haline çevirir. Belkide terazi programı içindeki yordamların yöntemi en az açık olanıdır, fakat asıl işlemler oldukça basittir.

İkilik bir sayı içinde her bit pozisyonu 2'nin kuvvetlerini temsil etmektedir. Bir 8-bitlik ikilik sayı

$$b7 \times 2^7 + b6 \times 2^6 + b5 \times 2^5 + b4 \times 2^4 + b3 \times 2^3 + b2 \times 2^2 + b1 \times 2^1 + b0$$

olarak yazılabilir ve yeniden düzenlenirse

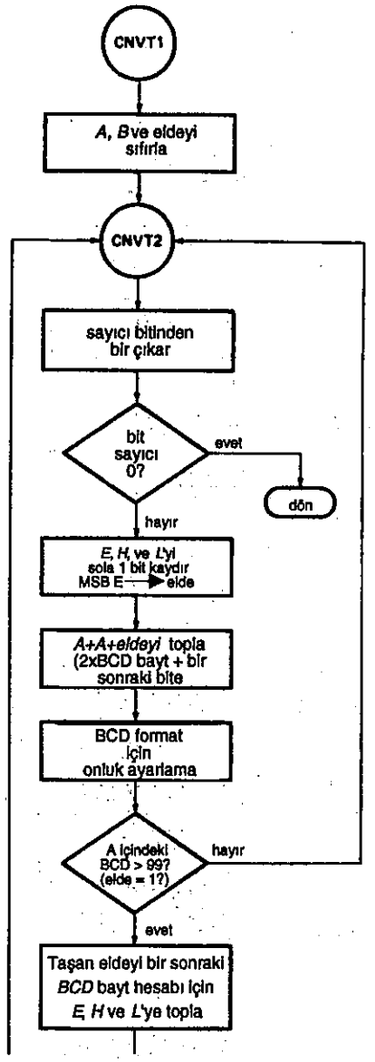
$$\text{İkilik Sayı} = ((((((2b7 + b6)\,2 + b5)2 + b4)2 + b3)2 + b2)2 + b1)2 + b0$$

olarak ifade edilebilir. Burada b7 - b1 ikilik bitlerdir. İçiçe parantezler içindeki her işlem BCD olarak DAA komutunun yardımı ile yerine getirilirse sonuç BCD şeklinde olacaktır. Bu BINCVT'nin bir BCD eşdeğeri üretmesi yöntemidir.

BINCVT'nin CNVT1'e kadar olan kısmı bit sayısını onluk 25 yapar ve iki BCD rakamı B içinde üretmek için CNVT1 çağrılır. Bu iki BCD rakam saklanır, ve bit sayısı yeniden 25 yapılır, ve CNVT1 bir sonraki iki BCD rakamı üretmek için çağrılır. Bütün yazmaçlar kullanıldığından dört BCD rakam saklanmak üzere yığının üstüne itilir. Dört en büyük değerlikli BCD basamak bit sayıcısını yükleme ve CNVT1'i çağırma döngülerini iki defa yerine getirmekle üretilir.

Bu dört BCD karakter *H* ve *L* içine taşınırlar, ve dört en az küçük değerlikli BCD rakam yığından *B* ve *C* içine alınırlar.

Bu çevirmenin kalbi CNVT1 ve CNVT2 yordamlarıdır. Şekil 12.11 akış diyagramını göstermektedir. Elde ve *A* ile *B* kaydedicileri sıfırlandıktan son-



Şekil 12.11 CNVT1 alt yordamı için akış diyagramı

ra *E*, *H* ve *L* içindeki 3 baytlık ikilik sayı 1 bit sola kaydırılır. E'nin en büyük değerlikli biti elde içine taşınır. Kayma aşağıda anlatıldığı gibi yapılır. DAD *H*, *HL*'nin içeriğini 1 bit sola kaydırır. H'nin en büyük değerlikli biti eldeye kayar. *E* kaydedicisi akümülatöre taşınır. RAL eldenin içeriğini *A*'nın en küçük değerlikli bitine kaydırır ve *A*'nın içeriğini 1 bit sola kaydırır. *A*'nın en büyük değerlikli biti sola eldenin içine kaydırılır. Kaymadan sonra, kaydırılmış sonuç E içinde saklanır.

Üretilmekte olan 2 BCD karakteri *B* kaydedicisi içinde tutulur. Üzerlerinde çalışılmak üzere akümülatöre taşınırlar. ADC *A* komutu değerlerini iki katına çıkarır ve *E*'den dışarı çıkıp elde içinde tutulan biti toplar. Bunu içiçe parantezli eşitlikteki yöntem takip eder. *A* içindeki baytı BCD formatında tutmak için DAA işlemi yapılır ve sonuçta elde edilen iki BCD karakteri yeniden *B* içinde depolanır.

DAA tarafından bir elde üretilmezse, program CNVT2'ye geri döner ve çevirme işlemine devam eder. Eğer DAA, *A* içindeki bayt onluk 99'dan büyük olduğu için bir elde üretirse, bu taşan elde bir sonraki iki daha büyük değerlikli BCD rakamı üretmekte kullanılmak üzere *L* kaydedicisi ile toplanmalıdır. Bu eldeyi *L* ile toplamanın etkisi *H* ve *E* üzerinde yayılır. Daha sonra program çevirmeye devam etmek için CNVT2'ye geri döner. CNVT1 üzerinde her geçişte bir sonraki daha büyük değerlikli iki BCD rakam B kaydedicisi içinde üretilir. CNVT1'in her çalışmasından sonra (CNVT2'nin 24 defa çalışması), EHL içinde orijinal ikilik sayı eksi üretilen iki BCD rakamın değerine eşit olan ikilik bir sayı depolanır. *D* içindeki bit sayıcısı DEC D komutunun döngü içersindeki pozisyonu sebebiyle başlangıçta 25_{10} veya 19_{16}, sayısına eşitlenmelidir. Eğer bu yordamın nasıl çalıştığının detaylarını gerçekten görmek istiyorsanız *E*, *H*, ve *L* içine bazı değerler yerleştirip yordam boyunca işlemleri takip ediniz. Büyük bir grafik kâğıdı bütün bitleri düzgün olarak takip etmenize yardım edebilir.

ÖZET

Bu bölümün önceki kısımları mikroişlemci kontrollu, ya da akıllı, bir terazinin donanım ve yazılımını göstermiştir. Terazi burada anlatılabilecek kadar basit olarak tasarlanmıştır. Fakat okuma sırasında muhtemel ilerlemeler ve uygulamalar aklınıza gelmiş olabilir.

Böyle basit bir terazinin sayısal kısmının tamamı 8048 veya 8748 gibi tekyongalı mikrobilgisayarlarla tasarlanabilir. Fakat, 8080A/8085A çevirici dili hakkında size daha çok deneyim kazandırmak için 8085 A kullanılmıştır.

Ağırlık, paund başına fiyat ve toplam fiyatı aynı anda gösterebilmek için daha fazla yedi-parçalı gösterge ilave edilebilir. Etiketleri paketlerin üzerine yapıştırabilmek için ağırlık, paund başına fiyat ve toplam fiyat bir yazıcıya gönderilebilir. Toplam fiyat bir elektronik yazar kasaya gönderilebilir ve otomatikman girilebilir.

Darayı çıkarmak için bir çıkarma yordamı ilave edilebilir. Terazi paund başına fiyat yerine paund başına birim çevirme faktörü girilerek civata, vida, veya para saymakta kullanılabilir. Bir diğer uygulama da mektup terazisidir. Her mektup fiyat oranı için başvuru tabloları ilave edilerek terazi gerekli posta ücretini hesaplamakta kullanılabilir. Akıllı teraziyi bir posta ücreti ölçere bağlamakla uygun pulların otomatik olarak basılması sağlanabilir.

İhtimaller hemen hemen sonsuzdur ve bu da neden mikroişlemcilerin bizim hayatımızda önemli etkileri olduğunu göstermektedir. Bu bölümde şimdi 6800 mikroişlemcisi üzerine kurulu bir EPROM programlayıcının donanım ve yazılımını tartışacağız.

MC6800 TABANLI BİR EPROM PROGRAMLAYICI

Mikroişlemcilerle çalışırken çok faydalı olan bir alet, bir EPROM programlayıcıdır. Geliştirilen programlar bir EPROM içine yazılır ve prototip alet üzerinde denenebilir. Eğer EPROM içine depolanan program bu alet içinde çalışmazsa, *morötesi ışınla silinebilir*. Daha sonra EPROM yeniden programlanabilir.

Bu bölümde *2716 ya da eşdeğer MOS EPROM*ları için bir programlayıcıyı tartışacağız. Bir atlama telinin yerini değiştirmek birimin aynı zamanda 2732 *EPROMları programlamasını da sağlar.*

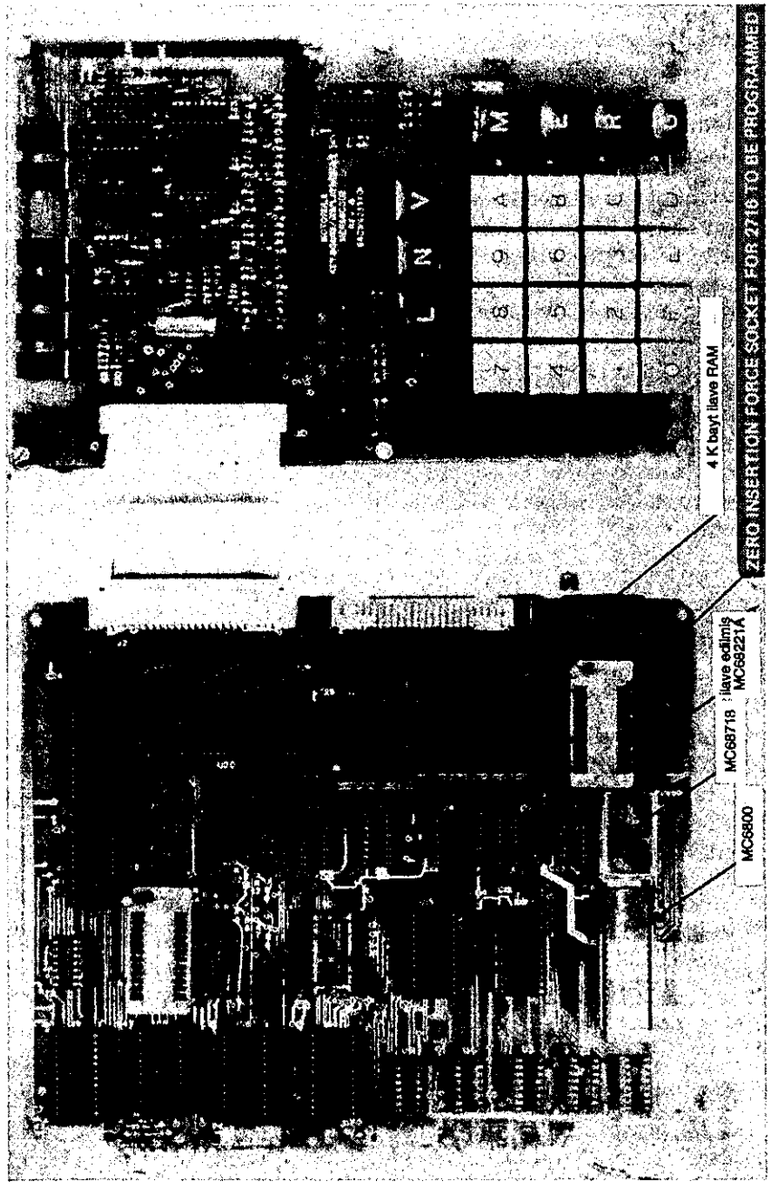
DONANIM TANIMLARI

+ 25 ve +5 V güç kaynakları dışında programlayıcının tamamı iki MEK6800D2 PC kartı üzerine kurulmuştur. Şekil 12.12 programlayıcı devresinin tamamı ile iki PC kartının fotografını göstermektedir. Önce MEK6800D2'nin donanımı anlatılacak ve daha sonra programlayıcı için eklenen devreler açıklanacaktır.

TEMEL MEK6800D2 KARTI: Şekil 10.5 MEK 6800D2 mikroişlemci kartının şemasını ve Şekil 11.4 ise klavye-gösterge kartının şemasını göstermektedir. Gerekli durumlarda bu şekillere bakınız.

Temel CPU kartı bir MC6800 CPU, bir MC6871 saat üreteci, iki MC6821 PIA, bir MC6850 ACIA, iki MC6810 içinde 256 bayt RAM, ve bir MC6830 içinde 1 kilobayt ROM ile birlikte gelmektedir. ROM Motorola JBUG monitor programı ile programlanmıştır.

Üç ilave MC6810 RAM, iki ilave ROM, ve adres hattı ve veri hattı tamponları için ilave yuva boşlukları mevcuttur. Bu tamponlar kartın büyük bir bellek ile bağlanması durumunda gereklidir. Başka entegre devrelerin ilavesi için tel sarma bölgesi mevcuttur.



Şekil 12.12 MEK6800D2 Mikroişlemci kartı ve klavye-Gösterge Kartının Fotografı (Motorola)

MC6810 RAM lardan biri JBUG monitor programı tarafından karalama tahtası olarak kullanılır, ve MC6821 PIA lardan biri klavye-gösterge kartını bağlamak için kullanılır. Böylece, kullanıcıya sunulan bellek ve I/O fonksiyonları: 0000H ile 007FH adresleri arasındaki 128 baytlık RAM; bir MC6821 içindeki 8004H ve 8006H adreslerindeki iki paralel I/O portu; MC6850 ACIA içinde 8008H ve 8009H adreslerindeki bir seri I/O portu; ve E000H ile E3FFH adresleri arasında yerleştirilmiş 1 kilobaytlık monitor programıdır.

Klavye-gösterge kartı, klavye-gösterge donanımına ilave olarak Kansas City standart formatındaki verileri manyetik teybe yazan ve teypten okuyan devreleri de içermektedir. Bu özellik uzun programları saklamak için çok kullanışlıdır.

2716 EPROM PROGRAMLAYICI İÇİN DONANIM İLAVELERİ

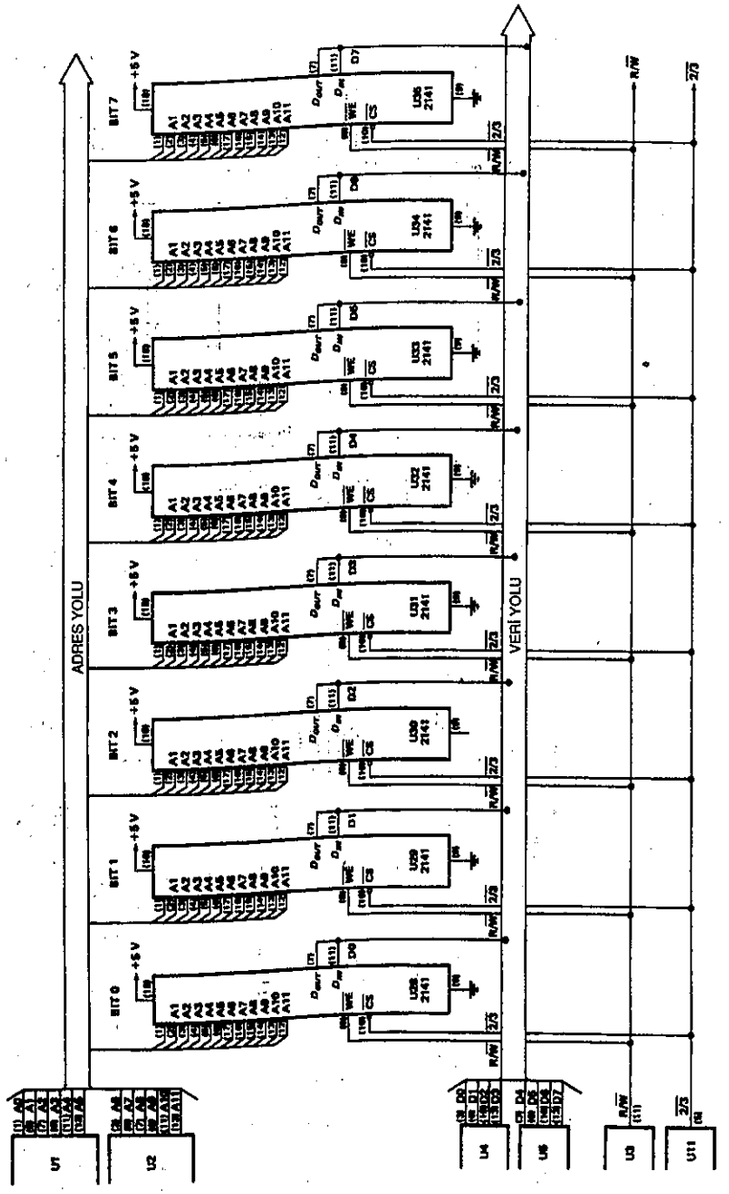
2716 için programlanacak verilerin başka bir 2716 veya RAM' dan gelmesini sağlayacak donanım temel CPU kartına eklenmiştir (Teypteki programlar önce RAM a yüklenir ve daha sonra EPROM' u programlamakta kullanılır).

24 bacaklı sıfır-sokma-kuvveti (ZIF) soket kopyalanacak 2716 için *U12* entegre devre pozisyonuna lehimlenmiştir, diğer bir sokette programlanacak 2716 için tel sarma bölgesine lehimlenmiştir. *ZIF* soketler yalnızca ufak bir manivela aşağı ittirildiği zaman entegre devre bacakları üzerine kenetlenirler. Böylece entegre devreler, bacakların bükülme tehlikesi olmadan sokulabilir ve çıkartılabilir. Şekil 10.5 bir 2716 için *U12* bağlantılarını göstermektedir. Gösterildiği gibi kopyalanacak 2716,6000 H adresinden başlamaktadır.

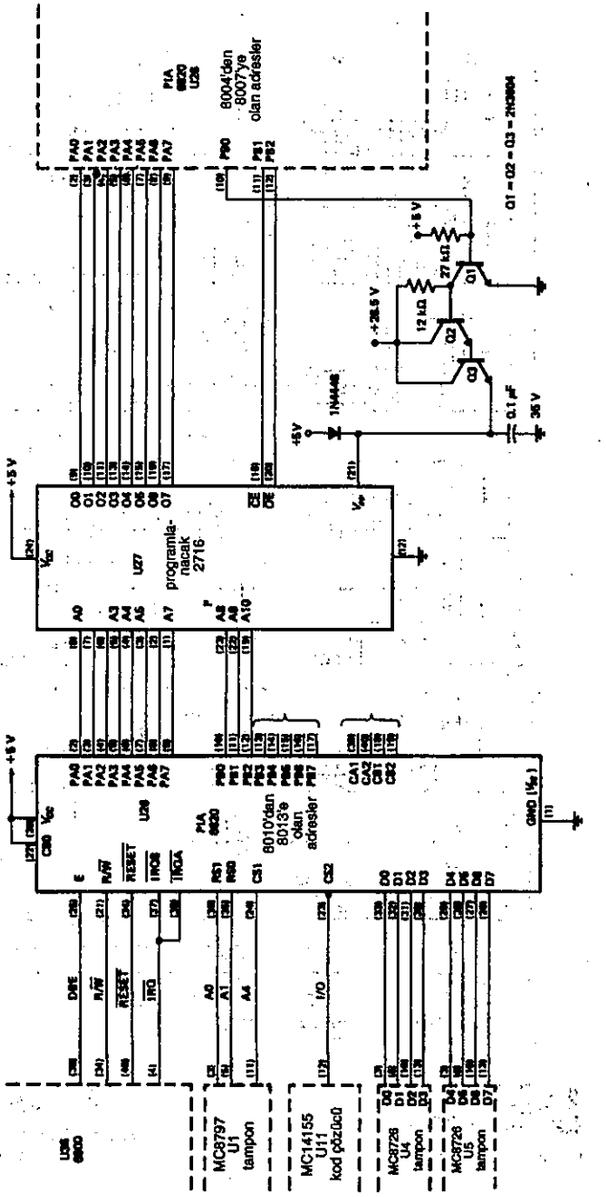
Tel-sarma bölgesine eklenen sekiz MM 5275N 4 K x 1 statik RAM lar 4 kilobaytlık RAM depolama bölgesi verir. Bu da bir 2716 veya 2732 programının tamamının RAM a bir defada yüklenmesini sağlar.

Şekil 12.13 ilave RAM şemasını göstermektedir. *U11* kod çözücüsünden gelen $\overline{2/3}$ hattı bu RAM bloğunu yetkilendirmek için kullanılır; böylece blok 2000H den 3FFFH 'ye kadar olan 8 kilobitlik bellek bölgesini kapsar. Bellek gerçekte 2000H ile 2FFFH arasını kullandığından, 3000H ile 3FFFH arası geri bükülmüştür. Diğer bir deyişle en öndeki rakamdaki 2 veya 3 kullanılan 4 kilobaytlık bölgeye ulaşır.

U1 den *U5* e kadar soket boşluklarına ilave belleği sürmek için tamponlar eklenmiştir. *U7* soket boşluğundaki sekiz girişli 7430 VE DEĞİL kapısı girişlerinden biri alçak olduğu durumda veri taşıt tamponlarını yetkilendirir. *U14* ve *U16* soketlerine eklenmiş iki MC6810 RAM, programcı için yazılan programı teypte tutup gerektiğinde çalıştırmak üzere RAM a yüklenmesini sağlar.



Şekil 12.13 MEK6800D2 mikro işlemci kartına ilave edilen 4KRAM blok devresi



Şekil 12.14 Programlanacak 2716 ve MEK6800D2 Mikroişlemci kartına ilave edilen *U26 PIA* için devre bağlantıları

Tel sarma bölgesine ilave edilmiş bir başka MC6821 PIA *(U27)* adreslerin programlanacak 2716'ya kilitlenmesi için kullanılır. Şekil 12.14 bu *PIA* ve programlanacak 2716'ya bağlanan *U20 PIA* için bağlantıları göstermektedir. İlave edilen *PIA* 8010H ile 8013H arasındaki adresleri kapsar.

Şekil 12.14 aynı zamanda programlama sırasında 2716 için gerekli olan $+25V_{pp}$ kaynağını açıp kapayan anahtarlama devresinide göstermektedir. 2716'nın programlanması ya da onaylanmasının dışında 25V açık olacağından, anahtar işlemci açıldığında açık olacak şekilde tasarlanmıştır. Bir reset'ten sonra MC6821'in bütün çıkışları bir iç direnç tarafından yükseğe çekilir. Anahtar girişindeki yüksek seviye, anahtarı açık duruma getirir. Anahtar açık iken +5 Volta bağlı diyot iletime geçer ve V_{pp} bacağına normal okuma işlemi için gerekli gerilimi sağlar. +25 Volta bağlı anahtar kapalı iken diyot terskutuplanır.

Dıştan kullanılan güç kaynakları ± % 5, +5 V ve V_{pp} anahtar için gerekli 26.5 V kaynaklardır. 2716 nın V_{pp} bacağında V_{pp}, +25V, ±1V olarak belirtilmiştir, fakat transistor anahtarda oluşacak gerilim düşümünden dolayı daha yüksek bir gerilim anahtar kaynağı olarak kullanılmalıdır. V_{pp} bacağındaki 0,1µF değerindeki kondansatör anahtarlama sırasında oluşabilecek ve 2716'yı bozabilecek aşırı taşmayı önlemekte yardımcı olur.

YAZILIM TANITIMI

JBUG MONİTÖR FONKSİYONLARI VE ÇALIŞMASI

JBUG monitör fonksiyonları EPROM programlayıcı yazılımının temel kısımları olduğundan öncelikle bu fonksiyonlar ve kullanımları tartışılacaktır.

Şekil 12.12'de onaltılık veriyi girmekte kullanılan 16 tuşa ilaveten sekiz harfli tuş olduğuna dikkat ediniz. Bu tuşlar aşağıda belirtilen monitör fonksiyonları için kullanılır.

M- Belleği kontrol etme ve değiştirme
E- Çalışmakta olan bir işlemden çıkış
R- *P, X, A, B, CC, S* kaydedicilerinin içeriklerini kontrol etme
G- Belirtilen programa git ve bu programı çalıştırmaya başla
P- Bellekten gelen verileri manyetik şerit üzerine del
L- Manyetik şeritten belleğe yükle
N- Komutları teker teker çalıştır
V- Duruş noktalarını yerleştir (kaldır).

Bir resetten sonra göstergenin dört-rakamlı adres kısmının en soldaki rakam pozisyonunda bilgi istemi karakteri olarak bir çizgi (-) görülür. Bilgi istemi karakteri size mikroişlemcinin bir komut girilmesini beklediğini belirtir. *M* fonksiyonunu ya da komutunu kullanmak için önce klavyeden istenen dört-rakamlı onaltılık adres girilir. Bu adres, göstergenin adres bölgesinde gösterilir. *M* tuşuna basıldığında göstergenin veri alanında (iki basamak) gösterilen adresteki veri gösterilir. Bu veri istenirse gerekli iki sayı girilerek değiştirilebilir.

Veri gösterge alanı yeni veriyi gösterir. Bundan sonra iki seçenek vardır. E tuşuna basmak bu komutu sona erdirir ve işlemcinin yeni bir komut için hazır olduğunu belirten bilgi istemi karakterine döner. *G* tuşuna basılırsa, adresler 1 arttırılarak adres bölgesinde gösterilir ve bu adreslerin içerikleride veri göstergesinde gösterilir. Bu adresteki veri değiştirilebilir veya aynen bırakılır. Daha sonra *E* tuşuna basmak bilgi istemi karakterine dönüşü, ya da *G*tuşuna basmak bir sonraki adresle içeriğinin gösterilmesini sağlar. Böylece *M* komutu bir programı RAM içine her seferinde 1 bayt olarak girmemizi sağlar.

İlgileneceğimiz bir sonraki komut, programları çalıştırmakta kullanılan *G* (git) fonksiyonudur. Bilgi istemi karakteri varken başlangıç adresi klavyeden girilir. *G* komutuna basınca program çalıştırılmaya başlar. MEK6800D2 yazılım gösterge taraması kullandığından, program çalışırken gösterge silinir.

Eğer program bir döngü içinde takılırsa ya da başka bir şekilde başarısız olursa, *E* tuşuna basmak programdan çıkışa ve bilgi istemi karakterine dönüşe sebep olur. *E* tuşu hemen hemen doğrudan işlemcinin $\overline{\text{NMI}}$ girişine bağlan-mıştır. Böylece *E* tuşuna basmak işlemciyi $\overline{\text{NMI}}$ servis yordamına yöneltir ve bu yordam kontrolu monitor programına geri verir. Bazı program hataları, $\overline{\text{NMI}}$ yordamının kullandığı bölgede dahil olmak üzere RAM'ın kullanılabilen bütün bölgelerine istenmeyen veriler yazabilir. Böyle bir durumda monitor kontroluna dönmenin tek yolu CPU kartı üzerindeki elle reset düğmesini kullanmaktır.

R, N, ve *V* komutları Bölüm 13'te anlatılmıştır. Geriye kalan komutlar *P* ve *L* komutlarıdır.

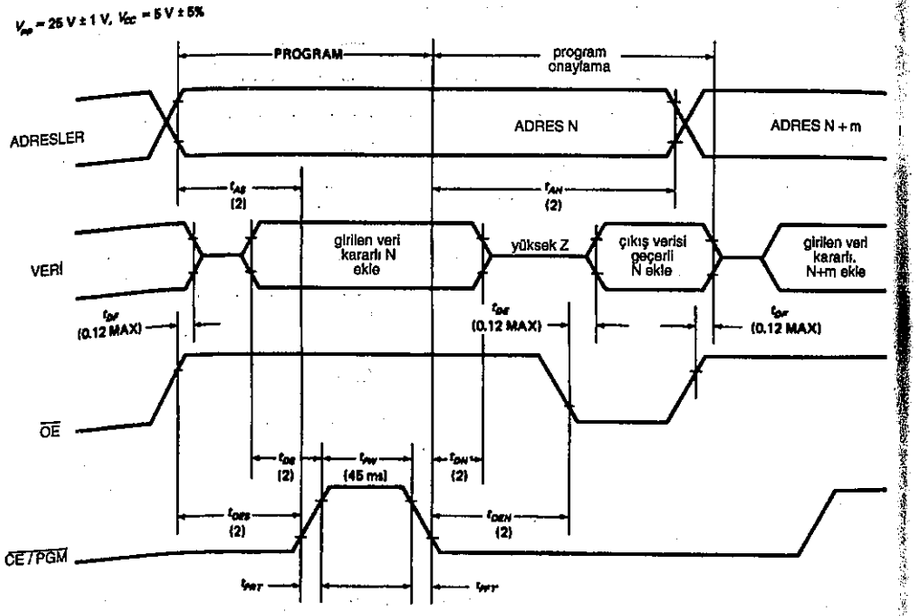
P komutu bir programı bellekten manyetik teyp üzerine yazmak için kullanılır. M komutu kullanılarak gönderilecek programın başlangıç adresi A002H ve A003H adreslerine yüklenir. Bitiş adresi ise A004H ve A005H adreslerine yüklenir, ve bilgi istemi karakterine dönmek için E tuşuna basılır. Klavye-gösterge modülü üzerindeki teyp çıkışı kaydedicinin mikrofon girişine bağlanır. Kaydedici, kaydetme modunda başlatılır, ve daha sonra P tuşuna basılır. *P* komutu çalışırken gösterge silinir. Teybe program yüklemenin bittiğini belirtmek için bilgi istemi karakteri gözükür.

L komutu manyetik teyp içindeki bir programı RAM a okumak için kullanılır. Öncelikle teyp kaydedicinin kulaklık çıkışı klavye gösterge kartının teyp girişine bağlanır. *Teyp kaydedici "playback"* modunda çalışmaya başladıktan sora *L* tuşuna basılır. Teypten okunan bir program kaydedildiği adresin aynısından başlayarak RAM a yüklenir. Başlangıç adresi monitor karalama bölgesindeki A002-3H adresine yazılır ve gerektiğinde kontrol edilebilir.

Özet olarak, EPROM programlayıcı, *M* fonksiyonunu programları RAM a girmek, *P* fonksiyonunu programları teybe depolamak, *L* fonksiyonunu programları teypten RAM a okumak, ve *G* fonksiyonunu programcının programlarını çalıştırmak için kullanır.

2716'nın PROGRAMLANMASI

2716 Mos EPROM Bölüm 3'te tartışılmıştı. Bu cihazın tam veri kağıdı Ek C'de gösterilmiştir. Başlangıçta ve silindikten sonra 2716'nın bütün bitleri 1 durumundadır ve 0'lar istenilen yerlere programlanırlar. Bir bit bölgesi 0 olarak programlandıktan sonra ancak bütün belleği silerek 1 durumuna değiştirilebilir.



Şekil 12.15 *(a)* Intel 2716 EPROM'u Programlama Dalga Şekilleri.

2716 ve 2758 Programlama Karakteristikleri: (1)

T_A = 25°C ±5°C, V_{CC} [2] = 5 V ±5%, V_{PP} [2,3] = 25 V ±1 V

DC Programlama Karakteristikleri

Sembol	Parametre	Min	Tip	Max	Birim	Test Şartları
I_{LI}	Giriş akımı (herhangi bir giriş için)			10	µA	V_{IN} = 5.25 V/0.45
I_{PP1}	V_{PP} kaynak akımı			5	mA	$\overline{CE}$/PGM = V_{IL}
I_{PP2}	Programlama darbesi sırasında V_{PP} kaynak akımı			30	mA	$\overline{CE}$/PGM = V_{IH}
I_{CC}	V_{CC} kaynak akımı			100	mA	
V_{IL}	Giriş alt seviyesi	−0.1		0.8	V	
V_{IH}	Giriş üst seviyesi	2.0		V_{CC} + 1	V	

AC Programlama Karakteristikleri

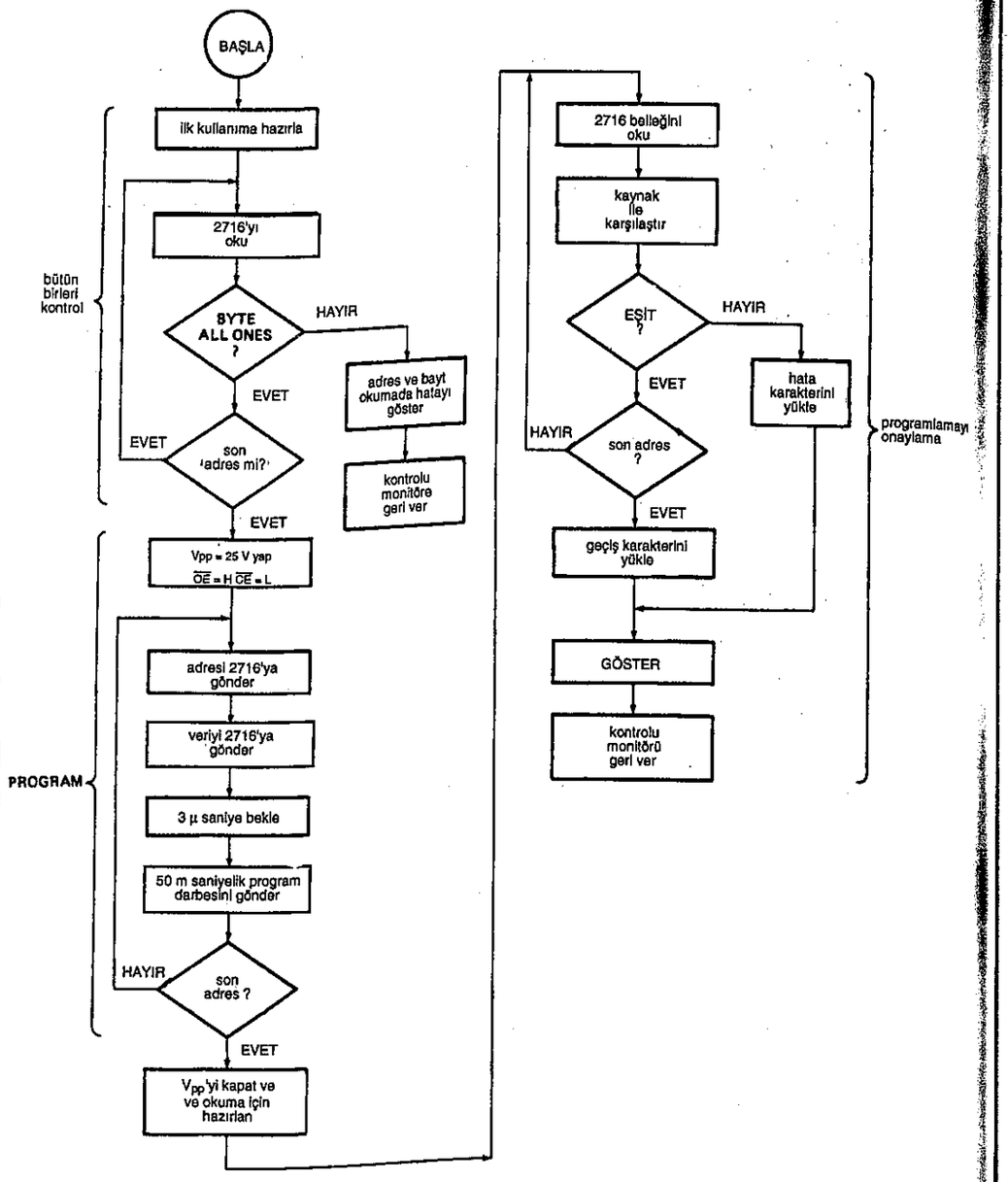
Sembol	Sembol	Min	Tip	Max	Birim	Test Şartları
t_{AS}	Adres duruş zamanı	2			µs	
t_{OES}	$\overline{OE}$ duruş zamanı	2			µs	
t_{DS}	Veri duruş zamanı	2			µs	
t_{AH}	Adres tutma zamanı	2			µs	
t_{OEH}	$\overline{OE}$ tutma zamanı	2			µs	
t_{DH}	Veri tutma zamanı	2			µs	
t_{DF}	Çıkış kayma gecikmesi için çıkışı yetkilendirme	0		120	ns	$\overline{CE}$/PGM = V_{IL}
t_{OE}	Çıkış gecikmesi için çıkışı yetkilendirme			120	ns	$\overline{CE}$/PGM = V_{IL}
t_{PW}	Program darbe genişliği	45	50	55	ms	
t_{PRT}	Program darbesi yükselme zamanı	5			ns	
t_{PFT}	Program darbesi düşüş zamanı	5			ns	

1. İntelin standart entegre garantisi yalnızca burada belirtilen özelliklere göre programlanan entegreler için geçerlidir.

2. V_{CC}, V_{PP} ile aynı zamanda ya da daha önce uygulanmalı ve V_{PP} ile aynı zamanda ya da daha sonra ayrılmalıdır. 2716/2758 entegrenin bozulmasını önlemek için V_{PP} değeri 25 ± 1V olan bir karta bağlanmamalı yada karttan ayrılmamalıdır.

3. V_{PP} bacağına programlama sırasında uygulanacak maksimum gerilim ± 26 volttur. V_{PP} kaynağını anahtarlarken 26 voltluk maksimum değeri aşmamak için önlemler alınmalıdır.

Şekil 12.15 (devamı) (b) Intel 2716 EPROM'un zamanlama karakteristikleri



Şekil 12.16 Bir 2716 EPROM Programlayıcı için Akış Diyagramı

Şekil 12.15 Intel 2716'nın programlama dalga şekillerini ve karakteristiklerini göstermektedir. Entegreyi programlamak için önce V_{pp} 25 volta yükseltilir. İstenen adres entegre uygulanır, ve $\overline{OE}$ yüksek duruma yükseltilir. Programlanacak bayt entegrenin 00 ile 07 arasındaki veri çıkışlarına uygulanır. Bu çıkışlarda en az 2μsaniyelik kararlı verilerden sonra $\overline{CE}$/PGM girişine 50 m saniyelik program darbesi uygulanır. Program darbesi alçak pozisyona döndükten sonra veri en az 2 μ saniye daha tutulmalıdır. Bir adres programlandıktan sonra $\overline{OE}$ alçak duruma getirilir ve programlamanın başarılı olduğunu görmek için o adres okunur. Eğer $\overline{OE}$ yüksek olarak bırakılırsa bir sonraki adres ve veri gönderilebilir ve $\overline{CE}$/PGM bacağına bir başka 50m saniyelik programlama darbesi uygulanabilir. Bir programlama oturumu sırasında cihazın bütün adresleri veya bir kısmı programlanabilir. Adresler herhangi bir sıra ile programlanabilir.

2716 PROGRAMLAYICI İÇİN AKIŞ DİYAGRAMI

Şekil 12.16, 2716'yı programlamak için akış diyagramını göstermektedir. Program üç ana kısımdan oluşur. İlk olarak programlanacak bütün adreslerin 1 olup olmadığı kontrol edilir. Bir başka deyişle bütün adresler silinmiş mi? Daha sonra istenen adresler programlanır. Son olarak 2716 'daki her adres okunur ve kaynak adreslerdeki verilerle karşılaştırılır. Eğer herhangi bir adres onaylanmazsa altı *F*'den oluşan hata işareti yüklenir ve gösterilir. Eğer bütün adresler onaylanırsa altı A yüklenir ve gösterilir.

BİR 2716 PROGRAMLAYICI İÇİN ÇEVİRİCİ DİL PROGRAMI

Şekil 12.17 bu akış diyagramanın değiştirilmiş bir MEK6800D2 üzerinde gerçekleştirilmesi için çevirici dil programını göstermektedir.

JBUG monitorun *M* komutunu kullanarak 2716 içinde programlanacak başlangıç adresi 0002H ve 0003H RAM adreslerine girilir. Bitiş adresi artı 1 0004H ve 0005H RAM adreslerine girilir. Veri kaynağının başlangıç adresi 0006H ve 0007H adreslerine girilir ve monitore geri dönmek için E tuşuna basılır. Başlangıç, bitiş ve kaynak adresleri kullanıcı tarafından girildiğinden program verisi işlemcinin 65 kilobaytlık adres bölgesinin her hangi bir yerinden gelebilir. Veri 2716'nın istenen herhangi bir bölgesine programlanabilir. Kopyalanacak 2716'nın başlangıç adresinin 6000H, ve 4 kilobaytlık RAM belleğinin başlangıç adresinin 2000H olduğunu hatırlayınız.

Programlayıcı programının başlangıç adresi 0010H klavyeden girilir ve *G* tuşuna basılır. Bu, programın bütün 1'ler programını kontrol etmesine ve 2716'yı onaylamasına sebep olur.

Şekil 12.17'deki çevirici programda *PIA* ları ilk kullanıma hazırlamak için çok miktarda dizilmiş adresleme kullanıldığına dikkat ediniz. Bütün *PIA* lar 8000H yakınlarındaki bellek adreslerine haritalandıklarından bu sayı dizin kaydedicisine taban adresi olarak yüklenir. Daha sonra herhangi bir *PIA* kontrol veya veri portuna STAA 10, X gibi iki baytlık bir dizinli deyimle ulaşabilir.

Üç *PIA* veri yön kaydedicisine onları çıkış olarak kullanmak için FFH kodu gönderilir. Daha sonra bağlantıyı veri yön kaydedicilerinden asıl port kaydedicilerine değiştirmek için, bütün PIA kontrol kaydedicilerine bir kontrol kelimesi gönderilir.

Bütün *PIA* lar ilk kullanıma hazırlandıktan sonra, program kullanıcının yüklediği 2716 başlangıç adresini WRKADD adındaki tamponda ya da çalışma adresinde saklar. Kullanıcının yüklediği kaynak adresi RDADD adlı tamponda ya da okuma adresinde saklanır. WRKADD ve RDADD adresleri her adres programlandıktan sonra arttırılır. Bu adresleri kullanıcının yüklediği diğer adreslerden ayırmak kullanıcının yüklediği adreslerin daha sonra kullanımlar için saklanmasını sağlar.

Daha sonra $\overline{OE}$ ve $\overline{CE}$ 'yi alçak seviyeye ayarlamak için 2716'ya bir adres ve bir kontrol sinyali gönderilir. Bu adres için 2716 çıkış kelimesi okunur ve FFH ile karşılaştırılır. Eğer okunan kelime FFH'ye eşit değilse, program EXIT'e atlar. EXIT hata adresini yükler ve oradan okunan karakterleri DISBUF (gösterge tamponu) adlı RAM tamponu içine yükler. E0FEH adresindeki monitor programı yordamı OUTDS'ye olan bir JMP (atla) komutu ile hata adresi ve karakter gösterilir. OUTDS klavyede bir tuş basılıncaya kadar adres ve veriyi gösterir. Diğer bir deyişle OUTDS'ye atlan.ık bulunan ilk hatanın adresini gösterir ve kontrolu monitore geri verir.

Eğer okunan karakter FFH ise, okunan son adres dizin kaydedicisine taşınır, bir sonraki belleği göstermek için arttırılır, ve WRKADD tamponuna yeniden yüklenir. X içindeki arttırılmış adres kullanıcı tarafından yüklenmiş bitiş adresi ile karşılaştırılır. Eğer istenen bitiş adresine ulaşılmamışsa, program bir sonraki adresi okumak ve kontrol etmek için geri döner. Eğer bitiş adresine gelinmişse 0070H adresindeki program parçasına atlama yapılır.

Burada PIA ları yeniden ilk kullanıma hazırlamak için dizinli adresleme kullanılır. V_{pp}'yi 25 Volta çevirmek, ve $\overline{OE}$'yi yüksek ve $\overline{CE/PGM}$ 'yi alçak yapmak için *U20*'nin B portu üzerinden 2716'ya bir kontrol kelimesi gönderilir. Daha sonra *U20* 'nin *PA0* dan *PA7*'ye olan bacakları veriyi 2716'ya göndermek üzere çıkış portu olarak değiştirilir.

Yeniden ilk kullanıma hazırlama işleminden sonra kullanıcının yüklediği başlangıç adresi yeniden WRKADD tamponuna, ve kullanıcının yüklediği kaynak adresi RDADD adresine kopyalanır. Şimdi asıl programlamayı PGMLUP yapar. U26'nın *A* ve *B portları* üzerinden bir adres 2716'ya gönderi-

lir. Kaynak adresinden bir veri baytı okunur ve *U20*'nin *A portu* üzerinden 2716'nın çıkışlarına gönderilir. NOP komutu tarafından 3.2 µsaniyelik bir bekleme üretildikten sonra, $\overline{CE/PGM}$ 'yi yüksek seviye yapan bir kontrol kelimesi 2716 'ya gönderilir. *DELAY*, 50 µ saniye gecikme için döner ve daha sonra $\overline{CE/PGM}$ 'yi alçak yapan bir kontrol kelimesi 2716 'ya gönderilir.

Şu andaki programlama adresi WRKADD tamponundan alınır, arttırılır ve WRKADD tamponuna yeniden depolanır. Arttırılmış çalışma adresi kullanıcının yüklediği bitiş adresi ile karşılıştırılır. Eğer bitiş adresine ulaşılmamışsa, RDADD (çalışılan kaynak adresi) içeriği dizin kaydedicisine getirilir, arttırılır ve RDADD tamponuna yeniden depolanır. Daha sonra program bir sonraki adresi programlamak için geri döner. Bütün adresler programlanıncaya kadar PGMLUP tekrarlanır ve sonunda VERIFY kısmına atlanır.

VERIFY ilk olarak V_{pp}'yi +5 Volta indirir. $\overline{OE}$ yüksek seviye olarak tutulur, böylece 2716'nın çıkışları yüzer durumda tutulurlar. Bu çıkışların programlama çıkışı olarak kullanılan *U20*'nin *A portu* ile mücadele etmemesi için yapılır. Daha sonra program *U20*'nın A portunu giriş portu olarak değiştirir. $\overline{OE}$ 'yi alçak seviye yapmak için 2716'ya bir kontrol kelimesi gönderilir. Bu okunabilmeleri için çıkışları yetkilendirir.

STRTAD, WRKADD tamponuna kopyalanır, ve SRCSTR, RDADD tamponuna kopyalanır. WRKADD içindeki adres 2716'ya gönderilir. Bu adresteki veri okunur ve RDADD ile gösterilen veri ile karşılaştırılır. Eğer 2716'dan okunan bayt kaynaktan okunan bayt ile aynı değilse program *FAIL* kısmına atlar. *FAIL* gösterge tamponunun bütün adreslerine ve DISBUF'a 0FH yükler ve OUTDS'ye atlar. Önceden de açıklandığı gibi OUTDS, DISBUF 'un içeriğini gösterir ve kontrolu monitore geri verir. Böylece, bir hata durumu tamamen F göstererek belirtilir.

Eğer 2716'dan okunan bayt kaynak adresinden okunan bayt ile aynı ise son okunan adres WRKADD tamponundan dizin kaydedicisine alınır ve arttırılır. Arttırılmış adres kullanıcının yüklediği ENDADD ile karşılaştırılır. Eğer bitiş adresine ulaşılmamışsa program 2716 içindeki bir sonraki adresi kontrol etmek üzere CKMORE noktasına geri döner. Eğer son adrese ulaşılmışsa akümülatöre 0AH yüklenir ve DISPLY noktasına atlama yapılır. DISPLY bu 0AH'yi DISBUF'un bütün adreslerine yükler. Bu yapıldığı OUTDS'ye bir atlama bütün LED'lerde *A* gösterir ve kontrol monitore döner.

Bu 2716 programlayıcı programının tartışılmasını tamamlar. 2716 ları silme yöntemi Ek C içindeki veri yapraklarında anlatılmıştır.

Bölüm 13'te burada anlatılan terazi ve EPROM programlayıcısı gibi mikroişlemci tabanlı aletlerin aksaklıklarını giderme teknikleri tartışılacaktır.

```
Textronix       M6800       ASM       V 3.3

                ;*************************************************************************
                ;
                ;                    2716  EPROM PROGRAMLAYICI
                ;
                ;*************************************************************************
                ; BU PROGRAM ŞEKİL 12.12'DE GÖSTERİLDİĞİ GİBİ DEĞİŞTİRİLEN BİR
                ; MOTOROLA MEK 6800D2 MİKRO İŞLEMCİ GELİŞTİRME KARTI
                ; ÜZERİNDE GELİŞTİRİLMİŞTİR. PROGRAMLANACAK 2716 U27 SOKETİ
                ; İÇİNE YERLEŞTİRİLİR. PROGRAM VERİSİ BAŞLANGIÇ ADRESİ 6000 H
                ; OLAN BİR BİR BAŞKA 2716'DAN, 2000 H DEN BAŞLAYAN 4 K RAM
                ; BLOĞUNDAN VEYA İŞLEMCİNİN HERHANGİ BİR ADRESLEME
                ; BÖLGESİNDEN GELEBİLİR
                ; MOTOROLA JBUG MONİTOR PROGRAMI BU PROGRAMIN PARÇASI
                ; OLARAK KULLANILIR JBUG MONİTORU ROM İÇİNDE E000 - E3FF
                ; ADRESLERİNDEDİR BU PROGRAMIN ÜÇ ANA KISMI VARDIR.
                ; BAŞLANGIÇ, BİTİŞ VE KAYNAK ADRESLERİ GİRİLDİKTEN SONRA İLK
                ; BÖLÜM İSTENEN BÜTÜN ADRESLERİN HEP BİR DOLU OLDUĞUNU
                ; KONTROL EDER (TAMAMEN SİLİNMİŞ). BİR SONRAKİ BÖLÜM
                ; PROGRAMLAMAYI YAPAR VE SON BÖLÜM PROGRAMLANAN
                ; EPROM'UN KAYNAKLA AYNI OLUP OLMADIĞINI KONTROL EDER


                ; EŞİTLEMELER:
0002  STRTAD    EQU   0002H    ; 2716 İÇİNDEKİ BAŞLANGIÇ ADRESİ
0004  ENDADD    EQU   0004H    ; 2716 İÇİNDEKİ SON ADRES +1
0006  SRCSTR    EQU   0006H    ; KAYNAĞIN BAŞLANGIÇ ADRESİ
0008  WRKADD    EQU   0008H    ; PROGRAMLANACAK 2716 İÇİN GEÇİCİ DEPO
000A  RDADD     EQU   000AH    ; OKUNACAK KAYNAK İÇİN GEÇİCİ DEPO
8000  PIABSE    EQU   8000H    ; PIA'LARIN TABAN ADRESİ
A00C  DISBUF    EQU   0A00CH   ; GÖSTERGE TAMPONUNUN BAŞLANGICI
E0E0  DLY1      EQU   0E0E0H   ; MONİTOR İÇİNDEKİ GECİKME
E149  OUTDS3    EQU   0E149H   ; MONİTOR İÇİNDEKİ KLAVYE BAŞLANGICI
A020  XDSBUF    EQU   0A020H
E3CA  DIGTBL    EQU   0E3CAH
8022  SCNREG    EQU   8022H
8020  DISREG    EQU   8020H
A01C  SCNCNT    EQU   0A01CH

0010            ORG   0010H    ; ASSEMBLE 0010'DAN BAŞLIYACAK


                ; INITIALIZE PIAS

0010  CE8000          LDH   #PIABSE  ; # $ 8000
0013  86FF            LDA   A #0FFH  ; PORT KARAKTERLERİ ÇIKIŞA
0015  A710            STA   A 10H,X  ; U26 PA0 - PA7'YE GÖNDER
0017  A712            STA   A 12H,X  ; U26 PB0 - PB7'YE GÖNDER
0019  A706            STA   A 06H,X  ; U20 PB0 - PB7'YE GÖNDER
001B  8604            LDA   A #04H   ; 6820 PORTLARINA ULAŞMAK İÇİN
                                      ; KONTROL KELİMESİ
001D  A711            STA   A 11H    ; 4 PORTA GÖNDER
001F  A713            STA   A 13H,X  ;
0021  A705            STA   A 05H,X  ;
```

Şekil 12.17 2716 Eprom programlayıcı için Çevirici Dil Programı


```
0023  A707            STA   A 07H,X

                ; PROGRAMLANACAK ADRESLER SİLİNMİŞ Mİ KONTROL ET

0025  DE02            LDX   STRTAD   ; BAŞLANGIÇ ADRESİNİ WRKADD'A
                                      ; KOPYALA
0027  DF08            STX   WRKADD
0029  DE06            LDX   SRCSTR   ; KAYNAK BAŞLANGIÇ ADRESİNİ
                                      ; RDADD'A KOPYALA
002B  D609            STX   RDADD
002D  9608      CKNXT LDA   A WRKADD ; ADRESİN EN BÜYÜK DEĞERLİKLİ
                                      ; BAYTINI AL
002F  0609            LDA   B WRKADD + 1
0031  CEE000          LDX   #PIABSE  ; #$8000
0034  E710            STA   B 10H, X ; ADRESİN EN KÜÇÜK DEĞERLİKLİ
                                      ; BAYTINI 2716'YA GÖNDER
0036  A712            STA   A 12H, X ; ADRESİN EN BÜYÜK DEĞERLİKLİ
                                      ; BAYTINI 2716'YA GÖNDER
0038  8601            LDA   A #01H   ; VPP = 5 V, OE = L, CE = L KONTROL
                                      ; KELİMESİ
003A  A706            STA   A 06H, X ; U20 ÜZERİNDEN 2716'YA GÖNDER
003C  A604            LDA   A 04H, X ; 2716'YI OKU
003E  81FF            CMP   A #0FFH  ; OKUNAN BAYT TAMAMEN BİR Mİ?
0040  260B            BNE   EXIT     ; HAYIR, HATA ADRESİ İLE ÇIK
0042  DE08            LDX   WRKADD   ; ÇALIŞMA PROGRAM ADRESİNİ AL
0044  08              INX            ; ADRESİ ARTTIR
0045  DF08            STX   WRKADD   ; ÇALIŞMA ADRESİNİ YENİDEN YÜKLE
0047  9C04            CPX   ENDADD   ; SON ADRES Mİ?
0049  26E2            BNE   CKNXT    ; HAYIR, BİR SONRAKİ ADRESİ
                                      ; KONTROL ET
004B  2023            BRA   PGMSTR   ; EVET, PROGRAMLAMAYA GİT
004D  BDE29A    EXIT  JSR   MDIS2    ; A İÇİNDEKİ 2 ONDALIK SAYI A VE B
                                      ; İÇİNDE 2 BAYTA
0050  B7A010          STA   A 0A101H ; A'YI DISBUF'A GÖNDER
0053  F7A011          STA   B 0A011H ; B'Yİ DISBUF'A GÖNDER
0056  9608            LDA   A WRKADD ; HATA ADRESİNİN EN BÜYÜK
                                      ; DEĞERLİKLİ BAYTINI AL
0058  BDE29A          JSR   MDIS2    ; BİR KARAKTERLİK BAYTA ÇEVİR
005B  CEA00C          LDX   #0AA00CH ; DISBUF BAŞLANGICI
005E  A700            STA   A 00H, X ; ADRESİN EN BÜYÜK DEĞERLİKLİ
                                      ; BAYTINI DISBUF'A GÖNDER
0060  E701            STA   B 01H, X ; BİR SONRAKİ KARAKTERİ DISBUF'A
                                      ; GÖNDER
0062  9609            LDA   A WRKADD +1; HATA ADRESİNİN ALT BAYTINI AL
0064  BDE29A          JSR   MDIS2    ; BİR KARAKTERLİK BAYTA ÇEVİR
0067  A702            STA   A 02H, X ; KARAKTERİ DISBUF'A GÖNDER
0069  E703            STA   B 03H, X
006B  7EE0FE          JMP   OUTDS    ; HATA ADRESİNİ VE OKUNAN
                                      ; KARAKTERİ GÖSTER
006F  01              NOP
006F  01              NOP
0070  8604      PGMSTR LDA   A #04H   ; VPP = 25 V, OE = H, CE = L
0072  CE8000          LDX   #8000H   ; PIA'LARIN TABAN ADRESİNİ YÜKLE
0075  A706            STA   A 06H, X ; VPP = 25 V, CE = 0, OE = 1
0077  4F              CLR   A        ; A = 00, PIA'NIN DDR'SİNE ULAŞMAK
```

Şekil 12.17 (Devam Ediyor)

```
                                    ; İÇİN KONTROL KELİMESİ
0078  A705          STA  A 05H, X   ; VERİ YÖNLENDİRME KAYDEDİCİSİNİ
                                    ; AÇ
007A  43            COM  A          ; A = FF, ÇIKIŞ İÇİN YÖN KELİMESİ
007B  A704          STA  A 04H, X
007C  8604          LDA  A #04H     ; PORT'A ULAŞMAK İÇİN KONTROL
                                    ; KELİMESİ
007E  A705          STA  A 05H, X   ; U20'NİN A PORTUNU KONTROL İÇİN
                                    ; GÖNDER
0081  DE02          LDX  STRTAD     ; BAŞLANGIÇ ADRESİNİ ÇALIŞMA
0083  DF08          STX  WRKADD     ; ADRESİ TAMPONUNA KOPYALA
0085  DE06          LDX  SRCSTR     ; KAYNAK BAŞLANGIÇ ADRESİNİ
                                    ; OKUMA
0087  DF0A          STX  RDADD      ; ADRES TAMPONUNA KOPYALA
0089  CE8000  PGMLUP LDX  #8000H    ; PIA'LARIN ADRESİNİ YÜKLE
008C  9608          LDA  A WRKADD   ; EPROM ADRESİNİN BÜYÜK DEĞERLİKLİ
                                    ; BİTİNİ AL
008E  A712          STA  A 12H, X   ; ADRESİN EN BÜYÜK DEĞERLİKLİ
                                    ; BİTİNİ EPROM'A GÖNDER
0090  9609          LDA  A WRKADD + 1; WRKADD'IN EN KÜÇÜK DEĞERLİKLİ
                                    ; BAYTINI AL
0092  A710          STA  A 10H, X   ; 2716'YA GÖNDER
0094  DE0A          LDX  RDADD      ; ŞU ANDAKİ OKUMA ADRESİNİ YÜKLE
0096  A600          LDA  A 00H, X   ; BAYTI KAYNAKTAN OKU
0098  B78004        STA  A 8004H    ; VERİYİ 2716'YA GÖNDER
009B  01            NOP             ; 3.2 µ SANİYE BEKLE
009C  8606          LDA  A #06H     ; VPP = 25V, OE = 1, CE = 1
009E  B78006        STA  A 8006H
00A1  CE0F00        LDX  #0F00H     ; GECİKME SABİTİNİ YÜKLE
00A4  09      DELAY DEX             ; 50 µs GECİKME
00A5  26FD          BNE  DELAY
00A7  8604          LDA  A #04H     ; CE KARAKTERİNİ 0 YAP
00A9  B78006        STA  A 8006H    ; KONTROL PORTUNA GÖNDER
00AC  DE08          LDX  WRKADD     ; ŞU ANDAKİ ÇALIŞMA ADRESİNİ YÜKLE
00AE  08            INX
00AF  DF08          STX  WRKADD     ; YENİDEN YÜKLE
00B1  9C04          CPX  ENDADD     ; SON ADRES Mİ?
00B3  2707          BEQ  VERIFY     ; EVET, ONAYLA
00B5  DE0A          LDX  RDADD      ; OKUMA ADRESİNİ YÜKLE
00B7  08            INX
00B8  DF0A          STX  RDADD      ; RDADD TAMPONUNA YENİDEN YÜKLE
00BA  20CD          BRA  PGMLUP     ; BİR SONRAKİ ADRESİ
                                    ; PROGRAMLAMAYA GİT
00BC  CE8000  VERIFY LDX  #8000H    ; PIBASE'Yİ YÜKLE
00BF  8605          LDA  #A 05H     ; VPP = 5V, OE = 1, CE = 0
00C1  A706          STA  A 06H, X   ; VPP'Yİ KAPAT OE = 1
00C3  4F            CLR  A          ; A = 00, DDR'YE ULAŞMAK İÇİN
                                    ; KONTROL KELİMESİ
00C4  A705          STA  A 05H, X   ; KONTROL KAYDEDİCİYE GÖNDER
00C6  A704          STA  A 04H, X   ; VERİ YÖNÜNÜ GİRİŞE ÇEVİR
00C8  8604          LDA  A #04H     ; PORT'A ULAŞMAK İÇİN KONTROL
                                    ; KELİMESİ
00CA  A705          STA  A 05H, X   ; KONTROL KAYDEDİCİYE GÖNDER
00CC  8601          LDA  A #01H     ; VPP = 5V, OE = 0, CE = 0
```

Şekil 12.17 (Devam Ediyor)


```
00CE  A706          STA  A 06H, X   ; 2716 ÇIKIŞINI YETKİLENDİR
00D0  DEF02         LDX  STRTAD     ; BAŞLANGIÇ ADRESİNİ ÇALIŞMA
                                    ; TAMPONUNA KOPYALA
00D2  DF08          STX  WRKADD
00D4  DE06          LDX  SRCSRTR    ; KAYNAK ADRESİNİ OKUMA
                                    ; TAMPONUNA KOPYALA
00D6  DF0A          STX  RDADD
00D8  CE8000  CKMORE LDX #800H
00DB  9608          LDA  A WRKADD   ; ADRESİN EN BÜYÜK DEĞERLİKLİ
                                    ; BAYTINI AL
00DD  A712          STA  A 12H, X   ; 2716'YA GÖNDER
00DF  9609          LDA  A WRKADD +1; ADRESİN EN KÜÇÜK DEĞERLİKLİ
                                    ; BAYTINI AL
00E1  A710          STA  A 10H, X   ; 2716'YA GÖNDER
00E3  A604          LDA  A 04H, X   ; 2716'YI OKU
00E5  DE0A          LDX  RDADD      ; ŞU ANDAKİ OKUMA ADRESİNİ YÜKLE
00E7  A100          CMP  A 00H, X   ; ORADAKİ KARAKTERİ 2716'DAN GELEN
                                    ; KARAKTERLE KARŞILAŞTIR
00E9  2610          BNE  FAIL       ; AYNI DEĞİLSE, FAIL'E ÇIKIŞ
00EB  08            INX
00EC  DF0A          STX  RDADD      ; OKUMA ADRESİNİ YENİDEN YÜKLE
00EE  DE08          LDX  WRKADD     ; WRKADD'I YÜKLE
00F0  08            INX
00F1  DF08          STX  WRKADD     ; WRKADD'I YENİDEN YÜKLE
00F3  9C04          CPX  ENDADD     ; SON ADRES Mİ?
00F5  26E1          BNE  CKMORE     ; HAYIR, BİR SONRAKİ ADRESİ
                                    ; KONTROLA GİT
00F7  860A          LDA  A #0AH     ; BAŞARI KARAKTERİ = A
00F9  2002          BRA  DISPLAY    ; AAAA – AA GÖSTER
00FB  860F    FAIL  LDA  A #0FH     ; BAŞARISIZLIK KARAKTERİ = F
00FD  CEA00C  DISPLAY LDX #0A00CH   ; TAMPON BAŞLANGIÇ ADRESİNİ
                                    ; GÖSTER
0100  A700          STA  A 00H, X   ; BÜTÜN TAMPON
0102  A701          STA  A 01H, X   ; ADRESLEME BAŞARILI VEYA BAŞARISIZ
0104  A702          STA  A 02H, X   ; KARAKTERİ GÖNDER
0106  A703          STA  A 03H, X
0108  A704          STA  A 04H, X
010A  A705          STA  A 05H, X
010C  BDE0FE        JSR  OUTDS      ; BAŞARILI MI? BAŞARISIZ MI? GÖSTER
010F  3F            SWI
                ;*****************************************************
                ; DISBUF İÇİNDEKİ 6 SAYIYI GÖSTERMEK
                ; İÇİN YORDAM
                ;*****************************************************
      E0FE          ORG  0E0FEH
E0FE  CEA00C  OUTDS LDX  #DISBUF    ; BAŞLANGIÇ ADRESİNİ AL
E101  A600    OUTDS1 LDA A 0,X      ; İLK SAYIYI AL
E103  4C            INC  A
E104  08            INX
E105  FFA020        STX  XDSBUF     ; GÖSTERİCİYİ SAKLA
E108  CEE3C9        LDX  #DIGTBL–1
E10B  08      OUTDS2 INX
E10C  4A            DEC  A          ; DİZİNİN OLDUĞU YERİ GÖSTER
E10D  26FC          BNE  OUTDS2
E10F  7F8022        CLR  SCNREG     ; GÖSTERGEYİ SİL
```

Şekil 12.17 (Devam Ediyor)

```
E112  A600        LDA   A 0, X        ; DİZİYİ AL
E114  B78020      STA   A DISREG      ; PARÇALARI AYARLA
E117  B6A01C      LDA   A SCNCNT
E11A  B78022      STA   A SCNREG      ; SAYIYI SEÇ
E11D  CE004D      LDX   #4DH          ; 1 µs GECİKMEYİ AYARLA
E120  8DBE        BSR   DLY1          ; 1 µs GECİKME
E122  FEA020      LDX   XDSBUF        ; GÖSTERİCİYİ YENİDEN YÜKLE
E125  8CA012      CPX   #DISBUF + 6
E128  271F        BEQ   OUTDS3
E12A  74A01C      LSR   SCNCNT        ; HAYIR, BİR SONRAKİ SAYIYA GİT
E12D  20D2        BRA   OUTDS 1

;*****************************************************
; A'NIN ALT NİBİL'INI B'YE VE A'NIN ÜST NİBİLINI
; A'NIN ALT NİBİLINA TAŞIYAN YORDAM
;*****************************************************

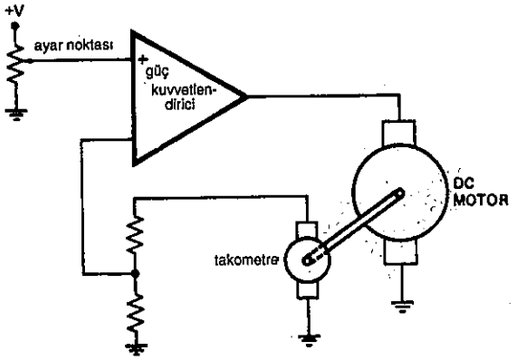
      E29A        ORG   0E29AH
E29A  16    MDIS2 TAB
E29B  C40F        AND   B #0FH        ; ALT NİBİL'I MASKELE
E29D  84F0        AND   A #0F0H       ; ÜSTNİBİL'I MASKELE
E29F  44          LSR   A
E2A0  44          LSR   A
E2A1  44          LSR   A
E2A2  44          LSR   A             ; ÜST NİBİL ALT NİBİL'A
E2A3  39          RTS
                  END
```

Şekil 12.17 (Devam Ediyor)

ENDÜSTRİYEL İŞLEM KONTROLUNUN GÖZDEN GEÇİRİLMESİ

Mikroişlemci ve *mikrobilgiyasarların* önemli etkide bulunduğu bir alan *endüstriyel işlem kontroludur*. Endüstriyel işlem kontrolu ilk olarak sistem değişkenlerinin (motor hızı, sıcaklık, bir tanktaki sıvı seviyesi, veya bir maddenin kalınlığı gibi) ölçülmesi ve daha sonra her değişkenin değeri, o değişken için belirlenmiş bir değere eşit oluncaya kadar sistemin ayarlanması ile ilgilenir. Sistem kontrolörü her değişkeni ayar-noktası değerinde tutmalı ve sistemde olabilecek herhangi bir değişikliği, örneğin motor üzerindeki yükün artması gibi, mümkün olduğu kadar hızla ve hassasiyetle telafi etmelidir. Basit bir örnek, bir işlem değişkenin kontroluna geleneksel yaklaşımı gösterecek ve kontrol sistemlerinde kullanılan bazı terimleri açıklayacaktır.

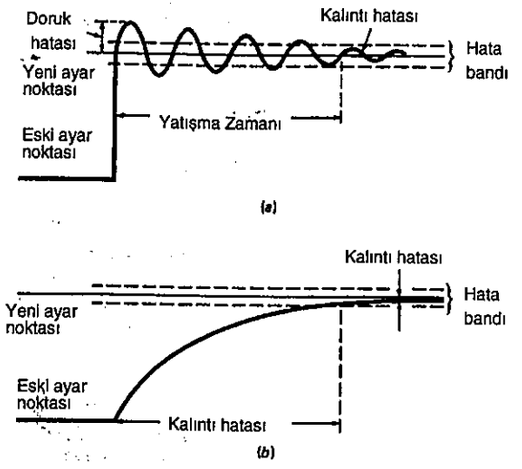
Şekil 12. 18 bir dc motorun hızını kontrol etmek için kullanılan bir devreyi göstermektedir. Motorun şaftına bir dc jeneratör, ya da takometre bağlanmıştır, ve bu motorun hızına orantılı bir gerilim üretir. Çıkış gerilimi tipik olarak dakikada binlerce dönüş için birkaç volttur. Takometreden gelen gerilimin bir bölümü güç kuvvetlendiricisinin (-) girişine geri beslenir. Güç kuvvetlendiricinin (+) girişine ayar noktası olarak pozitif bir gerilim uygulanır. Güç uygulandığında takometreden (-) girişe geri beslenen gerilim yaklaşık olarak (+) girişindeki ayar noktası gerilimine eşit oluncaya kadar motor hızlanır. Bu geri besleme metodu *oransal kontrol* olarak adlandırılır. Böyle bir kontrol döngüsü yükün çok değişmediği uygulamalarda motor hızını sabit tutar. Bazı Winchester disk sürücüleri ve yüksek kaliteli pikaplar bu tip hız kontrolunu kullanırlar.



Şekil 12.18 Takometre Geribeslemesi ile bir dc motorun hızının kontrol edilmesi

Yükün ya da ayar noktasının büyük miktarlarda değiştiği uygulamalarda daha karmaşık bir geribesleme sistemi kullanılır.

Şekil 12.18 deki motor-hızı örneğine benzer basit bir kontrol döngüsünün ayar noktasındaki bir değişikliğe vereceği muhtemel iki tepki şekil 12.19 da gösterilmiştir. Şekil 12.19 *a az sönümlü* bir tepkiyi göstermektedir. Bu durumda motor hızı veya sıcaklık gibi bir değişken yeni ayar noktasını aşar ve bir süre aşağı yukarı salınır. Değişkenin verilen bir hata aralığı (band) içine girmesine kadar geçen süre *yatışma süresi* olarak adlandırılır. Atlamalar yatıştıktan sonra dahi kuvvetlendiricinin sonlu kazancından dolayı bir kalıntı hatası olacaktır. Şekil 12.19*b* sistem tepkisi üzerinde *aşırı sönümün* etkisini göstermektedir. Bu durumda değişken ayar noktasını aşmaz. Değişken, hata bandı içine gelir ve orada kalır. Fakat, uzun bir süre geçmesine rağmen gene kuvvetlendiricinin sonlu kazancından dolayı bir kalıntı hatası olduğuna dikkat ediniz. Yalnızca oransal kontrol kullanan bir geri besleme kontrol sisteminde ayar noktası ile gerçek çıkış arasında bazı kalıntı hatası olacaktır.

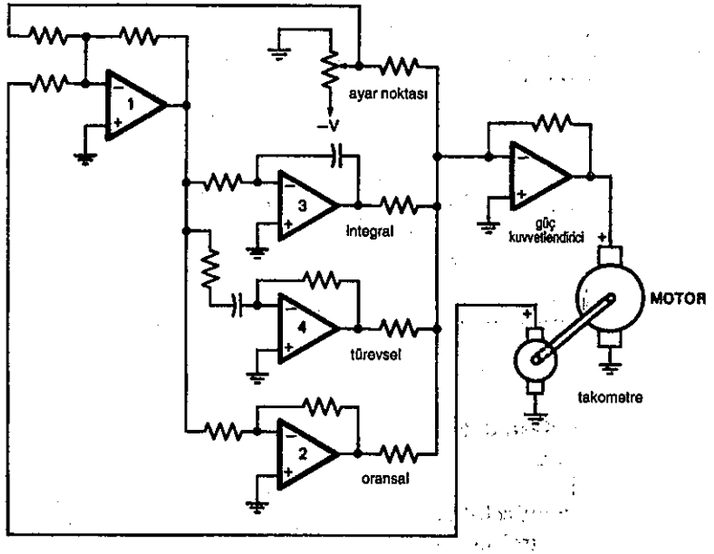


Şekil 12.19 Ayar noktasındaki bir değişikliğe bir sistemin tepkisi
***a)* Az sönümlü *b)* Aşırı sönümlü**

Kalıntı hatasının düzeltilmesi için *integral geribesleme* kullanılır. İntegral geribesleme hata gerilimi ile hatanın mevcut olduğu zamanın çarpımına orantılı bir sinyaldir. Şekil 12.20 integral geribeslemenin nasıl üretilebileceğini ve basit oransal kontrola nasıl ilave edileceğini göstermektedir. Kuvvetlendirici 1 ayar noktası gerilimi V_s ile takometreden gelen geri besleme gerilimi

V_T'yi karşılaştırır. Kuvvetlendiricilerin evirici yapısında bağlanmalarından dolayı V_s'nin negatif olması gerektiğine dikkat ediniz. Eğer ayar noktası ile gerçek motor hızı arasında bir hata yoksa, kuvvetlendirici 1 çıkışı 0V olur ve 2, 3, ve 4 nolu kuvvetlendiricilerin etkisi olmaz. Eğer V_s ve V_T arasında bir hata varsa 2, 3, ve 4 nolu kuvvetlendiricilerin girişlerine bir sinyal uygulanır. Kuvvetlendirici 2 oransal geri besleme üretir. Kuvvetlendirici 3 bir *integral* alıcıdır ve *integral geribesleme* üretir. Hata olduğu sürece integral alıcının çıkışı hatayı ortadan kaldırmak için yukarı veya aşağı doğru yükselir. Fakat *integral* alıcının çıkışında yeterli bir çıkış değeri olması için hata sinyalinin bir süre mevcut olması gerekir bu da integral alıcının cevap vermekte yavaş davranmasına sebep olur.

Yavaş tepkinin tedavisi türevsel geribesleme kullanmaktır. *Türevsel geri* besleme V_s ve V_T arasında *değişim* hızına orantılı bir sinyaldir. Şekil 12.20'deki 4 nolu kuvvetlendirici türevsel geribesleme veren bir türev alıcıdır. Girişinde bir hata gerilimi gözükür gözükmez türev alıcı hatayı düzeltmek için hızlı bir geribesleme sinyali verir. Fakat, girişindeki kondansatörden dolayı türev alıcının yalnızca çok kısa bir süre içinde etkisi vardır.



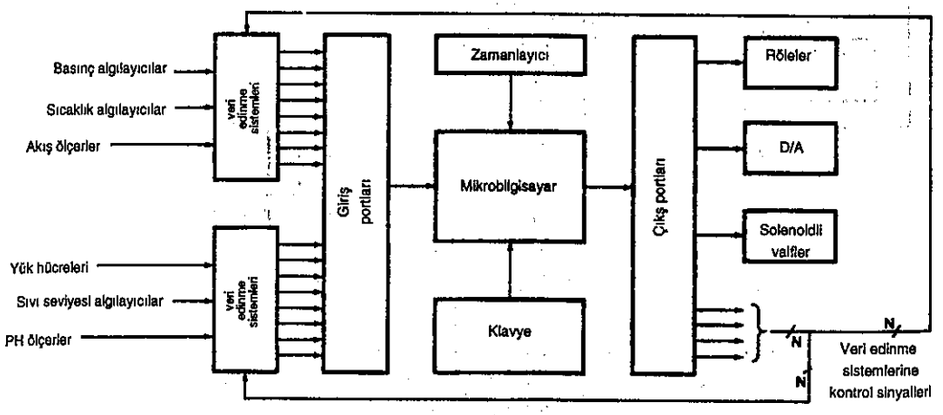
Şekil 12.20 Oransal, integral ve türevsel geribesleme kontrol gösteren işlem kontrol döngüsü

Üç tip geribeslenmenin hepsini kullanan süreç kontrol döngüleri *oransal-integral-türevsel*, veya PID, kontrol döngüleri olarak adlandırılır. *İşlem değişkenleri* bir bilgisayarın mikro saniyelik çalışmasından daha yavaş değiştikleri için, basit I/O devreleri olan bir mikrobilgisayar Şekil 12.20 de gösterilen analog devreli PID döngülerinin birkaçının bütün fonksiyonlarını yerine getirebilir.

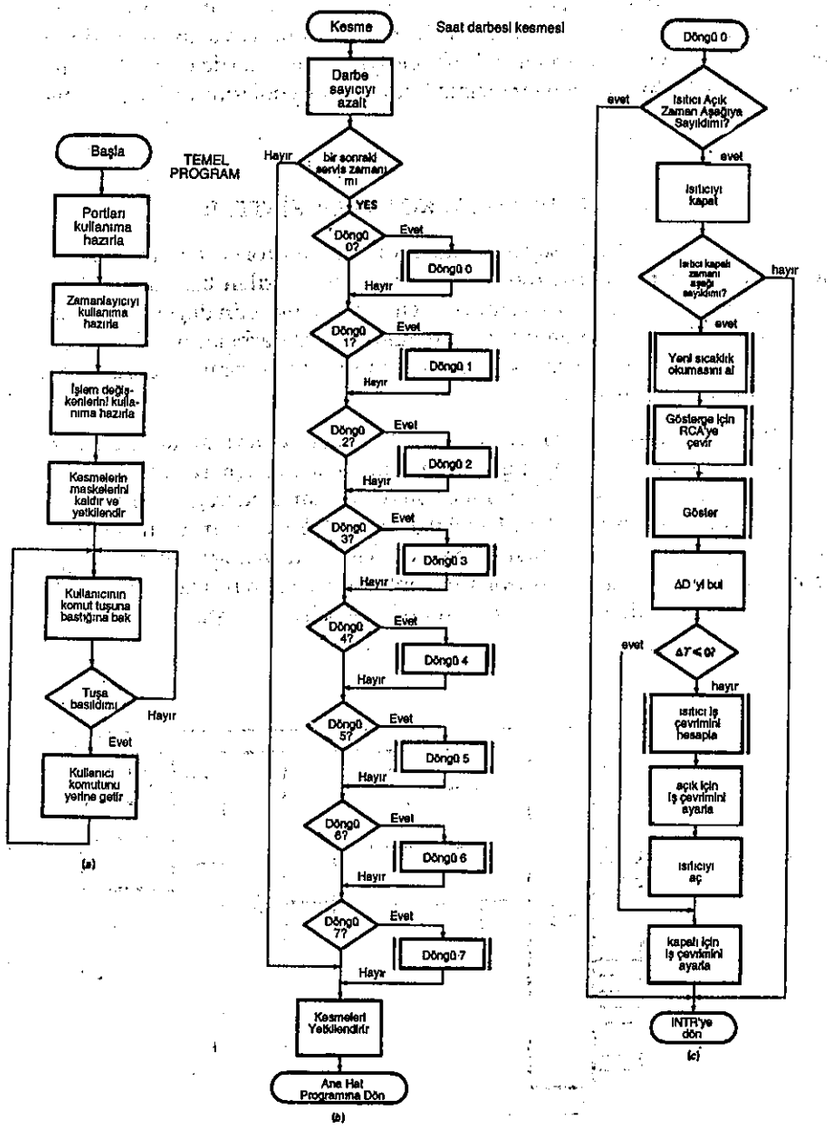
SDK-85-TABANLI BİR İŞLEM KONTROL SİSTEMİ

Şekil 12.21 mikrobilgisayar tabanlı bir işlem kontrol sisteminin blok diyagramını göstermektedir. Kesme üzerine servis verilen bir klavye, kullanıcının ayar noktalarını girmesini ve işlem değişkenlerinin değerlerini okumasını sağlar. Bir programlanabilir zamanlayıcı, saat darbelerini sağlar. Saat darbeleri bilgisayara bir sonraki kontrol döngüsüne ne zaman servis vereceğini söylemekte kullanılır.

Veri edinme sistemleri *basınç, sıcaklık* ve *diğer tipteki* algılayıcılardan gelen sinyalleri mikrobilgisayar girişinde okunabilmeleri için sayısal sinyallere çevirir. Veri edinme sistemleri mikrobilgisayar tarafından gönderilen sinyaller tarafından kontrol edilir. Mikrobilgisayarın çıkış portlarından gönderilen diğer sinyaller rölelere, solenoidlere ve sayısaldan analoga çeviricilere gider. Bu elemanlar, işlem değişkenlerini kontrol etmek için ısıtıcıları, akış valflerini, motorları ve diğer aktüatörleri (hareketlendirici) çalıştırır.



Şekil 12.21 Mikrobilgisayar tabanlı işlem kontrol sisteminin blok diyagramı



Şekil 12.22 SDK-85 Kesme Sürmeli İşlem Kontrolör Sistemi için Program Modüllerinin Akış Diyagramları

Şekil 12.22 mikrobilgisayar tabanlı kontrol sistemi için yazılan bir programın nasıl çalıştığını akış diyagramı şeklinde göstermektedir. Güç kaynağı açıldıktan sonra temel program portları, zamanlayıcıyı ve süreç sabitlerini ilk kullanıma hazırlar. Daha sonra kullanıcının klavyeden bir komut girmesini ya da zamanlayıcıdan bir saat darbesi gelmesini bir döngü içinde bekler. Bir saat darbesi geldiğinde mikrobilgisayar bir kesme servis yordamını çağırır. Servis yordamı saat darbelerini sayar. Her 20 saat darbesinde mikrobilgisayar sekiz kontrol döngüsünden birine servis vermek üzere bir yordam çağırır. Bir kontrol döngüsüne hizmet verildikten sonra mikrobilgisayar temel programa döner ve bir sonraki saat darbesini ya da kullanıcı komutunu bekler. Asıl kontrol yordamları birbirinden bağımsızdır, böylece herbiri bağımsız olarak geliştirilebilir ve test edilebilir.

Şekil 12.22 c'deki DÖNGÜ 0 bir su banyosunun sıcaklığını bir ayar noktası sıcaklığına getiren ve o sıcaklıkta tutan bir kontrol döngüsü servis yordamı örneğidir. DÖNGÜ 0 analogdan sayısala çevirme yapan bir yordamı ve gerçek ve ayar noktası sıcaklıkları arasındaki farka bağlı olarak geri besleme değerini hesaplayan bir yordamı çağırır. Isıtıcı elemana gönderilen güç, gönderilen darbelerin iş çevrimlerini değiştirerek kontrol edilir.

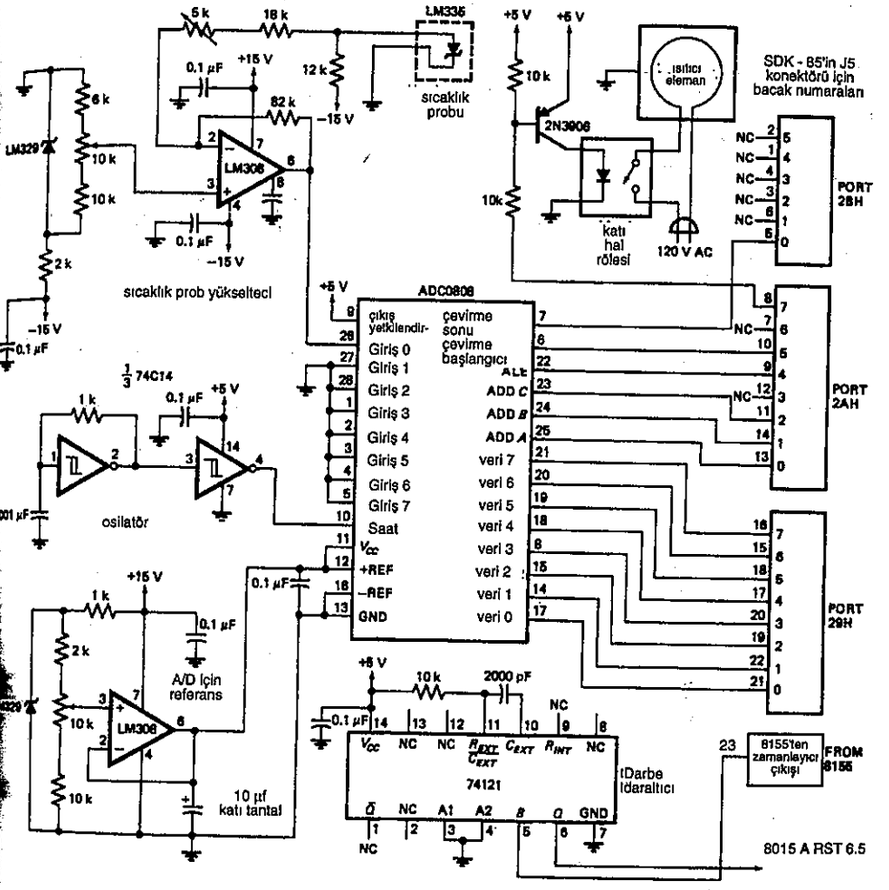
Takip eden bölümde bir su banyosunun sıcaklığını 0 ile 99°C aralığındaki ayar noktalarında gözlemek ve kontrol etmek için bir SDK-85 kartına ilave edilmiş donanımı tartışacağız. Daha sonra SDK-85 tabanlı bir kontrol sistemi için programı gösterecek ve tartışacağız.

BİR SDK-85 İŞLEM KONTROL SİSTEMİ İÇİN DONANIM

SAAT DARBESİ İÇİN DONANIM: Bu kontrolör için saat darbeleri Bölüm 11'de anlatıldığı gibi kesme tabanlı olarak üretilir. 8155 zamanlayıcı 1 KHz kare-dalga darbe katarı çıkışı için programlanmıştır. Şekil 12.23'de gösterilen bir darbe daraltıcı devre darbelerin yüksek zamanını yaklaşık 15μ saniyeye düşürür, böylece bir darbe işlemciye birden fazla kesim uygulayamaz. Darbe daraltıcının çıkışı 8085 A'nın RST 6.5 girişine bağlanır. Her 1m saniyede bir saat darbe kesimi üretilir.

SICAKLIK KONTROL DÖNGÜSÜ İÇİN DONANIM

Endüstride, örneğin kimyasal maddelerin üretiminde, ortak bir ihtiyaç bazı bileşikleri bir su banyosu içinde belirli bir sıcaklığa kadar ısıtmak, bu sıcaklığı bir süre aynen tutmaktır. Sıcaklığın ayar noktasını aşmaması veya ayar noktası etrafında aşağıyukarı fazla titreşmemesi çok önemlidir. Bu mikrobilgisayar kontrolu ile kolaylıkla başarılabilir. Şekil 12.23, sıcaklık kontrol döngüsü için bir SDK-85'e eklenmesi gereken parçaları göstermektedir.



Şekil 12.23 Bir su banyosunun sıcaklığını ölçmek ve kontrol etmek için SDK-85'e eklenen devreler.

Ana parçalar bir sıcaklık algılayıcı, bir analogtan sayısala çevirici ve bir katı hal rölesidir.

Sıcaklık algılayıcı bir LM335'tir. LM335, +10mV/C° 'lik doğrusal sıcaklık katsayısına sahip bir zener diyottur. Düşük-kaymalı LM308 kuvvetlendiricisi bunu analogdan sayısala çevirici için 40 mV/C° 'ye yükseltir. Bu istenen 0 ile 99°C aralığının analogdan sayısala çevirici aralığının büyük kısmını kullanmasını sağlar. Kuvvetlendiricinin evirmeyen girişine uygulanan negatif referans gerilimi, çıkışın 0.0 Voltta 0.0°C 'ye ayarlanmasına izin verir. LM 335-10 ile +100°C aralığında çalışacak şekilde üretilmiştir. Bir sıcaklık probu LM335'i 1.4 inç (6.3mm) çaplı cam tüpün ucunda yüksek sıcaklıkta silikon kapayıcı ile kapatılmasıyla yapılmıştır.

Analogdan sayısala çevirme bir National Semiconductor ADC 0808 veri toplama sistemi tarafından yapılır. ADC 0808 'in kalbi 8 bit-ikilik çıkışlı ardışık-yaklaştırmalı analogdan sayısala çeviricidir. Analogdan sayısala çeviricinin girişinde sekiz-girişli bir çoklayıcı bulunmaktadır. Analogdan sayısala çevirici tarafından sayısallaştırılan giriş kanalı ADD A, ADD B, ve ADD C girişlerindeki 3 bitlik adres tarafından belirlenir. Sekiz girişli bir veri toplama sistemi seçilmiştir, böylece sekiz değişken aynı analogdan sayısala çevirici ile gözlenebilir. Bir *Schmitt tetikleme osilatörü* analogdan sayısala çevirici için 300KHz'lik bir saat üretir. LM 329 az-kaymalı Zener diyotun gerilim düşmesi, analogdan sayısala çevirici için V_{cc} ve 5.12 Voltluk V_{REF} üretilmesi için bir *LM* 308 kuvvetlendirici tarafından tamponlanır. Bu referans gerilimi ile analogdan sayısala çevirici herbiri 20mV olan 256 adıma sahip olur.

Şekil 12.24 ADC0808'in zamanlama dalga şekillerini ve parametrelerini göstermektedir. Cihaza ilk olarak gönderilecek sinyallerin sırasına dikkat ediniz. İstenen kanalın adresi çoklayıcı adres girişlerine gönderilir. 50 nanosaniye sonra, adres kilitleme yetkilendirmesi (ALE) yüksek olarak gönderilir. 2.5µ saniye sonra çevirmeye başlama sinyali önce yüksek ve sonra alçak yapılır, ve ALE alçak yapılır. Çevrim sonu sinyali yüksek olduğundan veri analogdan sayısala çeviriciden porta okunabilir.

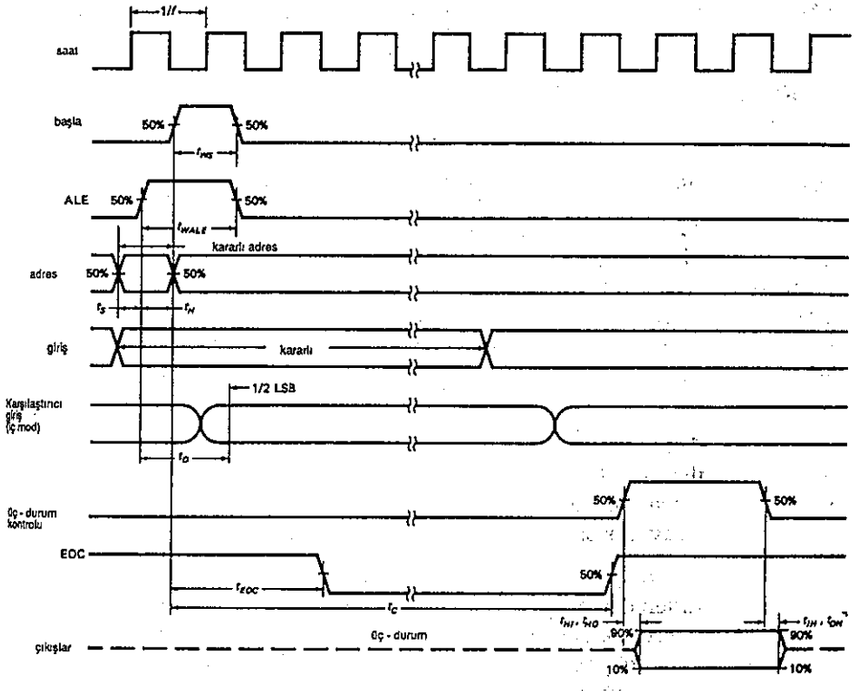
Isıtıcı, 120*Vac* gerilimli sıcak plaka veya bir bardak kahve için su ısıtmakta kullanılan 120*Vac* gerilimli daldırma ısıtıcısı olabilir. Bir katı hal rölesi ısıtıcıyı açar ve kapar. Bir mikrobilgisayar, değişken iş çevrimleri için röleyi açıp kapayarak sıcaklık çıkışını kontrol eder.

Kullanılan röle optik olarak yalıtılmış, sıfır gerilimde iletime geçip, sıfır akımda iletimden çıkan tiptedir. Bu tip röle biraz pahalıdır, fakat çok hızlı açılıp kapanabilir, kontrol girişi ve kaynak çıkışı binlerce voltluk elektriksel yalıtım sağlar, ve güç devamlı olarak ac hat geriliminin yarım çevrimlerinin birkaç katı sayıda sağlanacağından çok az elektromanyetik karışım üretir.

Çok düşük güç uygulamalarında bir analogdan sayısala çevirici ve güç kuvvetlendiricisi bir ısıtıcıyı sürmek için kullanılabilir. Fakat, bu durum yüksek güç uygulamalarında pratik değildir çünkü güç kuvvetlendiricisi yük kadar veya daha fazla güç harcar. Örneğin, 5000 Wattlık bir ısıtıcıyı süren kuvvetlendirici 5000W veya daha fazla güç harcayacaktır. Sayısaldan analoga çevirici ve kuvvetlendirici yaklaşımı da ac hat gerilimini kolayca kullanamaması nedeniyle dezavantajlıdır. Darbe genişlik modülasyonu tekniği ac-dc (evrensel) motorların hızlarını kontrol etmekte kullanıldıkları için ilave avantaja sahiptir.

Potter-Brumfield E0M1DB72 gibi bir röle açılması için 5 voltta yalnızca 11mA akım gerektirir ve uygun bir soğutucu ile 25A akımı kontrol edebilir.

Rölenin girişindeki sürücü transistör üç amaca hizmet etmektedir: röle için sürücü akımını sağlar, port bacağını röleden yalıtır ve gücün ilk uygulanması sırasında röleyi iletim dışı tutar. Bu, bir resetten sonra port bacakları yüzer olduğundan iyi bir fikirdir.



Şekil 12.24 ADC0808 Veri Toplama Sistemi Zaman Dalga Şekilleri

KONTROLÖR PROGRAMI

ÖZET: Program yüksek şekilde yapısal olarak yazılmıştır. Yapısal programlamanın anlamı yapılan her işin ya da çözülen her problemin bağımsız olarak yazılabilecek ve hataları bulunup düzeltilebilecek iç içe küçük modüllere parçalanması demektir. İdealde her modülün bir giriş ve bir çıkış noktası olmalıdır. Şekil 12.22 bu programın nasıl yapılandığının akış diyagramlarını göstermektedir. Takip eden bölümlerde programın her ana kısmının çalışmasını açıklayacağız. Şekil 12.25 çevirici dil listesinin tamamını göstermektedir.

ANAHAT VEYA TEMEL PROGRAM: Programın anahat bölümü *port 29H*'yi *giriş, port 2AH*'yi *çıkış* ve *port 2B*'yi *giriş* olarak kullanıma hazırlar. Aynı zamanda 8155 zamanlayıcıyı 1KHz kare dalga için ayarlar, değişik kontrol sabitlerinin başlangıç değerlerini atar, kesmelerin maskelerini kaldırır, ve kesmeleri yetkilendirir. Daha sonra saat darbe kesmesini beklerken JMP HERE komutunu üst üste defalarca yerine getirir. Normalde bir anahat programı kullanıcının bir ayar noktasını değiştirmesini ya da sıcaklık gibi bir değişkenin o andaki değerini okumak istemesini algılayabilmek için klavyeyi taramalıdır. Yerden tasarruf etmek için klavye taraması burada ihmal edilmiştir.

INTR YORDAMI: INTR yordamı saat darbelerini sayar ve her 20 saat darbesinden sonra bir sonraki kontrol döngüsüne hizmet verir. 8085A, zamanlayıcıdan bir RST 6.5 kesme girişi aldığında kesmelerin yetkilendirmelerini kaldırır ve 0034H'ye gider. SDK-85 monitor programı bu adreste 20C8H'ye atlama komutuna sahiptir. 8085A, 2034H'de INTR yordamına geldiğinde kaydedicileri yığının üstünde saklar ve INTCNT adındaki bellek adresinden kesme sayısını alır. (INTCNT anahat programında ve her kontrol döngüsüne hizmet verildikten sonra 14H(20_{10}) sayısına eşitlenir). Sayıcı azaltılır ve yeniden depolanır. Eğer sayıcının azaltılması onu sıfıra kadar düşürmezse yordam EXIT2'ye atlar. Daha sonra kaydedicileri yeniden yükler, kesmeleri yetkilendirir, ve bir sonraki saat darbesini beklemek için anahat programına döner.

Eğer 20_{10}(14H) kesme sayılırsa, INTR yordamı hizmet verilecek bir sonraki kontrol döngüsü numarasını LPCNT isimli bellek adresinden alır. Bu numara hangi kontrol döngüsü yordamının çağrılacağını belirlemek için azaltılır. Her kontrol döngüsü bir sonra hizmet verilecek kontrol döngüsünün numarasını LPCNT'ye yükler ve akümülatörde 0FFH değeri ile anahat programına döner. Akümülatör içindeki 0FFH döngü kontrol edicinin kalan kısımlarının çalışmasının durmasına sebep olur. Eğer LPCNT bir şekilde hatalı bir sayı alırsa ve kontrol edici tarafından bir döngü bulunmazsa, LPCNT anahat programına dönmeden önce 01H ile yüklenir. Bu bir sonra döngü 0'a hizmet verilmesine sebep olur. Eğer bir döngü bulunur ve hizmet verilirse, INTCNT 20_{10}(14H) sayısına eşitlenir, kaydediciler yeniden depolanır, kesmeler yetkilendirilir, ve programın çalışması ana programa döner. LPCNT ve

INTCNT gibi değişkenlerin takiplerinin kolay olduğu isimlendirilmiş bellek adreslerinde tutulduğuna dikkat ediniz.

DÖNGÜ 0: Su banyosunun sıcaklığını kontrol eden yordam *döngü 0*'dır. Şekil 12.22 bu yordam için akış diyagramını göstermektedir. Katı hal rölesi için iletimde olma zamanı DCHI isimli bir sayıyı aşağı sayarak belirlenir. İletimde olmama zamanı ise DCLO isimli bir sayı tarafından belirlenir. Başlangıçta anahat programı DCHI ve DCLO 'yu 01H'ye eşitler. Bu yüzden *DÖNGÜ 0* programı ilk defa çağrıldığında program ADRD adındaki analogtan sayısala çevirici yordamına gider. ADRD yordamı bir analogtan sayısala çevrim yapmak için ADC0808'i kullanır. Bu işlemi B kaydedicisinden kendisine geçen kanal numarası için yapar. Sıcaklığın gerçek ikilik değerinin 2 katı olan ikilik sayı akümülatör içinde *DÖNGÜ 0*'a döner. En büyük değerlikli biti maskeleyerek ve sonucu 1 bit pozisyonu sağa döndürerek sayı 2 ile bölünür ve sıcaklığın gerçek ikilik değerini derece Celcius, olarak verir. İkilik sayı bellekte saklanır.

Daha sonra sıcaklığın ikilik değerini SDK-85 üzerinde gösterebilmek için BCD'ye çeviren *CNVT1* adındaki yordam çağrılır. *CNVT1* terazi programı için Şekil 12.5'teki BINCVT yordamının küçültülmüş şeklidir. *CNVT1* iki BCD rakamı B kaydedicisi içinde geri döndürür. Bunlar akümülatöre taşınır ve SDK-85 monitor yordamı UPDDT bu BCD rakamları SDK-85 göstergesinin veri alanında göstermek için çağrılır.

DÖNGÜ 0 kontrol programının bir sonraki kısmı gerçek sıcaklık ile ayar noktası sıcaklığını karşılaştırır ve iş çevrim sabitlerini buna göre ayarlar. İkilik sıcaklık değeri yeniden çağrılır ve ayar noktası sıcaklığından çıkarılır. Eğer fark (hata) 0 ya da negatif ise iletim zamanı çıkışı (DCHT) 0 yapılır, iletimde olmama zamanı (DCLO) 7FH yapılır ve yordamdan çıkılır. Eğer ayar noktası ve gerçek sıcaklık arasındaki fark pozitif ise sıcaklık düşük demektir ve ısıtıcı çalıştırılmalıdır. Bu durumda DCHI'nın değeri ısıtıcıya sabit bir çalışma zamanı vermek için 04H'ye eşitlenir. iletimde olmama zamanı DCLO aşağıdaki şekilde hesaplanır. Eğer ΔT büyük ise DCLO küçük olmalıdır, böylece ısıtıcı uzun bir iş çevrimi için çalışır. Eğer ΔT küçük ise DCLO büyük olmalıdır böylece ısıtıcı kısa bir iş çevrimi için çalışır. Bu ilişkiye ilk yaklaşım ΔTxDCLO = 100_{10} = 64H'dir. Bu durumda DCLO değeri yalnızca 64H'nın ΔT ile bölünmesinin sonucudur. Örneğin, eğer ΔT = 20°C ya da 14H ise, 64H÷14H DCLO için 5 değerini verir. LXI B, 0064H komutu 64H'yi B ve C kaydedicilerine yükler, ve daha sonra DIV1 yordamı bu sayıyı akümülatör içindeki ΔT ile böler. Bölüm, C kaydedicisi içinde döner, akümülatöre taşınır ve DCLO içine depolanır. Isıtıcı çalıştırılır ve kontrol döngüsü yordamında çıkılır. Çıkış işleminin bir kısmı, bir sonra servis verilecek döngünün numarasını LPCNT içine yüklemek ve kesme sayıcısını 14H'ye eşitlemektir. Akümülatör 0FFH ile yüklenmelidir, böylece INT dönüşünde programın çalışması döngü kontrol edicisini atlar.

```
00001                          ; BU PROGRAM 8 İŞLEM DÖNGÜSÜNE SIRAYLA HİZMET
                               ; VERİR.
00002                          ; 2800H ADRESİNDE BİR GENİŞLEME RAM'I OLAN INTEL
                               ; SPK-85 ÜZERİNDE
00003                          ; ÇALIŞACAK ŞEKİLDE YAZILMIŞTIR. KONTROL
                               ; DÖNGÜLERİNİN ZAMANLAMASI
00004                          ; KART ÜZERİNDEKİ 8155 ZAMANLAYICI İLE YAPILIR.
                               ; PROGRAMDAKİ KONTROL.
00005                          ;
00006                          ; DÖNGÜSÜ 0 BİR SU BANYOSUNUN SICAKLIĞINI DARBE
                               ; GENİŞLİĞİ METODU
00007                          ; İLE KONTROL EDER
00008                          ;
00009                          ;
00010  2000          LPCNT    EQU   2000H   ; SERVİS VERİLECEK BİR
                                             ; SONRAKİ DÖNGÜ.
00011  2001          INTCNT   EQU   2001H   ; KESME SAYICI
00012  2002          DCHI     EQU   2002H   ; ISITICI AÇIK SAYICI
00013  2003          DCLO     EQU   2003H   ; ISITICI KAPALI SAYICI
00014  036E          UPDDT    EQU   036EH   ; MONİTÖR GÖSTERGE
                                             ; YORDAMI
00015  2004          HOT      EQU   2004H   ; ŞU ANDAKİ SICAKLIK
00016  2005          STPNT    EQU   2005H   ; AYAR NOKTASI SICAKLIĞI
00017
00018                          ; ANA PROGRAMIN BAŞLANGICI
00019
00020  2010    >     ORG   2010H           ; ANA PROGRAMIN BAŞLANGICI
00021  2010  31C220  LXI   SP, 20C2H        ; YIĞIN GÖSTERİCİYİ AYARLA
00022  2013  3E00    MVI   A, 00H          ; ZAMANLAYICI ALÇAK
                                             ; SAYICISI
00023  2015  D32C    OUT   2CH
00024  2017  3E4C    MVI   A, 4CH          ; ZAMANLAYICI YÜKSEK
00025  2019  D32D    OUT   2DH             ;
00026  201B  3EC2    MVI   A, 0C2H         ; ZAMANLAYICIYI BAŞLAT,
                                             ; PORT A GİRİŞ
00027  201D  D328    OUT   28H             ; PORT B ÇIKIŞ, PORT C GİRİŞ
00028  201F  3E14    MVI   A, 014H         ; KESME SAYICIYI 20 ONDALIK
00029  2021  320120  STA   INTCNT          ; SAYIYA AYARLA
00030  2024  3E01    MVI   A, 01H
00031  2026  320020  STA   LPCNT           ; İLK DÖNGÜYE AYARLA
00032  2029  320220  STA   DCHI            ; ISITICI AÇIK SAYICIYI AYARLA
00033  202C  320320  STA   DCLO            ; ISITICI KAPALI SAYICIYI
                                             ; AYARLA
00034  202F  3E80    MVI   A, 80H          ; ISITICIYI KAPAT
00035  2031  D32A    OUT   2AH
00036  2033  3E0D    MVI   A, 0DH          ; KESMELERİN MASKELERİNİ
                                             ; KALDIR
00037  2035  30      SIM
00038  2036  FB      EI
00039  2037  C33720  >  HERE  JMP   HERE    ; KESME İÇİN BURADA BEKLE
00040                                        ; ANA PROGRAMI BENZET
00041                                        ; KULLANICININ KOMUT
```

Şekil 12.25 Sıcaklık kontrol eden SDK-85 işlem Kontrolör Sistemi için çevirici dil listesi


```
00042                                        ; GİRMESİNİ BEKLE
00043
00044                          ; ZAMANLAYICI KESMESİ İLE PROGRAM BURAYA GİDER
00045
00046  203A  C5        INTR   PUSH  B
00047  203B  D5               PUSH  D
00048  203C  E5               PUSH  H
00049  203D  F5               PUSH  PSW     ; HERŞEYİ SAKLA
00050  203E  3A0120           LDA   INTCNT  ; KESME SAYICIYI AL
00051  2041  3D               DCR   A       ; AZALT
00052  2042  320120           STA   INTCNT  ; SAYICIYI SAKLA
00053  2045  C27920  >        JNZ   EXIT2   ; INTCNT SIFIR MI?
00054  2048  3A0020           LDA   LPCNT   ; SERVİS VERİLECEK
                                             ; DÖNGÜNÜN
00055  204B  3D               DCR   A       ; NUMARASINI AL VE BU
                                             ; DÖNGÜYE SERVİS
00056  204C  CC0028  >        CZ    L00P0   ; VERECEK YORDAMI ÇAĞIR
00057  204F  3D               DCR   A
00058  2050  CC6C28  >        CZ    L00P1
00059  2053  3D               DCR   A
00060  2054  CC7428  >        CZ    L00P2
00061  2057  3D               DCR   A
00062  2058  CC7C28  >        CZ    L00P3
00063  205B  3D               DCR   A
00064  205C  CC8428  >        CZ    L00P4
00065  205F  3D               DCR   A
00066  2060  CC8C28  >        CZ    L00P5
00067  2063  3D               DCR   A
00068  2064  CC9428  >        CZ    L00P6
00069  2067  3D               DCR   A
00070  2068  CC9C28  >        CZ    L00P7
00071  206B  7                RAL           ; EN BÜYÜK DEĞERLİKLİ BİT
                                             ; ELDEYE
00072  206C  DA7420  >        JC    EXIT1   ; ELDE VARSA DÖNGÜ
                                             ; BULUNDU, YOKSA
00073  206F  3E01           MVI   A, 01H  ; ÇIK, LPCNT'Yİ RESETLE
00074  2071  320020           STA   LPCNT
00075  2074  3E14    EXIT1   MVI   A, 14H  ; INCTNT'Yİ 20 YAP
00076  2076  320120           STA   INTCNT
00077  2079  F1      EXIT2   POP   PSW     ; BÜTÜN KAYDEDİCİLERİ
                                             ; YENİDEN YÜKLE
00078  207A  E1               POP   H
00079  207B  D1               POP   D
00080  207C  C1               POP   B
00081  207D  FB               EI
00082  207E  C9               RET           ; ANA PROGRAMA DÖN
00083
00084          20C8    >      ORG   20C8H   ; KESME 6.5'UN ATLAMA
00085  20C8  C33A20    >      JMP   INTR    ; ADRESİNİ SET EDER
00086
00087
00088                          ; SICAKLIK KONTROL EDİCİYE SERVİS VEREN YORDAM
00089
```

Şekil 12.25 (Devam ediyor)

```
00090          2800  >          ORG   2800H
00091 2800     3A0220    L00P0   LDA   DCHI   ; ISITICI AÇIK SAYICISINI AL
00092 2803     3D               DCR   A
00093 2804     320220           STA   DCHI   ; AZALTILAN SAYIYI SAKLA
00094 2807     C25F28    >      JNZ   EXIT   ; ANA PROGRAMA DÖN
00095 280A     3E01             MVI   A, 01H ; DCHI'YI ISITICI KAPALI
00096 280C     320220           STA   DCHI   ; DEĞERİNDEN GEÇECEk
                                            ; ŞEKİLDE.
00097 280F     3E80             MVI   A, 80H ; AYARLA.
00098 2811     D32A             OUT   2AH
00099 2813     3A0320           LDA   DCLO   ; ISITICI KAPALI SAYICISINI AL
00100 286      3D               DCR   A
00101 2817     320320           STA   DCLO   ; AZALTILAN SAYIYI SAKLA
00102 281A     C25F28    >      JNZ   EXIT   ; ANA PROGRAMA DÖN
00103 281D     0600             MVI   B, 00H ; KANAL ADRESİNİ B'DE SAKLA
00104 281F     CDA428    >      CALL  ADRD   ; A – D ÇEVİRMESİNİ YAP
00105 2822     E6FE             ANI   0FEH  ; ALT BİTİ DÜŞÜR
00106 2824     0F               RRC         ; İKİYE BÖL
00107 2825     320420           SIA   HOT    ; SICAKLIĞI SAKLA
00108 2828     4F               MOV   C, A   ; ÇEVİRME İÇİN YÜKLE
00109 2829     CDC728    >      CALL  CNVT1  ; GÖSTERME İÇİN BCD'YE
00110 282C     78               MOV   A, B   ; ÇEVİR
00111 282D     CD6E03           CALL  UPDDT  ; SICAKLIĞI GÖSTER
00112 2830     3A0420           LDA   HOT    ; SICAKLIĞI YENİDEN AL
00113 2833     47               MOV   B, A   ; B İÇİNDE SAKLA
00114 2834     3A0520           LDA   STPNT  ; AYAR NOKTASI SICAKLIĞINI
                                            ; YÜKLE.
00115 2837     90               SUB   B      ; SICAKLIK HATASINI BUL
00116 2838     57               MOV   D, A   ; HATAYI SAKLA
00117 2839     CA5528    >      JZ    DONE   ; ISITICIYI KAPALI TUT VE HATA
00118 283C     FA5528    >      JM    DONE   ; SIFIR İSE ÇIK
00119 283F     016400           LXI   B, 0064H ; DEĞİLSE DCLO'YU HESAPLA
00120 2842     CDD828    >      CALL  DIV1   ; 0064H'Yİ HATA İLE BÖL
00121 2845     79               MOV   A, C   ; BÖLÜM DCLO DEĞERİDİR.
00122 2846     320320           STA   DCLO
00123 2849     3E04             MVI   A, 04H ; DCHI'YI 4 DÖNGÜ İÇİN
                                            ; AYARLA
00124 284B     320220           STA   DCHI
00125 284E     3E00             MVI   A, 00  ; ISITICIYI AÇ.
00126 2850     D32A             OUT   2AH
00127 2852     C35F28    >      JMP   EXIT
00128 2855     3E01      DONE   MVI   A, 0H  ; DCHI İÇİN
00129 2857     320220           STA   DCHI   ; GEÇİŞ DEĞERİ
00130 285A     3E7F             MVI   A, 7FH ; DCLO İÇİN UZUN
00131 285C     320320           STA   DCLO   ; KAPALI DEĞERİ
00132 285F     3E02      EXIT   MVI   A, 02H ; BİR SONRAKİ DÖNGÜYÜ
                                            ; GÖSTER.
00133 2861     320020           STA   LPCNT
00134 2864     3E14             MVI   A, 014H ; INT CNT'Yİ 20'YE AYARLA
00135 2866     320120           STA   INTCNT
00136 2869     3EFF             MVI   A, 0FFH ; KESME YORDAMINA
00137 286B     C9               RET         ; DÖNÜŞTE DÖNGÜDEN GEÇİŞ
00138                                       ; KONTROL EDİCİSİ
```

Şekil 12.25 (Devam ediyor)


```
00139          ;*************************************************
00140          ; BURADA SAHTE DÖNGÜLER VAR
00141          ;*************************************************
00142
00143 286C     3E03      L00P1   MV1   A, 03H ; BİR SONRAKİ DÖNGÜYÜ
                                            ; GÖSTER.
00144 286E     320020           STA   LPCNT
00145 2871     3EFF             MVI   A, 0FFH ; DÖNGÜDEN GEÇİŞ ALGILAYICI
00146 2873     C9               RET         ; AYARLA
00147
00148 2874     3E04      L00P2   MVI   A, 04H ; BİR SONRAKİ DÖNGÜYÜ
                                            ; GÖSTER
00149 2876     320020           STA   LPCNT
00150 2879     3EFF             MVI   A, 0FFH
00151 287B     C9               RET
00152
00153 287C     3E05      L00P3   MVI   A, 05H ; BİR SONRAKİ DÖNGÜYÜ
                                            ; GÖSTER.
00154 287E     320020           STA   LPCNT
00155 2881     3EFF             MVI   A, 0FFH
00156 2883     C9               RET
00157
00158 2884     3E06      L00P4   MVI   A, 06H ; BİR SONRAKİ DÖNGÜYÜ
                                            ; GÖSTER.
00159 2886     320020           STA   LPCNT
00160 2889     3EFF             RVI   A, 0FFH
00161 288B     C9               RET
00162
00163 288C     3E07      L00P5   MVI   A, 07H ; BİR SONRAKİ DÖNGÜYÜ
                                            ; GÖSTER
00164 288E     320020           STA   LPCNT
00165 2891     3EFF             MVI   A, FFH
00166 2893     C9               RET
00167
00168 2894     3E08      L00P6   MVI   A, 08H ; BİR SONRAKİ DÖNGÜYÜ
                                            ; GÖSTER
00169 2896     320020           STA   LPCNT
00170 2899     3EFF             MVI   A, 0FFH
00171 289B     C9               RET
00172
00173 289C     3E01      L00P7   MVI   A, 01H ; BİR SONRAKİ DÖNGÜYÜ
                                            ; GÖSTER
00174 289E     320020           STA   LPCNT
00175 28A1     3EFF             MVI   A, FFH
00176 28A3     C9               RET
00177          ;
00178          ; A/D ÇEVİRİCİYİ KONTROL EDEN YORDAM
00179          ;
00180                ; PORT 29H A'DAN D'YE VERİ
00181                ; PORT 2AH BİT 7 ISITICI, BİT 5 ÇEVİRMEYİ BAŞLAT
00182                ; BİT 4 ALE, 2,1,0 BİTLERİ KANAL ADRESİ
00183 28A4     3E80      ADRD    MVI   A, 80H ; ISITICI KAPALI İÇİN KONTROL
```

Şekil 12.25 (Devam ediyor)

```
00184 28A6 B0          ORA   B       ; KANAL ADRESİ İLE BİRLEŞTİR
00185 28A7 D32A        OUT   2AH
00186 28A9 3E90        MVI   A, 90H  ; ALE'Yİ GÖNDER
00187 28AB B0          ORA   B       ; KANAL ADRESİNİ TUT
00188 28AC D32A        OUT   2AH
00189 28AE 3EB0        MVI   A, 0B0H ; ÇEVİRME BAŞLAMAYI GÖNDER
00190 28B0 B0          ORA   B       ; KANAL ADRESİNİ YÜKSEK TUT
00191 28B1 D32A        OUT   2AH
00192 28B3 3E80        MVI   A, 80H  ; ALE VE BAŞLAMAYI KAPAT
00193 28B5 B0          ORA   B       ; KANAL ADRESİNİ TUT
00194 28B6 D32A        OUT   2AH
00195 28B8 DB2B  EOCL  IN    2BH     ; ÇEVRİM BİTİNİN ALÇAĞA
00196 28BA 0F          RRC           ; GİTMESİNİ BEKLE
00197 28BB DAB828 >    JC    EOCL    ;
00198 28BE DB2B  EOCH  IN    2BH     ; ÇEVRİM BİTİNİN ALÇAĞA
00199 28C0 0F          RRC           ; GİTMESİNİ BEKLE
00200 28C1 D2BE28 >    JNC   EOCH
00201 28C4 DB29        IN    29H     ; ANALOGTAN SAYISALA
00202 28C6 C9          RET           ; VERİYİ OKU VERİYİ
                                     ; VERİYİ AKÜMÜLATÖRDE SAKLA
00203
00204
00205                  ; İKİLİK SAYIYI BCD'YE ÇEVİREN YORDAM
00206                  ; BU YORDAM C KAYDEDİCİSİ İÇİNDEKİ 8 BİTLİK İKİLİK
00207                  ; SAYIYI B KAYDEDİCİSİ İÇİNDEKİ İKİ BCD SAYIYA ÇEVİRİR.
00208                  ; BCD ARALIĞI 00-99 ARASINDADIR.
00209
00210
00211 28C7 1609  CNVT1 MVI   D, 09H  ; BİT SAYICIYI YÜKLE
00212 28C9 AF          XRA   A       ; A'YI SİL
00213 28CA 47          MOV   B, A    ; B'Yİ SİL
00214
00215 28CB AF    CVNT2 XRA   A       ; A'YI SİL
00216 28CC 15          DCR   D       ; BİT SAYICIYI AZALT
00217 28CD C8          RZ            ; BİTMİŞSE DÖN
00218 28CE 79          MOV   A, C    ; EN BÜYÜK DEĞERLİKLİ BİTİ
                                     ; ELDEYE KAYDIR
00219 28CF 17          RAL           ; KAYMIŞ SAYIYI C İÇİNDE SAKLA
00220 28D0 4F          MOV   C, A    ; BCD'Yİ A'YA TAŞI
00221 28D1 78          MOV   A, B    ; A'YI İKİ İLE ÇARP, C'DEN
                                     ; GELEN ELDEYİ TOPLA
00222 28D2 8F          ADC   A       ; BCD ŞEKLİNDE TUT
00223 28D3 27          DAA           ;
00224 28D4 47          MOV   B, A    ; BCD'Yİ SAKLA
00225 28D5 C3CB28 >    JMP   CNVT2   ; BİTİNCEYE KADAR DÖN
00226
00227
00228                  ; 16 BİTLİK SAYIYI 8 BİTLİK SAYIYA BÖLEN YORDAM
00229
00230
00231                  ; BU YORDAM B VE C KAYDEDİCİLERİ İÇİNDEKİ 16 BİTLİK
00232                  ; SAYIYI D KAYDEDİCİSİ İÇİNDEKİ 8 BİTLİK SAYI İLE BÖLER
```

Şekil 12.25 (Devam ediyor)


```
00233                  ; BÖLÜM C İÇİNDE VE KALAN B İÇİNDE GERİ DÖNER
00234
00235 28D8 1E09  DIV1  MVI   E, 09H  ; BİT SAYICIYI YÜKLE
00236 28DA 78          MOV   A, B    ;
00237 28DB 47    DIV2  MOV   B, A    ; ALT BÖLMENİN SONUCUNU
                                     ; SAKLA
00238 28DC 79          MOV   A, C    ; C'NİN EN BÜYÜK DEĞERLİKLİ
                                     ; BİTİ ELDEYE
00239 28DD 17          RAL           ; DAHA SONRA B'NİN EN
                                     ; KÜÇÜK DEĞERLİKLİ BİTİNE
                                     ; TAŞI
00240 28DE 4F          MOV   C, A    ; KAYDIRILMIŞ SONUCU SAKLA
00241 28DF 1D          DCR   E       ; BİT SAYICIYI AZALT
00242 28E0 CAED28 >    JZ    DIV3    ; BİTTİYSE ÇIK
00243 28E3 78          MOV   A, B    ; C'NİN EN BÜYÜK DEĞERLİKLİ
                                     ; BİTİ ELDE DE
00244 28E4 17          RAL           ; ELDEYİ B'NİN EN KÜÇÜK
                                     ; DEĞERLİKLİ BİTİNE ÇEVİR
00245                                ; 16 BİTLİK KAYMAYI BİTİR
00246 28E5 92          SUB   D       ; BÖLÜNENİ KAYAN SONUÇTAN
00247                                ; ÇIKAR
00248 28E6 D2DB28 >    JNC   DIV2    ; EVET, BİR SONRAKİ BİTİ
                                     ; BULMAK İÇİN GİT
00249 28E9 82          ADD   D       ; BÖLEN ÇOK BÜYÜK, GERİ
                                     ; TOPLA
00250 28EA C3DB28 >    JMP   DIV2    ; BİR SONRAKİ DÖNGÜYÜ YAP
00251 28ED 3EFF  DIV3  MVI   A, 0FFH ; BÖLÜMÜN TÜMLEYENİNİ AL
00252 28EF A9          XRA   C
00253 28F0 4F          MOV   C, A    ; ÇIKIŞ İÇİN TAŞI
00254 28F1 C9          RET           ; KESME YORDAMINA DÖN
00255                  END
```

Şekil 12.25 (Devam ediyor)

160 ms saniye sonra *DÖNGÜ 0* yeniden çağrıldığında DCHI azaltılır. Eğer DCHI 0'a kadar azalmamışsa program *DÖNGÜ 0*'dan çıkar ve INTR'ye döner. Eğer DCHI 0'a kadar azalmışsa ısıtıcı kapatılır ve DCLO azaltilır. *DÖNGÜ 0* 'a servis verildiği her zaman (160 ms saniye) DCLO bir azaltılır, bu durum DCLO 0 oluncaya kadar devam eder. *DCLO 0*'a ulaştığında yeni bir analogdan sayısala çevirme yapılır ve DCLO için geri besleme değeri yeniden hesaplanır. Bu kontrol döngüsü için geri besleme sıcaklık hatasından (ΔT) hesaplanan DCLO değeridir.

Buradaki önemli nokta, geri beslemeyi belirleyen program parçalarının küçük olması ve programın geri kalan kısmını etkilemeden kolaylıkla değiştirilebilir olmasıdır. Kullanılan basit oransal geribeslemenin yanısıra türevsel ve integral geribesleme üreten yordamlarda eklenebilir. Bunun için DCHI ve DCLO değerleri, ve ayar noktası yaklaştıkça bu değerlerin değişim hızları değiştirilebilmelidir. Belirli bir sistemde en iyi sonuçları elde etmek için denemeler sonucu geribesleme bölümü değiştirilebilir.

ADRD: ADC0808 veri toplama sistemini kontrol eden yordam ADRD yordamıdır. Sayısallaştırılacak kanal numarası *B* kaydedicisinin 3 düşük bitinde ADRD'ye geçer. Buentegre gönderilen diğer kontrol kelimeleri ile VEYA işlemine sokulur. Bu yordam içindeki komutlar, şekil 12.24'te gösterilen kontrol sinyallerini yaratırlar ve çevrim tamamlandığı zaman ikilik veriyi okurlar. Sonuç akümülatör içinde, çağıran programa döner.

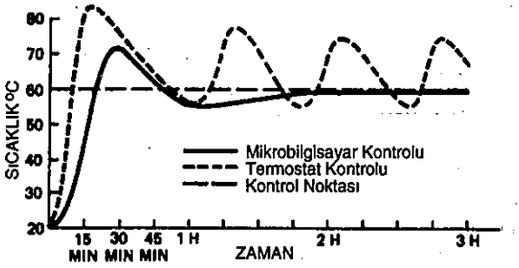
CNVT1: *C* içindeki 8-bit ikilik sayıyı *B* içinde iki BCD rakama çeviren yordam CNVT1 yordamıdır. Yordamın çalışması daha önce terazi örneğinde açıklanmıştır. Yordamdan dönen maksimum değer 99 dur. 99'dan büyük sıcaklık değeri için önde bir 1 sayısı olduğu varsayılır, fakat gösterilmez.

DIV1: DIV1 yordamı *B* ve *C* içindeki 16 bitlik sayıyı akümülatör içindeki 8 bitlik sayı ile böler. Yordam sola kaydır ve çıkart yöntemini kullanır. Bölüm, *C* kaydedicisi içinde ve kalan *B* kaydedicisi içinde çağıran programa döner.

ÖZET VE SONUÇ

Şekil 12.26 mikrobilgisayar kontrolu ve geleneksel termostat aç-kapa kontrolu ile elde edilen sıcaklık-zaman tepkilerinin karşılaştırılmasını göstermektedir. Her iki sistemde görülen başlangıçtaki aşırı taşma ve termostat eğrisinde görülen büyük salınımlar kullanılan sıcak plakanın büyük sıcaklık ataletinden dolayıdır. Mikrobilgisayar kontrollu sistemdeki başlangıç taşması ve yaklaşık 1°C 'lik kalıntı hatası daha karmaşık geri besleme hesaplama formülü kullanarak kolaylıkla ortadan kaldırılabilir.

Hesaplanmış geri besleme kullanmanın avantajları açıktır. Bu avantajlardan dolayı daha çok endüstriyel ve araştırma aleti bilgisayar gücünü kullanmaktadır. Kullanılan bilgisayarlar, 8 veya 16 işlem kontrol döngüsünü gözleyen programlanabilir küçük kontrolörlerden, bir fabrikadaki bütün işlemleri kontrol eden oda büyüklüğündeki bilgisayarlara kadar değişir.



Şekil 12.26 Mikrobilgisayar kontrollu sistem ve termostat kontrollu sistemin sıcaklık-zaman tepkilerinin karşılaştırılması

TEKRAR SORULARI VE PROBLEMLER

1. *Mikroişlemci-tabanlı* bir aleti öğrenmenin mantıklı adımlarını sayınız.

2. *Monitor programını tarif* ediniz.

3. *SDK-85 tabanlı* terazide ağırlık verisini *ADC*'den alma işlemini anlatınız.

4. *Modüler* programlamayı tarif ediniz. Modüler programlamanın *iki avantajını* belirtiniz.

5. *Eşitlemeleri, tampon,* paketli BCD, ve tamsayı *çarpmayı* tarif ediniz.

6. Terazi programı içinde *KYBINT* yordamının işlevini anlatınız.

7. Terazi programı içindeki çarpma sizce neden *ardışık toplama* ile yapılmamıştır?

8. Terazi programı sonundaki *"3-S"* gecikme yordamının gerçek gecikmesini hesaplayınız. Saat frekansını 3MHz kabul ediniz.

9. Terazinin gösterme yordamında kullanılan başvuru tablosu tekniğinin *ASCII karakterleri EBCDIC karakterlere* çevirmekte nasıl kullanılabileceğini anlatınız. Bu kodlar için Tablo 3.2'ye bakarak bu çevirme için başvuru tablosunun ilk beş elemanını yazınız.

10. *BCDCVT* tarafından *BCD* sayıyı ikilik eşdeğerine çevirmek için kullanılan işlemi açıklayınız.

11. *İki tane ikilik* sayıyı mikroişlemci ile çarpmanın *üç metodunu* kısaca anlatınız.

12. *DAD H Komutu* yürütülmeden önce eğer *H* ve *L* kaydedicileri 034A içeriyorsa, komut yürütüldükten sonra *H* ve *L* hangi sayıları içerecektir? İkilik bit dizisinin bir sola kaydığını gösteriniz.

13. *EPROM programlayıcı* için kullanılan *MEK6800D2* mikrobilgisayar kartının bellek haritasını çiziniz.

14. *ZIF* soketi ve bilgi istem karakterini tarif ediniz.

15. *MEK6800D2* üzerindeki *E* tuşu niçin *6800 CPU*'nun girişine bağlanmıştır.

16. Şekil 12.15'e bakınız.

    a. PGM darbesinin minimum ve maksimum genişlikleri nedir?

    b. PGM darbesi düşük seviyeye gittikten sonra veri ne kadar süre ile geçerli tutulmalıdır?

    c. PGM darbesi yüksek seviyeye gitmeden önce veri nekadar süre ile geçerli olmalıdır?

17. Şekil 12.16 ve 12.17'ye bakınız.

    a. EPROM programlayıcı programının üç temel kısmını anlatınız.

    b. Cihazın programlanmadan önce niçin bütün 1'ler için kontrol edildiğini açıklayınız.

    c. 2716 nasıl silinir, açıklayınız.

    d. Kullanıcı tarafından yüklenen başlangıç adresi ve kaynak adresi niçin tamponlara kopyalanır?

    e. Dizinli adreslemeyi tarif ediniz. PIA'ları ilk kullanıma hazırlamak için genişletilmiş adreslemeden niçin daha etkilidir?

    f. 614KHz'lik saat kabul ederek 50 m saniyelik gecikme döngüsünün gerçek gecikmesini hesaplayınız.

    g. Program onaylaması sırasında çevirici dilde oluşan bir hatanın, altı F yerine hata adresinin gösterilmesi ile nasıl belirtileceğini gösteriniz.

    h. Monitor içindeki OUTDS yordamının bir tuş basıldığında kontrolu nasıl monitöre verdiğini anlatınız.

    i. Program içinde PGM darbesi gönderilmeden önce niçin bir NOP komutu yazılmıştır?

18. Bir çıkış portunun D0 hattına bir katı hal röle bağlanmıştır. Çıkış hattındaki yüksek seviye, röleyi iletime geçirir ve bir kapı kilidini açan solenoid üzerinde büyük bir akım akmasını sağlar. 8080A/8085A veya 6800 komutlarını kullanarak 3 saniye içinde üç onaltılık karakter dizisi girildiğinde kilidi açan bir çevirici dil programı yazınız.

REFERANSLAR/BİBLİOGRAFYA

Intel Memory Design Handbook, Intel Corporation Santa Clara, CA, 1981.

M6800 Programming Reference Manual, M68PRM(D), MotorolaSemiconductor Products, Inc., Phoenix, AZ, 1976.

Intel MCS-80/85B User's Manual, Intel Corporation, Santa Clara, CA, January 1979.

8080/8085 Assembly Language Programming Manual, Intel Corporation, Santa Clara, CA, 1978.

M6800 Microcomputer System Design Data, Motorola Semiconductor Products, Inc. Phoenix, AZ, 1976.

Intel SDK-85 User's Manual, Intel Corporation, Santa Clara, CA, December 1977.

MEK6800D2 Manual, Motorola Semiconductor Products, Inc., Austin, Texas, 1976.

BÖLÜM 13

MİKROİŞLEMCİ TABANLI SİSTEMLERİN TEMEL ÖRNEKLERİNİN GELİŞTİRİLMESİ VE ARIZA GİDERİLMESİ

Bu bölüm iki kısımdan oluşmaktadır. Birinci bölümde mikroişlemci tabanlı bir cihazın geliştirilmesi sırasında donanım ve yazılımın geliştirilmesi ve hatadan ayıklanması için kullanılan yöntemleri tartışacağız. İkinci bölümde cihazın üretilmesi için gerekli adımları ve mikroişlemci tabanlı cihazın üretimi sırasında ya da servisi sırasındaki arızaları gidermekte kullanılan alet ve teknikleri anlatacağız.

Bu bölümün sonunda;

1. Geliştirme kartları üzerinde *ROM, RAM* ve *I/O* portların test edilmesi ve hatadan ayıklanması adımlarını tanımlayabileceksiniz.
2. Durdurma noktaları ve *tek-adım kullanarak,* geliştirme kartları ile programları test etme ve *hatadan ayıklama* adımlarını tanımlayabileceksiniz.
3. *Lojik zaman/durum analizleyici ile mikroişlemci, ROM, RAM* ve *I/O* portlarının test edilmesi ve hataların ayıklanması yöntemlerini tanımlayabileceksiniz.
4. Bir mikroişlemci tabanlı elemanla hata ayıklanması için bir mikroişlemci geliştirme sistemi ve *devre-içi emülatör kullanımını* tanımlayabileceksiniz.
5. Bir elemanın tasarımından, bitirilmiş elemana kadar olan akışı ve her adım-da çalışan teknisyenlerin fonksiyonlarını tanımlayabileceksiniz.
6. *Bir mikroişlemci tabanlı* elemandaki *arızanın giderilmesi* için işaret analizinin nasıl kullanıldığını tartışabileceksiniz.

Bir Temel Örnek Yapılması ve Hata Ayıklanması

Genel Yorumlar

Bölüm 1'de yapı hakkında tartışılan bazı noktaların burada tekrarlanması ve genişletilmesi gerekmektedir.

Güç Kaynakları : Güç kaynaklarını büyük ölçekli entegre devrelere bağlamadan önce çıkışlarına bir osiloskop (hafızalı olursa iyi olur) bağlayıp açılma anındaki geçici gerilimleri gözleyiniz. Bazı güç kaynakları, ani geçici gerilimlerine sahiptir ve bu MOS LSI devrelerini bozar. Eğer değişken bir kay-

nak kullanıyorsanız, bağlamadan önce doğru gerilime ayarlandığına emin olunuz.

Fiziksel Yapı ve Telle Bağlama : Bir temel örnekte birçok takıp çıkarmaya dayanabilecek iyi kalitede entegre devre soketleri kullanınız. Kötü yapılmış soketler, problemleri çözmekten çok daha fazla problem yaratırlar.

Seramik paketli entegre devreleri elden geçirirken dikkatli olunuz. Bunlar pahalıdır ve düştükleri zaman kolayca ikiye bölünür.

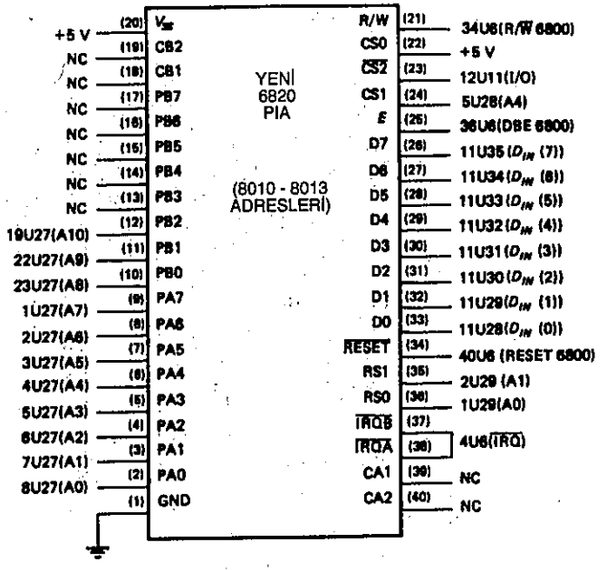
Bölüm 1'de tartışıldığı gibi atlama kondansatörlerinin çok fazla olması gibi bir durum yoktur. Uçların karta girdiği yerlerde kaynak gerilim uçları ile toprak arasında 15-50µF elektrolitik kondansatör ile 0.1µF disk kondansatörleri paralel bağlamayı unutmayınız.

Renk-kodlu Telle Bağlama: Eğer tel-sargılı bağlantı kullanıyorsanız ve renk kodlaması mümkün değilse, kodlayabileceğiniz kadar kodlama yapınız. Özellikle tel-sargılı bağlantı yapıyorsanız, Şekil 13.1'dekine benzer bir telle bağlama diyagramı normal bir şematikten daha az hata üretir. Hangi bağlantıyı yaptığınızı takip etmenin yolu her bağlantı yapıldığında üzerine şeffaf işaret kalemi ile işaret koymaktır. Entegre devreleri ve diğer çok bacaklı elemanları çizim üzerinde numaralayarak belirtiniz ve temel örnek üzerinde bunlara karşı gelen cihazları etiketleyiniz.

Telle bağlama tamamlandığında, her entegre devrenin bacak bağlantılarını iki defa kontrol ediniz. Bir kere yüksek bacaktan düşük bacağa doğru ve bir kerede düşük bacaktan yüksek bacağa doğru kontrol ediniz. Her bacağın doğru bağlandığına emin olunuz. Bu size aşırı gelebilir, fakat bir temel örneğin çalışmamasına, tasarım hatalarından sonra en çok telle bağlama hataları sebep olmaktadır.

Temel Örnek (Örnek Devre) Donanımın Yapılması ve Test Edilmesi

Bir temel örneği hızla inşa etmek için birimin tamamını ya da büyük bir kısmını inşa edip gücü uygulama ve çalışmasını ümit etme eğilimi vardır. Hataların veya yanlış çalışmaların birbirlerini etkilemelerinden dolayı bu büyük parçada arıza giderilmesi çok zor ve zaman alıcı olabilir. Uzun dönemde bir temel örneği geliştirme ve çalıştırmanın en hızlı yolu, donanım ve yazılımı her biri tek başına inşa edilebilir ve test edilebilir küçük modüller halinde inşa etmektir. Bölüm veya modüller başlangıçta küçük inşa edilirlerse hataları genelde daha kolay ortadan kaldırılabilir. Bölümlerin hataları ayıklandıkça daha büyük bölümler oluşturmak için birbirleriyle bağlanır ve bu bölümler tüm ünite biraraya getirilinceye kadar test edilir ve hataları ayıklanır.



Şekil 13.1 : Telle Bağlama Sırasındaki Hataları Azaltmak için Telle Bağlama Diyagramı

TEMEL CPU MODÜLÜ : Mikroişlemci tabanlı bir aletin temel örneğinin geliştirilmesini basitleştirmek için, temel örnek Bölüm 12'deki örneklerde kullanılan SDK-85 veya MEK6800D2 gibi hazır geliştirme kartları üzerine kurulabilir. Bu tip kartlar hemen hemen bütün işlemciler için mevcuttur.

Bunların çoğu, programın *RAM*'lara girilip çalıştırılmasını sağlayan ve kart üzerindeki *ROM* içinde bulunan monitor programlarına sahiptir. Bazıları kart üstünde tuştakımına ve yedi parçalı göstergelere sahiptir. Diğer kartlar haberleşme için teletayp ya da CRT terminaller gerektirirler. Ana saat, *CPU* ve monitor *ROM* entegre devrelerine ilaveten, çoğu kart birkaç yüz ya da bin baytlık *RAM*'a, iki veya daha fazla *I/O* port entegre devrelerine ve bir seri *I/O* port entegre devresine sahiptir. Daha fazla *RAM, ROM* ve *I/O* port için kart üzerinde soket yerleri ilave edilebilir. Kullanıcının ekleyeceği devreler için genelde bir telle bağlantı bölgesi ayrılır.

Birçok mikroişlemci geliştirme kartı üretici tarafından birleştirilmiş ve test edilmiş olarak gelir. SDK-85 ve MEK6800D2 gibi kit halinde gelenler ise kolaylıkla birleştirilebilir. Eğer onları dikkatli bir şekilde inşa ederseniz genelde ilk seferde çalışırlar. Buradaki ana nokta, bu kartların temel örneklerin az

problemlerle elde edilmesini sağlamalarıdır. Daha sonra hazır bir kart kullanmadan bir CPU modülünün nasıl geliştirileceğini tartışacağız. Burada, bu geliştirme kartlarından birini kullanarak temel örneğin nasıl inşa ve test edildiğini ve hataların nasıl ayıklandığını göstereceğiz.

RAM VE ROM MODÜLLERİ İLAVE ETMEK : *RAM* ve *ROM*'ları ilave ederken soketleri, adres ve veri hatlarının birbirlerine kolayca papatya zinciri olarak bağlanabilecekleri şekilde yerleştiriniz. Her entegre devrenin yanına, V_{cc} ile toprak arasına bir 0.1μF atlama kondansatörü ilave ediniz. Bütün bağlantıları dikkatle kontrol ediniz. Daha sonra bütün pahalı bellek entegre devrelerini yerlerine takmadan önce, bütün tellerin bağlanmaları gereken yerlere bağlandıklarını bir ohmmetre ile kontrol ediniz. Özellikle *RAM*'larda bir sıra boyunca adres hatlarını telle papatya zinciri olarak bağlarken teli bir bacak kaydırmak çok kolaydır. Bütün bunlar aşırı gelebilir, fakat yalnızca birkaç dakika alır ve genellikle bellekler çalışırken bulunması daha uzun süren hataları ortaya çıkarır.

Belleklere güç uygulandıktan sonraki ilk iş onları test etmektir. *ROM*'lar için bilinen bir program kullanılabilir. Monitor, bellek kontrol etme komutu ile okunur ve bilinen programın listesi ile karşılaştırılır. Her adresin kontrol edilmesi gerekmez, fakat bütün adres hatlarının doğru çalıştığına emin olabilecek yeterlikte adres rasgele kontrol edilmelidir. Eğer *ROM* okunamazsa, bölümün RAM'lar hakkındaki bir sonraki kısmı problemin nasıl çözüleceğini tartışacaktır.

RAM'lar için hem okuma hemde yazma fonksiyonları test edilmelidir. Bunun kolay yolu monitor programının hepsini ya da bir kısmını test edilecek RAM içine kopyalayan kısa bir program yazmaktır. Şekil 13.2, bu işlemi bir 8080A/8085A sistemi için yapan bir yordam göstermektedir. Yordam, monitor programından bir bayt okur ve onu *RAM* içindeki bir adrese yazar.

HL ve *DE* göstericileri, belleklerdeki bir sonraki adresleri göstermek için arttırılır ve bayt sayıcısı azaltılır. Bütün adresler yazıldıktan sonra göstericiler yeniden kullanıma hazırlanır. Daha sonra *RAM* içindeki her adres okunur ve Monitor *ROM*'u içinde kendisine karşı gelen adres ile karşılaştırılır. Eğer iki bayt eşit değilse program çıkış komutu RST1'e atlar. Eğer baytlar eşleşirse, program bütün adresler kontrol edilinceye kadar devam eder ve daha sonra kontrol monitor programına döner. SDK-85 için RST1 komutu bütün kaydedicilerin içeriklerini yığının üzerine saklar ve kontrolu monitor programına verir. Daha sonra kaydedici kontrol komutu kullanılarak kaydedicilerin içerikleri okunur ve göstergeler üzerinde gösterilir. Örneğin Şekil 13.2'de, yürütmeden sonra *H* ve *L* kaydedicileri monitor programından okunan son adresi ve *D* ve *E* kaydedicileri de *RAM* içinde okunan son adresi gösterecektir. Eğer bu adres eklenen *RAM*'ın son adresi ile aynı ise, bu *RAM*'a yazma ve bu *RAM*'dan okuma başarılı olmuş demektir.

Eğer okuma-yazma işlemi başarılı değilse, bir sonraki adım bunun sebebini bulmaktır. Onaylama yordamının bir hata bulup çıkmadan ne kadar çalıştığı size problem için bir ipucu verecektir. Eğer program çıkmadan önce *RAM* adresinin çoğu boyunca çalışmışsa, kötü bir entegre devre ya da aktif olmayan bir üst seviyeli adres hattı problemin sebebi olabilir. Herhangi bir entegre devrede olduğu gibi, entegre devrenin numarasını doğru okuduğunuza emin olunuz. Örneğin bir 2114, 1Kx4 şeklinde organize edilmiş 4k bitlik statik RAM'dır ve 2141 ise 4K x1 şeklinde organize edilmiş 4k bitlik statik *RAM*'dır. Bu iki entegre devrenin birbiriyle değiştirilemiyeceği açıktır. Eğer program birinci baytların karşılaştırılmasından sonra çıkarsa, problem test programı ya da sinyallerin entegre devre *RAM*'larına erişmesi olabilir. Şekil 13.2'deki yazma-onaylama programı kart ile gelen ve daha önce test edilmiş *RAM*'lar üzerinde test edilebilir ve böylece bunun problem kaynağı olması ortadan kaldırılır. Bu ihtimalin ortadan kaldırılması ile bir sonraki adım *RAM*'ların uygun sinyal ve verileri alıp alamadıklarını test etmektir. Eğer Şekil 13.2'deki test programı içindeki JZ TEST komutu şartsız başlangıca atla (JMP START) komutu ile değiştirilirse program sonsuz bir döngü içine girecektir. Çalıştırıldığı zaman, monitor programını *RAM* içine defalarca kopyalamaya çalışacaktır. Bir osiloskop kullanarak adres, veri, yonga yetkilendirme, yazma sinyallerinin cihaza uygun olarak gelip gelmediklerini kontrol ediniz. Bir okuma işlemi sırasında *RAM* veya *ROM*'a gelen sinyallerin uygunluğunu test edebilmek için bir başka sonsuz döngü yazılabilir. Bu şekildeki donanım ve yazılım kombinasyonları problemin kaynağını bulmakta kullanılabilir.

I/O PORTLARI VE PROGRAMLANABİLİR ÇEVRE BİRİM ENTEGRE DEVRELERİ

İstenilen RAM ve ROM'lar ilave edilip, hataları ayıklandıktan sonra ilave I/O portları bağlanabilir ve bağlantılar bir ohmmetre ile kontrol edilebilir. Daha sonra her bir portu kontrol etmek için kısa bir program kullanılabilir. Eğer portlar için bir *PIA* kullanılıyorsa, program ilk olarak *PIA*'yı kullanıma hazırlamalıdır. Eğer bir çıkış portu olarak kullanıma hazırlanırsa, program ona bir bayt yazabilir. Daha sonra bir osiloskop kullanarak baytın çıkış hatlarına gelip gelmediğini kontrol edebilirsiniz. Eğer yazılan bayt çıkışa gelmemişse, bir atlama veya dallanma komutu ile porta-yazma programını sonsuz döngü haline getirebilirsiniz. Program baytı porta üstüste yazarken, siz bir osiloskop ya da lojik analizleyici kullanarak cihaza doğru adres, veri yetkilendirme sinyallerinin geldiğini kontrol edebilirsiniz. Eğer devreye doğru sinyaller geliyorsa ve devre hâlâ çalışmıyorsa, program içinde PIA'nın kullanıma hazır-lanması kısmını yeniden kontrol ediniz. Bazı programlanabilir çevrebirim entegre devrelerinin kullanıma hazırlanmaları için iki veya daha fazla kontrol kelimesi gönderilmesi gerektiğini hatırlayınız.

```
START:  LXI   B (BAYT SAYISI)              ; BAYT SAYICIYI AYARLA
        LXI   H, (MONİTOR İÇİN BAŞLANGIÇ ADRESİ)
        LXI   D, (RAM İÇİN BAŞLANGIÇ ADRESİ)
LOOP:   MOV   A,M                          ; MONİTOR'DAN BAYTI AL
        STAX  D                            ; RAM'A YAZ
        DCX   B                            ; BAYT SAYICIYI AZALT
        MOV   A,C                          ; HEPSİ YAZILDI MI?
        ORA   B                            ;
        JZ    TEST                         ; EVET, ONAYLAMAYA GİT
        INX   H                            ; MONİTOR İÇİNDEKİ BİR
                                           ; SONRAKİ ADRESİ GÖSTER
        INX   D                            ; RAM İÇİNDEKİ BİR
                                           ; SONRAKİ ADRESİ GÖSTER
        JMP   LOOP                         ; BİR SONRAKİ BAYTI YAZMAYA GİT
TEST:   LXI   B, (BAYT SAYISI)
        LXI   H, (MONİTOR İÇİNDEKİ BAŞLANGIÇ ADRESİ)
        LXI   D, (RAM İÇİNDEKİ BAŞLANGIÇ ADRESİ)
LOOP 2: LDAX  D                            ; RAM'I OKU
        CMP   M                            ; MONİTOR İLE KARŞILAŞTIR
        JNZ   EXIT                         ; YANLIŞ İSE ÇIK
        DCX   B                            ; BAYT SAYICIYI AZALT
        MOV   A,C                          ; BAYT SAYICI SIFIR MI?
        ORA   B                            ;
        JZ    EXIT                         ; BİTTİYSE ÇIK
        INX   H                            ; BİR SONRAKİ ADRESE GİT
        INX   D                            ;
        JMP   LOOP 2                       ; BİR SONRAKİ RAM BAYTINI OKU
EXIT:   RST1
```

Şekil 13.2 RAM'ın Bir Kısmını Test Etmek İçin 8080A/8085A Yordamı

Bir giriş portu da aynı şekilde test edilebilir. İlk kullanıma hazırlandıktan sonra port akümülatör içine okunur. Kaydedicileri kontrol komutu akümülatör içeriğini göstermek için kullanılır. Giriş hatlarından V_{cc} yada toprağa geçici olarak bağlanan dirençler, giriş port hatlarına bir veri baytı koymak için kullanılabilir. Portu defalarca okuyan bir başka sonsuz-döngü programı bir osiloskop kullanarak uygun adres, veri ve yetkilendirme sinyallerinin cihaza gelip gelmediklerini kontrol eder.

Diğer programlanabilir çevrebirim entegre devreleri de benzer şekilde eklenebilir ve hataları ayıklanabilir. Eğer bir cihaz veri çıkışı için kullanılıyorsa, bu cihazı ilk kullanıma hazırlamak ve cihaza veri göndermek için kısa bir program yazılabilir. Bir osiloskop ya da lojik analizleyici, çevrebirim entegre devresinin doğru veri verip vermediğini kontrol etmekte kullanılabilir. Eğer cihaz veri girişi için kullanılırsa, giriş sinyalleri dirençler ve darbe üreteçleri kullanarak elde edilebilir. Devre ilk kullanıma hazırlayacak ve buradan CPU'ya veri okuyacak kısa bir program yazılabilir. Burada okunan verinin doğru olup olmadığı kontol edilebilir.

DIŞ ARABİRİM DEVRELERİ : Dış arabirim devrelerini inşa etmek ve test etmekte kullanılan yöntemler inşa edilen devreye bağlıdır. Hatırlanması gereken önemli nokta devreleri mümkün olduğu kadar parçalar halinde inşa etmek ve hatalarını ayıklamaktır.

Örneğin, Bölüm 12'deki mikroişlemci tabanlı terazide ilk olarak yük hücresi kuvvetlendiricileri kurulmuş ve test edilmiştir. Kuvvetlendiriciler doğru olarak çalıştıktan ve kalibre edildikten sonra *MC14433* analogtan sayısala çevirici ilave edilmiştir. Basamak-strob çıkışı üzerinden tetiklenen bir osiloskop her basamak için BCD veri çıkışını kontrol etmekte kullanılmıştır.

Bir diğer örnek olarak Şekil 11.14'teki *MEK6800D2* gösterge modülü gibi bir dış göstergeye arabirim yapmanız gerektiğini varsayalım. İlk olarak gösterge parçalarını paralel olarak telle bağlayınız. *Ortak-katod LED* göstergeleri için bütün parça hatlarını 150Ω dirençler ile geçici olarak +5V'a bağlayınız. Her basamağı iki ortak katodunu topraklayarak test ediniz. Her *basamak bir "8"* göstermelidir. Daha sonra dirençleri parça uçlarından ayırın ve parça-sürücü transistörleri ilave edip, test ediniz. Bunlarda onaylandıktan sonra, basamak-sürücü entegre devrelerini ilave edip test ediniz. Gösterge donanımının hepsi doğru olarak çalıştıktan sonra bir çıkış portuna bağlayınız ve bir program yordamı ile göstergeye bir veri yazmayı deneyiniz. Bir test yordamı bütün parçaları ışıklandırabilir ve her basamağı 1ms ışıklandıracak şekilde dolaşabilir. Eğer programlayıcı gösterge tarama programı yazdıysa bu program bu adımda girilebilir ve test edilebilir.

Gösterge başarıyla çalıştığı zaman klavye *U10* kodlayıcıya bağlanabilir. Donanımdaki hatalar ayıklandıktan sonra tuş tarama yordamı girilebilir ve test edilebilir.

Yukarıdaki örnekler basit gibi görülebilir fakat devreleri kolaylıkla test edilebilen ve hata ayıklanabilen kısımlar halinde kurma ihtiyacı hiçbir zaman göz ardı edilemez.

PROGRAM MODÜLLERİNİN TEST EDİLMESİ, HATALARIN AYIKLANMASI VE BAĞLANMALARI

Program modüllerini test etmenin ve *hatalarını ayıklamanın birçok metodu* vardır. Bunlardan biri durdurma noktaları ve tek adım gözleme ile kaydedicileri kontrol etme komutlarını kullanmaktır. Bu teknik Bölüm 12'de terazi ve EPROM programlayıcı örneklerini geliştirmekte kullanıldı. İkinci bir metod lojik analizleyiciyi geniş bir şekilde kullanmaktadır. Güçlü fakat pahalı olan bir diğer metod Şekil 8.26'da gösterilen bir geliştirme sisteminin gelişmişini kullanmaktadır. Bu teknik, sık sık *devre içi emülasyonu* olarak adlandırılır. Takip eden bölümlerde bu üç metodu tartışacağız. İlk olarak Bölüm 12'deki ilk iki örneği geliştirmek için durdurma noktaları ve monitor komutlarının nasıl kullanıldığını göstereceğiz.

DURDURMA NOKTALARINI VE SDK-85'İN MONİTOR KOMUTLARINI KULLANARAK TERAZİ PROGRAMININ TEST EDİLMESİ VE HATA AYIKLANMASI

Bu metodu kullanmak için gerekli genel adımlar şu şekildedir. İlk olarak, başlamak için mantıklı ve kabul edilebilir büyüklükte bir program modülü seçiniz. Örneğin terazi programında başlanacak mantıklı modül veriyi analogtan sayısala çeviriciden alma kısmıdır. Uygun bir diğer kısımda gösterge yordamı *DSPLY* olabilir.

İkinci olarak seçilen kısma dikkatle bakarak hangi ilk kullanmaya hazırlama işlemlerinin gerekli olduğunu bulunuz. Gerekli girişleri ve üretilen çıkışları belirleyiniz. Modülün başına kendi kendine yeterli olabilmesi için birkaç adım ekleyebilirsiniz. Örneğin bir çarpma yordamını test edebilmek için yordamın giriş kaydedicilerini bilinen verilerle yükleyiniz böylece çıkışın doğru olup olmadığını kontrol edebilirsiniz. Örneğin terazi programı içindeki BCDCVT alt yordamı, *D* ve *E* içindeki paketli dört BCD rakamı 16 bitlik ikilik eşdeğerine dönüştürür. Yordamı test etmek için programın başına LXI D,1234H komutu eklenmelidir. *D* ve *E* içine yüklenen 1234H yordamın bitiminde *H* ve *L* içinde $04D2_{16}$ sayısını üretir.

Modül bittiğinde kontrolu monitor programına geri veren bir komut modülün sonuna konmalıdır. SDK-85 kartı için RST1 ya da CF_{16} komutu bu işi yapar. RST1 komutu program yürütmesini 0008H adresine gönderir. SDK-85 monitor programının bu adreste bütün kaydedicileri bellek içinde saklayan ve kontrolu monitore geri veren bir yordamı vardır. Daha sonra RST1 komutundan önce her kaydedicinin içerdiği veriyi gösterebilmek için kaydedici kontrol komutu kullanılabilir. Program akışındaki bu duruş ve kaydedicilerin saklanması ile kontrolun monitore dönmesi durma noktası olarak adlandırılır. Amacımızın program modülünü kendi kendine yeter şekilde sabitleştirmek olduğunu hatırlayınız. Test edilmek için tanımlanmış girişleri ve tahmin edilebilir çıkışları olmalıdır.

Üçüncü olarak program modülünün onaltılık kodlarını, monitor programının belleği değiştir komutunu kullanarak kart üstündeki RAM'a yükleyiniz. Bu SDK-85'te şu şekilde yapılmaktadır. Bir resetten sonra bellek değiştirme tuşuna basınız. Daha sonra tuş takımından birinci onaltılık adresi giriniz. Bu adresteki veriyi göstermek için NEXT tuşuna basınız. İstenilen onaltılık kodu girebilmek için istenilen tuşlara basınız. Bir sonraki adrese gitmek için NEXT tuşuna basınız. Bütün onaltılık program baytları girildikten sonra monitor kontroluna dönmek için EXEC tuşuna basınız.

Dördüncü olarak programı çalıştırınız. Bunu SDK-85 üzerinde yapmak için GO tuşuna basınız, program modülünün başlangıç adresini giriniz ve *yürütme (EXEC)* tuşuna basınız. Program parçası bittiğinde ve kontrol monitore dönünce, programın her kaydedici içinde doğru sonuçlar üretip üretmediğini kontrol etmek için kaydedici gözden geçirme komutunu kullanınız.

BİR PROGRAM MODÜLÜNDEKİ HATALARIN AYIKLANMASI

Çoğu zaman olduğu gibi, *program modülü doğru olarak çalışmazsa, hata ayıklama yöntemini başlatınız.* Eğer program modülü elle çevrilmişse her komut için doğru onaltılık kodun girildiğini kontrol ediniz. *Atlama adreslerini yeniden kontrol ediniz.* 8080A/8085A için önce düşük adres baytının ve daha sonra yüksek adres baytının belleğe yüklendiğini hatırlayınız. Doğrudan komutların ikinci veya ikinci ve üçüncü baytlarının yerlerinde olmasını garanti ediniz. Yığına itme komutlarının uygun sırada ve aynı sayıda yığından çıkarma komutları tarafından takip edilmesine dikkat ediniz. Eğer bir program yığından çıkarma komutlarından daha çok yığına itme komutlarına sahipse, yordam her çalıştığında yığın RAM içinde aşağı doğru büyüyecektir ve belkide RAM içindeki programın üzerine yazacaktır. Daha sonra RAM içindeki program ile yazılı versiyonu karşılaştırın ve program RAM içine yazılırken bir hata yapılmadığını ve hiçbir komutun dışarıda bırakıl-madığını kontrol ediniz. Eğer hatalar bulunursa düzeltilebilir ve modül yeniden denenebilir. Eğer hata bulunmazsa ve program hâlâ çalışmıyorsa bir sonraki adım test edilen program parçasının büyüklüğünü azaltmaktır.

Program modülünün büyüklüğü basit olarak program içinde daha önceki bir noktaya durma noktası koymakla küçültülebilir. Bunu yapmak için herhangi bir komutu *RST1*'in onaltılık kodu olan CF ile değiştiriniz. Şekil 12.6'daki terazi programının analogtan sayısala çeviriciden veriyi alma kısmına

bakınız. Bir durma noktası koymanın iyi bir yeri program çıkışındaki 001DH adresindeki *ANI* komutu yerine koymaktır. Eğer programın bu noktaya kadar olan kısmı doğru çalışırsa, durma noktasından monitore çıkış analogtan sayısala çeviriciden gelen en önemli basamağı akümülatöre yükleyecektir. Bu noktada üç temel durum ortaya çıkabilir. İlk olarak durma noktasından çıkışta akümülatörde doğru karakter olabilir. Bu durumda *ANI* komutu yerine konur ve CF bir sonraki küçük parçayı test etmek için program içinde daha sonraki bir noktaya yerleştirilir. İkinci olarak durma noktası çıkışında akümülatörde yanlış karakter olabilir. Bu durumda BCD ve strob hatlarının giriş portuna bağlanış sıralarını kontrol ediniz.

Eğer *RST1* komutu ile girilen durdurma noktaları problemi bulmanıza yardım etmezse, tek-adım komutunu deneyiniz. İsminden de anlaşılacağı gibi bu komut sizin programı her seferinde bir komut olarak yürütmenize izin verir. Tek adım komutunu SDK-85 üzerinde kullanmak için SINGLE STEP tuşuna basılır ve klavyeden başlangıç adresi girilir. *NEXT* tuşuna basıldığında bu adresteki komut yürütülür ve gösterge bir sonraki komut adresine arttırılır. Eğer *EXEC* tuşuna daha sonrada EXAM REG tuşuna basılırsa bütün kaydedicileri kontrol edebilirsiniz. SINGLE STEP tuşuna yeniden basmak ve daha sonra *NEXT* tuşuna basmak yordam içindeki bir sonraki komutu yürütür. Bu gerçekte her komuttan sonra bir durdurma noktası koymanın kısa yoludur. Eğer herhangi bir noktada bir kaydedici içinde doğru veri gözükmezse, doğru veriyi girip yordamın geri kalan kısmını yürütmeye devam edebilirsiniz. Bu size doğru veri verildiğinde yordamın geri kalan kısmının doğru çalışıp çalışmadığını anlatır. Daha sonra geri dönebilir ve o adımda kaydediciye niçin yanlış veri geldiğini kontrol edebilirsiniz. Her program yordamı farklı olduğundan, o yordam için izlenecek en iyi yaklaşımı düşünmelisiniz. Problemin bulunabileceği bölgeyi daraltıp, problemi çözecek denemeleri düşünmeyi deneyiniz.

PROGRAM MODÜLLERİNİN BAĞLANMASI

Komşu modüller RAM'a girildikten ve hataları ayıklandıktan sonra daha büyük modüller halinde bağlanabilirler. Örneğin terazi programı içindeki veriyi analogtan sayısala çeviren kısım *DSPLY* yordamı ile bağlanabilir. Böylece ağırlık dört *LED* gösterge üzerinde gösterilebilir. *KYBINT* yordamının hataları ayıklandıktan sonra, programın 82 ile 90 nolu satırları arasındaki kısmı da ilave edilebilir. Bu, programın sessiz-terazi kısmının hata ayıklanmasını tamamlar.

Programın geri kalan kısmı modüler yapıda bağlanır. Bir yordam önce bağımsız olarak hatalarından ayıklanır ve daha sonra hataları ayıklanmış ana program üstüne ilave edilir.

DURDURMA NOKTALARINI VE MEK6800D2'NİN MONİTOR KOMUTLARINI KULLANARAK EPROM PROGRAMLAYICI PROGRAMININ TEST EDİLMESİ VE HATALARININ AYIKLANMASI

MEK6800D2 üzerindeki bir program modülünü ya da yordamı test etmenin genel adımları yukarıda tartışılan SDK-85'in adımlarına çok benzemektedir, fakat monitor programındaki değişikliklerden dolayı özellikler farklıdır. Bu yüzden MEK6800D2 ile geliştirilen *EPROM programlayıcı programı için yöntem burada tekrarlanacaktır.*

İlk olarak mantıklı ve kabul edilebilir büyüklükte program modülünü başlangıç için seçiniz. Örneğin EPROM programlayıcı programı içinde ilk kullanıma hazırlama ile ve daha sonra programlanan 2716'nın *silinip silinmediğini kontrol eden program modülü* ile başlamak iyi bir başlangıçtır.

İkinci olarak seçilen parçada hangi ilk kullanıma hazırlamanın gerektiğine dikkatlice bakınız. Gerekli girişleri ve beklenen çıkışları belirleyiniz. Modülün başlangıcına kendi kendine yeterli olabilmesi için birkaç program adımı eklemeniz gerekebilir. Örneğin, gecikme yordamını test edebilmek için başlangıçta gecikme sabitini yüklemeniz gerekmektedir. Bu yordamın amacı çıkış darbesinin kritik genişliğini kontrol etmek olduğundan, programın sürekli darbe katarı üretmesi için gerekli adımları ilave edebilirsiniz. Daha sonra bir osiloskop ile *U26*'nın PB1 bacağında 50ms darbe genişliğini kontrol edebilirsiniz. Şekil 13.3 bunun için gerekli ilaveleri göstermektedir. Bir modülü kontrol ederken bir PIA kullanmanız gerekiyorsa, PIA'nın doğru şekilde kullanıma hazırlandığını garanti etmelisiniz.

Eğer bir program yukarıda anlatılan gecikme döngüsünde olduğu gibi test için sonsuz döngüye girmezse, modülün sonunda kontrolu monitore geri döndürecek bir komut bulunmalıdır. Eğer bu komut konulmazsa, işlemci modülü bitirecek ve daha sonra RAM içine girip orada bulunan rasgele komutları yürütmeyi deneyecektir.

```
START:  LDX   #$8000        ; PIA'LARIN TABAN ADRESİNİ YÜKLE
        LDAA  #$FF          ; ÇIKIŞ İÇİN KODU YÜKLE
        STAA  06, X         ; PIA'YA GÖNDER
        LDAA  #$04          ; PORT KONTROLU KODUNU YÜKLE
        STAA  07,X          ; U26' NIN KONTROL PORTUNA GÖNDER
        LDAA  #$06          ; PB1 YÜKSEK KODUNU YÜKLE
        STAA  06,X          ; U26'NIN B PORTUNA GÖNDER
        LDX   #$0F00        ; GECİKME SABİTİNİ YÜKLE
DELAY:  DEX                 ; AZALT
        BNE   DELAY         ; SIFIR DEĞİL İSE DÖN
        LDAA  #$04          ; PB1 ALÇAK KODUNU YÜKLE
        STAA  $8006         ; U26'NIN B PORTUNA GÖNDER
        JMP   START         ; ÜSTÜSTE TEKRARLA
```

Şekil 13.3 : EPROM Programlayıcı içindeki DELAY yordamını test edecek 6800 yordamı

MEK6800D2 için yordam sonuna konulan bir *yazılım kesme* komutu (SWI), kontrolu JBUG monitor programına verir. SWI aynı zamanda program sayıcı içeriğini, dizin kaydedici içeriğini ve *A* ve *B* akümülatör içeriklerini yığının üstüne iter. JBUG monitorun MPU kaydedicilerini gözden geçirme komutu R bu kaydedicilerin içeriklerini göstermek için kullanılabilir.

Bir modülün test edilebimesi için tanımlanmış girişleri ve beklenen çıkışlarının olması gerektiğini hatırlayınız. İlave olarak bir testin bozucu olmamasını garanti ediniz. Örneğin EPROM programlayıcıda uzun süre uygulanan 25V V_{pp}, 2716'yı aşırı ısıtıp bozabilir. Bir boş soket 6821 PIA'ya hep 1'ler olarak okunacağından, programın ilk kısmı boş bir soket ile test edilebilir. Programın birinci kısmı çalışırken, *Vpp* bacağının yanlış zamanda kazaen 25V değerini almadığından emin olmak için kontrol edilebilir.

Üçüncü olarak, JBUG monitorun belleği gözden geçir ve değiştir komutu M kullanılarak program modülü kart üstündeki RAM'a girilebilir. Tuş takımı ile başlangıç adresini ve bu adreste istenen onaltılık kodu giriniz. *G* tuşuna basmak adresi bir arttırır ve bir sonraki program baytının girilmesine imkân verir. Bütün program girildikten sonra kontrolu monitore geri vermek için *E* tuşuna basınız. Gerekli başlangıç ve bitiş adreslerini RAM içinde 0002H'den 0007H'ye kadar olan bölgeye girdiğinizden emin olunuz.

Dördüncü olarak program modülünü çalıştırınız. Bunu yapmak için başlangıç adresini girip G tuşuna basınız. Eğer modül SWI komutuna kadar çalışırsa, monitorun bir sonraki komutu beklediğini belirten bir bilgi isteme

karakteri gözükecektir. *R* komutu kaydedicilere bakmak için kullanılabilir. Eğer program bir döngü içinde takılır ve SWI komutuna ulaşamazsa monitor kontroluna dönmek için *E* tuşuna basınız.

BİR PROGRAM MODÜLÜNDE HATA AYIKLANMASI

Eğer program modülü doğru olarak çalışmazsa, her komut için doğru onaltılık kod girildiğini ve hiçbir komutun dışarıda bırakılmadığını kontrol ediniz. Eğer program elle çevrilmişse, yazılı programın her bir komut için doğru onaltılık koda sahip olduğunu kontrol ediniz. Eğer hiçbir kodlama ve yazma hatası bulunmazsa, test edilen modülün büyüklüğünü azaltınız.

Program modülünün büyüklüğü, programın daha önceki bir yerine SWI komutu girilerek kolayca azaltılabilir. SWI bir durdurma noktası yaratır ve sizin monitor komutlarını kullanarak kaydedici ve bellek tamponlarının içeriklerini kontrol etmenize imkân verir. Örneğin, EPROM programlayıcı programında 2716 içinde tamamen 1'ler olduğunu kontrol eden kısmın çalışmadığını varsayalım. SWI durdurma noktası 003E adresine, CMPA #$FF komutu yerine geri kaydırılır. Bu noktada program 2716'nın yalnızca birinci adresini adresleyecek ve okuyacaktır. Dizin kaydedicisi 8000H, *A* akümülatörü FFH ve *B* akümülatörü de 2716'ya gönderilen adresin en küçük değerlikli baytını içermelidir. Aynı zamanda, bir osiloskop kullanarak 2716'nın bacaklarına doğru adresler ve yetkinlendirme sinyallerinin gelip gelmediği kontrol edilebilir. Eğer adres ve yetkilendirme sinyalleri 2716 soketine gelmiyorsa muhtemel bir sebep PIA'nın ilk kullanıma doğru hazırlanmamış olmasıdır. Eğer sinyaller 2716'ya geliyorsa fakat doğru değillerse, bu modülün hangi parçasının problemli olduğuna dair bir ipucudur.

Bir program modülü içinde problemleri takip etmek için MEK6800D2 çok kullanışlı iki komuta sahiptir: durdurma noktalarının konulması ve kaldırılması için *V* ve birer birer izleme komutu *N*.

V komutu ile konulan bir durdurma noktasının avantajı SWI komutu tarafından değiştirilen komutun monitor tarafından saklanmasıdır. SWI yordamı aynı zamanda bütün kaydedicilerin içeriklerini yığının üstünde saklar. Kaydediciler kontrol edildikten sonra, *E* tuşuna ve sonrada *G* tuşuna basılması komutun RAM içine yeniden konulmasına sebep olur. Daha sonra program bir sonraki durdurma noktasına yürütülür. Bu sizin bir durdurma noktasından diğerine kolaylıkla gitmenize izin verir. JBUG monitoru bu tipte beş durdurma noktasının konulmasına izin verir. Her birisi istenen adresi girerek ve *V* tuşuna basılarak konulur. *V* komutu durdurma noktalarının konulması ve kaldırılmasını doğrudan yerleştirmeden daha hızlı ve az hata ile

yapma imkânı verir. *V* komutu durdurma noktaları bilgi isteme karakteri mevcutken *V* tuşuna basılarak kaldırılabilir.

Komut izleme komutu *N*, program boyunca her seferinde bir komut yürüterek ilerlemeye izin verir. Bir program bir durdurma noktasına kadar çalıştıktan sonra *N* tuşuna basmak işlemcinin bir sonraki komutu yürütmesine ve durmasına sebep olur. Kaydediciler gözden geçirilebilir. *N* tuşuna yeniden basmak işlemcinin bir sonraki komutu yürütmesine ve yeniden durmasına sebep olur. Bu şekilde bir program modülü boyunca her seferinde bir adım ilerleyebilirsiniz ve problemi bulmak için her adımda kaydedicileri kontrol edebilirsiniz.

PROGRAM MODÜLLERİNİN BAĞLANMASI : Bir program modülündeki hatalar ayıklandıktan sonra bu modül bir *başka program modülü ile bağlanabilir* ve *birleşim test edilebilir.* Örneğin Şekil 12.17'deki EPROM programlayıcı programında PIA'ları ilk kullanıma hazırlayan ve 2716'nın tamamen 1'lerden oluştuğunu kontrol eden kısımdaki hatalar ayıklandıktan sonra EXIT yordamını ilave ediniz. Bu yordam 2716'dan okunan ve tamamen 1 olmayan ilk baytın adresini ve bu baytı göstermelidir. Bütün 1'leri test eden yordamı boş bir soket üzerinde çalıştırmak devamlı tamamen 1'ler okuyacaktır ve program hiçbir zaman hata çıkışına gitmeyecektir. Program, 2716 soket bacaklarından bir tanesini toprağa bağlayarak hata çıkışına gönderilebilir. Bu durumda EXIT yordamı 2716 içinde test edilen ilk adresi ve toprağa bağlayarak programladığınız baytı gösterecektir. Eğer bu kısım çalışmazsa durdurma noktaları ve birer izleme komutları problemi bulabilmek için kullanılırlar.

Bu kısım doğru olarak çalıştıktan sonra ilave edilecek bir sonraki kısım Şekil 12.17'de 0007H adresinden başlayan 2716 programlama modülüdür. Bu kısım başlangıçta programlama soketinde 2716 yok iken test edilebilir. İlk olarak 009BH adresine bir durdurma noktası koyup programı o noktaya kadar çalıştırmak isteyebilirsiniz. Daha sonra sokete, doğru adres, veri ve yetkilendirme sinyallerinin geldiğini kontrol edebilirsiniz. Bir durdurma noktası konulacak bir sonraki yer 00A1H adresi olabilir. Program bu durdurma noktasına kadar çalıştıktan sonra PGM darbesinin ($\overline{CE}$=1) 2716 soketine gelip gelmediğini kontrol edebilirsiniz. Durdurma noktası konulacak bir sonraki mantıklı yerde 00ACH adresidir. Burada PGM darbesinin yeniden kapatıldığını kontrol edebilirsiniz. Buraya kadar herşey doğru çalışırsa, önceki durdurma noktalarının hepsini kaldırıp 00BCH adresine yeni bir durdurma noktası koyunuz. Bu programın PGMLUP'tan dönmesini sağlar. Eğer 50 ms. gecikmenin hassaslığını ölçmediyseniz, 2716 soketinin $\overline{CE}$/PGM bacağına bir osiloskop bağlayarak program çalışırken PGM darbesinin genişliğini ölçebilirsiniz.

Bu parça tamamen test edildikten sonra, programın VERIFY kısmı RAM'a girilir ve önceki kısımlarla birlikte test edilir. Eğer yeniden boş bir 2716 soketi kullanılırsa, VERIFY programı *FAIL* çıkışına gider. FAIL yordamı yedi-parçalı LED göstergelerin hepsinde *F* gösterir. VERIFY kısmını test etmek için durdurma noktası konulacak iyi bir yer 00D0H adresidir. Program bu durdurma noktasına kadar çalıştıktan sonra, 25V Vpp'nin kapanıp kapanmadığını ve 2716'nın çıkışa yetkilendirildiğini kontrol edebilirsiniz. Bu önemlidir çünkü Vpp $\overline{CE}$ yüksek iken *uzun süre 25V değerinde tutulursa 2716 aşırı ısınır ve ölür.*

Eğer sinyaller doğru ise ve program FAIL çıkışına problemsiz gelirse, programlama soketine silinmiş bir 2716 sokup onu *programlayabilirsiniz.* Biraz pahalı olan 2716 kendisini bozabilecek her şey tamamen kontrol edilmeden sokete sokulmamalıdır. Şimdi bütün 2kbaytı programlamak yerine monitor programından 100 bayt kadar bir bölgeyi 2716 içine programlamayı deneyiniz. 2716'nın tamamen programlanması yaklaşık 2 dakika tutacağından programın çalışıp çalışmadığını kontrol etmek için bu süreyi beklemelisiniz. Başka bir test yapmak için 2716'yı silmeli veya bir başka 2716 kullanmalısınız.

Eğer 2716 doğru olarak programlanırsa, kısa bir süre sonra gösterge tamamen A'lar göstermelidir. Eğer gösterge tamamen F'ler gösterirse problemin programlama bölümünde mi yoksa onaylama kısmında mı olduğunu bulmalısınız. 2716'nın programlanıp programlanmadığını bulmak için 2716'yı 6000H adresindeki kopyalanacak ROM soketine taşıyınız. JBUG monitorunun belleği gözden geçir ve değiştir komutu *M*'yi kullanarak 2716'yı okuyabilir ve monitor.programının basılı çıktısı ile karşılaştırabilirsiniz. Eğer 2716 doğru programları içerirse yeniden programlama soketi içine sokunuz ve durdurma noktaları ya da tek adım izleme ile *onaylama modülü üzerinden geçiniz.*

Bütün kontrol ve test etme çok zahmetli görülebilir fakat hata ayıklamanın ve çalışan bir temel örnek üretmenin en hızlı yoludur.

BİR TEMEL ÖRNEĞİ, GELİŞTİRME KARTI VE MONİTOR PROGRAMI OLMADAN GELİŞTİRME, TEST ETME VE HATALARINI AYIKLAMA

Mikroişlemci tabanlı bir aletin temel örneğini bir geliştirme kartı ve monitor programı olmadan geliştirmek, bir lojik analizleyici veya emülatör mevcut değilse çok zaman alıcı ve zahmetli olabilir. Bu kısım da önce bir lojik analizleyicinin donanım ve yazılımı geliştirmekte nasıl kullanıldığını gösterecek daha sonra emülatörün kullanılmasını göstereceğiz.

LOJİK ANALİZLEYİCİ KULLANARAK DONANIMIN TEST EDİLMESİ VE HATALARIN AYIKLANMASI

Lojik analizleyicilerin mikroişlemcilerle kullanılmasını tartışmadan önce, geri dönüp Bölüm 5'in sonundaki *lojik analizleyiciler* ve gösterme şekilleri hakkındaki tartışmayı özetlemek iyi bir düşüncedir. Orada da söylendiği gibi bir lojik analizleyici çok kanallı bir osiloskopa benzer. Paralel girişlerdeki veri örneklenir ve örnekler bellekte saklanır. Tetikleme meydana geldiğinde bellekte depolanan örneklenmiş veri bir CRT üzerinde istenilen formatla gösterilir. Tetikleme bir dış kaynaktan ya da tetikleme devresinden gelebilir. Bir tetikleme devresi, giriş veri kelimesi ile kullanıcı tarafından programlanmış kelimeyi anahtar aracılığı ile karşılaştırır. İki kelime eşleştiğinde devre belleğin içeriğini göstermesi için *analizleyiciye bir tetikleme darbesi gönderir.* Çoğu tetikleme devreleri, standart bir osiloskopu tetiklemek için kullanılabilen bir ön panel çıkışına da sahiptir.

Çoğu analizleyici veri akışının tetikleme öncesi, ortada tetikleme veya tetikleme sonrası olarak gösterilmesine izin veren ayarlanabilir tetikleme gecikmesine sahiptir. Her giriş kanalında 1024 bellek biti olan bir birimde tetikleme öncesi modu, tetiklemeden önce alının 1024 örneği gösterir. Ortada tetikleme modu tetikleme darbesini gösterge için 512 sayıncaya kadar geciktirir, böylece göstergede tetiklemeden önceki 512 örnek ile tetiklemeden sonraki 512 örnek gösterilir. Tetikleme sonrası modu tetiklemeyi 1024 sayıncaya kadar göstergeye geciktirir, böylece gösterge tetiklemeden sonraki 1024 örneği gösterir.

Örneklerin alınmasını belirleyen saat, test edilen birimin sistem saatinden ya da bir asenkron iç osilatörden gelebilir. Eğer sistem saati kullanılırsa, alet *lojik-durum analizleyicisi olarak adlandırılır,* çünkü her veri örneği test edilmekte olan makinenin bir durumuna yada makine döngüsüne karşı gelir. Bir lojik-durum analizleyicisi yazılım ve program akışındaki hata ayıklanması için mükemmel bir alettir. Buna rağmen eğer kilit-modu kapasitesi varsa bir lojik analizleyicisi zıplamaları yakalamakta çok faydalıdır.

Glitchleri algılayabilmek için veri örneklerini almakta, sistem saat frekansının 5 ile 10 katı olan asenkron saat kullanılır. Bu şekilde yüksek hızlı asenkron saat modu kullanan bir alet lojik-zaman analizleyicisi olarak adlandırılır. İsminden de anlaşılacağı gibi bu tip, zamanlama problemlerini ve zıplamaları bulmakta faydalıdır.

BİR MİKROİŞLEMCİYİ BİR LOJİK-DURUM YADA LOJİK - ZAMAN ANALİZLEYİCİSİ İLE GELİŞTİRMEK

Geliştirmedeki ilk adım saat ve reset devrelerini bağlamak ve test etmektir. Eğer saat ve reset devreleri 8080A veya 6800 sistemlerinde olduğu gibi ayrı bir entegre devrede ise mikroişlemciye bağlamadan önce çalıştırıp çıkışını kontrol ediniz. Eğer saat ve reset devreleri mikroişlemci ile aynı entegre devre içinde ise cihazın her bacağına doğru mantık seviyesi bağlandığına emin olunuz. Birçok MOS girişinin açık bırakılamıyacağını hatırlayınız. Saat darbe seviyelerini ve zamanlamayı kontrol edecek en iyi alet belkide standart bir osiloskoptur.

Saat ve reset devrelerini kontrol ettikten sonra mikroişlemci entegre devresini bağlayınız. Her bir bacağa dikkat ediniz. 10 kΩ veya daha büyük dirençler kullanarak veri taşıt hatlarını NOP komutunun ikilik koduna bağlayınız. 8080A/8085A için NOP kodu 00H, 6800 H için ise 01H'dir. Bir reset/yeniden başlatma işleminden sonra, işlemci adres hatlarında bir sayma dizisi göstermelidir. Sistem saati kullanan ve 0000H adresinde tetiklenen bir lojik - durum analizleyicisi bunu gösterecektir. Bu noktada bir bellek okuma sinyalinin de olduğunu kontrol ediniz.

Bir 8080A sisteminde bir sonraki adım, eğer kullanılıyorsa, 8228 sistem kontrolörünü ve bir taşıt sürücüsünü bağlamaktır. NOP dirençleri 8228'in bellek tarafına taşınır ve yukarıdaki test tekrar edilir.

8228 tipinde bir eleman kullanmayan bir sistemde veya 8228 kontrol ediltikten sonra bağlanacak bir sonraki kısım, eğer kullanılıyorsa, ROM ve RAM kod çözücüleridir. Bunları adres uçlarına ve güç kaynağına bağlayınız. Bir durum analizleyicinin tetikleme değerlendirici anahtarlarını, birinci kod çözücü çıkışı tarafından seçilecek bellek bloğu içinde seçilen ilk adreste tetikleme yapacak şekilde ayarlayınız. İşlemci sürekli olarak NOP işlemlerini yaparken kod çözücü çıkışı seçilen bloktaki her adres için bir darbe göstermelidir. Bir yonga-seçme çıkışı test edildikten sonra, tetikleme değerlendirici anahtarı kodlayıcı tarafından seçilecek bir sonraki bloğun ilk adresinde tetikleme yapacak şekilde değiştirilebilir.

Kod çözücüler doğru olarak çalıştığı zaman bir ROM soket bağlayabilirsiniz. Adres hatlarını, yetkilendirme hatlarını ve güç kaynağını cihaza bağlayınız; fakat veri çıkışlarını açık bırakınız. Bilinen bir programlı bir ROM'u sokete sokunuz. İşlemci NOP işlemlerini yaptıkça ROM'un içindeki her adresten geçecektir. ROM'un çıkışlarına bağlanmış bir durum analizleyici ROM'dan okuduğunuz kelimelerin sıralı olup olmadığını görmenizi sağlar. Eğer sıralı değillerse durum analizleyiciyi, ROM'un bacaklarına ulaşan adreslerin doğru sırada olduklarını kontrol etmekte kullanınız. Belleklere bağlamaktaki ortak

problem değiştirmeli adres hatlarıdır. Eğer ROM çıkışları doğru sırada ise NOP dirençlerini sökünüz ve ROM çıkışlarını işlemci veri hatlarına bağlayınız.

Temel örnek geliştirmenin geri kalan kısmındaki testleri yapmak için EPROM içindeki program yordamları kullanılabilir. Örneğin, RAM sisteme eklemek için soketler bağlanır. RAM içindeki takip eden her adrese artan sayılar yazan bir program EPROM içine konur ve çalıştırılır. RAM soketinin bir durum analizleyici ile kontrol edilmesi doğru adres, veri ve yetkilendirme sinyallerinin alınıp alınmadığını belirtir. Eğer doğru iseler RAM entegre devreleri yerlerine sokulur ve RAM içine yazan, geri okuyan ve onaylayan bir başka program çalıştırılır. Eğer bir RAM doğru mantık sinyalleri alıyorsa, fakat hâlâ programlanamıyor ya da okunamıyorsa, bir standart iki kanallı osiloskop zamanlamayı kontrol etmek ve adres, veri ve kontrol sinyallerinin tamamen mantık seviyeleri içinde olduğunu kontrol etmek için kullanılır.

I/O portları bağlanır ve benzer şekilde test edilirler. EPROM içindeki kısa bir program porta bir kelime yazar. Port adresinde tetiklenen bir durum analizleyici verinin porttan doğru zamanda çıkıp çıkmadığını gösterir. Porta tamamen 1 ya da tamamen 0 yazmanın ikisi de denenmelidir.

Şu ana kadar temel örnek geliştirmenin çoğu lojik-durum analizleyici kullanarak yapılmıştır. Bir örnek, bir lojik-zaman analizleyicinin neden değerli bir alet olduğunu gösterecektir. Bir test programı çalışırken bir kesme giriş hattının rasgele darbelenmesi dışında bütün donanımın doğru çalıştığını varsayalım. Problemi bulmak için kesme giriş hattı zaman analizleyici için bir tetikleyici olarak kullanılır. Analizleyicinin çoklu girişleri adres yolu, veri yolu ve kontrol yolu üzerindeki aktiviteleri gösterebilir. Bu sinyal hatlarından biri üzerinde kesme hattı darbesine karşı gelen bir geçişe bakınız. Eğer teller her birinin etrafında sarılmışsa bir yazma yada okuma darbesi kesme hattına bağlanabilir.

Donanım yapıldıktan ve hataları ayıklandıktan sonraki iş yazılımın test edilmesi ve hatalarının ayıklanmasıdır. Bunun için bir lojik-durum analizleyicisi çok faydalı olabilir.

LOJİK-DURUM ANALİZLEYİCİ KULLANARAK YAZILIMIN TEST EDİLMESİ VE HATALARIN AYIKLANMASI

Programların hatalarının monitor programı kullanarak ayıklanması gibi lojik-durum analizleyici ile yapılan hata ayıklamanın da başarı anahtarı her seferinde küçük kısımları test etmek ve hatalarını ayıklamaktır. Eğer inşa edilecek birim bir gösterge içeriyorsa, birime veri gönderecek bir yordam ile

başlamak uygundur. Gösterme yordamı, hatası ayıklandığında sisteme bir pencere gibi davranır. Bir program modülü, kaydedici içeriklerini ya da bir hesabın sonuçlarını kontrol etmeniz için göstergeye gönderebilir.

Üzerinde çalışılacak bir sonraki modül, eğer klavye kullanılıyorsa klavye tarama modülü olmalıdır. Hataları ayıklandığında bu yordam birkaç ilave ile verileri RAM'a girmenize izin vermelidir. Daha sonra yapılmakta olan aletin diğer program modülleri mantıki bir sıra ile test edilebilir ve hataları ayıklanabilir.

Daha önce anlatıldığı gibi, program modülü tek başına yeterli olacak şekilde yapılmalıdır. Çevre birim entegre devrelerinin kullanıma hazır hale getirildiklerinden emin olunuz. Modülün sonuna bir HALT veya JMP komutunun konulmasını sağlayınız, böylece CPU'nun belleğin geri kalan bölümünde depolanmış rasgele kelimeleri yürütmesini önlemiş olursunuz. Program modülü tek başına yeterli olacak hale geldikten sonra modülü bir EPROM içine gömünüz. Daha sonra EPROM'u bir temel örnek kartı üzerine takıp program modülünü çalıştırınız.

Bu bir lojik-durum analizleyicinin olaya girdiği noktadır. Analizleyiciyi, porta ya da belleğe gönderilen verileri gözlemekte kullanınız. Eğer program çalışmıyorsa ve analizleyicide yeteri kadar girişiniz varsa, program yürütülürken adres yolu ve veri yolu akışlarını kontrol ediniz. Eğer analizleyicinin yalnızca 16 girişi varsa program modülünün doğru sırayla çalıştığını kontrol etmek için adres yoluna bakınız. Şekil 5.23.*b*'de olduğu gibi analizleyici adres hattının içeriğini 1'ler ve 0'lar şeklinde veya Şekil 5.23*c*'de olduğu gibi daha kolay okunabilen onaltılık şekilde gösterebilir. Tipik olarak 16 kelime ve tetikleme kelimesi her seferinde bu formatta gösterilir. Eğer adres hatları doğru sırayla geliyorsa analizleyici girişlerini, komut ve verilerin doğru sırayla geldiğini kontrol etmek için veri yoluna bağlayınız. Eğer ilk 16 örnekte hata bulunmazsa, bir sonraki 16 adresi göstermek için tetikleme kelimesini değiştirin ve programı çalıştırınız. Daha sonra bu blok için adres yolu ve veri yolu sıralarını kontrol edebilirsiniz. Bir dallanma komutu için uzaklığın yanlış hesaplandığını ve programın yanlış adrese dallandığını bulabilirsiniz. Belki de bir komut için yanlış onaltılık kod EPROM'a girilmiş olabilir, ya da bir giriş portundan okunacak bir veri kelimesi okunmayabilir. *Buradaki temel nokta analizleyicinin, sizin işlemcinin ne yaptığını adım adım izlemenize imkân sağlamasıdır.*

Programda bulunan her hata düzeltilir. EPROM silinir, düzeltilen şekliyle yeniden pogramlanır ve yeniden denenir. Bu döngü, modülü çalıştırabilmek için gerektiği kadar tekrar edilir. Bir modül çalıştığı zaman, bir diğeri EPROM içinde o'na ilave edilir ve analizleyicinin yardımı ile hatalar ayıklanabilir.

DEVRE-İÇİ BİR EMÜLATÖR YADA SİMÜLATÖR VE BİR GELİŞTİRME SİSTEMİ KULLANARAK DONANIM VE YAZILIM GELİŞTİRİLMESİ

Geliştirme Sisteminin İncelenmesi

Devre içi emülatör veya simülatörler mikroişlemci geliştirme sistemleri ile kullanıldıklarından (Şekil 8.26), incelemeye geliştirme sisteminin fonksiyonu ile başlamak uygundur. Geliştirme sistemleri genellikle bir kontrol ünitesi, programların girilmesi için bir CRT terminal, programları saklamak için bir ya da daha fazla flopy disk sürücü ve programların kağıda basılması için bir satır yazıcıdan oluşur. Kontrol ünitesi tipik olarak bir CPU ve ROM içinde bir monitor programına sahiptir. Monitor programı, programların girilmesine ve çalıştırılmasına imkân verir. İlave olarak, kontrol ünitesi 64 veya 32 Kbaytlık bir RAM ve birçok I/O portu içerir.

Geliştirme sisteminin iki temel fonksiyonu çevirici dili programlarını düzeltme ve çevirmedir. İlk olarak editör programı bir flopy disk ya da teypten RAM'a yüklenir. Editör programı CRT terminal kullanarak bir çevirici dili programını açıklamalarla birlikte RAM'a girmenize imkân verir. RAM içinde programların girildiği bölge, editör çalışma bölgesi olarak adlandırılır. Çeşitli editör komutları bütün programı yazmadan bazı komutları eklemeyi ya da çıkarmayı kolaylaştırır.

Bütün program ya da istenen program modülü *editör çalışma bölgesine* girildikten sonra bu program bir disk ya da *teype yüklenir*. Bu program kaynak dosya olarak adlandırılır. Daha sonra bir çevirici program RAM'a yüklenir. Eğer çevirici sistemde yeteri kadar RAM varsa, editör ve çevirici aynı anda RAM içinde olabilir. Eğer yeterince RAM yoksa, çevirici RAM içinde editörün yerine geçer.

Çevirici program çalıştırıldığında, bir programı çevirebilmek için genellikle *iki geçiş gerektirir.* Birinci geçişte, çevirici kaynak dosyayı okur ve RAM içinde bir *semboller tablosu* yapar. Bunun anlamı çevirici programın, program içinde kullanılan her bir sembolik adrese bir *mutlak adres* eşitlemesidir. Örneğin Şekil 12.6'daki terazi programının sonundaki liste bir sembol tablosudur. İkinci geçişte çevirici, her adres için makina ya da *amaç kodunu* üretir. Sizin verdiğiniz komutlara göre üretilen amaç kodu ile çevirici birçok şey yapabilir. Kodu saklamak üzere bir disk ya da teype gönderebilir. Bu bir *amaç dosyasının* yaratılması olarak adlandırılır. Şekil 12.6'da terazi için yapıldığı gibi çevirici, makine kodunu basılmak üzere bir yazıcıya gönderir. Bu bir liste dosyası olarak adlandırılır. Diğer bir seçenekte çeviricinin programın makine kodunu belirli bir RAM bölgesine yüklemesidir. Bu son seçenek emülatör ile kullanmak için çok değerlidir.

EMÜLATÖR DONANIMI : Emülatörün temel kavramı, kartın ya da geliştirilmekte olan sistemin parçaları yerine büyük bir RAM, I/O portları ve geliştirme sisteminin kontrol fonksiyonunu kullanmaktır. Bir emülatör birçok yönden daha önceden geliştirme kartları için anlatılan güçlü monitor programları gibi çalışır. Bunu yapmak için temel örneğin temel CPU grubu donanımı birleştirilir ve ohmmetre ile kontrol edilir. Devre içi emülatörden çıkan bağlantı kablosunun sonundaki bir soket temel örnek içindeki CPU entegre devre yerine sokulur. Şekil 13.4 INTEL ICE-85 emülatör donanımının bir fotografını ve blok diyagramını göstermektedir.

İki büyük PC kartı, Şekil 8.26'daki *geliştirme sisteminde* gösterilen Intel Intellec kontrol kasasına direkt olarak bağlanabilir. Biri kart izleme kartıdır, diğeri ise işlemci kartıdır. Bir kurdela kablo bölümü *işlemci kartını* iki yönlü tamponları içeren modüle bağlar. Tamponun diğer tarafındaki çok iletkenli bir kablo, modülü 40 bacaklı erkek entegre devre konnektörüne bağlar, bu konnektör prototipin 8085A soketine sokulmaktadır. Geliştirilmekte olan sistem için, emülatörün bağlantı kablosundan gelen tepki sokete bir 8085A sokulduğundaki tepki ile aynıdır. Diğer bir deyişle, temel örneğe gönderilen sinyaller ve temel örnekten bu sinyallere verilen cevaplar 8085A soketindekilerle aynıdır.

BİR DEVRE-İÇİ EMÜLATÖRÜN KULLANIMI

Birçok emülatör ve simülatör, çalışmalarını ve kullanımlarını anlatan uzun elkitaplarına sahiptirler. Bu yüzden eğer bir emülatör ile çalışıyorsanız onunla ilgili elkitabına bakınız. Bu bölümde emülatör kullanımının ve kapasitelerinin bir özetini vereceğiz.

Bütün prensip başlangıçta temel örnekte bulunanlar yerine geliştirme sisteminin RAM, ROM ve I/O portlarını kullanmaktır. Temel örnek donanımının her parçası eklenip hataları kaldırıldıkça, geliştirme sistemindeki parçanın fonksiyonlarını üzerine alır.

Emülatörü kullanmaya başlamak için, temel örneğin *temel CPU* grubu *entegre devreleri bağlanır* ve CPU yerine emülatörden gelen bağlantı kablosu sokulur. Daha sonra CPU grubu geliştirme sistemi, RAM'ın içindeki bir programla test edilebilir. Burası emülatörün işlem içine girdiği yerdir.

Emülatör test programını yürütür ve ICE-85 kullanılması durumunda RAM içinde yürütülen her makina döngüsü için 4 baytlık bilgiyi izleme kartı üzerine depolar. Depolanan 4 bayt: statü, adresin üst baytı, adresin alt baytı ve veridir. Bunlar bazen anlık veri olarak adlandırılır. ICE-85'te, bir durdurma noktasından önceki son 44 makine döngüsü için anlık veri, izleme RAM'ı içinde saklanır. Bu anlık baytlar, adres ve veri yollarında programın akışını izle-

mek için okunabilir ya da yazdırılabilir. Emülatör komutları herhangi bir yere durdurma noktaları koymak için kullanılabilir. Programlar tam hızda veya her seferinde bir adım olarak yürütülebilir.

Bir bölümdeki hatalar ayıklandıktan sonra diğeri kurulur. Test programı, geliştirme sisteminin editörü ile düzeltilir, çevirici fonksiyonları ile çevrilir ve emülatör fonksiyonları ile çalıştırılır. Birimlerin bu şekilde birleştirilmesi temel örnek geliştirme dönemlerini hızlandırır. Eğer bir program modülünde bir hata bulunursa, editör fonksiyonlarına geri döner ve kaynak programındaki hatayı düzeltirsiniz, düzeltilmiş kaynak programını çevirir ve sandalyenizi terketmeden düzeltilmiş programı yeniden deneyebilirsiniz.

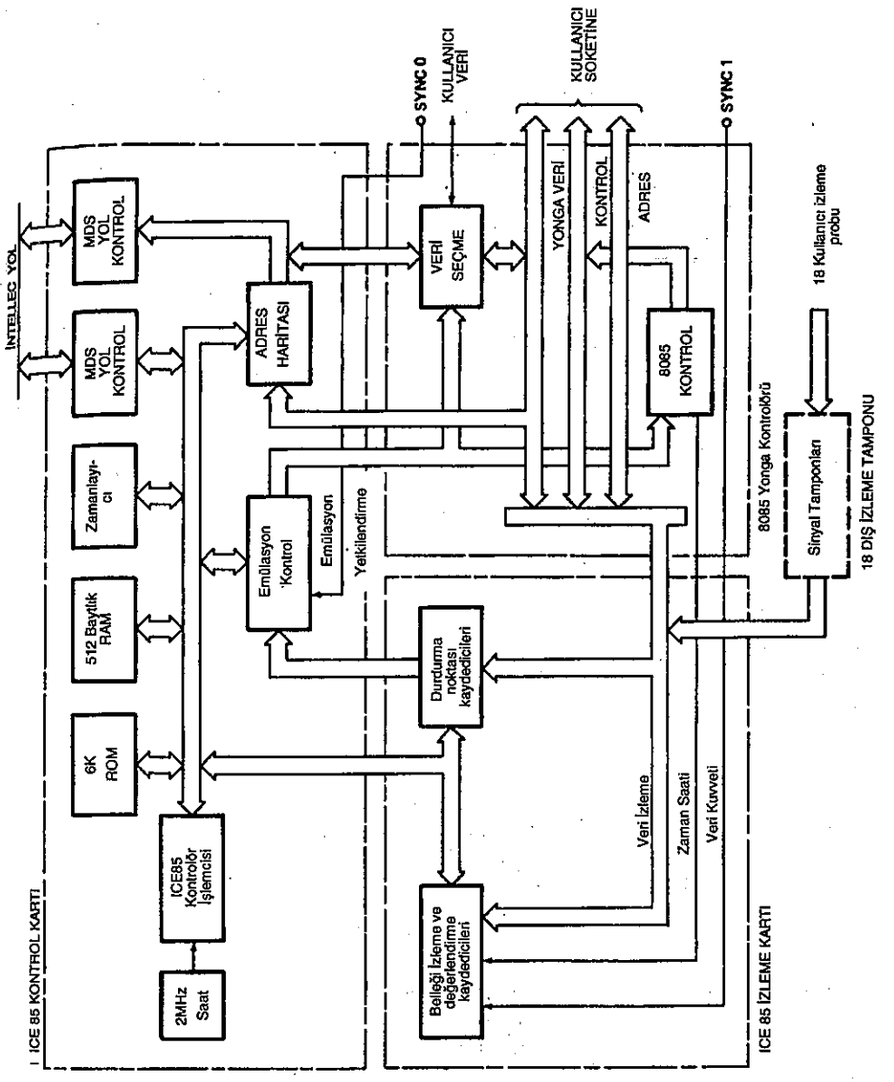
Anlık veriye ilave olarak, hataları bulmanıza yardım etmek için emülatörün bazı kuvvetli özellikleri vardır. *Deneme programı* bir durdurma noktasına kadar çalıştıktan sonra *sorgulama moduna* girer. Bu mod da emülatör programından, bellek içeriklerini, kaydedici içeriklerini, bayrakları ve hatta 8085A'nın HOLD, INTR, READY ve RESET bacaklarındaki sinyalleri göstermesini ya da basmasını isteyebilirsiniz. Diğer komutlar bunların herhangi birinin değiştirilmesini ve emülasyon programının devam etmesini sağlar.

Önceden de belirtildiği gibi bir emülatör birçok yönden geliştirme kartları üzerindeki kuvvetli bir monitör programının işlevlerini görür.

ÜRETİM SÜREÇLERİ : Mikroişlemci tabanlı bir aletin geliştirme adımları sırasında prototipi test etme ve hataları ayıklama için kullanılan aletleri ve teknikleri tartıştık. Şimdi bu prototipin satılabilir bir alet haline gelmesi için gerekli üretim adımlarını inceleyeceğiz.



Şekil 13.4 (a) Intellec Devre içi Emülatörü/85 Donanımı (ICE 85)



Şekil 13.4 (b) ICE-85 Blok Diyagramı (Intel Corporation)

ÜRETİM TEMEL ÖRNEĞİNİN ÇALIŞTIRILMASI : Bir aletin, temel örnek *geliştirme mühendisliği bölümünde tamamen test edilmesi* ve *hatalarının ayıklanmasından* sonra, *üretim mühendisliği bölümü bir baskılı devre kartını, çerçeveyi, kasayı, konnektörleri, düğmeleri* ve *etiketleri tasarlar.* Bu tasarım tamamlandığında ve parçalar toplandığında üretim bölümü belki 10 birimi temel örnek çalışması için inşa eder. Bu 10 birim muhtemelen test edilmek ve hataların ayıklanması için geliştirme mühendisliği bölümüne geri gönderilir. Bunun sebebi bazı marjinal zamanlama ve diğer faktörlerin ilk temel örnekte şans eseri çalışmış olmasıdır. Fakat, bunlar üretim temel örnek birimlerinin 10 tanesinden 5 tanesinde çalışmayabilir. 10 birimdeki hatalar ayıklandıktan, sıcaklık testleri yapıldıktan ve onaylandıktan sonra alet tam üretim için serbest bırakılır. Şemalar son şeklini alır, operatör servis elkitapları yazılır.

ÜRETİM : Üretimin bir bölümünde *çerçeve; baskılı devre kartı kenar konnektörleri, trafolar, anahtarlar, potlar, konnektörler* vb. ile mekanik olarak birleştirilir. Diğer bir bölümde dirençler, kondansatörler, entegre devreler ve entegre devre soketleri baskılı devre kartı üzerine monte edilir. Yüklenmiş PC kartları bütün elemanları hızla lehimleyen bir lehim makinesinden geçirilir. Herhangi bir hassas eleman elle lehimlenebilir. Lehimlendikten sonra, lehim akışlarını temizlemek için PC kartı çözücü ile temizlenir.

KART TESTİ : *PC kartları birleştirilmiş çerçevenin içine konulmadan önce üretimin kart test bölümüne* gönderilir. Burada teknisyenler her kartı test eder ve arızaları giderirler. Bir teknisyen ilk olarak; lehim köprüleri, elemanların beraber kısa devre olmuş uçları, kırılmış ya da yerine konulmamış elemanlar, ters konulmuş entegre devreler, diyotlar veya elektrolitik kondansatörler vb. için gözle dikkatli bir inceleme yapar. Bundan sonra kart birçok yönden test edilebilir.

Birçok kartın üretildiği büyük firmalarda Şekil 5.20'deki Sentry Entegre Devre Test edici gibi bilgisayar kontrollu kart test edici kullanılabilir. Bu makineler tanı testlerini çalıştırabilir ve ilave olarak kartın testi geçip geçmediğini söyleyebilir, çoğu zaman muhtemel hata kaynağını da belirtebilir. Bilgisayar kontrollu test edicilerin temal dezavantajı, yüksek maliyetleridir.

Kartları test etmenin ikinci bir yolu bir test düzeneğidir. *Test düzenekleri çok değişik olabilir,* fakat ana fonksiyonları karta güç ve giriş sinyallerini sağlamak ve karttan gelen çıkış sinyallerini okumaktır. Teknisyen karttaki arızaları gidermek için bir osiloskop veya lojik analizleyici kullanır. Problemler bulunup, çözüldükçe notlar alınır ve böylece aynı hata tekrarlarsa tamiri daha kolay olur. Eğer bir problem üstüste oluşursa mühendislere danışılır.

Kartları kontrol etmenin bir diğer yolu da test edilecek kart dışında tamamen birleştirilmiş aleti kullanmaktır. Çerçeveye test edilecek kart yerine bir genişletme kartı eklenir. Bir genişletme kartı, çerçeveye bağlandığı kenardaki her bacağın diğer kenardaki kenar konnektör soketteki her bacağa izi olan bir

PC kartıdır. Test edilecek kart, kenar konnektör soketine sokulur. Genişletme kartı, test edilecek kartı alet içindeki diğer kartlardan daha yükseğe yerleştirir, böylece test probları kolayca bağlanabilir. Bütün PC kartları test edildikten ve hataları ayıklandıktan sonra üretim test bölümüne gönderilir.

ÜRETİM TEST YÖNTEMLERİ

Bir üretim test teknisyeni ilk olarak, güç kaynağı regülatör kartını çerçeve içine sokar ve bellek ve mikroişlemci kartları sokulmadan *önce güç-kaynağı gerilimlerini kontrol eder.* Bu çok önemlidir, çünkü değişik kaynak gerilimleri gerektiren yarıiletken bellekleri bozulabilir. Örneğin +12V, +5V olmadan uygulanabilir. ac güç ilk defa uygulandığında, değişken-gerilimli bir ac kaynak, gerilimi yavaş yavaş 120V ac değerine getirmek için kullanılabilir. Bu, büyük elektrolitik kondansatörlerde dielektriğin birikmesine süre tanır. Bütün güç kaynağı gerilimleri kontrol edildikten sonra, yöntem aletin büyüklüğüne göre değişir. Saat, işlemci, ROM, RAM ve I/O gibi değişik fonksiyonlar için PC kartları kullanan büyük bir alette, ilk olarak saat kartı takılmalı ve test edilmelidir. Kart, belki de kart testi kısmında test edilmiş olabilir, fakat şimdi çerçeve içinde test edilmeli ve hataları ayıklanmalıdır. Takılacak ve test edilecek diğer kartlar: işlemci kartı, RAM, ROM ve çevrebirim kontrolör kartlarıdır. Belki de küçük bir alet bütün bu üç fonksiyona sahiptir, böylece güç kaynağı kontrol edildikten sonra bu tek kart takılır ve hataları ayıklanır.

Aletin tamamı birleştirildikten ve çalıştıktan sonra kaba bir kalibrasyon yapılır ve 24 ya da 48 saat çalışmak üzere bir pişirme süresince bir raf üzerinde bırakılır. Bu çalışmadan sonra alet yeniden kontrol edilir ve gerekirse hataları ayıklanır.

Daha sonra ikinci bir pişirme periyodun da çalışır olarak bırakılır, bu periyot belli bir süre bir sıcaklık bölmesi içinde çalışmayı da kapsayabilir. Bunun sebebi yarı iletken parçaların birkaç saat çalışmadan sonra bazı sebeplerden dolayı çalışmaz olabileceğidir. Eğer parçalar ilk 10 ile 100 saat arasında doğru şekilde çalışırlarsa, muhtemelen birçok sene daha bozulmadan çalışacaktır.

TEST, KALİBRASYON VE KALİTE KONTROLU

Alet pişirme yönteminden geçtikten sonra, test ve *kalibrasyon bölümüne* gider. Burada kalibrasyon teknisyenleri, aletin en iyi performansı için bütün ayarları hassas olarak ayarlarlar. Kalibrasyondan sonra alet bütün özellikleri karşıladığını onaylayan *kalite kontrol* teknisyenine gider. Alet kalite kontrolu geçtikten sonra gönderilmek üzere paketlenir.

ARAZİ SERVİSLERİ : Eğer bir alet müşteriye gittikten sonra bozulursa bir *arazi servis mühendisi* ya da *arazi servis teknisyeni* aleti tamir etmek için görevlendirilir. Alet belki fabrika içindeki arazi servis bölümüne geri getirilir veya bir arazi servis teknisyeni aleti tamir etmek için müşterinin bulunduğu yere gider.

MİKROİŞLEMCİ TABANLI ALETLER İÇİN ÜRETİM TESTİ VE ARAZİ SERVİSİ ALETLERİ VE TEKNİKLERİ

GENEL YORUMLAR VE İPUÇLARI

Bir üretim testi veya arazi servisi teknisyeni için en temel aletler: çok kanallı osiloskop, bir sayısal voltmetre, alet için servis elkitabı ve sistematik yaklaşımdır. Teknisyenler problemleri bulma kabiliyetleri ve hızla tamir etmelerinden dolayı değerlendirildiklerinden, birçok muhtemel problem kombinasyonunun bulunduğu bir alette arızaları gidermek için sistematik yaklaşımı kullanmak çok önemlidir.

1. Belirtileri açık olarak belirle. Belirtileri yazmak çoğu zaman çok yardımcı olur.
2. Problemin bulunacağı bölgeyi daraltmak için yapılması gereken deney ya da testleri düşünmeye çalış.
3. Güç kaynağı gerilimlerinin belirtilen sınırlar içinde olduğunu kontrol et, böylece mikroişlemci devresindeki düşük V_{cc} tarafından yaratılacak problemleri izlemek için vaktinizi harcamazsınız.
4. Saat sinyallerinin mevcut ve doğru olduklarını kontrol et.
5. Çok kartlı bir sistemde, temel çalışma için gerekli kartlar dışındaki bütün kartları çıkar. Bu problemli kartı bulabilmenizi kolaylaştırabilir. Yedek kartlarınız varsa onları yanlış çalışan birimdeki kartlarla birer birer değiştirmeyi deneyin.
6. Problemi belirli bir kart üzerine daraltabilirseniz, 9. ve 10. bölümlerde anlatılan hatalı entegre devreyi bulan arıza giderme tekniklerini kullanabilirsiniz.
7. Eğer sayısal voltmetre ve osiloskop kullanımı problemi bulmakta yeterli değilse daha karmaşık aletler gerekebilir. Daha sonraki bir bölümde bunlardan bazılarını tartışacağız.

Diğer faydalı bir ipucu da üzerinde çalıştığınız her aletin servis elkitabının bir kopyasını elde etmektir. Birçok birimdeki arızaları giderirken ve bir problemi izlerken karşılaştığınız ilave dalga şekillerini ve gerilimleri elkitabına yazmak isteyebilirsiniz. Aynı zamanda el kitabının arkasına belirtileri ve bunları düzeltmek için yaptığınız işlemleri listeleyebilirsiniz. Bu liste ve notlar, eğer bir alet üstünde bir yıl çalışmayıp sonra yeniden onun üzerinde çalışmaya başladığınızda çok değerlidir. Notlar sizin bıraktığınız yerden hızla toparlamanıza yardım eder.

Arazi servisi durumundaki faydalı bir teknikte servis vereceğiniz her alet için yedek PC kartları taşımaktır. Eğer bir problem hızla bulunamazsa, problemli kart iyi olduğu bilinen bir tanesi ile değiştirilebilir. Bu müşterileri mutlu eder; çünkü aletin çalışmama zamanını en aza indirir. Aynı zamanda çalışmayan kartı fabrikaya götürüp daha karmaşık aletlerle problemi bulmanıza izin verir. Bu yaklaşımın bir diğer faydalı noktası, omuzunuzun üzerinden çalışmanıza merakla bakan bir müşterinin olmadığı fabrikada beyninizin daha hızlı çalışabilir olmasıdır. Eğer bu yapılmazsa, böyle zor problemler için biraz sonra anlatılacak test aletlerinden birini araziye götürebilirsiniz.

ZOR PROBLEMLER İÇİN TEST ALETLERİ

Mikroişlemci tabanlı bir alet ya da sistemdeki problem bir standart çok kanallı osiloskop ile hızla bulunamadığı zaman daha karmaşık aletler kullanılmalıdır. Zor problemleri çözmek için kullanılan dört aleti özet olarak anlatacağız.

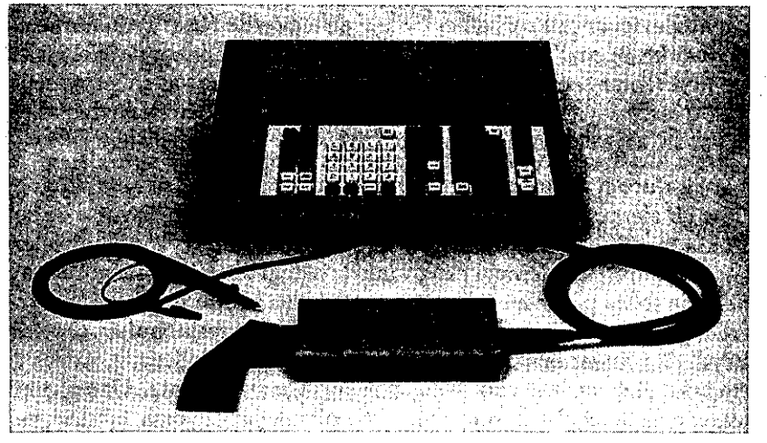
LOJİK ANALİZLEYİCİLER : Lojik-durum analizleyiciler ve lojik zaman analizleyiciler mikroişlemci tabanlı sistemlerde hata ayıklanması için çok faydalıdır. Bu analizleyicilerin güçlü bir özelliği yalnızca paralel girişteki kelime ile anahtar-seçmeli kelime eşleştiğinde tetikleme yapan şifreli tetiklemedir. Bu analizleyiciler içindeki bellek aynı zamanda, tetiklemeden önce ve sonra yollarda ne olduğunu görmenizi sağlar.

Lojik-durum analizleyicinin arazide kullanımı önceden anlatıldığı gibidir. Eğer bir alet programının bir parçasını doğru olarak çalıştırıyorsa ve belirli bir noktada takılıyorsa, aletin adres yolunu analizleyicinin girişlerine bağlayın ve şifreli tetiklemeyi programın takıldığı bölgedeki ilk adrese ayarlayın. Daha sonra program çalışırken analizleyici üzerinde gösterilen dizi ile program çıkışındaki diziyi karşılaştırıp problem noktasını bulun. Öntetikleme kaydı ile probleme sebep olan adres dizisine bakabilirsiniz. Adres yolu üzerindeki dizi yerine veri yolu dizisini kontrol ediniz. Uzun dizileri karşılaştırmak, analizleyici bunu otomatik olarak yapmıyorsa, bıktırıcı ve zaman alıcı olabilir. Örneğin Tektronix 7D01 analizleyici, çalışan bir sistemden alınan paralel veriyi kaydeder ve ekranın yarısında gösterir, ekranın diğer

yarısında ise çalışmayan sistemden alınan paralel veriyi gösterir. Şekil 5.23 *b* de gösterildiği gibi iki yarıda uyuşmayan bit çalışmayan sistem göstergesinde parlatılır. Bu, eğer karşılaştırmak için çalışan bir birim varsa çok iyi sonuçlar verir. Fakat arazideki durum çoğu zaman böyle değildir. Bu problemi çözebilmek için Fluke 9010A Mikrosistem Arıza Gidericiye benzer aletler geliştirilmiştir.

FLUKE 9010A MİKROSİSTEM ARIZA GİDERİCİ

Şekil 13.5 deki 9010A resminden de görebileceğiniz gibi bir klavye, bir gösterge ve sonunda bir entegre devre bağlantı fişi bulunan bir bağlantı kablosuna sahiptir. Birim aynı zamanda bir mini kaset teyp kaydedici içerir. 9010A arıza gidermek için aşağıdaki şekilde kullanılır. Tamamen çalışan bir birimdeki mikro işlemci entegre devresi sökülür ve yerine bağlantı kablosu sonundaki fiş yerleştirilir. Daha sonra 9010A'nın öğrenme fonksiyonu yürütülür. Bu fonksiyon RAM, ROM ve okunup, yazılacak I/O kaydedicilerini bulur ve haritalarını çıkarır. Aynı zamanda ROM bloklarının imzalarını hesaplar. Bütün bu parametreler 9010A'nın RAM'ına ve/veya bir minikaset teybe depolanır. Çalışmayan birim üzerindeki mikroişlemci sökülür ve yerine bağlantı kablosu sonundaki fiş yerleştirilir. Daha sonra bir otomatik test fonksiyonu yürütülür. Bu mod içinde 9010A, çalışmayan birimdeki taşıtları, RAM, ROM, portlar, güç kaynağı ve saati otomatik olarak test eder. Takılmış düğüm noktaları veya komşu izlerdeki kısa devreler gibi bulunan bir problem göstergede belirtilir. 9010A diğer testleri yapmak içinde programlanabilir.



Şekil 13.5 : Fluke 9010A Mikrosistem Arıza Giderici

9010A gibi bir aletteki önemli nokta, uyartılı-tepki gücünü ve otomatik test etmeyi arazi servisinin içine taşımasıdır. Daha önceleri bu tipteki *otomatik test aletleri* (ATE), boyutları ve maliyetleri yüzünden fabrika içinde üretim test alanı sınırlı idiler. 9010A bir elçantası içine kolaylıkla sığar ve yalnızca bir bağlantı kablosu ve arazi servis mühendisinin servis vereceği her işlemci için bir teyp gerektirir. 9010A'yı kullandığında, arazi servis mühendisi, arızasını gidereceği her tip birimin programlama dilini çok iyi bilmek zorunda değildir.

TASARIM İÇİN KENDİ-KENDİNE TEST VE İŞARET ANALİZİ

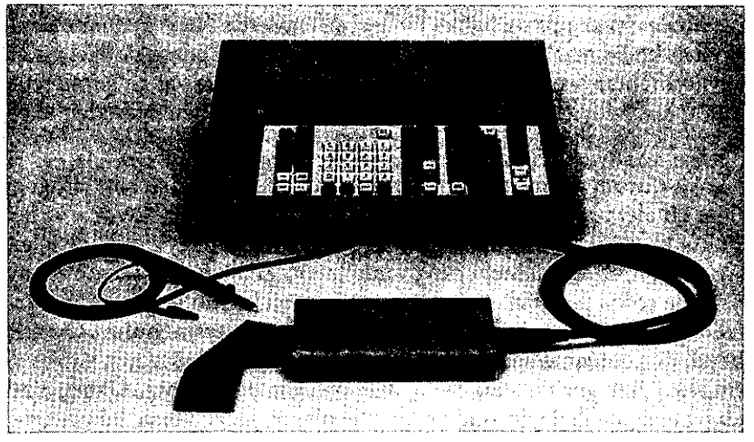
Mikroişlemci tabanlı aletlerdeki kuvvetli bir eğilim, tasarımlarına *kendi-kendine test* ve tanı yordamlarını ilave etmektir. Bu, aletlerde arıza gidermek için gerekli aletleri basitleştirir. Fakat, tasarım sırasında biraz daha fazla çaba gerektirir; çünkü test için gerekli fonksiyonel blokları kolayca ayırabilecek bir donanım yaratılmalıdır. Birçok durumda elde edilen fayda bu zamanın harcanmasına değer.

Çalışan mikroişlemci kontrollu bir alet gerçekte sayısal geribesleme döngülerinin toplamıdır. CPU bir komutu almak için ROM'a bir adres gönderir. CPU'ya geri beslenen komut CPU'nun bir sonraki işlevini belirler, vs. Bu geri besleme döngüleri bağlı olduğu sürece aletin arızasının giderilmesi çok zor olabilir. *Işaret analizi* için bazı döngüler açılır. Aletin bireysel parçaları bir döngü test programı ile çalıştırılır. Test programı sistem parçasını incelerken devrenin her noktasında belirli bir 1'ler ve 0'lar dizisi gözlenir. *Işaret analizleyici* karmaşık bir matematiksel yolla bu 1'ler ve 0'lar dizisini dört basamaklı onaltılık gösterge şekline dönüştürür. Bu dört basamaklı gösterge test edilen noktanın işareti olarak bilinir. Şekil 13.6 bir 6800 mikroişlemci sisteminde, bir ROM- RAM kod çözücü etrafında analizleyici ile bulunabilecek test işaretlerini göstermektedir. Şekil 13.7 Hewlett-Packard 5004A işaret analizleyicinin bir fotoğrafını göstermektedir. Boyutlarının yalnızca bir masa-modeli sayısal voltmetre kadar olduğuna dikkat ediniz.

yarısında ise çalışmayan sistemden alınan paralel veriyi gösterir. Şekil 5.23 *b* de gösterildiği gibi iki yarıda uyuşmayan bit çalışmayan sistem göstergesinde parlatılır. Bu, eğer karşılaştırmak için çalışan bir birim varsa çok iyi sonuçlar verir. Fakat arazideki durum çoğu zaman böyle değildir. Bu problemi çözebilmek için Fluke 9010A Mikrosistem Arıza Gidericiye benzer aletler geliştirilmiştir.

FLUKE 9010A MİKROSİSTEM ARIZA GİDERİCİ

Şekil 13.5 deki 9010A resminden de görebileceğiniz gibi bir klavye, bir gösterge ve sonunda bir entegre devre bağlantı fişi bulunan bir bağlantı kablosuna sahiptir. Birim aynı zamanda bir mini kaset teyp kaydedici içerir. 9010A arıza gidermek için aşağıdaki şekilde kullanılır. Tamamen çalışan bir birimdeki mikro işlemci entegre devresi sökülür ve yerine bağlantı kablosu sonundaki fiş yerleştirilir. Daha sonra 9010A'nın öğrenme fonksiyonu yürütülür. Bu fonksiyon RAM, ROM ve okunup, yazılacak I/O kaydedicilerini bulur ve haritalarını çıkarır. Aynı zamanda ROM bloklarının imzalarını hesaplar. Bütün bu parametreler 9010A'nın RAM'ına ve/veya bir minikaset teybe depolanır. Çalışmayan birim üzerindeki mikroişlemci sökülür ve yerine bağlantı kablosu sonundaki fiş yerleştirilir. Daha sonra bir otomatik test fonksiyonu yürütülür. Bu mod içinde 9010A, çalışmayan birimdeki taşıtları, RAM, ROM, portlar, güç kaynağı ve saati otomatik olarak test eder. Takılmış düğüm noktaları veya komşu izlerdeki kısa devreler gibi bulunan bir problem göstergede belirtilir. 9010A diğer testleri yapmak içinde programlanabilir.



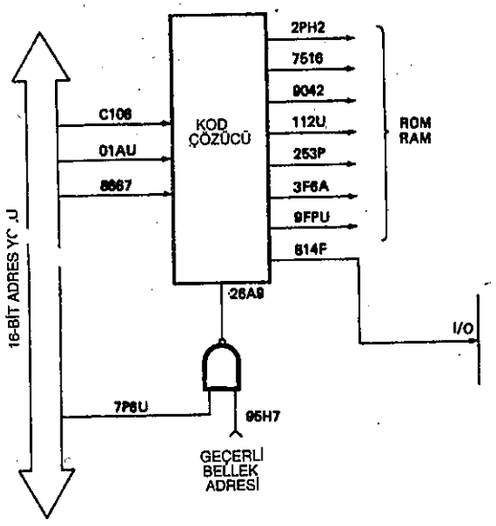
Şekil 13.5 : Fluke 9010A Mikrosistem Arıza Giderici

9010A gibi bir aletteki önemli nokta, uyartılı-tepki gücünü ve otomatik test etmeyi arazi servisinin içine taşımasıdır. Daha önceleri bu tipteki *otomatik test aletleri* (ATE), boyutları ve maliyetleri yüzünden fabrika içinde üretim test alanı sınırlı idiler. 9010A bir elçantası içine kolaylıkla sığar ve yalnızca bir bağlantı kablosu ve arazi servis mühendisinin servis vereceği her işlemci için bir teyp gerektirir. 9010A'yı kullandığında, arazi servis mühendisi, arızasını gidereceği her tip birimin programlama dilini çok iyi bilmek zorunda değildir.

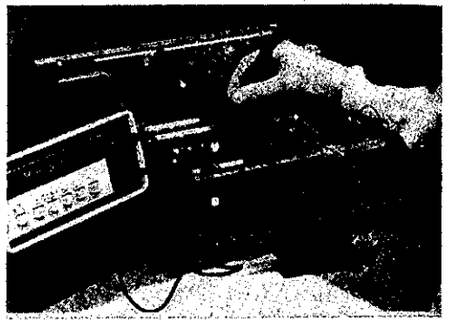
TASARIM İÇİN KENDİ-KENDİNE TEST VE İŞARET ANALİZİ

Mikroişlemci tabanlı aletlerdeki kuvvetli bir eğilim, tasarımlarına *kendi-kendine test* ve tanı yordamlarını ilave etmektir. Bu, aletlerde arıza gidermek için gerekli aletleri basitleştirir. Fakat, tasarım sırasında biraz daha fazla çaba gerektirir; çünkü test için gerekli fonksiyonel blokları kolayca ayırabilecek bir donanım yaratılmalıdır. Birçok durumda elde edilen fayda bu zamanın harcanmasına değer.

Çalışan mikroişlemci kontrollu bir alet gerçekte sayısal geribesleme döngülerinin toplamıdır. CPU bir komutu almak için ROM'a bir adres gönderir. CPU'ya geri beslenen komut CPU'nun bir sonraki işlevini belirler, vs. Bu geri besleme döngüleri bağlı olduğu sürece aletin arızasının giderilmesi çok zor olabilir. *Işaret analizi* için bazı döngüler açılır. Aletin bireysel parçaları bir döngü test programı ile çalıştırılır. Test programı sistem parçasını incelerken devrenin her noktasında belirli bir 1'ler ve 0'lar dizisi gözlenir. *Işaret analizleyici* karmaşık bir matematiksel yolla bu 1'ler ve 0'lar dizisini dört basamaklı onaltılık gösterge şekline dönüştürür. Bu dört basamaklı gösterge test edilen noktanın işareti olarak bilinir. Şekil 13.6 bir 6800 mikroişlemci sisteminde, bir ROM- RAM kod çözücü etrafında analizleyici ile bulunabilecek test işaretlerini göstermektedir. Şekil 13.7 Hewlett-Packard 5004A işaret analizleyicinin bir fotoğrafını göstermektedir. Boyutlarının yalnızca bir masa-modeli sayısal voltmetre kadar olduğuna dikkat ediniz.



Şekil 13.6 : 6800 Sisteminde ROM/RAM kod çözücü etrafındaki işaret dizisi



Şekil 13.7 : Hewlett Packard 5004 işaret Analizleyici

Bir işaret analizleyici ile bir teknisyen, bir sistem bölümünü inceleyerek ve her noktada elde ettiği işaretleri şema üzerinde veya bir işaret tablosunda listelenen veya devre kartı üzerine basılan doğru işaretlerle karşılaştırarak arazıları giderebilir. İşaret hatanın ne olduğu konusunda herhangi bir tanı bilgisi taşımaz. HP5005A standart onaltılık semboller yerine *0-9, A, C, F, H, P* ve *U* sembollerini kullanır. Doğru olmayan bir işaret, teknisyene o noktadaki sinyalin yanlış olduğunu anlatır. Buradaki teori , eğer cihaza giren sinyaller doğru fakat bir ya da birkaç çıkış sinyali yanlış ise aletin kötü olduğu ve değiştirilmesi gerektiğidir.

Bu yaklaşımda teknisyen, çoğu arıza gidermeler için devrenin çalışmasının detaylarını çok az bilebilir. Bir işaret analizleyici, bir tanı yordamı çalışırken teknisyenin noktadan noktaya karşılaştırma ile bir parçanın doğru çalışıp çalışmadığını bilmesini sağlar. Teknisyene daha ileri yardım için, işaret analizleyici ile kullanılmak üzere tasarlanan çoğu aletler detaylı arıza giderme

akış diyagramlarına sahiptir. Bu, belirli bir belirti için hangi test döngüsünün ilk olarak çalıştırılacağını, işaret analizleyici ile hangi noktaların kontrol edileceğini ve belli bir noktada yanlış işaret bulunduğunda hangi entegre devrenin hatalı olduğunu anlatır.

Bir örnek bunun pratikte nasıl kullanıldığını açıklayabilir. Varsayalım ki test edilecek entegre devre mikroişlemci entegre devresinin kendisidir. Bunu yapmak için mikroişlemci veri yolu, RAM, ROM vb. birimlerden ayrılmalıdır. Eğer bu tasarım sırasında planlanmışsa yalnızca bir atlayıcı fişini entegre devre soketinden çıkarmak yeterli olacaktır. Dirençleri V_{cc} ya da toprağa bağlayan bir diğer atlayıcının sokulması işlemcinin veri girişlerine NOP veya bir başka komut yükleyecektir. Mikroişlemci çalıştıkça her bacaktaki işaret okunur ve iyi olduğu bilinen bir birimdeki aynı bacaktan alınan işaret ile karşılaştırılır. Bu noktada ROM ve RAM kod çözücüleri de kontrol edilebilir.

Eğer mikroişlemci entegre devre işaretleri doğru ise, ROM bölümü yeniden bağlanabilir ve test yordamları ile önceden programlanmış bir ROM sistemin diğer parçalarını incelemekte kullanılır. Ön panel anahtarları tanı programının istenen kısmını seçmekte kullanılabilir. Daha sonra işaret analizleyici test yordamı çalışırken aletin her parçasındaki işaretleri kontrol etmekte kullanılabilir.

Göstergeleri olan bazı aletler bu göstergeleri tanı testlerinin sonuçlarını göstermekte kullanabilir. Örneğin, 5004A işaret analizleyicinin ROM'unda tanı yordamları bulunmaktadır, böylece kullanıcı herhangi bir anda birimin doğru çalıştığını kontrol edebilir.

Bu aletlerin gelişmişleri sürekli olarak kendilerini test eden ve hatanın tipini ve kaynağını kullanıcıya belirten aletlerdir. Problemi yumuşak bir sesle tanımlayan bir konuşma sentezleyici arıza gidermeyi daha ilginç yapabilir.

TEKRAR SORULARI VE PROBLEMLER

1. Bir *temel örneği* her seferinde bir modül olarak inşa etmenin *avantajı* nedir?
2. Geliştirme kartına ilave edilen 4 kilobaytlık *RAM* bloğunu test etme yöntemlerini açıklayınız.
3. *8085A* veya 6800 için 2000H-27FFH adresleri arasındaki *RAM*'ı test edecek kısa bir *çevirici dil* yordamı yazınız.
4. *Durdurma* noktası ve *tek-adımı* tarif ediniz.

5. Bir geliştirme kartı üzerindeki *I/O portunu* test etme yöntemini açıklayınız.

6. Bir program modülündeki *hata ayıklama* adımlarını sıralayınız.

7. Bölüm 12'deki terazi programında *BCDCVT* ve *BINCVT* yordamlarını test etmek için gerekli *çevirici dil* ilavelerini gösteriniz.

8. Bölüm 12'deki *EPROM programlayıcı* programında *OUTDS* yordamını test etmek için gerekli çevirici dil ilavelerini gösteriniz.

9. *Lojik-zaman analizleyici*, lojik-durum *analizleyici*, tetikleme öncesi, orta tetikleme ve tetikleme sonrası kayıtlarını tarif ediniz.

10. *Bir lojik-durum* analizleyici ile bir *CPU* ya da *CPU* grubunu test etme ve hataları ayıklama yöntemini tanımlayınız.

11. *Lojik-durum analizleyici* kullanarak *ROM, RAM* ve *I/O* portları test etme ve hataları ayıklama yöntemlerini açıklayınız.

12. Şekil 8.26'daki Intel Intellec gibi bir mikroişlemci geliştirme sistemin temel fonksiyonlarını tanımlayınız.

13. *Editör, çevirici dil, kaynak dosyası, sembol tablosu* ve *amaç* dosyasını tarif ediniz.

14. Bir program yordamındaki hataları ayıklamak için devre içi *emülatörün* nasıl kullanıldığını açıklayınız.

15. Bir aletin üretim sürecini, ilk tasarımdan bitirilmiş ürüne kadar gösteren bir akış diyagramı çiziniz.

16. Aşağıdaki teknisyenlerin herbirinin işlerini tarif ediniz : mühendislik, *kart testi, üretim testi, kalibrasyon, kalite kontrol* ve arazi servisi.

17. *Fluke 9010A* Mikrosistem Arıza Giderme gibi test aletlerinin temel avantajlarını belirtiniz.

18. İşaret analizini tarif ediniz ve mikroişlemci tabanlı bir alette *arıza gidermek için* nasıl kullanıldığını anlatınız.

REFERANSLAR / BİBLİOGRAFYA

M6800 MEK6800D2 Evaluation Kit II Manual, Motorola Semiconductor Products, Inc., Austin, TX, 1976.

Intel SDK-85 User's Manual, Intel Corporation, Santa Clara, CA, December 1977.

Intel In-Circuit-Emulator/85 Microcomputer Development System Operator's Manual, Intel Corporation, Santa Clara, CA, 1978.

Intel In-Circuit-Emulator/85 Microcomputer Development System Hardware Reference Manual, Intel Corporation, Santa Clara, CA, 1978.

Application Note 222-A Designer's Guide to Signature Analysis, Hewlett-Packard Company, Palo Alto, CA, April 1977.

Frohwerk, Robert A.: Signature Analysis: A New Digital Field Service Method, Hewlett-Packard Company, Palo Alto, CA, 1977.

Bechtold, Jim (ed.): "*Signature Analysis : A New Concept in Digital Troubleshooting*, "Hewlett-Packard Bench Briefs, vol. 17, no. 2,pp.1-4, March-April 1977; Hewlett-Packard Company, Palo Alto, CA.

Farly, Bruce G., "*The Role of Logic State Analyzers in Microprocessor Based Designs*, "Application Note 167-13 Data Domain Measurement Series, Hewlett-Packard Company, Colorado Springs, CO. Reprinted from 1975 Wescon Professional Program, September 16-19, 1975, Copyright 1975 Western Electronic Show and Convention.

"*Troubleshooting a Microprocessor*, " Logic Analyzer Application Note, no. 57K1.0, Tektronix, Inc., 1977.

Fluke 9000 Series Micro System Troubleshooters Data Sheet, John Fluke Mfg. Co, Everett, WA, 1981.

BÖLÜM 14

YÜKSEK-SEVİYELİ DİLLER

Önceki bölümlerde çevirici dil programlarının ve hatalarının ayıklanmasını tartışmıştık. Burada *BASIC, FORTRAN* ve *PASCAL* gibi yüksek-seviyeli dillerle programlamadan bahsedeceğiz. Aynı zamanda bu yüksek-seviyeli dillerin çevirici ve makine dilleri ile olan ilişkilerini göstereceğiz.

Bu bölümün sonunda siz

1. Genel amaçlı iletim sisteminin fonksiyonlarını tanımlayabileceksiniz.
2. Bir yorumlayıcının kullanımını tarif edebileceksiniz.
3. Bir derleyicinin kullanımını anlatabileceksiniz.
4. Birçok yüksek-seviyeli mikrobilgisayar dilini sayabileceksiniz.
5. Basit bir *BASIC* programını yazıp değiştirebileceksiniz.
6. Basit bir *FORTRAN* programını yazıp değiştirebileceksiniz.
7. Basit bir *PASCAL* programını yazabilecek veya değiştirebileceksiniz.

ÖZET

Bölüm 12'de çevirici dil programını mümkün olan durumlarda bağlanabilir bir seri modüller şeklinde yazmanın iyi bir fikir olduğunu söylemiştik. Örneğin, terazi programı birbirine bağlı altyordamların bir toplamıdır. Bu yaklaşımda en azından iki avantaj vardır. İlk olarak her yordam tek başına test edilebilir ve hataları ayıklanabilir. İkinci olarak, hataları ayıklanmış yordamlar bir dosya içinde tutulur ve ilerideki programlar için kullanılabilir. BCD'den ikiliğe, çarpma, ikilikten BCD'ye yordamları ana program içinde yazılabilir, fakat bağımsız altyordamlar olarak gelecekteki kullanımlar için daha kolay depolanırlar.

Çevirici dil programlarının daha hızlı yazılmasına yardım etmek için birçok mikroişlemci üreticileri kendi mikroişlemcileri için faydalı çevirici dil programları ve yordamlarının bir kütüphanesini tutarlar. Örneğin Intel, 8080A/8085A çevirici dil yordamları ve programlarının geniş bir kütüphanesine sahiptir. Yıllık bir ücret karşılığı müşteri bütün kütüphane programlarının listesini elde edebilir. Birçok kısa programın yazılı hali doğrudan listenin içinde yer alır. Daha uzun programlar kağıt şerit, flopy disk ya da küçük bir ücret karşılığı basılı olarak elde edilebilir.

Bunların yüksek seviyeli dillerle olan ilgisi şu şekildedir. *BASIC* gibi yüksek-seviyeli bir dille programlama, önceden ayarlanmış yordamlar kütüphanesi ile çalışmaya benzer. Yüksek seviyeli dille yazılmış programlarda yalnızca yordamın çalışması ya da fonksiyonu yazılır. Daha sonra bir derleyici programı ya da yorumlayıcı programı yüksek-seviyeli deyimleri yürütülebilecekleri eşdeğer makine dili yordamlarına çevirir.

Bir ideal yüksek-seviyeli dil, yazılan veya konuşulan İngilizce'nin iyi tanımlanmış bir şekli olabilir. Bölüm 11'de tanımlanan basit tek-kelimelik emirler dışında teknoloji henüz bu noktaya ulaşamamıştır. Fakat basit bir *BASIC* dili deyimi size bu dilin çevirici dilden daha çok İngilizce'ye benzediğini gösterecektir.

Bölüm 12'den terazi için yazılan çevirici dil programında paund başına fiyat ile ağırlığı çarparak toplam fiyatı veren yordam biraz karmaşıktır. BASIC te ise *LET P = W x S* deyimi bu çarpımı belirtir. Buradaki x çarpmayı göstermektedir.

YÜKSEK - SEVİYELİ DİL PROGRAMLARINI YAZMAK VE YÜRÜTMEK İÇİN KULLANILAN SİSTEMLER VE PROGRAMLAR

DONANIM: Yüksek-seviyeli dil programları genellikle büyük bir bilgisayar, minibilgisayar ya da mikrobilgisayar sistemleri üzerinde geliştirilmiştir. CPU ya ilaveten sistem 64 K veya daha çok RAM, bir CRT terminal, programları depolamak için bir flopy ya da sabit disk, basılı kopya elde etmek için bir yazıcı gerektirir. Şekil 8.26'daki Intellec Sistemi gibi çoğu mikrobilgisayar geliştirme sistemleri yüksek-seviyeli dil programlarını geliştirmekte kullanılabilir. Apple *II, IBM* veya *TRS-80* gibi kişisel bilgisayarlar da yüksek-seviyeli dil programlarını geliştirmekte kullanılır.

YAZILIM: Bir kullanıcının programlarını geliştirmesi ve çalıştırması için bir sistem, kullanıcının donanımla iletişimini sağlayan programlara sahip olmalıdır. En *basiti bir monitor programıdır.*

MONITOR PROGRAMI: Monitor programı muhtemelen sistemin ROM u içinde olur. SDK-85 ve MEK6800D2 geliştirme kartlarında olduğu gibi Monitor ROM'u sistemi ilk kullanıma hazırlayan yordamı içerir. Monitor ROM 'u aynı zamanda, kullanıcının teletayp ya da klavye kullanarak makine-dili programları belleğe yüklemesine, makine dili programları yürütmesine, kaydedicileri gözden geçirmesine, durdurma noktaları kurmasına imkân sağlar. Monitor programının temel bir fonksiyonu kullanıcının bir diskten sistem RAM'ına bir iletim sistemi yüklemesine izin vermektir.

İŞLETİM SİSTEMLERİ: Genel amaçlı bir işletim sistemi çalıştırıldığı zaman kullanıcının sistemin donanım detaylarını bilmeden yüksek seviyeli dil programları yaratmasına izin veren bir sistem programları ya da yordamları kümesidir. Örneğin, bir kullanıcı programı yazar ve verinin diskte nasıl saklandığını bilmeden flopy disk üzerinde saklayabilir. Kısaca bir işletim sistemi "kullanıcı dostu" yapar.

İşletim sistem yordamları ROM içinde olabilir, fakat genellikle sistem başlatıldığında bir diskten RAM' a yüklenirler. Bu şekilde yüklenen bir sistem disk işletim sistemi veya DOS olarak adlandırılır. RAM kullanımını azaltmak için bazı sistemler başlangıçta işletim sisteminin yalnızca temel kısımlarını yükler. Diğer yordamlar, gerektiği zaman diskten RAM 'a yüklenir.

İşletim Sistemi

1. Veri depolamak için disk dosyalarının yaratılması, kopyalanması, silinmesi ve değiştirilmesini,

2. Programları girilmesini, edit edilmesini ve diskler üzerine depolanmasını,

3. Ayrı geliştirilmiş program modüllerinin bağlanmasını,

4. Bağlanmış programların RAM 'a yüklenip yürütülmesini,

5. Programlardaki hataların ayıklanmasını sağlar. Hata ayıklama yordamı, durdurma noktalarının konulmasına ve bellek ve kaydedicilerin gözden geçirilmesine imkân verir. Bazı hata ayıklayıcıları, bellekteki makine kodlarını eşdeğeri olan çevirici dili komutlarına çeviren ters çeviricileri de içerir.

İşletim sistemleri aynı zamanda sistemin bellek bölgesini idare eder, ve verileri yazıcılara ve diğer çevre birimlerine transfer eder.

EDİTÖR PROGRAMLARI: Editör programları, kullanıcının kaynak programlarını veya yazıları RAM çalışma bölgesine kolayca girmesine izin verir. Editör, imleci CRT ekranda dolaştırıp yazıları veya program hatalarını düzeltmeyi kolaylaştıran komutlara sahiptir. Çoğu editörler bir programın herhangi bir yerinde klavyeden bir ya da *birkaç tuşa basarak* satırların eklenip, silinmesi imkânını sağlar. *Basit bir editör programı işletim sisteminin bir parçası olabilir.* Özel uygulamalar için olan daha güçlü editörler örneğin, kelime işlemciler ayrı olarak yüklenir.

Kullanıcının yüksek-seviyeli kaynak programı, tamamlandıktan ve düzeltildikten sonra bir disk üzerindeki dosyaya kopyalanır. Bu kaynak programının şekli mikrobilgisayarın komutu yürütmesi için gerekli olan ikilik makine kodundan çok farklıdır. Programın yüksek-seviyeli şekilden yürütülebilir koda çevrilmesi birkaç değişik yolla yapılabilir. Bir yol, yorumlayıcı kullanmaktır.

YORUMLAYICI PROGRAMLARI: Bir yorumlayıcı çalıştığında kaynak programının bir deyimini okur, her yüksek-seviyeli deyimi *makine koduna çevirir* ve bu *kodu anında yürütür.* Daha sonra bir sonraki yüksek-seviyeli dil deyimini okur, çevirir ve yürütür. Şekil 14.1 a, bunu akış diyagramı şeklinde göstermektedir.

Yorumlayıcının avantajı kullanıcı dostu olmasıdır. Yorumlayıcı doğrudan kaynak programını yürüttüğü için, bir program içindeki hataları bulup düzeltmek daha kolaydır. Eğer bir hata bulunursa, kullanıcı kaynak programını düzeltir ve yeniden çalıştırır.

Yorumlayıcının dezavantajı, yorumlanan programların derleyici tara-fından çevrilen aynı programdan 5 ile 25 kat daha yavaş çalışmasıdır. Bunun sebebi yorumlayıcının çalıştırılacak programın tam bir makine-kodu versiyonunu yaratmamasıdır. Yorumlayıcı, programın her deyimini programı her yürütüşünde çevirmek zorundadır. Diğer bir deyişle, bir yorumlayıcı da çevirme zamanı yürütme zamanının devamlı olarak bir parçasıdır. Bir sonraki bölümde tartışılacak derleyici ile karşılaştırabilmek için bunu hatırlayınız.

DERLEYİCİ PROGRAMLARI: Bir derleyici programı çalıştırıldığı zaman yüksek-seviyeli dil programının hepsini okur ve iki ya da daha fazla geçişten sonra bütün programı yeri değiştirilebilir makine koduna çevirir. Yeri değiştirilebilirin anlamı atlama adresleri ve çağırma adresleri gibi adreslere sembolik isimlerin verilmesidir, böylece program bellek içinde herhangi bir yere yerleştirilebilir.

Derleme tamamlandıktan sonra program bağlanmalıdır. Bunun anlamı kullanıcı programının ihtiyacı olan bazı matematiksel yordamlar veya I/O yordamları bir yordamlar kütüphanesinden okunmalı ve kullanıcı programının koduna ilave edilmelidir.

Bütün program modülleri bağlandıktan sonra yerleştirilmelidir. Bütün sembolik adresler mutlak adreslerle değiştirilmelidir. Bütün bu adımların sonucu, RAM içine yüklenebilen ve yürütülebilen bir makine dili programıdır.

Pascal dilinin bazı derleyicileri yüksek-seviyeli dil deyimlerini *P* kodu olarak adlandırılan bir ara koda çevirir. Daha sonra *P* kodu yorumlanır. Diğer derleyiciler yüksek-seviye dilini makinenin çalıştıracağı çevirici diline çevirir. *Çeviri diline derlemenin avantajı yüksek-seviyeli dil sözlüğünün parçası olmayan hataları ayıklanmış çevirici dil yordamlarının derleyiciden çıkışa ilave edilebilmesidir.* Sonuç, standart bir çevirici programı ile makine koduna çevrilir.

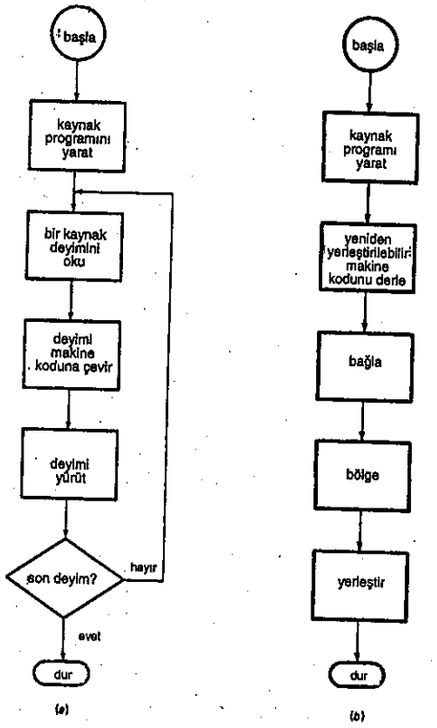
Şekil 14.1 b, bir yorumlayıcının çalışması ile karşılaştırabilmeniz için bir derleyicinin çalışmasını akış diyagramı şeklinde göstermektedir. Derleyici

yaklaşımının avantajı tüm programın bir defada çevrilmesidir. Üretilen makine-kodu versiyonu programın her çalışmasında yürütülür. Çevirme zamanı yorumlayıcıda olduğunun aksine yürütme zamanının bir parçası değildir. Bu sebepten; derlenmiş programlar, yorumlanmış programlardan 5 ile 25 kez daha hızla çalışırlar.

Derleyicilerin dezavantajı bir hata bulunduğunda ortaya çıkar. Hata kaynak programında düzeltilmeli ve daha sonra bütün program yeniden derlenmeli, bağlanmalı, yerleştirilmeli ve yüklenmelidir.

YAYGIN YÜKSEK-SEVİYELİ DİLLER

En azından 25 tane yaygın yüksek-seviyeli dil vardır ve BASIC gibi bazı dillerin sayısız diyalektleri bulunmaktadır. Bu yüzden burada en yaygın olan bazılarının kısa özetini vereceğiz.



Şekil 14.1 Derleyici ve Yorumlayıcı Çalışmasının Karşılaştırılması
***a)* Yorumlayıcı *b)* Derleyici**

FORTRAN: Fortran *FOR*mula *TRAN*slation kelimelerinin kısaltılmışıdır ve bu temel kullanımı için bir ipucu vermektedir. Bu dil karmaşık matematiksel işlemler için birçok fonksiyon içerdiğinden mühendislik hesaplamaları için idealdir. 1950'lerde IBM tarafından geliştirilmiştir, ve ilk yaygın bilgisayar dillerinden biridir. Bu dilin birçok versiyonu geliştirilmiştir. Yaklaşık 10 yıl için en popülerlerinden biri FORTRAN IV idi. 1977 yılında Amerikan Ulusal Standartlar Enstitüsü (ANSI) FORTRAN için bir standart yayımladı. Bu bütün üreticilerin dilin bir şeklini kullanmaları için bir girişimdi, böylece programlar taşınabilir olabilirdi. Taşınabilirin anlamı, bir makinede yazılan programların diğer birçok makinede çalışabilmesidir. ANSI standart FORTRAN 77 bu bölümün daha sonraki bir kısmında gösterilecektir. FORTRAN genellikle bir derleyici ile kullanılır.

Her ne kadar problem çözmek için etkili bir dil ise de ve mühendislik programları için büyük bir kütüphane mevcutsa da, FORTRAN programları yazarken gerekli formatlamadan dolayı yeni başlayanlara öğretmek için biraz zor bir dildir. Bu yüzden 1960'ların başında Darmouth kolejindeki iki profesör BASIC dilini geliştirdi.

BASIC: BASIC, FORTRAN kadar güçlü değildir, fakat kullanılması kolaydır ve birçok problem için yeterince güçlüdür. Birçok BASIC versiyonu geliştirilmiştir. Bunlar yalnızca 4 kilobaytlık bellek gerektiren ve ev bilgisayarların kullanılan çok düşük versiyonlardan bilgisayar sistemlerinde çalışan ve 32 kilo baytlık bellek gerektiren genişletilmiş versiyonlar arasında değişir. Daha sonra standart BASIC ile nasıl programlama yapıldığını göstereceğiz. BASIC genellikle bir yorumlayıcı ile kullanılır, fakat *BASIC derleyicileri* de mevcuttur.

ALGOL: Bir diğer mühendislik-yönlendirilmiş dil ALGOL'dür. İsim ALGOrithmic Language'in bir kısaltılmışıdır. Avrupa'da ve bazı üniversite bilgisayar merkezlerinde yaygındır, fakat FORTRAN'ın baskın olmasından dolayı *endüstride yaygın değildir.*

COBOL: COBOL, Common Business Oriented Language kelimelerinin kısaltılmışıdır. İsminden de anlaşılacağı gibi bu *dil hesaplar, maaşlar gibi hesap tutma işlerini kolayca yapmak için geliştirilmiştir.* COBOL program deyimleri semboller yerine birçok kelime kullanır. Örneğin bir COBOL deyimi SUBTRACT WITHDRAWALS FROM OLD BALANCE GIVING NEW BALANCE (yeni dengeyi bulmak için, eski değerden tutulmaları çıkar) şeklinde olabilir. İngilizce'ye benzer deyimler COBOL programlarını programcı olmayanlar için daha anlaşılır yapar.

PL/1 ve PL/M: IBM tarafından geliştirilen PL/1, FORTRAN 'dan daha güçlü olan geniş bir dildir. Büyük bir dil olduğundan, *çok büyük bir derleyici gerektirir.* Bu yüzden yalnızca belli büyüklükteki sistemlerde çalışabilir.

Birçok kullanıcının PL/1 'ın bütün gücüne ihtiyacı olmadığından birçok alt küme dili geliştirilmiştir. Bunlardan biri PL/M'dir ve Intel tarafından geliştirilmiştir. PL/M için bir Intel derleyicisi Şekil 8.26 daki Intellec sistemi ile birlikte PL/M programlarını 8080A/8085A makine koduna çevirmek için kullanıldı. PL/M, kullanıcının 8080A programlarını çevirici dilinden daha hızlı yazmasını direk olarak sağladı. PL/M dilinin dezavantajı, birçok uygulamalarda PL/M ile yazılmış program derlendiğinde çevirici dille yazılan programdan daha uzun makine kodu üretmesidir. Bunun anlamı, derlenen programın daha fazla bellek bölgesine ihtiyacı olmasıdır.

APL: IBM tarafından geliştirilen APL dili, kullanıcıya komutları tanımlama izni verdiği için çok güçlüdür. Aynı zamanda çok geniş bir kütüphaneye ve önceden tanımlanmış fonksiyonlara sahiptir. Fakat, çalıştırmak için büyük bir makine gerektirir ve standart daktilo terminallerinde bulunmayan birçok sembol kullanılır. APL deyimleri oldukça şifrelidir ve bu yüzden APL programlarının takip edilebilmesi için birçok açıklamaya ihtiyaç vardır.

PASCAL: Pascal, Fransız matematikçi Blaise Pascal anısına bu ad verilen ve 1970 lerin başında İsviçre'de Nichlaus Wirth tarafından geliştirilen bir dildir. FORTRAN 'dan daha modüler ve BASIC 'ten daha kullanışlıdır. Yakın geçmişte *Pascal'a olan ilgi çok derli toplu bir dil olmasındandır.* Böylece bir yorumlayıcı veya derleyici küçük bir sistem içine sığabilir. Pascal derlendiği zaman çok verimli makine-kodu programları üretir. Derlenmiş bir Pascal Programı FORTRAN veya BASIC ile derlenmiş aynı programdan birkaç kez *daha hızlı* çalışır.

Pascal'ın organizasyonu yapısal programları yazmayı kolaylaştırır. Yapısal programlamada, yalnızca bazı iyi tanımlanmış yapı ve bloklara izin verilir. İdeal olarak, her yapının yalnızca bir giriş noktası ve bir çıkış noktası vardır. Bu yüzden her yapı bağımsız olarak yazılabilir ve hataları ayıklanabilir. Bu, çevirici dil programları için anlatılan modüler yaklaşımın genişletilmiş şeklidir. Yapısal programlama kuralları, programı hataların zor düzeltildiği "mantıksal spagetti" haline çeviren bir yerden başka bir yere atlamaları yasaklar.

ADA: Ada A.B.D. Savunma Bakanlığı tarafından geliştirilen hayli yapısal bir dildir. Pascal, ALGOL ve PL/1 dillerinin en iyi özelliklerini içerir ve kontrol işlemlerinin gerektirdiği şekilde, gerçek-zaman çok işlemlidir. *Ada, karmaşık alt yordamlara benzeyen yazılım bileşenleri kavramı üzerine kurulmuştur.* Bileşenler ayrı ayrı derlenir, kataloglanır ve bir program kütüphanesi içinde depolanır. Kütüphane, yalnızca özel bir uygulama için gerekli bileşenlerin seçilmesi ile kişiye özel yapılabilir. Daha sonra derleyici bu bileşenleri derleme sırasında gerektiği zaman çağırır. Ada tamamen geliştirdiğinde, donanımtasarımcının büyük ölçekli entegre cihazları kataloglarındaki bileşenleri kullanıp donanım sistemlerini geliştirmesine benzer şekilde, programcının yazılım elemanları kataloğunu kullanarak karmaşık programları geliştirmesine izin verecektir.

BASIC İLE PROGRAMLAMA

BASIC, programların kolaylıkla yazılabilmelerinden dolayı en popüler yüksek-seviyeli dillerden biri olmuştur. Daha az pahalı olan ve tüm BASIC kapasitesine sahip ev bilgisayar sistemleri APPLE II, TRS-80 ve H-8'in geliştirilmesiyle daha da çok popüler olmuştur. *Ne yazık ki her şirket dilin kendine ait bir versiyonunu geliştirmiştir.* En azından 100 çeşit BASIC versiyonu vardır, bunlar çok küçük sistemlerden, orijinal Darmouth BASIC versiyonundan daha güçlü ve çok bellek kullanan gelişmiş versiyonlara kadar olan bir aralıktadır. Bu yüzden, eğer bir makinede BASIC çalışıyorsanız o makinenin özelliklerini öğrenmek için elkitabını okumalısınız. Bu bölümde orta yollu bir BASIC versiyonu anlatılacaktır.

PROGRAM FORMATI

Şekil 14.2'de, terminalden girilen iki sayıyı toplayan ve sonucu terminale basan basit bir BASIC programı gösterilmiştir. Çalıştırıldığı zaman program kullanıcının toplanacak *iki sayıyı (A ve B)* girmesini bekler. İki sayı girildikten sonra bunları toplar ve *Y (toplam)* sayısını basar. Program daha sonra geri döner ve kullanıcının bir sonra eklenecek iki sayıyı girmesini bekler. Programın, çevirici dilde yazılan programdan çok daha fazla İngilizce'ye benzediğini görebilirsiniz.

SATIR NO	DEYİM
210	INPUT A, B
220	LET Y = A + B
230	PRINT "Y ="; Y
240	GOTO 210

Şekil 14.2 İki Sayıyı Toplayan ve Toplamı Basan Basit BASIC Programı

Şekil 14.2 de gösterildiği gibi her BASIC deyimi istenen yürütme sırasını gösteren bir satır numarasına sahiptir. Satır numaraları 0 ile 9999 arasındaki herhangi bir sayı olabilir. Örneğin satır numaraları 1, 2, 3, 4 vs. olabilir. Fakat, bu şekilde peşpeşe numaralar kullanılır ve bir deyim dışarıda bırakılırsa, bu deyimi sokabilmek için bütün satır numaralarının değiştirilmesi gerekir. Bu yüzden Şekil 14.2 de gösterildiği gibi, deyimlere 5 veya 10 arttırılan satır numaraları verilir. Bu durumda deyimler minimum bir düzenleme ile aralara sokulabilir.

BASIC deyimleri INPUT, LET veya PRINT gibi yapılacak işle başlar. Deyimin ikinci kısmı yapılacak işin amacını verir. Daha ileride, yaygın BASIC deyimlerinin listesini verecek ve tartışacağız.

SABİTLER, DEĞİŞKENLER VE SAYILAR

BASIC dilinde hem sabitlere, hem de değişkenlere sembolik isimler verilir. Sembolik isim, alfabedeki herhangi bir harf veya harf tarafından takip edilen 0 ile 9 arasındaki sayı olabilir. Şekil 14.2 de gösterilen programda kullanılan *A* kullanıcı tarafından girilecek bir değişkeni ve *B* ikinci değişkeni temsil etmektedir. Bu iki sayının toplamı *Y* harfi ile temsil edilmektedir. LET C = 12 gibi bir deyim *C* olarak adlandırılmış bir değişken ya da sabite 12 değerini vermektedir. Değişkenler ve sabitler için diğer uygun isim örnekleri *P6, V9, R3* ve *H* olabilir. BASIC dilinin bir eleştirisi daha uzun ve böylece daha anımsatıcı sabit ve değişken isimlerine izin vermemesidir.

BASIC değişik formattaki sayıları tanır ve yazar. Sayılar tamsayı veya gerçek sayılar olabilir. Tamsayılar ondalık kısmı olmayan sayılardır. Örnek olarak 6,6.00 ve -9.0 verilebilir. Gerçek sayılar ondalık kısmı olan sayılardır. Örnekler 5.26, -7.02 ve 0.0136 olabilir. BASIC çalıştırabilen çoğu makine altı ya da yedi basamaklı sayılara izin verir. Binleri gösteren virgüllere izin verilmez. Çok büyük ve çok küçük rakamlar bilimsel gösterimle temsil edilirler. Örneğin 120.000.000.000.000 sayısı 1.2E13 ile ve .0000123 sayısı 1.23E-5 ile temsil edilir. E harfi, kendisini takip eden sayının 10 sayısının üssü olduğunu belirtir.

YAYGIN ARİTMETİK OPERATÖRLER

BASIC dili birçok aritmetik operatör kullanır. Tablo 14.1 yaygın operatörleri ve bunlar için BASIC dilinde kullanılan sembolleri göstermektedir. Toplama ve çıkarma sembolleri aritmetik sembollerle aynıdır. Çarpma için kullanılan sembol yıldız işaretidir. (*) Çünkü *X*, değişken ya da sabit ismi *X* ile karşılaştırılabilir. BASIC dilinde çarpmayı göstermek için iki büyüklük arasında mutlaka yıldız işareti kullanılmalıdır. Bölme kesme (/) sembolü ile gösterilir, çünkü çoğu terminalde ÷ sembolü yoktur.

Üs alma bir sayının kuvvetini alma işlemidir. *LET X = R ↑2* deyiminin anlamı $X = R^2$ dir. Üs tamsayı olmak zorunda değildir, böylece bu operatör bir sayının köklerini bulmakta da kullanılır. Örneğin *LET X = R ↑.5, R* değişkeninin kare kökünü üretir ve *LET X = R ↑.3333*, X için *R* değişkeninin küp kökünü bulacaktır.

TABLO 4.1 YAYGIN TEMEL ARİTMETİK OPERATÖRLERİ

Öncelik	Sembol	İşlem
1	( )	*Gruplama*
2	↑	*Üs alma*
3	*	*Çarpma*
3	/	*Bölme*
4	+	*Toplama*
4	—	*Çıkarma*

Parantezler, değişkenler veya sabitleri gruplandırmak için kullanılır. Böylece uygun işlemler yerine getirilir. Örneğin *LET Y= A/B+C* ile *LET Y=A/(B+C)* deyimlerinin değişik anlamları vardır. Eğer bir BASIC deyiminde birden çok operatör varsa, işlemler belirli bir sıraya göre yürütülür. Parantez içindeki işlemler devamlı ilk olarak yerine getirilir. *Y = A/(B+C)* önce *B* ile C'yi toplar ve sonucu A'yı bölmek için kullanır. Diğer işlemler gösterilen sırada yerine getirilir. Örneğin *LET Y = A+B*C* deyiminde, *B* ile C çarpılır ve sonuç *A* ile toplanır. *LET Y = C-R ↑2*3.14159* deyiminde, *R*'nin karesi alınır, daha sonra 3.14159 ile çarpılır ve Y değerini elde etmek için sonuç C sayısından çıkarılır. Eğer bir deyimde aynı öncelikli operatörler kullanılırsa, bunlar soldan sağa doğru yerine getirilirler. *A/B*C* deyimi A'yı B'ye böler ve sonucu *C* ile çarpar. *B*C* çevresine parantezleri koymak önce çarpma işleminin yapılmasına ve bölme işleminin daha sonra yapılmasına sebep olur.

YAYGIN İLİŞKİ OPERATÖRLERİ

Aritmetik operatörlere ilaveten, BASIC Tablo 14.2 de gösterilen ilişki operatörlerini de kullanır.

TABLO 14.2: BASIC DİLİNDEKİ İLİŞKİ OPERATÖRLERİ

SEMBOL	ANLAM
=	*Atama veya eşit*
>	*'den büyük*
<	*'den küçük*
<> veya #	*Eşit değil*
,	*Ayırma*

Eşit sembolünün iki kullanımı vardır. Bir tanesi normal aritmetik anlamı olan eşit, diğeri ise atama anlamıdır. Bu ikinci anlama bir örnek *LET A = A+1* deyimidir. Normal aritmetikte bu imkansızdır. Fakat bu deyimin BASIC dilindeki anlamı *A*'nın depolanmış değerini *A + 1* ile değiştir demektir. Diğer bir deyişle, *A*'nın depolanmış değerini bir arttırmaktır.

Diğer ilişki operatörleri kendilerini açıklayıcı niteliktedir. Deyimler içinde *IF A>B THEN* (satır numarası) şeklinde kullanılırlar. Programın bu adımı yürütülürken A, B'den büyük ise, program yürütülmesi deyim sonundaki satır numarasına atlayacaktır.

KÜTÜPHANE FONKSİYONLARI

Basit aritmetik operatörlerin yanısıra BASIC daha karmaşık birçok matematiksel fonksiyona sahiptir. Tablo 14.3, yaygın olarak mevcut olan fonksiyonları ve tanımlarını göstermektedir.

TABLO 14.3

BASIC DİLİNDE YAYGIN OLARAK MEVCUT FONKSİYONLAR

FONKSİYON	MATEMATİKSEL FORM	TANIM
ABS (X)	$\|X\|$	X'in mutlak değeri
ATN (X)	arctan X	Tanjantı X olan radyan cinsinden açı
COS (X)	cos X	X'in kosinüsü (X radyan olarak)
EXP (X)	e^X	Doğal logaritma tabanının x. üssü
INT (X)	[X]	X'ten küçük en yakın tamsayı
LOG (X)	In X	X'in doğal logaritması
RND (X)		0 ve 1 arasında rasgele sayı, 0 ve 1 dahil değil
SİN (X)	sin X	X'in sinüsü (X radyan olarak)
SQR (X)	$\sqrt{X}$	X'in kare kökü
TAN (X)	tan X	X'in tanjantı (X radyan olarak)

SIN, COS ve TAN, X açısının radyan cinsinden olmasını ister. ATN bir açının değerini radyan cinsinden verir. EXP *(X)* doğal logaritmanın tabanı olan e sayısının X. üssünü hesaplar. LOG *(X)*, X'in doğal, ya da e tabanında logaritmasını hesaplar. ABS *(X)*, X'in mutlak değerini verir. SQR *(X)*, X'in karekökünü, *LET Y = X↑.5* deyiminden daha hızlı hesaplar. Geri kalan iki kütüphane fonksiyonunun biraz daha fazla açıklamaya ihtiyacı vardır.

INT (X) bir sayıyı yuvarlamanın bir şeklidir. Bir sayıyı devamlı olarak en yakın küçük (daha negatif) tamsayıya yuvarlar. Örneğin, *INT* (6.7) deyimi 6 ve *INT*(-6.7) deyimi -7 verir. En yakın tamsayıya yuvarlama ise parantez içindeki büyüklüğe 0.5 ilave etmekle elde edilir. *INT* (6.7 + 0.5) deyimi 7 ve *INT* (-6.7 + 0.5) ise -7 sonucunu verir.

RND (X), 0 ile 1 arasında rasgele bir sayı, örneğin 0.123456 üretir. 0 ve 1 değerleri hiçbir zaman üretilmez. Birçok BASIC sistemi bu fonksiyonun doğru şekilde çalışması için X'in değerinin 0 olmasını gerektirir. Bazı makinelerde reset'ten sonra rasgele sayılar aynı diziyi takip ederler, bu yüzden sayılar gerçekten rasgele değildir. 0 ile 1'den farklı bir sayı arasındaki rasgele sayılar basitçe *RND (X)* fonksiyonunu bir sabit ile çarpmakla elde edilir. *LET A=50*RND(0)*, 0 ile 50 arasındaki rasgele bir sayı üretir. *LET A = INT (6*RND(0)) + 1*, 1 ve 6 arasında rasgele bir sayı üretir, bu da zar oyunu için faydalıdır.

BASIC DEYİMLERİ

Şekil 14.2 deki basit program, üç BASIC deyimi sundu. Tablo 14.4, bu deyimlerin ve yaygın olarak kullanılan diğer BASIC deyimlerinin listesini örneklerle göstermektedir.

TABLO 14.4

ÖRNEKLERLE BASIC DEYİMLERİ

```
REM, BU BASIC DEYİM ÖRNEKLERİNİN BİR LİSTESİDİR
LET Y = A* X↑2 + B* X + C
LET P = P + 1
LET C = SQR (A↑ 2 + B↑ 2)
PRINT N
PRINT "LÜTFEN ÇEVİRME SAYISINI YAZINIZ"
PRINT A, B, C
```

```
PRINT "CEVAP"; N
INPUT A
INPUT A, B, C
GOTO 990
GOSUB 720
RETURN
STOP
END
IF C3 = > Y4 THEN 210
IF Y = 7 THEN 420
IF C3 = > Y4  THEN 210 ELSE 990
IF SQR (X) = 8 THEN 290
FOR A = 1 TO 20
.
.
.
NEXT A
FOR P = 2 TO 20 STEP 2
      LET Y = 5* P* (C + 3)
      PRINT "P ="; P, "Y ="; Y
NEXT P
DATA 0.2, 3.1, 2.0, 1.8, 0.9
READ R1, R2, R3, R4, R6
```

Burada her bir deyimin anlamını tartışacak ve program yazmak için operatörlerle nasıl kullanıldıklarını göstereceğiz.

Tablo 14.4 teki ilk deyim REM, açıklama için kullanılmaktadır. REM'den sonra yazılan herhangi bir şey basılır fakat program çalışırken ihmal edilir. Çevirici dil programlarında olduğu gibi, çok sayıda açıklama programın bir süre kullanılmadıktan sonra yeniden kullanılması durumunda kolayca anlaşılabilir olmasına yardım eder.

LET deyimi, bir program içindeki değişken veya sabitlere değer atamak için kullanılır. Belki de BASIC programları içinde en çok kullanılan deyimdir.

PRINT, o satırda kendisini takip eden büyüklüklerin bir yazıcı ya da terminalde basılmasına sebep olur. PRINT N, N'nin nümerik değerini basar. Tırnak işaretleri arasındaki herşey yazıldığı gibi basılır. Örneğin, PRINT "CAT" deyimi CAT harflerini basar. Virgüller ve noktalı virgüller bir satırdaki basılı büyüklükleri formatlamakta kullanılır. Bir yazıcı aletindeki her satır, her biri 15 boşluktan oluşan dört bölgeden oluşur. Basıma bir bölge içindeki en solda olan boşluktan başlanır. PRINT A, B, C deyimi A değerini birinci bölgede, B değerini ikinci bölgede, ve C değerini üçüncü bölgede basar. Bu format veri tablolarını yaratmakta kullanışlıdır. Eğer bir PRINT deyimindeki büyüklükler arasında noktalı virgül kullanılırsa, büyüklükler aynı bölge içinde ardarda basılırlar. PRINT *A*; ", " ;*B*; "," ; *C* *A, B* ve *C*'nin değerlerini yalnızca virgüllerle ayrılmış olarak basar.

INPUT çalıştırıldığında program kullanıcının terminalden bir değer girmesini bekler. Kullanıcının bir değer girmesi gerektiğini belirtmek için terminalde bir soru işareti basılır. Eğer INPUT deyimi INPUT *A, B, C* gibi birden çok büyüklük içeriyorsa program devam etmeden önce her bir büyüklük için değerlerin girilmesini bekler.

GOTO (satır numarası), belirtilen satır numarasına şartsız olarak atlamaya sebep olur. *GOSUB* (satır numarası), belirtilen satır numarasından başlayan bir altyordamı çağırır. Bir GOSUB deyimi yürütüldüğü zaman bir dönüş adresi depolanır. Böylece eğer altyordamın sonuna bir RETURN deyimi konulursa altyordamın bitmesinden sonra ana programa geri dönülür. Çevirici dil programlarında olduğu gibi çok kullanılan yöntemleri altyordam olarak yazmak bellek bölgesinden tasarrufu sağlar.

Bir STOP deyimi ve bir *END* deyiminin her biri programın yürütülmesini durdurur. Bir programın en son deyimi olarak konulan END deyimi kontrolu monitore göre verir ve bilgisayarın bellekteki rasgele verileri yürütmeyi denemesini engeller. STOP deyimi program içindeki herhangi bir yere konulabilir ve END deyiminin satır numarası bilinmediği zaman GOTO (END) deyiminin yerine kullanılır. Çok uzun programlarda bu mümkündür.

IF... THEN (satır numarası) deyimi iki ya da fazla büyüklüğün karşılaştırılmasına göre şartlı atlamalar üretir. Tablo 14.2 deki herhangi bir ilişki operatörü kullanılabilir. IF C3 = > Y4 THEN 210 deyimi eğer C3 'ün değeri Y4 'ün değerine eşit veya daha büyükse program yürütülmesini 210 nolu satıra atlatır. Bazı BASIC versiyonları IF... THEN (satır numarası) ELSE (satır numarası) şeklinde bir deyime sahiptir. Bu deyim iki hedef satırdan birini belirtir. Bazı BASIC versiyonları, AND (VE), OR(VEYA) ve NOT (DEĞİL) gibi çok kullanışlı ilişki operatörlerine sahiptir. Örneğin, bu ope-

ratörlerle IF C3>7 AND Y2 = 3.2 THEN 320 gibi bir deyim kullanılabilir. Bu deyim, yalnızca C3'ün değeri 7'den daha büyük ve Y2'nin değeri 3.2'ye eşit olduğu zaman 320 nolu satıra atlamaya sebep olur. Eğer OR operatörü kullanılırsa iki şarttan yalnızca birinin doğru olması ile 320 nolu satıra atlama yapılabilir.

Bir *FOR ...NEXT* döngüsü, bir işlemi belli sayıda üstüste yerine getirmek için kullanılır. Tablo 14.4 teki ilk örnekte FOR deyimi ve NEXT deyimi arasın-daki deyimler 20 kez yürütülür ve daha sonra program, NEXT *A* deyiminden sonraki deyime ilerler. Döngünün her geçişinde *A*'nın değeri 1 arttırılır.

Tablo 14.4 teki ikinci FOR ... NEXT döngüsü bir başka kullanımdır. FOR deyiminin sonundaki STEP 2 döngünün her geçisinde *P* değişkeninin 2 arttırıldığını belirtir. Gösterilen kısa program parçası, *P*'nin 2'den 20'ye olan ve 2şer artan değerleri bu değerler için hesaplanan *Y* değerlerini basar. P'nin başlangıç ve bitiş değerleri, ve adım büyüklüğü tamsayı olmak zorunda değildir. Örneğin, FOR N = 2.5 TO 9.6 STEP 0.01 veya FOR N = 9.6 TO 2.5 STEP -0.01 gibi FOR deyimleri yazılabilir.

DATA ve READ deyimleri birlikte birçok LET deyimi yerine kullanılırlar. 5 günlük sürelerde yağan ortalama yağmuru bulan bir program yazılmakta olduğunu varsayalım. Her gün yağan yağmuru girmek için bir LET deyimi kullanılabilir. Örneğin LET R1= 0.2, LET R2 = 3.1, LET R3 = 2.0 gibi. Daha basit bir yol, Tablo 14.4 teki örnekte gösterilen READ ve DATA deyimlerini kullanmaktır. Veri değerleri, DATA kelimesinden sonra istenen sırayla yazılır ve virgülle ayrılır. READ ve DATA deyimleri aynı sayıda elemana sahip olmalıdır. Yürütüldükleri zaman READ deyimi her sembole karşı gelen değeri DATA deyiminden alır ve o sembole atar.

Burada bazı yaygın BASIC operatörleri, fonksiyonları, ve deyimlerinin ilk tanımlamaları tamamlanmaktadır. Bu kadar küçük bir küme ile birçok BASIC programı yazılabilir. Şimdi, BASIC dilinin temellerini daha iyi gösterebilmek için bazı program örnekleri göstereceğiz.

PROGRAM ÖRNEKLERİ

Bu noktada terazi programının ya da EPROM programlayıcı programının parçalarının BASIC dilinde nasıl yazılacağını göstermek uygun olabilir. Bu kolaylıkla yapılamaz. Bazı durumlarda mikroişlemci tabanlı akıllı aletlerin programları BASIC dilinde yazılır ve daha sonra aleti çalıştırmak için derlenir. Fakat çoğu BASIC versiyonları bu gibi programları yazmak için I/O, kesme ve bit işleme kabiliyetine sahip değildir. Örneğin, standart BASIC birkaç porttan giriş ve çıkışa izin vermez.

Bazı gelişmiş BASIC versiyonları bu problemi portlardan okuma ve portlara yazma deyimleri ilave ederek çözdüler. Bazı gelişmiş versiyonlar gerekli çevirici dil yordamlarının BASIC programı içine ilave edilmesine izin verir. Bu, bazı uygulamalar için en iyi programlama tekniğidir, bit işlemleri için çevirici dil ve karmaşık matematiksel işlemler için ise BASIC kullanılır. Terazi ve EPROM programlayıcı programları bu şekilde yazıldı; fakat sonunda hepsi ya da çoğu çevirici dilde yazıldı. Bu yüzden, bu bölümdeki örnekler BASIC dilinin diğer kullanımlarını göstermektedir.

Çoğu BASIC programı, etkileşimli şekilde yazılır. Yani kullanıcı bir bilgisayar ya da mikrobilgisayara bağlı terminal başında oturur ve programı girer. Program bir oyun programı olabilir, bir mühendislik ya da matematik problemini çözmek olabilir, ya da parasal defter tutma işlerinden biri olabilir. Program çalıştıkça kullanıcı programın gerektirdiği şekilde verileri girerek, soruları cevaplayarak ya da direktifler vererek programla etkileşir. Böylece kullanıcı ile makine arasında bir diyalog yerine getirilir. Burada gösterilen BASIC programı bu tipte bir programdır.

PARA ATMA: Şekil 14.3, bir parayı belli sayıda atmak (kullanıcı tarafından yazılan sayıda) ve tura sayısı ile yazı sayısını basmak için yazılan basit bir programı göstermektedir. Yazıların sayısı *T* ve turaların sayısı *H*, başlangıçta 20 ve 30 nolu satırlarda 0'a eşitlenir. Okuyucu için direktif 31 nolu satırdaki PRINT deyimi ile basılır. Bir PRINT deyiminde tırnak işaretleri arasındaki yazıların aynen yazıldığını hatırlayınız. Satır numaralarını 10 aralıklı olarak kullanmak, programcının diğer satırların numaralarını kaydırmadan bu satırı eklemesine izin vermiştir.

INPUT *F* deyimi yürütüldüğü zaman makine, kullanıcı bir sayı girinceye kadar bekler. F için sayı girildikten sonra program "para atma" FOR ... NEXT döngüsünü F defa kullanır. Döngünün her geçişinde, 60 nolu satırdaki LET R=RND (0) deyimi 0 ve 1 arasında rasgele bir sayı üretir. Eğer sayı 0.5'ten küçük ise program 100 nolu satıra atlar ve tura sayısını bir artırır. Eğer sayı 0.5'ten büyük ise program 80 nolu satıra gider ve yazı sayısını bir arttırır. Her iki durumda da uygun sayıcılar arttırıldıktan sonra bir sonraki atma yapılır. Bütün atmalar yapıldıktan sonra tura sayısı ve yazı sayısı basılır. Şekil 14.3 b'deki örnek çalıştırmadan görüldüğü gibi tırnak işaretleri arasındaki yazı aynen yazılır. Tırnak işareti olmayan bir değişkenin değeri basılır. Ayırıcı olarak bir noktalı virgül kullanılmasının anlamı o değişkenin değerinin hemen eşit işaretinden sonra basılması demektir. Burada kullanılan bir virgül, değişkenin değerinin bir sonraki bölgeye basılmasına sebep olur.

140 nolu satırdaki GOTO 20 deyimi programı bir sonsuz döngü içine sokar. Bir defa başlayınca kullanıcı makineyi resetleyen bir kaçış ya da kırılma tuşuna basıncaya kadar defalarca çalışır. Her defasında kullanıcıdan atış sayısını girmesi istenir.

A.
```
10 REM PARA ATMA PROGRAMI
20 LET T = 0
30 LET H = 0
31 PRINT "İSTENEN ATIŞ SAYISINI YAZINIZ."
40 INPUT F
50 FOR N = 1 TO F
60 LET R = RND (0)
70 IF R < . 5 THEN 100
80 LET T = T + 1
90 GOTO 110
100 LET H = H + 1
110 NEXT N
120 PRINT "YAZI SAYISI ="; T
130 PRINT "TURA SAYISI ="; H
140 GOTO 20
```

B.
```
İSTENEN ATIŞ SAYISINI YAZINIZ
? ---- ÖRNEĞİN 50 SAYISINI GİRİNİZ
YAZI SAYISI = 27
TURA SAYISI = 23
İSTENEN ATIŞ SAYISINI YAZINIZ
? -------
```

Şekil 14.3: Para atma için BASIC Programı ***a)*** **Program listesi** ***b)*** **Örnek Çalışma**

FAHRENHEIT SICAKLIKLARI CELCIUS VE KELVİN'E ÇEVİRME

Şekil 14.4a, bir seri Farhrenheit sicaklığı Celcius ve Kelvin eşdeğeri sıcaklıklara çeviren kısa bir program göstermektedir. Her çevirmenin sonucu en yakın onda bir dereceye yuvarlanır ve Şekil 14.4*b* de gösterildiği gibi basılır.

Çevrilecek Fahrenheit sıcaklıkları 15 ve 20 nolu satırlardaki DATA deyimlerinde içerilmektedir. Not: Çok fazla ilaveler beklenmiyorsa 5 artmalı

satır numaraları kullanılabilir. READ F deyimi DATA deyiminden bir sayı okur ve 30 nolu satırdaki LET deyimi Celcius eşdeğerine çevirir. Bu LET deyimi yürütüldüğü zaman ilk olarak F değerinden 32 çıkarır, sonucu 5 ile çarpar ve en son olarak bu sonucu 9 ile böler. Celcius değeri veride gösterilen basamak sayısından daha çok ondalık yere sahip olabilir; bu yüzden her sonuç bir derecenin en yakın onda birine yuvarlanır. Bu 40 nolu satırdaki LET deyimi içindeki INTEGER fonksiyonu ile yapılır.

Sonuç en yakın onda bir değerine yuvarlanacağı için yukarıda hesaplanan C değeri 10 ile çarpılır. INT *(X)* fonksiyonunun az önceki özet anlatımında gösterildiği 0.5 ilave edilir; böylece INT fonksiyonu yerine getirildiğinde hem pozitif hem de negatif sayılar en yakın tamsayıya yuvarlanır. Yuvarlamadan sonra ondalık noktayı yerine koyabilmek için sonuç 10 ile bölünür. Bir örnek bu işlemi açıklamaya yardım edecektir. 46.3° Fahrenheit sıcaklığı 7.94444°C Celcius'a karşı gelir. 10 ile çarpmak 79.4444 ve 0.5 eklemek 79.9444 verir. INT fonksiyonu bunu 79'a yuvarlar ve 10 ile bölmek 7.9°C sonucunu verir.

A.
```
5 REM FAHRENHEİT'İ CELCIUS VE KELVİN'E ÇEVİRME
10 PRINT "FAHRENHEIT", "CELSIUS", "KELVİN"
15 DATA 7.2, 31.0, 46.3, 53.3, 64.7
20 DATA 73.6, 84.2, 94.7, 98.6, 99.2
25 READ F
30 LET C = (F - 32)* 5/9
35 REM EN YAKIN ONDA BİR DERECEYE YUVARLA
40 LET C = INT ((10* C) + . 5)/10
45 LET K = C + 273.2
50 PRINT F, C, K
```

B.

FAHRENHEIT	CELSIUS	KELVIN
7.2	-13.8	259.4
31	-.6	272.6
46.3	7.9	281.1
53.3	11.8	285
64.7	18.2	291.4
73.6	23.1	296.3
84.2	29	302.2
94.7	34.8	308
98.6	37	310.2
99.2	37.3	310.5

Şekil 14.4: Fahrenheit Sıcaklıkları Celcius ve Kelvine Çevirmek için BASIC programı *a)* Program listesi *b)* Örnek Çalıştırma

Kelvin sıcaklık eşdeğeri, 45 nolu satırda yuvarlanmış *C* değerine 273.2 toplanarak bulunabilir. 50 nolu satırdaki PRINT deyimi, *F*, *C* ve *K* değerlerini Şekil 14.4b de gösterildiği gibi üç farklı bölgede basar. Çünkü ayırıcı olarak virgül kullanılmıştır. Önden gelen ve takip eden sıfırların yazılmadığına dikkat ediniz.

50 nolu satırdan sonra bir END deyimi olmadığı için program geri döner ve 15 nolu satırdaki DATA deyiminden bir sonraki değeri okur ve sonuçları basar. Birinci DATA deyimindeki değerler kullanıldıktan sonra, ikinci DATA deyimindeki değerler okunur, çevrilir ve sonuçlar basılır. Bütün değerler kullanıldıktan sonra, çoğu makine programı bitirir ve Şekil 14.4b de gösterildiği gibi 25 nolu satırdaki READ deyiminin okuyacağı veri kalmadığını belirten bir mesaj yazar. Programı bitirmenin daha ilginç bir yolu DATA deyiminin sonuna imkânsız bir değer örneğin -500 koymak olabilir. Bu değerin kontrolu her READ deyiminden sonra bir IF ... THEN (satır numarası) deyimi ile yapılabilir. İlave edilen satır numarası -500 değeri okunduğunda END deyimine atlamaya sebep olur.

Bu program birçok yönde genişletilebilir. Her DATA deyimine daha fazla değer ilave edilebilir, ve gerektiği kadar DATA deyimi de ilave edilebilir. READ ve DATA deyimleri yerine, istenilen aralıktaki her 0.1°C için bir çevirme tablosu yapmak için bir FOR... NEXT döngüsü kullanılabilir. FOR F = 0 TO 100 STEP 0.1 deyimi, 0 ile 100°F arasındaki 1000 sayı için F, C ve K değerlerini

basar. Bu, insanların sıkıcı ve tekrarlı buldukları cinsten bir iştir, fakat bilgisayarlar bunu kolaylıkla ve hassas olarak yapar.

ÇEK DEFTERİ KAYDI: Önceden de bahsedildiği gibi BASIC finansal kayıt tutma işlerini yapmak için de kullanılabilir. Şekil 14.5 yazılan çeklerin, yatırılan paraların ve hesabın şu andaki durumunun dökümünü yapan bir programı göstermektedir. Format, bir çek defterinde tutulan kayıtlara benzemektedir.

Kullanıcıdan ilk olarak geçen ay sonundaki hesap durumunu girmesi istenir. Daha sonra kullanıcıya çek ya da yatırma makbuzu numarasını, tarihi, ve yatırılan ya da çekilen paranın miktarını girmesi direktifi verilir. Miktarın başındaki eksi işareti çekilen parayı belirtmektedir. Yatırılan para için işaret gerekmez. Her numara, tarih ve miktar girişi yapıldıkça program bunları tablo şeklinde yazar, yeni hesap durumunu hesaplar ve uygun sütunda basar. Şekil 14.5b, birkaç giriş için program dökümünü göstermektedir. Eğer hesap durumu negatif olursa bir uyarı mesajı basılır. 110 ve 120 nolu satırlarda *D*'den sonra dolar işareti kullanıldığına dikkat ediniz. Bu *D*'yi bir dizi değişkeni olarak tanımlar ve tarihin bildiğimiz ay/gün/yıl formatında girilmesine izin verir. Bu *C* ve *A* gibi normal değişkenlerle mümkün değildir.

A.

```
 70 REM ÇEK DEFTERİ İŞLEM KAYIT PROGRAMI
 80 PRINT "GEÇEN AY SONUNDAKİ HESAP DURUMUNU GİR"
 90 INPUT B
100 PRINT "İŞLEM NUMARASINI, TARİHİNİ, MİKTARI GİR"
110 INPUT C, D$, A
120 PRINT C, D$, A
121 LET B = B + A
122 IF B < 0 THEN 150
130 PRINT "YENİ HESAP DURUMU" , , B
140 GOTO 110
150 PRINT "HOP, FAZLA PARA ÇEKİYORSUN",
151 PRINT "YENİDEN KONTROL ET"
160 GOTO 80
```

```
B. GEÇEN AY SONUNDAKİ HESAP DURUMUNU GİR
? 365.27
İŞLEM NUMARASINI, TARİHİNİ, MİKTARI GİR
? 25,  12/1/78,  -65.92
25              12/1/78   -65.92
YENİ HESAP DURUMU    299.35
? 26, 12/7/78 ,  221.73
26              12/7/78   221.73
YENİ HESAP DURUMU    521.08
? 27,  12/14/78,  -530.72
27              12/14/78  -530.72
HOP, FAZLA PARA ÇEKİYORSUN,
YENİDEN KONTROL ET
GEÇEN AY SONUNDAKİ HESAP DURUMUNU GİR
?
```

Şekil 14.5 Çek defteri Kaydı için BASIC Programı
***a)* Program Listesi *b)* Örnek Çalıştırma**

KARAKTER DİZİLERİ

BASIC, sayıların yanısıra harflerin ve karakterlerin dizisiyle de çalışabilir. Bir karakterler dizisi sembolik olarak $ işareti tarafından takip edilen harfle temsil edilir. Bu bir dizi değişkeni olarak adlandırılır. Dizi değişkenleri bir kelime veya cümlenin basit bir sembolle temsil edilmesine izin verir. Örneğin, bir program başlangıcında LET Y$ = "EVET", LET N$ = "HAYIR", LET E$ = "BİR HATA YAPTIN" deyimleri olabilir.

Bu durumda programcı program içinde bu kelime veya cümleleri kullanmak istiyorsa yalnızca dizi değişkeni kullanılmalıdır. Örneğin, PRINT E$, "BİR HATA YAPTIN" cümlesini basar. Dizi değişkenleri aynı zamanda emirlerin sayı yerine kelimelerle girilmesine imkân sağlarlar. Şekil 14.6, bir oyun sonunda oyuncunun yeniden oynamak isteyip istemediğini belirlemek için dizi değişkenlerinin nasıl kullanılacağını göstermektedir.


```
915 DIM A$ (5)
920 DIM Y$ (4)
925 DIM N$ (5)
930 DIM E$ (15)
935 LET Y$ = "EVET"
940 LET N$ = "HAYIR"
945 LET E$ = "BİR HATA YAPTIN"
950 PRINT "OYUNU YENİDEN OYNAMAK İSTİYOR MUSUNUZ?"
955 PRINT "EVET VEYA HAYIR YAZ"
965 INPUT A$
970 IF A$ = Y$ THEN (OYUNUN BAŞLANGIÇ NUMARASI)
975 IF A$ = N$ THEN 995
980 PRINT E$
985 PRINT
990 GOTO 955
995 END
    READY
```

Şekil 14.6 BASIC İçinde Dizilerin Kullanılması Örneği

Boyut deyimleri, DIM, her dizi için gerekli bellek miktarını belirtir. Bazı makineler bir dizinin uzunluğunu 72 karakter ile sınırlandırır, diğer makineler her dizi için 255 karaktere kadar izin verir. Şekil 14.6 deki programda her dizi değişkeni boyutlandırılıp, tanımlandıktan sonra kullanıcının girdiği A$ dizisi Y$ ile karşılaştırılır. Eğer girilen dizi tanımlanan Y$ dizisi ile tamamen aynı ise, Program yürütülmesi oyun programının başlangıcına atlar. Eğer A$, Y$ dizisine eşit değilse, N$ ile karşılaştırılır. Tam bir eşitlik program yürütülmesini 995 nolu satırdaki END deyimine gönderir. Eşitlik olmaması programı hata mesajı olan E$ dizisini basmaya gönderir ve soru tekrarlanır. 985 nolu satırdakine benzer boş bir PRINT deyimi dökümün daha iyi gözükmesi için boş bir satır bırakır.

Dizilerin bir diğer kullanımı, yalnızca belirli şifreleri yazan kişilerin bilgisayara erişmesine izin vermektir. Burada da girilen karakterler önceden tanımlanmış dizi değişkeni ile karşılaştırılır. Eğer iki değişken eşleşirse bilgisayar *"READY" (hazır)* mesajını basar. Aksi halde bilgisayar *"Kusura bakma, bu şifre beni çalıştırmıyor"* mesajını basar.

DİZİLER

BASIC dilindeki bir diğer faydalı özellik diziler ile çalışabilme yeteneğidir. Matematikte bir sayı dizisi ya da sütunu tek-boyutlu dizi olarak adlandırılır. Bu dizinin boyutu listenin uzunluğudur. İki ya da daha fazla sütunlu listeler iki-boyutlu diziler olarak tanımlanır. Bir satranç tahtası iki-boyutlu bir dizidir. Şekil 14.4b deki tablo verisi de iki-boyutlu bir dizidir. Üç-boyutlu diziler iki-boyutlu dizilerin iki ya da daha fazla sayfalarından oluşan ya da üç-boyutlu tic-tac-toe oyununa benzer yapılardır.

BASIC dilinde bir diziyi adlandırmak için tek harf kullanılır. Herhangi bir boyutunda 10 değerden fazla değeri olan bir dizide bellek yeri ayırmak için DIM deyimi kullanılması gerekir. Örneğin DIM X (27) deyimi tek boyutlu X dizisi için 27 bellek yeri ayırır. BASIC alt indisleri normal şekilde yazamadığı için bir dizi içindeki elemanlar ya da değerler X(1), X(2), X(3), ..., X(27) olarak tanımlanırlar. DIM Y(12, 4) 12 satır ve 4 sütunlu, iki-boyutlu bir dizi tanımlar. İki-boyutlu bir dizide her eleman bir satır numarası ve bir sütun numarası ile tanımlanır. Örneğin Y(1,2), birinci satır ve ikinci sütundaki elemanı belirtir. Benzer şekilde üç-boyutlu bir dizideki bir eleman, satır numarası, sütun numarası ve sayfa numarası ile tanımlanır.

Diziler çok güçlü vasıtalardır ve çoğu kullanım örnekleri burada göstermek için çok uzundur. Şekil 14.7, 300 veri değerini ya da test sonucunu giren, azalan sırayla dizip basan bir programın listesini göstermektedir. Bu program tek-boyutlu bir dizi kullanır ve dizilerin kullanımları hakkında küçük bir fikir verir.


```
 90 REM TEST SONUÇLARI DİZİSİNİ GİRMEK
100 DIM X (300)
110 PRINT "TEST SONUÇLARI SAYISINI GİR"
120 INPUT N
130 PRINT "HER SATIRDA 5 TEST SONUCU GİR"
140 PRINT "SONUÇLARI VİRGÜL İLE AYIR"
150 FOR I = 1 TO N STEP 5
160 INPUT X (I), X(I + 1), X(I + 2), X(I + 3), X(I + 4)
170 NEXT I
180 PRINT
190 REM SONUÇLARI AZALAN SIRAYLA DİZMEK
200 FOR I = 1 TO N - 1
210     FOR J =  I + 1 TO N
220     IF X(I) > X(J) THEN 260
230     LET T = X(I)
240     LET X(I) = X(J)
250     LET X(J) = T
260     NEXT J
270 NEX I
280 PRINT
310 PRINT "AZALAN SIRAYLA SONUÇLAR"
320 PRINT
330 FOR I = 1 TO N STEP 10
340 PRINT X(I); X(I + 1); X(I +2); X(I + 3); X(I +4); X(I +5)
350 PRINT X (I + 6); X(I + 7); X(I + 8); X(I +9)
360 PRINT
370 NEXT I
380 END
```

Şekil 14.7 300 veri değerini ya da test sonucunu girmek ve azalan sırayla dizip basmak için yazılmış BASIC programı

Programın başındaki DIM X (300) deyimi dizinin maksimum büyüklüğünü 300 eleman yapar. Gerekirse boyut deyimi daha büyük olabilir, fakat belleği gereksiz olarak meşgul etmenin anlamı yoktur. Sıralanacak sonuçların sayısını girdikten sonra, kullanıcıya her satırda virgüllerle ayrılmış 5 sonuç girmesi söylenir. Program çalıştığında bu sonuçlar bellekte DIM X (300) deyimi ile ayrılan yerlere sırasıyla depolanırlar. *N* sonucun hepsi girildikten sonra program, sonuçları azalan sırayla dizme bölümüne gider.

Sıralama bölümü şu şekilde çalışır. İlk sonuç dizideki ikinci sonuçla karşılaştırılır. Eğer ilk sonuç ikinciden büyükse, birinci yerinde kalır ve

üçüncü sonuç ile karşılaştırılır. Eğer birinci sonuç, ikinci sonuçtan küçük ise ya da ikinci sonuca eşitse *T* olarak adlandırılan geçici bir depolama yerine taşınır. İkinci sonuç birinci pozisyona taşınır ve daha sonra birinci sonuç *T* yerinden dizi içindeki ikinci pozisyona taşınır. Diğer bir deyişle, 230 ile 250 arasındaki satırlar kâğıtları elle karıştırmaya benzer şekilde iki sonucun sıralarını değiştirir. Bu şekilde program listedeki her sonuçla birinci pozisyondaki sonucu karşılaştırır ve büyüklüklerine bağlı olarak yerlerini değiştirir veya değiştirmez. I, FOR... NEXT döngüsünün birinci çalışmasından sonra en büyük sayı dizinin birinci pozisyonuna taşınmış olur. I, FOR... NEXT döngüsünün ikinci çalışması ise dizi içindeki ikinci en büyük sayıyı 2 nolu pozisyona yerleştirir. Bu döngünün her çalışmasında bir sonraki en büyük sayı, dizinin bellek içindeki bir sonraki pozisyonuna yerleştirilir. Eğer sıralama yordamını takip etmekte zorlanıyorsanız, 5 veya 10 sayılık bir dizi kullanarak yordamı çalıştırınız.

Bütün sonuçlar azalan sırada olduğunda, bir başlık basılır. Daha sonra sonuçlar her satırda 10 tane olmak üzere basılır. Her satırda 10 sonuç basarak bütün dizi kolaylıkla görülebilecek şekilde bir sayfaya basılır.

Program kolaylıkla her test sonucu ile birlikte bir öğrenci numarası içerecek şekilde değiştirilebilir. Bölümün bu kısmında BASIC ile yapılabileceklerin küçük bir parçasını gösterebildik. Daha çok bilgi ve örnek için bölümün sonundaki referanslara danışınız.

FORTRAN İLE PROGRAMLAMA

FORTRAN, belli bir anlamda BASIC dilinin atası olduğundan birçok yönden birbirlerine çok benzerler. FORTRAN daha güçlüdür ve ilk bakışta biraz daha zor gelebilir. Fakat sistematik olarak yaklaşılırsa FORTRAN, öğrenmek için BASIC dilinden daha zor değildir.

Her ne kadar FORTRAN en eski yüksek seviyeli dillerden biriyse de, bazı sebeplerden dolayı hâlâ en popüler olanlarından biridir. FORTRAN dilinde yazılmış çok büyük mühendislik **programları kütüphaneleri** mevcuttur ve FORTRAN programlarını çalıştırmak için birçok bilgisayar kurulmuştur. Hatta Intel Şekil 8.2b da gösterilen 8080A tabanlı geliştirme sisteminde çalışacak güçlü FORTRAN-80 derleyicisini geliştirmiştir. Intel FORTAN-80 aşağıda tartışılacak ANSI FORTRAN 77 üzerine kurulmuştur.

VERİ TİPLERİ

TAMSAYI: Tamsayı (ya da sabit nokta) sabitlerinde ondalık nokta yoktur. Pozitif veya negatif olabilirler. Örnekler +3, -726 ve 92 olabilir. Çoğu bilgisayarlar bir tamsayı sabitinin büyüklüğünü ± 32000 ile sınırlandırırlar. Bir programda 999'den büyük sayılar yazıldığında binleri ayırmak için virgül kullanılmaz.

Bir tamsayı değişkenine (yalnızca tamsayı değerler alabilen) genellikle alfabedeki I'dan N'ye kadar olan harflerden biri ile başlayan bir isim verilir. İsim en fazla altı harfe, ya da harf ve sayıya sahip olabilir. Tamsayı değişken isimlerine örnek olarak INCOME, NUMBER ve MAX gösterilebilir.

GERÇEK SAYILAR: Gerçek (ya da kayar nokta) sayıları bir tamsayı kısmı ve bir ondalık kısmından oluşur. 3.7, . 0003, ve 3.14159 gerçek sayı örnekleridir. Her gerçek sayı Bölüm 7'de gösterilen kayar nokta formatlarından biri ile 4 bellek baytına depolanır. Çok büyük ve çok küçük sayılar bilimsel gösterimde yazılabilir. Örneğin *3.7E6* gösteriminin anlamı $3.7x10^6$ *ve 2.15E-16* gösteriminin anlamı $2.15x10^{-6}$ dır.

Gerçek değişkenlere genellikle tamsayı değişkenler için ayrılmış harflerin dışındaki bir harfle başlayan isimler verilir. Bu isimlerde en fazla altı harfe veya harf ve sayıya sahip olurlar, ve bir harfle başlamak zorundadırlar.

MANTIKSAL VE KARAKTER VERİSİ: Değişkenler ve sabitler mantıksal tipte veya karakter tipinde olabilir. Mantıksal veriler yalnızca *"doğru"* veya *"yanlış"* değerlerine sahiptir.

Karakter verisi, uzunluğu 1 ile 255 arasında olan alfabe harfleri veya sembollerin dizisidir. Karakter verisi her iki ucundaki tek tırnak işareti ile sonlandırılır. Örneğin 'GİRİLEN VERİ KANUNİ ARALIK DIŞINDADIR' bir karakter verisidir. Mantıksal veya karakter verilere de en fazla altı harf ve sayıdan oluşan isimler verilebilir.

DİZİLER VE DOSYALAR: FORTRAN, verilerin diziler ve dosyalar içinde çalışılabilmesine izin verir. Bir dizi en fazla yedi boyutlu olabilir ve dizi içindeki herhangi bir yerdeki herhangi bir değere koordinatlarını belirterek erişilebilir. Örneğin, *FARNHT (2,3)* iki-boyutlu bir dizide satır 2 ve sütun 3'teki değeri tanımlar.

İç dosyalar temelde seri depolama şeklindedir. Veriler sıralı olarak yazılmalı ve okunmalıdır. Bunu hatırlamanın bir yolu teyp ve disk dosyalarını düşünmektir.

ARİTMETİK, İLİŞKİ VE MANTIK OPERATÖRLERİ

Tablo 14.5, FORTRAN operatörlerinin bir listesini ve önceliklerini göstermektedir. En yüksek öncelikli operatör listenin en üstündedir. BASIC dilinde olduğu gibi ilk olarak en yüksek öncelikli operatörler icra edilirler. Üs almada BASIC'te kullanılan ↑yerine iki yıldız işareti kullanılmaktadır. *FORTRAN*'daki Z = X ** 2 + Y ** 2 deyimi $X^2 + Y^2$'ye eşittir. Diğer aritmetik semboller BASIC ile aynıdır.

FORTRAN'daki ilişki operetörleri > sembolü yerine bir çift harf kullanmaktadır. Eğer 6800 'ün bağıl dallanma anımsatıcılarını hatırlıyorsanız bunlar tanıdık gelecektir .LT. "den az"ı temsil etmektedir. .LE. ise "az veya eşit"i .EQ. "eşittir"i, .NE. "eşit değildir"i, .GT. "den büyük"ü ve .GE. "den büyük veya eşit"i temsil etmektedir. Harflerden önceki ve sonraki noktalara dikkat ediniz.

Mantıksal ve boole operatörleri mantıki değişkenlerle, çoğunlukla da IF deyimleriyle kullanılır. Örneğin IF (A.OR.B) deyimi eğer mantıksal değişken A doğru veya mantıksal değişken B doğru ise (A.OR.B)'den sonra yazılan deyimi yürüt demektir. Burada .EQV.'ın anlamı "e eşdeğer" ve .NEQV. 'n anlamı "e eşdeğer değil" dir. "A.EQV.B"nin anlamı A ve B'nin aynı mantıksal durumda (doğru veya yanlış) olduğudur.

ALT YORDAMLAR VE FONKSİYONLAR

Aritmetik operatörlere ilave olarak FORTRAN, bir sayının kare kökünü almak gibi fonksiyonları yerine getiren iç alt yordamlara sahiptir. Bunlar öz fonksiyonlar olarak adlandırılırlar. Tablo 14.6 yaygın olarak kullanılan bazı fonksiyonları ve şekillerini göstermektedir. Bu kütüphanedeki bir fonksiyon bir deyim içinde kullanılarak çağrılabilir. Örneğin C = SQRT (A**2 + B**2) yöntemin derlenmesi sırasında otomatik olarak kare kök alma fonksiyonunu çağırır.

TABLO 14.5

AZALAN ÖNCELİK SIRASINA GÖRE FORTRAN OPERATÖRLERİ

ÖNCELİK	SEMBOL	TANIM
1	( )	Gruplama
2	**	Üs alma
3	*	Çarpma
3	/	Bölme
4	+	Toplama
4	-	Çıkarma
5	.LT.	'den az
5	.LE.	den az veya eşit
5	.EQ.	eşit
5	.NE.	eşit değil
5	.GT.	'den büyük
5	.GE.	den büyük veya eşit
6	.NOT.	Mantıksal değil
7	.AND.	Mantıksal ve
8	.OR.	Mantıksal veya
9	.EQV.	Mantıksal eşitlik
9	.NEQV.	Mantıksal eşit değil

TABLO 14.6 FORTRAN'IN ÖZ FONKSİYONLARI (INTEL)

ŞEKİL	FONKSİYON
SIGN (a1, a2)	*a2*'nin işaretini *a1*'e taşı
IDIM (a1, a2)	*a1-a2*>0 ise bu sayı, aksi halde 0 döner (tamsayı)
DIM (a1, a2)	*a1-a2* >0 ise bu sayı, aksi halde 0 döner (gerçek)
MAXO (a1, ..., an)	Listedeki en büyük tamsayıyı bulur
AMAXI(a1, ..., an)	Listedeki en büyük gerçek sayıyı bulur
AMAXO (a1, ..., an)	Listedeki en büyük sayıyı bulur, gerçek sayı olarak döner
MAXI (a1, ..., an)	Listedeki en büyük gerçek sayıyı bulur, tamsayı olarak döner
MINO (a1, ..., an)	Listedeki en küçük tamsayıyı bulur
AMINI (a1, ..., an)	Listedeki enküçük gerçek sayıyı bulur
AMINO (a1, ..., an)	Listedeki en küçük tamsayıyı bulur, gerçek sayı olarak döner
MINI (a1, ..., an)	Listedeki en küçük gerçek sayıyı bulur, tamsayı olarak döner
SQRT (a)	*a*>0 için $\sqrt{a}$ hesaplanır
EXP (a)	$e^{**}a$ hesaplanır
ALOG (a)	*a* > 0 için log *(a)* hesaplanır.
ALOG 10 (a)	*a* > 0 için log10 *(a)* hesaplanır
SIN (a)	*a*'nın sinüsü hesaplanır
COS (a)	*a*'nın kosinüsü hesaplanır
TAN (a)	*a*'nın tanjantı hesaplanır
ASIN (a)	*a*'nın ters sinüsü hesaplanır
ACOS (a)	*a*'nın ters kosinüsü hesaplanır
ATAN (a)	*a*'nın ters tanjantı hesaplanır
ATAN2 (a1, a2)	*a1/a2*'nin ters tanjantı hesaplanır
SINH (a)	*a*'nın hiperbolik sinüsü hesaplanır
COSH(a)	*a*'nın hiperbolik kosinüsü hesaplanır
TANH (a)	*a*'nın hiperbolik tanjantı hesaplanır
INT (a)	*a*'yı tam sayıya çevirir
IFIX (a)	*a*'yı tam sayıya çevirir
REAL (a)	*a*'yı gerçek sayıya çevirir
FLOAT (a)	*a*'yı gerçek sayıya çevirir
ICHAR (a)	*a*'yı tam sayıya çevirir
AINT (a)	*a*'yı tam sayıya kes
ANINT (a)	en yakın bütün sayıya yuvarla (gerçek)
NINT (a)	en yakın tamsayıya yuvarla
IABS (a)	*a*'nın mutlak değerini hesapla (tamsayı)
ABS (a)	*a*'nın mutlak değerini hesapla (gerçek)
MOD (a1, a2)	*a1/a2*'den kalanı hesapla (tam sayı)
AMOD (a1, a2)	*a1/a2*'den kalanı hesapla (gerçek)
ISIGN (a1, a2)	*a2*'nin işaretini *a1*'e taşı

PROGRAM ŞEKLİ VE DEYİMLER

PROGRAM ŞEKLİ: FORTRAN deyimlerini tartışmadan önce, bir FORTRAN programının şekline bakmak iyi bir fikirdir. Şekil 14.8 e, girilen Fahrenheit sıcaklıkları Celcius ve Kelvin'e çevirip sonuçları basan kısa bir FORTRAN programını göstermektedir. FORTRAN programları genellikle 72 sütun olarak düzenlenmiş kâğıtlar üzerine yazılır. Her sütun bir karakterin girilebileceği bir boşluğu temsil eder, bu yüzden her satırda maksimum 72 karakter olabilir. İlk beş sütun etiketler veya bir C için ayrılmıştır. FORTRAN, BASIC'te olduğunun aksine her satır için bir satır numarası gerektirmez. (Şekil 14.8a da derleyici tarafından en sola konulan satır numaraları gereksizdir). Buna rağmen, bir GOTO, DO, veya diğer deyimlerle çağrılan herhangi bir satıra bir ile beş basamaktan oluşan bir etiket verilir. Birinci sütundaki bir C bu satırın bir açıklama satırı olduğunu belirtir.

Altıncı sütun çoğunlukla bir ayırıcı olarak hizmet verir. Eğer bir FORTRAN deyimi bir satırdan daha uzun ise 6. sütuna konulan bir devam karakteri bu satırın içeriğinin bir önceki satırın devamı olduğunu belirtir. Deyimler ve açıklamalar satırın geri kalan 66 boşluğuna yazılır.

Bir FORTRAN programında isteğe bağlı olan ilk deyim PROGRAM deyimidir. Bu deyim programa altı karaktere kadar olan bir isim verir. PROGRAM FCNVRT buna bir örnektir. Bundan sonra değişik deyimler kullanılabilir. Bir programda bundan sonra gelen deyimler çoğunlukla TYPE deyimleridir.

TİP DEYİMLERİ: TİP deyimleri bir değişken ya da sabit ismi tarafından belirtilen verinin tipini onaylar veya değiştirir. Örneğin INTEGER deyimi NUMBER değişkeninin yalnızca tamsayılı değerleri alabileceğini belirtir. NUMBER, N ile başladığı için otomatik olarak tamsayı büyüklük olarak tanımlanır, deyim yalnızca onu onaylar. Fakat, KELVIN isimli gerçek bir değişkenin istendiğini varsayalım. KELVIN 'in başındaki K bu değişkeni tamsayı olarak tanımlar. Buna rağmen REAL KELVIN deyimi tamsayı tabanlı bu atamayı kaldırır ve KELVIN gerçek değerler alabilir.

Diğer TIP deyimleri mantıksal ya da karakter tiplerini tanımlar. Örneğin LOGICAL PASS deyimi PASS değişkenin mantıksal tipte olduğunu belirtir. CHARACTER * 15 NAME ise NAME değişkeninin enfazla 15 karakteri olan bir karakter değişkeni olduğunu belirtir. Hatta gerekmediği durumlarda dahi, TİP deyimleri programın başında dökümantasyon amacıyla bütün sabit ve değişkenleri tipleri ile birlikte göstermek için kullanılırlar.

Veri dizileri DIMENSION deyimi ile tanımlanır. Örneğin CHESS (8, 8); sekiz satır ve sekiz sütundan oluşan iki-boyutlu bir diziyi tanımlar. DIMENSION FARNHT (100), 100 elemanlık bir-boyutlu bir diziyi tanımlar.

ISIS-II FORTRAN-80 VI 0 COMPILATION OF PROGRAM UNIT FCNVRT OBJECT MODULE PLACED IN: F1: FCNVRT. OBJ

COMPILER INVOKED BY: FORT 80: F1: FCNVRT. SRC

```
1          PROGRAM FCNVRT
       C   FAHRENHEIT SICAKLIKLARI CELCIUS VE KELVINE
       C   ÇEVİRİR
       C   GİRİŞ CİHAZI KONSOL= BİRİM 5
       C   ÇIKIŞ CİHAZLARI SATIR YAZICI= BİRİM 4, KONSOL= BİRİM
       C   6
       C   BİRİM 4= :LP: ÇALIŞMA ZAMANINDA BELİRTİLECEKTİR.
2          REAL CLSIUS, KELVIN, FARNHT
3          INTEGER NUMBER
4          DIMENSION FARNHT (100)
5          WRITE (6, 9)
6        9 FORMAT ('FAHRENHEIT'IN CELCIUS VE KELVIN'E
         * ÇEVRİLMESİ')
7          WRITE (4, 10)
8       10 FORMAT ('1 FAHRENHEIT'IN CELCIUS VE KELVIN'E
         * ÇEVRİLMESİ')
9       11 WRITE (6, 12)
10      12 FORMAT ('ÇEVRİLECEK SICAKLIK SAYISINI GİRİNİZ')
11      13 WRITE (6,14)
12      14 FORMAT ('MAKSIMUM SAYI 100. 10'DAN KÜÇÜK SAYI İÇİN
         * 00N YAZINIZ. 10'DAN BÜYÜK SAYI İÇİN 0NN YAZINIZ')
13         READ (5, 20) NUMBER
14      20 FORMAT (I3)
15         IF (NUMBER. GT. 100) GOTO 11
       C   SAYI ÇOK BÜYÜKSE KOMUTLARI TEKRARLAYINIZ.
16         WRITE (6, 25)
```

```
17    25  FORMAT ('BÜTÜN FAHRENHEIT, DEĞERLERİNİ VİRGÜL
         * İLE AYIRARAK YAZINIZ')
18        READ (5, *) (FARNHT (I), I = I, NUMBER)
      C   DEĞERLERİ ALINIZ VE "FARNHT" DİZİSİ İÇİNE DEPOLAYI-
      C   NIZ
19        WRITE (6, 40)
20        WRITE (4, 40)
21    40  FORMAT (2X, 'FAHRENHEIT', 4X, 'CELSIUS', 4X,
         * 'KELVIN',/)
      C   TABLO BAŞLIĞINI YAZ VE SATIR ATLA
22        DO 60, I = 1, NUMBER
23        CLSIUS = (ANINT ((FARNHT (I) -32)* 50/9) )/10
      C   CELCIUS SICAKLIĞI EN YAKIN .1 DERECEYE YUVARLAN-
      C   MIŞ OLARAK HESAPLAYINIZ
24        KELVIN = CLSIUS + 273.2
25        WRITE (6, 50) FARNHT (I), CLSIUS, KELVIN
26        WRITE (4, 50) FARNHT (I), CLSIUS, KELVIN
27    50  FORMAT (4X, F6.1, 6X, F6.1, 4X, F6.1)
28    60  CONTINUE
29        END

MODULE INFORMATION:

     CODE AREA SIZE      = 0417H   1047 D
     VARIABLE AREA SIZE  = 01CAH    458 D
     MAXIMUM STACK SIZE  = 000AH     10 D
     39 LINES READ

                              0 TOTAL PROGRAM ERROR (S)

0 PROGRAM ERROR (S) IN        END OF FORTRAN COMPILATION
PROGRAM UNIT FCNVRT
```

(a)

FAHRENHEIT'İN CELCIUS VE KELVINE ÇEVRİLMESİ

FAHRENHEIT	CELSIUS	KELVIN
-32.0	-35.6	237.6
5.0	-15.0	258.2
100.0	37.8	311.0
98.6	37.0	310.2
0.0	-17.8	255.4

(b)

Şekil 14.8 ***a)*** **Fahrenheit sıcaklıkları Celcius ve Kelvine çeviren Fortran Programının Derleyici Listesi** ***b)*** **Örnek Çalıştırma**

ATAMA DEYİMLERİ: Atama deyimleri BASIC dilindeki LET deyimlerine benzemektedir. Bunlardan ilki standart aritmetik deyimlerine çok benzeyen aritmetik atama deyimleridir. Örnek olarak *K*=7, *P*= 3.22, *C*= SQRT (*x***2 + Y**2) verilebilir.

Atama deyimleri, mantıksal ve karakter büyüklüklerine de değer vermekte kullanılabilir. Örneğin LUVTAX = 'FALSE' veya NAME ='HALL' gibi

DATA DEYİMLERİ: DATA deyimleri değişkenlere, dizilere ve dizi elemanlarına başlangıç değerlerini veren birçok atama deyimleri yerine kullanılır. Örneğin, DATA *I, J, K*/5, 10, 15/ deyimi *I*'ya 5, *J*'ye 10 ve *K*'ya 15 başlangıç değerlerini verecektir.

GOTO DEYİMLERİ: GOTO deyimleri program yürütülmesinin şartsız atlamalarına sebep olur. En basit GOTO deyimi GOTO (deyim etiketi) şeklindedir. Bu deyim program yürütülmesinin deyim içindeki etiketle belirlenen satıra atlamasına sebep olur. FORTRAN'daki bir etiketin 1 ile 9999 sayıları arasındaki sayılardan meydana geldiğini hatırlayınız.

Bir diğer GOTO ise, GOTO (stl, stl, stl, ...) **ifade** şeklindedir. Burada Stl bir deyim etiketini ve ifade bir tamsayı ifadeyi belirtmektedir. GOTO (110, 220, 330) K deyimi program yürütülmesinin *K* = 1 iken 110 nolu deyime, *K* = 2 iken 220 nolu deyime ve *K* = 3 iken 330 nolu deyime atlamasına sebep olur.

IF DEYİMLERİ: FORTRAN, program yürütülmesinin şartlı atlamasına sebep olan değişik IF deyimlerine sahiptir. En basit IF deyimi yukarıda anlatılan GOTO deyimine benzeyen IF (ifade) stl1, stl2, stl3 şeklindedir. Burada ifade bir matematiksel ifadeyi stl ise bir deyim etiketini gösterir. IF (*B* + *C* - *A*) 110, 220, 330 deyimi sonucunda program yürütülmesi *B* + *C* - *A*'nın sonucu sıfırdan küçükse, 110 nolu deyime atlar. Eğer ifadenin sonucu sıfıra eşitse program

```
20          FORMAT...
          ELSE IF (LINE. EQ.2) THEN
            WRITE (6,20) FEDTAX
40          FORMAT...
          ELSE IF (LINE. EQ.3) THEN
            WRITE (6, 60) STATAX
60          FORMAT...
          ELSE
            WRITE (6, 80) FICA
80          FORMAT
          END IF
          RETURN
          END
ANA PROGRAMIN BİR SONRAKİ DEYİMİ
```

Şekil 14.10 Bir Altyordama Dönüştürülmüş IF Blokları

READ, WRITE VE FORMAT DEYİMLERİ: Bu üç deyim önemlidir, çünkü bir kart okuyucudan, konsoldan ya da diğer bazı birimlerden veri girişi için ve konsol ya da satır yazıcıya veri çıkışı için kullanılırlar. Belki de aynı zamanda BASIC dilinin geliştirilmesinin temel sebebidir.

Tipik bir *READ* deyimi *READ (BIRIM#, stl) (değişken, değişken, ...)* şeklindedir. Birim numarası, programın çalıştırıldığı sistemdeki bir giriş cihazına verilen numaradır. Örneğin Şekil 14.8a daki programın çalıştırıldığı İntel sisteminde konsol, 5 nolu birim olarak numaralandırılmıştır. Stl bir *FORMAT* deyiminin deyim etiketini temsil eder. *READ* (5, 20) *NUMBER* deyiminin anlamı *NUMBER* değişkeni için *BIRIM* # 5'ten bir değer girilmesi ve bu değeri *NUMBER* olarak adlandırılan bellek yerine depolamaktır. Deyim içindeki 20, okunacak verinin şeklinin etiketi 20 olan *FORMAT* deyiminde verildiğini belirtir. Etiket 20'deki format deyimi örneğin 20 *FORMAT* (I3) şeklinde olabilir. Bunun anlamı okunacak sayının üç basamaklı bir tamsayı olduğudur.

Bu noktada FORTRAN'da neden bir FORMAT deyimine ihtiyaç olduğu fakat BASIC'te olmadığını merak edebilirsiniz. Bunun cevabı, *FORTRAN*'ın

geliştirildiği zaman programların ve komutların bilgisayara girilmesinin temel şekli Bölüm 3'te gösterildiği gibi 80 sütunluk delikli kartların kullanılmasıdır. Bir tek kart ile birçok veri değeri gönderilebileceğinden, READ deyimi bir kartta kaç değer olduğunu ve her değerin kaç basamak olduğunu bilmek zorundadır. Çoğunlukla CRT konsol veya teletayp ile kullanılan BASIC ise serbest formattaki girişler için geliştirilmiştir, böylece değerler ayırıcılarla birlikte yazılabilir.

FORTRAN 'daki *WRITE* deyimleri de genellikle bir FORMAT deyimine sahiptir. En basit *WRITE* şekli, *WRITE (BIRIM #,* stl) (değişken, değişken, ...) şeklindedir. Şekil 14.8 a daki bir örnek *WRITE* (4,50) *FARNHT* (1), *CLSIUS*, *KEVIN*'dir. Bu deyim, değişken değerleri *FARNHT* (1), *CLSIUS*, ve *KELVIN*'in, *BIRIM*#4 olan satır yazıcıya yazılacağını belirtir. Basılacak değerlerin formatları 50 ile etiketlenmiş FORMAT deyiminde gösterilmiştir.

FORMAT deyimlerinin biraz daha fazla tartışılması bunun anlamını görmenizde size yardım edecektir. Tablo 14.7 FORMAT deyimlerinde kullanılan bazı tekrarlanabilir edit tanımlayıcılarını göstermektedir. Bazı örnekler bunların açıklanmasına yardım edecektir. FORMAT deyimindeki I3, üç basamaklı bir tamsayının girileceğini veya çıkışa verileceğini belirtmektedir. F6.1, işaret ve ondalık noktada dahil olan toplam altı karakterlik bir gerçek sayıyı temsil etmektedir. Tanımlayıcıdaki 1,sayıda ondalık noktanın sağında bir basamak olduğunu gösterir. -237.3, okumak veya yazmak için F6.1tanımlayıcı gerektiren büyüklüklere bir örnektir. *E* ve *F* tanımlayıcıları giriş işlemleri için aynıdır, fakat çıkış için *Ew.d.Ee* gerçek sayıların bilimsel gösterimde basılmasına izin verir. Örneğin *E4.1E2*, -7.6×10^{-6} gibi bir sayının basılmasına izin verir.

Mantıksal değişkenler yalnızca *TRUE* ve *FALSE* değerlerine sahip olur. Bir L5 tanımlayıcı bunların girişine veya çıkışına izin verir. Alfanümerik karakter girişi için A20 tanımlayıcı, 20 karakterlik bir dizinin girişine veya çıkışına izin verir.

Bir FORMAT deyiminde, bir karttan birkaç veri değerinin okunacağını veya bir satıra basılacağını belirten birçok tanımlayıcı, virgüllerle ayrılmış şekilde yazılabilir. Bir X bir boşluğu belirtmek için kullanılır. Şekil 14.8a daki programda yazılan *FORMAT (4x, F6.1, 6X, F6.1, 4X, F6.1)* deyiminin anlamı: sol kenardan başlayarak dört boşluk, bir gerçek sayının değeri, altı boşluk, bir gerçek sayının değeri, dört boşluk ve bir gerçek sayının basılmasıdır.

TABLO14.7

FORTRAN FORMAT DEYİMLERİNDEKULLANILAN TEKRARLANABİLİR EDİT TANIMLAYICILARI

Iw	Tamsayı tanımlayıcı
Fw.d	Gerçek sayı tanımlayıcı
Ew.d	Gerçek sayı tanımlayıcı
Ew.d.Ee	Gerçek sayı tanımlayıcı
Lw	Mantıksal tanımlayıcı
A	Değişken uzunlukta alfanümerik tanımlayıcı
Aw	Sabit uzunlukta alfanümerik tanımlayıcı
Bw	İkilik sayı tanımlayıcı
Zw	Onaltılık sayı tanımlayıcı
I, F, E, L ve A	Edit edilecek veri tipini belirtir
B ve Z	Edit edilecek verinin sayı tabanını belirtir
w=	Edit edilen tüm alanın genişliğini temsil eden,sıfır olmayan işaretsiz tamsayı sabiti
d=	Ondalık noktadan sonraki basamak sayısını temsil eden işaretsiz tamsayı sabiti
e=	Üs alanının genişliğini temsil eden, sıfır olmayan işaretsiz tamsayı sabiti

Kaynak: Intel

Karakter ve kelimelerin çıkışlarını basitleştiren bir başka FORTRAN deyimi Şekil14.8a da defalarca kullanılmıştır. Bunun için WRITE deyimi, basitçe birim numarasını ve basılacak kelimelerin tek tırnak işareti içinde gösterildiği FORMAT deyiminin etiketini belirtir. Örneğin WRITE (6, 9) deyiminde basılacak kelimeler, 9 FORMAT('FAHRENHEIT'IN CELSIUS VE KELVIN'E ÇEVRİLMESİ') şeklindedir. Eğer mesaj bir satırdan uzun olursa, 6. sütuna bir devam karakteri konur ve mesaj Şekil 14.8a da 14 nolu deyimde gösterildiği gibi bir sonraki satırda devam eder. Şekil 14.8a da 40 nolu satırda gösterildiği gibi bir FORMAT deyimi içindeki tek tırnak işareti başka tanımlayıcılarla da kullanılabilir. Boşluklarla ayrılan üç sütun başlık bu deyim ile basılmıştır. Deyimin sonundaki kesme işareti yazıcıya bir sonraki satırın başına dönüp yazmaya oradan devam etmesini belirtir.

Tek I/O cihazının konsol olması durumunda, liste-yönlendirmeli format olarak adlandırılan bir teknik verilerin serbest şekilde girilmesine izin vererek girişi basitleştirir. Bu durumda READ deyimi READ (BİRİM#, *) (değişken) şeklindedir. FORMAT deyimi etiketinin yerinde yıldız işareti FORMAT deyiminin gerekmediğini ve yazılan değerlerin yazıldıkları gibi okunacaklarını belirtir. Karakter veya kelimeler birbirlerinden tek tırnak işareti ile ayrılmış olarak yazılabilirler. Şekil 14.8a da derleyici satır sayısı 18'de gösterilen READ deyimi bu duruma bir örnektir.

PROGRAM ÖRNEĞİ

Daha önceki bölümlerde FORTRAN deyimleri ve formatlama tartışıldı. Şekil 14.8a daki program içindeki birçok deyim açıklandı ve örnek olarak kullanıldı. Bu bölüm bütün programın çalışmasını açıklamaktadır.

Bir PROGRAM deyimi ve bazı açıklamalardan sonra gerçek ve tamsayı değişkenleri tanımlanmıştır. Bu isimlerin BASIC isimlerinden daha fazla anımsatıcı olduklarına dikkat ediniz. DIMENSION deyimi 100 değer için FARNHT isimli bir dizi tanımlar

Program başlığı konsola ve satır yazıcıya yazılır. Yazıcıya gönderilen mesajda başlıktan önceki 1, yazıcıya mesaj basmadan önce bir sonraki sayfanın başına gitmesini söyleyen bir kontrol karakteridir. Bu, dökümün güzel görünmesini sağlamak için başlığı sayfanın başına yazar.

Daha sonra, kullanıcıya verilen direktifler yalnızca konsola gönderilir. READ(5, 20) NUMBER ve FORMAT (I3) deyimleri, kullanıcının yapılacak çevirmelerin sayısını girmesini bekler. Normal FORMAT deyimi FORMAT (I3)'ün girilen sayı 100'den az ise, kullanıcının önceki pozisyonlarda sıfır girmesini gerektirdiğine dikkat ediniz. Liste yönlendirmeli formatlama burada kullanılabilir, fakat normal formatlama bu tip problemi göstermek için kullanılmıştır.

Eğer uygun olmayan bir sayı girilirse, IF deyimi direktifleri yeniden basmak üzere programı 11 nolu etikete geri gönderir. Uygun bir sayı girilirse kullanıcıya Fahrenheit değerlerini virgüllerle ayrılmış olarak girmesi için konsoldan direktif verilir. READ (5, *) (FARNHT (I), I = 1, NUMBER) bu değerleri yazıldıkları gibi alır ve bellek içinde FARNHT dizisine depolar. Bu örtük bir DO döngüsü içerdiğinden değişik bir READ deyimidir. Bu deyim I = 1'den I=NUMBER'a kadar olan indeks için giriş cihazından bir değer okunacağını ve I'ıncı eleman olarak FARNHT dizisine girileceğini belirtir. Bu birçok değeri girmek için yaygın olan kısa bir yol olduğu için gösterilmiştir.

Bütün değerler girildiğinde, çıkış tablo başlıkları konsol ve satır yazıcıda basılır. Daha sonra her Fahrenheit sıcaklık için eşdeğer Celsius ve Kelvin

değerlerini hesaplayan ve basan bir DO döngüsü kullanılır. Celcius eşdeğerini hesaplayan deyim ilk olarak Fahrenheit değerinden 32 çıkarır ve sonucu 50/9 ile çarpar. Bu istenen değerin 10 katı bir sonuç verir. Tablo 14.6'da gösterilen kütüphane fonksiyonlarından biri olan ANINT bu sonucu en yakın tamsayıya yuvarlar. 10 ile bölmek ondalık noktayı uygun yerine yerleştirir ve sonucu en yakın 0.1°C'ye yuvarlar. Kelvin basitçe Celcius değerine 273.2 ilave ederek hesaplanır.Şekil 14.8b girilen beş değer için program sonuçlarını göstermektedir.

BİR FORTRAN PROGRAMININ DERLENMESİ VE ÇALIŞTIRILMASI

Çalışabilecek bir FORTRAN programının derlenmesi çok-adımlı bir süreçtir.

KAYNAK DOSYANIN YARATILMASI: İlk adım, Şekil 14.8a da gösterilen programa benzer temel program birimini editör programını kullanarak yazmak ve RAM'a girmektir. Programın hepsi girildikten sonra flopy disk ya da teyp üzerine yüklenir. Bu kaynak dosyası olarak adlandırılır.

DERLEME: İkinci olarak kaynak dosyasını yeniden yerleştirilebilir amaç koduna dönüştüren derleyici programı kullanılır. Yeniden yerleştirilebilir deyiminin anlamı sembolik adresler kullanılmasıdır, böylece modül daha sonra hafızada herhangi bir yere yerleştirilebilir. Bu noktada derleyici, program biriminin şekil 14.8a da gösterilen döküme benzer bir dökümünü çıkarır. Kendi referansı için, derleyici dökümdeki her deyimin sol tarafına bir satır numarası koyar. Dökümde de gösterildiği gibi aynı zamanda amaç modülü için gerekli bellek bayt sayısını da gösterir. FARNHT içindeki her gerçek değer 4 bayt gerektirdiğinden, yalnızca bu dizi 400 baytlık değişken deposu gerektirir.

Derleyici aynı zamanda program biriminde bulunan sentaks hatalarını da veya hiç hata bulunmadığını belirtir. Hata bulunursa, kaynak dosyası hata yok mesajı görünceye kadar düzeltilir ve yeniden derlenir.

BAĞLAMA: Üçüncü olarak, düzeltilen amaç modülü kütüphane fonksiyonlarının ve I/O yordamlarının amaç modüllerine bağlanmalıdır. *SQRT(X), ANINT (X)* gibi öz fonksiyonların aslında altyordam olduklarını hatırlayınız. Eğer bir program birimi içinde kullanılırlarsa bunların kodu birimin kodu ile bağlanmalıdır. *READ* ve *WRITE* deyimleri ile çağrılan I/O yordamları da ana modüle bağlanmak zorundadır.

YERLEŞTİRME: Dördüncü olarak, bütün program amaç kısımları birlikte bağlandıktan sonra yerleştirilmelidirler. Bunun anlamı bütün sembolik adreslerin mutlak adreslerle değiştirilmesidir. Sonuç RAM'a girilebilen ve yürütülebilen makine kodu programıdır. Aynı zamanda, depolama için bir disk dosyasına yüklenebilir.

PASCAL İLE PROGRAMLAMA

Pascal, FORTRAN ve BASIC'ten daha yeni bir dildir; bu yüzden onların bazı problemleri Pascal'da yoktur. Pascal programcının hata ayıklaması kolay olan yapısal modüler programları kolayca yazmasına izin verir. Pascal mikroişlemci tabanlı küçük sistemlerde çalışabilecek şekilde kolaylıkla uyarlanabilir.

BİR PASCAL PROGRAMININ GENEL YAPISI

Bir Pascal programının iki temel parçası bulunur. Bir başlık ve bir blok. Şekil 14.11a tam bir programı göstermektedir. Başlık, program ismini ve INPUT ya da OUTPUT gibi önceden tanımlanmış parametre listesini verir. Bir örnek PROGRAM FCNVRT (INPUT, OUTPUT) olabilir.

Programın blok kısmı iki parçadan oluşur: bir tanımlama parçası ve ana gövde. İlk olarak yazılması gereken tanımlama parçasında bütün değişkenlere, dizilere, fonksiyonlara, dosyalara, kümelere vb. isimler verilir ve tipleri tanımlanır. Örneğin, eğer K isimli bir tamsayı değişken program içinde kullanılacaksa, bloğun başında VAR K: INTEGER deyimi ile tanımlanmalıdır.

Programın ana gövdesi tanımlanmış veri tiplerini işleyen deyimleri içermektedir. BEGIN ile başlar ve END. ile biter. END deyiminden sonra bir nokta gereklidir. Deyimler noktalı virgüllerle ayrılır. Satır numaraları kullanılmaz.

TANIMLAMALAR: Şekil 14.12 bir Pascal programının başlangıcında yapılabilecek beş çeşit tanımlamanın sentaks diyagramını göstermektedir. Yalnızca bir program içinde kullanılması gereken çeşitler tanımlanmalıdır, böylece bir programda hepsine ihtiyaç duyulmayabilir. Bir sentaks diyagramı her çeşit tanımlama için doğru format ve noktalamayı göstermektedir.

```
PROGRAM FCNVRT;
(*FAHRENHEIT' I CELCIUS VE KELVINE ÇEVİRİR*)
VAR
      LP: TEXT;
      DISK: FILE OF CHAR;
      FAREN: REAL;
FUNCTION CNVRT (F: REAL): REAL;
      (* FAHRENHEIT'I CELCIUS'A ÇEVİRİR*)
BEGIN (* CNVRT*)
      CNVRT: =(ROUND ((F-32.0)*50.0/9.0))/10.0
END; (*CNVRT*)
```

```
BEGIN (* ANAHAT PROGRAMI*)
    REWRITE (LP, 'PRINTER:'); (* SATIR YAZICIYI DOSYA OLARAK
AÇAR*)
    REWRITE (DISK, '#4: DATA. TEXT'); (* DATA İSİMLİ DİSK DOSYASI
NI AÇAR*)
    WRITELN ('FAHRENHEIT 'IN CELCIUS VE KELVİN
ÇEVRİLMESİ'); (*KONSOLA*)
    WRITELN (LP 'FAHRENHEIT'IN CELCIUS VE KELVIN
ÇEVRİLMESİ) ; ("YAZICI*)
    WRITELN (' FAHRENHEIT DEĞERLERİNİ GİR - SON DEĞERDEN
SONRA - 1000.0 YAZ');
    REPEAT
    READ (FAREN);
    IF FAREN = 1000.0 THEN
      WRITELN (DISK, FAREN)
    UNTIL FAREN = -1000.0;
CLOSE (DISK, LOCK); (*DOSYALARI KAPATMANIN STANDART
OLMAYAN YOLU*)
RESET (DISK,'#4: DATA.TEXT');
WRITELN (LP, 'FAHRENHEIT CELSIUS KELVIN');
    WHILE NOT EOF (DISK) DO
    BEGIN
     READLN (DISK, FAREN);
     WRITELN (LP, FAREN: 8:1, CNVRT (FAREN):12:1, CNVRT
(FAREN) +273.2:8:1)
    END; (*WHILE *)
    CLOSE (LP)
END . (* ANAHAT PROGRAMININ SONU*)
```

(a)

FAHRENHEIT'IN CELSIUS VE KELVINE ÇEVRİLMESİ

FAHRENHEIT	CELSIUS	KELVIN
12.0	-11.1	262.1
45.7	7.6	280.8
345.5	174.2	447.4
-200.7	- 129.3	143.9

Şekil 14.11 Fahrenheit Sıcaklıkları Celcius ve Kelvin'e Çeviren Pascal Programı a) Program Listesi b) Örnek çalıştırma

Beş çeşit tanımlama: *ETIKET, CONST (sabit), TIP, VAR (değişken)* ve *PROCEDURE* veya *FUNCTION* tanımlamalarıdır. Sentaks diyagramın en alt satırı *BEGIN* kelimesi ile başlayan, noktalı virgüllerle ayrılan deyimleri içeren, ve *END.* kelimesi ile biten programın ana gövdesini temsil eder.

LABEL (etiket): Pascal'daki bir LABEL, yürütmeyi başka bir deyime şartsız olarak atlatmak için bir GOTO deyimi ile kullanılır.Bir etiket bir ile dört basamaktan oluşan tamsayı olmalıdır. Sentaks diyagramı bir LABEL tanımlanması için doğru formatın ;

LABEL 33; veya LABEL 26,78; olduğunu gösterir.

CONST: *Tanımlanmış bir sabit (CONST)* bir sayı veya bir karakterler dizisi olabilir. Doğru formatta tanımlanmış sabitler:

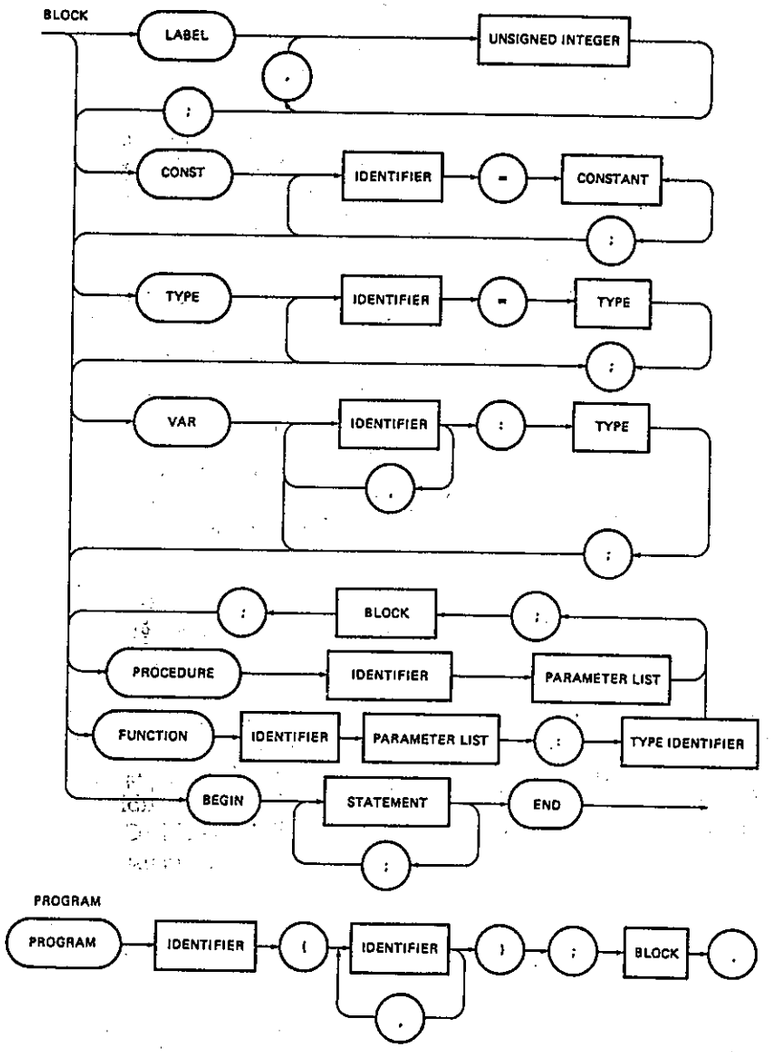
CONST PI = 3.14159; ADDON = -32; OK ='YES'; olabilir.

VAR: *Bir değişken ismi herhangi bir uzunlukta olabilir,* fakat bazı derleyiciler yalnızca ilk sekiz harfi kullanırlar. Bir blok içinde kullanılan herhangi bir değişken bloğun başlangıcında tanımlanmalı ve tipi belirtilmelidir. Dört değişken tipi REAL, *INTEGER, CHARACTER (kısacası CHAR),* ve BOOLEAN'dır. Bu tipler için bazı tanımlama örnekleri şu şekildedir.: VAR INCREMENT:REAL; COUNT: INTEGER; JOE: CHAR; RESULT: BOOLEAN;.

Gerçek değişkenler bir tamsayı ve ondalık kısım içerirler, tamsayı değişkenleri yalnızca tamsayı kısmına sahiptir. Karakter değişkenleri değer olarak alfabenin herhangi bir harfine veya noktalama işaretine sahip olabilir. Boolean değişkenleri yalnızca *TRUE (doğru)* veya *FALSE (yanlış)* değerine sahip olurlar.

Bu dört basit değişken tipine ilave olarak Pascal, yapısal tipler olarak bilinen daha karmaşık değişken tiplerine izin verir. Bunlar dizileri, kümeleri ve dosyaları içerir. Bir dizi değişkeninin tanımlamasına örnek olarak VAR A:

ARRAY [1.. 100] OF INTEGER: verilebilir. Bunun anlamı A kelimesinin tamsayı değerli bir diziyi temsil etmesidir.



Şekil 14.12 Pascal Tanımlamalarının Sentaks Diyagramı (Springer,Verlag)

TYPE: Pascal,kullanıcının yeni tipte değişkenleri tanımlamasına ve onlara değerler vermesine izin verir. Örneğin, TYPE GÜN= (PAZAR, PAZARTESİ,SALI, ÇARŞAMBA, PERŞEMBE, CUMA, CUMARTESİ) deyimi PAZAR'dan CUMARTESİ'ye kadar değerleri alan GÜN tipi bir değişken tanımlar. Bu TRUE veya FALSE değerlerine sahip olabilen önceden tanımlanmış BOOLEAN tipine benzer. Bir diğer TYPE tanımlaması örneği TYPE RENK= (KIRMIZI, PORTAKAL, SARI, YEŞİL, MAVİ, ÇİVİT RENGİ,MOR): olabilir.

Yeni bir tip tanımlandıktan sonra bir değişken tanımlanması içinde diğer tipler gibi kullanılabilir. Örneğin VAR *X*: RENK, *X* değişkeninin RENK tipi olduğunu ve yalnızca buna karşı gelen TYPE tanımlamasındaki değerleri alabileceğini belirtir. Bu şekilde yeni veri tipleri tanımlayabilmek Pascal'ın çok güçlü bir özelliğidir.

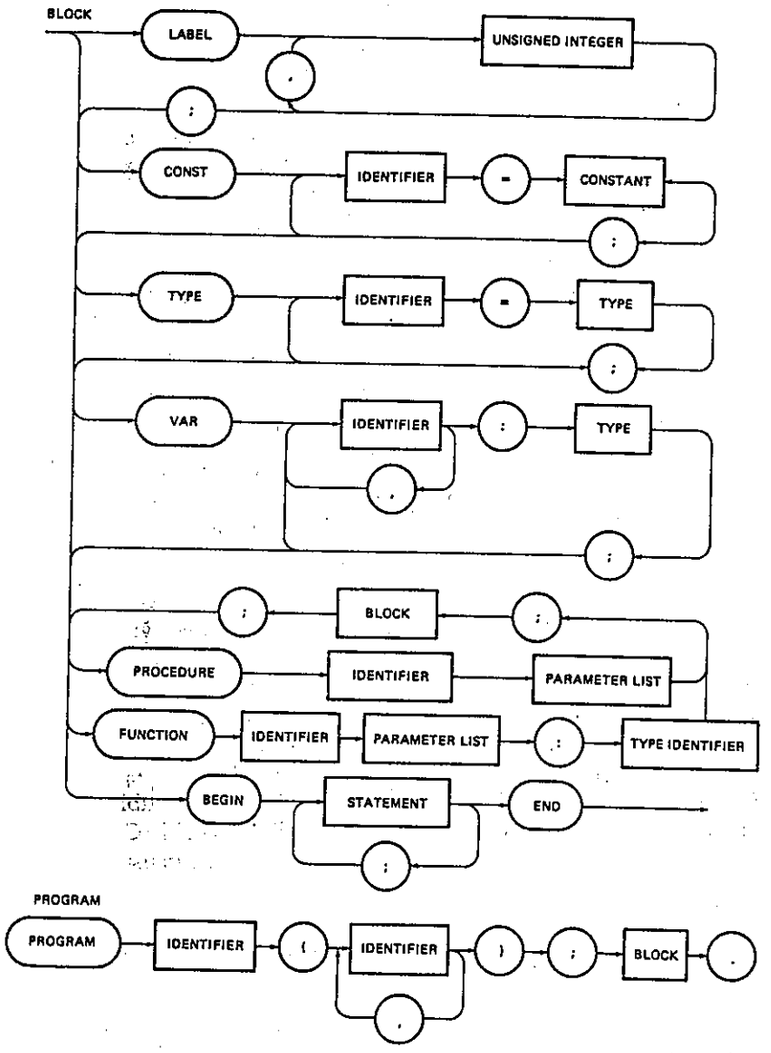
PROCEDURE ve FUNCTION: Pascal'da altyordamlar program içinde gerektiği yerlerde isimleri ile çağrılırlar. PROCEDURE veya FUNCTION tamamlandıktan sonra yürütme programın ana gövdesindeki bir sonraki deyime döner.

Bir *PROCEDURE, bir işi yapmak için yazılan genel bir altyordamdır.*

Bir FUNCTION, bir açının sinüsünü hesaplayan veya bir sayının kare kökünü alan ve sonuçta belirli bir sayı döndüren bir alt-yordamdır. Tablo 14.8 Pascal'ın önceden tanımlanmış veya özfonksiyonlarının bir listesini ve açıklamalarını göstermektedir.

Altyordamların herhangi bir yere yerleştirildiği çevirici dilin aksine Pascal'da bir PROCEDURE veya FUNCTION kendisini çağıran bloktan önce bir noktaya yerleştirilmelidir. Bir PROCEDURE veya FUNCTION altbloğuna bir başlık verilmelidir. Bu başlık yordama bir isim verir. Örneğin "PROCEDURE MINMAX" veya "FUNCTION FCNVRT" gibi. Daha sonra, programın bütünü için tanımlanmış değişkenler, tipler, sabitler vb. tanımlanmalıdır. PROCEDURE veya FUNCTION içinde tanımlanan değişkenler yerel değişkenlerdir. Gerekli tanımlamalar yapıldıktan sonra altbloğun deyimleri sıralanır. Bu, Şekil 14.11a nın üst kısmındaki FUNCTION FCNVRT'de gösterilmiştir. Bir PROCEDURE listesi, bir tam kısa programın listesine çok benzemektedir.

ARRAY [1.. 100] OF INTEGER: verilebilir. Bunun anlamı A kelimesinin tamsayı değerli bir diziyi temsil etmesidir.



Şekil 14.12 Pascal Tanımlamalarının Sentaks Diyagramı (Springer,Verlag)

TYPE: Pascal,kullanıcının yeni tipte değişkenleri tanımlamasına ve onlara değerler vermesine izin verir. Örneğin, TYPE GÜN= (PAZAR, PAZARTESİ,SALI, ÇARŞAMBA, PERŞEMBE, CUMA, CUMARTESİ) deyimi PAZAR'dan CUMARTESİ'ye kadar değerleri alan GÜN tipi bir değişken tanımlar. Bu TRUE veya FALSE değerlerine sahip olabilen önceden tanımlanmış BOOLEAN tipine benzer. Bir diğer TYPE tanımlaması örneği TYPE RENK= (KIRMIZI, PORTAKAL, SARI, YEŞİL, MAVİ, ÇİVİT RENGİ,MOR): olabilir.

Yeni bir tip tanımlandıktan sonra bir değişken tanımlanması içinde diğer tipler gibi kullanılabilir. Örneğin VAR *X*: RENK, *X* değişkeninin RENK tipi olduğunu ve yalnızca buna karşı gelen TYPE tanımlamasındaki değerleri alabileceğini belirtir. Bu şekilde yeni veri tipleri tanımlayabilmek Pascal'ın çok güçlü bir özelliğidir.

PROCEDURE ve FUNCTION: Pascal'da altyordamlar program içinde gerektiği yerlerde isimleri ile çağrılırlar. PROCEDURE veya FUNCTION tamamlandıktan sonra yürütme programın ana gövdesindeki bir sonraki deyime döner.

Bir *PROCEDURE, bir işi yapmak için yazılan genel bir altyordamdır.*

Bir FUNCTION, bir açının sinüsünü hesaplayan veya bir sayının kare kökünü alan ve sonuçta belirli bir sayı döndüren bir alt-yordamdır. Tablo 14.8 Pascal'ın önceden tanımlanmış veya özfonksiyonlarının bir listesini ve açıklamalarını göstermektedir.

Altyordamların herhangi bir yere yerleştirildiği çevirici dilin aksine Pascal'da bir PROCEDURE veya FUNCTION kendisini çağıran bloktan önce bir noktaya yerleştirilmelidir. Bir PROCEDURE veya FUNCTION altbloğuna bir başlık verilmelidir. Bu başlık yordama bir isim verir. Örneğin "PROCEDURE MINMAX" veya "FUNCTION FCNVRT" gibi. Daha sonra, programın bütünü için tanımlanmış değişkenler, tipler, sabitler vb. tanımlanmalıdır. PROCEDURE veya FUNCTION içinde tanımlanan değişkenler yerel değişkenlerdir. Gerekli tanımlamalar yapıldıktan sonra altbloğun deyimleri sıralanır. Bu, Şekil 14.11a nın üst kısmındaki FUNCTION FCNVRT'de gösterilmiştir. Bir PROCEDURE listesi, bir tam kısa programın listesine çok benzemektedir.

TABLO 14.8

ÖNCEDEN TANIMLANMIŞ PASCAL FONKSİYONLARI

ARİTMETİK FONKSİYONLARI	
abs (X)	X'in mutlak değerini hesaplar. Sonuç, X ile aynı tiptedir. Gerçek veya tamsayı olabilir.
sqr (X)	X*X'i hesaplar. Sonuç, X ile aynı tiptedir. Gerçek veya tamsayı olabilir.
sin (X)	Takip eden fonksiyonlar için X gerçek veya tamsayı olabilir. Sonuç her zaman gerçek tiptedir.
cos(X)	
arctan(X)	
exp (X)	
In (X)	(Doğal logaritma)
sqrt (x)	(kare kök)

TRANSFER FONKSİYONLARI	
trunc (X)	X gerçek tipte olmalıdır. X > = 0 için sonuç X'ten küçük yada eşit tamsayıdır. X< 0 için sonuç X'ten büyük ya da eşit en küçük tamsayıdır.
round(X)	X gerçek tipte olmalıdır. Sonuç X'in yuvarlanmış hali olan bir tamsayıdır. Yani, X ≥0 için round (X) = trunc (X + 0.5) ve X < 0 için round (X) = trunc (X - 0.5)
ord (X)	X'in tipi ile tanımlanan değerler kümesinde X argümanının ordinal sayısı
chr (X)	X tamsayı olmalıdır ve sonuç ordinal sayısı X olan karakterdir (eğer varsa)

OPERATÖRLER

Tablo 14.9, Pascal'ın aritmetik, mantıksal ve ilişki operatörlerinin bir listesini göstermektedir. Aritmetik operatörlerin öncelikleri Tablo 14.9'da gösterilmiştir. Çarpma ya da bölme, toplama veya çıkarmadan önce yapılır. Pascal'da üs alma operatörü yoktur. DIV ve MOD adlı iki yeni operatör vardır.

DIV tamsayı bölmesidir. Bir tamsayı başka bir tamsayı ile bölündüğünde, bölümün yalnızca tamsayı kısmı alınır. Örneğin 5 DIV 2'nın sonucu 2'dir. MOD operatörü bir tamsayı bölmeden kalanı hesaplar. Örneğin 5 MOD 2'nin sonucu 1 'dir. MOD operatörü bir değişkenin diğeri tarafından tam bölünüp bölünmediğini belirlemekte faydalıdır.

TABLO 14.9

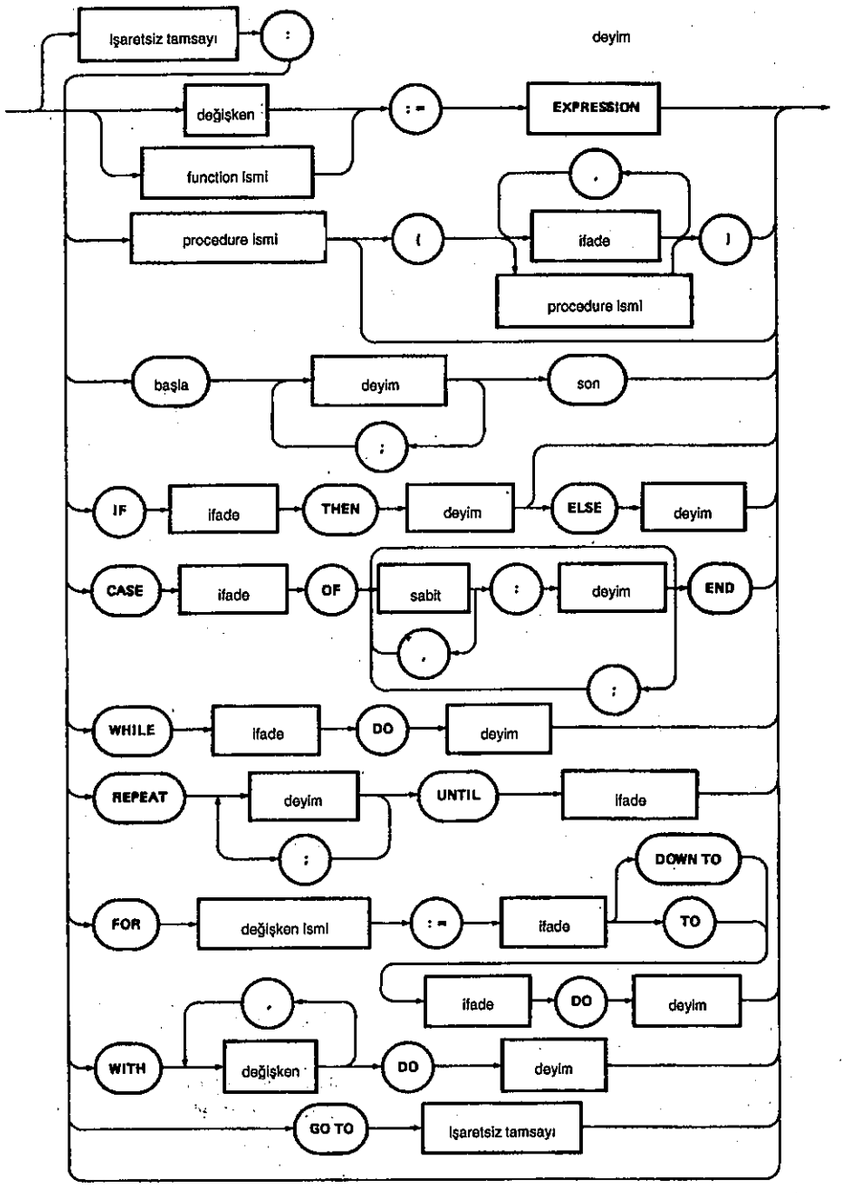
PASCAL'IN ARİTMETİK, MANTIKSAL VE İLİŞKİ OPERATÖRLERİ

ÖNCELİK	OPERATÖR	TANIM
1	NOT	Boole Değili
2	*	Çarpma
2	/	Bölme (gerçek)
2	DIV	Tamsayı Bölme
2	MOD	Modülüs (Tamsayı bölmeden kalan)
2	AND	Boole VE
3	+	Toplama
3	-	Çıkarma
3	OR	Boole VEYA
4	=	Eşit
4	< >	Eşit değil
4	<	Küçük
4	< =	Küçük veya eşit
4	> =	Büyük veya eşit
4	>	Büyük
4	IN	Bir küme içinde

DEYİMLER

Bütün gerekli tanımlar yapıldıktan sonra, programın ana gövde deyimleri yazılır. Bu noktada programın ana gövdesi yazılmadan önce programcının ne tanımlanacağını nasıl bildiğini merak edebilirsiniz. Bunun cevabı programcının başlangıçta muhtemelen bunu bilmediğidir. Programlar sürekli yaklaşımla yazılır. Daha çok değişken gerektikçe bunlar tanımlama listesine eklenirler. Sahte değişkenler listeye eklenebilir ve daha sonra ihtiyaç duyulmazsa listeden çıkarılır.

Şekil 14.13 yaygın Pascal deyimlerinin bir sentaks diyagramı göstermektedir. Birincisi bir atama deyimidir.



Şekil 14.13 Pascal deyimleri için Sentaks Diyagramı (Springer-Verlag)

ATAMA DEYİMİ : Bu deyim BASIC'teki LET deyimine karşı gelir. İki nokta üstüste ve eşit işaretini birlikte kullanır. Bazı örnekler :

X : = Y + 2, X : = X + 1, CUBE : = X * X * X ve Y : = SQRT (X).

PROCEDURE DEYİMİ : Bu deyim bir prosedürü çağırmak için kullanılır. Prosedürün ismi ile başlar ve parantez içinde prosedüre geçirilecek parametreleri içerir. MINMAX (3, 7, 9) deyimi MINMAX prosedürünü çağırır ve bu prosedüre 3,7 ve 9 değerlerini geçirir.

BEGIN ve END : BEGIN ve END deyim değillerdir. Bunlar bir grup deyimi içeren parantez gibidirler. Programın ana gövdesinin başında bir BEGIN ve sonunda bir END kullanılır. Bunlar aynı zamanda bir çok deyimin toplanıp bir birleşik deyim olarak ele alınmasında da kullanılırlar. Şekil 14.11 deki program örneği bunu göstermektedir. Sentaks diyagramında bir deyimin belirtildiği yerde, BEGIN ve END ile kapatılmış bir birleşik deyim kullanılabilir. Bu END deyiminin program sonunda kullanılan END deyiminin aksine sonunda bir nokta bulunmaz.

FOR... DO : Bu deyim bir seri deyimin belli sayıda tekrarlanmasına imkân sağlar. Önceden gördüğümüz *FOR ... NEXT* döngüsüne benzemektedir. Şekil 14.14 1 ile 16 arasındaki sayıların kareköklerinin hesaplanıp basılmasını göstermektedir. BEGIN ve END arasındaki birleşik deyim 16 defa yürütülecektir.

PROGRAM ROOT 2

```
VAR I:INTEGER; X : REAL;
BEGIN
FOR I: = 1 TO 16 DO
        BEGIN X: = I;
        WRITELN (I, SQRT (X) );
        END;
WRITELN ('DONE');
END.
```

Şekil 14.14 Pascal'da bir FOR ... DO döngüsü örneği.

REPEAT ... UNTIL : Bu deyimi kullanarak bir deyim ya da deyimler grubu bir şart yerine getirilinceye kadar tekrarlanır. Örneğin şekil 14.14 deki REPEAT ... UNTIL döngüsü kullanıcı tarafından yazılan Fahrenheit değerlerini imkânsız değer -1000 girilinceye kadar okur.

WHILE ... DO : Tekrarlı işleri yapabilmenin bir diğer yolu bu deyimi kullanmaktır. WHILE H > 0 DO READ (TIME) bir örnek olarak verilebilir.

Şekil 14.11 deki program bir başka WHILE ... DO döngüsü örneğini göstermektedir.

GOTO: Bu deyim, bir etikete şartsız atlamaya sebep olur. Etiket bir ile dört basamak arasında olan bir tamsayıdır.

IF... THEN ... ELSE : Bu deyim içindeki ifade doğru ise THEN kelimesinden sonraki deyim yürütülür. Eğer ifade yanlış ise, yürütme şekil 14.15'da gösterildiği gibi ya ana programla ya da ELSE kelimesinden sonra gelen deyimle devam eder. Şekil 14.15b de gösterildiği gibi IF deyimleri içiçe yazılabilir.

```
a) IF Y = 1 THEN P: =7 ELSE P: = 14;

b) IF Y = 1 THEN P: = 7
     ELSE IF Y = 2 THEN P: = 14
     ELSE WRITELN ('UYGUN OLMAYAN DEĞER');
```

Şekil 14.15 IF... THEN... ELSE deyimlerinin Pascal örnekleri
a) Tek-seviyeli IF ...THEN ... ELSE b) İçiçe IF'ler.

CASE ve WITH: Bunlar standart sentaks diyagramına ilave edildiği için Şekil 14.13 te gösterilmişlerdir, fakat onları burada anlatmayacağız. Daha geniş bilgi için bölümün sonundaki referanslardan birine danışınız.

KONSOL GİRİŞ VE ÇIKIŞI

Eğer giriş ve çıkışlar konsoldan yapılacaksa, birçok sistemde yalnızca READ, READLN, WRITE ve WRITELN deyimlerine ihtiyaç vardır. Bunun sebebi konsolun giriş ve çıkış olarak önceden bağlanmış olmasıdır. Program başlığındaki parametre listesi (INPUT, OUTPUT) bunu belirtmektedir.

WRITE ve WRITELN: Bu deyimler belirtilen büyüklükleri sistem-konsolunda basar; fakat WRITELN mesajı gönderdikten sonra bir taşıyıcı dönüşü ve satır besleme gönderir. Bu yüzden bir WRITELN deyimi, bir sonraki WRITE ya da WRITELN deyimi tarafından gönderilen mesajın bir sonraki satıra basılmasına sebep olur.

Bazı örnekler WRITE ve WRITELN deyimlerinin kullanım şeklini en iyi şekilde gösterecektir (bakınız Şekil 14.16). WRITE ve WRITELN kelimelerinden sonraki parantezler içindeki büyüklükler bir satıra yazılır. REAL değişkenlerin değerleri c örneğinde olduğu gibi bilimsel gösterimde yazılır. d örneğinde değişkenden sonraki birinci tamsayı istenen toplam basamak sayısını belirtir ve ikinci tamsayı da ondalık noktanın sağındaki basamak sayısını verir. Eğer bu değerler verilirse basılan sayı bilimsel gösterimde olmaz. Şekil 14.11*a* daki programda bu sayılar aynı zamanda basılan sıcaklıkların başlıkların altında ortalanmış olarak basılması için kullanıldı. Eğer belirtilen alan genişliği sayıdan büyük ise sayı alanın sağ kısmına basılır. Tek tırnak içine kapatılmış karakter dizileri yazıldıkları şekilde basılırlar. e ve f örnekleri WRITE ve WRITELN arasındaki farkı göstermektedir.

DEYİM	BASILAN SONUÇ
a. WRITE (1, 2,3, 4)	1 2 3 4
b. WRITE ('BİR HATA YAPTIN')	BİR HATA YAPTIN
c. WRITE (SQRT (R))	1.50000000E + 00 (R = 2.25 için)
d. WRITE (SQRT (R): 6: 2)	1.50 (R =2.25 için)
e. WRITELN ('CEVAP=', A)	CEVAP = (A'nın değeri)
WRITELN ('ANLADIN MI?')	ANLADIN MI?
f. WRITE ('CEVAP =', A)	
WRITELN ('ANLADIN MI?)	CEVAP = (A'nın değeri) ANLADIN MI?

Şekil 14.16 WRITE ve WRITELN deyimlerinin örnekleri

READ ve READLN : Bu iki deyim arasındaki fark gene bir taşıyıcı dönüşüdür. I'nın tamsayı değişken olduğu bir READLN (I) deyimi makinenin, kullanıcının I için uygun bir değer girmesini beklemesine sebep olur. Değer girildikten sonra bir taşıyıcı dönüşü yazılmalıdır. Eğer birden fazla değer yazılırsa yalnızca taşıyıcı dönüşü girilmeden önceki son değer girilir. Bir READ deyimi ile, taşıyıcı dönüşü yazılmadan önce boşluklarla birbirinden ayrılmış bir kaç değer yazılabilir. Şekil 14.17 READ ve READLN deyimlerinin

birkaç örneğini göstermektedir. Veri sıralı dosyalarda teyp ve disklere de depolanabilir.

Dosyanın ikinci bir anlamı satır yazıcı, disk sürücü ya da kağıt-şerit okuyucu gibi bir giriş ve çıkış birimidir. Dosya teriminin bu kullanımı, bilgisayara girişlerin yalnızca delikli kartlar dosyası ile yapıldığı günlerden kalmadır.

DEYİM	YAZILAN VERİ	SONUÇ
READ (FAREN)	-40.0 (dönüş)	FAREN= -40.0
READ (A, B, C)	5(boşluk) 7(boşluk) 9(dönüş)	A = 5, B = 7, C = 9
READLN (A, B, C)	5(boşluk) 7(boşluk) 9(dönüş)	A =9
READLN (A, B, C)	5(dönüş) 7(dönüş) 9 (dönüş)	A = 5, B = 7, C =9

Şekil 14.17 READ ve READLN deyimlerinin örnekleri

DİGER BİRİMLERE GİRİŞ VE ÇIKIŞLAR: DOSYALAR

Bilgisayar dilinde dosya kelimesinin iki anlamı vardır. Birinci anlamı verilerin depolandığı bir bellek yerleri dizisidir. Bu tipteki bir iç dosya bir seri RAM bölgesidir.

Dosya kelimesinin ikinci bir anlamı satır yazıcı, disk sürücü veya kâğıt-şerit okuyucu gibi bir giriş-çıkış birimleridir. Dosya teriminin bu kullanımı bilgisayara girişlerin yalnızca delikli kartlar dosyası ile yapılabildiği günlerden kalmadır.

Şekil 14.11a daki program Fahrenheit sıcaklıkları Celcius veKelvin'e çevirir ve sonuçları bir tablo şeklinde basar. Bu program Şekil 14.8a da gösterilen FORTRAN örneğine benzer şekilde de yazılabilir. Fakat dosyaların kullanımını göstermek için farklı bir yaklaşım seçilmiştir.

Bu program, bir seri Fahrenheit değerini konsoldan girmekte ve onları DATA isimli bir dosya içinde flopy diskte depolamaktadır. İstenen bütün değerler girildiğinde imkansız bir değer olan -1000 yazılması REPEAT...UNTIL döngüsünü sona erdirmektedir. Tablo başlıklarını bastıktan sonra program bir DO...WHILE döngüsüne girer. Bu döngü disk dosyası DATA'dan bir değer okur, Celcius ve Kelvin'e çevirir ve Fahrenheit, Celcius ve Kelvin değerlerini satır yazıcı dosyasına basar.

Bir dış dosyadan veriler okunmadan önce veya dış dosyaya veriler yazılmadan önce bu dosya sembolik olarak sisteme bağlanmalıdır, ya da

açılmalıdır. Bu işlem Şekil 14.11a da gösterildiği gibi REWRITE deyimi ile yapılır. REWRITE (LP, 'YAZICI:') deyimi YAZICI isimli bir satır yazıcıyı açar. REWRITE (DISK, '#4:DATA. TEXT') deyimi DATA. TEXT isimli bir disk dosyasını açar. READ(FAREN) deyimini kullanarak verilerin konsoldan okunması sırasında,WRITELN(DISK, FAREN) deyimi her bir veriyi DATA.TEXT isimli disk dosyasına depolar. WRITELN deyiminde dosya ismi DATA. TEXT'ten bahsetmeye gerek yoktur, çünkü REWRITE deyimi dış dosya DISK'e gönderilen herhangi bir verinin disk üzerindeki DATA.TEXT dosyasına gönderileceğini belirtir.

Bir disk veya satır yazıcıya yazılan veri, doğrudan cihaza gitmez. Bir tampon üzerinden gider. Bu gereklidir, çünkü veriler bir diske her seferinde 1 bayt olarak yazılamaz. Bir disk dosyası bir ya da daha fazla sektörlerden oluşur. Bir sektör 256, veya çift yoğunluk için 512 bayttan oluşur (bölüm 5'te diskler hakkındaki kısma bakınız). Diske depolanacak veri önce 256 veya 512 baytlık tampona yüklenir. Tampon dolu olduğunda disk üzerindeki bir sektör üzerine yüklenir. CLOSE deyimi tampon içindeki verinin sonuna bir dosya-sonucu (EOF) karakteri ekler ve tampon tam dolu olmasa dahi tamponun içeriğini bir sektöre yazar. Bir satır yazıcı tamponu, genellikle 80 karakter olan bir satır için yeterli olan veriyi tutabilir.

Veri, tampon içine her seferinde bir bölge olmak üzere sırasıyla yazılır ve disk dosyası içine de yığınlar olarak yazılır. Bir dosya gösterici, dosya içindeki veri baytlarının yerlerinin nerede olduklarını belirtir. Her veri değeri girildiğinde, dosya gösterici 1arttırılır.

Dosya kapandığında, dosya gösterici dosyanın sonunu, gösterecek şekilde bırakılır. Dosyanın okunabilmesi için dosya gösterici yeniden dosyanın başlangıcına taşınmalıdır. Bu RESET deyimi ile yapılır. Şekil 14.11a'daki RESET (DISK, '#4: DATA. TEXT') deyimi dosya göstericiyi disk dosyası DATA. TEXT içindeki ilk bölgeye taşır. Takip eden WHILE ... DO döngüsü ile DATA. TEXT dosyasından bir değer okur, ve Fahrenheit, Celcius ve Kelvin değerlerini basar. Bu işlem dosya-sonu karakteri bulununcaya kadar devam eder. Daha sonra PRINTER dosyası kapatılır ve program sona erer. DISK dosyasının, programın başlangıcında karakterler dosyası olarak tanımlandığına dikkat ediniz. Programı biraz basitleştirmek için Fahrenheit değerleri disk dosyası içine gerçek sayılar yerine ASCII karakterler olarak depolanırlar. FAREN bir REAL değişken olarak tanımlandığı için bu değerler Celcius ve Kelvin'e çevrilmeleri için READLN (DISK, FAREN) deyimi tarafından sayıya çevrilirler. WRITELN deyimi ise hesaplanan değerleri satır yazıcıya göndermek için ASCII karakterlere çevirir. WRITELN (LP,FAREN: 8:1, CNVRT (FAREN): 12:1, CNVRT (FAREN)= 273.2: 8: 1); deyiminde olduğu gibi CNVRT fonksiyonu çağrıldığında, FAREN'in değeri CNVRT ifadesi içindeki sahte değişken F yerine otomatik olarak yerleştirilir. Gerçekte

FUNCTION CNVRT içinde yuvarlama yapmaya gerek yoktur, çünkü WRITELN (FAREN: 8: 1) deyimi bir sonucu otomatik olarak noktadan sonra bir basamağa yuvarlar. Buna rağmen ROUND deyimi önceden tanımlanmış fonksiyonların nasıl kullanıldığını göstermek için kullanılmıştır. Bu program yazıldığı şekilde Pascal kapasitesi olan bir APPLE II bilgisayarında çalışabilir.

MİKROBİLGİSAYAR-TABANLI ÜRETİMİN GELİŞTİRİLMESİNDEKİ EĞİLİMLER

Mikroişlemcilerin gelişmesi birçok alet ve sistemlerin akıllı olmasına sebep olmuştur. Donanım ile yapılması zor ve hatta imkansız olan bazı işler yazılım ile yapılabilir. Hatta SDK-85 veya MEK6800D2 gibi bazı küçük genelleştirilmiş sistemler birçok değişik işi yapmak için programlanabilir. Bu yüksek maliyetle mücadele edilebilmek için üreticiler, aşağıdaki yaklaşımları kullanırlar:

1. Mümkün olduğu yerlerde programlanacak veya ayrı bir donanımla yapılacak işleri azaltmak için akıllı çevrebirim cihazları kullanılır. 825IA gibi seri port yongaları, işlemciyi seri veriyi gönderme ve alma işleminde rahatlatır. 8087 gibi matematik işlemcileri program yordamları ile yapılan birçok matematik işlemleri yerine getirebilir. Buradaki önemli nokta matematik işlemci bu işlemleri yaparken ana işlemci başka işleri yerine getirmesidir.

2. Mümkün olduğu kadar çok program yüksek-seviyeli dillerde geliştirilir. Programlar, yüksek-seviyeli dillerde genellikle daha hızlı yazılırlar ve hata ayıklanması ve programların değiştirilmesi daha kolaydır. Çoğu programlar için yüksek-seviyeli dil programcısı detaylı donanım yapısını, örneğin çevirici dil programcının bilmek zorunda olduğu kaydedici yapısını, bilmek zorunda değildir.

Derlenmiş yüksek-seviyeli dil programları genellikle daha çok bellek bölgesi kapsarlar ve dikkatli yazılmış çevirici dil programlarından biraz daha yavaş çalışır. Fakat, belleğin bit başına olan maliyetinin azalması ve yüksek-seviyeli dillerle aletin daha hızlı geliştirilmesi bu dezavantajları ortadan kaldırır.

Çevirici dil yalnızca hızın çok kritik olduğu veya programın çok miktarda bağımsız bit testi ve değiştirilmesini gerektirdiği durumlarda kullanılır.

Intellec sistem (bakınız Şekil 8.26) üzerinde çalışan ISIS-II gibi işletim sistemleri değişik dillerde geliştirilmiş amaç-kodu modüllerinin bağlanmasına izin verir. Bu özelliği kullanarak programcı programın bir kısmını matematik işlemlerde kuvvetli olan FORTRAN ile, başka bir kısmını I/O işlemlerinde kuvvetli olan PL/M, ile ve bit işlemleri gerektiren diğer bir kısmını da çevirici dil ile yazabilir.

3. Yapısal, modüler programlama teknikleri kullanılır. Hangi dil kullanılırsa kullanılsın, programlar bağımsız olarak hataları ayıklanabilen ve yeniden kullanılabilen bloklar veya modüller şeklinde yazılmalıdır. İdeal olarak her bloğun bir giriş ve bir çıkış noktası olmalıdır (Yapısal programlama hakkında daha fazla bilgi için bu bölümün sonundaki referanslara bakınız).

4. Yazılım dağıtıcılarından, test edilmiş ve hataları ayıklanmış yazılımlardan mümkün olduğu kadar fazlası alınırsa fabrika içindeki çabalar özel uygulamaların geliştirilmesine yönlenebilir. Bu tekerleği yeniden icat etmeyi önler. Ada gibi yeni diller, programcıların karmaşık yazılım elemanlarının bir kütüphane kataloğunu kullanarak programlarını geliştirmelerine izin verir. Birçok hazır program bulunabilmektedir ve bunların çoğu firma tabanlı veya silikon yazılım şeklindedir. ROM içine depolanmış programlar ya da yordamlar firma tabanlı olarak adlandırılır. Monitorler, editörler ve derleyiciler genellikle firma tabanlı olarak bulunmaktadır.

Silikon yazılımın bir diğer örneği INTEL 80130'dur. Bu cihaz bir gerçek-zaman işletim sistemi geliştirmek için kullanılan yazılım geliştirme bloklarını içerir. Cihaz aynı zamanda 8259A tipinde bir öncelikli kesme kontrolörüne ve üç tane 16-bit programlanabilir zamanlayıcıya sahiptir. Bunlar, bir seri süreci kontrol etmek için kullanılabilen bir gerçek-zaman işletim sisteminin önemli donanım parçalarıdır (Bölüm 12'deki endüstriyel kontrol uygulaması basit bir gerçek-zaman sisteminin bazı elemanlarını içermektedir).

Silikon donanım-yazılımın bir diğer örneği Western Digital Corporation tarafından üretilen beş-yongalık entegre devre kümesidir. Bu yonga kümesi bir derleyici ya da yorumlayıcıya gerek duymadan Pascal *P* kodunu direkt olarak yürütebilir.

Mikrobilgisayar evrimindeki bütün eğilim tek entegre devreler üzerine daha çok donanım ve yazılım koymaktır. Bundan dolayı bilgisayarlar küçülmekte, daha güçlü olmakta ve telefonlar kadar yaygınlaşmaktadır. Bilgisiyar zekâsı çok hızlı bir şekilde evrimleşmektedir. Bugünlerde bilgisayarları akıllı şebekelerle birbirine bağlayıp bilgi ve kaynakların insanların ortak kullanımına sunmak mümkündür. Bilgisayarların insanları birleştirmesinin, nükleer silahların yokedici gücünü yenip yenemeyeceğini zaman gösterecektir.

TEKRAR SORULARI VE PROBLEMLER

1. Aşağıdakileri tanımlayın ve kullanımlarını anlatın.

   a) Monitor programı
   b) Editör programı
   c) Derleyici programı
   e) Yorumlayıcı programı

2. Kaynak programını ve amaç programını tarif ediniz.

3. Bir derleyici ve bir yorumlayıcı arasındaki farkları açıklayınız. Her birinin bir avantajını ve dezavantajını veriniz.

4. Aşağıdaki yüksek-seviyeli diller nelerin kısaltılmışlarıdır?

   a) FORTRAN *b)* BASIC *c)* ALGOL *d)* COBOL *e)* APL

5. BASIC deyimlerine niçin birbirine çok yakın peşpeşe numaralar verilmez?

6. Aşağıdaki listede BASIC değişkenleri için geçerli isimleri belirleyiniz.

   a) A *b)* BET *c)* C6 *d)* D*
   e) 7F *f)* A1 *g)* ZZ *h)* Z

7. BASIC aritmetik operatörlerini, sembolleri ile birlikte bir deyim içinde yerine getiriliş önceliklerine göre sıralayınız.

8. *a)* Aşağıdaki deyimlerde işlemlerin yerine getiriliş sırasını tanımlayınız.

```
LET Y = A + B * C
LET X = A↑2 + B* 7
LET P = SQR (A/7 + C - B * 3)
```

   b) A = 7, B = 1 ve C = 5 ise Y, X ve P değerlerini hesaplayınız.

9. Aşağıdaki BASIC deyimlerini açıklayınız.

   a) LET X = X + 1
   b) LET Y = R * SIN (A)
   c) PRINT "ŞİMDİ BİR SAYI GİRİNİZ";
   d) PRINT A, X, P
   e) GOSUB 290
   f) RETURN
   g) IF Z = 7 THEN 310
   h) INPUT N

10. Şekil 14.3 teki para atma programına, kullanıcı negatif bir atma sayısı girdiğinde END deyimine atlamaya sebep olacak bir deyim ekleyiniz.

11. Bir zar atışını gerçekleştiren bir program yazınız ve daha sonra iki zar atışını gerçekleştiren bir program yazınız. İki zarın birbirinden bağımsız olduğunu ve LET A = INT (6* RND (0)) + 1 deyiminin 1 ile 6 arasında bir sayı verdiğini hatırlayınız.

12. Şekil 14.4 te 40 nolu satırdaki deyimi en yakın 0.01°'ye yuvarlama yapacak şekilde yeniden yazınız.

13. Şekil 14.4 teki programı, Fahrenheit değerlerinin Şekil 14.7 deki programda test sonuçlarının girildiği gibi bir diziye girileceği şekilde yeniden yazınız.

14. Kayar-nokta sayısını ve sabit-nokta sayısını tarif ediniz.

15. Aşağıdakilerden hangileri geçerli FORTRAN değişken isimleridir?

   a) MODE *b)* BYTE *c)* FAHRENHEIT
   d) C3 *e)* CATCH-22 *f)* TO FAR

16. Aşağıdaki FORTRAN ifadelerinin anlamlarını açıklayınız.

a) Y = A - B/2
b) C = SQRT (A** 2 + B** 2)
c) B = ANINT (A/7)
d) Y. LE. X
e) A = R* (B + 6* (S/T) + 9)
f) Y = R* SIN (A)

17. Aşağıdakilerden hangileri geçerli FORTRAN deyim etiketleridir?

a) 27 *b)* LOOP *c)* D3 *d)* 999
e) C *f)* -23 *g)* 50

18. FORTRAN' daki TYPE deyimlerinin amacını açıklayınız.

19. Aşağıdaki FORTRAN deyimlerinin kullanımlarını açıklayınız.

a) DIMENSION
b) DATA
c) GO TO
d) IF (ifade) [stl, stl, stl]
e) IF (ifade) (deyim)
f) CALL [isim] (değişken, değişken, değişken, ...)
g) READ (BİRİM #, format stl) [değişken]
h) WRITE (BİRİM # format stl) [değişken]
i) FORMAT ( )

20. Aşağıdaki WRITE-FORMAT deyim çiftlerinin çıkışlarını tanımlayınız.

a) WRITE (4, 20) D3, İSİM, MAAŞ
b) 20 FORMAT (2x, I5, 10X; A20, 10X, F4.2)
c) WRITE (6, 40)
d) 40 FORMAT ('BİR HATA YAPTIN")

21. Değeri kullanıcı tarafından radyan olarak girilen bir açının sinüsünü basan bir FORTRAN programı yazınız.

22. Şekil 14.8a daki programı, girilen Celcius sıcaklaklıkları, Fahrenheit ve Kelvin'e çevirecek şekilde yeniden yazınız.

23. 0 ile 90° arasındaki her derecenin sinüsünü tablo şeklinde basan bir FORTRAN programı yazınız. SIN (X) öz fonksiyonu için X'in radyan olması gerektiğini unutmayınız.

24. Bir Pascal Programı için aşağıdakileri tanımlayınız.

a) Başlık *b)* Blok *c)* Tanımlamalar

25. Aşağıdakilerden hangileri Pascal'da geçerli tanımlamalardır?

a) VAR NUMBER: REAL,
b) VAR 7 SEC: INTEGER:
c) CONST e = 2.71828, PASS = 'YES';
d) VAR TEST 2: INTEGER;
e) VAR PASS: BOOLEAN

26. TYPE tanımlamasının Pascal'daki kullanımını açıklayınız.

27. Pascal'da bir FUNCTION ve bir PROCEDURE arasındaki fark nedir?

28. Pascal aritmetik operatörleri, en yüksek öncelikten en düşük önceliğe doğru sıralayınız.

29. 26 DIV 4 ve 26 MOD 4 deyimlerinin sonuçları nelerdir?

30. Aşağıdaki Pascal deyimlerinin etkilerini açıklayınız.

a) P: = P + 1;
b) CUBE (23)
c)
```
FOR K = 1 TO 40 DO
    BEGIN Y: = K;
      WRITELN (Y, Y*Y);
    END
```

d) REPEAT
READ (P);
WRITELN SIN (P);
UNTIL P > = 1.57

e) IF R > 2 THEN GO TO 222
ELSE IF R = 2 THEN GO TO 444
ELSE GO TO 888

31. WRITE ve WRITELN deyimleri arasındaki farkı açıklayınız.

32. Dosya teriminin iki anlamını tarif ediniz.

33. Girilen bir fiyat için yüzde 6 satış vergisini hesaplayan ve fiyatı, vergiyi ve toplamı yazan bir Pascal programı yazınız.

34. Şekil 14.11 a daki programı, girilen Celcius sıcaklıklarını, Fahrenheit ve Kelvin'e çeviren ve sonuçları basan bir program olacak şekilde değiştiriniz. (F = 9 C/5 + 32.)

REFERANSLAR/BİBLİOGRAFYA

Jensen, Kathleen, and Niklaus Wirth: *Pascal User Manual and Report*, 2d ed., Springer-Verlag, New York, 1978.

Bowles, K. L.: *MICROCOMPUTER Problem Solving Using Pascal*, Springer-Verlag, New York, 1977.

Haag, James N.: *Comprehensive Standard FORTRAN Programming*, Hayden Book Company, Rochelle Park, NJ, 1969, 1976.

Kieburtz, Richard B.: *Structured Programming and Problem Solving with Pascal*, Prentice-Hall, Englewood Cliffs, NJ, 1978.

Spencer, Donald D.: *Game Playing with BASIC*, Hayden Book Company, Rochelle Park, NJ, 1977.

Mullish, Henry: *A Basic Approach to BASIC*, Wiley, New York, 1976.

Meissner, Loren P.: *Rudiments of FORTRAN*. Addison-Wesley, Reading, MA, 1971.

Preliminary ISIS-II FORTRAN-80 COMPILER OPERATOR'S MANUAL, Intel Corporation, Santa Clara, CA, 1978.

FORTRAN-80 Programming Manual, Intel Corporation, Santa Clara, CA, 1978.

Krouse, Tim (Design Engineering): "*Pascal Language: Parts One throught Five*," Electronic Design, vols, 19, 20, 23, 25, September 13 and 27, November 8, December 6, 1978.

EK A

DURUM DİYAGRAMLARI VE ALGORİTMİK DURUM MAKİNE ŞEMALARI

En basit sayısal sistemler dışındaki sistemler senkrondur. Senkronun anlamı bütün çıkış değişimlerinin bir saat darbesi geldiği zaman olmasıdır. Şekil *A-1a JK* flip-floplarla yapılmış 2 bitlik ikilik senkron sayıcıyı göstermektedir. Hem *QA* hem de *QB* 0 olduğu zaman 2 nolu kapının *P* çıkışı yüksek olur. Eğer E sinyali yüksek ise *QB* ve *QA* çıkışları 00, 01, 10 ve 11 ikilik durumlarından defalarca geçer.

Bu devrenin çalışması, doğruluk tabloları, zamanlama dalga şekilleri veya Boole ifadeleri ile temsil edilebilir. Yaygın olarak kullanılan diğer iki grafiksel gösterim ise durum diyagramları ve algoritmik durum makine şemalarıdır.

DURUM DİYAGRAMLARI

Şekil A-1b, Şekil A-1a daki 2-bitlik ikilik sayıcı için bir durum diyagramını göstermektedir. Bir durum diyagramı devrenin geçtiği durumların sırasını gösterir. Aynı zamanda, saat darbesi geldiğinde devrenin bir durumdan başka bir duruma gitmesi için gerekli şartları gösterir.

Büyük oval balon, durumu belirleyen bir sembol ya da kod içerir. Oklar bir durumdan diğerine geçiş yönünü gösterir. Geçiş oklarının yanındaki semboller bir sonraki duruma geçmek için gerekli şartları ve bir önceki durum için çıkış seviyelerini gösterir. Şekil *A-1b* için bu sembollerin formatı *E/P, QB, QA* şeklindedir. Örneğin 0 durumundan 1 durumuna geçmek için *E 1* olmalıdır. 0 durumunda, P çıkışı yüksek, *QB* ve *QA* alçaktır. Büyük oval balonların solundaki dönen ok işaretleri, eğer kesme işaretinin solundaki şartlar mevcutsa sistemin bu durumda kalacağını belirtir. Örneğin durum 1 yakınındaki döngü, E alçak olduğu durumda sistemin 1 durumunda kalacağını belirtir. Saat darbelerinin bir etkisi olmayacaktır.

Mikroişlemciler karmaşık durum sistemlerinin veya makinelerin bir örneğidir. Şekil 9.19, bir 8085A mikroişlemcisi için geçiş durum diyagramını göstermektedir.

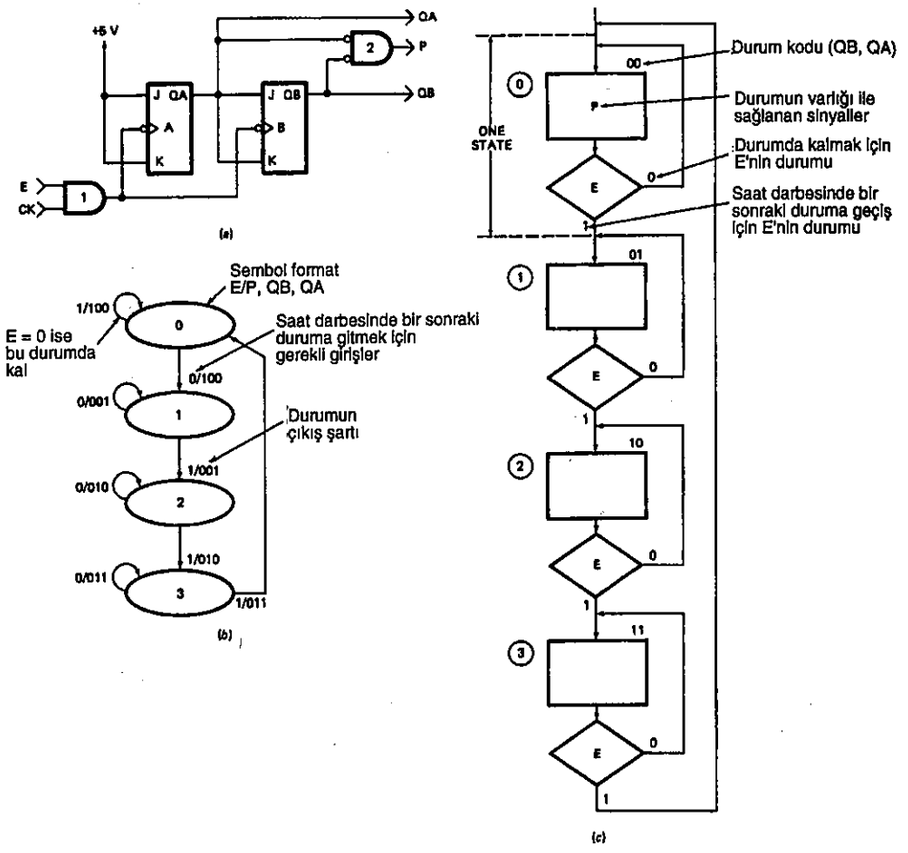
ALGORİTMİK DURUM MAKİNE ŞEMALARI

Bir ardışıl sistemin çalışmasını göstermenin yaygın bir grafiksel yolu *algoritmik durum makine* (ASM) *şemaları* kullanmaktır. Bir algoritma bir problemi çözmek için bir formül ya da metoddur. Sayısal sistemler veya makineler, bir durumlar serisinden geçerek kontrol problemlerini çözmekte kullanılırlar. Bu da *algoritmik durum makinesi* isminin orijinidir.

Şekil A-1*c*, Şekil A-1*a* daki basit ardışıl makine için ASM şemasını göstermektedir. Bir durum, bir dikdörtgen kutu ile temsil edilir. Durum ismi (sayısı) kutunun solundaki bir daire içindedir. Her durumun kodu durum kutusunun üstüne yazılır. Makinenin durum değiştirmesi için bazı şartların sağlanması gerekiyorsa, bu şartlar baklava dilimi şeklindeki karar kutuları içinde belirtilir. Örneğin Şekil A: 1*c* de, saat darbesi geldiğinde makinenin 0 durumundan 1 durumuna geçmesi için *E* sinyali 1 olmalıdır. Eğer bir durum için bir çıkış sinyali ile karşılaşılırsa, bu sinyalin adı durum kutusu içine yazılır. Şekil A-1 *c* de *P* sinyali 0 durumu ile ilgilidir ve bu yüzden durum 0 kutusu içine *P* yazılmıştır.

ÖZET

Durum diyagramları ve algoritmik durum makine şemaları, ardışıl sayısal sistemlerin tasarımları için çok güçlü araçlardır. Daha karmaşık sistemlerin tasarım ve analizlerinde durum diyagramlarını ve ASM şemalarını kullanmak ile ilgili daha geniş bilgi için aşağıdaki kitaplara başvurunuz.



Şekil A-1 : Saat yetkilendirmeli ve "0" durum algılayıcılı iki-bitlik ikilik sayıcı *a)* Devre *b)* Durum diyagramı *c)* Algoritmik durum makine şeması

REFERANSLAR / BİBLİYOGRAFYA

Fletcher, William I.: *An Engineering Approach to Digital Design*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

Peatman, John B.: *Digital Hardware Design*, McGraw-Hill, New York, 1980.

Winkel, David, and Franklin Prosser: *The Art of Digital Design: An Introduction to Top-Down Design*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

EK B

YENİ MANTIK SEMBOLLERİNİN AÇIKLAMASI*

İÇİNDEKİLER

BAŞLIK	SAYFA
1. GİRİŞ	806
2. SEMBOL KOMPOZİSYONU	806
3. BELİRTİCİ SEMBOLLER	808
3.1. Genel Belirtici Semboller	808
3.2. Girişler ve Çıkışlar İçin Belirtici Semboller	808
3.3. Çerçevenin İçindeki Semboller	811
4. Bağımlılık Gösterilişi	811
4.1. Genel Açıklama	811
4.2. G, VE	815
4.3. Bağımlılık Gösterilişinin Genel Uygulaması İçin Kurallar	816
4.4. V, VEYA	817
4.5. N, Değil (Özel VEYA)	818
4.6. Z, (Bağlantılar Arası)	818
4.7. C, Kontrol	819
4.8. S, Set ve R, Reset	820
4.9. EN, Yetkilendirme	821
4.10. M, Mod	822
4.11. A, Adres	825
5. İKİ KARARLI ELEMANLAR	828
6. KODLAYICILAR	829
7. ETKİLEYİCİ GİRİŞLERİ ÜRETMEK İÇİN BİR KODLAYICI KULLANIMI	830
8. ETKİLEYİCİ GİRİŞLERİ ÜRETMEK İÇİN İKİLİK GRUPLANDIRMA KULLANIMI	831
9. GİRİŞ ETİKETLERİNİN SIRASI	832
10. ÇIKIŞ ETİKETLERİNİN SIRASI	832

* *Bipolar Microcomputer Component Data Book for Design Engineers, 3. Baskı, 1981, Texas Instruments kitabından.*

TABLOLARIN LİSTESİ

TABLO	BAŞLIK	SAYFA
I.	Genel Belirtici Semboller	809
II.	Girişler ve Çıkışlar İçin Belirtici Semboller	812
III.	Çerçeve İçindeki Semboller	813
IV.	Bağımlılık Gösterilişinin Özeti	827

Yeni mantık sembollerinin bu açıklaması ile ilgili bir sorunuz varsa lütfen aşağıdaki adrese başvurunuz.

F.A. Mann MS 84
Texas Instruments Incorporated
P.O. Box 225012
Dallas, Texas 75265
Telefon (214) 995-3746

IEEE Standartları
Institute of Electrical and
Electronics Engineering
345 East 47 th Strect
New York, N.Y. 10017
adresinden alınabilir.

International Electrotechnical
Commission (IEC) yayınları
American National Standarts
Institute, Inc.
1430 Broadway
New York, N.Y. 10018
adresinden alınabilir.

1. GİRİŞ

Uluslararası Elektroteknik komisyonu (IEC), dahili mantığı göstermeden bir sayısal mantık devresinin her girişi ile her çıkışı arasındaki ilişkiyi gösterebilmek için çok kuvvetli bir sembolik dil geliştirmektedir. Bu bölümün temeli, Bölüm 4'te anlatılacak bağımlılık gösterilişidir.

Sistem, ABD'de IEEE/ANSI Standartı Y32. 14-1973 'te sunulmuştur. O zaman bağımlılık gösterilişinin tam olarak gelişmemesinden dolayı, VE, VEYA, değilleme gibi temel fonksiyonları temsil eden bildiğimiz ayırıcı dikdörtgen şekillerin yerine geçmekten çok daha fazla bir şey sunamadı. Fakat bugün durum aynı değildir.

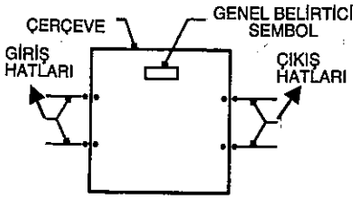
Uluslararası olarak IEC Teknik komite TC-3 'ün çalışma grubu 2, 1960'ların ortasında başlayan ve 1972 'de basılan (Yayın 117-15) orijinal çalışma ve bunu takip eden kanun maddeleri ve eklerini toplamak için yeni bir döküman (Yayın 617-12) hazırlamaktadır. ABD için IEEE komitesi SCC 11.9, IEEE Std 91/ANSI Y32.14 yayınını düzeltmektedir. Texas Instruments her iki organizasyonun çalışmalarına katılmaktadır ve TTL veri kitabına ait bu Ek yeni standartlar beklentisi için yeni mantık sembolleri sunmaktadır. Standartlar geliştikçe yapılan değişiklikler TTL veri kitabının ilerideki baskılarında göz önüne alınacaktır.

Yeni sembolik kitabın aşağıdaki açıklamaları bir özet şeklindedir ve standart yayımların sonunda içerildiğinden oldukça kısaltılmıştır. Bu, yeni cihazları geliştirecek kişilere yeterli olması amacıyla hazırlanmamıştır, amaç bu kitaptaki sembolleri anlaşılabilir yapmaktır; sembolleri fonksiyonel blok diyagramları ve/veya fonksiyon tabloları ile karşılaştırmak daha iyi anlamaya yol açacaktır.

2. SEMBOL KOMPOZİSYONU

Bir sembol, bir çerçeve veya çerçeve kombinasyonları ve bir ya da daha çok belirtici sembolden oluşmaktadır. Sembollerin şekilleri önemli değildir. Şekil 1'de gösterildiği gibi genel belirtici semboller, elemanlar tarafından hangi mantıksal işlemin yerine getirildiğini kesinlikle anlatmak için kullanılır. Giriş hatları sol tarafa, çıkış hatları sağ tarafa yerleştirilir. Bu gösterimde bir istisna yapılırsa sinyal akış yönü Şekil 11'de gösterildiği gibi bir okla belirtilir.

Daima benzeri dahili mantık durumlarına sahip, tek ve daha küçük parçalara bölünemez elemanın bütün çıkışları, elemanın içindeki bir etiket ya da



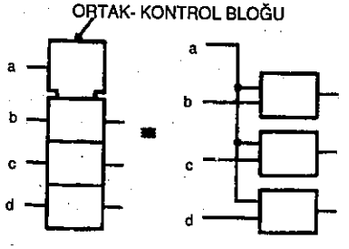
Şekil 1 : Sembol Kompozisyonu

ilgili bir belirtici sembol tarafından aksi belirtilmedikçe, elemanın fonksiyonu ile belirlenir.

Elemanların çerçeveleri, takip eden düzenler geçerli olduğu zaman bitişik olabilir veya gömülebilir.

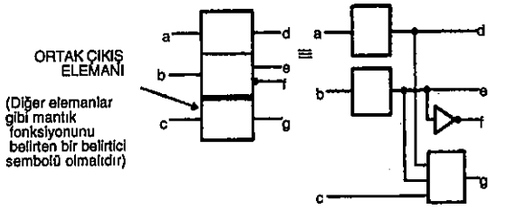
Çerçevelerin ortak hattı, sinyal akış yönünde olduğu zaman elemanlar arasında bir mantık bağlantısı yoktur. Çerçevelere ortak hat sinyal akış yönüne dik olduğu zaman elemanlar arasında en azından bir mantık bağlantısı vardır. Elemanlar arasındaki mantık bağlantıları sayısı belirtici sembollerin kullanılmasıyla açıklığa kavuşacaktır ve bu ileride aynı başlık altında daha geniş olarak tartışılacaktır. Eğer ortak hattın her iki yanında bir belirti yoksa yalnızca bir bağlantı olduğu varsayılacaktır.

Bir devre, bir elemandan daha fazla elemana ortak olan bir ya da daha fazla girişe sahip olduğunda ortak-kontrol bloğu kullanılabilir. Bu IEC sisteminde kullanılanlardan farklı şekillendirilmiş tek çerçevedir. Bağımlılık gösterilişi tarafından aksi belirtilmedikçe Şekil 2, ortak-kontrol bloğuna olan bir girişin ortak-kontrol bloğu altındaki her elemana giriş olduğunu göstermektedir.



Şekil 2 : Ortak-kontrol bloğunun gösterimi

Dizideki bütün elemanlara bağlı olan ortak çıkış, bir ortak-çıkış elemanının çıkışı gibi gösterilebilir. Ayırıcı görsel özelliği üstündeki çift çizgidir. Şekil 3'te gösterildiği gibi ortak-çıkış elemanının başka girişleride olabilir. Ortak çıkış elemanının fonksiyonu genel belirtici semboller tarafından gösterilmelidir.



Şekil 3 : Ortak-çıkış elemanının gösterimi

3. BELİRTİCİ SEMBOLLER

3.1. Genel Belirtici Semboller

Tablo I, bu veri kitabında kullanılan genel belirtici sembolleri göstermektedir. Bu karakterler, sembol ya da eleman tarafından temsil edilen cihazın temel fonksiyonunu tanımlamak için, sembol ya da sembol elemanının üst orta kısmına veya geometrik merkezine yerleştirilir.

3.2. Girişler ve Çıkışlar İçin Belirtici Semboller

Girişler ve çıkışlar için belirtici semboller Tablo II'de gösterilmiştir ve mantık polaritesi ve analog sinyal belirticileri dışında birçoğu, kullanıcılara aşina olmalıdır. Eski mantıki çevirme göstericinin anlamı harici 0 durumunun dahili 1 durumu üretmesidir. Dahili 1 durumu aktif durum demektir. Mantıki çevirme safi mantık diyagramlarında kullanılabilir; harici 1 ve 0 mantık durumlarını, yüksek ve alçak seviyelere bağlamak için pozitif mantığın veya negatif mantığın kullanıldığı belirtilmeli ya da kabul edilmelidir. Mantık polarite göstericileri, kullanılan mantık gösterimini belirtme ihtiyacını ortadan kaldırır ve bu kitapta gerçek cihazları sembolleştirmede kullanılmaktadır. Üçgen polarite göstericinin varlığı alçak mantık seviyesinin dahili 1 durumu (aktif durum) üreteceğini belirtir, ya da bir çıkış durumunda dahili 1 harici alçak seviye üretecektir. Bir dinamik giriş için geçişin aktif

yönünün pozitif mantık, negatif mantık ve polarite göstermekle nasıl belirtildiğine dikkat ediniz. Dahili giriş ve çıkış uygulamaları Bölüm 4'te anlatılacak bağımlılık gösterilişinin anlaşılmasını gerektirir.

Bir elemanlar dizisinde aynı genel belirtici sembol ya da giriş ve çıkışlarla ilgili belirtici sembol, dizinin bütün elemanlarının içinde gözükecekse bu belirtici elemanlar genellikle yalnızca birinci elemanda gösterilir. Bu karmaşıklığı önlemek ve tanımayı hızlandırmak için yapılır. Benzer şekilde daha küçük elemanlara bölünen büyük benzer elemanlar bölünmemiş bir çerçeve ile temsil edilirler. SN54LS440 sembolü bu prensibi göstermektedir.

TABLO I : GENEL BELİRTİCİ SEMBOLLER

SEMBOL	TANIM	ÖRNEK
&	VE kapısı yada fonksiyonu	SN7400
>1	VEYA kapısı yada fonksiyonu. Sembol çıkışı aktif yapmak için en azından bir girişin aktif yapılması gerektiğini belirtmek için seçilmiştir.	SN7402
=1	Özel VEYA. Çıkışı aktif yapmak için bir ve yalnız bir giriş aktif olmalıdır.	SN7486
=	Mantık eşitlik. Bütün çıkışlar aynı durumda durmalı.	SN74180
2K	Çift sayıda giriş aktif olmalı	SN74180
2K+1	Tek sayıda giriş aktif olmalı	SN74180
1	Bir giriş aktif olmalı	*
Δ veya Δ	Alışılmış çıkış kapasitesinden fazla kapasitesi olan tampon veya eleman (sembol sinyal akış yönüne göre yönlendirilmiştir)	SN74S436
x/y	Schmitt tetikleyici, histerisisli eleman	SN74LS18
	Kodlayıcı, kod çevirici (DEC/BCD, BIN/OCT, BIN/7-Parçalı)	SN74LS347

SEMBOL	TANIM	ÖRNEK
MUX	Çoklayıcı, veri seçici	SN74150
DMUX veya DX	Çoktan bire indirgeyici	SN74138
Σ	Toplayıcı	SN74LS385
P-Q	Çıkarıcı	SN74LS385
CPC	İleri elde üretici	SN74182
π	Çarpıcı	SN74LS384
COMP	Büyüklük karşılaştırıcı	SN74LS682
ALU	Aritmetik Mantık Ünitesi	SN74LS381
	Yeniden tetiklenebilir tek kararlı	SN74LS422
	Yeniden tetiklenemez tek kararlı (tek-atış)	SN74121
	Kararsız eleman. Dalgaşeklini göstermek isteğe bağlı	SN74LS320
	Senkron olarak başlayan kararsız	SN74LS624
	Tamamlanan darbe ile duran kararsız eleman	
SRGm	Kayma yazmacı, m= bit sayısı	SN74LS695
CTRm	Sayıcı m= bit sayısı 2^m döngü uzunluğu	SN74LS590
CTR DIVm	Döngü uzunluğu m olan sayıcı	SN74LS668
ROM	Yalnızca okunan bellek	SN74187
RAM	Rasgele erişimli okuma/yazma belleği	SN74170
FIFO	İlk giren-ilk çıkan bellek	SN74LS222
I-0	Güç açılışı elemanı 0 durumuna götürür	SN74AS877
ϕ	Çok karmaşık fonksiyon sınırlı detaylı "gri kutu" sembolü özel kararlar altında gösterilir.	SN74LS608

Genel belirtici sembollerin hepsi bu kitapta kullanılmamıştır, fakat burada bütünlüğü bozmamak için ilave edilmişlerdir.

3.3. Çerçeve İçindeki Semboller

Tablo III çerçeve içinde kullanılan bazı sembolleri göstermektedir. Özellikle açık-emiter, açık-kollektör ve üç-durumlu çıkışların değişik sembolleri olduğuna dikkat ediniz. Bir *EN* girişinin devrenin bütün çıkışlarını etkilediğine ve girişler üzerinde etkisi olmadığına dikkat ediniz. Bir yetkilendirme girişi yalnızca belli çıkışları ve/veya bir ya da daha fazla girişi etkilediğinde, bir bağımlılık gösteriliş şekli bunu belirtecektir (Bakınız 4.9). EN girişinin değişik tipte çıkışlar üzerindeki etkileri gösterilmiştir.

Bir *D* girişinin, bir depolama elemanının veri girişi olduğuna özellikle dikkat etmek önemlidir. Dahili 1 durumunda, *D* girişi depolama elemanını kendi 1 durumuna ayarlar, ve dahili 0 durumunda depolama elemanını kendi 0 durumuna resetler.

İkilik gruplama sembolleri daha geniş olarak Bölüm 8'de açıklanacaktır. İkilik-ağırlıklı girişler sırayla düzenlenmiştir ve en küçük değerlikli ve en büyük değerlikli hatların ikilik ağırlıkları numaralarla belirtilmiştir. Bu veri kitabında giriş ve çıkış hatlarının ağırlıkları yalnızca ikilik gruplama sembolü kullanıldığında ikinin üsleri ile temsil edilirler, aksi halde onluk sayılar kullanılacaktır. Gruplanmış girişler, matematik fonksiyonların işleyeceği ya da bağımlılık gösterilişi için tanımlayıcı sayı olabilecek dahili bir sayı üretir. Şekil 28 'e bakınız. Bellek adreslerinde sıkça kullanılırlar.

Yönler değiştirildiğinde ikilik gruplama sembolleri, çıkışlarla kullanılabilir. Kavram, girişler için kullanılan kavramlarla aynıdır ve ağırlıklı çıkışlar devre içinde gelişmesi beklenen dahili sayıyı belirtir.

Bu kitapta çerçeveler içinde kullanılan diğer semboller IEC/IEEE standartlarına göredir fakat burada gösterilmemişlerdir. Genellikle bunlar aritmetik işlemlerle ilgilidirler ve kendi kendine açıklayıcıdır.

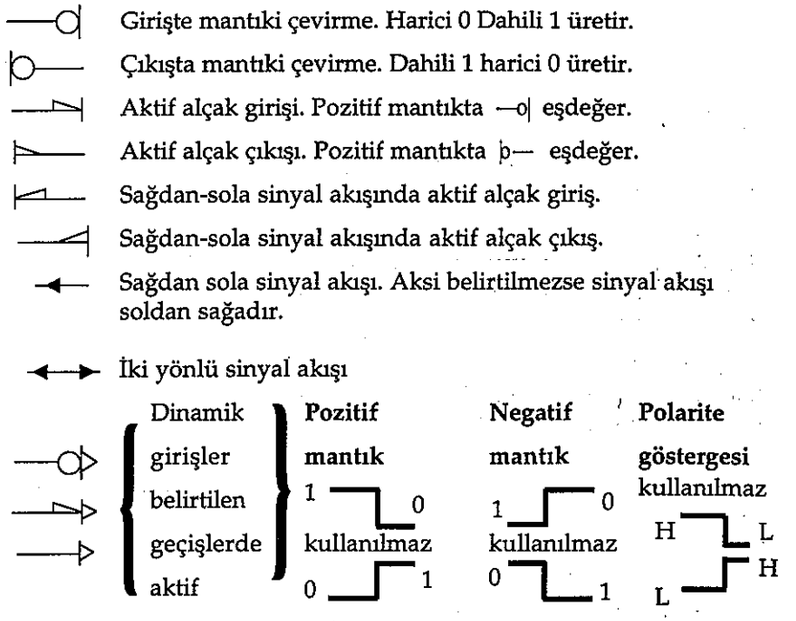
Bir çerçeve içinde standart olmayan bir bilgi gösterildiğinde genellikle köşeli parantezler içine alınır [bunun gibi].

4. BAĞIMLILIK GÖSTERİLİŞİ

4.1. Genel Açıklama

Bağımlılık gösterilişi, IEC sembollerini önceki sistemlerden ayıran ve özlü, anlamlı sembolleri mümkün kılan kuvvetli bir araçtır. Girişler, çıkışlar ve giriş ve çıkışlar arasındaki ilişkileri, ilgili bağlantı ve bütün elemanları göstermeden temsil etme olanağını sağlar. Bağımlılık gösterilişi tarafından sağlanan bilgi, bir elemanın fonksiyonu için belirtici sembol tarafından sağlanan bilgiyi tamamlar.

TABLO II : GİRİŞ VE ÇIKIŞLAR İÇİN BELİRTİCİ SEMBOLLER

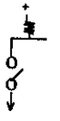


TABLO III : ÇERÇEVE İÇİNDEKİ SEMBOLLER

Geciktirilömiş çıkış(1 darbe tetiklemeli bir flip-flopun). Çıkış, değişikliği başlatan giriş (örneğin bir C girişi) başlangıçtaki harici durum ya da seviyesine döndüğünde değişir. Bakınız §5

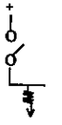
İki eşikli giriş (Histerizisli giriş)

Kapatılmadığı zaman karşılaştırmalı olarak düşük-empedans alçak seviye sağlayabilen NPN açık-kollektör veya benzeri çıkış. Harici *yukarı çekiş (pull-up)* gerektirir. Pozitif-mantık telli-VE bağlantısı yapılabilir.



Pasif *yukarı çekiş* çıkışı *(pull-up)* NPN açık-kollektör çıkışına benzer fakat dahili bir pasif yukarı-çekmeye sahiptir. (+5V'a yükselir)

Kapatılmadığı zaman karşılaştırılmalı olarak düşük empedans yüksek seviye sağlayabilen NPN açık emiter veya benzeri çıkış. Harici aşağı çekme gerektirir. Pozitif-mantık telli-VEYA bağlantısı yapılabilir.



Pasif aşağı çekiş çıkışı NPN açık emiter çıkışına benzer

fakat dahili bir aşağı çekmeye sahiptir. (0'a düşer)

3-durumlu çıkış

EN — Yetkilendirme girişi

Dahili 1 durumundayken bütün çıkışlar yetkilendirilir. Dahili 0 durumundayken açık-kollektör ve açık-emiter çıkışları açıktır, üç-durumlu çıkışlar normal olarak tanımlanmış dahili mantıksal ve harici yüksek-empedans durumundadır ve bütün diğer çıkışlar dahili 0 durumundadır.

J, K,R,S,T / D — Flip-floplarla ilgili olağan tanımlar (örneğin : *R* = Reset, *T* = Togel)



elemanına eşdeğer bir depolama elemanına veri girişi

→ m / ← m — Sağa (sola) kayma girişleri, $m = 1, 2, 3, \ldots$ $m = 1$ ise, genellikle gösterilmez.

+m / -m — Yukarı (aşağı) sayma girişleri, $m = 1, 2, 3, \ldots$ $m = 1$ ise, genellikle gösterilmez.



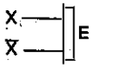
İkilik guruplama *m*, 2 nin en yüksek üssüdür.

CT = 15 — İçerik-ayarlayan giriş, aktif olduğunda bir kaydedicinin içeriğinin belirtilen değeri almasına sebep olur.

CT = 8 — İçerik çıkışı, kaydedicinin içeriği belirtildiği gibi olduğunda aktif olur.



Giriş hattı gruplama. İki ya da daha fazla terminalin bir mantık girişini etkilediğini belirtir. Örnek : SN7450'nin çiftleştirilmiş genişletme girişleri



"1" — Sabit durum çıkışı daima dahili 1 durumunda kalır. Örnek için SN74185 'e bakınız

Bir sembol içinde birbirine dayanmış mantık elemanları arasındaki dahili bağlantılar gösterilen sembollerle belirtilebilir. Her mantık bağlantısı ortak çizginin bir ya da her iki kenarındaki belirtici sembollerin varlığıyla gösterilebilir ve bağlantı sayısı hakkında bir karışıklık ortaya çıkarsa dahili bağlantı sembollerinden biri kullanılabilir.

Dahili (zahiri) giriş, devrenin başka bir yerinde ortaya çıkan ve doğrudan bir terminale bağlı olmayan bir giriştir. Dahili (zahiri) bir çıkışta benzeri şekilde bir terminale doğrudan bağlanmamıştır.

Bağımlılık gösterilişi için kabul edilen düzende etkileyen ve etkilenen terimleri kullanılacaktır. Hangi girişlerin etkileyen ya da etkilenen olduğunun açık olmadığı durumlarda (örneğin bir VE ilişkisinde), seçim uygun bir şekilde yapılacaktır.

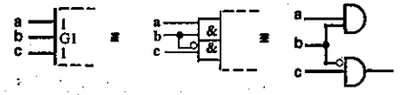
Bugüne kadar on çeşit bağımlılık tanımlanmıştır ve bunların hepsi bu veri kitabında kullanılmıştır. Aşağıda sunuldukları sırayla listelenmiş ve 4.11 'i takip eden Tablo IV'te özetlenmiştir.

Bölüm	Bağımlılık tipi veya diğer konular
4.2	G, VE
4.3	Bağımlılık Gösterilişi için Genel Kurallar
4.4	V, VEYA
4.5	N, Çevirme (ÖZEL VEYA)
4.6	Z, Bağlantılar Arası
4.7	C, Kontrol
4.8	S, Set ve R, Reset
4.9	EN, Yetkilendirme
4.10	M, Mod
4.11	A, Adres

4.2. G (VE) Bağımlılığı

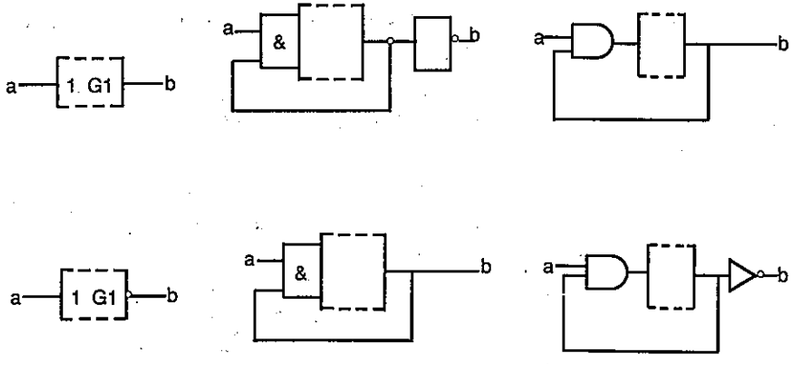
İki sinyal arasındaki yaygın bir ilişki bu sinyalleri *VE*'lemektir. Bu, geleneksel olarak bir *VE* kapısı ve kapının girişlerine bağlanan girişlerle gösterilmiştir. 1972 IEC yayımı ve 1973 IEEE/ANSI standardı, bağımlılık gösterilişini kullanarak VE ilişkisini göstermenin çeşitli yollarını göstermiştir. Bundan sonra diğer dokuz bağımlılık şekilleri geliştirilirken *VE* bağımlılığını göstermenin yolları bire düşürülmüştür.

Şekil 4'te *b* girişi *a* girişi ile VE'lenmiş ve *b*'nin tümleyeni *c* ile *VE*'lenmiştir. *G* harfi *VE* ilişkisini belirtmek için seçilmiş ve *b* girişinde sembol içine yerleştirilmiştir. Sembol tasarımlayıcı tarafından uygun görülen bir sayı (burada 1) *G* harfinden sonra ve etkilenen her girişten sonra konulmuştur. c girişindeki 1 üzerindeki çizgiye dikkat ediniz.



Şekil 4 : Girişler arasındaki G bağımlılığı

Şekil 5'te *b* çıkışı, *a* girişini bir *VE* ilişkisiyle etkiler. Altta olan örnek *VE*'lenenin b'nin dahili mantık durumu (çevirme işareti ile etkilenmeyen) olduğunu gösterir. Şekil 6, *a* girişinin dinamik *b* girişiyle *VE*'leneceğini gösterir.



Şekil 5 : Çıkış ve girişler arasındaki G bağımlılığı



Şekil 6 : Bir Dinamik girişle *G* bağımlılığı

G bağımlılığının kuralları şu şekilde özetlenebilir.

Bir *Gm* giriş veya çıkışı (*m* bir sayı) dahili 1 durumunda bekliyorsa, *Gm* tarafından etkilenen bütün giriş ve çıkışlar normalde tanımlanan dahili mantık durumlarında kalır. *Gm* giriş veya çıkışı kendi 0 durumunda bekliyorsa, *Gm* tarafından etkilenen bütün giriş ve çıkışlar kendi dahili 0 durumlarında kalır.

4.3. Bağımlılık Gösterilişinin Genel Uygulaması İçin Kurallar

Bağımlılık ilişkilerini uygulama kuralları genelde *G* bağımlılığı için gösterilen sıranın aynısını takip eder.

Bağımlılık gösterilişini uygulamak

1) Diğer giriş ve çıkışları etkileyen girişi (çıkışı), ilgili ilişkiyi belirten harf sembolünü (örneğin *VE* için *G*) takip eden ve uygun olarak seçilmiş tanımlayıcı bir sayı ile etiketleyerek ve,

2) Bu etkileyen giriş (veya çıkış) tarafından etkilenen her giriş ve çıkışı aynı sayıyla etiketleyerek başarılır.

Eğer etkilemeyi yapan, etkileyici giriş veya çıkışın dahili mantık durumunun tümleyeni ise, etkilenen giriş veya çıkıştaki tanımlayıcı sayının üzerine bir çizgi yerleştirilir. Şekil 4'e bakınız.

Etkileyen iki giriş veya çıkış aynı harfe ve aynı tanımlayıcı sayıya sahipse bunlar birbiriyle bir *VEYA* ilişkisi gösterirler. Şekil 7'ye bakınız.



Şekil 7 : *VEYA*lanmış etkileyen girişler

Etkilenen giriş veya çıkış fonksiyonunu göstermek için bir etiket gerekiyorsa (örneğin *"D"*), bu etiketten önce etkileyici girişin tanımlayıcı sayısı kullanılır. Şekil 12'ye bakınız.

Bir giriş veya çıkış birden fazla etkileyen giriş tarafından etkilenirse, etkileyen girişlerin herbirinin tanımlayıcı sayısı virgüllerle ayrılmış şekilde etkilenenin etiketinde gösterilir. Bu sayıları normal okuma sırası etkileyen ilişkilerin sırasıyla aynıdır. Şekil 12'ye bakınız.

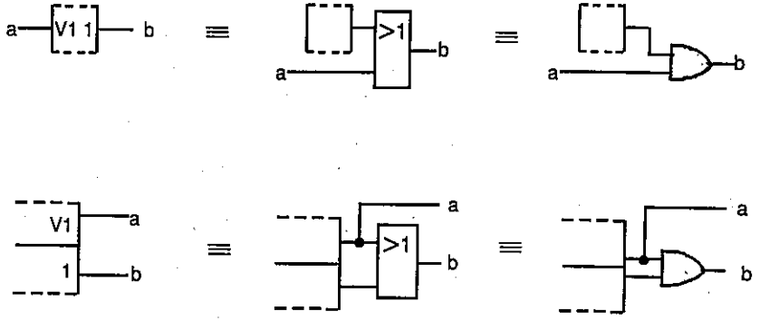
Etkilenen giriş veya çıkışların fonksiyonlarını temsil eden etiketler sayı olmak zorundaysa (örneğin bir kodlayıcının çıkışları), etkileyen girişler veya etkilenen girişler ve çıkışlarla ilgili tanımlayıcı sayılar karışıklığı önlemek için seçilen başka karakterlerle değiştirilir (örneğin Yunan harfleri). Şekil 8'e bakınız.



Şekil 8 : Sayıları değiştirme

4.4. V(VEYA) Bağımlılığı

VEYA bağımlılığını temsil eden sembol *V*'dir. Şekil 9'a bakınız.

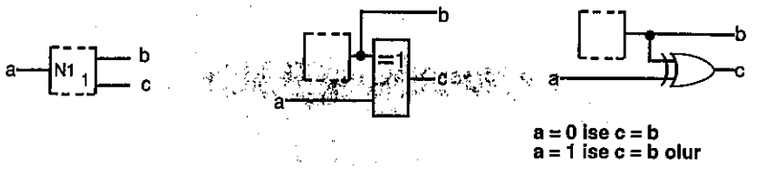


Şekil 9: *V (VEYA)* bağımlılığı

Bir V_m giriş ya da çıkışı kendi dahili 1 durumunda ise, V_m tarafından etkilenen bütün giriş ve çıkışlar kendi dahili 1 durumunda kalır. V_m giriş veya çıkışı kendi dahili 0 durumunda ise, V_m tarafından etkilenen bütün giriş ve çıkışlar normal olarak tanımlanmış kendi dahili mantık durumlarında kalır.

4.5. N (Çevirme) (ÖZEL VEYA) Bağımlılığı

Çevirme bağımlılığını temsil eden sembol *N* harfidir. Şekil 10'a bakınız. Bir N_m giriş veya çıkışı tarafından etkilenen her giriş veya çıkış, N_m giriş veya çıkışı ile bir özel *VEYA* ilişkisi içindedir.



Şekil 10 : N (çevirme) (ÖZEL VEYA) bağımlılığı

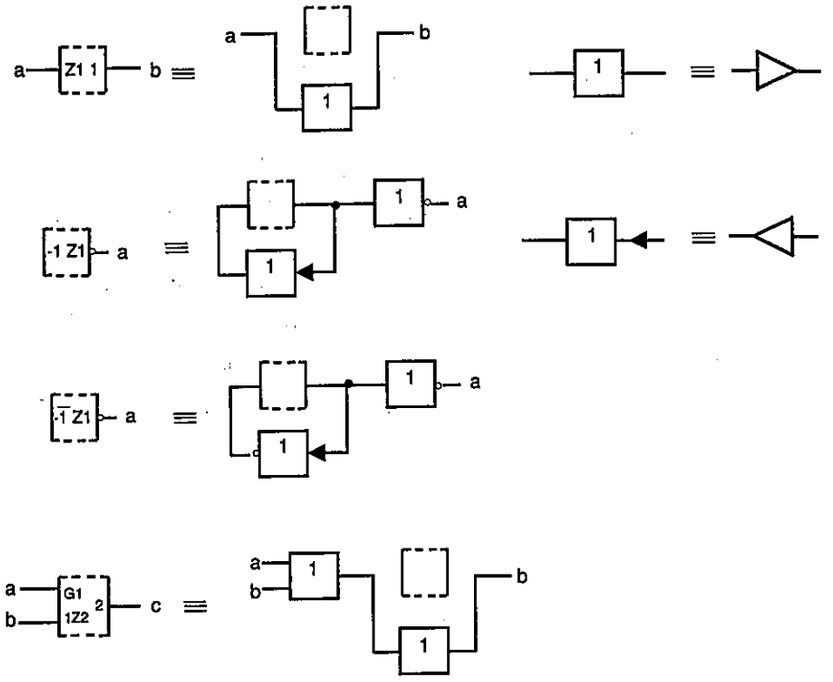
Bir N_m giriş veya çıkışı kendi dahili 1 durumundaysa, N_m tarafından etkilenen her giriş veya her çıkışın dahili mantık durumu normal durumun tümleyenidir. Bir N_m giriş veya çıkışı kendi dahili 0 durumundaysa, N_m tarafından etkilenen bütün giriş ve çıkışlar normal olarak tanımlanmış kendi dahili mantık seviyelerinde kalır.

4.6. Z (Bağlantılararası) Bağımlılık

Bağlantılararası bağımlılığı temsil eden sembol *Z* harfidir.

Bağlantılararası bağımlılık girişler, çıkışlar, dahili girişler, ve/veya dahili çıkışlar arasındaki dahili mantık bağlantılarını belirtmek için kullanılır.

Bir Z_m giriş veya çıkışı tarafından etkilenen bir giriş veya çıkışın dahili mantık durumu ilave bağımlılık gösterilişi tarafından değiştirilmezse, Z_m giriş veya çıkışının dahili mantık durumuyla aynı olacaktır. Şekil 11'e bakınız.



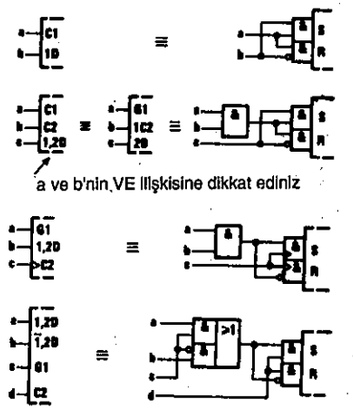
Şekil 11 : *Z* (Bağlantılararası) bağımlılığı

4.7. C (Kontrol) Bağımlılığı

Kontrol bağımlılığını temsil eden sembol *C* harfidir.

Kontrol girişleri genellikle depolama elemanlarının veri girişlerini (*D*, *J*, *K*, *R* veya *S*) yetkilendirmek veya yetkisizleştirmek için kullanılır. Kendi dahili 1 durumlarını (aktif olma) statik veya dinamik olarak alabilir. İkinci durumda, Şekil 12'deki üçüncü örnekte olduğu gibi dinamik giriş sembolü kullanılır.

Bir C_m giriş veya çıkışı kendi dahili 1 durumundaysa, C_m tarafından etkilenen girişler elemanın fonksiyonu üzerinde normal olarak tanımlanmış etkilere sahip olacaklardır, yani girişler yetkilendirilecektir. Bir C_m giriş veya çıkışı kendi dahili 0 durumundaysa, C_m tarafından etkilenen girişler yetkisizleştirilir ve elemanın fonksiyonu üzerinde etkileri olmaz.



Şekil 12 : *C* (Kontrol) Bağımlılığı

4.8. S (Set) ve R (Reset) Bağımlılıkları

Set bağımlılığını temsil eden sembol *S* harfidir. Reset bağımlılığını temsil eden sembol R harfidir.

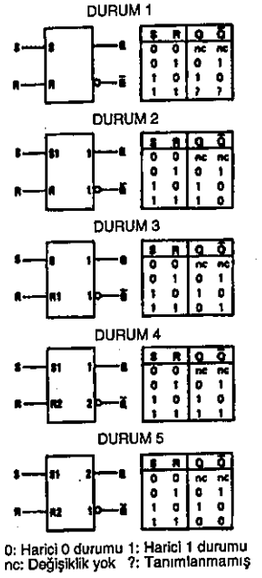
Set ve Reset bağımlılıkları iki kararlı bir elemanda $R = S = 1$ kombinasyonun etkisinin belirtilmesi gerektiğinde kullanılır. Şekil 13'teki durum 1, *S* veya *R* bağımlılığını kullanmaz.

Bir S_m girişi kendi dahili 1 durumundaysa, S_m girişi tarafından etkilenen çıkışlar *R* girişinin durumuna bakmaksızın $S = 1, R = 0$ kombinasyonuna cevap verirmiş gibi tepki gösterirler. Şekil 13'teki 2, 4 ve 5 durumlarına bakınız.

Bir R_m girişi kendi 1 durumundaysa, R_m girişi tarafından etkilenen çıkışlar, *S* girişinin durumuna bakmaksızın $S = 0$, $R = 1$ kombinasyonuna cevap verirmiş gibi tepki gösterir. Şekil 13'teki 3, 4 ve 5 durumlarına bakınız.

S_m veya R_m girişi kendi dahili 0 durumundaysa etkisi yoktur.

4 ve 5 durumlarındaki tümleyensiz çıkış dizileri yalnızca geçici olarak kararlıdırlar. Girişlerin her ikisinin aynı anda $S = R = 0$ durumuna dönüşü, önceden görülemeyen kararlı ve tümleyenli çıkış dizisi yaratır.

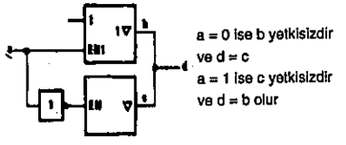


Şekil 13 : *S* (Set) ve *R* (Reset) Bağımlılıkları

4.9. EN (Yetkilendirme) Bağımlılığı

Yetkilendirme bağımlılığını temsil eden sembol *EN* harflerinin kombinasyonudur.

Bir EN_m girişinin çıkışlar üzerinde *EN* girişine benzer etkisi vardır (Bakınız 3.1), fakat yalnızca tanımlayıcı sayı m ile etiketlenen çıkışları etkiler. Buna karşılık bir *EN* girişi bütün çıkışları etkiler ve girişleri etkilemez. Bir EN_m girişinin etkilenen bir giriş üzerindeki etkisi C_m girişinin etkisine benzer. Şekil 14'e bakınız.



Şekil 14 : EN (Yetkilendirme) Bağımlılığı

Bir EN_m girişi kendi dahili 1 durumundaysa, EN_m tarafından etkilenen girişler elemanın fonksiyonu üzerinde normal olarak tanımlanan kendi etkilerine sahiptir ve bu giriş tarafından etkilenen çıkışlar normal olarak tanımlanmış kendi dahili mantık durumunda kalırlar, yani bu girişler ve çıkışlar yetkilendirilir.

Bir EN_m girişi kendi dahili 0 durumundaysa, EN_m tarafından etkilenen girişler yetkisizleştirilir ve elemanın fonksiyonu üzerinde bir etkisi olmaz ve EN_m tarafından etkilenen çıkışlar da yetkisizleştirilir. Açık-kollektör çıkışları kapatılır, üç-durumlu çıkışlar kendi normal olarak tanımlanmış dahili mantık durumundadırlar, fakat yüksek empedans gösterirler ve bütün diğer çıkışlar kendi dahili 0 durumunda kalır.

4.10. M (Mod) Bağımlılığı

Mod bağımlılığını temsil eden sembol M harfidir.

Mod bağımlılığı bir elemanın giriş ve çıkışlarının etkisinin elemanın çalıştığı moda bağlı olduğunu belirtmek için kullanılır.

Eğer bir giriş veya çıkış değişik çalışma modları için aynı etkiye sahipse, geçerli etkileyici M_m girişlerinin tanımlayıcı sayıları etkilenen giriş veya çıkış etiketlerinde parantezler arasında ve kesme işareti ile ayrılmış olarak gösterilir. Şekil 19 'a bakınız.

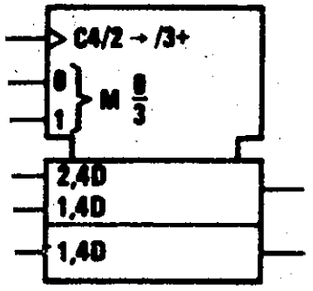
4.10.1. Girişleri Etkileyen M Bağımlılığı

M bağımlılığı, girişleri *C* bağımlılığı gibi etkiler. Bir M_m giriş veya çıkışı kendi dahili 1 durumundaysa, bu M_m giriş veya çıkışı tarafından etkilenen girişler elemanın fonksiyonu üzerinde normal olarak tanımlanmış etkilerini göstereceklerdir, yani girişler yetkilendirilir.

Bir M_m giriş veya çıkışı kendi dahili 0 durumundaysa, bu M_m giriş veya çıkışı tarafından etkilenen girişlerin elemanın fonksiyonu üzerinde bir etkisi olmaz. Etkilenen bir girişin kesmelerle ayrılmış birkaç etiket kümesi varsa (örneğin C4/2→ /3+), M_m giriş veya çıkışının tanımlayıcı sayısı olan bir setin etkisi yoktur ve ihmal edilebilir. Bu yanlış çalışan bir girişin bazı fonksiyonlarının yetkisizleştirilmesini temsil eder.

Şekil 15'teki devrenin herhangi bir anda olacak dört moddan birini (0, 1, 2 ve 3) kontrol eden *b* ve *c* girişleri vardır. *d*, *e* ve *f* girişleri, *a* girişi tarafından dinamik kontrola (saatli) tabi *D* girişleridir. Seri içindeki 1 ve 2 sayıları, modları belirtmek için seçilmiştir; böylece *e* ve *f* girişleri yalnızca mod 1'de

yetkilendirilir (paralel yükleme için) ve *d* girişi yalnızca *Mod* 2'de yetkilendirilir (seri yükleme için). a girişinin üç fonksiyonu vardır. Veri girişi için saattir. *Mod* 2'de verinin sağa kaymasına neden olur, bunun anlamı kontrol bloğundan uzağa kaymaktır. *Mod* 3'te kaydedicinin içeriğinin bir artmasına sebep olur.



Bütün işlemler senkron değildir

Mod 0 da (b= 0, c= 0), girişlerin etkisi olmadığı için çıkışlar o andaki durumlarında kalır.

Mod 1'de (b= 1, c= 0), e ve f girişlerinden paralel yükleme olur

Mod 2'de (b= 0, c= 1), d 'den aşağı doğru kayma ve seri yükleme olur.

Mod 3'de (b= 1, c= 1), her saat darbesinde yukarı doğru 1 artarak sayma meydana gelir.

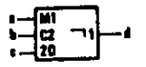
Şekil 15 : Girişleri etkileyen *M* (mod) bağımlılığı

4.10.2. Çıkışları Etkileyen M Bağımlılığı

Bir M_m girişi veya M_m çıkışı kendi dahili 1 durumundaysa, etkilenen çıkışlar normalde tanımlanan dahili mantık durumlarında kalır yani çıkışlar yetkilendirilir.

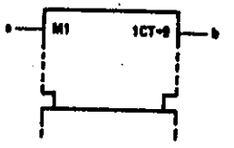
Bir M_m girişi veya M_m çıkışı kendi dahili 0 durumundaysa, her çıkışta bu M_m giriş veya çıkışının tanımlayıcı sayısını taşıyan etiket setinin etkisi yoktur ve ihmal edilebilir. Çıkış, kesme işareti ile ayrılmış değişik etiketler setine sahipse (örnek : 2,4 / 3,5), yalnızca bu M_m girişi veya M_m çıkışının tanımlayıcı sayısının gözüktüğü kümeler ihmal edilmelidir.

Şekil 16'da mod 1, bir çıkış kendi dahili 1 durumundayken mevcuttur. Gecikmiş çıkış sembolü yalnızca devre darbe-tetiklemeli flip-flop gibi çalıştığı mod 1 'de (a girişi 1) etkilidir. Bölüm 5'e bakınız. Giriş a = 0 olduğunda devre mod 1 durumunda değildir, böylece gecikmiş çıkış sembolünün bir etkisi yoktur ve devre şeffaf kilit gibi çalışır.



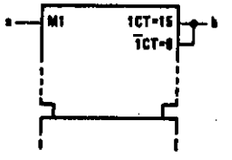
Şekil 16 : Flip-flop tipinin mod ile belirlenmesi

Şekil 17'de a girişi mod 1'i gerçekleştiren kendi dahili 1 durumundaysa, *b* çıkışı yalnızca kaydedicinin içeriği 9'a eşit olduğu durumda kendi dahili 1 durumunda olur. Çıkış *b*, ortak-kontrol bloğu içinde yalnızca mod 1 fonksiyonuyla tanımlandığından, a girişi kendi dahili 0 durumunda olduğunda, kaydedicinin içeriğine bakmaksızın bu çıkışta kendi dahili 0 durumunda kalacaktır.

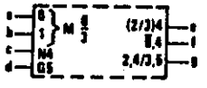


Şekil 17 : Ortak - kontrol bloğunun çıkışını yetkisizleştirme

Şekil 18'de a girişi mod 1'i gerçekleştiren kendi dahili 1 durumundaysa, b çıkışı yalnızca kaydedicinin içeriği 15'e eşit olduğu durumda kendi dahili 1 durumunda olur. a girişi kendi dahili 0 durumundaysa, b çıkışı yalnızca kaydedicinin içeriğinin 0 olması durumunda kendi dahili 1 durumunda kalır.



Şekil 18 : Bir çıkışın fonksiyonlarını belirleme



Şekil 19 : Mod tarafından etkilenen bağımlılık ilişkileri

Şekil 19'da *a* ve *b* girişleri 0, 1, 2 ve 3'ü üretmek için ikili ağırlıklıdır. Bu, dört moddan hangisinin mevcut olduğunu belirler.e çıkışında çevirmeye sebep olan etiket (*c*= 1 ise), yalnızca 2 ve 3 modlarında etkilidir. *Mod 0* ve 1 'de bu çıkış hiç etiket yokmuş gibi kendi normal tanımlanmış durumunda kalır. f çıkışındaki etiket kümesi, mod 0 olmadığı durumda etkilidir, böylece e çıkışı 1, 2 ve 3 modlarında çevrilir (c = 1 ise). *Mod 0*'da etiket setinin bir etkisi yoktur, böylece çıkış kendi normal tanımlanmış durumunda kalır. Bu örnekte 0,4 (1/2/3) 4'e eşdeğerdir. g çıkışında iki etiket seti vardır. Çevirmeye sebep olan (*c= 1* ise) ilk set yalnızca mod 2'de etkilidir. g ve d üzerinde VE bağımlılığına tabi ikinci set yalnızca mod 3'te etkilidir.

Mod 0'da hiçbir bağımlılık ilişkisinin çıkışlar üzerinde etkisi olmadığına dikkat ediniz, böylece *e, f* ve *g* aynı durumda kalır.

4.11. A (Adres) Bağımlılığı

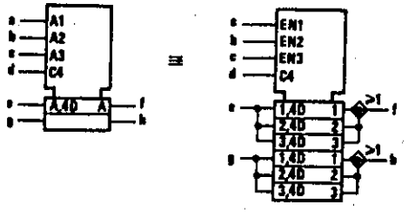
Adres bağımlılığını temsil eden sembol *A* harfidir. Adres bağımlılığı çok boyutlu dizilerin belirtilen bir bölümünü seçmek için adres kontrol girişlerini kullanan elemanların (özellikle belleklerin) açıkça temsil edilmelerini sağlar. Bellek dizisinin böyle bir bölümü bir kelime olarak adlandırılır. Adres bağımlılığının amacı bütün dizinin sembolik temsilini sağlamaktır. Bu genel bölümün belirli bir elemanında gösterilen bir giriş, dizinin seçilen bütün bölümlerindeki karşı gelen elemanlara ortaktır. Bu genel bölümün belirli bir elemanında gösterilen bir çıkışın etiketi, bu çıkışın bir açık-devre çıkışı veya bir üç-durumlu çıkış olduğunu belirtirse, bu belirtme yalnızca dizinin çıkışını belirtir, dizinin bölümlerinin çıkışını belirtmez.

Bir etkileyen adres girişi tarafından etkilenmeyen girişler, dizinin bütün bölümlerinde kendi normal olarak tanımlanmış etkilerini gösterir. Fakat, bir adres girişi tarafından etkilenen girişler, yalnızca bu adres girişi tarafından seçilen bölümler üzerinde kendi normal olarak tanımlanmış etkilerini gösterir.

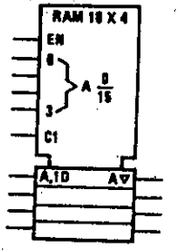
Bir etkileyen adres girişi, *A* harfi ve bu harfi takip eden ve bu giriş tarafından seçilen dizinin belirli bölümünün adresine karşı gelen bir tanımlayıcı sayı ile etiketlenir. Sembol tarafından temsil edilen genel bölge içinde A_m girişi tarafından etkilenen giriş ve çıkışlar *A* harfi ile etiketlenir.

Şekil 20, 2-bitlik 3-kelimelik belleği her kelime için ayrı bir adres hattı ile göstermektedir ve işlemi açıklamak için EN bağımlılığı kullanmaktadır. Kelime 1'i seçmek için a girişi kendi 1 durumuna getirilir, bu da mod 1 'i gerçekleştirir. Bu durumda veri "1,4 *D*" işaretli girişlerden girilebilir. 2 ve 3 nolu kelimeler seçilmedikçe, veri "2,4 *D*" ve "3,4 *D*" işaretli girişlerden girilemez. Çıkışlar, yalnızca aktif EN fonksiyonlar tarafından seçilmiş çıkışların VEYA lanmış çıkışlarıdır.

Etkileyen adres girişlerinin tanımlayıcı sayıları bu girişler tarafından seçilen bölümlerin adreslerine karşı gelir. Bunlar diğer etkileyen bağımlılık girişlerinden (örneğin : *G, V, N,* ...) farklı olmak zorunda değildir, çünkü sembol tarafından temsil edilen genel bölümde *A* harfi ile değiştirilir. Dizinin bölümlerine bağımsız olarak ve belkide aynı anda erişmek amacıyla, A_m girişine etkileyen değişik kümeler varsa, *A* harfi 1*A*, 2*A*, ... olarak değiştirilir. Dizinin aynı bölümlerine erişmelerinden dolayı, *A* girişlerinin bu kümeleri aynı tanımlayıcı sayılara sahip olabilir. Şekil 21 bu kavramın bir başka gösterimidir.



Şekil 20 : *A*(Adres) bağımlılığı



Şekil 21 : 16-kelime *X* 4-bitlik Rasgele erişimli bellek oluşturan 3-durum çıkışlı dörtşeffaf kilitli 16 bölüm dizisi.

TABLO IV : Bağımlılık Gösterilişinin Özeti

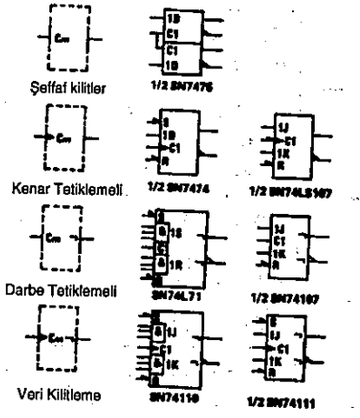
Bağımlılık Tipi	Harf Sembolü	1- DURUMUNDAKİ ETKİLEYEN GİRİŞ	0 - DURUMUNDAKİ ETKİLEYEN GİRİŞ
Adres	A	Hareket izin verir (Adres seçilir)	Hareketi önler (Adres seçilmez)
Kontrol	C	Harekete izin verir	Hareketi önler
Yetkilendirme	EN	Harekete izin verir	Girişlerin hareketini önler ◊ çıkışları kapalı ∇ çıkışları harici yüksek empedansta, dahili mantık durumunda değişiklik yok Diğer çıkışlar dahili 0 durumunda
VE	G	Harekete izin verir	0 durumunu zorlar
Mod	M	Harekete izin verir	Hareketi önler (mod seçilmez)
Çevirme (ÖZEL VEYA)	N	Durumun tümleyenini alır	Etkisiz
RESET	R	Etkilenen çıkış S= 0, R= 1 gibi tepki gösterir	Etkisiz
SET	S	Etkilenen çıkış S= 1, R= 0 gibi tepki gösterir	Etkisiz
VEYA	V	1 durumuna zorlar	Harekete izin verir
Bağlantılararası	Z	1 durumuna zorlar	0 durumuna zorlar

**Bu harf sembolleri etkileyen girişte (veya çıkışta) görülür ve bir sayı tarafından takip edilir. Bu giriş tarafından etkilenen her giriş (veya çıkış) aynı sayıyla etiketlenir. EN, R ve S etiketlerinden sonra sayı gelmezse yukarıdaki tanım uygulanmaz. Bu girişlerin hareketleri bölüm 3.1'de çerçeve içindeki semboller başlığı altında anlatılmıştır.*

5. İKİ KARARLI ELEMANLAR

Dinamik giriş sembolü, gecikmiş çıkış sembolü ve bağımlılık gösterilişi dört ana tipteki iki kararlı elemanları ayırma araçlarını sağlar, ve senkron ve asenkron girişlerin kolayca ayırdedilmesini yapar. Şekil 22'ye bakınız. Birinci sütun gerekli ayırıcı özellikleri, diğer sütunlarda örnekleri gösterir.

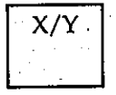
Şeffaf kilitler seviye-çalışmalı kontrol girişine sahiptir. *C* girişi kendi dahili 1 durumunda olduğu süre içinde *D* girişi aktiftir. Çıkışlar anında cevap verir. Kenar-tetiklemeli elemanlar *C*'nin aktif geçişinde *D, J, K, R* ve *S* girişlerinden veri kabul ederler. Darbe - tetiklemeli elemanlar kontrol darbesi gelmeden önce verinin hazır olmasını gerektirirler. Veri, *C* girişi kendi 1 durumunda olduğu sürece aynen tutulacağı için *C* girişi statik olarak kabul edilir. *C*, 0 durumuna dönünceye kadar çıkış geciktirilir. Veri-çıkış-kilitleme elemanı *C* 'nin aktif geçişten kısa bir süre sonra dinamik kabul edilmesinin dışında, darbe-tetiklemeli şekline benzer veri girişleri yetkisizleşmiştir ve veri tutulmak zorunda değildir. Fakat, *C* girişi kendi başlangıç harici seviyesine dönünceye kadar çıkış geciktirilir. Senkron girişlerin bağımlılık etiketleriyle (*1D, 1J, 1K, 1S, 1R*), *C* girişlerine bağlı olmayan asenkron girişlerle (*S, R*) karşılaştırıldıklarında kolayca tanınabildiklerine dikkat ediniz.



Şekil 22 : ***İki kararlı*** **Devrelerin Dört Tipi**

6. KODLAYICILAR

Bir kodlayıcı ya da kod çeviricinin genel sembolü Şekil 23'te gösterilmiştir. *X* ve *Y*, giriş ve çıkışlardaki bilgiyi temsil etmek için kullanılan kodun uygun belirticileri ile değiştirilebilir.



Şekil 23 : Kodlayıcı Genel Sembolü

Kod çevirmenin belirtilmesi aşağıdaki kurala bağlıdır:

Giriş koduna bağlı olarak girişlerin dahili mantık durumları bir dahili değer belirtir. Bu değer çıkış koduna bağlı olarak çıkışların dahili mantık durumları tarafından yeniden üretilir.

Girişlerin dahili mantık durumları ile dahili değer arasındaki ilişkinin gösterilmesi aşağıdaki şekilde başarılır:

1) Girişleri sayılarla etiketleyerek. Bu durumda dahili değer kendi dahili 1 durumunda olan girişlerle ilgili ağırlıkların toplamına eşittir.

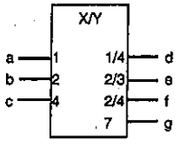
2) X'i giriş kodunun uygun gösterimi ile değiştirerek ve girişleri bu kodu belirten karakterle etiketleyerek.

Dahili değer ile çıkışların dahili mantık durumları arasındaki ilişkiler şu şekillerde belirtilir:

1) Her çıkışı, çıkışın dahili 1 durumuna yol açan dahili değerleri temsil eden sayılarla etiketleyerek. Bu sayılar Şekil 24'te gösterildiği gibi bir kesme işareti ile ayrılmalıdır. Bu etiketleme, *Y*'nin bir bağımlılık tipini temsil eden bir harfle değiştirildiği zaman da kullanılabilir. (Bölüm 7'ye bakınız). Dahili değerlerin sürekli bir aralığı bir çıkışın dahili 1 durumunu üretiyorsa bu aralığın başlangıç ve bitişini belirten iki sayı aralarında üç nokta ile kullanılabilir. Örneğin : 4 ... 9 = 4/5/6/7/8/9 gibi.

2) *Y*'yi çıkış kodunun uygun bir gösterimi ile değiştirerek ve çıkışları Şekil 25'te gösterildiği gibi bu kodu belirten karakterle etiketleyerek.

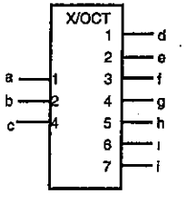
Alternatif olarak genel sembol, girişler ve çıkışlar arasındaki ilişkilerin belirtildiği bir tabloya yapılan uygun bir referanslada kullanılabilir. Bu bir *PROM*'u programladıktan sonra sembolleştirmek için tavsiye edilen bir yoldur.



Doğruluk Tablosu

Girişler			Çıkışlar			
c	b	a	g	f	e	d
0	0	0	0	0	0	0
0	0	1	0	0	0	1
0	1	0	0	0	1	0
0	1	1	0	1	1	0
1	0	0	0	1	0	1
1	0	1	0	0	0	0
1	1	0	0	0	0	0
1	1	1	1	0	0	0

Şekil 24 : *X/Y Kod Çevirici*



Doğruluk Tablosu

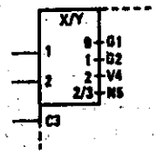
Girişler			Çıkışlar						
c	b	a	j	i	h	g	f	e	d
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	1
0	1	1	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	1	0	0	0	0
1	1	0	0	1	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0

Şekil 25 : *X/Sekizlik Kod Çevirici*

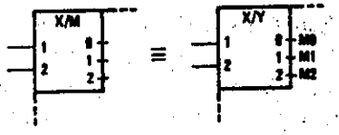
7. ETKİLEYİCİ GİRİŞLERİ ÜRETMEKTE KODLAYICI KULLANIMI

Bazı durumlarda bağımlılık notasyonu için etkileyici girişlerinin bir seti, bir elemanın belirli girişlerindeki sinyallerin kodlarını çözerek üretilir. Bu durumda kodlayıcı sembolü gömülmüş bir sembol olarak kullanılır. Şekil 26'ya bakınız.

Bir kodlayıcı tarafından üretilen bütün etkileyen girişler aynı tipte ise ve tanımlayıcı sayıları kodlayıcının çıkışında gösterilen numaralara karşı geliyorsa *Y* (belirtici sembol *X/Y* içindeki), bağımlılık tipini temsil eden harf ile değiştirilebilir. Bu durumda etkileyen girişlerin belirticileri silinmelidir. Şekil 27'ye bakınız.



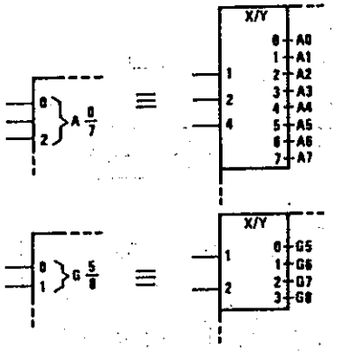
Şekil 26 : Değişik tipteki bağımlılıkların üretilmesi



Şekil 27 : Bir tip bağımlılığın üretilmesi

8. ETKİLEYİCİ GİRİŞLERİN ÜRETİLMESİ İÇİN İKİLİK GRUPLAMANIN KULLANILMASI

Bir kodlayıcı tarafından üretilen bütün etkileyen girişler aynı tipte iseler ve kodlayıcının çıkışlarında gösterilen numaralara karşı gelmeleri şart olmayan ardışıl tanımlayıcı sayıları varsa ikilik gruplama sembolü kullanılabilir (3.1'e bakınız). k tane harici hat 2^k tane dahili giriş üretir. Bağımlılık tipini temsil eden harfi takip eden parantez $\frac{m1}{m2}$ tarafından takip edilir. Şekil 28'de gösterildiği gibi, m1 en küçük tanımlayıcı sayıyla ve m2 en büyük tanımlayıcı sayıyla değiştirilmelidir.



Şekil 28 : İkilik Gruplama Sembolünün Kullanımı

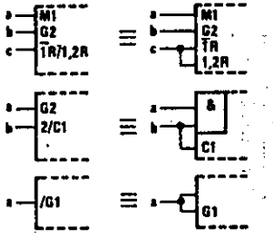
9. GİRİŞ ETİKETLERİNİN SIRASI

Bir girişin, diğer girişler tarafından etkilenen bir tek fonksiyonel etkili girişi varsa, bu fonksiyonel etkinin belirtici sembolünün (eğer varsa) önünde etkileyici girişlere karşı gelen etiketler bulunur. Önde gelen bu etiketlerin soldan-sağa sırası, etkilerin veya değişikliklerin uygulanması gereken sıradır. Birbirinden ayrı olarak gözönüne alınan etkileyici girişlerden herhangi birinin mantık durumu etkilenen girişin etkisiz olmasına sebep olursa, etkilenen girişin eleman üzerinde fonksiyonel etkisi olmaz.

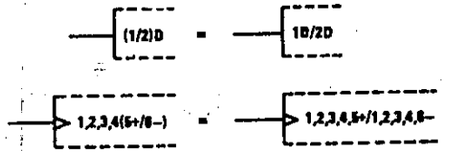
Bir giriş birkaç farklı fonksiyonel etkilere veya birkaç farklı etkileyici girişler kümesine sahipse, hareketin moduna bağlı olarak, giriş gerektiği sıklıkta gösterilebilir. Fakat bu gösterim metodunun dezavantajlı olduğu durumlar vardır. Bu durumlarda giriş, kesme işareti ile ayrılmış farklı etiket setleri ile bir defa gösterilir. Şekil 29'a bakınız. Bu etiket kümelerinin sırasının bir anlamı yoktur. Bir girişin fonksiyonel etkilerinden biri elemanın etiketsiz girişlerinden biriyse, gösterilen etiketlerin ilk setinden önce bir kesme işareti kullanılır.

Bir birleşim elemanın bütün girişleri yetkisizleştirilmişse (elemanın fonksiyonu üzerinde hiç etkisi yoksa), elemanın çıkışlarının dahili mantık durumları sembol tarafından belirtilmez. Bir ardışıl elemanın bütün girişleri yetkisizleştirilmişse, bu elemanın içeriği değişmez ve çıkışlar o andaki dahili mantık seviyelerinde kalır.

Etiketler cebirsel teknikleri kullanarak parçalanabilir.



Şekil 29 : Giriş Etiketleri



Şekil 30 : Giriş etiketlerini Parçalama

10. ÇIKIŞ ETİKETLERİNİN SIRASI

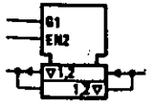
Bir çıkış, etkileyici girişlerin veya çıkışların tanımlayıcı sayıları olup olmadıklarına bakmaksızın, birkaç değişik etikete sahipse bu etiketler aşağıdaki sırayla gösterilirler:

1) Gecikmiş çıkış sembolü gösterilecekse, bu sembol ilk olmalıdır. Uygulanacakları girişlerin belirticileri gerekirse bu sembolün önünden gelir.

2) Daha sonra çıkışların dahili mantık durumlarındaki değişmeleri belirten etiketler gelir. Bu etiketlerin soldan-sağa sırası, etkilerinin uygulanacağı sırayla aynıdır.

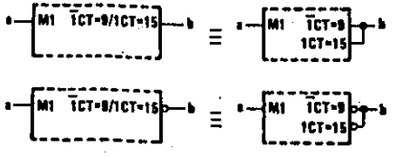
3) Daha sonra çıkışın, girişler ve elemanın diğer çıkışları üzerindeki etkilerini belirten etiket kullanılır.

Açık-devre veya *üç-durumlu çıkış sembolleri*, çıkış hattına komşu sembolün dış çerçevesinin hemen içine yerleştirilir. Şekil 31'e bakınız.



Şekil 31 : 3-durumlu sembollerin yerleştirilmesi

Bir çıkışın, alternatif fonksiyonları temsil etmek için birkaç farklı etiketler kümesine ihtiyacı varsa (örneğin hareket moduna bağlı olarak), bu kümeler çerçevenin dışına bağlanacak değişik çıkış hatları üzerinde gösterilebilir. Fakat, bu gösterim metodunun avantajlı olmadığı durumlar vardır. Bu durumlarda giriş, kesme işaretleri ile ayrılmış değişik etiket kümeleri ile bir defa gösterilir. Şekil 32'ye bakınız.

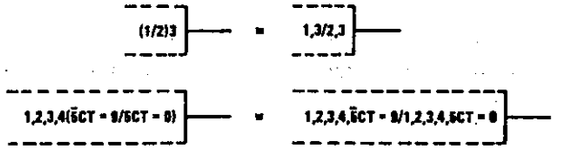


Şekil 32 : Çıkış etiketleri

Bir etiket kümesi içindeki etkileyen girişlerin nümerik karakterlerle ayrılmamış iki komşu tanımlayıcı sayısı virgül ile ayrılmalıdır.

Bir kesme işareti olmayan bir çıkışın etiket seti, kendi dahili 0 durumundaki bir etkileyen M_m girişinin tanımlayıcı sayısını içeriyorsa, bu etiket kümesinin bu çıkış üzerinde bir etkisi yoktur.

Etiketler cebirsel teknikleri kullanarak parçalanabilir.



Şekil 33 : Çıkış Etiketlerinin Parçalanması

Yeni Mantık Sembollerinin Bu Açıklaması İle İlgili Bir Sorunuz Varsa Lütfen Aşağıdaki Adrese Başvurunuz.

F.A.Mann MS 84
Texas Instruments Incorporated
P.O. Box 225012
Dallas, Texas 75265
Telefon (214) 995-3746

IEEE Standartları Institute of Electrical and Electronics Engineering 345 East 47th Strect New York, N.Y. 10017 adresinden alınabilir.

International Electrotechnical Commission (IEC) Yayınları American National Standarts Institute, Inc.
1430 Broadway
New York, N.Y. 10018
adresinden alınabilir.

EK C

VERİ SAYFALARI:

INTEL 2716

ZILOG Z 80

İÇİNDEKİLER

	SAYFA
INTEL 2716 16K (2K X8) UV SİLİNEBİLİR PROM	836
Bacak Yapısı	837
DC Karakteristikleri	838
AC Karakteristikleri	838
AC Dalgaşekilleri	840
Silme Karakteristikleri	840
Elemanın Çalışması	
ZILOG Z80 VE Z80A	843
Bacak Yapısı	846
Zamanlama Dalgaşekilleri	848
Komut Seti	849
Z80 AC Karakteristikleri	853
AC Zamanlama Diyagramı	856
DC Karakteristikleri	857
Z 80A AC Karakteristikleri	858

INTEL

2716*

16 K (2K X 8) UV SİLİNEBİLİR PROM

- *Hızlı Erişim Zamanı*
  - **350 ns max. 2716-1**
  - **390 ns max. 2716-1**
  - **450 ns max. 2716**
- *Tek +5V Güç Kaynağı*
- *Düşük güç harcaması*
  - **525 mW max. Aktif güç**
  - **132 mW max. Bekleme gücü**
- *Intel 5V ROM lara (2316 E, 2332 ve 2364) ve 2732 EPROM'a bacak uyumlu*
- *Basit Programlama Ihtiyaçları*
  Tek adresli programlama
  Bir 50 ms darbeli programlar
- *Okuma ve programlama sırasında TTL uyumlu giriş ve çıkışlar*
- *Tamamen statik*

Intel 2716, 16384-bitlik, morötesi silinebilir ve elektriksel olarak programlanabilir yalnızca - okunur bellektir *(EPROM). 2716* tek 5V güç kaynağıyla çalışır, statik bekleme modu vardır ve hızlı, tek adres yerli programlama özelliği vardır. *EPROM'lar tasarımı hızlı, kolay ve daha ekonomik yapar.* Üretim için, 2716 kullanması hızla bacak uyumlu 16 K ROM (2316E) veya yeni 32 K veya 64K ROM'lara (2332 ve 2364) dönebilir.

Tek 5 Voltluk kaynak ve 350 ns'lik erişim zamanı ile *yüksek performanslı* yeni +5V mikroişlemcileri, Intel 8085 ve 8086 ile kullanılmak için çok idealdir. 2716, aynı zamanda, erişim zamanını arttırmadan güç harcamasını düşüren statik bekleme modlu ilk *EPROM* dur. *Maksimum aktif güç harcaması 525 mW*'tır ve maksimum bekleme güç harcaması ise yalnızca 132 mW'tır ve bu %75'lik bir tasarruftur.

** 18 ve 20 nolu bacaklar bütün 16K, 32 K ve 64 K EPROM ve ROM ailesi ile uyum için yeniden isimlendirilmiştir. Kalıp, fabrikasyon süreci ve özellikler aynı kalır ve bu değişiklikten hiç etkilenmez.*

2716 bugüne kadar geliştirilen en basit ve hızlı *EPROM* programlama metodu olan tek darbe TTL seviye programlama metoduna sahiptir. *Yüksek gerilim darbelere ihtiyaç yoktur*, çünkü bütün programlama kontrolları *TTL* sinyalleri tarafından yapılır. Böylece kart üstünde, sistem içinde, arazide programlamak mümkündür. Herhangi bir adresi herhangi bir zamanda *2716*'nın tek adres yeri programlaması ile tek tek, sırasıyla ya da rasgele programlamak mümkündür. 16384 bitin hepsini programlamak yalnızca 100 saniyedir.

PROGRAMLAMA

Programlama özellikleri Data Catalog PROM / ROM Programming Instructions kitabında 4-83. sayfada anlatılmıştır.

Mutlak Maksimum Değerler*

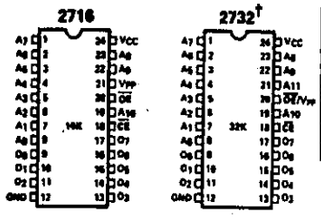
Kutuplama altında sıcaklık ..... -10 °C ile + 80°C arasında

Depolama sıcaklığı ..... -65°C ile +125°C arasında

Toprağa göre bütün giriş ve çıkış gerilimleri ..... +6 V ile -0,3 V arasında

Programlama sırasında toprağa göre Vpp kaynak gerilimi ..... +26,5 V ile -0,3 V arasında

BACAK YAPISI*

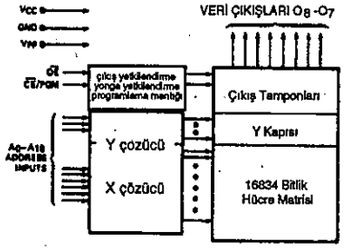


† Özellikler için 2732'nin veri kağıdına bakınız

MOD SEÇİMİ

Bacaklar / MOD	CE/PGM (18)	OE (20)	V_{pp} (21)	V_{cc} (24)	Çıkışlar (9-11, 13-17)
Okuma	V_{IL}	V_{IL}	+5	+5	$D_{çıkış}$
Bekleme	V_{IH}	Önemsiz	+5	+5	Yüksek Z
Program	V_{IL}, V_{IH}'ye Darbelenir	V_{IH}	+25	+5	$D_{giriş}$
Program Onaylama	V_{IL}	V_{IL}	+25	+5	$D_{çıkış}$
Program Yasaklama	V_{IL}	V_{IH}	+25	+5	Yüksek Z

BLOK DİYAGRAMI



BACAK İSİMLERİ

Aç-Ag	Adresler
CE/PGM	Yonga Yetkilendirme/Programlama
OE	Çıkış Yetkilendirme
O_8–O_7	Çıkışlar

** UYARI "Mutlak Maksimum Değerler" altında listelenen değerlerin üzerinde çalışmaya zorlamak elemanın bozulmasına sebep olabilir. Bu yalnızca zorlama değeridir, ve elemanın bu değerde fonksiyonel çalışması veya bu tarifnamenin çalışma bölümlerinde belirtilen değerlerin üstünde çalışması kastedilmez. Uzun süreli olarak mutlak maksimum değerlere maruz bırakmak elemanın güvenilirliliğini etkileyebilir.*

Okuma sırasında DC ve Ac çalışma şartları

	2716	2716-1	2716-2
Sıcaklık Aralığı	0°C – 70°C	0°C – 70°C	0°C – 70°C
V_{cc} güç kaynağı [1,2]	5V ± 5%	5V ± 10%	5V ± 5%
V_{pp} güç kaynağı [2]	V_{CC} ± 0.6V [3]	V_{CC} ± 0.6V [3]	V_{CC} ± 0.6V [3]

OKUMA İŞLEMİ

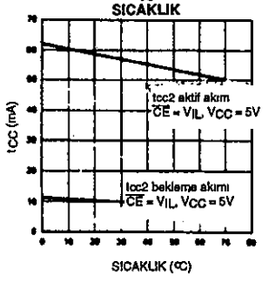
DC ve Çalışma Karakteristikleri

Sembol	Parametre	Limitler Min.	Tipik [4]	Max.	Birim	Şartlar
I_{LI}	Giriş Yük Akımı			10	µA	V_{IN} = 5.25V
I_{LO}	Çıkış Kaçak Akımı			10	µA	V_{OUT} = 5.25V
I_{PP1} [2]	V_{pp} Akımı			5	mA	V_{PP} = 5.85V
I_{CC1} [2]	V_{cc} Akımı (Bekleme)		10	25	mA	$\overline{CE}$ = V_{IH}, $\overline{OE}$ = V_{IL}
I_{CC2} [2]	V_{cc} Akımı (Aktif)		57	100	mA	$\overline{OE}$ = $\overline{CE}$ = V_{IL}
V_{IL}	Giriş Alçak Gerilim	-0.1		0.8	V	
V_{IH}	Giriş Yüksek Gerilim	2.0		V_{CC}+1	V	
V_{OL}	Çıkış Alçak Gerilim			0.45	V	I_{OL} = 2.1 mA
V_{OH}	Çıkış Yüksek Gerilim	2.4			V	I_{OH} = -400 µA

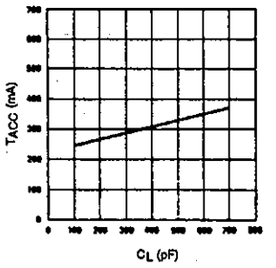
Notlar

1. V_{CC} V_{pp} ile aynı anda veya daha önce uygulanmalı ve V_{pp} ile aynı anda veya daha sonra ayrılmalıdır.
2. V_{pp}, programlama süresi dışında doğrudan V_{cc}'ye bağlanabilir. Bu durumda kaynak akımı I_{CC} ve I_{PP1}'in toplamı olmalıdır.
3. 0.6 Voltluk tolerans, V_{pp} kaynağı bacağını okuma sırasındaki V_{cc}'den programlama için 25 Volta anahtarlayan bir sürücü devresi kullanılmasına izin verir.
4. Tipik değerler T_A = 25°C ve nominal kaynak gerilimleri içindir.
5. Bu parametre yalnızca örneklendi ve % 100 test edilmedi.

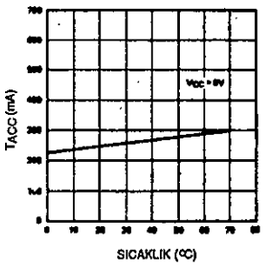
TİPİK KARAKTERİSTİKLER
t_{CC} AKIMI ve SICAKLIK



ULAŞMA ZAMANI ve KONDANSATÖR



ULAŞMA ZAMANI ve KONDANSATÖR



Sembol	Parametre	2716-1 Limitleri			2716-1 Limitleri			2716-2 Limitleri			Birim	Test Şartları
		Min	Typ[4]	Max	Min	Typ[4]	Max	Min	Typ[4]	Max		
t_{ACC}	Adresten çıkış gecikmesine			450			350			390	ns	$\overline{CE} = \overline{OE} = V_{IL}$
t_{CE}	$\overline{CE}$ den çıkış gecikmesine			450			350			390	ns	$\overline{OE} = V_{IL}$
t_{OE}	çıkış yetkilendirmeden çıkış gecikmesine			120			120			120	ns	$\overline{CE} = V_{IL}$
t_{DF}	çıkış yetkilendirme yükselten çıkış yüzere	0		100	0		100	0		100	ns	$\overline{CE} = V_{IL}$
t_{OH}	Adresten çıkış tutmaya	0			0			0			ns	$\overline{CE} = \overline{OE} = V_{IL}$

Kondansatör $T_A = 25°C$, f = 1 MHz

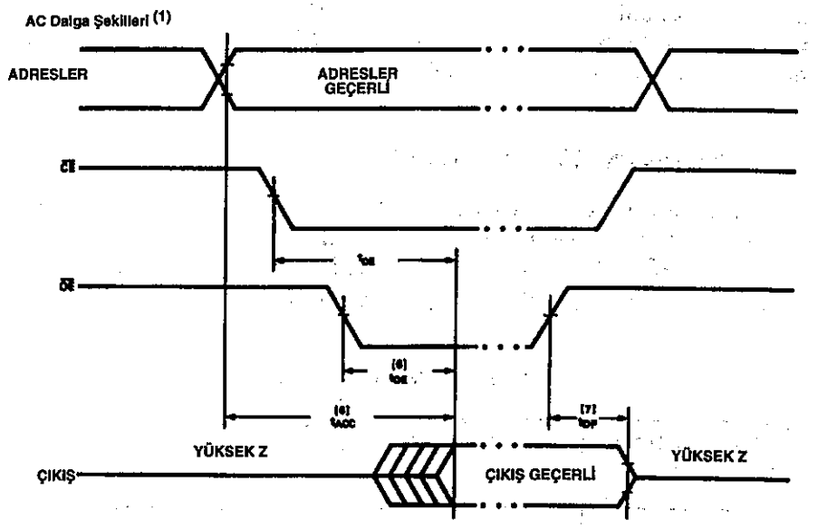
Sembol	Parametre	Tipik	Max.	Ünit	Şartlar
C_{IN}	Giriş kondansatörü	4	6	pF	$V_{IN} = 0V$
C_{OUT}	Çıkış kondansatörü	8	12	pF	$V_{OUT} = 0V$

AC Test Şartları

Çıkış Yükü: TTL kapı ve C_L = 100 pF
Giriş yükselme ve düşme zamanları: ≤ 20ns
Giriş darbe seviyeleri: 0.8V —22V
Zaman Ölçümleri Referans Seviyesi:

Girişler: 1V ve 2V
Çıkışlar: 0.8V ve 2V

AC Dalga Şekilleri (1)



Not:

1.V_{CC} , V_{PP} ile aynı anda veya daha önce uygulanmalı ve V_{PP} ile aynı anda veya daha sonra ayrılmalıdır.

2.V_{PP} , programlama süresi dışında doğrudan V_{CC1} 'e bağlanabilir. Bu durumda kaynak akımı I_{CC} ve I_{PP}1'in toplamı olmalıdır.

3.0.6 Voltluk tolerans, V_{PP} kaynağı bacağını okuma sırasındaki V_{CC}'den programlama için 25 Volta anahtarlayan bir sürücü devresi kullanılmasına izin verir.

4.Tipik değerler TA = 25 °C ve nominal kaynak gerilimleri içindir.

5.Bu parametre yalnızca örneklendi ve %100 test edilmedi.

6.$\overline{CE}$ 'nin düşme kenarından sona $\overline{OE}$, t_{ACC} üzerinde bir etki yapmadan $t_{ACC} - t_{OE}$ 'ye kadar geciktirilebilir.

7.tDF önce meydana gelen $\overline{OE}$ veya $\overline{CE}$ 'den belirtilir.

SİLME KARAKTERİSTİKLERİ

2716 'nın silinmesi yaklaşık 4000 Angstrom'dan (*Å*) daha kısa dalgaboylu ışıklara maruz kalmakla olur. *Güneş ışığı ve belli tipteki florasan lambalar* 3000-4000 *Å* aralığında dalgaboylarına sahiptir. Veriler, florasanla oda aydınlatma seviyesindeki ışık altında sürekli olarak bırakılan tipik bir 2716'nın silinmesinin yaklaşık 3 yıl alacağını gösterir, direk güneş ışığı ise bir 2716'nın yaklaşık 1 haftada silinmesine sebep olur. Eğer 2716 bu aydınlatma şartları altında uzun süre kalacaksa, istenmiyen silinmeleri önlemek için 2716 penceresinin üzerine yerleştirilmek üzere İntel, opak etiketler üretmektedir.

2716 için tavsiye edilen silme yöntemi 2537 *Angströmluk (Å)* kısadalga morötesi ışığa tabi tutmaktır. Silme için birleştirilmiş doz (morötesi ışık şiddeti X maruz kalma zamanı) en azından 15 W-sn/cm^2 olmalıdır. 12000 $\mu W/cm^2$ güç değerli bir morötesi lambayı bu dozda kullanarak silme yaklaşık 15 ile 20 dakika tutar. Silme sırasında 2716 1 inçlik lamba tüpleri içine yerleştirilmelidir. Bazı lambaların tüplerinde filtreler vardır ve bu filtreler silmeye başlamadan önce çıkarılmalıdır.

2716'NIN ÇALIŞMASI

2716'nın beş çalışma modu Tablo I'de gösterilmiştir. Beş modda da bütün girişlerin TTL seviyelerinde olmasına dikkat edilmelidir. Gerekli güç kaynakları +5V, V_{cc} ve V_{pp} 'dir. V_{pp} güç kaynağı üç programlama modu sırasında 25 V, ve diğer iki mod sırasında 5V olmalıdır.

Tablo I : MOD SEÇİMİ

BACAKLAR / MOD	$\overline{CE}$/PGM (18)	$\overline{OE}$ (20)	Vpp (21)	Vcc (24)	ÇIKIŞLAR (9-11. 13-17)
Okuma	V_{IL}	V_{IL}	+5	+5	D_{OUT}
Bekleme	V_{IH}	Önemsiz	+5	+5	YÜKSEK Z
Program	V_{IL}'den V_{IH}'ye darbe	V_{IH}	+25	+5	D_{IN}
Program Bekleme	V_{IL}	V_{IL}	+25	+5	D_{OUT}
Program Tutma	V_{IL}	V_{IH}	+25	+5	YÜKSEK Z

OKUMA MODU

Verileri çıkıştan alabilmek için 276 'nın mantıksal olarak sağlaması gereken iki kontrol fonksiyonu vardır. Yonga Yetkilendirme ($\overline{CE}$) güç kontroludur ve eleman seçiminde kullanılmalıdır. Çıkış Yetkilendirme ($\overline{OE}$) çıkış kontroludur ve eleman seçiminden bağımsız olarak verilerin çıkış bacaklarına yönlendirilmesinde kullanılmalıdır. Adreslerin kararlı olduğu varsayılarak, adres erişim zamanı (t_{ACC}) $\overline{CE}$'den çıkışa olan gecikmeye eşittir (t_{CE}). $\overline{CE}$'nin alçak olduğu ve adreslerin en azından t_{ACC}-t_{OE} süreli için kararlı olduğu varsayılırsa, veri $\overline{OE}$'nin düşme kenarından 120 *ns* (t_{OE}) sonra çıkışta mevcuttur.

BEKLEME MODU

2716 aktif güç harcamasını 525 mW'tan 132 mW'a, % 75 düşüren bekleme moduna sahiptir. $\overline{CE}$ girişine bir TTL yüksek sinyali uygulayarak 2716 bekleme moduna sokulur. Bekleme modunda, $\overline{OE}$ girişinden bağımsız olarak çıkışlar yüksek empedans durumundadır.

ÇIKIŞ SEÇİLMEMESİ

İki ya da daha fazla 2716'nın çıkışları aynı veri yolu üzerinde birlikte VEYAlanmış olarak bağlanmış olabilir. Bu şekildeki bağlantıda 2716 lar arasında veri yolu yarışını önlemek için yalnızca bir 2716 'nın çıkışı seçilmelidir *($\overline{OE}$ alçak)*. Diğer 2716 ların çıkışları *OE* girişini yüksek TTL seviyesine çekerek seçilmemelidir.

PROGRAMLAMA

Başlangıçta ve her silişten sonra 2716 'nın bütün bitleri "1" durumundadır. Veri istenilen bit yerlerini 0 programlayarak sunulur. Yalnızca 0 'lar programlanmasına rağmen bir veri kelimesinde hem 1 hem de 0'lar gösterilebilir. Bir 0'ı 1'e çevirmenin tek yolu morötesi ışıkla silmektir. V_{pp} güç kaynağı 25V ve $\overline{OE}$, V_{IH} durumundayken 2716 programlama modundadır. Programlanacak veri paralel 8 bit olarak veri çıkışı bacaklarına uygulanır. Adres ve veri girişleri için gerekli seviyeler TTL seviyeleridir.

Adres ve veri kararlı olduğunda 50 m saniyelik, aktif yüksek TTL programlama darbesi $\overline{CE}$/*PGM* girişine uygulanır. Program darbesi programlanacak her adres yerine uygulanmalıdır. Herhangi bir yeri, herhangi bir zamanda, tek tek, sıralı ya da rasgele olarak programlayabilirsiniz. Program

darbesinin maksimum genişliği 55 msaniyedir. 2716, $\overline{CE}$/*PGM* girişine uygulanacak bir DC sinyalle programlanmamalıdır.

Birçok 2716 'nın aynı veri ile paralel olarak programlanması, programlama gereksinimlerinin basitliği dolayısıyla kolaylıkla yapılabilir. Paralel bağlanmış 2716 'ların benzer girişleri aynı veri ile programlanacakları zaman beraber bağlanırlar. $\overline{CE}$/*PGM* girişlerine uygulanan yüksek seviyeli TTL darbesi paralel bağlı 2716 'ları programlar.

PROGRAM TUTMA

Paralel bağlanmış birçok 2716 'nın farklı veri ile programlanmasıda yapılabilir. Paralel 2716 'ların $\overline{CE}$/*PGM* dışındaki bütün benzeri girişleri ($\overline{OE}$ dahil) ortak olabilir. 2716 nın $\overline{CE}$/*PGM* girişine V_{pp} = 25 V iken uygulanan bir TTL seviyesi program darbesi 2716 yı programlar. $\overline{CE}$/PGM girişindeki alçak seviye diğer 2716 'ların programlanmasını engeller.

PROGRAM ONAYLAMA

Programlanmış bitlerin doğru olarak programlanıp programlanmadığını belirlemek için bir onaylama işlemi yapılmalıdır. Onaylama V_{pp}=25 V iken yapılabilir. Programlama ve program onaylama dışında V_{pp} 5V olmalıdır.

ZILOG

Zilog Z 80 üretim hattı, mikrobilgisayar parçaları, geliştirme sistemleri ve destekleme yazılımının olduğu tam bir settir. Z 80 mikrobilgisayar parça seti yüksek-performanslı mikrobilgisayar sistemlerini başka mantık elemanlarına gerek kalmadan ve *en az sayıda ucuz* ve *standart bellek elemanları kullanarak imal etmek için gerekli bütün devreleri kapsamaktadır.*

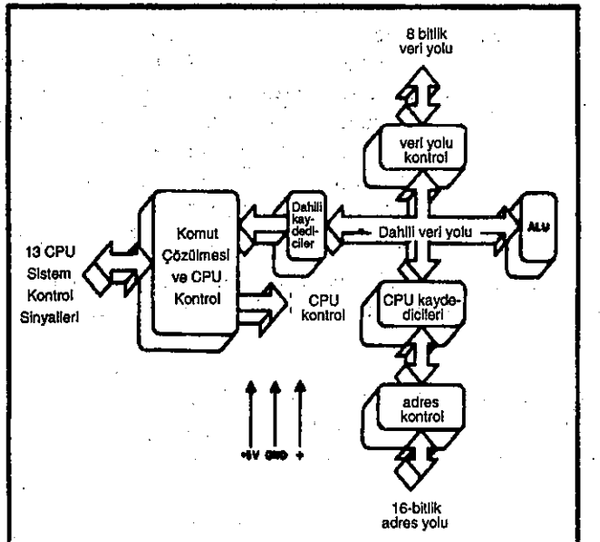
Z 80 ve Z 80A CPU ları rakipsiz hesaplama gücüne sahip *üçüncü nesil tek yongalı mikroişlemcilerdir.* Hesaplama gücündeki artışlar ikinci nesil mikroişlemcilerle karşılaştırıldıklarında daha yüksek sistem performansına ve bellek kullanımının daha verimli olmasına yol açar. İlave olarak tek gerilim gereksinimlerinden dolayı ve bütün çıkış sinyalleri standart bellek veya çevrebirim devrelerini kontrol etmek için tamamen kodlanabildiği ve zamanlanabildiği için, Z 80 ve Z 80A CPU ları sistemlerin içine kolaylıkla adapte edilebilir. Devre *N - kanallı*, iyon çoğaltılmış, silikon kapılı MOS süreci kullanarak gerçekleştirilmiştir.

Şekil 1, CPU 'nun bir blok diyagramıdır ve Şekil 2 programcının erişebileceği 208 bitlik okuma / yazma belleğini içeren dahili kaydedici yapısını göstermektedir. *8-bitlik tek kaydediciler* veya *16-bitlik kaydedici* çifti olarak kullanılabilen altı genel amaçlı kaydedicilerden iki set bulunmaktadır. Programcı bir grup değiştirme deyimini kullanarak ana ya da alternatif kaydediciler setinden birine erişebilir. Alternatif set işlemlerin önplan/arka plan modunda yapılmasına izin verir ya da çok hızlı kesme cevabı için kullanılabilir. Her CPU, çok seviyeli kesmelere, sınırsız içiçe yordamlara ve birçok veri tipi işlemesinin basitleşmesine izin veren *16-bitlik yığın göstericiye sahiptir.*

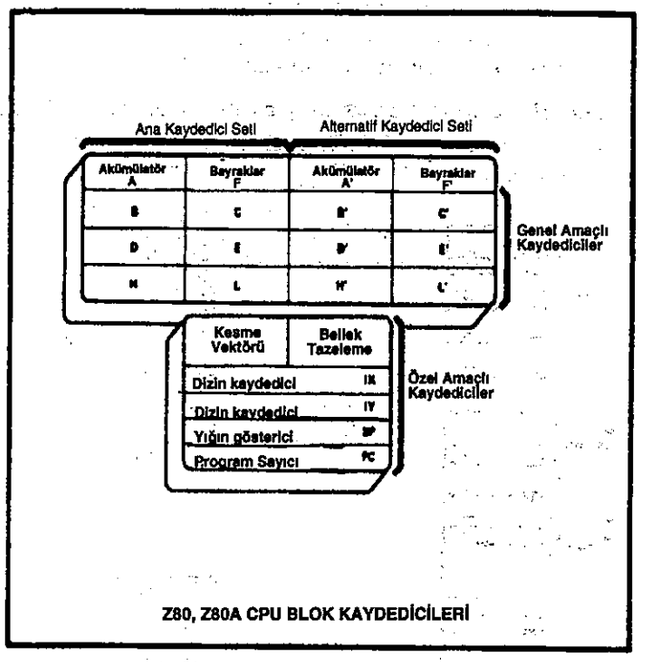
İki 16-bitlik dizin kaydedici, tablolu veri işlemeye ve yeniden yerleştirilebilir kodun kolay gerçekleştirilmesine imkan verir. Tazeleme kaydedici, harici dinamik belleklerin otomatik ve tamamen şeffaf olarak tazelenmesini sağlar. I kaydedici, bir güçlü kesme cevap modunda bir kesme servisi adres tablosunu gösteren göstericinin üst 8-bitini oluşturmak için kullanılır, göstericinin alt 8-biti kesmeyi yapan eleman tarafından sağlanır. Daha sonra bu servis adresine bir dolaylı çağırma yapılır.

ÖZELLİKLER

- Tek yonga, *N*-kanal silikon kapı *CPU.*
- 158 komut-tam yazılım uyumluluğu için 8080A'nın 78 komutunun hepsini içerir. Yeni komutlar, dizinli, bit ve göreceli gibi daha faydalı adresleme modları ile 4-, 8- ve 16- bitlik işlemleri içerir.
- 17 dahili kaydedici
- Hızlı kesme cevabının üç modu ve bir maskelenemez kesme.
- Hemen hemen hiç harici devre kullanmadan *standart hızlı statik* ve *dinamik belleklerle* doğrudan *bağlanabilme.*
- Deyim yerine getirme *hızı* 1.0 µs.
- Tek 5V DC kaynak ve *tek-fazlı* 5 Volt saat.
- 4-, 8- veya 16-bitlik işlemlerde diğer *tek yonga* mikrobilgisayarlardan daha iyidir.
- Bütün bacaklar *TTL* uyumlu
- İçinde dinamik *RAM* tazeleme devresi bulunmaktadır.



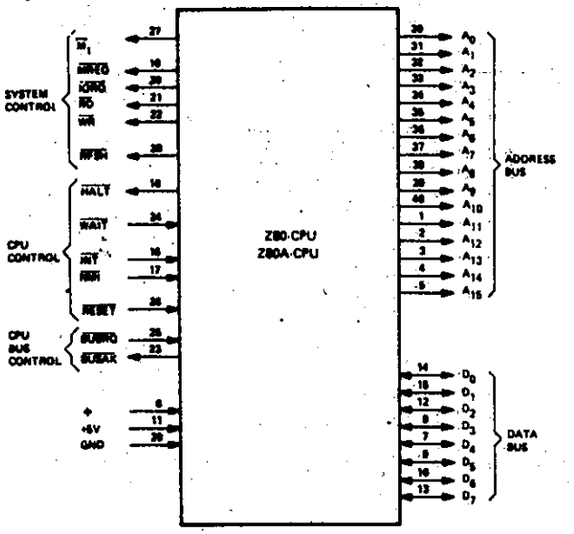
Z80, Z80A CPU BLOK DİYAGRAMI



Z80, Z80A CPU BLOK KAYDEDİCİLERİ

A_0 - A_{15} *(Adres yolu)*: Üç - durumlu çıkış, aktif yüksek. A_0-A_{15}, 16-bitlik adres yolunu oluşturur. Adres yolu, bellek (64 Kbayta kadar) veri değişmeleri ve I/O elemanları veri değişmeleri için adresleri sağlar.

D_0-D_7 *(Veri yolu)*: Üç-durumlu giriş/çıkış, aktif yüksek. D_0-D_7, 8-bitlik iki yönlü veri yolunu oluşturur. Veri yolu, bellek ve *I/O* elemanları ile veri değişmeleri için kullanılır.



Z80 - Z80A CPU Bacak yapısı

$\overline{M_1}$ *(Makine çevrimi bir)*: Çıkış, aktif alçak. $\overline{M_1}$ şu andaki makine çevriminin, bir komut yerine getirmenin işlem kodunu bellekten getirme çevrimi olduğunu belirtir.

$\overline{MREQ}$ *(Bellek isteği)*: Üç-durumlu çıkış, aktif alçak. *Bellek istek sinyali*, adres taşıtının bir *bellek okuma* veya *bellek yazma* işlemi için geçerli bir adres tuttuğunu belirtir.

$\overline{IORQ}$ *(Giriş/Çıkış isteği)*: Üç-durumlu çıkış, aktif alçak. $\overline{IORQ}$ sinyali, adres yolunun alt yarı kısmının *I/O* okuma veya yazma işlemi için bir geçerli *I/O* adresi tuttuğunu belirtir. Bir $\overline{IORQ}$ sinyali, bir kesme cevabı vektörünün veri yoluna yerleştirilebileceğini belirten *bir kesme alındığı* ve *onaylandığı zamanda üretilebilir.*

$\overline{RD}$ *(Bellek Okuma)* : Üç-durumlu çıkış, aktif alçak. $\overline{RD}$, CPU 'nun bellekten veya bir *I/O* eleman veri istediğini belirtir. Adreslenen *I/O* elemanı veya bellek bu sinyali *CPU* veri yoluna göndermek için kullanır.

$\overline{WR}$ *(Belleğe yazma)* : Üç-durumlu çıkış, *aktif alçak.* $\overline{WR}$, *CPU* veri yolunun adreslenen bellek ya da *I/O* elemanına depolanacak geçerli veri tuttuğunu belirtir.

$\overline{RFSH}$ *(Tazeleme)* : Çıkış, aktif alçak. $\overline{RFSH}$, adres yolunun alçak 7 bitinin dinamik bellekler için tazeleme adresini içerdiğini belirtir ve şu andaki $\overline{MREQ}$ sinyali bütün dinamik bellekleri *tazeleme okuması yapmak* için kullanmalıdır.

$\overline{HALT}$ *(Tutma durumu)* : Çıkış, aktif alçak. $\overline{HALT}$, CPU'nun bir yazılım *HALT* komutunu yerine getirdiğini ve işlem yeniden başlamadan önce bir maskelenemez ya da maskelenebilir (maske yetkilendirilmiş halde) kesme beklediğini belirtir. Tutma durumunda, CPU bellek tazeleme işlemini sürdürmek için *NOP* komutlarını yerine getirir.

$\overline{WAIT}$ *(Bekle):* Giriş, aktif alçak. $\overline{WAIT}$, *Z 80 - CPU*'ya adreslenen belleğin yada *I/O*elemanının veri transferi için hazır olmadığını belirtir. Bu sinyal aktif olduğu sürece *CPU bekleme durumuna* girmeye devam eder.

$\overline{INT}$ *(Kesme isteği)* : Giriş, *aktif alçak.* Kesme isteği sinyali I/O elemanları tarafından üretilir. Dahili yazılım kontrollu *kesme yetkilendirme* flip-flopu (IFF) yetkilendirilmişse, o andaki komutun bitiminde isteğe cevap verilir.

$\overline{NMI}$ *(Maskelenemeyen kesme)* : Giriş, *aktif alçak.* Maskelenemeyen kesme istek hattı $\overline{INT}$'den daha yüksek önceliğe sahiptir ve kesme yetkilendirme flip-flopunun statüsüne bakılmaksızın o andaki komutun bitiminde her zaman isteğe cevap verilir. $\overline{NMI}$, Z 80-CPU'yu 0066H adresinden *yeniden başlamaya zorlar.*

$\overline{RESET}$: Giriş, *aktif alçak.* $\overline{RESET}$, CPU'yu şu şekilde ilk kullanıma hazırlar. Kesme yetkilendirme flip-flopu resetlenir, program sayıcı, *I* ve *R* kaydedicileri sıfırlanır, kesme 8080A moduna ayarlanır. Reset zamanı süresinde, *adres* ve *veri yolları yüksek empedans durumuna gider* ve bütün kontrol çıkış sinyalleri *aktif olmayan duruma gider.*

$\overline{BUSRQ}$ *(Yol isteği)* : Giriş, *aktif alçak.* Yol istek sinyali *NMI* 'dan daha yüksek önceliğe sahiptir ve her zaman şu andaki makine çevriminin bitiminde yerine getirilir ve diğer elemanların kontrol edebilmeleri için CPU adres yolu, veri yolu ve üç-durumlu çıkış kontrol sinyallerinin bir *yüksek empedans durumuna gitmelerini ister.*

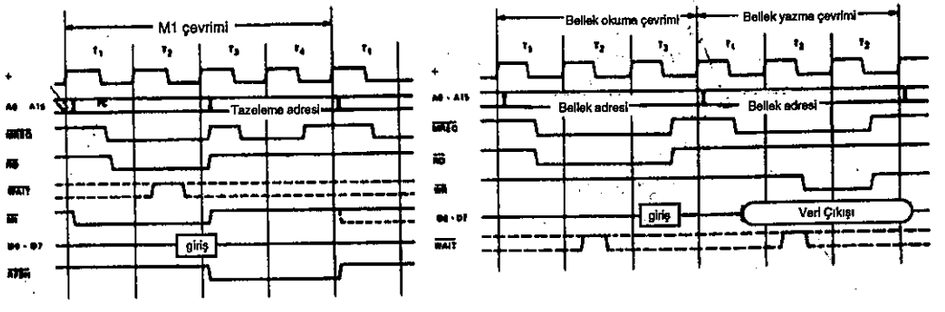
$\overline{BUSAK}$ *(Yol onaylama)* : Çıkış, aktif alçak. *Yol onaylama,* isteği yapan elemana CPU adres yolu, veriyolu ve üç-durumlu çıkış kontrol sinyallerinin kendi *yüksek empedans* durumlarına ayarlandığını ve *harici elemanların* bu *sinyalleri kontrol edebileceğini* belirtir.

ZAMANLAMA DALGAŞEKİLLERİ KOMUT İŞLEME KODUNUN GETİRİLMESİ

Program sayıcısı içeriği (PC) çevrimin hemen başında adres yoluna yerleştirilir. Yarım saat zamanı sonra $\overline{MREQ}$ aktif olur. $\overline{MREQ}$ 'nun düşen kenarı dinamik bellekleri yonga yetkilendirme olarak doğrudan kullanılabilir. $\overline{RD}$ aktif iken bellek verisinin *CPU* veri yoluna yetkilendirilmesini belirtir. *CPU* veriyi saat durumu T_3'ün yükselen kenarında örnekler. *CPU* dahili olarak kodu çözerken ve komutu yerine getirirken, getirme çevriminin T_3 ve T_4 saat durumları dinamik bellekleri tazelemek için kullanılır. Tazeleme kontrol sinyali $\overline{RFSH}$, bütün dinamik bellekleri tazeleme okuma işleminin yapılması gerektiğini belirtir.

BELLEK OKUMA VEYA YAZMA ÇEVRİMLERİ

Burada bellek okuma veya yazma çevrimlerinin işleme kodunun getirilmesi *(M_1 çevrimi)* dışındaki zamanlaması gösterilmiştir. $\overline{MREQ}$ ve $\overline{RD}$ sinyalleri, getirme çevriminin tamamen aynısı olarak kullanılır. Belleğe yazma çevriminde, adres yolu kararlı olduğunda $\overline{MREQ}$ aktif olur böylece dinamik bellekler için yonga yetkilendirme olarak doğrudan kullanılabilir. Veri yolundaki veri kararlı olduğunda $\overline{WR}$ hattı aktif olur, böylece hemen hemen bütün yarı iletken belleklere *R/W* darbesi olarak kullanılabilir.

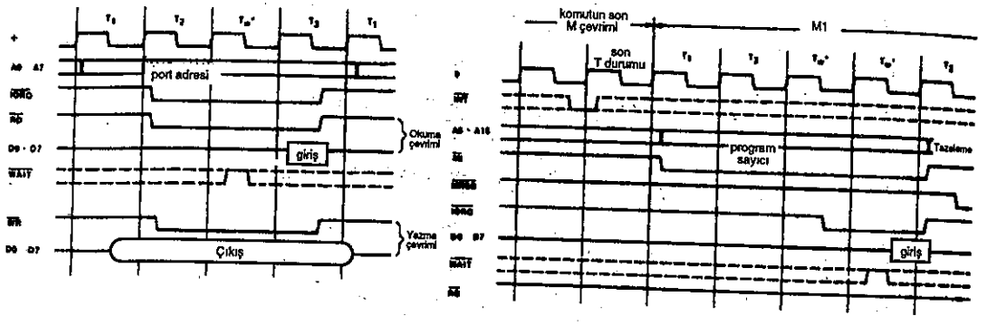


GİRİŞ VEYA ÇIKIŞ ÇEVRİMLERİ

Burada bir *I/O okuma* veya *I/O yazma işleminin zamanlamasını gösterilmektedir.* I/O işlemleri sırasında bir bekleme durumunun otomatik olarak sokulduğuna (T_w*) dikkat ediniz. Bu fazladan zaman, I/O işlemi sırasında I/O portunun adresin kodunu çözmesine ve bir bekleme gerekiyorsa $\overline{WAIT}$ hattının aktif duruma gelmesine sebep olur.

KESME İSTEĞİ / ALINDI ÇEVRİMİ

Kesme sinyali CPU tarafından herhangi bir deyimin sonundaki son saat darbesinin yükselen kenarıyla örneklenir. Bir kesme kabul edildiğinde özel bir M_1 çevrimi üretilir. M_1 çevrimi sırasında $\overline{IORQ}$ sinyali aktif olur ($\overline{MREQ}$ yerine). Bu, kesmeyi yapan eleman 8-bitlik vektörü, veri yoluna yerleştirebileceğini belirtir. İki bekleme durumu (T_w*), bu çevrime otomatik olarak eklenir, böylece Z 80 çevre birim kontrolörlerinde kullanılan benzer salınma önceliği kesme planı kolaylıkla uygulanabilir.



Z 80, Z 80A KOMUT SETİ

Aşağıda verilen özet, *Z 80, Z 80A* komut setini, çevirici dil komut kısaltması ve deyim tarafından yerine getirilen *sembolik işlemle birlikte gösterir.* Daha detaylı bir liste Z 80 - CPU teknik el kitabında ve çevirici dil programlama el kitabında bulunabilir. Komutlar aşağıdaki kategorilere bölünmüştür :

8- bitlik yüklemeler
16-bitlik yüklemeler
Değişmeler
Bellek bloğu taşımaları
Bellek bloğu aramaları
8 - bitlik aritmetik ve mantık
16-bitlik aritmetik
Genel amaçlı akümülatör & bayrak işlemleri
Karışık grup
Dönmeler ve kaymalar
Bit Set, Reset ve Test
Giriş ve Çıkış
Atlamalar
Çağırmalar
Yeniden başlamalar
Dönüşler

Tabloda aşağıdaki terminoloji kullanılmıştır.

b ≡ Herhangi bir 8-bitlik kaydedici veya bellek yerinde bit numarası

cc ≡ Bayrak koşul kodu

- NZ ≡ Sıfır değil
- Z ≡ Sıfır
- NC ≡ Elde değil
- C ≡ Elde
- PO ≡ Tek eşlik veya taşma yok
- PE ≡ Çift eşlik veya taşma
- P ≡ Pozitif
- M ≡ Negatif (eksi)

d ≡ Herhangi bir 8-bitlik hedef kaydedicisi veya bellek yeri

dd ≡ Herhangi bir 16-bitlik hedef kaydedicisi veya bellek yeri

e ≡ Bağıl atlamalar ve dizinli adreslemede kullanılan, 8-bitlik 2'nin tümleyeni uzaklık

L ≡ Sayfa sıfırda 8 özel çağırma yeri. Ondalık gösterimde bunlar 0, 8, 16, 24, 32, 40, 48 ve 56 'dır.

n ≡ 8-bitlik ikilik sayı

nn ≡ 16-bitlik ikilik sayı

r ≡ Herhangi bir 8-bitlik genel amaçlı kaydedici (A, B, C, D, E, H veya L)

s ≡ Bir 8-bitlik kaynak kaydedicisi veya bellek yeri

s_b ≡ Belirli bir 8-bitlik kaydedici veya bellek yerinde bir bit

ss ≡ 16-bitlik kaynak kaydedicisi veya bellek yeri

alt indis "L" ≡ 16-bitlik kaydedicisinin alçak 8-biti

alt indis "H" ≡ 16-bitlik kaydedicisinin üst 8-biti

( ) ≡ ( ) içindeki içerik bir bellek yeri ya da I/O port sayısı için bir gösterici olarak kullanılır.

8-bitlik kaydediciler : A, B, C, D, E, H, L, I ve R 'dir.

16-bitlik kaydedici çiftleri AF, BC, DE ve HL'dir.

16-bitlik kaydediciler SP, PC, IX ve IY 'dir

Gerçekleştirilen adresleme modları aşağıdakilerin kombinasyonlarını içerir.

Doğrudan İndisli

Doğrudan genişletilmiş
Kaydedici

Değiştirilmiş sayfa sıfır
Örtük

Bağıl Kaydedici dolaylı

Genişletilmiş Bit

8-bitlik yüklemeler

Kısa Komut	Sembolik İşlem	Açıklamalar
LD r, s	r ← s	s ≡ r, n, (HL), (IX+e), (IY+e)
LD d, r	d ← r	d ≡ (HL), r (IX+e), (IY+e)
LD d, n	d ← n	d ≡ (HL), (IX+e), (IY+e)
LD A, s	A ← s	s ≡ (BC), (DE), (nn), I, R
LD d, A	d ← A	d ≡ (BC), (DE), (nn), I, R

16-bitlik yüklemeler

Kısa Komut	Sembolik İşlem	Açıklamalar
LD dd, nn	dd ← nn	dd ≡ BC, DE, HL, SP, IX, IY
LD dd, (nn)	dd ← (nn)	dd ≡ BC, DE, HL, SP, IX, IY
LD (nn), ss	(nn) ← ss	ss ≡ BC, DE, HL, SP, IX, IY
LD SP, ss	SP ← ss	ss = HL, IX, IY
PUSH ss	$(SP-1) \leftarrow ss_H$; $(SP-2) \leftarrow ss_L$	ss = BC, DE, HL, AF, IX, IY
POP dd	$dd_L \leftarrow (SP)$; $dd_H \leftarrow (SP+1)$	dd = BC, DE, HL, AF, IX, IY

Değişimler

Kısa Komut	Sembolik İşlem	Açıklamalar
EX DE, HL	DE ↔ HL	
EX AF, AF'	AF ↔ AF'	
EXX	$\begin{pmatrix}BC\\DE\\HL\end{pmatrix} \leftrightarrow \begin{pmatrix}BC'\\DE'\\HL'\end{pmatrix}$	
EX (SP), ss	$(SP) \cdots ss_L$, $(SP+1) \cdots ss_H$	ss ≡ HL, IX, IY

Bellek bloğu taşımaları

Kısa Komut	Sembolik İşlem	Açıklamalar
LDI	(DE) ← (HL), DE ← DE+1 HL ← HL+1, BC ← BC-1	
LDIR	(DE) ← (HL), DE ← DE+1 HL ← HL+1, BC ← BC-1 Repeat until BC = 0	
LDD	(DE) ← (HL), DE ← DE-1 HL ← HL-1, BC ← BC-1	
LDDR	(DE) ← (HL), DE ← DE-1 HL ← HL-1, BC ← BC-1 Repeat until BC = 0	

Bellek bloğu aramaları

Kısa Komut	Sembolik İşlem	Açıklamalar
CPI	A-(HL), HL ← HL+1 BC ← BC-1	A-(HL) yalnızca bayrakları set eder. A etkilenmez.
CPIR	A-(HL), HL ← HL+1 BC ← BC-1, Repeat until BC = 0 or A = (HL)	
CPD	A-(HL), HL ← HL-1 BC ← BC-1	
CPDR	A-(HL), HL ← HL-1 BC ← BC-1, Repeat until BC = 0 or A = (HL)	

8-bitlik yüklemeler

Kısa Komut	Sembolik İşlem	Açıklamalar
ADD s	A ← A + s	CY elde bayrağıdır.
ADC s	A ← A + s + CY	
SUB s	A ← A - s	
SBC s	A ← A - s - CY	s ≡ r, n, (HL) (IX+e), (IY+e)
AND s	A ← A ∧ s	
OR s	A ← A ∨ s	
XOR s	A ← A ⊕ s	

8-bitlik ALU

Kısa Komut	Sembolik İşlem	Açıklamalar
CP s	A - s	s ≡ r, n (HL) (IX+e), (IY+e)
INC d	d ← d + 1	d ≡ r, (HL) (IX+e), (IY+e)
DEC d	d ← d - 1	

16-bitlik aritmetik

Kısa Komut	Sembolik İşlem	Açıklamalar
ADD HL, ss	HL ← HL + ss	ss ≡ BC, DE HL, SP
ADC HL, ss	HL ← HL + ss + CY	
SBC HL, ss	HL ← HL - ss - CY	
ADD IX, ss	IX ← IX + ss	ss ≡ BC, DE, IX, SP
ADD IY, ss	IY ← IY + ss	ss ≡ BC, DE, IY, SP
INC dd	dd ← dd + 1	dd ≡ BC, DE, HL, SP, IX, IY
DEC dd	dd ← dd - 1	dd ≡ BC, DE, HL, SP, IX, IY

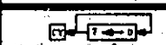
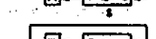
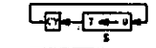
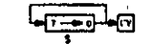
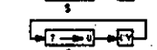
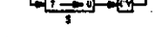
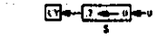
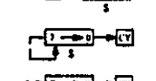
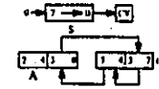
Akümülatör ve Bayrak

Kısa Komut	Sembolik İşlem	Açıklamalar
DAA	Toplama veya çıkarmadan sonra A içeriğini paketli BCD'ye çevirir.	İşlenenler paketli BCD şeklinde olmalıdır.
CPL	$A \leftarrow \overline{A}$	
NEG	A ← 00 - A	
CCF	$CY \leftarrow \overline{CY}$	
SCF	CY ← 1	

Karışık

Kısa Komut	Sembolik İşlem	Açıklamalar
NOP	İşlem yok	
HALT	CPU'yu durdur	
DI	Kesmeleri yetkisiz yap	
EI	Kesmeleri yetkilendir	
IM 0	Kesme modu 0	8080A modu 0038H'ye çağırma
IM 1	Kesme modu 1	
IM 2	Kesme modu 2	Dolaylı çağırma

Dönmeler ve Kaymalar

Kısa Komut	Sembolik İşlem	Açıklamalar
RLC s		s ≡ r, (HL) (IX+e), (IY+e)
RL s		
RRC s		
RR s		
SLA s		
SRA s		
SRL s		
RLD		
RRD		

Bit İşaretleri

Kısa Komut	Sembolik İşlem	Açıklamalar
BIT b, s	$Z \leftarrow \overline{s_b}$	Z sıfır bayrağı
SET b, s	$s_b \leftarrow 1$	s ≡ r, (HL) (IX+e), (IY+e)
RES b, s	$s_b \leftarrow 0$	

Giriş ve Çıkış

Kısa Komut	Sembolik İşlem	Açıklamalar
IN A, (n)	A ← (n)	Bayrakları set eder.
IN r, (C)	r ← (C)	
INI	(HL) ← (C), HL ← HL + 1 B ← B - 1	
INIR	(HL) ← (C), HL ← HL + 1 B ← B - 1 B = 0 oluncaya kadar tekrarla	
IND	(HL) ← (C), HL ← HL - 1 B ← B - 1	
INDR	(HL) ← (C), HL ← HL - 1 B ← B - 1 B = 0 oluncaya kadar tekrarla	
OUT(n), A	(n) ← A	
OUT(C), r	(C) ← r	
OUTI	(C) ← (HL), HL ← HL + 1 B ← B - 1	
OTIR	(C) ← (HL), HL ← HL + 1 B ← B - 1 B = 0 oluncaya kadar tekrarla	
OUTD	(C) ← (HL), HL ← HL - 1 B ← B - 1	
OTDR	(C) ← (HL), HL ← HL - 1 B ← B - 1 B = 0 oluncaya kadar tekrarla	

Atlamalar

Kısa Komut	Sembolik İşlem	Açıklamalar
JP nn	PC ← nn	
JP cc, nn	cc doğru ise PC ← nn, aksi halde devam et	cc: NZ PO, Z PE, NC P, C M
JR e	PC ← PC + e	
JR kk, e	cc doğru ise PC ← PC + e aksi halde devam et	kk: NZ NC, Z C
JP (ss)	PC ← ss	ss = HL, IX, IY
DJNZ e	B ← B - 1, if B = 0 devam et, aksi halde PC ← PC+e	

Çağırmalar

Kısa Komut	Sembolik İşlem	Açıklamalar
CALL nn	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$, PC ← nn	
CALL cc, nn	cc şartı yanlış ile devam et, aksi halde CALL nn ile aynı	cc: NZ PO, Z PE, NC P, C M

Yeniden başlamalar

Kısa Komut	Sembolik İşlem	Açıklamalar
RST L	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$, $PC_H \leftarrow 0$ $PC_L \leftarrow L$	

Dönüşler

Kısa Komut	Sembolik İşlem	Açıklamalar
RET	$PC_L \leftarrow (SP)$, $PC_H \leftarrow (SP+1)$	
RET cc	cc şartı yanlış ise devam et, aksi halde RET ile aynı	cc: NZ PO, Z PE, NC P, C M
RETI	Kesmeden dönüş, RET ile aynı	
RETN	Maskelenemez kesmeden dönüş	

Aksi belirtilmedikçe $T_A = 0\ ^oC - 70\ ^oC$, $V_{CC} = +5V \pm\ \%5$

Sinyal	Sembol	Parametre	Min	Max	Birim	Test Şartları
ϕ	t_c	Saat periyodu	4	[12]	µsec	
	$t_{w(\phi H)}$	Saat darbe genişliği, saat yüksek	180	[E]	nsec	
	$t_{w(\phi H)}$	Saat darbe genişliği, saat alçak	180	2000	nsec	
	$t_{r,f}$	Saat yükselme ve düşme zamanı		30	nsec	
A0-15	t_D (AD)	Adres çıkış gecikmesi		145	nsec	CL= 50 pF
	t_F (AD)	Yüzere gecikme		110	nsec	
	t_{acm}	$\overline{MREQ}$ (bellek çevirim)'den önce kararlı adres	[1]		nsec	
	t_{aci}	$\overline{IORQ}$, $\overline{RD}$ veya $\overline{WR}$ (I/O çevrimi)'den önce kararlı adres	[2]		nsec	
	t_{ca}	$\overline{RD}$, $\overline{WR}$, $\overline{IORQ}$ veya $\overline{MREQ}$'dan kararlı adres	[3]		nsec nsec	
	t_{caf}	Yüzer durumda RD veya $\overline{WR}$'den kararlı adres	[4]			
D_{0-7}	t_D(D)	Veri çıkış gecikmesi		230	nsec	C_L = 50pF
	t_F(D)	Yazma çevrimi sırasında yüzere gecikme		90	nsec	
	t_S ϕ (D)	M1 çevrimi sırasında saatin yükselen kenarına veri ayarlama zamanı	50		nsec	
	t_{cdm}	M2 ile M5 sırasında saatin düşen kenarına veri ayarlama zamanı	60		nsec	
	t_{cdf}	$\overline{WR}$ (bellek çevrimi) öncesi kararlı veri	[5]		nsec	
	t_{cdf}	$\overline{WR}$ (I/O Çevrimi) öncesi kararlı veri	[6]		nsec	
	t_{cdf}	$\overline{WR}$'den kararlı veri	[7]		nsec	
	tH	Ayarlama zamanı için tutma zamanı	0		nsec	
$\overline{MREQ}$	t_{DL} ϕ (MR)	Saatin düşen kenarından $\overline{MREQ}$ gecikmesi, $\overline{MREQ}$ alçak		100	nsec	C_L = 50pF
	t_{DH} ϕ (MR)	Saatin yükselen kenarından $\overline{MREQ}$ gecikmesi yüksek		100	nsec	
	t_{DH} ϕ (MR)	Saatin düşen kenarından $\overline{MREQ}$ gecikmesi, $\overline{MREQ}$ alçak		100	nsec nsec	
	t_W $\phi(\overline{MRL})$	Darbe genişliği, $\overline{MREQ}$ alçak	[8]		nsec	
	t_W $\phi(\overline{MRH})$	Darba genişliği $\overline{MREQ}$ yüksek	[9]		nsec	

[12] $t_c = t_{w(\phi H)} + t_{w(\phi L)} + t_{r,f}$

[1] $t_{acm} = t_{w(\phi H)} + t_f - 75$
[2] $t_{acf} = t_c - 80$
[3] $t_{ca} = t_{w(\phi L)} + t_r - 40$
[4] $t_{caf} = t_{w(\phi H)} + t_r - 60$
[5] $t_{dcm} = t_c - 210$
[6] $t_{dci} = t_{w(\phi H)} + t_r - 210$
[7] $t_{cdf} = t_{w(\phi H)} + t_r - 80$

[8] $t_{w(MRL)} = t_c - 40$
[9] $t_{w(MRH)} = t_{w(\phi H)} + t_f - 30$

$\overline{IORQ}$	t_{DL} ϕ (IR)	Saatin yükselen kenarından $\overline{IROQ}$ gecikmesi, $\overline{IORQ}$ alçak		90	nsec	C_L = 50pF
	t_{DL} ϕ (IR)	Saatin düşen kenarından $\overline{IORQ}$ gecikmesi, $\overline{IORQ}$ alçak		110	nsec	
	t_{DH} ϕ (IR)	Saatin yükselen kenarından $\overline{IORQ}$ gecikmesi, $\overline{IORQ}$yüksek		100	nsec	
	t_{DH} ϕ (IR)	Saatin düşen kenarından $\overline{IORQ}$ gecikmesi, $\overline{IORQ}$ yüksek		110	nsec	
$\overline{RD}$	t_{DL} ϕ (IR)	Saatin yükselen kenarından $\overline{RD}$ gecikmesi, $\overline{RD}$ alçak		100	nsec	C_L = 50 pF
	t_{DL} ϕ (IR)	Saatin düşen kenarından $\overline{RD}$ gecikmesi, $\overline{RD}$ alçak		130	nsec	
	t_{DH} ϕ (IR)	Saatin yükselen kenarından $\overline{RD}$ gecikmesi, $\overline{RD}$ yüksek		100	nsec	
	t_{DH} ϕ (IR)	Saatin düşen kenarından $\overline{RD}$ gecikmesi, $\overline{RD}$ yüksek		110	nsec	
$\overline{WR}$	t_{DL} ϕ (IR)	Saatin yükselen kenarından $\overline{WR}$ gecikmesi, $\overline{WR}$ alçak		80	nsec	C_L = 50 pF
	t_{DL} ϕ (IR)	Saatin düşen kenarından $\overline{WR}$ gecikmesi, $\overline{WR}$ alçak		90	nsec	
	t_{DH} ϕ (IR)	Saatin düşen kenarından $\overline{WR}$ gecikmesi, $\overline{WR}$ yüksek		100	nsec	
	t_W (WRL)	Darbe genişliği, $\overline{WR}$ alçak	[10]		nsec	
$\overline{M1}$	t_{DL} (M1)	Saatin yükselen kenırından $\overline{M1}$ gecikmesi, $\overline{WR}$ alçak		130	nsec	C_L = 50 pF
	t_{DH} (M1)	Saatin yükselen kenarından $\overline{M1}$ gecikmesi, $\overline{M1}$ yüksek		130	nsec	
$\overline{RFSH}$	t_{DL} (RF)	Saatin yükselen kenarından $\overline{RFSH}$ gecikmesi, $\overline{RFSH}$ alçak		180	nsec	C_L = 50 pF
	t_{DH} (RF)	Saatin yükselen kenırından $\overline{RFSH}$ gecikmesi, $\overline{RFSH}$ yüksek		150	nsec	
$\overline{WAIT}$	t_S (WT)	Saatin düşen kanarından $\overline{WAIT}$ ayarılama zamanı	70		nsec	
$\overline{HALT}$	t_D (HT)	Saatin düşen kanırından $\overline{HALT}$ gecikme zamanı		300	nsec	C_L = 50 pF
$\overline{INT}$	t_S (IT)	Saatin yükselen kenarından $\overline{INT}$ ayarlama zamanı	80		nsec	
$\overline{NMI}$	t_W (NML)	Darba genişliği, $\overline{NMI}$ alçak	80		nsec	

[10] $t_W(\overline{WRL}) = t_c - 40$

$\overline{BUSRQ}$	t_s (BQ)	Saatin yükselen kenarına $\overline{BUSRQ}$ ayarlama zamanı	80		nsec	
$\overline{BUSAK}$	t_{DL} (BA)	Saatin yükselen kenarından $\overline{BUSAK}$ gecikmesi, $\overline{BUSAK}$ alçak		120	nsec	C_L = 50 pF
	t_{DH} (BA)	Saatin düşen kenarından $\overline{BUSAK}$ gecikmesi, $\overline{BUSAK}$ yüksek		110	nsec	
$\overline{RESET}$	t_s (RS)	Saatin yükselen kenarına $\overline{RESET}$ ayarlama zamanı	60		nsec	
	tF (C)	Yüzere gecikme ($\overline{MREQ}$, $\overline{IORQ}$ $\overline{RD}$ ve $\overline{WR}$)		100		
	t_{mr}	$\overline{IORQ}$ öncesi MI kenarlı (kesme onaylama)	[11]		nsec	[11] $t_{mr} = 2t_c + t_w$

NOTLAR:

A. $\overline{RD}$ aktif olduğu zaman veri CPU veri yoluna yetkilendirilmelidir. Kesm onaylaması sırasında *MI* ve IORQ aktif olduğunda veri yetkilendirilmel dir.

B. Bütün kontrol sinyalleri dahili olarak senkronizedir, böylece hepsi saat göre asenkron olabilir.

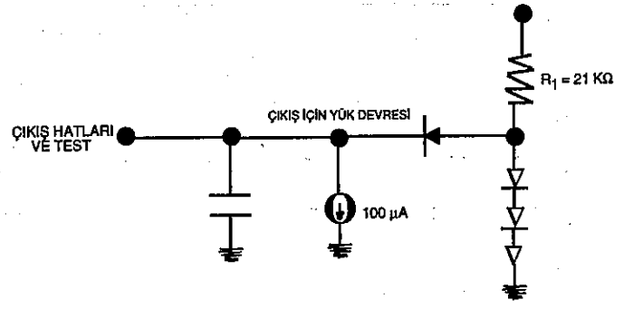
C. $\overline{RESET}$ sinyali en az 3 saat sinyali aktif olmalıdır.

D. Çıkış gecikmesi — Yüklenmiş kondansatör

$T_A = 70\,^{o}C$ $\qquad V_{cc} = +5\ V, \pm\%5$

Veri yolu için maksimum 200 pF ve adres ve kontrol hatları için 100 pF'a k dar yükteki her 50 pF artış için 10 nsaniye gecikme ilave et.

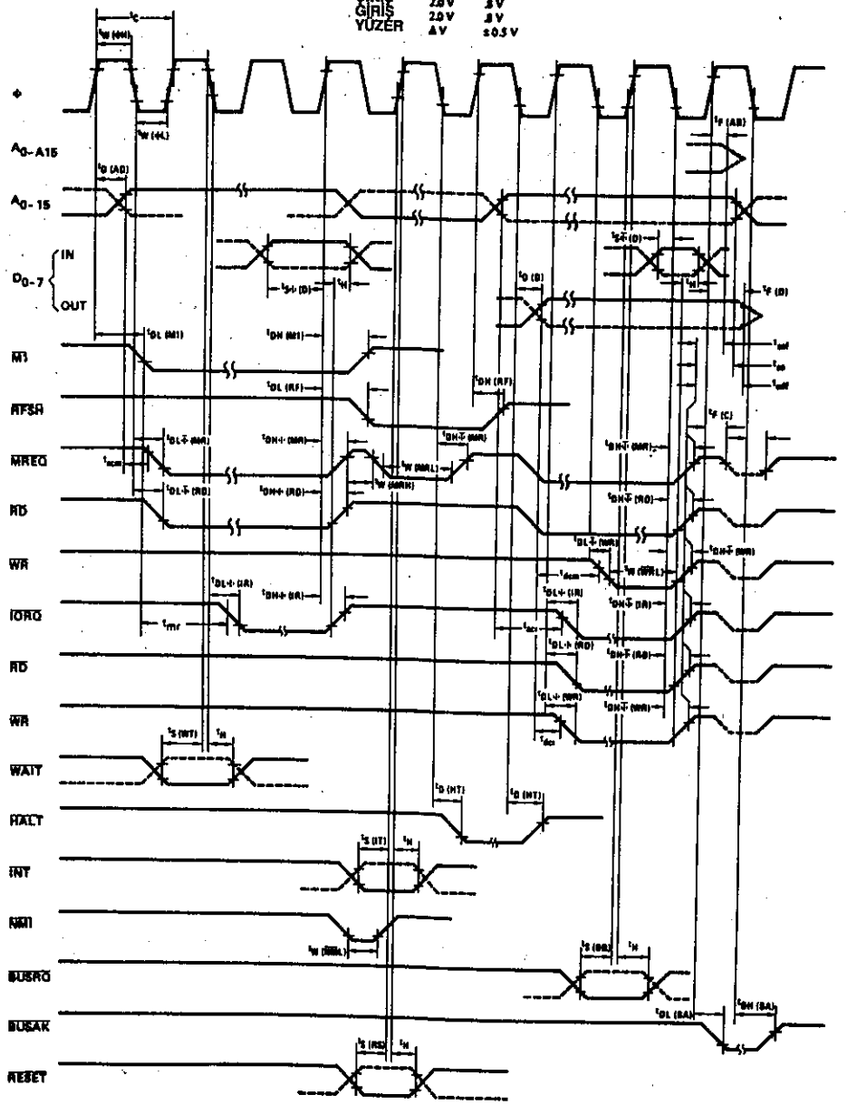
E. Tasarımda statik olmasına rağmen test t_W (ϕH)'nin maksimum 200 µsani olmasını garanti eder.



A.C. Zamanlama Diyagramı

Aksi belirtilmedikçe zaman ölçümleri takip eden gerilimlerde yapılmıştır.

	"1"	"0"
SAAT	V_{cc} -.6V	.45V
ÇIKIŞ	2.0 V	.8 V
GİRİŞ	2.0 V	.8 V
YÜZER	.4 V	±0.5 V



Mutlak Maksimum Değerler	Belirtilen çalışma aralığı
Başlama altında sıcaklık	
Depolama sıcaklığı	–65 °C, +150 °C
Herhangi bir bacakta toprağa göre gerilim	–0.3 V, +7V
Güç harcaması	1.5 W

*Uyarı:
Mutlak Maksimum Değerler altında listelenen değerlerin üzerinde çalışmaya zorlamak elemanın bozulmasına sebep olabilir. Bu yalnızca zorlama değeridir ve elemanın bu değerde fonksiyonel çalışması veya bu tarifnamenin çalışma bölümlerinde belirtilen değerlerin üstünde çalışması kastedilmez. Uzun süre olarak mutlak maksimum değerlere maruz bırakmak elemanın güvenilirliğini etkileyebilir.

Not: 780 CPU'nun bütün AC ve DC karakteristikleri I_{cc} dışında asker parçalar içinde aynı kalır. I_{cc} = 200 mA

Z80 CPU DC Karakteristikleri
Aksi belirtilmedikçe $T_A = 0^oC–70\ ^oC$, $V_{cc} = 5V, \pm\%5$

Sembol	Parametre	Min.	Typ.	Max.	Birim	Test Şartları
V_{ILC}	Saat girişi alçak gerilim	–0.3		0.45	V	
V_{IHC}	Saat girişi yüksek gerilim	V_{CC} – .6		V_{CC} + .3	V	
V_{IL}	Giriş alçak gerilim	–0.3		0.8	V	
V_{IH}	Giriş yüksek gerilim	2		V_{CC}	V	
V_{OL}	Çıkış alçak gerilim			0.4	V	t_{OL} = 1.8 mA
V_{OH}	Çıkış yüksek gerilim	2.4			V	t_{OH} = -250 µA
t_{CC}	Güç kaynağı akımı			150	mA	
t_{LI}	Giriş kaçak akımı			10	µA	V_{IN} = 0 to V_{CC}
t_{OH}	Yüzerde üç durumlu çıkış kaçak akımı			10	µA	V_{OUT} = 2.4 to V_{CC}
t_{LOL}	Yüzerde üç durumlu çıkış kaçak akımı			-10	µA	V_{OUT} = 0.44 V
t_{LD}	Giriş modunda veri yolu kaçak akımı			±10	µA	$0 \le V_{IN} \le V_{CC}$

Kondansatör T_A = 25 °C, F = 1 mHz
Ölçülmeyen Bacaklar Toprağa Bağlı

Sembol	Parametre	Max.	Birim
C_ϕ	Saat kondansatörü	35	pF
$C_{giriş}$	Giriş kondansatörü	5	pF
$C_{çıkış}$	Çıkış kondansatörü	10	pF

Z–80 CPU Ismarlama Bilgisi

C– Seramik
P– Plastik
S– Standard 5V ± %5, 0 °C – 70 °C
E– Genişletilmiş 5V ± %5, -40 °C +85 °C
M– Askeri 5V ± %10, – 55 °C ile 125 °C

Z80 A CPU DC Karakteristikleri
Aksi belirtilmedikçe $T_A = 0^oC–70\ ^oC$, $V_{cc} = 5V, \pm\%5$

Sembol	Parametre	Min.	Typ.	Max.	Birim	Test Şartları
V_{ILC}	Saat girişi alçak gerilim	–0.3		0.45	V	
V_{IHC}	Saat girişi yüksek gerilim	V_{CC} – .6		V_{CC} + .3	V	
V_{IL}	Giriş alçak gerilim	–0.3		0.8	V	
V_{IH}	Giriş yüksek gerilim	2.0		V_{CC}	V	
V_{OL}	Çıkış alçak gerilim			0.4	V	t_{OL} = 1.8 mA
V_{OH}	Çıkış yüksek gerilim	2.4			V	t_{OH} = -250 µA
t_{CC}	Güç kaynağı akımı			200	mA	
t_{LI}	Giriş kaçak akımı			10	µA	V_{IN} = 0 to V_{CC}
t_{OH}	Yüzerde üç durumlu çıkış kaçak akımı			10	µA	V_{OUT} = 2.4 to V_{CC}
t_{LOL}	Yüzerde üç durumlu çıkış kaçak akımı			-10	µA	V_{OUT} = 0.44 V
t_{LD}	Giriş modunda veri yolu kaçak akımı			±10	µA	$0 \le V_{IN} \le V_{CC}$

Kondansatör T_A = 25 °C, F = 1 mHz
Ölçülmeyen Bacaklar Toprağa Bağlı

Sembol	Parametre	Max.	Birim
C_ϕ	Saat kondansatörü	35	pF
$C_{giriş}$	Giriş kondansatörü	5	pF
$C_{çıkış}$	Çıkış kondansatörü	10	pF

Z80 A- CPU Ismarlama Bilgisi

C– Seramik
P– Plastik
S– Standard 5V ± %5, 0 °C – 70 °C

Sinyal	Sembol	Parametre	Min	Max	Birim	Test Şartları
	t_c	Saat periyodu	25	[12]	µsec	
ϕ	$t_{w(\phi H)}$	Saat darbe genişliği, saat yüksek	110	[E]	nsec	
	$t_{w(\phi H)}$	Saat darbe genişliği, saat alçak	110	2000	nsec	
	$t_{r,t}$	Saat yükselme ve düşme zamanı		30	nsec	
	t_D (AD)	Adres çıkış gecikmesi		110	nsec	
	t_F (AD)	Yüzere gecikme		90	nsec	
	t_{acm}	$\overline{MREQ}$ (bellek çevirim)'den önce kararlı adres	[1]		nsec	
A0-15	t_{aci}	$\overline{IORQ}$, $\overline{RD}$ veya $\overline{WR}$ (I/O çevrimi)'den önce kararlı adres	[2]		nsec	CL= 50 pF
	t_{ca}	$\overline{RD}$, $\overline{WR}$, $\overline{IORQ}$ veya $\overline{MREQ}$'dan kararlı adres	[3]		nsec nsec	
	t_{caf}	Yüzer durumda $\overline{RD}$ veya $\overline{WR}$'den kararlı adres	[4]			
	t_D(D)	Veri çıkış gecikmesi		150	nsec	
	t_F(D)	Yazma çevrimi sırasında yüzere gecikme		90	nsec	
	$t_S\phi$ (D)	M1 çevrimi sırasında saatin yükselen kenarına veri ayarlama zamanı	35		nsec	
D_{0-7}	$t_S\phi$ (D)	M2 ile M5 sırasında saatin düşen kenarına veri ayarlama zamanı	50		nsec	C_L = 50pF
	t_{cdm}	$\overline{WR}$ (bellek çevrimi) öncesi kararlı veri	[5]		nsec	
	t_{cdf}	$\overline{WR}$ (I/O Çevrimi) öncesi kararlı veri	[6]		nsec	
	t_{cdf}	$\overline{WR}$'den kararlı veri	[7]		nsec	
	tH	Ayarlama zamanı için tutma zamanı		0	nsec	
	$t_{DL}\phi$ (MR)	Saatin düşen kenarından $\overline{MREQ}$ gecikmesi, $\overline{MREQ}$ alçak		85	nsec	
	$t_{DH}\phi$ (MR)	Saatin yükselen kenarından $\overline{MREQ}$ gecikmesi yüksek		85	nsec	
$\overline{MREQ}$	$t_{DH}\phi$ (MR)	Saatin düşen kenarından $\overline{MREQ}$ gecikmesi, $\overline{MREQ}$ alçak		85	nsec nsec	C_L = 50pF
	$t_W\phi(\overline{MRL})$	Darbe genişliği, $\overline{MREQ}$ alçak	[8]		nsec	
	$t_W\phi(\overline{MRH})$	Darba genişliği $\overline{MREQ}$ yüksek	[9]		nsec	

[12] $t_c = t_{w(\phi H)} + t_{w(\phi L)} + t_r + t_f$

[1] $t_{acm} = t_{w(\phi H)} + t_r - 65$
[2] $t_{acf} = t_c - 70$
[3] $t_{ca} = t_{w(\phi L)} + t_r - 50$
[4] $t_{caf} = t_{w(\phi H)} + t_r - 40$
[5] $t_{dcm} = t_c - 170$
[6] $t_{dci} = t_{w(\phi)} + t_r - 170$
[7] $t_{cdf} = t_{w(\phi)} + t_r - 70$

[8] $t_{w(MRL)} = t_c - 30$
[9] $t_{w(MRH)} = t_{w(TH)} + t_f - 20$

$\overline{IORQ}$	$t_{DL}\phi$ (IR)	Saatin yükselen kenarından $\overline{IROQ}$ gecikmesi, $\overline{IORQ}$ alçak		75	nsec	C_L = 50pF
	$t_{DL}\phi$ (IR)	Saatin düşen kenarından $\overline{IORQ}$ gecikmesi, $\overline{IORQ}$ alçak		85	nsec	
	$t_{DH}\phi$ (IR)	Saatin yükselen kenarından $\overline{IORQ}$ gecikmesi, $\overline{IORQ}$yüksek		85	nsec	
	$t_{DH}\phi$ (IR)	Saatin düşen kenarından $\overline{IORQ}$ gecikmesi, $\overline{IORQ}$ yüksek		85	nsec	
$\overline{RD}$	$t_{DL}\phi$ (IR)	Saatin yükselen kenarından $\overline{RD}$ gecikmesi, $\overline{RD}$ alçak		85	nsec	C_L = 50 pF
	$t_{DL}\phi$ (IR)	Saatin düşen kenarından $\overline{RD}$ gecikmesi, $\overline{RD}$ alçak		95	nsec	
	$t_{DH}\phi$ (IR)	Saatin yükselen kenarından $\overline{RD}$ gecikmesi, $\overline{RD}$ yüksek		85	nsec	
	$t_{DH}\phi$ (IR)	Saatin düşen kenarından $\overline{RD}$ gecikmesi, $\overline{RD}$ yüksek		85	nsec	
$\overline{WR}$	$t_{DL}\phi$ (IR)	Saatin yükselen kenarından $\overline{WR}$ gecikmesi, $\overline{WR}$ alçak		65	nsec	C_L = 50 pF
	$t_{DL}\phi$ (IR)	Saatin düşen kenarından $\overline{WR}$ gecikmesi, $\overline{WR}$ alçak		80	nsec	
	$t_{DH}\phi$ (IR)	Saatin düşen kenarından $\overline{WR}$ gecikmesi, $\overline{WR}$ yüksek		80	nsec	
	t_W (WRL)	Darbe genişliği, $\overline{WR}$ alçak	[10]		nsec	
$\overline{M1}$	t_{DL} (M1)	Saatin yükselen kenırından $\overline{M1}$ gecikmesi, $\overline{WR}$ alçak		100	nsec	C_L = 50 pF
	t_{DH} (M1)	Saatin yükselen kenarından $\overline{M1}$ gecikmesi, $\overline{M1}$ yüksek		100	nsec	
$\overline{RFSH}$	t_{DL} (RF)	Saatin yükselen kenarından $\overline{RFSH}$ gecikmesi, $\overline{RFSH}$ alçak		130	nsec	C_L = 50 pF
	t_{DH} (RF)	Saatin yükselen kenırından $\overline{RFSH}$ gecikmesi, $\overline{RFSH}$ yüksek		120	nsec	
$\overline{WAIT}$	t_s (WT)	Saatin düşen kanarından $\overline{WAIT}$ ayarılama zamanı	70		nsec	C_L = 50 pF
$\overline{HALT}$	t_D (HT)	Saatin düşen kanırından $\overline{HALT}$ gecikme zamanı		300	nsec	
$\overline{INT}$	t_s (IT)	Saatin yükselen kenarından $\overline{INT}$ ayarlama zamanı	80		nsec	
$\overline{NMI}$	t_W (NML)	Darba genişliği, $\overline{NMI}$ alçak	80		nsec	

[10] $t_W(\overline{WRL}) = t_c - 30$

$\overline{BUSRQ}$	t_S (BQ)	Saatin yükselen kenarına $\overline{BUSRQ}$ ayarlama zamanı	50		nsec	
$\overline{BUSAK}$	t_{DL} (BA)	Saatin yükselen kenarından $\overline{BUSAK}$ gecikmesi, $\overline{BUSAK}$ alçak		100	nsec	C_L = 50 pF
	t_{DH} (BA)	Saatin düşen kenarından $\overline{BUSAK}$ gecikmesi, $\overline{BUSAK}$ yüksek		100	nsec	
$\overline{RESET}$	t_S (RS)	Saatin yükselen kenarına $\overline{RESET}$	60		nsec	
		ayarlama zamanı				
	tF (C)	Yüzere gecikme ($\overline{MREQ}$, $\overline{IORQ}$ $\overline{RD}$ ve $\overline{WR}$)		80		
	t_{mr}	$\overline{IORQ}$ öncesi MI kenarlı (kesme onaylama)	[11]		nsec	[11] $t_{mr} = 2t_c + t_{w(rl)} + t_f - 65$

NOTLAR:

A. *RD* aktif olduğu zaman veri *CPU* veri yoluna yetkilendirilmelidir. Kesme onaylaması sırasında *MI* ve *IORQ* aktif olduğunda veri yetkilendirilmelidir.

B. Bütün kontrol sinyalleri dahili olarak senkronizedir, böylece hepsi saate göre asenkron olabilir.

C. *RESET* sinyali en az 3 saat sinyali aktif olmalıdır.

D. Çıkış gecikmesi — Yüklenmiş kondansatör

$T_A = 70\ ^oC$ $V_Ç = +5\ V, \pm\ \%5$

Veri yolu için maksimum 200 pF ve adres ve kontrol hatları için 100 pF'a kadar yükteki her 50 pF artış için 10 nsaniye gecikme ilave et.

E. Tasarımda statik olmasına rağmen test t_W *(ϕH)*'nin maksimum 200 μsaniye olmasını garanti eder.

EK D

VOTRAX SC – 01 KONUŞMA SENTEZLEYİCİ FONEM TABLOSU

TABLO 1
FONEM TABLOSU

FONEM KODU	FONEM SEMBOLÜ	SÜRE (ms)	ÖRNEK KELİME
00	EH3	59	jacket
01	EH2	71	enlist
02	EH1	121	heavy
03	PA0	47	no sound
04	DT	47	butter
05	A2	71	made
06	A1	103	made
07	ZH	90	azure
08	AH2	71	honest
09	13	55	inhibit
0A	12	80	inhibit
0B	11	121	inhibit
0C	M	103	mat
0D	N	80	sun
0E	B	71	bag
0F	V	71	van
10	CH	71	chip
11	SH	121	shop
12	Z	71	zoo
13	AWI	146	lawful
14	NG	121	thing
15	AH1	146	father
16	OO1	103	looking
17	OO	185	book
18	L	103	land
19	K	80	trick
1A	J	47	judge
1B	H	71	hello
1C	G	71	get
1D	F	103	fast
1E	D	55	paid
1F	S	90	pass

FONEM KODU	FONEM SEMBOLÜ	SÜRE (ms)	ÖRNEK KELİME
20	A	185	day
21	AY	65	day
22	Y1	80	yard
23	UH3	47	mission
24	AH	250	mop
25	P	103	past
26	O	185	cold
27	I	185	pin
28	U	185	move
29	Y	103	any
2A	T	71	tap
2B	R	90	red
2C	E	185	meet
2D	W	80	win
2E	AE	185	dad
2F	AE1	103	after
30	AW2	90	salty
31	UH2	71	about
32	UH1	103	uncle
33	UH	185	cup
34	O2	80	for
35	O1	121	aboard
36	IU	59	you
37	U1	90	you
38	THV	80	the
39	TH	71	thin
3A	ER	146	bird
3B	EH	185	get
3C	E1	121	be
3D	AW	250	call
3E	PA1	185	ses yok
3F	STOP	47	ses yok

CH sesini üretmek için /T/, /CH/'den önce gelmelidir.
J sesini üretmek için /DI/J/'den önce gelmelidir.

TABLO 2 ÜRETİM ÖZELLİKLERİNE GÖRE FONEM KATEGORİLERİ

Sesli					Sesli Sürtünmeli	Sesli Durdurma	Sürtünmeli Durma	Sürtünmeli	Genizden	Sessiz
E	EH	AE	UH	OO1	Z	B	T	S	M	PA0
E1	EH1	AE1	UH1	R	ZH	D	DT	SH	N	PA1
Y	EH2	AH	UH2	ER	J	G	K	CH	NG	STOP
Y1	EH3	AH1	UH3	L	V		P	TH		
I	A	AH2	O	IU	THV			F		
11	A	AW	O1	U				H		
12	A2	AW1	O2	U1						
13	AY	AW2	OO	W						

Votrax, herhangi bir uyarı ve mecburiyete bağlı kalmaksızın herhangi bir zaman da üretim hattını veya özelliklerini değiştirme hakkını saklı tutar.
Copyright Votrax 1980

(İNDEKS)

A
8080 A mikroişlemcisi : 380-383
8085 A mikroişlemcisi : 380-383
A/D (Analogdan sayısala) çeviriciler :
Tek sayıcı tipi : 327-328
Tanımı : 311
Tek eğimli : 323
Çift eğimli : 324-327
Flaş : 321-322
Paralel karşılaştırma : 321-322
Simultane : 321-322
Özellikleri : 332
Mikroişlemcilere bağdaştırma : 590-591
Ardarda yuvarlamalı : 329-331
İzlemeli : 328-329
Açık kollektörlü TTL : 57
Açıklık süresi, örnekleme ve tutma : 333
Açılışta reset etme devresi : 239
ADA : 747
adam adım kontrol : 472-473
Adımlı Motor : 238, 586-588
Adres çözme : 449-452, 454-457, 531-533
Adres mandalı : 383
Adres tamponları : 433, 474-475
Adres yolu : 377
Akım çökmesi, tanımı : 47-48
Akım döngüsü : 604-606
Akım kaynaklandırma, tanımı : 47-48
Akış tabloları : 403-404
Akümülatör

		Sayfa
Intel 8080A mikroişlemcisi	:	380
Motorola MC 6800 mikroişlemcisi	:	493-499
Alan	:	624-625
Alçak güçlü Schottky TTL	:	60-62
ALGOL	:	746
Algoritmik makina	:	801-803
Altyordam	:	397-400, 408-412, 502-510, 661-667, 767
ALU (Aritmetik mantık birimi		
Amaç kodu	:	421
Anahtar titreşiminin giderilmesi	:	
Tuş takımı yazılımı	:	565-566
Donanım anahtarı	:	81-83, 197
Analog, tanımı	:	33-34
Analogtan sayısala çevirici (A/D)		
APL	:	747
Ardarda yuvarlamalı kaydedici (SAR)	:	331
Aritmetik mantık birimi	:	356-361
Arıza bilma, çeviri dili	:	422-423
ASCII Kodu	:	159-164
Asenkron Sayıcılar	:	208-219
B		
Bağımlılık gösterilişi	:	811-825
BAĞIMLILIK GÖSTERİLİŞİ	:	149-150
Bağlama, program modüleri	:	721-722, 743
BASIC	:	746, 748-765
Baskılı devre (PC) kartı	:	4
Başvuru tabloları	:	639, 657, 660
Baud hızı	:	242-243, 604
Bayrak flip-flopları	:	380-382, 530-531
Bayt kontrollu protokol	:	623

		Sayfa
BAYT- TANIMI	:	180
BCD	:	154-155
BCDIC	:	159-163
Belirsiz durum	:	46
Bellek haritalanmış I/O	:	452-453
Bellek haritaları	:	453-454
Bellek testi	:	300-307
Bellekleme durumları	:	460
Bellenir	:	384, 405
Bilgisayar yapısı	:	376
Bire tümleyen, tanımı	:	350
Biri yakalama	:	207
Birleşik Mantık	:	108-140, 164-166
Bit dilimli işlemciler	:	540-541
Bit, tanımı	:	35
Boole cebiri	:	39-42
Birleşik mantık devreleri için		
Bilinen mantak kapıları için		
De Morgan'ın Teoremi		
Dağılma özelliği	:	111, 112
Dernekler		
Basitleştirmeler		
BOOLE CEBİRİ	:	110-116, 133-135
Boru hattı	:	533-534
C		
CMOS Mantık	:	7, 60, 73-80
Kristal osilatör	:	89-90
Akım parametreleri	:	60, 75-77
Kapı yapısı (devren)	:	74
Kullanma ipuçları	:	80
Bağdaştırma	:	76-77, 96-97

Sayfa

Güç harcaması : 77
Yayılma gecikme zamanları : 60, 77
SOI, SOS : 80
Üç-durumlu çıkış : 78-79
Eşik gerilimi : 75-77
Transfer eğrisi : 76-77
İletim kapıları : 78-79
Gerilim parametreleri : 60, 75-77
COBOL : 746
CODEC : 338
CPU (Merkezi işlem birimi) : 376-382
CRT
Osiloskopun : 14
Televizyonun : 624-627
Kontrolör : 630-632
Karakter üreteci : 625-627
Kompozit video sinyal : 626
Alan : 626
Çerçeve : 626
Grafik : 632-633
Programlanabilir kontrolörler : 630-632
Raster : 626
Tarama : 626-627

Ç

Çarpımlar toplamı ifadesi : 110, 111
Çerçeve : 623-625
Çerçeveleme hatası : 244, 613
Çeviri dili : 404-412
Çeviriciyi kullanma : 421
Çevirim süresi, A/D için : 332

Sayfa

Çevrimsel-tekrarlama kontrolu (CRC) : 296
Çıkarma
İkilik : 348-352
Bire tümleyen çıkarma : 350
İkiye tümleyen çıkarma : 351
BCD ve üç-fazla BCD : 354
Sekizlik : 354
Onaltılık : 355
Çıkış yelpazesi
CMOS : 60
Tanımı : 49
ECL : 60, 65
TTL, standart : 44, 49, 60
Çoğullamalı göstergeler : 229-230
Çoğullayıcılar : 140-149

D

D Flip : 200-202
D Mandalı : 198, 202
D/A (Sayısaldan analoga) çeviriciler:
İkilik ağırlıklı direnç : 311-312
Özellikleri : 319-321
Tanımı : 311
Hibrit : 315-318
Monolitik : 315-317
Merdiven tipi : 313-315
Mikroişlemcilere bağdaştırma : 589-590
Dalgacık Sayıcılar : 208-219
Darbe daraltıcı : 547
Darbe Kod Modülasyonu (PCM) : 338
Darbe üreteçleri : 28
Değişkenler : 749

		Sayfa
DEĞİL kapısı (inverter)	:	40
PMOS		
Dekuplaj bağlantısı	:	7
Denetim-toplamı	:	296
Derleyici	:	744, 779
Devre-içi emülatör	:	727-728
Dinamik MOS	:	248-250
DIP (dual-in-line paket)	:	43
Diskler	:	292-299
Diziler	:	763-764, 766
DMA (Doğrudan bellek erişimi)	:	461-462
DMOS	:	73
Doğrudan I/O	:	452-453
Doğruluk tabloları	:	38-42
Dosyalar, PASCAL	:	766, 791-793
Durdurma noktaları	:	715-716
Durma durumu	:	462, 491, 468-469
Durum diyagramları	:	801-803
Düşme zamanı, tanımı	:	50
E		
EBAM	:	299
EBCDIC KODU	:	159-163
ECL (Emitör kuplaj	:	7, 60, 62-65
Kristal osilatörler	:	88-90
Aileler	:	64-65
Çıkış yelpazesi	:	60, 65
İç devre, çalışması	:	62-63
Gürültü payı, tasarım hileleri	:	64
Sonlandırma	:	64-65
Transfer eğrileri	:	62-65
Gerilim parametreleri	:	62-64

		Sayfa
Editör programı	:	727-728, 743
EEPROM	:	175-176
Endüstriyel işlem kontrol	:	686-689
Entegratör	:	325-326
EPROM	:	175-176
EPROM programlayıcı	:	631, 668-673
Eşik gerilimi	:	46, 78-79, 63-64
F		
FIFO	:	584
Flip-Floplar	:	200-207
Fonemler	:	862-863
Fonksiyonlar	:	767
FORTRAN	:	746, 765-779
Frekans Anahtarlama (FSK)	:	291, 606
Frekans sayıcılar	:	224-235
G		
Galpat	:	301
Geciktirme sabitleri	:	413-416
Geçici gerilim başarılması	:	7
Geçici kaydedici	:	383
Geçiş zamanları	:	50
Giriş-çıkış bağdaştırma	:	376, 515, 545-638
Giriş-çıkış portları	:	376, 452-456, 516-519
Glomper kıskacı	:	13, 17
GRAY KODU	:	154-154
Güç harcaması	:	60, 77
Gürültü payı	:	
CMOS	:	76
TTL	:	46

Sayfa

H

Halka Sayıcılar : 238-241
Hamming Kodu : 159
Hat sürücüler ve alıcılar : 94
Hibrit D/A çevirici : 315-318
Hız-güç çarpımı : 59-60
HMOS : 73

I - İ - J

I2L : 66-67
IBUG Monitör : 673-675
Integratör gerikesmesi : 688
Intel 8048 ailesi : 480-485
Intel 8080A sistem donanımı : 429-437
Intel 8085A sistem donanımı : 438-449
İkilik (tabanlı) sayılar (kod) : 34, 151
İkilik bölme : 364-366, 701-702
İkilik çarpma : 361-364, 664
İkilik çıkarma : 348-352
İkilik toplama : 342-344
İkilik virgül : 35
İkiye tümleyen, tanımı : 351
İletim hatları : 65, 90-93
İletim kapısı : 78-80
İndis kaydedici : 501-502
İşaret analizi : 736-737
İşaret biti : 352-353
İşlemsel kuvvetlendiri : 83-84, 312-314
İşlemsel kuvvetlendirici
İşlenen : 384
İşletim sistemleri : 743
İz, (manyetik dist için) : 294-298

Sayfa

JK Flip-Floplar : 203-206
Josephson Kavşağı Bellekleri : 282

K

6502 komut kümesi : 531-533, 535-536
K-haritası (Karnaugh Haritası) : 127-134
Kansas Şehri Standardı : 291, 571
Karakter dizileri : 761-762
Karakter üreteçleri : 625-627
Kararsız multivibratör : 86-87, 88-89
Karma mantık : 121-126
Karşılaştırıcılar : 83-84, 321-322
Kayan virgüllü sayılar : 368-371
Kaydedici çiftleri : 383
Kaydediciler : 236-256, 362-363, 517
Kaydırmalı kaydedicileri : 236-256
Kaynak program : 421
Kelime tanıyıcı : 19
Kesmeler (Interrupts) : 463-465, 521-523, 445, 552, 691-694
Kesmeli giriş-çıkış : 545, 619-620
KOD ÇEVİRİCİLERİ : 164-167, 179
KOD ÇÖZÜCÜ : 167-169
Kodlar :
ASCII : 159-163
BCD : 154-155
BCDIC : 159-163
İkilik : 34, 151
EBCDIC : 159-163
Üç-fazla BCD : 154, 155
Gray : 154, 155
Onaltılık (hex) : 152
Hollerith, delikli kartlar

		Sayfa
Sekizlik	:	151
Selectric, IBM	:	160-163
Yedi-parçalı	:	156
Komut kod çözücüsü	:	383, 491
Komut çevrimleri	:	429-432, 439-441, 519, 533-534
Komut kaydadeci	:	383, 491
Komut kümesi		
Intel 8048	:	480-485, 537
Intel 8080A / 8085A	:	384-403
MOS Technology 6502	:	531-533
Motorla MC 6800	:	492-504
Zilog Z-80	:	477-479
Kontrol kaydedicisi	:	562
Kontrol yolu	:	377
KOPLAYICI	:	169-170
Kristal osilatörler	:	88-90
Kütüphane fonksiyonları	:	751
L		
Lehimsiz temel örnek kartları	:	2-5
LIFO	:	401
Likid kristal göstergeler	:	585-586
M		
MACRO'lar	:	420-421
Makina çevrimleri	:	428-432, 439-441, 519, 533-534
Makina dili	:	384, 727
Mandallar	:	195-202
NMOS-TTL-NMOS	:	97
TTL-CMOS-TTL	:	61-62, 75-76, 96
TTL-ECL-TTL	:	98
TTL-I^2L-TTL	:	98

		Sayfa
TTL-PMOS-TTL	:	97
TTL alta ile arabağlantıları	:	61-62, 96
Mantık analizörleri	:	302, 306, 29, 734
Mantık darbe üreteci	:	9
Mantık fonksiyonları	:	
Bilinen kapılar için	:	38-42
74181 ALU için	:	358
Mantık kapıları	:	38-42
VE	:	38-40
Tanımı	:	38
VE DEĞİL	:	40
VEYA DEĞİL	:	41
VEYA	:	39
Anahtar olarak	:	
Özel VEYA DEĞİL	:	42
Özel VEYA	:	42
Mantık kapılarını bağdaştırma	:	97, 98-102
Mantık probları	:	12
Manyetik bellekler	:	284-300
Maskeleme	:	392, 465-468
MC 6800 komut kümesi	:	482-504
MEK 6800D2	:	668-672
Merkezi işlem birimi (CPU)	:	376-382
metin editörü	:	417-419
Mikroişlemci sistemi		
Mikroişlemciler, 16 bitlik	:	537-538
Mikrokomutlar	:	541
Minterim ifadesi	:	130
Mod sayıcılar, bölücüler	:	210
MODEM	:	606-610
Modemler	:	606-614

		Sayfa
Modüler programlama	:	524-525, 658-659, 646, 673-675, 715-717, 741, 780, 793-794
Monitör programı	:	441, 742
Monolitik (tek)	:	315
Monotonluk, D/A çevirici	:	321
MOS mantık aileleri	:	7
MOS mantık aileleri	:	67-80
MOS Technology 6502 sistem donanımı	:	528-530
Donanım	:	528-530
MOS ters çevirici	:	249, 250
MOS transistörler	:	67-69
Motor hız kontrolu	:	589
Motorla MC 6800 sistem donanımı	:	490-492
Motorla MC 6800 ün soyundan gelenler	:	525-527
Multivibratörler	:	86-88
N		
NAND Mandalı	:	198-200
Negatif mantık	:	118-121
Nibıl tanımı	:	451
Niceleme hatası	:	332
NMOS	:	78
NRZ	:	291
O		
Okuma çevrimi	:	519
Okuma çevrimi, RAM	:	269-275
Onaltılık Kod	:	152
Onaylı giriş-çıkışlar	:	553-564
Onluğa ayarlama işlemi	:	346
Operatörler	:	749, 766-767, 786
Optik kuplaj	:	102
Oransal kontrol	:	688
Osiloskop probları	:	16-17

		Sayfa
Dengeleme	:	16-17
Osiloskoplar	:	12-28
a.a. kuplaj	:	15
Değişen tarama modu	:	14
Otomatik tetikleme modu	:	20
Işın bulma düğmesi	:	13
Kalibratör çıkışı	:	17
Bölümlenmiş tarama modu	:	14
d.a. kuplaj	:	15
Geciktirilmiş tarama modları	:	22-23
Dış tetikleme kaynağı	:	18
Netlik kontrolu	:	14
Yatay ve düşey kontrollar	:	13
Işın şiddeti kontrolu	:	13
İç tetikleme kaynağı	:	18
Hat kaynağı	:	18
Normal tetikleme modu	:	20-21
Tek-tarama modu	:	21
Tarama mod kontrolları	:	14, 21-23
İz döndürme kontrolu	:	14
Tetikleme bekletme kontrolu	:	21
Tetikleme giriş kuplaj kontrolları	:	19-20
Tetikleme eğim ve seviye kontrolları	:	20
X-Y modu	:	23-25
Z ekseni	:	25
Otomatik test cihazları	:	734-738
Ö		
Örnekle ve tut devresi	:	27, 332
Örnekleyen osiloskop	:	26-28
Özel NOR (XNOR) kıpısı	:	42
Özel OR (XOR) kıpısı	:	42

		Sayfa
P		
2716 nın programlanması	:	675-685
Papatya zinciri:	:	7
Paralel yol standartları	:	593-603
Parite	:	183
PASCAL	:	747, 780-793
PC (baskılı devre) kartı	:	4
PCM (Darbe Kod Modülasyonu)	:	338
PIA	:	515-519
PID kontrolör	:	688
Piksel, tınımı	:	633
PL1	:	746
PMOS	:	73
PMOS ters çevirici	:	248
PARİTE HATASININ BULUNMASI	:	157, 158
Program sayıcı	:	383
Programlanabilir dizi mantığı (PAL)	:	184
Programlanabilir mantık dizisi (PLA)	:	183
PROM	:	175
PSW (Program durum kelimesi)	:	401
R		
RAM	:	263-283, 514
Rampa üreteci	:	323-324
Renk-kodlu tellerle bağlantı	:	6, 709
ROM	:	172-180, 511-514
Röleler	:	100-102
RS Mandal	:	198-200
S		
Saatler (clocks)		
Kristal osilatörler	:	88-90
Sayısal		
555 zamanlayıcı, multivibratörler	:	86-88
Gerçek-zaman, mikroişlemciler	:	552-553, 691, 694
Sabitler	:	749, 765-766
Saklama zamanı	:	55
Saklayıcı osiloskoplar	:	25-26
SAR (Artarda yuvarlamalı kaydedici)	:	331
Sayfa sınırı	:	532
Sayı sistemleri	:	35-37, 366-371
Sayı sistemleri Çevrimleri	:	36-37, 659-662, 701
Sayım penceresi	:	226
Sayısal Okumalı Osiloskoplar	:	28
Sayısal saat	:	215-218
Sayısal ses iletimi	:	338
Sayısal voltmetre (DVM)	:	11, 333-337
Sayısal, tınımı	:	33-35, 311
Schmitt tetiği	:	85
Schottky diyodu	:	59-62
Schottky TTL	:	7
SDK-85 için 8085A sistem şeması	:	443-445
SDK-85 mikroişlemci geliştirme kartı	:	441-442, 642-644
SDK-85 tabanlı işlem kontrol sistemi	:	689-694
Sekizlik kod	:	151
Sektör	:	294-296
Sembol tablosu	:	421
Senkron alıcı-verici	:	615-621
Senkron Sayıcılar	:	220-228
Seri giriş-çıkış	:	622-623
Seri veri haberleşmesi	:	243, 244
Sinyal uyumlanması	:	81-100
Sizde komutlar	:	419-420
Sonlandırma	:	63-64, 93-95
SSS zamanlayıcı	:	87-90
Süper işlemciler	:	537-539
İşlem kontrol sistemi	:	689-694

	Sayfa
T	
T Flip-Flop	208
Taban gösterici	538
Tam çıkarıcı	349
Tam dupleks iletim	623
Tam toplayıcı	344
TAMPON	111, 112
Tanımlamalar, PASCAL	780-785
Taramalı giriş-çıkış	545, 619-620
Taşma hatası	244
Tazeleme, RAM	278-280
Tek kararlı multivibratör	87-88
Tek komut yüretme	422
Tel sargı bağlantıları	2, 4
Televizyon	624-630
Temel örnek devresi	
Sayısal devreler için	1-2
Mikroişlemci temelli sistemler	708-709
Temel örnek geliştirme	1-5
Terazi, mikroişlemci tabanlı	640-668
Ters çevirici (tümleyici): tanımı	40
Bire tümleyen	350
İkiye tümleyen	351
Test aletleri	
Akım izleme probu	9
DVM	11
Mantık probu	12
Mantık durum analizörü	29
Ohmmetre	8
Osiloskoplar	12-28
Darbe üreteçleri	28
Zaman-işaret üreteci	28-29
İşaret analizörü	736-737

	Sayfa
Topraklama problemleri	95
Toplama	
İkilik	342-344
BCD ve üç-fazla BCD	345-346
Sekizlik	347
Onaltılık	347
Transistör anahtar, bipolar	51-52
Tree wiring	7
TTL	7, 43-62
Tutma durumları	461
Türev alıcı	687-688
U - Ü	
UART	243-244, 526, 529, 603-604
Uçucu bellekler	264
USART	243-245, 437
Üç-durumlu çıkış	58, 78-80
Üç-fazla BCD Kodu	154, 155
V -W -X	
VE fonksiyonu	38-39
Boole Cebiri ifadeleri	39
Şematik sembolü	38
Doğruluk tablosu	39
Veri iletim standartları	603-610
Veri toplama sistemleri	591, 691-694
Veri yolu	377
Veri yönü kaydedicisi	446, 517
VERİ DAĞITICILAR (YOL ÇOKLAYICILAR)	140-144
VMOS	73
Winchester disk sürücüleri	297-299
X-Y modu, tramler eğrileri için	23-25

		Sayfa
Y		
Yakalama süresi	:	333
Yan geçit Kapasitörleri	:	7, 62, 708-709
Yarım çıkarıcı	:	349
Yarım dupleks iletim	:	623
Yarım toplayıcı	:	343
Yayılma gecikme zamanı	:	44, 46, 60-64
Yazılım tanımı	:	422
Yazma çevrimi	:	521
Yazma çevrimi, RAM	:	277
Yedi-parçalı gösterge kodu	:	155-156, 571, 579-581
Yerinde Programlanabilir Kapı Dizileri (FPGA)	:	184
Yerinde programlanabilir Mantık Dizileri (FPLA)	:	180-183
Yerleşim zamanı	:	321
Yetki girişi	:	197-199
Yetki girişi, üç durumlu eleman	:	57-58
Yetkilendirme	:	465-468
Yığın göstericisi	:	383
Yığın işlemleri	:	400-402, 501-502
YOL ÇOKLAYICILAR	:	140-144
Yordam	:	397
Yorumlayıcı	:	744
Yük hücresi	:	640-641
Yük kupleli elemanlar	:	254-256, 283
Yükleme girişi	:	198
Yüksek seviyeli diller	:	741
Yükselme zamanı	:	50
Yürütme zamanı, program	:	413-414

		Sayfa
Z		
Z-80 sistem donanımı	:	476-477
Z-80A komut kümesi	:	477-479, 852-854
Zahiri toprak	:	312
Zaman - işaret üreteci	:	28-29
Zamanlama Diyogramları	:	194-240, 431-441, 457-459
Zamanlama parametreleri	:	457-460, 519-521, 533-557, 628-629, 675, 856

TERİMLER SÖZLÜĞÜ

Alçakta aktif (Düşek seviye): Giriş veya çıkışın sıfır volt seviyede olması.

Anma değeri (Nominal değer): Aktif devre elemanının çalışma bölgesindeki sınır değeri.

ASCII (American Standart Code Fore Information Interchange): Bilgi alışverişi için Amerikan Standart kodudur. 7 bit ile 128 tane karekter kodlanabilir. Bütün büyük, küçük harf, rakkamlar, noktalama işaretleri ve kontrol karakterleri için kullanılır.

Baud hızı: Bir saniyede iletimdeki bit sayısıdır. örnek olarak **2Kbit/sn.**

Bayt: İki tabanlı sayı sisteminde kullanılan sekiz bit uzunluğundaki bilgiye bir bayt denir.

Bellek haritası (Memory map): Veri alanlarının yazıldığı bölgedir. Kapasiteye göre değişen kullanma şekilleri vardır. RAM, ROM ve kullanılmayan bölge olarak kısımları vardır. Küçük programlar (yordamlar), büyük programlar yazılır veya yazılıdır. Adres sırasına göre bilgi yüklenir.

Boole Cebri: İki tabanlı sayı sistemi örneğin; $(1110)_2$

RS – 232C: Seri verileri uzak mesafelere göndermekte kullanılan ara bağlantı kablosu.

ALU (Aritmetik Mantık Birimi): CPU'den gelen istek ile aritmetik fonksiyonlar gerçekleştirilir. Sonra tekrar CPU'ya gönderilir.

Kristal osilatör: Kristale elektriki basınç uygulanmakla elde edilen sinyal üreteci.

Frekans birimleri; Hz, KHz, MHz ve GHz'dir.

(1Hz = 1 saykıl/saniye'dir.)

CPU (Merkezi İşlem Birimi): Giriş-çıkış aritmetik mantık birimi gibi birçok birimin kendisine bağlı olan merkez.

CRT (Cathode - Ray - Tube): Katottan çıkan anato (ekrana) giden ışık hüzmesi kullanılan tüp, ekran.

CRT Raster: Televizyon veya bilgisayar ekranında aydınlatmak için yapılan yatay ve düşey ekran ışık taraması.

Sıvı – kristal – display (Liquid – crystal display – LCD): Yedi parçalı sayısal gösterge ile karekterin gösterilmesi.

Darbe üreteci (clock puls): Kare dalga pals üreteci.

DIP (Dual – In – Line Package): Çift sıralı entegre ayakları, baskı devre veya sokete takılacak şekilde kullanılır.

Darlington bağlantı: İki transistörün arka arkaya bağlanması ile elde edilen güçlü bir transistör.

Diyot (Diode): İki farklı p ve n gibi yarı iletkenin birleştirilmesi ile elde edilen aktif eleman. Doğru (ileri) yönde uygulanan gerilimle birleşme yüzeyi daraldığı için iletime geçer. Ters yönde ise birleşme yüzeyi genişler ve iletimi keser.

BCD (Binary – Coded – Desimal): İkilik kodlanmış onluk sayılar: İkilik kodu BCD'ye çevirmek için 74185 kullanılır. BCD'yi ikilik koda çevirmek için genellikle 74184 kullanılır.

ROM (Read Only Memory): Üç temel tipi vardır. Mask programlanmış ROM'lar. Programlanabilir. ROM'lar yani PROM'lar ve silinebilir programlanabilir ROM'lar yani EPROM ve EEPROM'lar.

Mask programlanmış ROM'lar: Sabit programlanmış ROM'lardır.

PROM'lar: 74287 gibi programlanabilir PROM'lar.

EPROM'lar (Erasable PROMS): Silinebilir programlanabilir ROM'lar MOS devreler kullanılır.

EEPROM'lar (Electrically Erasable PROMS): Elektrikle silinebilir programlanabilir ROM'lar (EEPROM'lar) ultraviyole ışıkla silinebilir. MOS devreler kullanılır.

LED (Light Emitting Diode): Işık yayan diyot.

FFIO (First In First Out): İlk giren ilk çıkar. Hafızaya bilgi giriş-çıkışı.

LIFD (Last In First Out): Son giren ilk çıkar. Hafızaya bilgi giriş-çıkışı.

Flip – Flop'lar: Birinci transistör çıkışı ikinci transistörün girişine, ikinci transistörün çıkışı birinci transistörün girişine bağlı iki durumlu devre.

Giriş – Çıkış portları (Input – Output ports): Devrelere girişve çıkış yapılan bağlantı yeri. Örnek: Bilgisayarda klavye giriş portuna, monitör çıkış portuna bağlanır.

UART (Universel Synchronous or Asynchronous Receiver – Transmitter): Üniversal senkron veya asenkron alıcı – vericidir. UART'la benzer bir devre ile senkron haberleşmede karekterleri sabit bir hızda gönderilir.

Frekans: Saniyedeki titreşim sayısına 1Hz. denir.

1Hz = 1 saykıl/saniye'dir. (1Hz = 1 c/sn)

GRAY kodu: Çoğunlukla torna gibi bilgisayar kontrollu makinalarda veri kodlamada kullanılan ikilik kod. Mil üzerindeki diskin bölümlerinin değişmesi ile gösterilir.

BIT (BInary digiT): Bir devrede veya ünitede bir hattın üzerindeki "1" veya "0" olarak değeridir.

Akümülatör kaydedici: Dışardan girilen bilgileri giriş–çıkış portuyla MİB'ne aktaran kaydedici veya çıkış olarak verilen bilginin kaydedici ile aktarılması işidir. Veri girişi ve çıkışı kaydedicinin üzerinden yapılır. ALU'da aynı işi MİB'inden gönderilen aritmetik işlem yapılması için alan ve ALU'dan işlem yapıldıktan sonra tekrar MİB'e gönderen kaydedicilere akümülatör kaydedici denir.

Mantık Kapıları: Aşağıdaki mantık kapıları özelliklerine göre sembollerle gösterilir ve devre giriş-çıkış değerleri doğruluk tablosu ile gösterilir.

VE (and)

VEYA (or)

VE DEĞİL (Nand)

VEYA DEĞİL (Nor)

Özel VEYA (Exor)

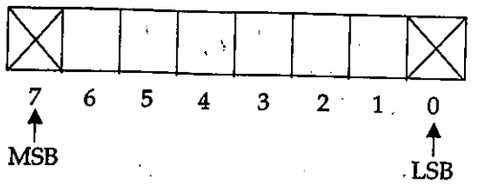
Özel VEYA DEĞİL (Exnor)

Doğruluk Tablosu: Mantık devreleri çalışmasını tablo halinde açıklamak için kullanılır.

Donanım: Bilgisayarın iç yapısını oluşturan tüm devreler.

En küçük değerlikli bit (LSB): 8 bitlik bir ünitede iki tabanlı sayı sistemiyle ileri sayarken sağdaki ilk basamak olarak kullanılan bit.

En büyük değerlikli bit (MSB): 8 bitlik bir ünitede iki tabanlı sayı sistemiyle ileri sayarken soldaki son olarak kullanılan 7.'nci bit.



Kenar tetikleme: Kare dalga şeklindeki bir sinyalin yükselen veya düşen kenarı kullanılır.

Karnaugh haritası: Sayısal devrelerde problemleri kurallar ve tablolar yardımı ile çözülmesi.

Geri besleme (Feedback): Bir devrede çıkış sinyalinin bir miktarını girişe vermek sureti ile daha kararlı çıkış almak için yapılan bağlantı.

Zıplama (titreşim) giderilmesi: Anahtarı açıp-kapatırken istenmeyen titreşimler olur. Bunu schmith trigger devreleri ile söndürmek (kararlı çıkış almak) mümkündür.

TÜRKİYE HARİTASI

