Lyubitsa Markudova
İskra Yovanovska
Yasna Domazetovska

III yıl için

DİJİTAL ELEKTRONİK VE MİKROİŞLEMCİLER

2011

Değerlendirici Komisyonu

- Dr. Mariya Katsarska, Elektroteknik ve Bilgisayar Teknoloji Fakültesi, profesör
- Müh. Sofiya Temkova, “Mihaylo Pupin” – ÜBOO öğretmeni
- Müh. Nevenka Smilevska, “Mihaylo Pupin” – ÜBOO öğretmeni

*
* *

Redaksiyon
Prof. Dr. Arif Ago

Düzeltici
Dr. Aktan Ago

Çeviri
Ervin Salih

*
* *

Bilgisayar tasarımı
Lyubitsa Markudova

*

Yayıncı: Makedonya Cumhuriyeti Eğitim ve Bilim Bakanlığı

Basımevi: Grafiçki Centar Ltd., Üsküp

Tiraz: 50

Makedonya Cumhuriyeti Eğitim Bakanlığı Nr. 22-5315/1 ve 30.11.2010 tarihli kararıyla işbu kitabın kullanılmasına izin verilmiştir.

CIP - Каталогизација во публикација
Национална и универзитетска библиотека “Св.Климент Охридски” , Скопје
621.38 . 049 .. 77 (075.3)
004 . 31 (075 . 3)

МАРКУДОВА, Љубица
Дигитална електроника и микропроцесори : [електротехничар за електроника и телекомуникации] / Љубица Маркудова, Искра Јовановска, Јасна Домазетовска. - Скопје : Министерство за образование и наука на Република Македонија, 2011. - 326, [1] стр. : илустр. ; 23 см

ISBN 978-608-226-175-1
1. Јовановска, Искра [автор] 2. Домазетовска, Јасна [автор]
COBISS.MK-ID 86462986

Önsöz

Bu ders kitabı mesleki lise okulları, eletroteknik meslek ve elektronik ile telekomuniksayon elektro teknisyen eğitim profili liselerin üçüncü sınıf öğrencileri için öngörülmüştür. Ders kitabı yeni ders programına göre hazırlanmış ve amacı, üçüncü sınıf Dijital Elektronik ve Mikroişlemci dersinden eğitim içeriklerini tamamıyla kapsamaktır. Ders programı 2006 yılında, orta meslek okullar reformu çerçevesinde yenilenmiştir. Yapılan yeniliklerle meslek eğitimde ders programların içeriğiyle eğitimin çağdaştırılması sağlanmıştır.

Dijital elektronik ve mikroişlemciler eğitim dersi, üçüncü sınıfta haftada 5 ders ya da yıl içerisinde 180 ders fonu var. Bu ders kitabın yazarlar takımı için mikroişlemcilerin ve bilgisayarların gelişimin ilgi, özel sorunluluk var. Bilgisayarların kullanımı daha başarılı olması için onları iyi tanımamız gerekiyor. Bu ders kitabı, on yıllık çalışma, yani elektronik ve mikroişlemci tekniği alanından çok kapsamlı öğretim mazlemenin on yıllık araştırma, analiz ve seçim sonucudur. Kitabın hazırlanmasında, ders öğretim malzemesi sunumu ve öğretimi hakkında verdikleri düşünce ve öğütlerle öğrencilere de büyük katkısı bulunur.

Ders kitabı 9 konu kapsıyor.

Birinci konu, Temel kombinasyon ve ardaşıl bileşenler, ikinci sınıfta okunan Dijital eletronik ve mikroişlemci dersinden eğitim malzemesinin tekrarlamasıdır. Bu malzemenin tekralanması, öğretim malzemenin anlama açısından büyük önemi var, çünkü mikroişlemciler böyle bileşenlerden oluşuyor.

Dersin ikinci konusu, Bilgisayarların temeli, birinci sınıfta okunan Bilişim (Enformatik) dersiyle ilişkilidir. Öğrenciler, temel donanım ve yazılım bileşenlerin özellikler ve işlevler konusunda bilgilerini onaylamaları gerekiyor.

Mikroişlemcinin genel modelinin temel özellikleri, oluştuğu parçaları ve çalışma şeklini, Mikroişlemcinin genel yapısı olan üçüncü konuda tanıyacağız.

Mikroişlemcilerin genel yapısını tanıdıktan sonra, onun farklı belleklerle ve çevrebirim aygıtlarla bağlanma şekillerini öğrenmek gerekiyor. Bu dördüncü konu olarak Mikroişlemcilerin bağlanması konunun öğretim amacı olur.

Beşinci konunun başlığı Mikroişlemcinin programlanmasıdır. Üçüncü sınıfta okunan Dijital elektronik ve mikroişlemci dersi, programlama dillerin kullanımın incelediği tek sıralı derstir.

Programlama dili olarak, alt seviye programlama dilleri grubuna ait olan asembler (birleştirici, çevirici) dili seçilmiştir. Ancak onun makine diliyle olduğu yakınlığı, mikroişlemcilerin ve bilgisayarın diğer parçaların donanım kaynaklarını daha iyi şekilde incelememizi sağlıyor.

Altıncı konu, Mikroişlemciler gerçek bir mikro denetleyici olarak, PIC16f84'nin donanımın ve yazılımın incelenmesiyle ilgileniyor. Bu konu, son iki - üç yılda öğrenciler tarafında özel ilgi çekiyor, çünkü bu konudan alınan teoretik bilgiler, üçüncü ve dördüncü sınıfta okunan Pratik eğitim dersinde pratik olarak kullanıyorlar.

Yedinci konuyla, 8 - bitlik mikroişlemciler (8085 mikroşilemci), İntel şirketinin gerçek mikroişlemcilerin incelenmesi başlıyor. Kendi performanslarıyla bu işlemci Pentium mikroişlemcinin performanslarıyla kıysalanamaz. Ancak, 8085 mikroişlem-

ci hala büyük miktarda üretiliyor, çünkü basit elektronik cihazlar ve aygıtlarda geniş çapta kullanılıyorlar. Bu mikroişlemci, mikroişlemciler tekniğinde bir anlamda “daimidir”.

Sekizinci konuda, 16 ve 32 bitlik mikroişlemcilerin incelenmesiyle, yavaş ancak emin adımlarla modern Pentium mikroişlemcilerine yaklaşıyoruz. Önceki nesil mikroişlemcilerin oluştuğu parçalarını ve çalışma şeklini tanımadan, Pentium mikroişlemcilerin mimarini incelemek doğru değildir. Bu öğretim konusunda, öğretim malzemenin adım adım sunulmasının büyük önemi vardır.

Son, dokuzuncu konu olarak Pentium mikroişlemcilerini inceliyor. Öğrencilerin bu kopmpleksli öğretim mazlemesini daha iyi anlamaları için pin diyagramların analizinde blok yaklaşım ve seçim kullanılmıştır.

Ders kitabında her konunun sonunda bilgilerin uygulanması ve öğrencilerin elde ettikeri bilgiyi kontrol etmek için sorular, ödevler ve etkinliker vardır.

Bu ders kitabın yazar takımı, öğretim malzemesini süreçlerin ve işlemlerin belirgin şekilde açıklamasıyla ve açıklayıcı çizim ve resimlerle, edebildiği kadar daha basit ve anlaşılır şekilde sunmayı denedi.

Bu ders kitabın hazırlanışı sırasında verilen destek için ailelerimize teşekkür ediyoruz. Ayrıca verdiği kullanışlı öğütlerle kitabın daha iyi dönüştürmesine katkı sağlayan derleyen komisyonuna da teşekkür ediyoruz.

Yazarlar dan

İçindekiler

1. Temel Kombinasyonel ve Ardaşıl Bileşenler....1
   1.1. Sayı Sistemleri....1
      1.1.1. İkili Sayı Sistemi....1
      1.1.2. Onaltılı Sayı Sistemi....3
   1.2. Temel Mantıksal Devreler....5
   1.3. Tümleşik Mantıksal Devreler....8
   1.4. Kombinasyonel (Bileşimsel) Devreler....11
      1.4.1. Çoğullayıcı – Çoğullama Çözücü....11
      1.4.2. Kodlayıcı ve Kod Çözücü....13
      1.4.3. Aritmetik Toplama Devresi....14
      1.4.4. Birbitli Aritmetik - Mantık Birimi....16
   1.5. Ardaşıl Devreler....18
      1.5.1. Flip - flop....19
      1.5.2. Yazmaçlar....24
      1.5.3. Sayaçlar....26
   1.6. Multivibratörler....31
      1.6.1. Tekkararlı Multivibratör....31
      1.6.2. Kararsız Multivibratör....32
   1.7. Yarı İletken Bellekler....33
      1.7.1. Bellek Yongaların Organizasyonu....33
      1.7.2. RAM Bellekleri....37
      1.7.3. ROM Bellekleri....39
Sonuçlar....41
Sorular ve ödevler....42

2. Mikrobilgisayarların Temeleri....46
   2.1. Mikrobilgisayar Sistemlerine Giriş....46
   2.2. Bilgisayar Organizasyonu....47
   2.3. Mikroişlemcinin Temel Organizasyonu....49
   2.4. Bellek Organizasyonu....51
   2.5. Bilgisayarın Çalışması....54
Sonuçlar....57
Sorular ve Ödevler....58

3. Mikroişlemcinin Genel Yapısı....60
   3.1. Mikroişlemcilerin Tarihsel Gelişimi....60

3.2. Veri Türleri 63
3.3. Mikroişlemcinin Genel Yapısı 68
3.3.1. Mikroişlemcide Yazmaçlar.. 69
3.3.2. Aritmetik Mantık Birimi 72
3.3.3. Yönetim Birimi....74
3.4. Mikroişlemcilerin Ortak Özellikleri 76
3.5. Mikroişlemcinin Pin Diyagramı 79
3.6. Yığın Belleğin Kullanımı 81
Sonuçlar 84
Sorular ve Ödevler....86

4. Mikroişlemci Sistemleri....88
4.1. Mikroişlemcinin Bağlanması Sırasında Genel Terimler 88
4.2. RAM ve ROM Bellekleriyel Bağlanmak....89
4.3. Sanal (Virtüel) Bellek....91
4.4. Önbelleğin Organizasyonu....95
4.5. Giriş - Çıkış Aygıtlarla Bağlanmak....97
4.6. Pratik Giriş - Çıkış Bağlantı Noktaları....101
4.7. Kesintili Sistemler....104
4.8. DMA Aktarım 106
4.9. Adresli Kod Çözümlenmesi 109
Sonuçlar 114
Sorular ve Ödevler....116

5. Mikroişlemcinin Programlanması....120
5.1. Programlama Dillerin Ayırımı....120
5.2. Çevirici Yönergeyi Oluşturan Parçalar 123
5.3. Adresleme Şekilleri....124
5.4. Genel Mikroişlemcinin Yönergeler Kümesi....128
5.4.1. Veriler Aktarma Yönergeleri....128
5.4.2. Aritmetik Yönergeler 130
5.4.3. Mantıksal Yönergeler....132
5.4.4. Döndürme ve Kaydırma Yönergeleri 133
5.4.5. Atlama ve Dallanma Yönergeleri 134
5.4.6. Alt Programla Çalışma Yönergeleri....136
5.5. Programların Yazılması 137
5.5.1. Doğrusal Programlar....138
5.5.2. Dallanmalı Programlar....139
5.5.3. Döngülü Programlar....142
Sonuçlar 144
Sorular ve Ödevler....146

6. Mikro Denetimciler....148
6.1. Mikro Denetimcilere Giriş. 148
6.2. PIC 16f84 Mikro Denetimcisi....151
6.3. PIC 16f84'ün Bellekleri....153
6.4. PIC 16f84'te Adresleme Şekilleri....156
6.5. PIC 16f84'ün Pin Diyagramı....158
6.6. PIC 16f84'ün Zamanlama Sistemi....160
6.7. PIC 16f84'te Kesintilerin Tanınması 163
6.8. PIC 16f84'te Yazmaçlar....166
6.9. PIC 16f84'ün Yönergeler Kümesi....170
6.10. PIC 16f84 Mikro Denetimcinin Bağlanması....173
6.11. PIC 16f84'ün Programlanması 175
Sonuçlar 180
Sorular ve Ödevler....182

7. Sekizbitli Mikroişlemciler (8085 mikroişlemci)....186
7.1. 8085 Mikroişlemcinin Pin Diyagramı 186
7.2. 8085 Mikroişlemcinin Yapısı....190
7.3. 8085 Mikroişlemcisi İçin Makine Döngüleri 194
7.4. 8085 Mikroişlemcide Adresleme Şekilleri....197
7.5. Yönergeler Kümesi....200
7.5.1. Veri Aktarma Yönergeleri....200
7.5.2. Aritmetik Yönergeler 202
7.5.3. Mantıksal Yönergeler....204
7.5.4. Döndürme Yönergeleri....205
7.5.5. Atlama Yönergeleri 206
7.5.6. Alt Programla Çalışma Yönergeleri....207
7.5.7. Yığıt Bellekle Çalışma Yönergeleri 208
7.6. 8085 Mikroişlemci İçin Programların Yazılması....209
7.6.1. Basit Bir Kavşağın Trafik Düzenlemesi İçin Yazılım 211
7.6.2. Kodların Kıyaslanması....213
7.7. 8085 Mikro Bilgisayar Sistemi İçin Tümleşik Bileşenler 214
7.7.1. 8212 Arabilim Bileşeni 215
7.7.2. 8255 Programlanabilir Bileşeni....217
Sonuçlar 221
Sorular ve Ödevler....223

8. 16 ve 32 Bitli Mikroişlemciler....227
8.1. 8086 Mikroişlemcinin Temel Özellikleri....227
8.2. 8086 Mikroişlemcinin Pin Diyagarmı 230
8.3. Minimum ve Maksimum Çalışma Düzeni 233
8.4. Gerçek Çalışma Düzeni 235
8.5. 8086 Mikroişlemcide Adresleme Şekilleri....237
8.6. 8086 Mikroişlemcide Yönergeler Kümesi....240
8.6.1. Veriler Aktarım Yönergeleri....241

8.6.2. Dizilerle İşlemler 243
8.6.3 Aritmetik Yönergerler 244
8.6.4 Mantıksal Yönergeler.....247
8.6.5 Kaydırmak ve Döndürme.....249
8.6.6 Atlama ve Alt Program Yönergeleri.....250
8.7 8086 Mikroişlemcide Programların Yazılması.....252
8.8 8086 Mikrobilgisayar Sistemini Oluşturan Tümleşik Devreler.....254
8.8.1 8254 Tümleşik Devrenin Kullanımı 254
8.8.2 Programalanabilir 8254 Zamanlayıcı 258
8.9 80286 Mikroişlemcinin Temel Özellikleri 262
8.10 80386 Mikroişlemcinin Temel Özellikleri.....265
8.11 80486 Mikroişlemcinin Temel Özellikleri.....270
Sonuçlar 272
Sorular ve Ödevler.....275

9. Pentium Mikroişlemciler 279
9.1 Pentium 1 Mikroişlemcisi 279
9.2 Pentium 1 Mikroişlemcinin Pin Diyagramı 283
9.3 Pentium Bellek Yönetimi.....285
9.3.1 Korumalı - Sanal Çalışma Düzeni.....289
9.4 Pentium Pro Mikroişlemci 291
9.5 Pentium 2 Mikroişlemci.....293
9.6 Pentium 3 Mikroişlemci.....296
9.7 Pentium 4 Mikroişlemci.....298
9.8 Pentium Dual Core Mikroişlemci.....301
9.10 Pentium Mikrobilgisayar Sistemlerin Oluşması İçin Tümleşik Devreler304
9.11 Pentium Mikroişlemcinin Bellekle Bağlanması İçin Tümleşik Devreler.....304
9.12 İntel 865 Yonga Kümesi Tümleşik Devresi 307
Sonuçlar 313
Sorular ve Ödevler.....315

1. Temel Kombinasyonel ve Ardaşıl Bileşenler

1.1. Sayı Sistemleri

Mikroişlemcilerin çalışmasını daha iyi anlamak için ikili ve onaltılı sayı sistemlerinin iyi tanımasının büyük önemi vardır. Hergünlük yaşamda insanlar onlu sayı sistemini kullanıyorlar. Bu sayı sisteminde sayılar on rakamdan oluşuyorlar, 0'dan 9'a kadar. Sağdan sola başlayarak, birinci rakam birliklerin sayısını veriyor, ikinci rakam onlukların sayısını tanımlıyor, üçüncü rakam yüzlüklerin sayısını, sıradaki üç rakam binliklerin sayısını belirliyor vs. Bundan her rakamın sayıda aldığı yere bağlı olarak kendi ağırlığı vardır. Bu şekilde birliklerin sayısını veren rakamların $10^0=1$ ağırlığı vardır, onlukların sayısını veren rakamların ağırlığı $10^1=10$, yüzlüklerin sayısını veren rakamların ağırlığı $10^2=100$'dür, vs.

1.1.1. İkili Sayı Sistemi

İkili sayı sisteminde sayılar iki rakamdan oluşuyor, 0 ve 1. Onlu sayı sisteminde olduğu gibi, ikili sayı sisteminde de rakamların farklı ağırlığı var, sadece üsün tabanı on yerine 2'dir. Böylece, sağdan sola başlayarak birinci rakamın $2^0=1$ ağırlığı vardır, ikinci rakamın $2^1=2$, üçüncü $2^2=4$, dördüncü $2^3=8$ vs.

Eğer ikili sayı sisteminde yazılmış bir sayıyı onlu yazılışa dönüştürmek istersek, aşağıdaki örnek 1.1'de gösterildiği gibi ikili sayıdan tüm rakamları topluyoruz, ancak ondan önce ağırlıklarıyla çarpıyoruz.

Örnek 1.1: 101011_2 sayısını, onlu sayı sisteminde tanımlamak istiyoruz

$2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$ - ağırlıklar

$1\ 0\ 1\ 0\ 1\ 1_2 = 1 - 2^{\circ} + 1 - 2^1 + 0 - 2^2 + 1 - 2^3 + 0 - 2^4 + 1 - 2^5 = 1+2+4+8 = 43_{10}$

İkili sayı sisteminde 0 ve 1 rakamlarına bitler deniyor. $2^0=1$ ağırlığıyla çarpılan bite, en düşük değerli bit deniyor, en yüksek üsüyle (örneğimizde 2^5) çarpılan bite ise en yüksek değerli bit deniyor.

Eğer onlu sayı sisteminde ifade edilmiş sayıyı, ikili sayı sistemine dönüştürmek istersek, devama açıklandığı şekilde hareket ediyoruz. Sayıyı 2 ile bölüyoruz ve elde edilen kalanı yazıyoruz. Bölmede elde edilen tam sayının sonucunu yeniden 2 ile bölüyoruz ve kenarda kalanı yazıyoruz. Bu işlemi sonuç 0 elde edilene kadar tekrarlıyoruz. Ondan sonra elde edilen kalanları okuyoruz, hem de son yazılan kalandan başlayarak ilk yazdığımız kalanı yönüne kadar okuyoruz. Bu işlem Örnek 1.2'de gösterilmiştir.

Örnek 1.2: 38_{10} sayısını sıfırla ve birler dizisi olarak ifade etmek istiyoruz.

	kalan
38: 2 = 19	0
19:2=9	1
9: 2 = 4	1
4: 2 = 2	0
2: 2 = 1	0
1: 2 = 0	1

Elde edilen sonuç $38_{10} = 100110_2$ olacak.

Onlu sayı sisteminde ifade edilen sayıyı ikili sayı sistemine dönüştürmek yani onlu sayıyı 1, 2, 4, 8, 16, 32, 64, 128 vs, sayılardan bazıların toplamı olarak ifade ederek daha kolay olabilir. Bu sayılar aslında ikili sayı sistemin ağırlıklarıdır. Bu ağırlıktan bazısı toplama girirse, o rakam yerine 1 yazıyoruz, ağrılık toplama girmezse, onun yerine 0 yazıyoruz. Bu işlem örnek 1.3'te gösterilmiştir.

Örnek 1.3.: 38 sayısı, 32, 4 ve 2 sayılarının toplamı olarak elde ediliyor. Buna göre sağdan sola giderek ikili sayıda ikinci, üçüncü ve altıncı rakam 1 olacak, çünkü ikinci rakamın $2^1=2$ ağırlığı var, üçüncü $2^2=4$ ve dördüncü $2^5=32$.

Bazı kimse ikili sayı sistemi neden bu kadar önemli olduğunu sorabilir. İkili sayı sistemi çok önemlidir, çünkü makine diliyle karşılıklıdır. Makine dili, bilgisayarın ve tüm dijital aygıtların dilidir. Makine dilinde bir ceryan akımı ya da gerilim (voltaj) olduğu anlamındadır, sıfır ise ceryan akımı olmamak ya da gerilim olmamak anlamındadır.

1.1.2. On Altılı Sayı Sistemi

İkili sayı sisteminde yazılan sayılar çok uzundur. Yazılmanın kısaltılması için on altılı sayı sistemi kullanılıyor. On altılı sayı sisteminde sayılar 16 rakamdan oluşuyor. İlk 10 rakam, onlu sayı sisteminde olduğu gibi 0'dan 9'a kadardır, diğer rakamlar ise harf ile ifade ediliyor: $A_{16}=10_{10,}$ $B_{16}=11_{10,}$ $C_{16}=12_{10,}$ $D_{16}=13_{10,}$ $E_{16}=14_{10,}$ $F_{16}=15_{10}$. 16 endeksi yerine, onaltılı sayı sistemde ifade edilen sayılarda, sayının sonunda H harfini katarak işaret ediliyor. On altılı sayı sisteminden onlu sayı sistemine dönüştürmek, ikili sayı sisteminde onlu sayı sistemine dönüştüğü gibi aynı şekilde yapılıyor, sadece taban 2 yerine, taban 16 ile çalışıyoruz, üsün değeri ise rakamın sayıdaki yerine bağlıdır. Bu işlem örnek 1.4.'te gösterilmiştir.

Örnek 1.4: $2AH= 2\cdot16^1 + A\cdot16^0 = 2\cdot16 + A\cdot1 = 32 + 10 = 42_{10}$

Eğer onlu sayıyı on altılı sayıya dönüştürmek istersek, o zaman sayıyı 16 ile bölüyoruz ve kalanları kenarda yazıyoruz.

Bizim için daha büyük önemden on altılı sayıyı ikili sayıya dönüştürmek ve tersi tanımlıyor. On altılı sayıyı ikili sayıya dönüştürmek için hesaplamaya hiç gerek yok. Sadece on altılı rakamları ikili rakamlarla değiştiriyoruz. On altılı sayı sisteminde bir rakam, ikili sayı sisteminde 4 rakamla temsil ediliyor. On altılı sayı sisteminden herhangi bir rakam 1, 2, 4 ve 8 sayıların toplamı olarak tanımlanabilir. Toplama giren rakamların olduğu yerde 1 yazılıyor, toplama girmeyen rakamların yerine 0 yazılıyor. Bu işlem 1.5, 1.6 ve 1.7 örneklerinde gösterilmiştir.

Örnek 1.5: $C_{(16)}=12_{(10)}$ sayısını 8 ve 4 sayıların toplamı olarak gösterilebilir ve o yüzden ikili sayı sisteminde bu sayı 1100 olacak.
2 ve 1'in yerlerine sıfır yazıyoruz, çünkü onlar toplama girmiyor.

Örnek 1.6: $7_{(16)}=7_{(10)}$ sayısı 4, 2 ve 1 sayıların toplamıdır. Onun için bu sayı ikili sayı sisteminde 0111'e eşit oluyor.

Örnek 1.7:

E4H = $\underline{1110\ 0100}_2$

$E_{16=}\ 14_{10}=8_{10}+4_{10}+2_{10}$ $\qquad$ $4_{16}=4_{10}$

Bir ikili sayı sisteminde yazılmış sayıyı on altılı sayı sistemine dönüştürmek istiyorsak, o zaman ikili sayıyı sağdan sola 4'er rakamlık gruplara ayırıyoruz ve öyle her bir gruba bir on altılı sayı denk geliyor.

Eğer soldan son grubun 4 rakamı yoksa, sola sıfırlar ekliyoruz. Bu örnek 1.8'de gösterilmiştir.

Örnek 1.8: 000110101001 ikili sayısı on altılı sayı sistemine dönüştürmek gerekiyor

0001 1010 1001 = 1A9H

$1_{16}=1_{10,}$ $A_{16}=10_{10=}$ $8_{10}+2_{10,}$ $9_{16}=9_{10}=8_{10}+1_{10}$

On altılı sayı sisteminde toplama işlemi, onlu sayı sistemine benzer şekilde, aynı yerde olan rakamların toplanmasıyla yapılıyor. İki rakamın toplamı 16'dan daha büyük ise, toplam 16 ile bölünüyor ve sonuç olarak o yerde kalanı yazılıyor, ondan sonra gelen yere elde ettiğimiz bölümü aktarıyoruz. Toplama işlemi örnek 1.9'da gösterilmiştir.

Örnek 1.9:

```
  1          - aktarma
  3 A 2 B H  - birinci sayı
+ A 9 C 1 H  - ikinci sayı
  E 3 E C H  - toplam
```

Bu iki sayının toplanması sırasında sağ taraftan üçüncü poziyonda aktarma yapılıyor.

$A_{16}+9_{16}=10_{10}+9_{10}=19_{10,}$ $19_{10}:16_{10}=1_{10}$ kalan 3_{10}

Elde ettiğimiz 3 kalanı sağ taraftan üçüncü pozisyonda yazıyoruz, 1 bölümünü ise dördüncü pozisyona aktarıyoruz. Dördüncü pozisyon için şu sonucu elde edeceğiz:

$A_{16}+3_{16}+1_{16=}$ $10_{10}+3_{10}+1_{10}=14_{10}=E_{16}$

Toplama işleminde daha alçak pozisyondan daha yüksek poziyona aktarma var, çıkarma işleminde ise daha yüksek pozisyondan daha alçak pozisyona alıntı (ödünç) yapılıyor. Onlu sayı sisteminde 10 alıntı ediliyor, onaltılı sayı sisteminde ise 16. Çıkarma işlemi sıradaki örnekte verilmiştir.

Örnek 1.10:

```
   16   16
  7 5 D  3
- 1 C A  8
  5 9 2  B
```

8 sayısı 3 sayısından daha büyüktür. Daha yüksek pozisyondan ödünç alıyoruz. Birinci pozisyonda 16 ve 3'ü topluyoruz ve elde edilen 19 toplamından 8 sayısını çıkarıyoruz. Birinci pozisyonda 19 - $8=11_{(10)}$ = $B_{(16)}$ sonucunu elde ediyoruz. Birinci pozisyon için ikinci pozisyondan alıntı aldığımız için, D_{16} sayısını bir sayı için azaltıyoruz ve yeni değeri C_{16} olacak. C_{16} - A_{16} = $2_{16.}$

İkili sayı sistemi bilgisayarla ve diğer dijital aygıtlarda en yaygındır, onlu sayı sistemi insan için en anlayışlıdır, on altılı sayı sistemi ise sunduğu kısa yazılış şekli olduğu için daha sık kullanılıyor.

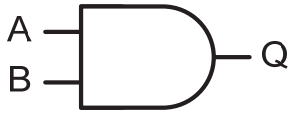
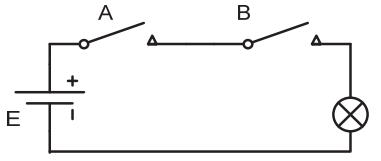
1.2. Temel Mantıksal Devreler

Dijital sinyallerin yapılması için kullanılan temel mantıksal devreler şunlardır: evirici (HAYIR), YA, VE, OVE (Olumsuz VE), OYA (Olumsuz YA), D - YA (Dışlamalı YA) ve D - OYA (Dışlamalı OYA) devreleridir. Bu mantıksal devreler için kullanılan sembolleri ve onların doğruluk tablolarını bilmek gerekiyor.Onların çalışmasını daha iyi anlamak için, her mantıksal fonksiyon elektrik devreyle temsil ediliyor. Elektrik devresi, iki anahtar düğme, tek yönlü gerilim kaynağı ve tüketici olarak ampulden oluşuyor.

VE devresi

VE devrenin çıkışında bir (1) elde etmek için tüm girişler yüksek seviyede, daha doğrusu mantıksal bir seviyesinde olmaları gerekiyor. VE devrenin çıkışında mantıksal sıfır elde etmek için girişlerden en az biri mantıksal sıfır seviyesinde olması yeterlidir.

Ampulun ışıklanması için her iki anahtarın kapalı olması gerekiyor.

Mantıksal sembol	Doğruluk tablosu			Elektrik devresi
	A	B	Q	
	0	0	0	
	0	1	0	
	1	0	0	
	1	1	1	

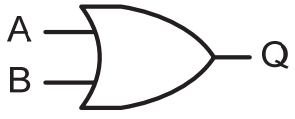
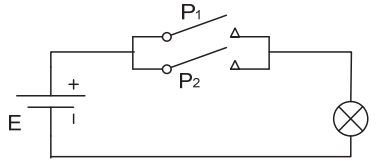
$Q=A \wedge B$

Resim 1.1 VE devrenin tanımlanması

YADA Devresi

YADA devrenin çıkışında sıfır elde etmek için tüm girişler alçak seviyede, daha doğrusu mantıksal sıfır seviyesinde olmalıdır. YADA devrenin çıkışında mantıksal bir elde etmek için girişlerden en az biri mantıksal bir seviyesinde olması yeterlidir.

Ampulun ışıklanması için en az bir anahtarın kapalı olması yeterlidir.

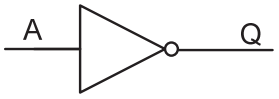
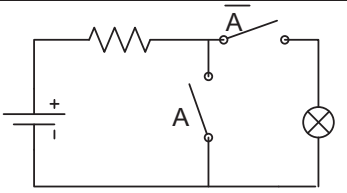
Mantıksal sembol	Doğruluk tablosu			Elektrik devresi
	A	B	Q	
	0	0	0	
	0	1	1	
	1	0	1	
	1	1	1	

$Q=A \vee B$

Resim 1.2 YADA devrenin tanımlanması

Evirici (DEĞİL)

Evirici giriş bütün birinci tümleyiciyi hesaplamak için kullanılıyor. Eviricinin girişinde mantıksal bir (1) varsa, o zaman çıkışta mantıksal sıfır elde edilecek ve tersi.

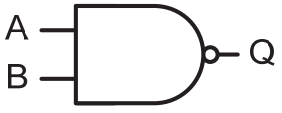
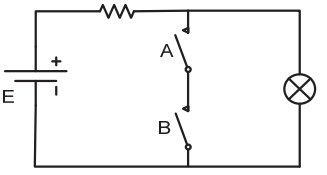
Mantıksal sembol	Doğruluk tablosu		Elektrik devresi
 $Q=\overline{A}$	A	Q	
	0	1	
	1	0	

Resim 1.3. Eviricinin tanımlanması

OVE Devresi

OVE devresi iki mantıksal devrenin bileşimidir, VE devresi ve evirici. Çıkışında sıfır sadece iki giriş bir olduğu zaman ya da yüksek seviyede olduğu sırada elde ediliyor. Çıkışta bir (1) elde etmek için sadece bir girişin alçak seviyede ya da mantıksal sıfır olması yeterlidir.

Ampulun sadece iki anahtarı kapalı olduğu zaman ışıklanmayacak. İki kapalı anahtar ampul için kısa bağlantıyı oluşturuyor.

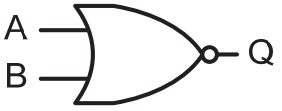
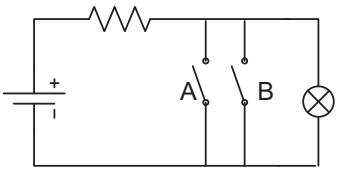
Elektrik sembolu	Doğruluk tablosu			Elektrik devresi
 $Q=\overline{A \wedge B}$	A	B	Q	
	0	0	1	
	0	1	1	
	1	0	1	
	1	1	0	

Resim 1.4. OVE devrenin tanımlanması

OYA Devresi

OYA devresi iki mantıksal devrenin bileşenidir, YADA devresi ve evirici. Çıkışta bir (1) sadece iki giriş sıfır ya da alçak seviyede olduğu zaman elde ediliyor. Çıkışta sıfır elde etmemiz için sadece bir girişin yüksek seviyede ya da mantıksal bir olması gerekiyor.

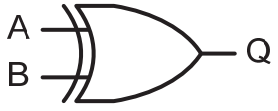
Ampul iki anahtardan en az bir anahtar kapalı olduğu zaman ışıklanmayacak, çünkü o zaman ampulde kısa bağlantı oluşuyor.

Mantıksal sembol	Doğruluk tablosu			Elektrik devresi
 $Q=\overline{A \vee B}$	A	B	Q	
	0	0	1	
	0	1	0	
	1	0	0	
	1	1	0	

Resim 1.5. OYA devrenin tanımlanması

D - YA ve D - OYA Devreleri

D - YA devresi giriş bitlerin kıyaslanmasını yapıyor. Giriş bitleri eşitse, çıkışta sıfır elde ediliyor, eşit değilse çıkış bir (1) elde ediliyor.

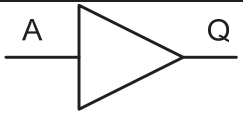
Mantıksal sembol	Doğruluk tablosu		
 Q=A⊕B	A	B	Q
	0	0	0
	0	1	1
	1	0	1
	1	1	0

Resim 1.6. D - YA devrenin tanımlanması

D - OYA iki mantıksal devrenin bileşenidir, D - YA ve evirici.Giriş bitleri eşit olduğu zaman çıkışta bir elde ediliyor, farklıysa sıfır elde ediliyor.

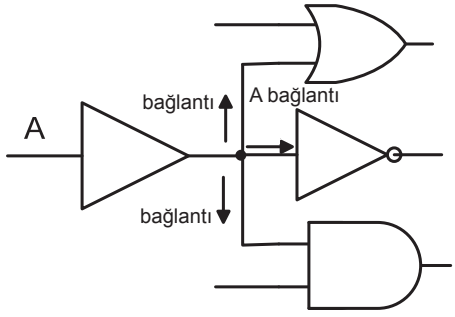
Arabellek (Bafer)

Arabelleğin çıkışı onun girişine eşittir.

Mantıksal sembol	Doğruluk tablosu	
 Q=A	A	Q
	0	0
	1	1

Resim 1.7. Arabelleğin tanımlanması

Arabellek sinyallerin işletmesini gerçekleştirmiyor, ancak diğer mantıksal devrelere kıyasen girişinde daha yüksek ceryan yükleri taşıyabilir. Böyle özellikten dolayı, arabellek, bir mantıksal devrenin çıkışında büyük sayıda diğer mantıksal devrelerin

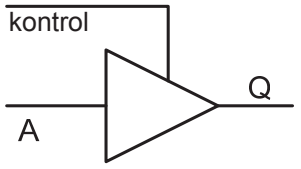


bağlanması gerekli olunca kullanılıyor. Bu resim 1.8'de gösterilmiştir

Resim 1.8. A çıkışının fazla mantıksal devrelerin arabellek aracılığıyla bağlanması

Sıradan arabellek dışında, dijital devrelerde **üç durumlu arabellek** olarak adlandırılan arabellek kullanılıyor**.** Bu arabellek türün üç iğnesi (pini) vardır: giriş, çıkış ve kontrol iğnesi. Kontrol biti sıfır olursa, o zaman çıkış yüksek empendans (iç di-

renç) durumundadır. Eğer kontrol biti bir (1) ise, o zaman çıkış girişe eşittir. Kontrol bitin aracılığıyla çıkış iğnenin açılmasını va kapanmasını yapıyoruz. Arabellek sıkça büyük sayıda aygıtın bir aktarım bağlantısına bağlanması gerektiğinde kullanılıyor.

Mantıksal sembol	Doğruluk tablosu		
	Kontrol	A	Q
	1	0	0
	1	1	1
	0	0	Yüksek empedans durumu
	0	1	Yüksek empedans durumu

Resim 1.9. Üç durumlu arabelleğin tanımlanması

1.3. Tümleşik Mantıksal Devreler

Temel mantıksal devrelerin bağlanmasıyla kombinasyonlu ve ardaşıl bileşenler gibi daha kompleksli devreler oluşuyor. Ancak, mantıksal devrelerin işlevsel bütüne bağlanmaları için belli koşulların yerine gelmeleri gerekiyor. Onlar aynı besleme gerilimi kullanmalıdır ve giriş ile çıkış mantıksal devrelerin uyumlu olmaları gerekiyor. Bu koşulları yerine getiren tümleşik devreler, tümleşik devre aileleri oluşturuyor. En tanıdık tümleşik devre aileleri TTL (Transistor - Transistor - Logic) ve CMOS (Complementary Metal - Oxide - Silicon) aileleridir. Bu aileler çerçevesinde birkaç tümleşik devre dizileri gelişmiş. Örneğin TTL ailesi çerçevesi içinde şu diziler vardır: TTL(74xx), TTL(74LSxx), TTL(74Sxx), TTL(74ALSxx), TTL(74ALSxx), CMOS ailesi çerçevesi içinde ise şu diziler vardır: CMOS(40xx), CMOS(74HCxx), CMOS(CD4xx).

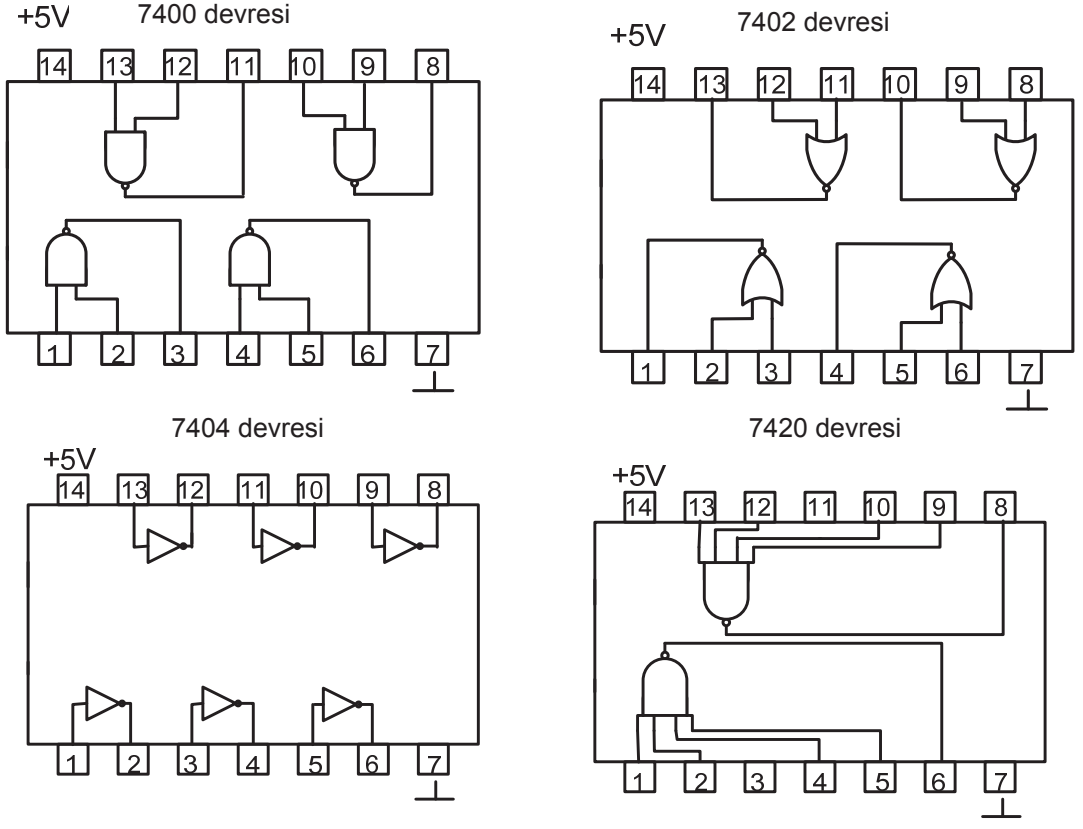
TTL ailesinden tüm tümleşik devreleri +5V±0,25V besleme gerilimi kullanıyorlar. Tablo 1.1'de mantıksal sıfır durumu ve mantıksal bir durumu için gerilimin giriş - çıkış seviyeleri varilmiştir.

Mantıksal seviye	**Giriş gerilimi**	**Çıkış gerilimi**
0	0,8V max	0,45V max
1	2V min	2,4 V min

Tablo 1.1. TTL tümleşik devre ailesi için gerilim değerleri

Giriş pinine (iğnesine) sinyal gelince ve gerilimin değeri 0.8 ve 2.0 V arası olursa, o zaman durum belirsiz (tanımlanmamış) olacak.

Resim 1.10'da TTL ailesinde birkaç tümleşik modelleri verilmiştir. Tüm bunlar standart TTL dizilerinden 74xx dizisine aittir ve çift bağlantı ambalajındadır. 7400 devresi dört OVE geçidi içeriyor ve her birin ikişer girişi var. 7402 devresi dört OYA geçidi içeriyor ve her birinin ikişer girişi var. 7404 altı birbiriyle aynı evirici içeriyor ve 7420 devresi iki OVE devresi içeriyor ve her birinin dörder girişi var. Kullanılmamış VE ve OVE geçitlerin girişleri besleme gerilimle bağlanmaları gerekiyor. Bu gerilim 5,5V'tan daha yüksekse, pull up rezistörlerin (dirençlerin) kullanılması gerekiyor. Kullanılmamış YADA ve OYA geçitlerin girişleri kütleyle bağlanmaları gerekiyor.



Resim 1.10. TTL aileden tümleşik devrelerin yapısı

Tablo 1.2'de mantıksal sıfır ve mantıksal bir seviyeleri için giriş ve çıkış ceryanın en yüksek değerleri verilmiştir

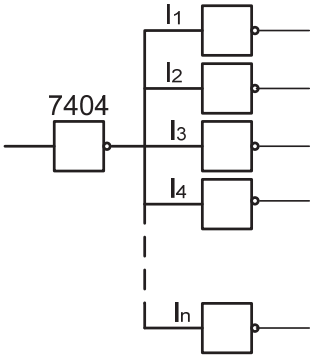
Mantıksal seviye	Giriş ceryanı	Çıkış ceryanı
0	- 1,6mA max	16mA max
1	0,04mA max	0,4mA max

Tablo 1.2. TTL ailesinde tümleşik devreleri için ceryan değerleri

Mantıksal sıfır seviyesi için giriş ceryanın önündeki eksi işareti, ceryanın yönü pin yönüne değil pinden yöneldiği demektir. Ceryanların en yüksek değerleri mantıksal devrelerin çıkış pinlerin taşıyabileceği yükü hesaplamak için kullanılıyorlar.

Bir mantıksal devrenin çıkışında başka kombinasyon devreleri bağlanabilir. Daha iyi uyumluluk sağlamak için, bağlanan mantıksal devrelerin giriş ceryanların toplamı, birinci mantıksal devrenin çıkış pinindeki ceryandan daha yüksek olmamalıdır.

Bir tümleşik devrenin birkaç diğer tümleşik devrelerle bağlanması resim 1.11'de gösterilmiştir. Resimde iki bağlantılı ambalajda, altı birbiriyle aynı evirici içeren 7407 tümleşik devre verilmiştir (Resim 1.10). Devrenin çıkışında matıksal sıfır olduğu zaman, çıkış ceryanın değeri $1_{çık}$<16τA olacak. Bağlanmış tümleşik devrelerin giriş ceryanların değerleri – 1_{gir}<1,6τA' dir. Ceryanların en yüksek değerlerini kıyaslayarak, bir 7404 tümleşik devrenin çıkışında fazlalık yük yapmadan on farklı devre bağlayabileceğimiz sonucuna varabiliriz.



Resim 1.11. Bir TTL devrenin dokuz başka devreyle bağlanması

7404 devrenin çıkışında mantıksal bir (1) olduğu zaman, ceryanın en yüksek değeri $1_{çık}$<0,4τA olacak, bağlanan devrelerin girişinde ise 1_{gir}<0,04τA. Sonuç, çıkışta mantıksal sıfır olduğu durumda gibi benzer şekilde, böyle durumda da bağlanabilir tümleşik devrelerin sayısı en çok 10 olabilir.

Tümleşik devrelerin çıkış pinlerinde taşıyabilecekten fazla yük varsa, o zaman bağlanma arabellek aracılığıyla yapılmalıdır. Arabellek diğer tümleşik devrelerine kıyasen üç kat daha fazla yük taşınabilmesini sağlıyor.

CMOS ailesinden tümleşik devrelerin avantajarı az tüketim ve hızlı çalışmasıdır. Bu tümleşik devreleri +5V ile +15V arası besleme gerilimi kullanıyorlar.

Mantıksal seviye	**Giriş gerilimi**	**Çıkış gerilimi**
0	1,5V	0,5V
1	3,5V	4,5V

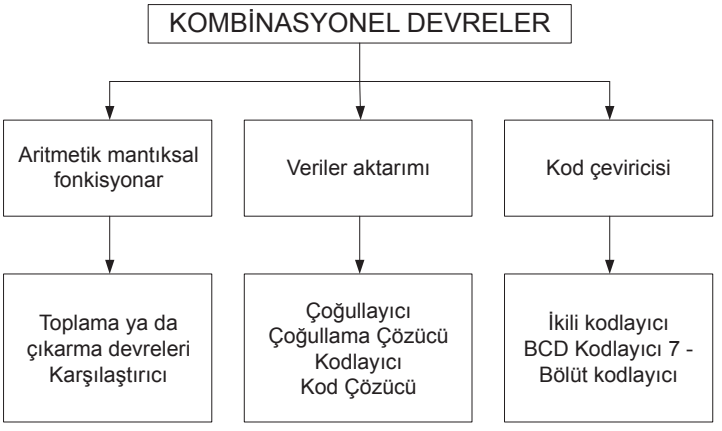
Tablo 1.3. CMOS ailesinden tümleşik devreleri için gerilim değerleri

Tablo 1.3'te CMOS ailesine ait olan tümleşik devrelerinde mantıksal sıfır durumu ve mantıksal birim durumu için gerilimin giriş - çıkış seviyeleri gösteriliyor.

CMOS ailesinden devrelerin mantıksal sıfır ve mantıksal birim seviyeleri için giriş ve çıkış ceryanları, TTL ailesinden devrelere kıyasen çok daha düşüktür ve µA büyüklük boyutundadır. Böyle bir durum CMOS tümleşik devrelerin yapılmış olduğu fetlerin yüksek giriş empendansından kaynaklanıyor.

1.4. Kombinasyonel (Bileşimsel) Devreler

Temel mantıksal geçitlerde daha kompleksli, kombinasyonel olarak adlandırılan devreler oluşabilir. Kombinasyonel devrelerde çıkışlardaki durum sadece girişlerin o anki durumlarına bağlıdır. Sıralı devrelerden farklı olarak, girişlerin ve çıkışların önceki durumu çıkışların verilen andaki duruma hiçbir etkisi yoktur.



Resim 1.12. Kombinasyonel devrelerin ayırımı

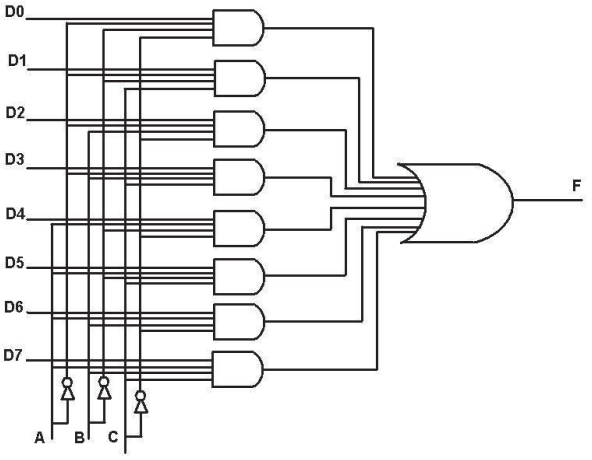
Resim 1.12'de kullanışına bağlı olarak kombinasyonel devrelerin ayırımı verilmiştir. Tüm bu devirler devamdaki metinde açıklanacaktır.

1.4.1. Çoğullayıcı – Çoğullama Çözücü

Çoğullayıcı (Giriş yol seçici) 2^n veri girişi, bir veri çıkışı ve girişlerde birini seçmeyi sağlayan n kontrol (adres) girişi oluşturan devredir. Seçilen giriş tek çıkışla bağlanıyor.

Resim 1.13'te 8 veri girişi, 3 kontrol (adres) girişi ve bir veri girişi içeren çoğullayıcının modeli verilmiştir. Tablo 1.4 onun doğruluk tablosudur. Sekiz VE devirden hepsi aslında kullanıcı verisi olan D'nin geçirilmesi için geçit tanımlıyor.

VE geçidinin açık olması için ikinci, üçüncü ve dördüncü girişte gelen bitlerin birli olması gerekiyor. Bu girişlerden herhangisi sıfır olursa VE devresi D verisini sıfırla çarpacak ve VE devrenin çıkışında sıfır elde edeceğiz.



Resim 1.13. Çoğullayıcının yapısı

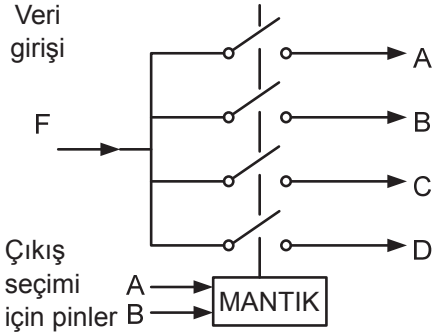
Adresleme			Giriş seçimi
A	B	C	
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

Tablo 1.4. Çoğullayıcı için doğruluk tablosu

Her VE geçidin kendine ait tek bir ABC adres kombinasyonu vardır. Aynı zamanda iki ya da fazla VE geçidin açık olması meydana gelemez. YADA devresi çıkışta tüm sekiz VE geçidinin çıkışlarını topluyor ve toplamı tek veri çıkışına aktarıyor. YADA devresinin çıkışında yedi sıfırı ve bir veri biti D olacak. YADA geçidine hangi veri biti gelecek ve tek çıkışa doğru devam edeceği A, B ve C bitlerinden oluşmuş adres kombinasyonuna bağlıdır

Çoğullama çözücü çoğullayıcıdan ters işlevi olan devredir, yani bir giriş ve fazla çıkışı olan devredir. Hangi çıkışın tek olan girişle bağlanacağı adres bitlerinden oluşmuş kombinasyona bağlıdır. Bu resim 1.14'te göstrilmiştir.

Örnek olarak alınan Çoğullama çözücünün bir veri girişi, iki kontrol (adres) girişi ve dört veri çıkışı vardır. AB giriş adres kodu hangi anahtarların kapalı olacağına karar veriyor ve hangi veri çıkışının tek veri girişiyle aynı değeri olacağına karar veriyor. 1.5 doğruluk tablosunda her dört veri çıkışı için adres kodları verilmiştir.



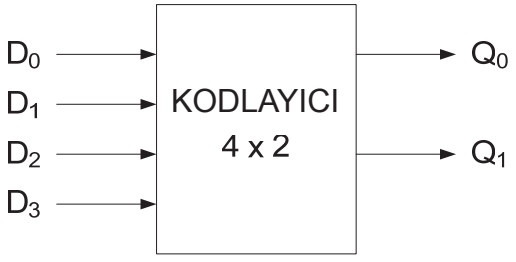
Resim 1.14. Çoğullama çözücünün işlevsel modeli

Adresleme		Giriş seçimi
A	B	
0	0	A
0	1	B
1	0	C
1	1	D

Tablo 1.5. Çoğullama çözünün doğruluk tablosu

1.4.2. Kodlayıcı ve Kod Çözücü

Birçok veri girişinden bir tanesinin seçen ve seçtiği veri girişini tek çıkışa gönderen çoğullayıcıdan farklı olarak, **kodlayıcı** tüm veri girişlerini aynı zamanda dikkate alarak onları tek ikili çıkış koduna dönüştürüyor. Resim 1.15'te dört girişi ve iki çıkışı olan kodlayıcının blok modeli ve onun doğruluk tablosu 1.6 gösterilmiştir.



Resim 1.15. Kodlayıcının blok modeli

Girişler				Çıkışlar	
D_3	D_2	D_1	D_0	Q_1	Q_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

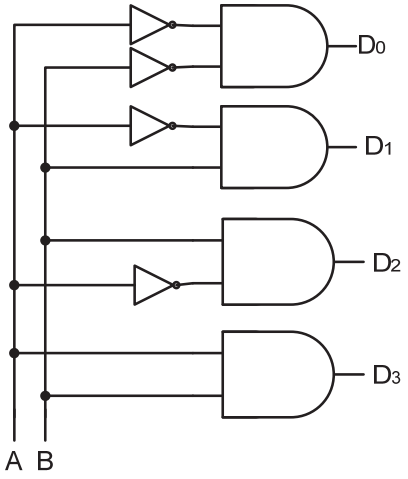
Tablo 1.6.Kodlayıcı için doğruluk tablosu

Doğruluk tablosundan, her ikili çıkış kombinasyonu için sadece bir girişin aktıf olmasını gerektiğini görebiliriz. Burada ortaya çıkan sorun, iki ya da fazla giriş yüksek seviyede olduğu zaman kodlayıcının yanlış çıkış kodun yaratmasından geliyor. Örneğin, eğer D_1 ve D_2 girişleri yüksek seviyedeyse, kodlayıcı ne 01 ne de 10 kodunu oluşturmayacak, onun yerine yanlış çıkış kodu olan 11 kodunu oluşturacak. Bunun çözümü girişlerin farklı önceliği olmaktır. Böylece girişte fazla birler olunca, o zaman çıkış en yüksek önceliği olan aktif girişe uyumludur. Tablo 1.7, resim 1.15'te verilen aynı kodlayıcının doğruluk tablosudur, ancak D_3 girişinin en yüksek öncelliği, D_0 girişinin ise en düşük öncelliği vardır. 1.7 doğruluk tablosundan görüyoruz ki eğer en yüksek öncelliği olan giriş aktif ise kodlayıcı daha alçak öncellikli girişleri dikkate almayacak, daha doğrusu onların durumu önemsizdir ve X ile işaretlenmiştir.

girişler				çıkışlar	
D_3	D_2	D_1	D_0	Q_1	Q_0
0	0	0	1	0	0
0	0	1	x	0	1
0	1	x	x	1	0
1	x	x	x	1	1

Tablo1.7.En önemli girişin öncellikli kodlayıcının doğruluk tablosu

Kod Çözücünün kodlayıcıdan ters işlevi var. İkili giriş koduna bağlı olarak bir çıkış seçiyor. Çıkışların sayısı girişlerin sayısına bağlıdır. Kod çözücünün girişine n - bitli sayı gelirse, o zaman kod çözücüsünle toplam 2^n çıkıştan sadece bir çıkış seçiliyor. Resim 1.16'da iki girişli ve dört çıkışlı kod çözücü gösterilmiştir ve onun doğruluk tablosu 1.8 verilmiştir. Bu kod çözücüsü dört bellek yonganın seçilmesi için kullanılabilir.



Resim 1.16. Kod çözücü için mantıksal diyagram

girişler		çıkışlar			
A	B	D_3	D_2	D_1	D_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Tablo 1.8. İki girişli ve dört çıkışlı kod çözücünün doğruluk tablosu

Kod çözücünün girişinde 00 kombinasyonu gelirse o zaman birinci bellek yongası seçilecek, eğer giriş kombinasyon 01 ise, o zaman ikinci bellek yongası seçilecek vs.

1.4.3. Aritmetik Toplama Devresi

Aritmetik toplama devresi olmayan bilgisayar düşünülemez. YADA devrenin gerçekleştirdiği mantıksal toplamada, 1+1 eşittir 1'e, aritmetik toplamada ise 1+1 sıfıra eşittir ve daha az değerli bitten daha değerli bite, daha doğrusu daha alçak bitten daha yüksek bite 1 akatarımı vardır. Bunu örnek 1.11 ile göstereceğiz.

Örnek 1.11: İki dört bitli sayı verilmiştir A=0101, B=1100. Önce mantıksal, ondan sonra aritmetik toplamayı yapacağız.

Resim 1.17'de gösterilmiş mantıksal toplamada, yan yana yer alan bitler birbirine etkilemiyor. Her pozisyondan bitler ayrı olarak toplanıyor ve bu işlemde resim 1.4'te verilen YADA devresi için doğruluk tablosu kullanılıyor.

```
A      0101
B      1100
Toplam 1101
```

Mantıksal toplama

Resim 1.17. Mantıksal toplama örneği

Aritmetik toplamada aktarma vardır. Resim 1.18 aritmetik toplama işlemi gösterilmiş, tablo 1.9'da ise bu tür toplamanın gerçekleştiği doğruluk tablosu verilmiştir.

```
Aktarma  11
A      0101
B      1100
Toplam 10001
```

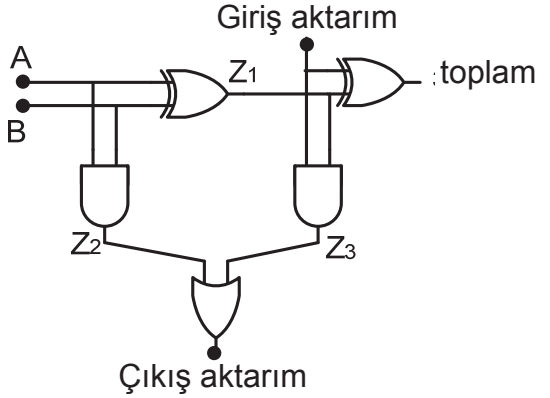
Aritmetik toplama

Resim 1.18 Aritmetik toplama örneği

Bit 1	Bit 2	toplam	aktarma
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Tablo 1.9 Aritmetik toplama için doğruluk tablosu

Üçüncü pozisyonda yer alan bitlerin toplamı sırasında, 1+1 sonucu olarak 0 elde ediyoruz, ancak bu bitlarin toplamında bit 1'in üçüncü poziyondan dördüncü pozisyona aktarması var. Dördüncü pozisyondaki bitlerin toplamı sırasında şunu elde ediyoruz: 0(A'dan) + 1(B'den) + 1(aktarmadan)=0. Toplam 0 ve dördüncü pozisyondan daha yüksek olan beşinci pozisyona aktarma 1 elde ediyoruz.



Resim 1.19. Aktarmalı aritmetik toplamanın devresi

Resim 1.19'de aritmetik toplama devresi verilmiştir. Bu devirin D - YADA, YADA ve VE temel mantıksal devrelerden oluşmuş olduğunu görebiliyoruz. Daha iyi görünürlük için iç bitlerini ya da ara sonuçlarını hesaplamayan denklemler verilmiştir. Tablo 1.10, resim 1.19'da gösterilmiş aritmetik toplama devrenin detaylı doğruluk tablosudur.

Girişler			İç bitler			Çıkışlar	
A	B	$P_{giriş}$	$Z_1=A \oplus B$	$Z_2=A \wedge B$	$Z_3=Z_1 \wedge P_{giriş}$	Toplam= $P_{giriş} \oplus Z_1$	$P_{çıkış}=Z_1 \wedge Z_2$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	1	0	0	1	0
0	1	1	1	0	0	0	1
1	0	0	1	0	0	1	0
1	0	1	1	0	0	0	1
1	1	0	0	1	0	0	1
1	1	1	0	0	1	1	1

Tablo 1.10. İki bitin aktarmayla aritmetik toplama devre için doğruluk tablosu

Resim 1.19'da verilmiş devre sadece bir bitli sayılar topluyor. Eğer dört bitli sayıları toplamak istersek, örnekte olduğu gibi, o zaman yan yana dört böyle toplayıcının bağlanması gerekiyor ve bağlanma o şekilde olacak ki aktarma için çıkış bit (carry out) sol taraftan sıradaki toplayıcıya aktarılıyor, aktramanın girişi için bit (carry in) sağ taraftan önceki toplayıcıdan geliyor

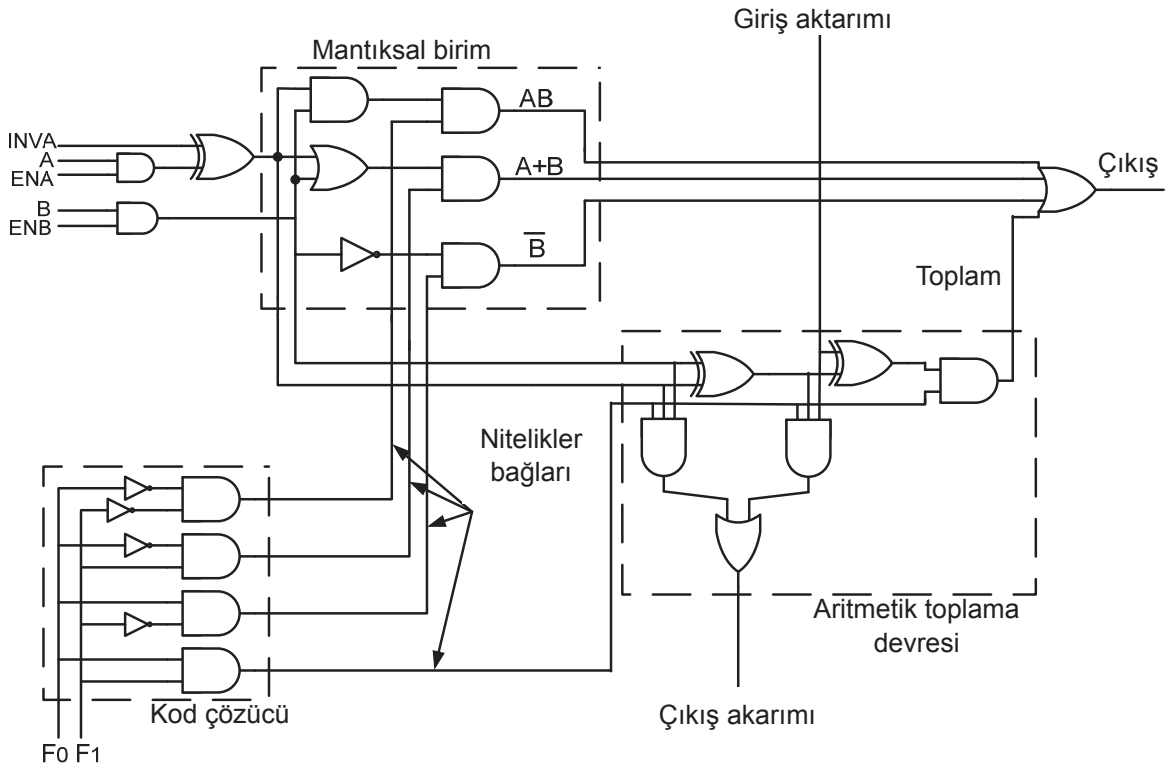
1.4.4. Bir bitli Aritmetik - Mantık Birimi

Aritmetik - mantıks birimi (Aritmetic Logical Unit) her mikroişlemcide var olan bölümdür. Birimin adı gösterildiği gibi, bu birim aritmetik - mantıksal işlemler gerçekleştiriyor.

Donanım açısından sadece dört işlem: mantıksal çarpma, mantıksal toplama, evirici ve aritmetik toplama işlemlerini yapan basit bir aritmetik - mantık biriminin mantıksal diyagramıyla tanışacağız. Tüm diğer basit ve kompleksli işlemler bu dört işlemden yazılımsal gerçekleşmeleridir.

Resim 1.20'nin üst sol bölümünde VE - devresi, YADA - devresi ve evirici bulunuyor. Alt sağ bölümde önceden gördüğümüz artimetik toplama devresi ya da aritmetik birim bulunuyor. Resim 1.20'nin alt sol bölümünde iki çıkışlı F_0 ve F_1 kod çözücüsü yer alıyor. AB, A+B (mantıksal), olumsuz B ya da A+B (aritmetiksel) işlemlerinde hangisinin gerçekleşeceği, F_0 ve F_1 bitlerine bağlıdır.

F_0F_X	işlem
0 0	A·B
0 1	A+B (mantıksal toplama)
1 0	Olumsuz B
1 1	A+B (aritmetik toplama)



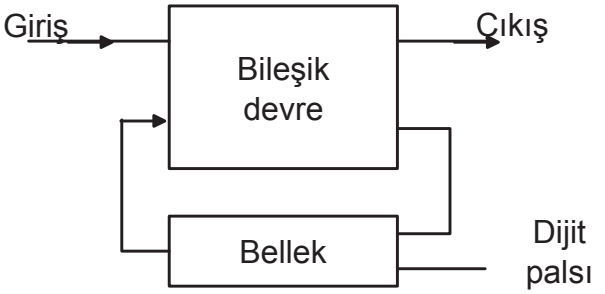
Resim 1.20. Bir bitli aritmetik mantıksal birimin modeli

Hesaplamak için tüm dört devrenin çıkışında (VE devresi, YADA devresi, evirici ve toplama devresi) birer VE geçidi bulunuyor. Bu VE geçitlerin herbirinde ikişer girişi var.

1. Birinci giriş hesaplama devresinden gelen sonuçtur.
2. İkinci giriş kod çözücüden gelen sinyaldır. Bu sinyal bir ise VE geçidi, elde edilen sonucu çıkış YADA geçidine geçirmek için açık olacaktır. Kod çözücünün sinyalı sıfır ise, o zaman VE geçidi kapalıdır. Çıkış YADA devrenin üç girişinde üç sıfır olacak ve kalan bir bit hesaplanan sonuca uygun olacak.

1.5. Ardaşıl Devreler

Bileşik devrelerden farklı olarak, ardaşıl devrelerde çıkışların durumu girişlerin sadece verilen andaki durumuna değil, onların önceki durumlarına da bağlıdır. Ardaşıl devreler girişlerin durumunu "hafıza" ettiklerinden dolayı onlar aynı zamanda bir bellek modül türü de tanımlıyor. Ardaşıl devrelerin yapısı ve onları giriş - çıkış sinyalleri resim 1.21'de verilmiştir.



Resim 1.21. Ardaşıl devrelerin çalışma düzeni

Ardaşıl devrelerde, çıkışların durumuna zamanın özel fizik büyüklüğü olarak büyük etkisi olduğunu söyleyebiliriz. Ondan ardaşıl terimi de kaynaklanıyor. Devrede tüm değişiklikler art arda gerçekleşiyor ve birbirine bağlıdır. Çıkış sinyalın sıradaki zaman biriminde Q_{t+1} durumunun belirlenmesi için girişin durumunun dışında, çıkışın önceki zaman biriminde Q_t durumun da bilinmesi gerekiyor.

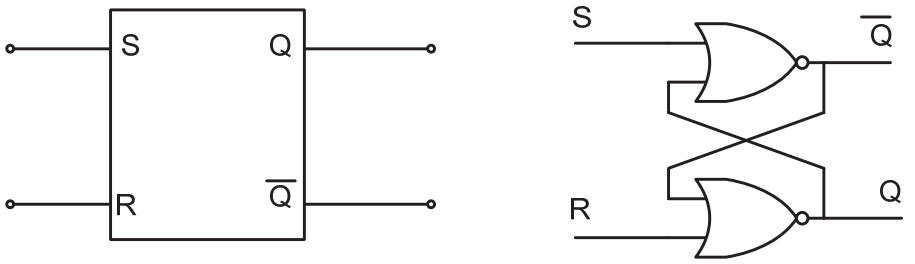
Ardaşıl devreler düzenli (senkron) ya da düzensiz (asenkron) olabilir. Düzensiz devrelerde çıkışların durumu sadece veri girişlerin değişmesine bağlıdır ve bu değişiklikler rastgeledir. Düzenli devrelerde pals sinyalın değeri veri girişin değişmesi veri çıkışın da değişmesine yol açan koşuldur. Giriş sinyallerin etkisi altında çıkış sinyallerin değişmesine izin veren sinyallere aktivatör ya da tetik denir (İngilizce - trigger). Bazı ardaşıl devrelerde tetiğin meydana gelmesi için dijit palsının ya da aktivatörün yüksek seviyede olması gerekiyor, bazılarında ise dijit palsın yükselen ya da azalan kenarı olması gerekiyor.

Ardaşıl devreler flip floplar, multivibratörler, yazmaçlar ve sayaçlardır. Bu ardaşıl devrelerin kombinasyonuyla daha kompleksli bellek yongaları elde ediliyor.

1.5.1. Flip - Flop

Flip flop bir bitlik veri korunabilen en küçük bellek modülüdür. Birkaç flip flop türü vardır: SR, JK, D ve T. Flip flopların çalışması tablolu, analitik ve grafiksel analize edilebiliyor.

SR flip flopu yapı açısından en basittir. Onun iki girişi var: ayarlama girişi S ve yeniden başlatma girişi R. Genelde iki çıkışı var Q ve $\overline{Q}$, ikinci çıkış birinci çıkışın birinci tümleyicisidir. Flip floplar OVE ya da OYA geçitlerden yapılabilir. Resim 1.22, OYA geçitlerden yapılan SR flip flopun sembolünü ve onun mantıksal modelini gösteriyor. Modelden pozitif dönüş bağı olduğu görünüyor, yani çıkışlar girişe dönüyor.



Resim 1.22. SR flip flopun sembolü ve mantıksal modeli

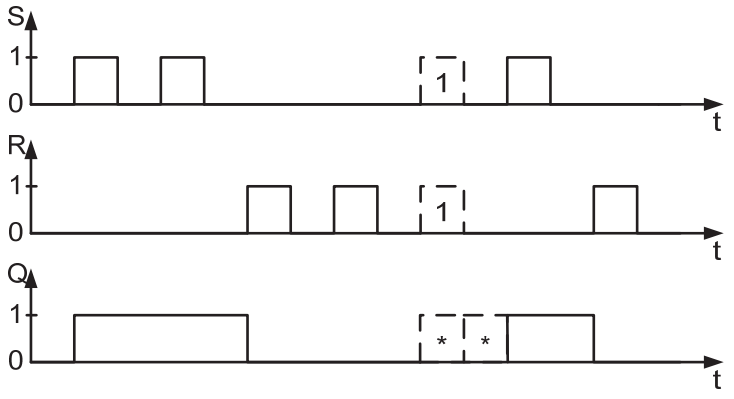
Tablo 1.11. SR flip flopun doğruluk tablosudur. Giriş S'te mantıksal 1 meydana gelince, Q çıkışı belirleniyor ve yüksek seviyede oluyor. R girişinde mantıksal bir meydana geldiğinde, Q çıkışı yeniden sıfırlanıyor ve alçak seviyede olacak. $\overline{Q}$ çıkışı her zaman Q çıkışının seviyesinde ters seviyededir. S=1 ve R=1 durumu kurallar dışıdır çünkü böyle bir durumda iki çıkış alçak seviyededir. İki giriş sıfır olunca çıkışların durumu değişmiyor.

S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	*

*= kural dışı durum

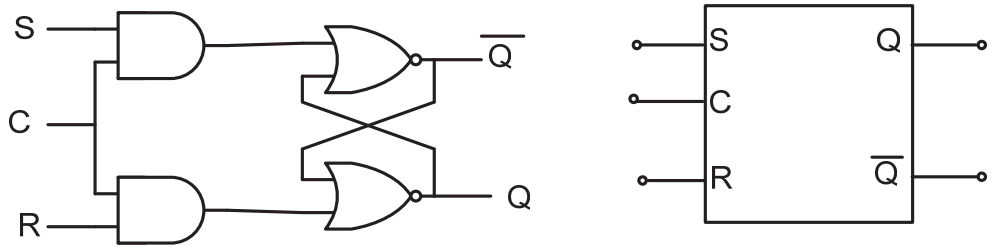
Tablo 1.11. SR flip flopun doğruluk tablosu

SR flip flopun zamanlama diyagramı resim 1.23'te gösterilmiştir



Resim 1.23. SR flip flopun zamanlama diyagramı

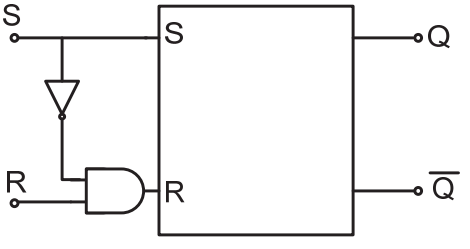
Resim 1.24'te düzenli SR flip flopu verilmiştir. Bu flip flopun üç girişi vardır: S, R ve dijit palsı C (clock).



Resim 1.24. Düzenli SR flip flopun mantıksal modeli ve sembolü

Dijit palsı iki V geçidine gönderiliyor ve, S ve R sinyallerin girmesi için VE geçidinin ne zaman açılacağına karar veriyor. Dijit palsı alçak seviyede olduğu zaman, iki çıkış değişmiyor, dijit palsı yüksek seviyede olduğu zaman ise düzenli flip flop düzensiz flip flop gibi davranıyor.

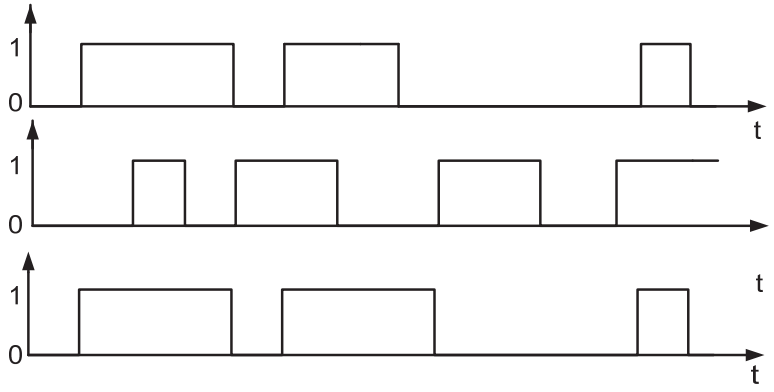
Girişte S=1 ve R=1 istenmeyen durumundan kaçınılma amacıyla bu girişlerden birini diğeriyle kıyaslama yapmakta daha büyük öncellik verilebilir. Bunu SR flip flopun girişine ek mantıksal devre katarak gerçekleştirebiliyoruz. Girişlerden hangisine öncellik verileceği sorusu ortaya çıkıyor. Resim 1.25. ayarlama girişine S verilen öncellikli SR flip flopun



sembolu verilmiştir.

Resim 1.25. S öncellikli SR flip flopun sembolü

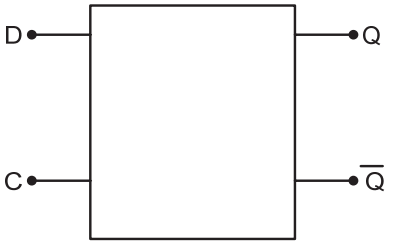
S girişine mantıksal birim gelirse, o zaman evirici aracılığıyla birim sıfıra dönüşecek. Bu sıfır yeniden başlatma girişini önemsiz yapacak çünkü R sinyalı sıfırla çarpılarak sıfır olur. Resim 1.26'da S girişin daha yüksek öncellikli SR flip flopun zamanlama diyagramı gösteriliyor.



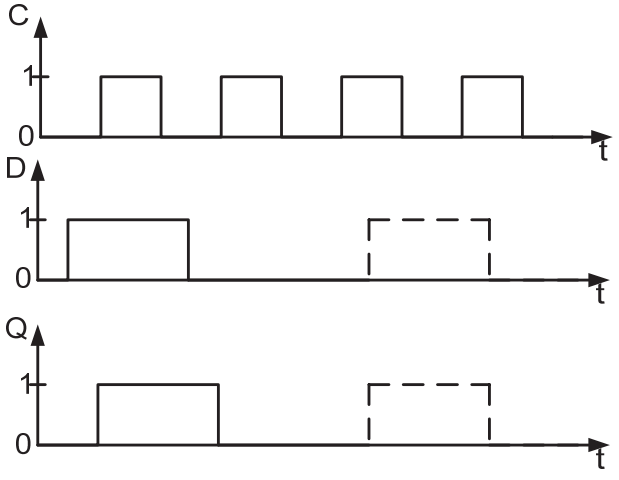
Resim 1.26.S girişinin daha yüksek öncellikli SR flip flopun zamanlama diyagramı

D flip flop çalışma açısından en basit devrelerden biridir. Onlarda girişte meydana gelen değişiklikler doğrudan çıkışa yansıyor. D flip flopu düzenli ve düzensiz olabilir.

Resim 1.27 düzenli D flip flopun sembolünü gösteriyor. Dijit palsı bire eşit olduğu zaman çıkış girişe eşittir Q=D, dijit palsı ise sıfıra eşit olunca çıkış palsın birden sıfıra düşmeden önce son hafıza edilmiş duruma eşit olacak $Q_{t+1}=Q_{t.}$



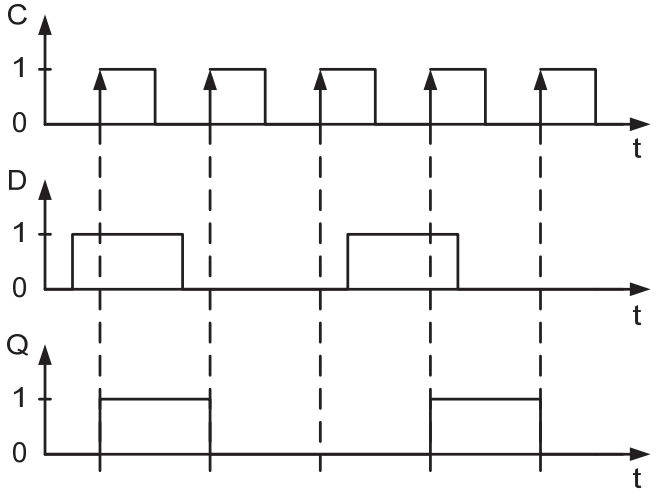
Resim 1.27. Düzenli D flip flopun sembolü



Resim 1.28. Düzenli D flip flopun zamanlama diyagramı

Resim 1.28'de düzenli D flip flopun zamanlama diyagramı gösteriliyor. Pals bir olduğu sürece, çıkış girişin durumunu takip ediyor.

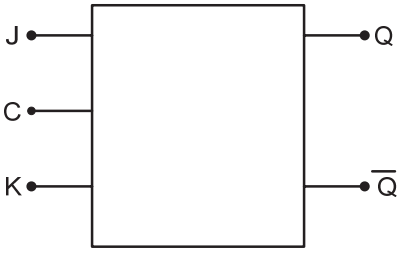
Ancak bazı flip floplarda, değişikliğin meydana gelmesi için dijit palsının yüksek seviyesi gerekmezse, palsın yükselme ya da azalma kenarı olması gerekiyor. Resim 1.29'da tetik olarak dijit palsın yükselen kenarı olan D flip flopun zamanlama diyagramı verilmiştir. Resimden dijit palsın yükselen kenarı olduğu anda çıkışın girişe eşit olduğu sonuca varabiliriz. Çıkış bu durumda bir sonraki yükselen kenara kadar kalacaktır. Dijit palsında birlinin kaldığı süre içinde giriş sinyali değişebilir, ancak çıkışın durumu değişmeyecek.



Resim 1.29. Tetik olarak dijit palsın yükselen kenarı olan D flip flopun zamanlama diyagramı

JK flip flopları en çok kullanılan flip floplardan biridir. JK flip flop yardımıyla SR flip flopların eksikliğine çözüm sağlanmıştır, daha doğrusu kural dışı olan S=1 ve R=1 durumu kaçınılıyor. JK flip flopların hemen hepsinde tetik dijit palsın yükselen ya da azalan kenarı oluyor, bazan da iki kenar aynı zamanda tetik oluyor.

J girişi flip flopun ayarlanma girişidir, K girişi ise yeniden başlatma girişidir.Her iki giriş aktif olduğu zaman, flip flopun çıkışında mantıksal seviyesi değişiyor. Yeniden başlatılmışsa o zaman aynısı ayarlanıyor, ayarlanmış ise yeniden başlatılıyor. Devamda JK flip flopun sembolu ve onun doğruluk tablosu verilmiştir.

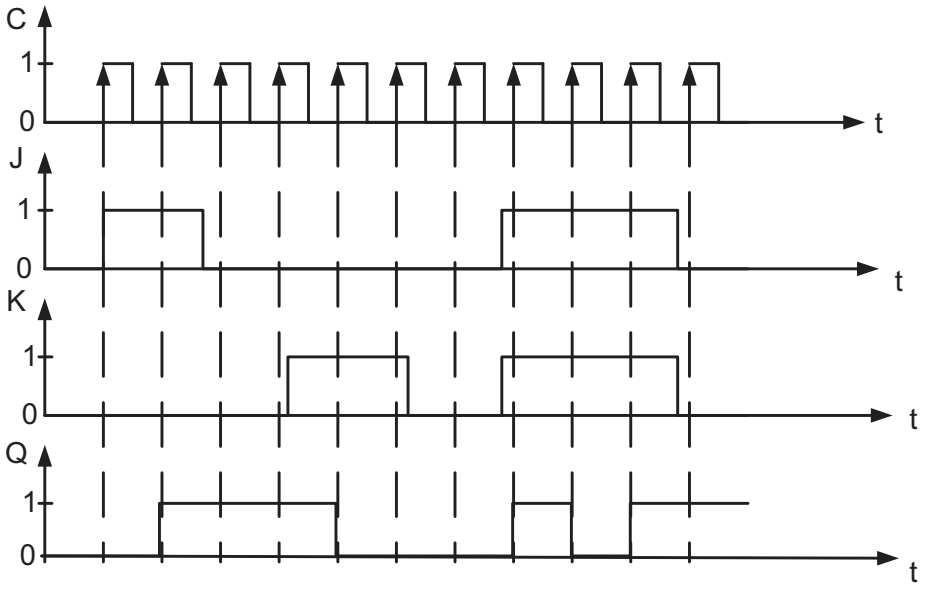


Resim 1.30. JK flip flopun sembolu

J	K	Q_{t+1}
0	0	Q_t
0	1	1
1	0	0
1	1	$\overline{Q}_t$

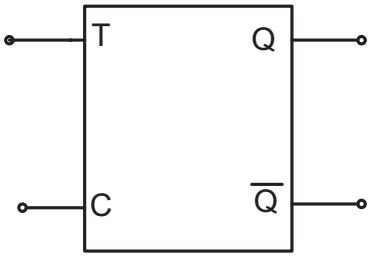
Tablo 1.12. JK flip flopun doğruluk tabloso

Resim 1.31'de tetik olarak kullanılan yükselen kenarlı JK flip flopun zamanlama diyagramı verilmiştir.



Resim 1.31. tetik olarak kullanılan yükselen kenarlı JK flip flopun zamanlama diyagramı.

J ve K girişleri kısa bağlanırsa **T flip flopu** elde ediliyor. Resim 1.32'de düzenli T flip flopun sembolü gösterilmiştir, tablo 1.13 ise onun doğruluk tablosudur.



Resim 1.32. T flip flopun sembolü

C	T	Q_{t+1}
0	0	Q_t
0	1	Q_t
1	0	Q_t
1	1	$\overline{Q}_t$

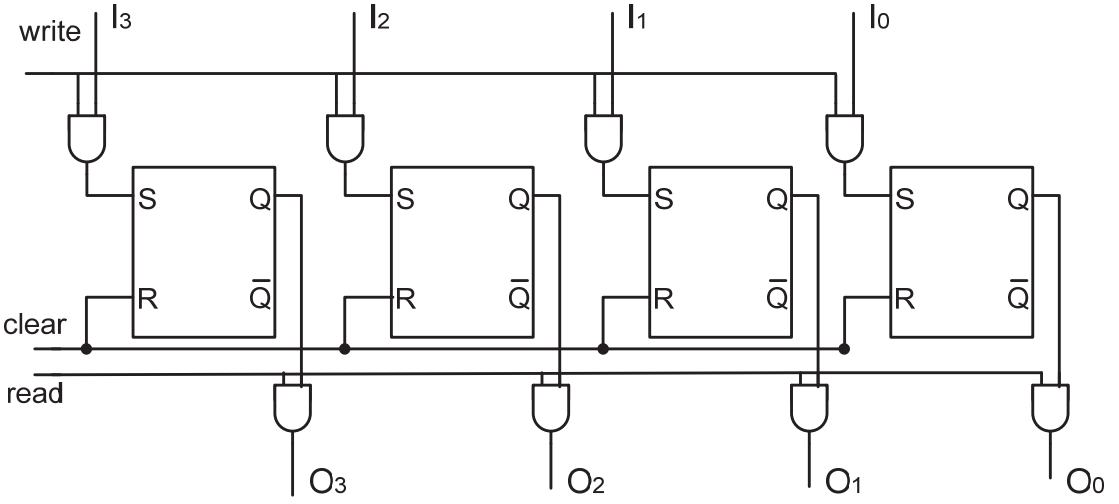
Tablo 1.13. T flip flopun doğruluk tablosu

Dijit palsı düşük seviyede olunca, flip flopun durumu değişmiyor. Flip flopun durumu dijit palsı yüksek seviyede olunca da değişmiyor, ancak T girişi mantıksal sıfıra eşit olunca değişyor. T flip flopun durumunun değişmesi için dijit palsın yüksek seviyede olması ve T girişinde mantıksal birim olması gerekiyor. Durumun değişmesi Q çıkışın önceki durumun tümleşmesiyle gerçekleşiyor.

1.5.2. Yazmaçlar

Yazmaçlar flip floplardan daha büyük bellek modülleridir. Bir yazmaçta kaç bit sığar, kaç flip floptan oluştuğuna bağlıdır (her bit için birer flip flop). Farklı yazmaçlar mevcuttur, ancak en çok kullanılan yazmaçlar sabit ve ötelemeli yazmaçlardır.

Sabit yazmaçlar birkaç bitlik büyüklüğünde verilerin geçici olarak saklanması için kullanılıyor. Resim 1.33'te paralel girişli ve paralel çıkışlı dört bitli sabit yazmacı olarak gösterilmiştir. Üç kontrol sinyalı var. Clear SR flip floplarını yeniden başlatmak (sıfırlamak) için kullanılıyor. Write bitlerin yazılması, Read ise önceden yazılmış bitlerin okunması için kullanılıyor. Verilerin yazılması başlamadan önce, Clear sinyalını yüksek seviyeye getirerek flip floplar sıfırlanıyor. Sıfırlamadan sonra Clear sinyali yeniden alçak seviyeye ayarlanıyor.



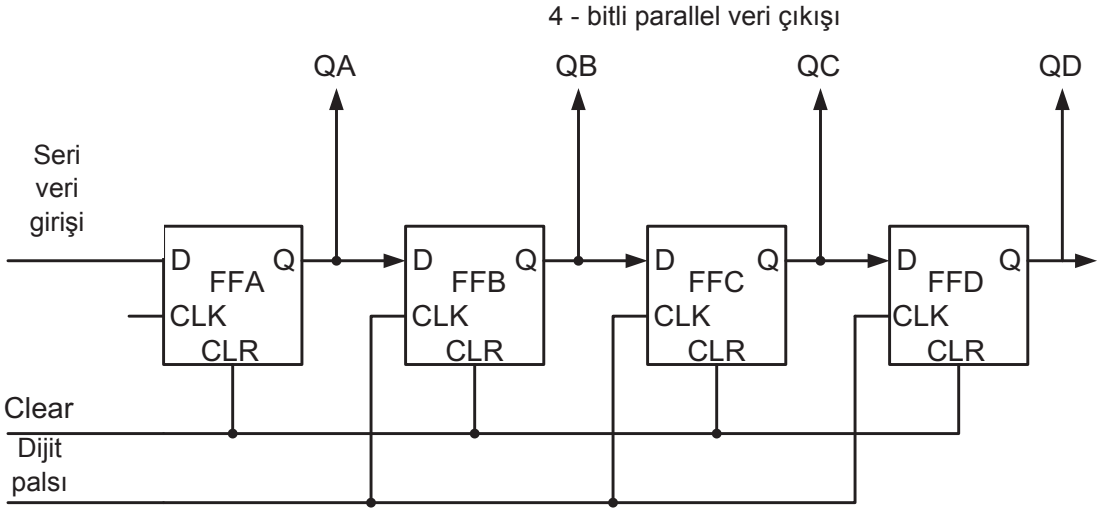
Resim 1.33. Sabit yazmacın mantıksal modeli

Veriler yazdığımız zaman, Write sinyali yüksek seviyede olmalıdır. Bu şekilde her flip flopun ayarlama girişleriyle bağlı olan VE geçitleri açılıyor. I_0'dan I_3'e kadar giriş sinyalleri (İnput) flip flopların çıkışlarına aktarılıyor. Giriş sinyali I mantıksal sıfır ise, flip flop ayarlanmayacak ve onun çıkışında durum, sıfırlamadan sonra olduğu gibi mantıksal sıfır kalacak. I sinyalı mantıksal bire eşit olursa, o zaman flip flop ayarlanacak ve Q=1 olacak.

Okuduğumuz zaman Read sinyalin aktif edilmesi gerekiyor. Eğer Read=1 her flip flopun çıkışında yer alan çıkış VE geçitleri açılıyor. Bu geçitlerin açılmasıyla sinyaller flip flopların çıkışlarından, O_0, O_1, O_2 ve O_3 çıkışlarına (Output) aktarılıyor.

Ötelemeli yazmaçlar ardaşıl şekilde bağlanmış flip floplardan oluşuyor ve bu arada her sonraki flip flopun girişine önceki flip flopun çıkış sinyali geliyor. Bu şekilde birinci flip flopun girişine gelen dizinin biti, bir flip floptan başka flip flopa, soldan sağa taşınacak. Dört tür ötelemeli yazmaç vardır: seri girişli ve paralel çıkışlı (Serial In to Parallel Out - SIPO), seri girişli ve seri çıkışlı (Serial in to Serial Out - SISO), paralel girişli ve paralel çıkışlı (Parallel In to Parallel out - PIPO) ve paralel girişli ve seri çıkışlı (Parallel In to Serial Out - PISO).

Resim 1.34'te seri girişli ve parallel çıkışlı ötelemeli yazmaç verilmiştir.



Resim 1.34. Seri girişli ve paralel çıkışlı ötelemeli yazmaçın mantıksal modeli

Her dört flip flop ortak dijit palsı kullanıyor. Baştan tüm flip flopların, FFA'dan FFD'ye kadar sıfırlanmış olduğunu (Clear girişi aracılığıyla) ve tüm veri çıkışların QA'dan QD'ye kadar sıfıra eşit olduklarını tahmin edelim. Eğer birinci dijit palsında, birinci flip flopun FFA girişinde mantıksal bir (1) olarak meydana gelirse, o zaman ayarlamayla QA=1 olacak. İkinci ve her sıradaki dijit palsında birinci flip flopta giriş alçak seviyede olacağını tahmin ediyoruz. İkinci dijit palsında birinci flip flop sıfırlanacak QA=0, ikinci flip flopun çıkışı ve QB mantıksal bir olacak. Mantıksal bir, bir yer için sağa taşınacak. Üçüncü dijit palsı gelince mantıksal bir üçüncü flip flopta olacak. Beşinci dijit palsına kadar mantıksal bir (1) tüm flip flopları geçecek ve tüm flip flopların çıkışında yeniden sıfırlar olacak çünkü birinci dijit palsından sonra yazmaçın seri girişi sabit alçak seviyede kalıyor.

Tablo 1.14'te her dört flip flopun beş dijit palsı süresi içinde içerikleri verilmiştir. Dördüncü palsta çıkışların durumu seri girişinden birer birer bit girilen dört - bitli veriye eşit olacaktır.

Bu demek ki ötelemeli yazmaç seri veri sinyalini paralel veri sinyaline dönüştürmüştür.

Dijit palsın sıra numarası	QA	QB	QC	QD
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	0	0	0	0

Tablo 1.14 Seri girişli ve paralel girişli ötelemeli yazmacın doğruluk tablosu

Seri girişli ve paralel çıkışlı ötelemeli yazmacın tersine, paralel girişli ve seri çıkışlı ötelemeli yazmaç giriş paralel veri kelimesini seri veri sinyaline dönüştürüyor. Bu ötelemeli yazmaçlar, daha hızlı veri aktarım amacıyla çok sayıda giriş bağlarını bir seri bağlantıya çoğullamak için kullanılıyor.

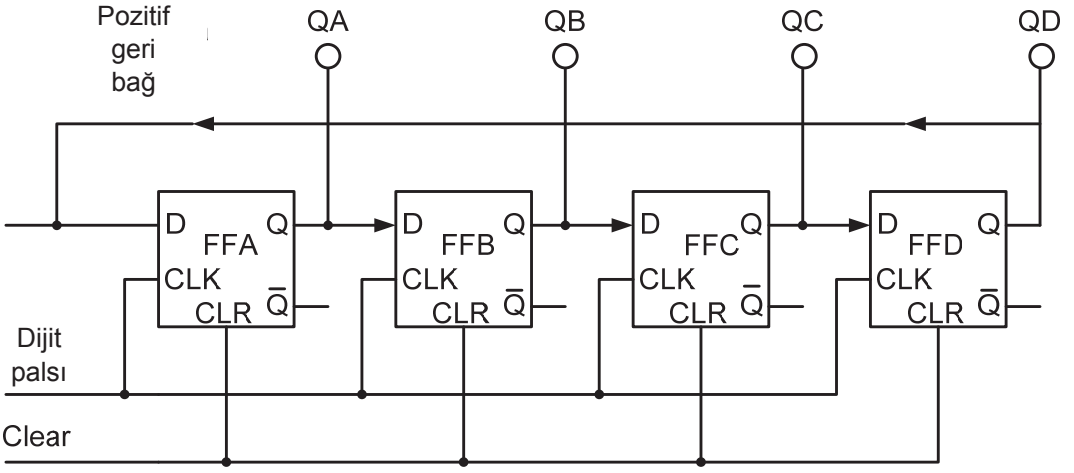
Seri giriş/seri çıkışlı ve paralel girişli/paralel çıkışlı ötelemeli yazmaçlar geçici sürede verilerin korunması için kullanılıyorlar ve geç kalma zamanı ekliyorlar. Giriş ve çıkış sinyalerin arasında zaman farkı yazmaçlarda flip flopların sayısına bağlıdır. Yazmaçlarda flip flopların sayısı genelde 4, 8, 16 vs. oluyor.

1.5.3. Sayaçlar

Ötelemeli yazmaçlar gibi sayaçlar da birkaç flip floptan oluşmuş ardaşıl devrelerdir. Farklı sayaç türleri vardır: asenkron ya da senkron sayaçlar, iliriye sayan sayaçlar ya da geriye sayan sayaçlar ve ikili, onlu ya da on altılı sayaçlar.

Ötelemeli yazmacın çıkışı, girişiyle bağlanırsa (pozitif geri bağ) o zaman **çembersel (dairesel) sayaç** elde ediliyor. Resim 1.35'te dört flip floptan oluşan senkron çembersel sayacın modeli verilmiştir. Çembersel sayaca senkron denir, çünkü onun tüm flip flopları aynı dijit palsını kullanıyor.

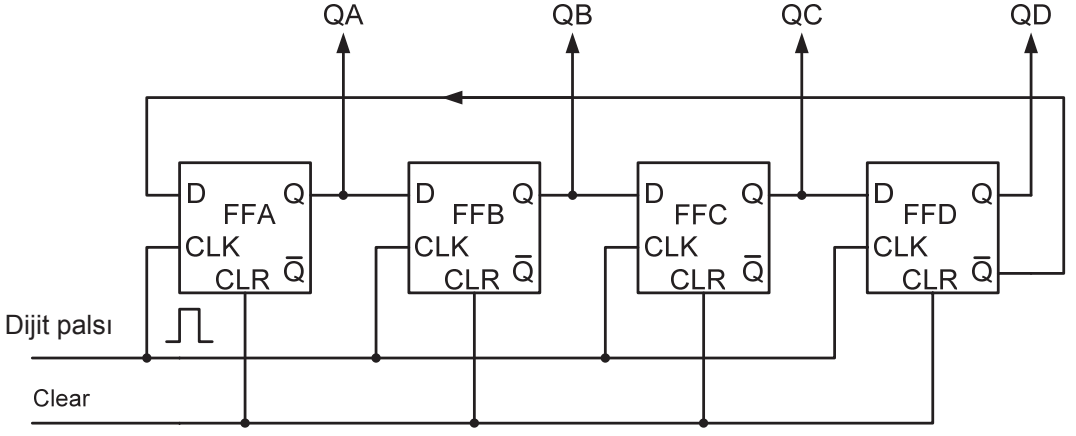
Dört bitli veri dört flip flop arasında durmaksızın dönüyor ve bu her dört dijit palsında tekrarlanıyor. Ancak verinin dönmesi başlamadan önce verini eklenmesi gerekiyor. Bu amaçla sayacı Clear sinyaliyle sıfırlıyoruz.



Resim 1.35. Çembersel sayacın matıksal modeli

Ondan sonra birinci flip flopun girişinde mantıksal bir meydana geliyor, bu dürtü (empuls) sadece bir dijit palsı kadar sürüyor. Bu dürtü aslında çemberli sayacın çalışmaya başlaması için gereken aktivatördür.

Eviricisiz çıkış yerine sayacın girişini evirici çıkışla bağlarsak, **Conson'un çembersel sayacı** olarak adlandırılan bambaşka bir sayaç elde edeceğiz. Onun modeli resim 1.36'da verilmiştir.



Resim 1.36. Conson'un çemberli sayacın mantıksal modeli

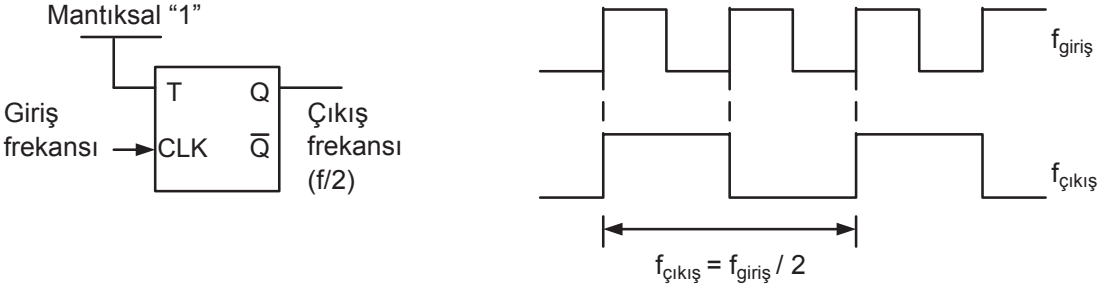
Conson'un sayacın dört yerine sekiz farklı durumu var. Yukarıda verilen örnekte çemberli sayacın durumları şunlardır: 1000, 0100, 0010 ve 0001. Conson'un sayacı önce ileriye doğru sayıyor, ilk dört dijit palsından sonra geriye doğru saymaya başlıyor. Tablo 1.15'te her sekizinci dijit palsından sonra devamlı tekrarlanan sekiz farklı durum verilmiştir.

Çembersel sayacında, yazmaça sıfırlanmadan sonra girilen bir tek idi ve aynısı durmaksızın dönüyor ve bir flip flopta başkasına doğru hareket ediyordu. Conson'un sayacında ilk dört durumda birlerin sayısı devamlı artıyor, çünkü FFD'nin evirici çıkışında mantıksal birli vardır. Bu birli tüm dört flip flopun ayarlamasını yapacak. Devamda evirici çıkış mantıksal sıfır oluyor, flip floplar ise FFA'dan başlayarak FFD'ye kadar sıfırlanmaya başlayacak (son dört durum).

FFA	FFB	FFC	FFD
0	0	0	0
1	0	0	0
1	1	0	0
1	1	1	0
1	1	1	1
0	1	1	1
0	0	1	1
0	0	0	1

Tablo 1.15. Conson'un çembersel sayacın doğruluk tablosu

Sayaçlarda en sıkça kullanılan flip flop T flip flopudur. T flip flopu sıkça **frekans dağıtıcı** olarak kullanılıyor. T girişi yüksek seviyeye bağlanırsa, o zaman eviricisiz çıkışta, giriş dijit palsın frekansından iki misli daha düşük frekanslı dijit palsı elde edeceğiz. Böyle bir frekans dağıtıcısı resim 1.37'de verilmiştir. Aynı resimde giriş dijit palsı ve eviricisiz çıkış Q'da elde edilen pals da gösterilmiştir.

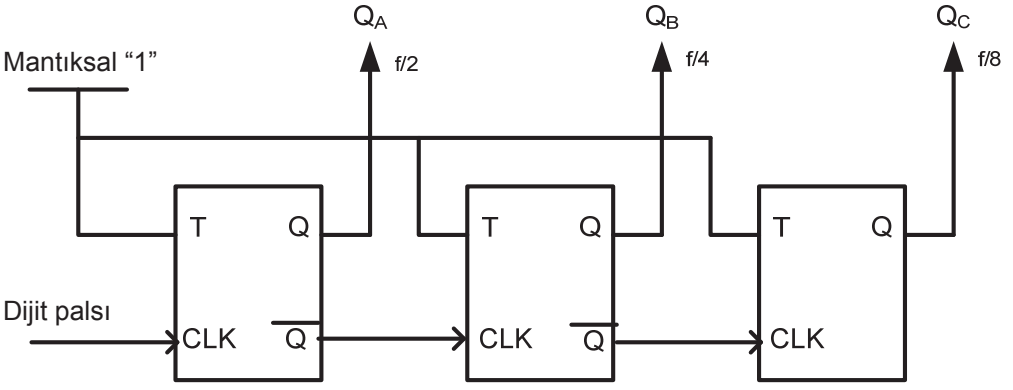


Resim 1.37. Frekans dağıtıcısı olarak T flip flopu

Giriş dijit palsında her yeni yükselen kenarın gelmesiyle çıkış Q'nun değeri değişiyor, daha doğrusu çıkış sinyalin değeri bir imişse sıfır oluyor ve tersi. T flip flopun çıkışında, giriş dijit palsın iki peryodu için bir periyod elde ediyoruz. Seri şeklinde t girişleri yüksek seviyede olan iki T flip flop bağlarsak, o zaman frekans dağıtıcının değeri 4 olacak. Seri şeklinde üç T flip flopu bağlarsak, o zaman çıkış frekansı giriş frekansından sekiz misli daha düşük olacak vs. Frekans dağıtıcının değeri 2^n denklemiyle hesaplanıyor. Denklemde kullanılan n flip flopların sayısına eşittir.

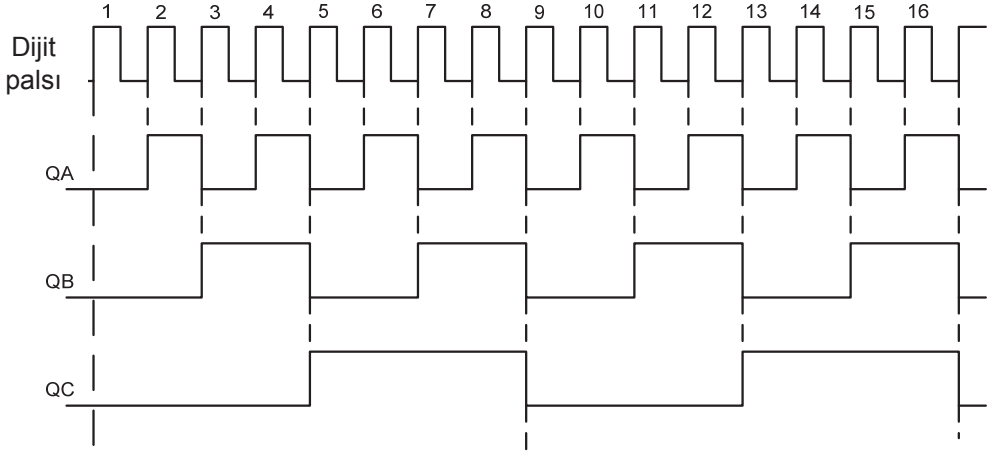
T girişlerin yüksek seviyede ayarlanmış birkaç T flip flopun seri şeklinde bağlanmasıyla **asenkron sayaçlar** elde ediliyor. Onlara asenkron deniyor çünkü onların flip flopları ortak dijit palsı kullanmıyor. Dijit palsı birinci flip flopun girişine getiriliyor, birinci flip flopun evirilmiş Q çıkışının sinyali ikinci T flip flopun dijit palsın girişine taşınıyor, ikinci flip flopun evirilmiş Q çıkışının sinyalı üçüncü flip flopun dijit palsın girişine taşınıyor vs.

Resim 1.38'de ileriye doğru sayan üç bitli asenkron sayacı gösteriliyor. Birinci dijit palsı sırasında tüm flip flopların sıfırlandırılmış olduğunu tahmin ediyoruz. İkinci dijit palsın gelmesiyle birinci flip flop ayarlanıyor. Üçüncü palsın gelmesiyle birinci flip flop sıfırlandırılıyor, ikincisi ise ayarlanıyor. Sıradaki palsla birinci flip flop ikinci kez ayarlanıyor, ikincisi de ayarlanmış kalıyor. Sekizinci dijit palsı sıraya gelince her üç flip flop ayarlanmış olacak. Sekizinci palstan sonra, dokuzuncu dijit palsı ile her üç flip flop sıfırlanacak ve süreç yeniden baştan başlayacak.



Resim 1.38. Asenkron sayacın mantıksal modeli

Sayaçların üç çıkışında durumların değişmesi resim 1.39. olduğu gibi zaman diyagramıyla gösterilebilir.



Resim 1.39. Asenkron sayacın zaman diyagramı

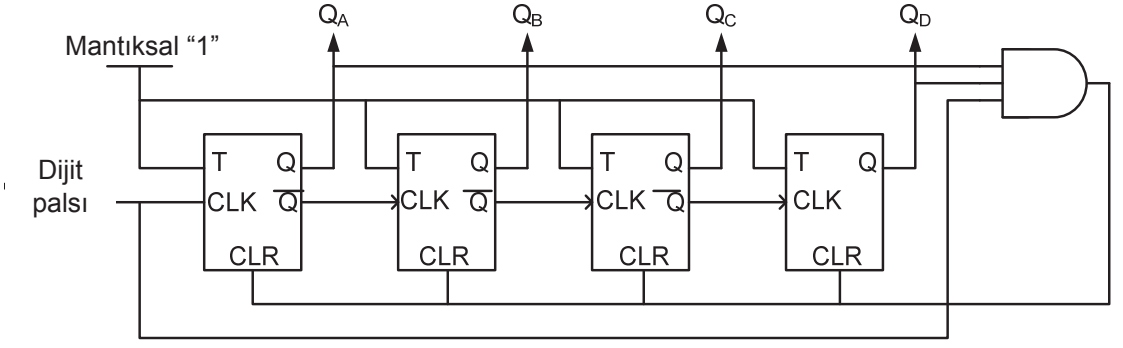
Her dijit palsı için QA, QB ve QC çıkışların durumunu yazarsak, sayacın doğrulu tablosunu elde ediyoruz:

Dijit palsı	Çıkış kombinasyonu			Ondalık değer
	QC	QB	QA	
0	0	0	0	0
1	0	0	1	1
2	0	1	0	2
3	0	1	1	3
4	1	0	0	4
5	1	0	1	5
6	1	1	0	6
7	1	1	1	7

Tablo 1.16. Üç - bitli asenkron sayacın doğruluk tablosu

Doğruluk tablosunda çıkışların durumu dışında, her üç çıkışın beraber verdiği ondalık değeri de verilmiştir. Bu ondalık değer aslında sayacın değeridir. Her yeni dijit palsın gelmesiyle sayaçın değeri bir için artıyor. Bu sayacın sayabildiği en büyük değer yedidir. Ondan sonra sayaç sıfırlandırılıyor ve sayaç tekrar sıfırdan saymaya başlıyor. Üç bitli sayacın 000'dan başlayarak 111'e kadar sekiz farklı durumun olabileceği sonucuna varabiliriz. Herhangi sayacın durumların sayısı 2^n'e eşittir. İfadede kullanılan n'in değeri sayacı oluşturan flip flopların sayısıdır, sayacın sayabileceği en büyük sayı durumların sayısından bir için daha küçüktür.

Resim 1.40'te **asekron onlu sayaç** gösterilmiştir**.**



Resim 1.40. Asenkron onlu sayacın mantıksal modeli

Resim 1.40'ta gösterilmiş sayaç t girişleri yüksek seviyede olan dört T flip floptan oluşmuştur. Ona göre frekans dağıtıcının değeri 2^4=16 olacak. Ancak bu sayaç 0'dan 15'e kadar saymayacak. Dört flip flopun çıkışında değerin durumu 1001=9 olunca, sayaç yeniden başlatılarak yeniden 0'dan saymaya başlayacak. Bu sayaç onlu olarak adlandırılıyor, çünkü on farklı durumu olabilir, 0000'dan başlayarak

1001'e kadar. Bunlar aslında onlu sayı sisteminden rakamların BCD kodlarıdır ve bundan dolayı bu sayaca BCD sayacı da denir.

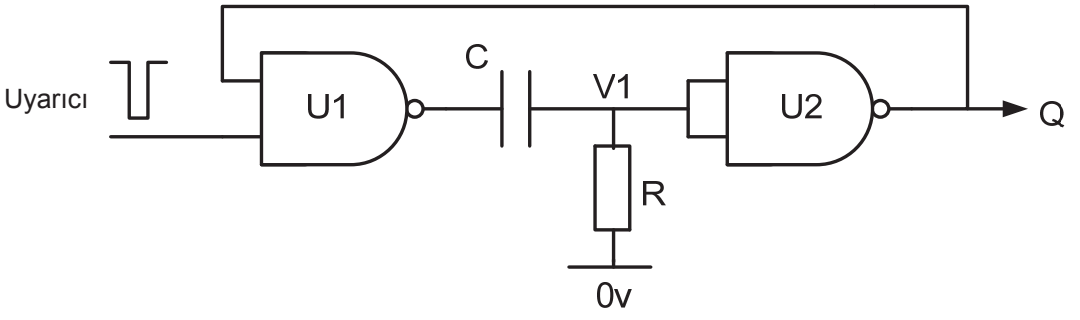
Dokuz değeri elde edilince sayaç resim 1.40'tan VE devrenin yardımıyla yeniden başlatılıyor. Bu VE devrenin üç girişi var: QA, QD ve giriş dijit palsı. Her üç giriş 1'e eşit olunca, VE devresi çıkışında bir veriyor ve sıfırlandırma sinyali (clear) etkinleşiyor.

1.6. Multivibratörler

Flip flopların, yazmaçların, sayaçların ve bellek yongaların çalışması, dijit palsı kullanmadan düşünülemez. Multivibratörler dijit palsı oluşturan ardaşıl devrelerdir. Multivibratörlerlerin genel ayırımı tek kararalı (monostabil) ve kararsız (astabil) olarak bölünüyor. Kararsız multivibratörler peryodik uyarı geçitleri oluşturuyor, tekkararlı multivibratörler ise sadece bir uyarı ya da çıkışında duraklama oluşturuyor.

1.6.1. Tekkararlı Multivibratör

Resim 1.41'de iki OVE devreden, rezistör (direnç elemanı) R, yoğunlaştırıcı (kondansatör) C ve pozitif geri bağdan oluşmuş tek kararlı (sabit) multivibratör gösterilmiştir. Bu multivibratörün sabit mantıksal bir durumu ve bir geçici mantıksal sıfır durumu vardır. Uyarıcı değişmediği halde multivibratörler sabit (dengeli) durumda bulunuyor. Uyarıcının değiştiği zaman multivibratör geçici duruma geçiyor, ancak adı da gösterdiği gibi bu durum kalıcı değil ve belli bir süre sonra kendiliğinden sabit duruma dönüyor.



Resim 1.41. Tek kararlı multivibratör modeli

Sabit durumda R rezistöründen ceryan akmıyor, U2 devre girişinde mantıksal sıfır var, çıkışta ise mantıksal bir vardır.

Kararlı durumda etkinleştirici mantıksal sıfıra eşittir. Bundan dolayı U1 devrenin girişinde iki birli çıkışında sıfır verecek. Sabit durumda kondansatörün iki ucu düşük potansiyel durumunda bulunuyor ve ondan ceryan akmıyor.

Etkinleştiricinin girişinde mantıksal sıfır meydana gelirse, o zaman U1 devrenin çıkışı mantıksal bir (1) olacak. Kondansatörün uçlarında gerilim bir anda değişemediği için U2 devrenin girişi de aynı şekilde mantıksal sıfır oluyor. U2'nin çıkışındaki mantıksal sıfırdan dolayı, etkinleştiricinin girişinde duraklama bittikten sonra da multivibratör bu durumda kalmayacak.

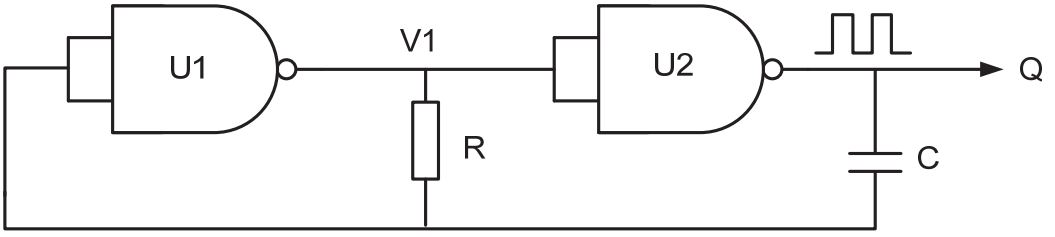
Kondansatör dolduğu sürede ondan ceryan geçiyor, dolduğu zaman ise ceryanın akımı duruyor ve V1 noktasında potansiyel yeniden mantıksal sıfır durumuna düşüyor. Bu durum U2 devrenin çıkışı yenden mantıksal bir'e U1 devrenin çıkışında ise mantıksal sıfır olmasına yol açıyor. Devamda kondansatör, rezistör aracılığıyla boşanmaya başlıyor. Multivibratör de sonunda kendisi sabit durumuna dönüyor.

Geçici durumun süresi ya da multivibratörün çıkışındaki duraklamanın süresi RC devrenin sabit elemanına ve T=0,7 RC denklemine göre hesaplanıyor.

1.6.2. Kararsız Multivibratör

Kararsız multivibratör sabit durumu olmayan serbest salıngaçtır (titreştirici). Onun iki geçici durumu vardır: mantıksal sıfır durumu ve mantıksal bir durumu. Çıkışı devamlı düşük seviyeden yüksek seviyeye ve ters yönde değiştiriyor ve o şekilde, peryodun RC devrenin sabitine bağlı olan dörtgen dürtülerden peryodik geçit oluşturuyor.

Resim 1.42'de OVE geçitlerden oluşmuş kararsız multivibratörün modeli gösterilmiştir. Bu OVE geçitlerin girişleri kısa bağlıdır ve böylece onlar evirici olarak işlev görüyor.



Resim 1.42. Kararsız multivibratör modeli

U2 devrede ikinci OVE çıkışı mantıksal bir'e eşit olursa, o zaman onun girişinde mantıksal sıfır vardır. Kondansatörün bir ucu U2 devrenin çıkışıyla bağlıdır ve mantıksal bir (1) durumundadır, diğer ucu ise R rezistörü aracılığıyla U1 devrenin girişiyle bağlıdır ve mantıksal sıfır durumundadır. Kondansatör dolmaya başlıyor.

Kondansatör dolduğu sürede kondansatör ve rezistör arasındaki noktada potansiyel devamlı olarak azalıyor. Bu nokta birinci U1 OVE devrenin girişiyle bağlıdır. Belli bir anda U1 devrenin girişi mantıksal sıfır olacak, çıkışı ise mantıksal bir durumuyla değişiyor. Böyle bir durum U2 devrenin çıkışında mantıksal birden mantıksal sıfıra değişmesine yol açıyor. Kondansatör U2 devrenin girişi yeniden değişene kadar kondansatör boşanmaya başlıyor ve dönge yeniden başlıyor.

Buradan şu sonuca varabiliriz: kondansatör dolduğu zaman multivibratörün çıkışında mantıksal bir (dürtü) vardır, kondansatör boşandıkça ise çıkışta mantıksal sıfır (duraklama) vardır. Elde edilen dijit palsın peryodu T=2,2RC denklemiyle hesaplanıyor.

1.7. Yarıiletken Bellekler

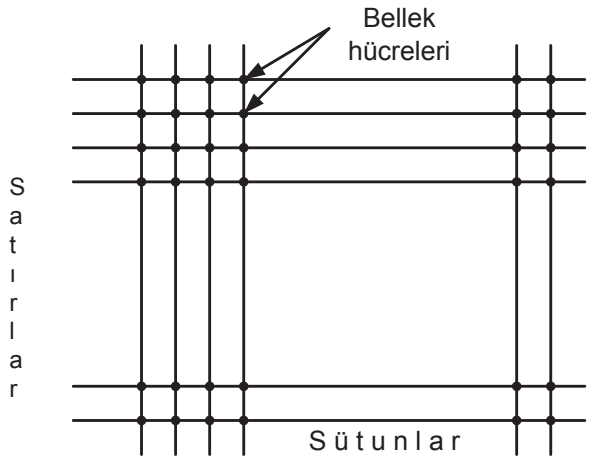
Yapılmış olduğu mazlemelerin türüne bağlı olarak bellekler yarıiletken ve manyetik bellekler olarak ayrılıyor. ROM ve RAM bellekleri yarıiletken belleklerdir. Kendi yapısında RAM ve ROM belleği olmayan bilgisayar düşünülemez. Yarıiletken bellekler tümleşik devre şeklinde yapılıyorlar. ROM ve RAM belleklerin işlevini ve türlerini tanımadan önce bellek yongaların (çiplerin) organizasyonu ve onların daha önemli giriş ve çıkış iğnelerini (pinlerini) inceleyeceğiz.

1.7.1. Bellek Yongaların Organizasyonu

Flip floplar bir bit büyüklüğünde bilgi hafıza eden en küçük bellek birimleridir. Yazmaçlar birkaç flip floptan oluşuyor ve onlar 4, 8, 16 bitli olabilirler. Yazmaçlardan sonra kapasitesi birkaç megabayta kadar yükselen bellek yongaları geliyor.

Bellek yongaları matris şeklinde düzenlenmiştir. Matris resim 1.43'te görüldüğü gibi dik satır ve sütundan oluşmuş yapıdır. Satırların ve sütunların kesiştiği noktalarda bir bitlik veri hafıza edebilen birer bellek hücresi vardır. Satırların sayısı bellek konumların sayısına eşittir, sütunlar sayısı ise bir bellek konumunda kaç bitin hafıza edilebileceğini gösteriyor.

Satırların ve sütunların sayısı bir bellek yonganın adres ve veri pinlerin sayısını tanımlıyor.



Resim 1.43. Bellek yonganın yapısı

Adres pinleri bir bellek yongası içeriğinde birçok bellek konumlarından birini seçmek için kullanılıyorlar. Adres pinlerin sayısı ve bellek konumların sayısı arasındaki bağımlılık şu ilişkiyle verilmiştir:

$$\text{Bellek konumların sayısı} = 2^{\text{adres pinlerin sayısı}}$$

2^{10} üsü 1024'e eşittir ve kısaca 1K (kilo) olarak yazılıyor. 2^{20} üsü 1048576'ya eşittir ve kısaca 1M (mega) olarak adlandırılıyor. 2^{30} üsü 1073741824'e eşittir ve kısaca 1G (giga) olarak yazılıyor. Aynı şekilde üsüleri ayrıştırırsak 1G=1K1K1K ya da 1M=1K1K olduğunu elde ediyoruz. Üsülerin daha basit şekilde hesaplamak için üs kuvvet katsayısını birlere ve onlara ayrıştırabiliriz. Bu şekilde onların üsü öneki veriyor. Bu yöntem örnek 1.12'de göstrilmiştir

Örnek 1.12: Bir bellek yongasında bellek konumların sayısını hesapla eğer adres pinlerin sayısı 17'yse.
Çözüm: $2^{17}=2^7 \cdot 2^{10}=128 \cdot 1024=128K$
Bu bellek yongası 131072 bellek konumu içeriyor ya da kısaca olarak 128 kilo bellek konumu içerdiğini söyleyebiliriz.

En değersiz (en az değeri olan) adres pini A_0'la işaretleniyor, en değerlisi ise A_{n-1} olarak işaretleniyor. Bu işaretleme şeklinde n endeksi tüm adres girişlerin toplam sayısıdır. Örneğin, bellek yongası 10 adres girişi içeriyorsa, o zaman onlar A_0 'dan A_9'a kadar işaretleniyor. Adres pinleri aslında, bellek yongasının içinde yerleştirilmiş

adres kod çözücünün girişleridir. Adres kod çözücüsü adres bitlerin giriş kombinasyonuna göre yongadan bir bellek konumun seçilmesi için kullanılıyor.

Her bellek konumun kendi adresi var ve bu adres aslında adres bitlerinden oluşan tek kombinasyondur. Adres tektir, çünkü aynı bellek yongası çerçevesinde iki ya da fazla konumun aynı adresi olması kabul edilemez.

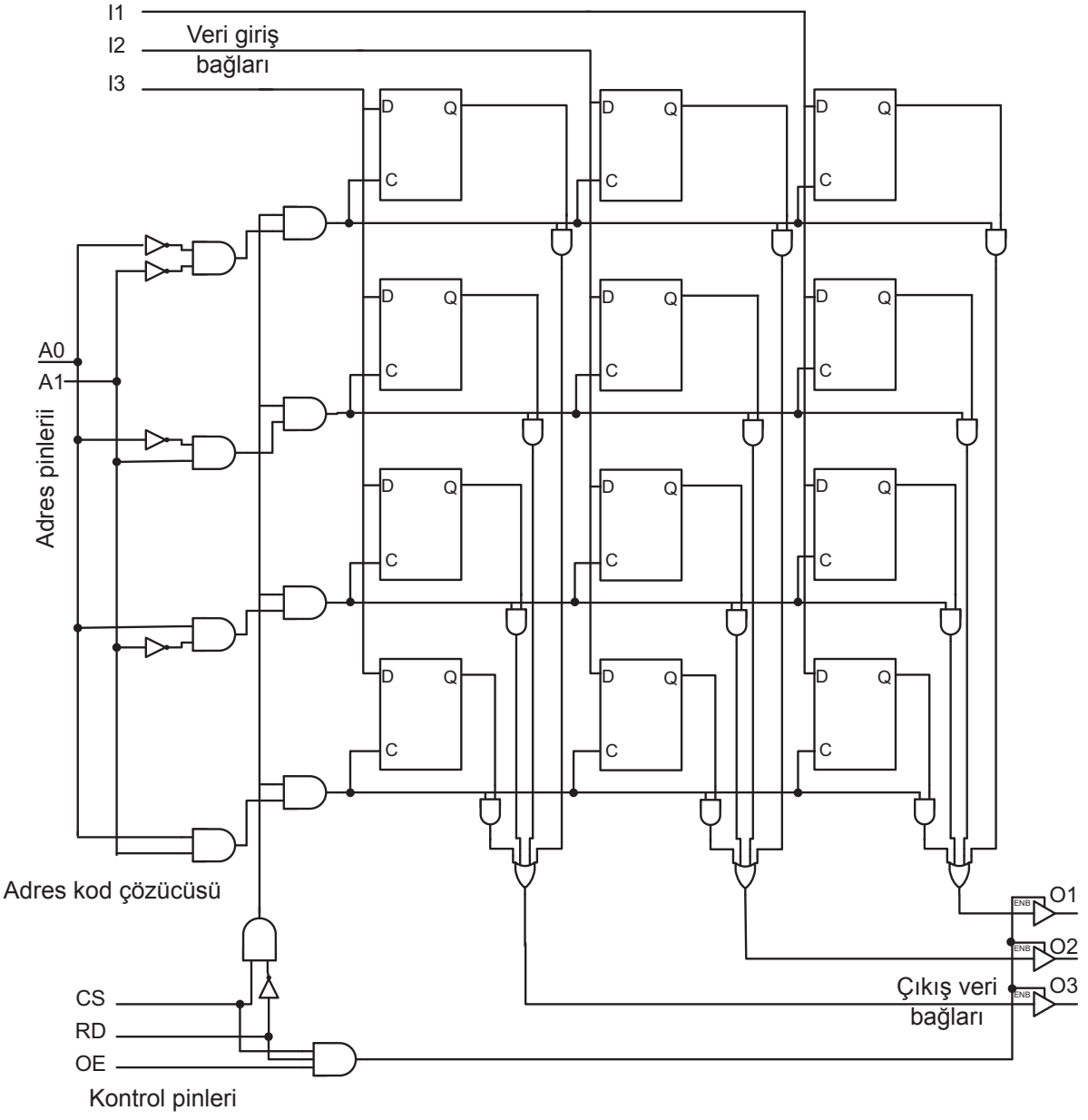
Veri pinleri. Veri pinleri bellek yongalarda verilerin girilmesi (yazılması) ve çıkarılması (okunması) için kullanılıyor. Veri pinleri sadece girişli, sadece çıkışlı ve iki yönlü olabilir. Veri pinleri girişli ise, İngilizce giriş anlamına gelen Input sözcüğünü ilk harfi olan I ile işaretleniyor. Çıkışlı pin ise İngizilce çıkış anlamına gelen Output sözcüğünden birinci harfi olan O ile işretleniyor. İki yönlü ise IO harfleriyle ya da İngilizce veri anlamına gelen *data* sözcüğünin ilk harfi olan D ile işaret ediliyor. Bir bellek yongasında veri pinlerin sayısı bellek matrisinde sütunların sayısına, yani bir bellek konumunun genişliğine bağlıdır. Bellek konumun genişliği içinde yerleşebilen bit sayısını tanımlıyor. Bellek yongaların en büyük kısmı 8 - bitlidir ve buna göre her bellek konumu 8 bit ya da 1 bayt sığdırabilir. 8 - bitli bellekler dışında 16 - bitli, 4 - bitli ve bir bitli belleklere de rastlanabilir. Bellek yongaların katalojik seçeneklerinde kapasite hakkında bilgi yanında, bellek konumların genişliği hakkında ek bilgi de veriliyor. Örneğin, 1Kx8 demek ki yonganın 1K bellek konumu var ve her konum 8 bit sığdırabilir. Bunun anlamı bu yonga toplam 8K bit biriktirebilir. 16Kx1 yonganın 16K konumu var ve her konuma sadece birer bit sığdırabilir.

Her bellek yonganın bir ya da fazla **seçim pinleri** vardır. Bilgisayarda büyük sayıda bellek yongaları vardır. Seçim pinlerin, veri yazmak ya da okumak için bunlardan sadece birini seçmesi gerekiyor. Bu pinler şu işaretlerin biriyle işaretleniyor: CS (chip select), CE (chip enable) ya da sadece S (select). RAM bellekleri için genelde $\overline{CS}$ ve $\overline{S}$ işaretleri kullanılıyor, ROM bellekeri için ise $\overline{CE}$ kısaltması kullanılıyor. Görüldüğü gibi $\overline{CS}$, $\overline{CE}$ ve $\overline{S}$ işaretleri olumsuzluk işareti içeriyorlar. Buna göre bu pinler sadece düşük seviyede oldukları zaman etkin oluyor. Pinler yüksek seviyede oldukları zaman, bellek yongası pasiftir ve ondan veriler çıkarılabilir ya da girilebilir.

Kontrol pinleriyle gerçekleşecek işlem belirleniyor, yani veriler çıkarılacak mı girilecek mi diye belirenliyor.ROM belleğin bir kontrol pini var $\overline{OE}$ (output enable) ya da $\overline{G}$ (gate). Verilerin veri pimlerinden çıkması için $\overline{CE}$ ve $\overline{CS}$ pinleri düşük seviyede olmalıdırlar. $\overline{OE}$ pini yüksek seviyede olunca, veri pinleri devre dışı kalıyor ve onlar için yüksek empemdans durumunda olduklarını diyoruz. Bir kontrol pini olunca, o zaman bu kontrol pini R/W (read/write) ile işaretleniyor. Bu pin alçak seviyede olunca, o zaman yeni veriler yazılıyor, yüksek seviyede olduğu zaman ise önceden yazılmış veriler okunuyor.

İki kontrol pini kullanılıyorsa, onlar WE (write enable) yazmak için ve OE (output enable) okumak için kullanlıyor. Her iki kontrol pini aynı zamanda aktif olamaz, ikisi pasif olursa o zaman veri pinleri için yüksek empendans durumunda olduklarını diyoruz.

Resim 1.44'te 12 palstan flip floptan oluşan bellek yonganın mantıksal diyagramı gösterilmiştir.



Resim 1.44. 4x3 bellek yongaların organizasyonu

Her flip flop bir bitlik bilgi saklayabilir ve ona göre bellek yonganın tümü toplam 12 bit saklayabilir. 12 flip flop dört satıra ve üç sütuna yerleştirilmiştir. Ona göre dört bellek konumu vardır ve her bir konumda üçer bit hafıza edilebilir. Bu yonganın $İ_1$,$İ_2$ ve $İ_3$ İle işaretlenen üç veri girişi var ve O_1,O_2 ve O_3 ile işaretlenen üç özel pinleri var.

Adres kod çözücüsü dört bellek konumdan birini seçmek için kullanılıyor. Tablo 1.17. tek bir bellek konumuna erişim sağlayan iki adres girişinin kombinasyonlarını gösteriyor.

A_0	A_1	Erişim
0	0	Bellek konumu numara 1
0	1	Bellek konumu numara 2
1	0	Bellek konumu numara 3
1	1	Bellek konumu numara 4

Tablo 1.17. Adres kod çözücünün doğruluk tablosu

RD girişi yüksek seviyedeyse, o zaman bazı bellek konumundan verileri okuyabiliriz, eğer RD=0 o zaman veri okuyamıyoruz, ancak yeni veriler yazabiliriz. OE girişi çıkış arabellekleri etkin hale getiriyor.

1.7.2. RAM Bellekleri

RAM belleği geçici bellektir. Eletrik akımın kesilmesi ardından RAM belleğinde bulunan veriler kayboluyor. RAM ve ROM bellekleri arasında en büyük fark RAM belleğin bilgisayar çalışırken programlanması, ROM belleği ise bilgisayara takılmadan önce programlanmasından oluşuyor. RAM bellekler çok hızlı belleklerdir. Onların hızı erişim zamanıyla ifade ediliyor. Erişim zamanı bellek yongasının girişte adres kombinasyonunu aldığı andan, çıkışta veri bitlerin meydana gelme anına kadar geçen zaman olarak tanımlanıyor. RAM belleklerde erişim zamanı birkaç nano saniye sürüyor.

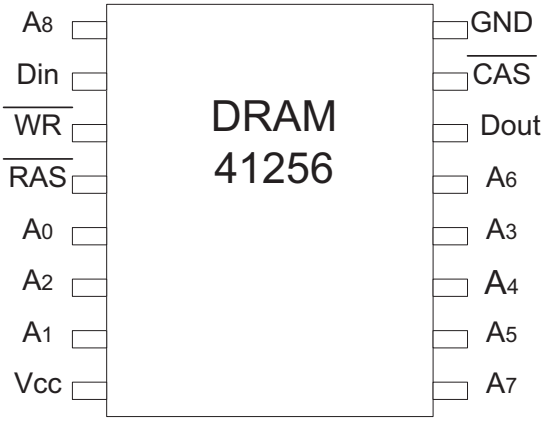
Verilerin korunması ve yenileme şekline göre iki RAM bellek türü var: DRAM ve SRAM bellekleri.

SRAM (Static RAM) statik RAM belleğidir ve hücrelerinde D flip flopları var. Onun yenilemesinde gerek yok ve DRAM belleğinden çok daha hızlıdır. Pahalı olduğundan dolayı SRAM bilgisayarda daha az boyutta kullanılıyor, genelde önbellek (keş belleği) olarak kullanılıyor

DRAM (Dinamic RAM) ya da dinamik RAM belleği, hücreleri bir tranzistör ve bir kondansatör içeren RAM belleğidir. Kondansatör şarjlı olması yüksek seviye (bir) durumu demektir, boş olması ise düşük seviye (sıfır) durumu demektir. Bu bellek türün çok kısa hafıza süresi var, çünkü zaman geçtikçe kondansatörün elektrisite akımı meydana geliyor.

Bundan dolayı, DRAM belleğindeki verilerin devamlı yenilenmesi (gerilimin yeniden yükselmesi) gerekiyor. DRAM belleği çok yavaştır, ancak iyi tarafları ucuz fiyatı ve büyük yoğunluğudur (bir yongada büyük sayıda bitlerin olması). Birkaç DRAM bellek türü bulunmaktadır. En eski DRAM bellek türü FPM (Fast Page Mode) DRAM belleğidir. Bitlerden oluşan matris şeklinde düzenlenmiş bellek yongasıdır. Aranan bite erişmek için satır numarası ve sütun numarası bilinmelidir. Bu bellek türünden sonra EDO (Extended Data Output) DRAM türü ortaya çıkmış. EDO DRAM belleğinde ikinci çağrının birinci çağrı bitmeden başlamasını sağlıyor. Bu şekilde belli sürede bellekten alınabilen veri miktarı artıyor.

SRAM belleklerden, SRAM 4016 belleğini anacağız. Onun 2Kx8 yapısı var ve buna göre 11 adres pini ve 8 veri pini bulunuyor. SRAM'ın kapasitesi en çok 128Kx8 olabilir. SRAM'dan farklı olarak, DRAM'ın çok daha yüksek kapasitesi var ve 64Mx1'den yukarıdır. DRAM belleklerin dezavantajı üreticilerin öngördüğü adres pinlerinden çok daha fazla adres pini aramalarıdır. Bundan dolayı, adres pinleri çoğullaştırılıyor. Bu şekilde aynı adres hattın üzerinden iki adres biti geçebiliyor, örneğin A_0 ve A_8 ya da A_3 и $A_{11,}$ ancak aynı zamanda değil. Önce biri geçiyor, ondan sonra diğeri.



Resim 1.45. DRAM bellek yongasının Pin (iğne) diyagramı

Resim 1.45'te 64Kx1 yapısı olan DRAM 41256 yonganın pin - diyagarmı gösterilmiştir. Resimden görüldüğü gibi bu yonganın 8 adres pini var, 64K bellek konumun adreslenmesi için ise 16 adres pini gerekiyor. Bu sorun iki ek pinin eklenmesiyle çözülmüştür: $\overline{CAS}$ (column address strobe) ve $\overline{RAS}$ (row address strobe). $\overline{RAS}$ pini aktif olduğu zaman, adres pinlerine A_0'dan A_7'ye kadar bitler geliyor. Ondan sonra $\overline{RAS}$ özel pini yüksek seviyeye çıkıyor ve onun yerine CAS pini aktifleşiyor. *CAS* pinin aktifleşmesiyle aynı adres pinlerine A_8'den A_{15}'e kadar bitlerin gelmesine izin veriliyor.

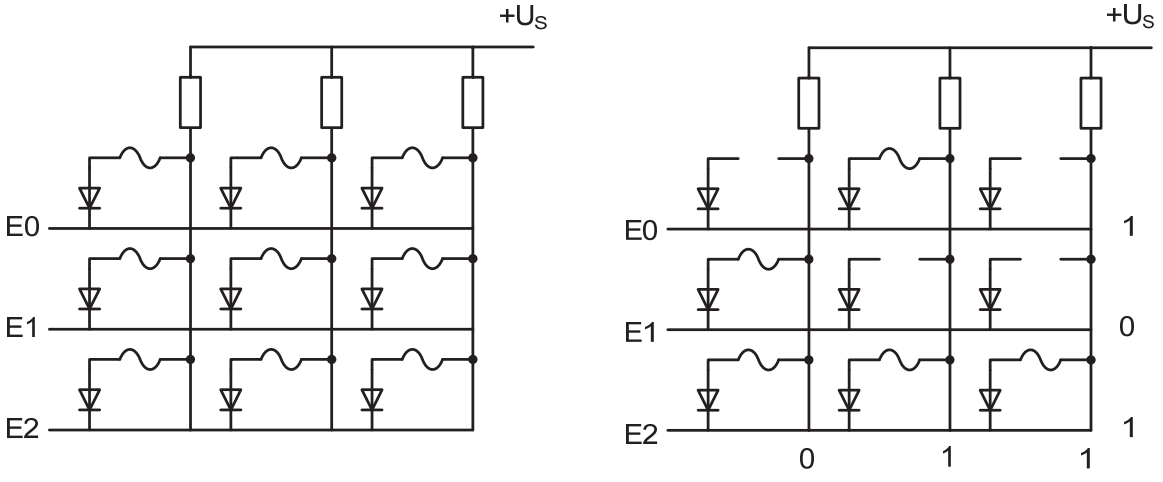
1.7.3. ROM Bellekleri

ROM (Read Only Memory) verilerin kalıcı olarak korunması için kullanılan bellektir. ROM belleğinde verilerin sadece okunmasını sağlıyor, yeni verilerin yazılması ise yapılamaz. Ceryan akımının kesilmesinden sonra, veriler ROM belleğinde kalıyorlar. ROM yongaları daha yüksek güvenlik derecesi sağlıyor, çünkü hafızalanmış içerikler istenmeyen değişikliklerden korunmuş oluyorlar. ROM belleğinde, bilgisayarı açtıktan sonra işletim sistemin kaldırılması için kullanılan programlar bulunuyor. Bu programlar BIOS (Basic Input Output System) adıyla biliniyorlar.

ROM bellek yongalarına verilerin girilmesi için farklı yöntem türleri bulunmaktadır. Ona göre farklı ROM bellek türleri vardır: maskeli ROM bellekler, PROM, EPROM, EEPROM ve fleş bellekleri.

Maskeli ROM belleği, içeriği üretim sırasında yazılan ROM bellekeridir. ROM yongaların üretilmesi için, mikro işlemci yongaların üretilmesi için kullanılan teknoloji yöntemlerine benzer yöntemler kullanılıyor. Maskeli ROM yongaları çok kullanılan ve değişmeyen programların korunması için kullanılıyorlar.

PROM programlanabilir ROM belleğidir. Bu bellek türünü kullanıcı kendi ihtiyaçlarına göre programlayabilir. Böyle bir belleğin yapısı resim 1.46'da gösterilmiştir. Diodlu PROM belleğinde, bellek matrisinin her hücresinde, erinir nikel - krom sigortayla seri şekilnde bağlanmış diyotta bulunuyor.



Resim 1.46. PROM yongasında bilgi yazmak yöntemi

Bellek programlanmış olmadığı zaman tüm diodlar matrisin satırları ve sütunlarıyla bağlıdırlar. Kullanıcı, mantıksal sıfır durumunun olmasını isteyen yerlerde sigor-

taların yanmasına yol açarak belleği kendi başına programlıyor. Bu süreç adım adım gerçekleşiyor. Satırlar birer birer adresleniyor, diyodun çekilmesi gereken veri bağlantılara negatif tetikleme gönderiliyor. O zaman diyottan ve sigortadan büyük miktarda ceryan akıyor ve sütun ve satır arasındakı bağ kesiliyor. PROM belleği bir defa programlandıktan sonra yeniden programlanamaz.

EPROM bellekleri (Erasable Programmabile Read Only Memory) bellek elemanları olarak izole edilmiş kapılı MOS transistörleri kullanıyor. Bellek programlanmış olmadığı zaman, veri hattında +5V büyüklükteki gerilim, MOS transistörlerde kanalın oluşması için yeterlidir ve o şekilde tüm bellek hücrelerin içeriği sıfırdır. Belleğin programlanması veri hattına yüksek gerilim (25V) getirilerek yapılıyor ve bu arada izole edilmiş kapının durumu mantıksal sıfırdan mantıksal bire değişiyor. Bu şekilde programlanmış bellek, kendi içeriğini on yıldan fazla değişmeden olduğu gibi kalabilir. Böyle programlanmış yonga mor ve ötesi ışıklarıyla 20 dakikalık süre içinde ışınlandırılırsa, belleğin içeriği kayboluyor, çünkü kanalın iletkenliği azalıyor ve elektronlar izole kapısını terk ediyor. Bundan sonra yonga yeniden programlanabilir. En sıkça kullanılan EPROM belleği 2716 tanımlama numaralı yongasıdır. Bu yonga 2Kx8 yapılıdır ve buna göre 11 adres pini ve 8 veri pini içeriyor. 27x_ serisinden şu numaralı yongaları anıyoruz: 2704 (512x8), 2708 (1Kx8), 2716 (2Kx8), 2732 (4Kx8), 2764 (8Kx8), 27128 (16Kx8), 27256 (32Kx8), 27512 (64Kx8) ve 271024 (128Kx8). Tüm bu yongaların 8 veri pinin içerdiğini, adres pinleri sayısı ise değişik olduğunu görebiliyoruz. Adres pinlerin sayısı, veri pinin sayısıyla çarpılırsa be seride 272nin ardına eklenecek sayı elde ediliyor.

EEPROM (Electrically Erasable Programmabile Read Only Memory) belleğinde de bellek hücreleri olarak izole edilmiş kapılı MOS transistörlerini kullanıyor, sadece EPRPM belleklerinde MOS transistörlerine karşın elektrotlar arasındaki izolasyon daha düşüktür. Programlama EPROM belleğine benzer şekilde yapılıyor, ancak daha düşük izolasyonu olduğu için +10V cıvarında daha az girim gerilimi kullanılıyor. İçeriklerin silinmesi, yazma geriliminden ters kutupluluk gerilimi kullanılarak elektrik yoluyla yapılıyor. Silmek yazılım kullanımıyla yapılıyor ve yeni içerikler hemen girilebilir. Bu Read Only Memory adını kullanmayı soru işareti altına koyuyor, ancak EEPROM belleğin içeriğinin çok seyrek, örneğin senede bir kez değiştiğini göz önüne alınmalıdır. RAM belleğin içeriği bir saniye içinde birkaç kez değişebilir.

Yarıiletken belleğin en yeni şekli **fleş belleği**dir. Adı yeniden programlanabilecek hızından geliyor (İngilizce flash - yıldırım). Fleş bellek seksenli yılların ortalarında meydana gelmiş ve hem fiyat açısından hem de işlevi açısından EEPROM ve EPROM belleklerin ortasıdır. EEPROM belleği gibi, fleş bellek elektrikli silme teknolojisini kullanıyor.

Bütün fleş belleğin silinmesi sadece birkaç saniye sürüyor. Paketleme yoğunluğu EPROM belleğiyle aynıdır (EEPROM'dan daha büyük) çünkü bir bitin afıza edilmesi için sadece bir transistör kullanıyor. Fleş bellekleri kendi sistemleri içinde, yongayı yatak temelinden çıkarmadan, yeniden programlanabilirler. Yongayı daha sıkça yatak temelinden çıkarılması ve yeniden takılması pinlerin hasar görmesine yol açabilir. Fleş bellekeri 100000 kez kadar programlanabilir.

Sonuçlar:

Bir sayının onlu sayı sisteminden ikili sayı sistemine dönüştürülmesi, sayının ikiyle bölünmesiyle ve elde edilen kalanları yazmakla yapılıyor. Elde edilen son kalan en büyük ağırlıklı bittir. Bir sayıyı ikili sitemden onlu sitemine dönüştürülmesi ikili sayının tüm rakamlarını toplamakla yapılıyor, ancak toplamadan önce rakamlar ağırlıklarıyla çarpılıyor 1, 2, 4, 8, 16, 32, 64, 128 vs.

On altılı sayı sisteminden herhangi bir rakam 8, 4, 2 ve 1 ağırlıkların toplamı olarak ifade edilebilir. Toplama giren ağırlıkların yerlerinde bir yazılıyor, toplama girmeyen ağırlıkların yerlerinde ise sıfır yazılıyor.

Mantıksal VE devrenin çıkışında mantıksal biri elde etmek için tüm girişlerim mantıksal bir olmaları gerekiyor. Mantıksal YADA devrenin çıkışında mantıksal sıfır elde etmek için tüm girişlerin mantıksal sıfır olmaları gerekiyor. Eviricinin girişinde mantıksal bir varsa, o zaman çıkışta mantıksal sıfır elde ediliyor ve tersi. D - YADA mantıksal devrenin çıkışında mantıksal bir elde etmek için girişlerde tek sayıda birlerin olması gerekiyor.

TTL ailesinden tümleşik devrelerde mantıksal sıfır için giriş gerilimin en yüksek değeri 0,8 V'tur, çıkış gerilimin en yüksek değeri ise 0,45 V'tur. Mantıksal bir için giriş gerilimin en alçak değeri 2V'tur, çıkış gerilimin en alçak değeri 2,4 V'tur.

Çoğullayıcı veri girişlerden birini seçen ve 2^n veri girişinden ve n kontrol girişinden oluşan devredir. Seçilen veri girişi tel veri çıkışıyla bağlanıyor.

Kod çözücünün girişinde n bitli sayı gelirse, o zaman onunla toplam 2^n çıkıştan bir çıkış seçiliyor.

Flip flop bir bitlik bilgiyi hafıza edebilen en küçük bellek modülüdür. Flip floplaın çalışması tablo, analitik ve grafik şeklinde analize edilebilir.

Flip floplara dijit palsı etkinleştiricidir, veri girişlerin etkisi altında veri çıkışlarında değişikliğin meydana gelmesi için koşuldur. Etkinleştirici dijit palsın mantıksal biri, onun yükselen ya da azalan kenarı olabilir.

Yazmaçların genel ayırımı sabit ve ötelemeli yazmaçlara ayırımıdır. Dört çeşit ötelemeli yazmaçlar var: seri girişli - paralel çıkışlı, seri girişli - seri çıkışlı, paralel girişli - seri çıkışlı ve paralel girişli - paralel çıkışlı. Sabit yazmaçlarda en önemli kontrol sinyalleri yeni veri yazma sinyali (READ) ve önceden yazılmış verileri okuma sinyalidir (WRİTE).

Ötelemli yazmaçın çıkışı onun girişiyle bağlanırsa (geri dönme bağlantısı) o zaman çemberli sayaç elde ediliyor.

Senkron sayaçta onun tüm flip flopları aynı dijit palsı kullanıyor. Asenkron sayaçlarda önceki flip flopun veri çıkışı bir sonraki flip flopun palsın girişine taşınıyor.

Asenkron sayaçları genelde T ya da JK flip floplardan yapılıdırlar. Bu sırada T flip flopun veri girişi mantıksal bir seviyesine bağlanıyor, JK flip flopun iki veri girişi ise kısa bağlantı kuruyorlar.

Bellek yongaları matris şeklinde, satırlardan ve sütunlardan oluşan ağ şeklinde düzenlenmiş oluyorlar. Satırların ve sütunların her kesişim noktasında birer bellek hücresi bulunuyor.

Bellek yongaların en önemli pinleri adres ve veri pinleridir. Adres pinleri bir bellek konumun seçilmesi için kullanılıyorlar, veri pinlerin aracılığıyla ise yeni veriler yazılıyor ya da önceden yazılmış veriler okunuyor.

İki tür RAM belleği vardır: DRAM ve SRAM. DRAM bellekleri flip floplardan yapılıdır ve onlar hızlı belleklerdir. SRAM belleği kondansatörlerden yapılıdır ve onun yenilenmesi, kaybolan elektrisitenin ikmal edilmesi gerekiyor.

Maskeli ROM belleği, içeriği üretim sırasında yazılan bellektir. PROM programlanabilir bellektir. Onu kullanıcı ihtiyaçlarına bağlı olarak programlayabilir. EEPROM belleği, elektrik yoluyla içeriği silinebilen ve yeniden yazılabilen bellektir. Fleş bellekleri 100000 keze kadar tekrar programlanabilir.

Sorular ve Ödevler

1.

A) $E3_{(16)} = ?_{(2)}$
B) $A8_{(16)} = ?_{(2)}$
C) $2B_{(16)} = ?_{(2)}$
Ç) $C5_{(16)} = ?_{(2)}$

2.

A) $49_{(10)} = ?_{(2)}$
B) $60_{(10)} = ?_{(2)}$
C) $47_{(10)} = ?_{(2)}$
Ç) $56_{(10)} = ?_{(2)}$

3.

A) $00111001_{(2)}=?_{(16)}=?_{(10)}$
B) $11100010_{(2)}=?_{(16)}=?_{(10)}$
C) $01000110_{(2)}=?_{(16)}=?_{(10)}$
Ç) $10100101_{(2)}=?_{(16)}=?_{(10)}$

4.

A) $D589_{(16)} + 55CC_{(16)} =?_{(16)}$
B) $45B7_{(16)} + 2233_{(16)} =?_{(16)}$
C) $12FD_{(16)} + C4C9_{(16)} =?_{(16)}$
Ç) $78D7_{(16)} + 192B_{(16)} =?_{(16)}$

---

5. İkili sayıları mantıksal ve aritmetiksel topla!

A) $11001110_{(2)} + 00111011_{(2)}= ?_{(2)}$
B) $11100111_{(2)} + 10010111_{(2)}= ?_{(2)}$
C) $10010011_{(2)} + 11000111_{(2)}= ?_{(2)}$
Ç) $01011101_{(2)} + 10110111_{(2)}= ?_{(2)}$

---

6. İki girişi olan kod çözücünün mantıksal diyagramını çiz. Doğrulu tablosunu yaz.

---

7. En tanıdık tümleşik devre aileleri ve onların temel özellikleri hangileridir?

---

8. Arabellekleme süreci nasıl kullanılır?

---

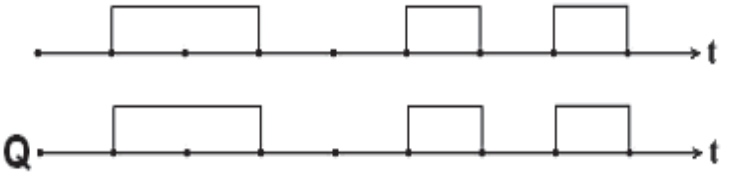
9. Bileşik devre türlerini say!

10. Öncellikle girişli kod çözücünün amacını açıkla?

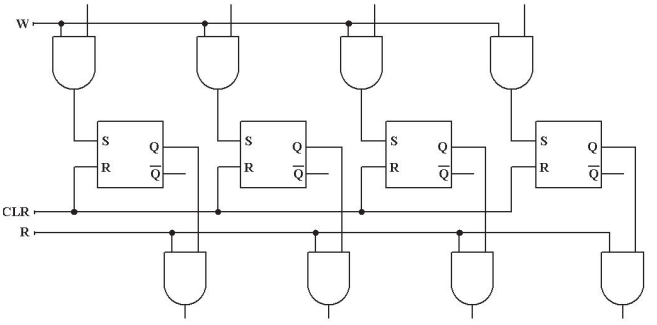
11. Aritmetik ve mantıksal toplama arasında fark nedir?

12. Ardaşıl devrelerin ismini açıkla!

13. Resimde bir flip flop çıkışının reaksiyonu gösterilmiştir. Hangi flip flop türü söz konusudur?



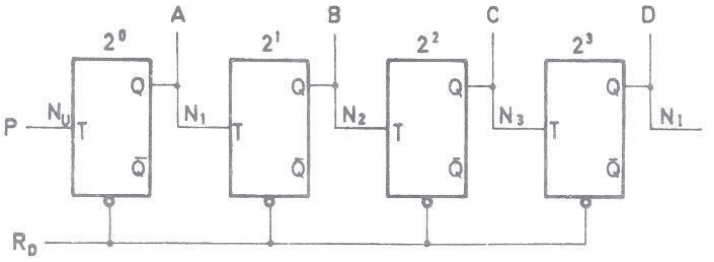
14. Yazmacın yeni içeriklerine girebilmesi için R,W ve CLR kontrol bağlantıların nasıl tetiklenmeleri gerekiyor? Hafızalanmış içerik kaç kez kullanılabilir?



15. 4 satır x 3 sütun yapılı bellek yonganın en az 4 pinini say. Bu pinlerin bütün adlarını yaz (İngiliz diliyle) ve onların kısaltmalarını yaz.

16. Bellek yonganın kapasitesi belirlenmiştir. Onun kaç adres girişi olacak?
    A) 256KB
    B) 1GB
    C) 32KB
    Ç) 2MB1

17. Verilen sayacın, sayacın başlangıç durumundan başlayarak, sayma kapsamında en yüksek değerini elde etmek için kaç dürtü gerekiyor? Onuncu giriş dürtüden sonra sayaçın çıkışlarında ne meydana gelecek?



18. Bellek yonganın yapısı belirlenmiştir. Yonga kaç bit hafıza edebilir? O kaç bayt hafıza edebilir?
    A) 16 satır x 2 sütun
    B) 4 satır x 6 sütun
    C) 8 satır x 8 sütun
    Ç) 16 satır x 4 sütun

19. 4 satır x 3 sütun yapılı bellek yongasında kod çözücünün ne işlevi var?

20. Bir bitlik aritmetik - mantık biriminin bileşim parçaları hangileridir?

21. Bir bitlik aritmetik - mantık biriminde kod çözücü ne için kullanılıyor?

22. 4 kullanıcı girişi olan çoğullayıcı çiz. Onun doğruluk tablosunu yaz.

23. Tek kararlı ve karasız multivibratör arasında kıyaslama yap.

24. Bir bellek yonganın en önemli pinler hangileridir?

25. Bellek yongaları matris şeklinde düzenlenmiştir. Açıkla!

26. SRAM ve DRAM bellek yongaları kıyasla!

27. PROM yongaları programlama sürecini açıkla!

28. Birbitli aritmetik - mantık biriminin giriş sinyallerin değerleri verilmiştir. Çıkış sinyallerin değerleri ve yapılan işlem türünü belirle.

F_0	F_1	İşlem türü	A	B	P_{gir}	Çıkış	$P_{çık}$
1	1		1	1	1		
1	0		0	0	1		
1	0		1	0	0		
1	0		1	0	0		
1	1		1	0	0		

29. Bellek yongaların içerisinde adres kod çözücüsünün fonksiyonunu açıkla!

2. Mikrobilgisayarların Temelleri

2.1. Mikrobilgisayar Sistemlerine Giriş

Mikrobilgisayar sistemi merkez işlem birimi, çalışma RAM belleği ve programları ve işletmeyle elde edilen sonuçların saklanması için kullanılan kalıcı bellekten oluşan sistem olarak tanımlanabilir. Tüm bu bileşenler gereklidir ve onlarsız verilerin işletilmesi yapılamaz. **Bilgisayar sistemleri** ve **mikrobilgisayar sistemleri** terimlerin ayrılmasını denesek, bilgisayar sistemlerin, mikrobilgisayar sistemlerinden yeni donanım ve yazılım bileşenleri ekleyerek elde edildiğini söyleyebiliriz. Örneğin, monitörü çıkış aygıtı olarak ve klavyeyi giriş aygıtı olarak eklersek, mikrobilgisayar sistemi kişisel bilgisayara gelişiyor. Mikrodenetleyiciler mikrobilgisayar sistemleri için diğer bir örnektir. Mikrodenetleyiciler, içinde merkez işlem birimi, RAM ve ROM içeren özel tasarlanmış tümleşik devrelerdir. Her üç bileşen bir yongada yerleştirilmiştir. Kişisel bilgisayarlarımızı klavyesiz, faresiz ve monitörsüz kullanamayacağız gibi, mikrodenetleyiciler de girişlerinde sensörler ve çıkışlarında uygulama birimi eklemeden kendiliğinden çalışamaz.

Bilgisayar tekniğinde mikro ön eki çok sık kullanılıyor. Örneğin, mikrobilgisayar, mikroişlemci, mikrodenetleyici. Mikro önekiyle onların büyük işlevine karşın sahip oldukları küçük boyutlarına vurgu koyuluyor.

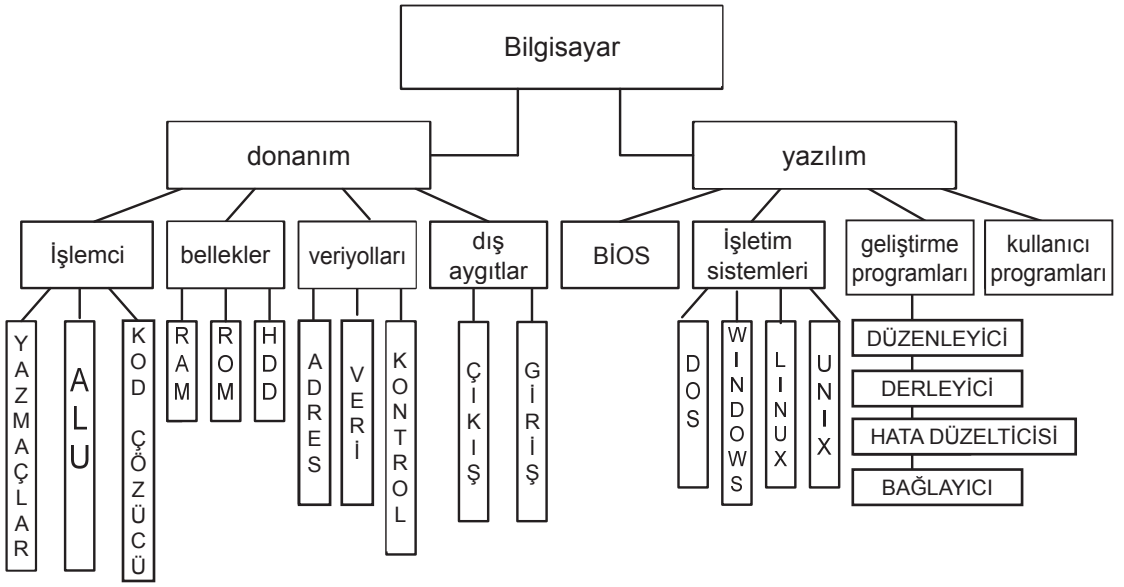
Bugün bilgisayar sistemleri her yerde kullanılıyor ve onlarsız modern hayat düşünülemez. İçinde mini - bilgisayar olmayan elektrik aygıtı yoktur. Bu anlamda telefonlara, televizyonlara, kameralara, mikrodalga fırınlara, CD - çalarlara, oyuncaklara ve evde kullanılan birçok başka aygıta düşünülüyor. Biraz daha kompleksli bilgisayarlar video oyunlar aygıtlarıdır. Onlarda grafiği daha fazla ilerlemiş ancak yazılım açısından çok sınırlıdırlar. Genelde bir mikroişlemci, birkaç MB bellek ve ekran (televizyonla bağlanma olanağı yoksa) içeriyor. İnsanların çoğu bilgisayar terimini işitince, kişisel bilgisayarlara (PC - Personal Computer) düşünüyorlar. Daha eski bilgisayarlara kıyasen, kişisel bilgisayarların çok daha yüksek bellek kapasitesi, sabit disk-

leri var ve çok sayıda dış aygıtlarla bağlanma olanaklara sahiptirler. Kişisel bilgisayarlarda ilk kez işletim sistem terimine rastlanıyor. İşletim sistemi aslında insan ve bilgisayar arasında arabulucu - bağlantı tanımlıyor. Bilgisayarlardan sonra serverler geliyor. Serverler bir ya da fazla mikroişlemci içerebilen, çok hızlı ve güçlü bilgisayarlardır. Serverler en sıkça telefon trafiğiyle, internet ağıyla ve başka yerel ya da küresel bilgisayar ağların yönetilmesi için kullanılıyor. Daha büyük organizasyonlar, bankalar, üniversite kütüphaneleri ve başka daha önemli kurumlar veri bankı bilgisayarlara sahiptir. Onların boyutları oldukça büyüktür ve zamanın ile enerjinin büyük kısmı çok büyük olan bellek konumunun organizasyonu için harcanıyor. En güçlü ve en pahalı süper bilgisayarlardır. Onlar sadece çok büyük bellek alanına değil, aynı zamanda süper hızlı işlemcilere sahiptir.

Tabii ki, tüm saydığımız bilgisayar sistemlerin incelemek olanağımız yok. Birkaç nesil kişisel bilgisayarların mikroişlemcilerini, onların oluşan parçalarını, işlevlerini, bellek organizasyonu ve giriş - çıkış aygıtlarla bağlanma şekillerini inceleyeceğiz.

2.2. Bilgisayar Organizasyonu

Bilgisayarların kompleksli yapısı var ve çoğu bilgisayarların yapısı aynıdır. Bilgisayarın İki temel bileşeni vardır: **donanım** ve **yazılım**. Resim 2.1 bilgisayarın blok - modelini gösteriyor.



Resim 2.1.Genel model bilgisayarın blok modeli

Donanım terimi altında bilgisayarın yapılmış olduğu mekanik parçaları tanımlanıyor. Bilgisayarı oluşturan donanım bileşenleri şunlardır: mikroişlemci, bellekler, veriyolları ve dış aygıtları. **Mikroişlemci** verileri işletiyor, **bellekler** verileri saklıyor, **veriyolları** ise verileri aktarıyor. **Dış aygıtları,** giriş - çıkış birimleri olarak da tanımlanıyor. Dış aygıtları, kullanıcı bilgilerini dijital sinyallere, sıfırlara ve birlere dönüştürüyor ve onun tersini yapıyor. Kullanıcı bilgiler, insanın kullandığı bilgilerdir. Kullanıcı bilgiler ses, resim, metin ya da video olabilir. Tüm donanım bileşenleri kitabın devamında detaylı açıklanacaktır.

Yazılım, donanımın kulanılmasını sağlayan programlar toplamıdır. Donanım ve yazılım birbirine koşullu olarak bağlıdır. Başlangıçta donanım, yazılımdan çok daha pahalıymış, ancak bugün durum eskiden tam tersinedir. Yazılımın gelişiminde özellikle önemli an **işletim sistemin** ortaya çıkmasıdır. İşletim sistemi, bilgisayarın çok büyük donanım kaynakları insana müsait olmasını sağlayan ve daha rahat çalışma sağlayan programlar toplamını tanımlıyor. İşletim sistemi donanıma en yakın olan yazılım parçasıdır. İşletim sisteminin aracılığıyla tüm diğer programlar donanıma ulaşabiliyorlar. En çok kullanılan işletim sistemi Windows işletim sistemidir. Windows işletim sistemi, insan tarafından pencerelerin açmasıyla bilgisayara farklı komutlar vermesini sağlıyor. Ağda bağlanmış bilgisayarlar için özel işletim sistemleri gerekiyor. Ağ bağlantılarında özellikle bilgisayarlar arasındaki iletişime (bilgi alış - verişine) ve verileri izinsiz kullanmaktan korumaya büyük önem veriliyor. Bizde ağda bağlanmış bilgisayarlar için UNIX işletim sistemi yaygın şekilde kullanılıyor.

Geliştirme sistemi altında uygulama programların yapılması için kullanılan yazılım parçası tanımlanıyor. Geliştirme sistemi yeni yazılım yapmak için kullanılan alet olduğunu söyleyebiliriz. Geliştirme sistemini daha iyi anlamak için, bu gruba ait olan bazı program türlerini açıklayacağız:

- Düzenleyiciler (editörler), programda metinin girilmesini, değiştirilmesini, şekillendirilmesini, hafıza edilmesini ve basılmasını sağlayan programlardır;
- Derleyiciler (çevirmenler) daha üst seviye programlama dilinde yazılan programları makine diline çeviriyor;
- Hata düzelteci (debagerler) hataları düzelten programlardır. Programın belli yerlerinde kontrol noktaları yerleştiriliyor ve programın akışı sırasında bu noktalarda durduruluyor ve programın durumu tespit ediliyor;
- Bağlayıcılar programın çalışması için çok önemli olan yeni yazılımı eski yazılımla bağlayan programlardır.

Uygulama programları giriş verilerini, bilgisayarı kullanarak çıkış verilerine dönüştürmek amacıyla yapılıyorlar. Uygulama programları makine dilinde ya da daha yüksek seviyeli programlama dilinde yazılabilirler.

Uygulama programlarını kullanıcılar kendileri yapabilir ya da hazır ürün olarak satın alınabilir (metin düzenleyici programlar, muhasebe programları, bank programları, bilgisayar oyunları vb.). Bazı uygulama programları varolan geliştirme sistemin daha da geliştirilmesi için de kullanılabilir.

2.3. Mikroişlemcinin Temel Organizasyonu

Mikroişlemci bilgisayarın "**beyinidir**". Onun iki temel işlevi var: programlar çalıştırsın ve anakarttaki diğer aygıtları yönetsin.

Resim 2.2. mikroişlemcinin dış görünüşü

Miroişlemci, programları ardaşıl bellek konumlarında sıralanmış baytlar kümesi şeklinde "görüyor". Mikroişlemci programları bayt bayt **çalıştırıyor.** Mevcut baytı işlettikten sonra, mikroişlemci RAM belleğinden sıradaki baytı okuyor, onu işletiyor ve bu süreç içerisinde programın sonu gelene kadar tekrarlanıyor.

Yönetim işlevi mikroişlemciyi master, anakartın efendisi yapıyor. Mikroişlemci bilgisayarın tüm bölümlerinden bilgi biriktiriyor, işletiyor, çalıştırıyor ve ondan sonra tüm bilgisayar sağlam şekilde çalışması için belli bir etkinlik başlatıyor.

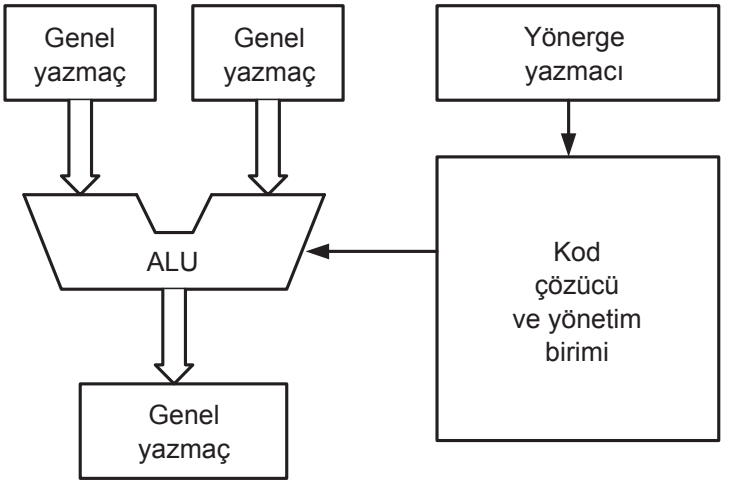
Yönergeler kümesi mikroişlemci tarafıdan çalıştırılabilen tüm yönergelerin kümesini tanımlıyor. Mikroişlemcinin tüm yönergeleri dört temel gruba ayrılabilir: aktarım yönergeleri, aritmetik yönergeleri, mantık yönergeleri ve program kontrol (denetim) yönergeleri.

- Aktarım yönergeleri şunlar için kullanılıyor: mikroişlemci içinde bir yazmaçtan başka yazmaca aktarmak için (move), bellekten yazmaca ya da ters yönde aktarım için (load, store) ve dış aygıtlardan okumak ve yazmak için (in, out).

- Aritmetik yönergeler toplama (add), çıkarma (substract), çarpma (multiply) ve bölme (divide) yönergeleridir.
- Mantıksal yönergeler grubuna şunlar aittir: mantıksal çarpma ya da VE devrenin işlemi (and), mantıksal toplama ya da YADA devrenin işlemi (or) ve birinci tümleyicinin hesaplanması (complement).
- Kontrol yönergeleri program akışını değiştiriyor. Kontrol yönergeleriyle sıradaki yönerge yerine herhangi bir yönerge çalıştırılabilir. Bu grup yönergelere atlama yönergesi (Jump) ve alt program çağırma yönergeleri (Call) aittir. Bu yönergeler koşullu ya da koşulsuz olabilir. Örneğin, atlama yönergesi, sadece önceki çalıştırılan yönergenin sonucu sıfıra eşitse çalıştırılabilir.

Resim 2.3.'te mikroişlemcinin temel organizasyonu gösterilmiştir. Bir mikroişlemciyi **oluşturan parçalar** şunlardır: yazmaçlar, arimetik - mantık birimi ve kod çözücüyle yönetim birimi.

Yazmaçlar mikroişlemcinin içinde bulunan hızlı bellek konumlarıdır. Genel yazmaçlar ve özel amaçlı yazmaçlar olmak üzere ikiye ayrılıyorlar. Genel yazmaçlar işlenmesi gereken verileri içeriyorlar (aktarma, toplama, bölme, tümleştirme, döndürme vs). Her özel amaçlı yazmaçın tam olarak belirlenmiş işlevi vardır. Bu grup yazmaçlara yönerge yazmacı aittir. Yönerge yazmacı işlem kodunu içeriyor. İşlem kodu yönergeleri makine dilinde tanımlanması için kullanılıyor. İşlem kodu sıfırlardan ve birlerden oluşan tek kombinasyondur ve yönergenin tanımlaması ve tanıması için kullanılıyor.



Resim 2.3. Temel model mikroişlemcinin blok modeli

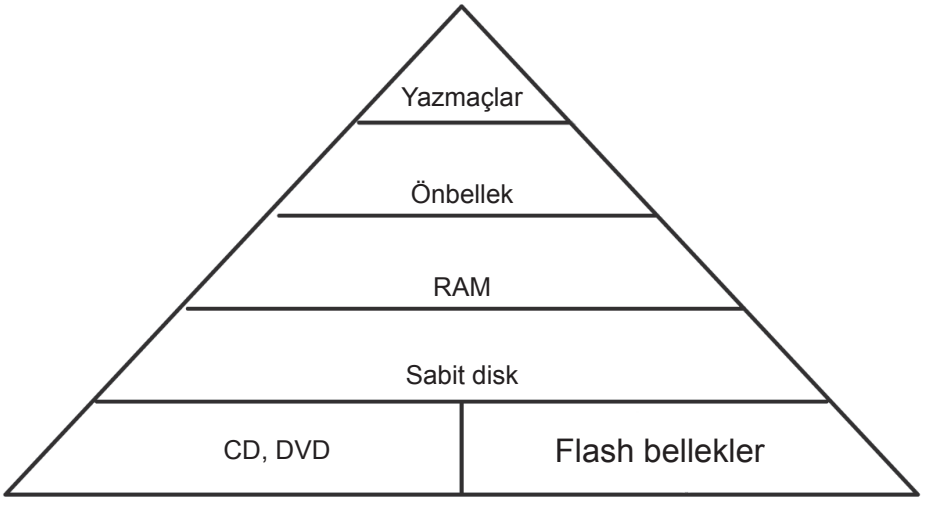
Aritmetik - mantık biriminde tüm aritmetik ve mantık işlemleri yapılıyor. Hatırlayalım, en basit aritmetik işlemleri: toplama,çıkarma, bölme ve çarpma işlemleridir. En basit mantık işlemleri ise VE, YADA ve olumsuzluk işlemleridir.

Kod çözücü işlem kodunu çözüyor. Kod çözücü yönetim birimiyle doğrudan bağlıdır. Yönetim birimi milyonlarca mantık devrelerden oluşmuştur. Yönetim birimi tüm aygıtlardan veriler alıyor ve ondan sonra karar veriyor, yazmaçların içeriğini değiştiriyor, pinlerin mantık durumunu değiştiriyor, programlar çağırıyor, dış aygıtlarını aktifleştiriyor vs.

2.4. Bellek Organizasyonu

Tüm veriler ve mikroişlemcinin çalıştırdığı programlar bellekte yerleşik olmalıdır. **Belleklerin en önemli özellikleri** onların kapasitesi ve hızıdır. Belleklerin kapasitesi baytlarla (B) ölçülüyor. Daha büyük kapasite birimleri şunlardır: kilobayt (KB), megabayt (MB), gigabayt (GB), terabayt (TB). Belleklerin hızı erişim zamanına göre ölçülüyor. Erişim zamanı belleğin istenilen veriyi araması ve bulması için gereken zamandır.

Bilgisayarlar, farklı bellek türleri içeriyor. Bellek organizasyonu her bellek türü için işlevlerin ayırımı ve aranan veriye erişim şekliyle tanımlanıyor. Resim 2.4'te belleklerin piramiti verilmiştir. Alt bölüme yaklaşırken bellek kapasitesi artıyor, çalışma hızı ise azalıyor. Daya büyük çalışma hızı (verilere erişim için daha az süre) bayt fiyatının yükselmesine neden oluyor. Piramitte bellekler hiyerarşik, önemlerine göre düzenlenmiştir. Bellekler tepeye ne kadar daha yakınsa, o kadar mikroişlemciye da yakındırlar.



Resim 2.4. Bellekler piramiti

Yazmaçlar mikroişlemci içinde bulunan çok hızlı bellek konumlarıdır. Yazmaçlar verileri işletmeden önce geçici olarak saklamak için ve sonuçlar elde edildikten hemen sonra yerleştirmek için kullanılıyorlar. Yazmaçlar bilgisayarda en hızlı bellek konumlarıdır, ancak onların kapasitesi sadece birkaç bayt büyüklüğündedir.

Çok kez mikroişlemci işlenmesi gereken verileri bellekten alınca bekleme (wait) olayı yaşanıyor. Bekleme zamanın azalması için **önbellek** (keş bellek) kullanılıyor. Önbellek çok hızlı bellektir. Mevcut programın daha çabuk çalıştığı için, daha yavaş belleklerden gereken veriler önceden getiriliyor ve önbellekte yerleşiyor. Mikroişlemcinin daha sıkı kullanan verileri kendi yazmaçlarında yerleştiremeyince veriler önbellekte saklanabilir. Önbellek bellek yonganın içinde ya da onun dışında bulunabilir. Önbellek hızlı RAM belleğidir, ancak küçük kapasiteye sahiptir.

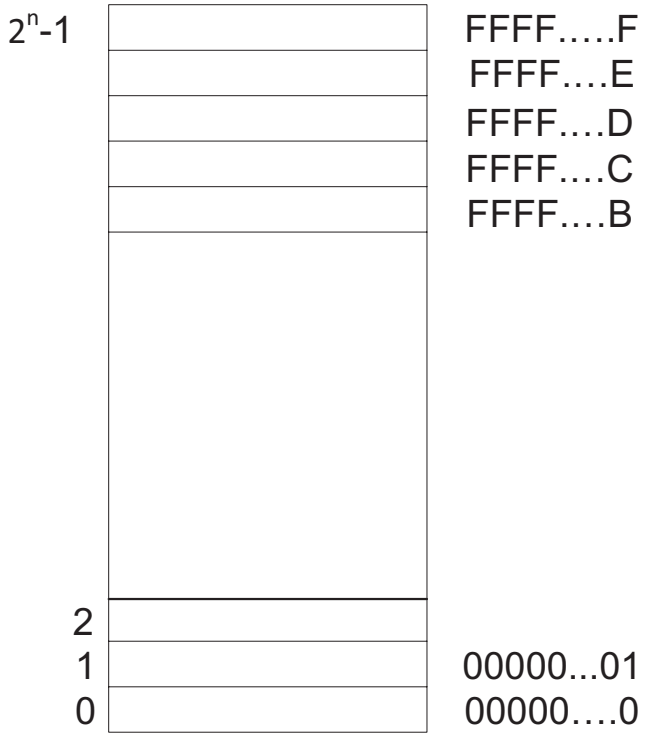
RAM İngilizce Random Access Memory sözcüklerin kısaltmasıdır ve **rastgele erişimli bellek** demektir. İşlemci belleğin tüm konumlarında aynı erişimi vardır, yani hiçbir konumun daha düşük ya da daha yüksek öncüllüğü yoktur. RAM belleğinden aynı zamanda veriler okunabilir ve yeni veriler yazılabilir. RAM belleğin değişir içeriği var ve **çalışma belleği** tanımlıyor. RAM belleğini verilen programın başarılı gerçekleşmesi için gereken tüm verilerin mikroişlemci tarafından geçici yazılabileği çalışma kağıdı olarak düşünebiliriz. Programlar genelde kalıcı, değişmez bellek aygıtlarında kaydediliyor (disk, disket), ancak onların çalıştırıldığı zaman, programlar RAM belleğinde kopyalanmalıdır. RAM belleği, mikroişlemci için önemli olduğundan dolayı, aynı zamanda birincil bellek olarak da biliniyor. RAM **geçici bellektir**. Elektrik kaynağında kesinti olursa RAM belleğinde veriler, bilgisayar tekrar açılınca yenilenmiyor. Bundan dolayı, bilgisayarı kapatmadan önce, RAM'daki veriler, SAVE simgesine tıklayarak sabit diskte kaydedilmelidir. Bunu resim 2.5.'te görebiliriz.

Resim 2.5. RAM'ın mikroişlemcide çalışma belleği olarak işlevi

Mikroişlemci bellekten çok daha hızlı aygıttır. Veri araması bazan çok kompleksli ve yavaş süren süreç olabilir. Çalışma hızına özellikle RAM belleğin ve önbelleğin büyük etkisi vardır. Mikroişlemci programları bayt bayt çalıştırıyor. Bu arada, mikroişlemci yönergeleri RAM belleğinden alıyor. RAM belleği mikroişlemciyi bilgilerle "besliyor", RAM ise sabit diskten besleniyor. Tüm programlar sabit diskte bulunuyor ve çalıştırmaları gerektiğinde RAM'a taşınıyorlar. Eğer bir program çok büyükse ve RAM'da öngörülmüş alana sığmıyorsa, o zaman program birkaç parçaya bö-

lünüyor ve parça parça çalıştırılıyor. RAM çok büyükse, o zaman kullanıcı programından (sabit diskte yerleşmiş olan) birden daha büyük parça alıyor ve alan parçayı mikroişlemcide çalıştırıyor. Bu şekilde RAM'dan sabit diske aktarma sayısı azalıyor ve öylece daha az zaman harcanacak. RAM belleğini yeni konumlarla genişlenmesi yeni konumların adreslenmesini sağlamak için veriyollarda büyük sayıda adres hatları gerektiriyor.

Resim 2.6.'da **RAM belleğin bellek haritası** tanımlanmıştır. Her satır birer bellek yeri tanımlıyor. Bellek yerlerin toplam sayısı adres hatların sayısına bağlıdır ve şu denklemle hesaplanıyor: n adres hatların sayısı ise, bellek yerlerin toplam sayısı=2^n. Sayma sıfırdan başlandığı için son bellek yeri 2^n - 1 sıra sayısıyla belirlenecek. RAM belleğin birinci en alt konumun adresi n sıfırlara eşit olacak, son en üst konumun adresi n birden oluşmuş olacak. Resim 2.6.'da bellek yerlerin adresleri ikili sayılarla değil, on altılı sayı sisteminde verilmiştir.



Resim 2.6. RAM belleğin bellek haritası

Örneğin, adres bitlerin sayısı 10 ise, o zaman bu belleğin kapasitesi 1K olacaktır, bellek yerleri ise 0'dan 1023'e kadar numaralandırmış olacak. Başlangıç adresi 0000000000B=000H olacak, son adres ise 1111111111 B=3FFH.

Bu şekilde tanımlanmış bellek haritası mikroişlemcinin çalışmasını daha kolay anlamayı ve programların çalıştırılmasını sağlıyor. Mikroişlemci programları bayt bayt çalıştırılıyor. Mevcut baytla bitirdiği zaman, sıradaki baytı bulmak için RAM belleğini çağrılıyor. Bir programın baytları genelde üst üste sıralanmıştır ve RAM belleğin ardaşıl bellek yerlerinde (satırlarında) yerleştirilmiştir.

RAM belleği birincil bellektir, sabit disk ve diğer dış bellekler ikincil belleklerdir. RAM, önbellek ve yazmaçlardan farklı olarak, **ikincil bellek** kalıcı bellek tanımlıyor. İkincil bellek daha yüksek kapasiteli olduğu için arama şekli çok önemlidir. Arama hızlı ancak basit olmalıdır. Bu yönde sanal sayfa mekanizmasını anacağız. İkincil bellek sayfalara ayrılıyor, birincil bellek ise çerçevelere ayrılıyor. Sayfalar ve çerçeveler aynı kapasitelidir. Gereklere bağlı olarak sayfalardan çerçevelere aktarım yapılabilir, ancak aktarım sırasında hangi sayfanın hangi çerçeveye yazdırılıdığına dikkat edilmelidir. Sanal belleğini daha detaylı ileride tanıyacağız.

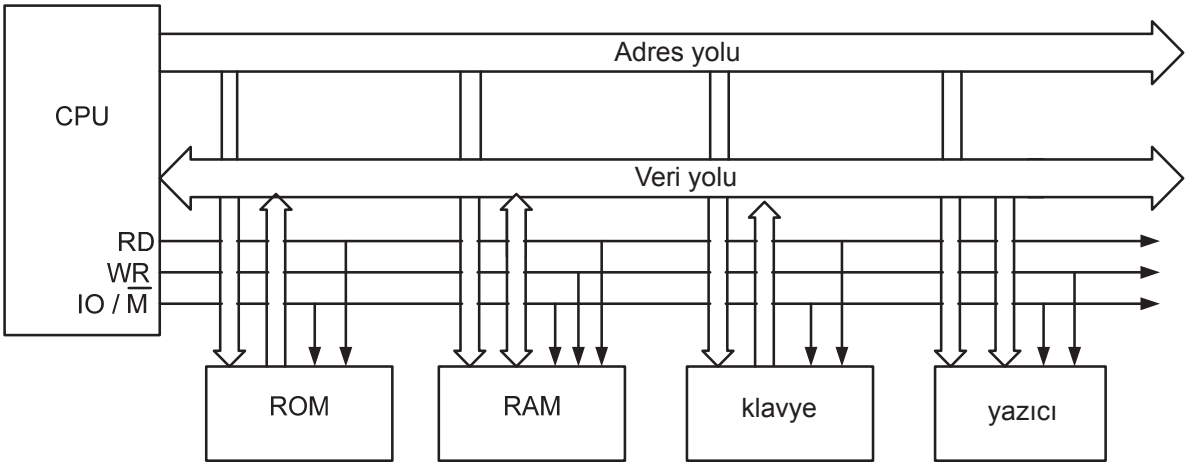
2.5. Bilgisayarın Çalışması

Veriyolları anakartında bulunan bakır hatlarıdır ve verilerin aktarımı için kullanılıyorlar. Birinci kişisel bilgisayarların sistem veriyolu olarak adlandırılan tek bir veriyolları varmış. Sitem veriyolu anakarta yerleşmiş 50 ile 100 arası bakır telden oluşuyormuş, bellek yongaların ve giriş - çıkış aygıtların takılması için aynı mesafede yerleşmiş bağlayıcıları (konektörleri) varmış. Modern bilgisayarlar birkaç özel amaçlı veriyolu içeriyor. Örneğin, işlemciyi bellekle bağlamak için bir veriyolu ve işlemcinin giriş - çıkış aygıtlarla bağlanması için başka bir veriyolu.

Resim 2.7.'de veriyolların bilgisayarda mikroişlemci, RAM, ROM ve birkaç dış aygıtı gibi farklı parçaları nasıl bağladıklarını gösteriyor. Veri aktarımı hatasız gerçekleşmesi için, özel tanımlanmış kurallar kullanılıyor ve bilgisayar sisteminde tüm aygıtlar bu kurallara uymalıdır. Böyle kurallar kümesine veriyolu protokolu (bus protocol) denir.

Veriyoluna bağlı olan aygıtlardan bazıları veri aktarımı başlatabilir ve bu aygıtlara ana aygıtlar ya da masters denir. Diğer aygıtlar pasiftir ve başkasının isteği üzerine veriler alabiliyorlar ya da verebiliyorlar ve onlara köle - slaves denir. Örneğin, merkez işlemci disk deneticiye veri bloku okuma emri verince, işlemci ana aygıtıdır, denetleyici ise köledir.

Bir veriyolun temel özellikleri çalışma frekansı ve genişliğidir. Veriyolun genişliği kaç bakır hattan oluştuğu, yani aynı zamanda kaç bitin taşınabileceği demektir. Çalışma frekansı MHz sıra büyüklüğündendir ve bakır hattan bir saniye içinde kaç bitin geçebileceğini gösteriyor. Çalışma frekansı ve veriyolun genişliğin çarpımıyla **veriyolun geçiş kapsamı** elde ediliyor. Geçiş kapsamının ölçü birimi bit sayısı/saniye'dir.



Resim 2.7. Bilgisayarın bileşen parçaların bağlanma şekli

Aktarılan sinyallerin türüne göre, veriyollar 3 temel gruba gruplanabilir: adres hatları (ADDRESS), veri hatları (DATA) ve denetim hatları(CONTROL).

Adres veriyolu adres bitlerini mikroişlemciden belleklere ve dış aygıtlarına aktarıyor. Adres sıfırlardan ve birlerden oluşan tek kombinasyondur ve sadece bir bellek yerine ya da dış yagıtın tanıması ve tanımlanması için kullanılıyor. Bellek mikroişlemciden adresi alınca, aranan bellek yerinin bulunması için bellek arama başlatıyor.

Veri hatları kullanıcı bilginin bilgisayarda bir aygıttan başka bir aygıta aktarılması için kullanılıyor. Veri veriyolu iki yönlüdür. Bellekten veri okunduğu zaman, veri veriyolu, bellekten mikroişlemciye veriler aktarıyor. Bellekte veri yazıldığı zaman ise veri aktarımı mikroişlemciden bellek yönünde gerçekleşiyor. Veriyollar 8, 16, 32 ve 64 - bitli olabilirler. Daha büyük genişlik daha büyük çalışma hızı demektir. Örneğin, 32 - bitli verinin 8 - bitli veriyolundan aktarılması gerekirse, o zaman 4 döngü veri aktarımı gerekecektir.

Denetim ya da yönetim hatları en büyük sayıdadır ve her yönetim hattı için özel işaret kullanılıyor. Kullanılan işaret aslında verilen denetim sinyalin İngilizce sözcüğün kısaltmasıdır. Kontrol hatları yapılması gereken işlemin tanımlanması için kullanılıyor. Bazı kontrol bitleri mikroişlemciden başka aygıtlara taşınıyor ve bir komut türü tanımlıyor. Başka kontrol bitler başka aygıtlardan mikroişlemciye aktarılıyor ve önceden verilmiş komutların nasıl gerçekleştikleri hakkında geri bilgi türü tanımlıyorlar. Verinin belleğe yazılacağını yoksa verinin bellekten okuyacağını kontrol veriyolundan iki kontrol hattı karar veriyor: $\overline{RD}$ ve $\overline{WR}$. Onlar READ (oku) ve WRİTE (yaz) sözcüklerin kısaltmalarıdır. Onların üzerindeki olumsuzluk mantıksal sıfıra aktif oldukları demektir. Aynı zamanda iki hattın aktif olma durumu olamaz (ya okuyacağız ya yazacağız). Bir diğer önemli kontrol hattı IO/M işaretli hattır. Bu işaret İnput Output/$\overline{Memory}$'nin kısaltmasıdır ve giriş çıkış aygıtlar/bellek anlamına geliyor. Bu sinyal İO/M=1 olunca, o zaman mikroişlemcinin bazı giriş - çıkış birimiyle (dış aygıtla) işletişim kurduğu demektir, $IO/\overline{M}$=0 olunca ise, o zaman miroişlemcinin bazı bellek modülüyle iletişim kurduğu demektir.

Örnek 2.1: $IO/\overline{M}$=0, $\overline{RD}$=0 ve $\overline{WR}$=1 kontrol sinyalleri verilmiştir. Mikroişlemci hangi aygıtla iletişim kuruyor ve nasıl işlem gerçekleştiriyor?

Çözüm: Mikroişlemci bazı bellek modülünden ($IO/\overline{M}$ =0) veri okuyor ($\overline{RD}$=0).

Veriyollar bilgisayarın anakartında yerleşmiş bulunuyorlar ve onların sayısı değişmiyor. Veriyol birimine hatlar sayısının artması bilgisayar fiyatını büyük ölçüde artmasına yol açıyor. Veriyolların fiyatı ve veriyolların genişliği arasında bazı uzlaşma yapılmalıdır. Örneğin, elimizde ne kadar fazla adres hattımız varsa, o kadar daha büyük bellek alanı doğrudan adreslenebilir. Eğer veriyolun n adres hattı varsa, o zaman toplam 2^n bellek yeri adreslenebilir. Her yeni nesil bilgisayarlarla veriyollarda hatlar sayısı artıyor, hem adres hem veri hem de kontrol hatların sayısı artıyor. Sıkça kullanılan süreç **çoğullama** sürecidir. Çoğullama, veri ve adres bitlerin bir veriyolundan aktarımı, ancak aynı zamanda değil, farklı zaman aralıklarla (önce bir sinyal türü sonra başka sinyal türü) gerçekleşen aktarım demektir. Adres ve veri veriyolunun çoğullaması olabilir. Veri veriyolundan veri ve adres bitleri taşınıyor. Bu bitlerden hangisinin taşınacağına ALE (Address Letch Enable) kontrol sinyali karar veriyor. ALE sinyalı yüksek seviyede olduğu zaman, veriyolundan adres bitleri aktarılıyor, ALE sinyali alçak seviyede olduğu zaman ise veri bitleri aktarılıyor. Bellekten bazı verinin okunması gerektiği zaman, her zaman önce verinin adresi aktarılıyor, ondan sonra da bellekte arama yapıldıktan sonra, veri aynı veriyolundan aktarılıyor. Buna göre, eğer adresin aktarıldığı sürede veri veriyolu çoğullanmazsa kullanılmadan kalır.

Sonuçlar:

Bilgisayarı oluşturan donanım bileşenleri şunlardır: mikroişlemci, bellekler, veriyollar ve dış aygıtlar. Mikroişlemci verileri işletiyor, bellekler verileri saklıyor, veriyolları ise aktarıyor. Dış aygıtlar giriş - çıkış adıyla da biliniyor. Dış aygıltar kullanıcı bilgileri dijital sinyallere, sıfırlara ve birlere dönüştürüyor ve aynısını ters yönde yapıyor.

Mikroişlemci bilgisyarın “beyinidir”. Mikroişlemcinin iki temel işlevi var: programları çalıştırıyor ve anakartta yerleşmiş olan diğer aygıtları yönetiyor.

Mikroişlemci şu parçalardan oluşmuştur: yazmaçlar, aritmetik - mantık birimi ve kod çözücüyle yönetim birimi.

Yönetim birimi tüm aygıtlardan veriler arıyor ve ondan sonra karar getiriyor, yazmaçların içeriğini değiştiriyor, pinlerin mantıksal durumunu değiştiriyor, programlar çağırıyor, dış aygıtları aktifleşiyor vs.

Belleklerin en önemli özellikleri bellek kapasitesi ve hızıdır. Belleklerin kapasitesi baytlarla (B) ölçülüyor. Daha büyük kapasite birimleri şunlardır: kilobayt (KB), megabayt (MB), gigabayt (GB) ve terabayt (TB). Bellek hızı erişim zamanı ile ölçülüyor.

RAM İngilizce Random Access Memory sözcüklerin kısaltmasıdır ve rastgele erişimli bellek anlamına geliyor. İşlemcinin tüm bellek konumlarına aynı erişimi var, yani hiçbir konum daha yüksek ya da daha düşük öncüllüğü yoktur.

RAM’dan aynı zamanda veriler okunabilen ya da yeni veriler yazılabilen bellektir. RAM değişebilen içerikli bellektir ve çalışma belleği tanımlıyor, yani RAM belleğini verilen programın başarılı gerçekleşmesi için gereken tüm verilerin mikroişleci tarafından geçici yazılabileği çalışma kağıdı olarak düşünebiliriz.

RAM geçici bellektir. Elektrik kaynağında kesinti olursa RAM belleğinde veriler bilgisayar tekrar açılınca yenilenmiyor.

Bellek yerlerin toplam sayısı adres hatların sayısına bağlıdır ve şu denklemle hesaplanıyor: n adres hatların sayısı ise, bellek yerlerin toplam sayısı=2^n RAM belleğin birinci en alt konumun adresi n sıfırlara eşittir, son en üst konumun adresi n birden oluşuyor.

Bir veriyolun temel özellikleri çalışma frekansı ve genişliğidir. Veriyolun genişliği kaç bakır hattan oluştuğunu tanımlıyor, yani kaç bitin aynı zamanda aktarılabileceğini tanımlıyor. Çalışma frekansı bir saniye içinde bakır hattan kaç bitin aktarılabileceğini gösteriyor. Çalışma frekansı ve veriyolun genişliği çarpılırsa, veriyolun geçiş kapsamı elde edilecek.

Sorular ve Ödevler

1. Mikroişlemcinin, belleklerin, veriyolların ve dış aygıtların işlevlerini kısaca açıkla!
2. Mikrobilgisayarların çalışması için işletim sisteminin önemini açıkla!
3. Bir gelişme programı hangi parçalardan oluşuyor? Her parçanın işlevini açıkla!
4. Mikroişlemcinin bilgisayarda işlevi nedir?
5. Bir mikroişlemcinin en önemli parçalarını say ve onların işlevlerini açıkla!
6. Mikroişlemcinin yönerge kümesi terimi altında neyi tanımlıyoruz?
7. Bellek modülün temel özellikleri hangileridir?
8. RAM rastgele erişimli geçici çalışma belleğidir. Açıkla!
9. DRAM belleğinin SRAM belleğine kıyasen avantajları ve dezavantajları nedir?
10. RAM'ın büyüklüğü, mikroişlmcinin çalışma hızına neden etkiliyor?
11. RAM belleğin bellek haritasını çiz ve adresleme şeklini açıkla!
12. Adres girişlerin sayısı, bir bellek yonganın kapasitesine nasıl etkiliyor?
13. Bellek yonganın giriş pinlerinin amacını açıkla! Belleğin veri pinleri ne zaman giriş pinleri ne zaman da çıkış pinleridir?

14. Bellek yongada birkaç seçme pini say!

15. Adres veriyolun çoğullucu ve veri veriyolun çoğullucunun kontrolü için hangi kontrol sinyalleri kullanılıyor?

16. Adres, very ve kontrol veriyolundan hangi sinyaller aktarılıyor?

17. Bir veriyolun temel özellikleri nedir?

18. 512 MHz çalışma frekanslı, on altı - bitli veriyolunun geçiş kapsamını hesapla?

3. Mikroişlemcinin Genel Yapımı

3.1. Mikroişlemcilerin Tarihsel Gelişimi

Bilgisayarların hızla gelişimi tümleşik devrelerin hızlı gelişimiyle koşullanıyor. Mikroelektronik tümleşik devrelerin - yongaların tasarımı, biçimlendirme ve üretimiyle ilgilenen elektroniğin dalıdır. Yonga silisyum tabağıdır. Bu tabakta yayınım süreci yardımıyla bor ve arsen atomları getiriliyor ve bu şekilde p ve n tabakaları elde ediliyor. İstenilen performansla yonga elde etmek istersek bu tabakaların kalınlığı ve onlarda atom yoğunluğu hassas şekilde hesaplamalıyız. Bir tümleşik devrenin temel özellikleri şunlardır: tümleşik devrenin güvenirliği, alan biriminde tümleşme derecesi, çalışma hızı, diğer tümleşik devrelerle bağlanma şekli, arabağ (arabirim) vb.

Mikroişlemci tüm bilgisayar sistemin çalışmasını birleştiren yonga olarak tanımlanıyor. Mikroişlemci bilgisayarın beyini olarak sayılıyor ve bilgisayarın gücü mikroişlemcinin performanslarına bağlıdır. Bilgisayarın adı bile onun mikroişlemcisinden kaynaklanıyor.

Mikroişlemcilerin tarihi kısa ancak çok yoğundur. 1960 yılında “İntel”’in başkanı, Gordon Mur’un öngördüğü kanun, sonraki 3 on yılda doğru olduğu kanıtlanmış. Bu kanuna göre bilgisayarların gücü ve tümleşik devrelerin karmaşıklığı her iki sene iki misli artarken, onların fiyatı yarıya düşecek. Tablo 3.1’de mikroişlemcilerin kronoljik gelişimi ve onların performansı verilmiştir.

1971 yılında ilk ticari mikroişlemci, 8008 mikroişlemcisi üretilmiştir. 8008 aslında 4004’ün 8 bitli verziyonuydu. İki sene sonra “İntel” modern sekiz bitli mikroişlemci 8080’ı tanıtmıştır. “İntel”le paralel olarak diğer dünyaca tanıdık elektronik bileşenler üreticileri de satışa, benzer sekiz bitli mikroişlemciler sunmuşlar, örneğin “Motorola” üretecinin MC 6899 mikroişlemcisi ya da “Zilog” şirketinin Z - 8.

“Zilog” mikrokontrolörlere dayanarak mikroişlemciler üretiyor.

Mikroişlemci	Üretim yılı	Hız MIP	Frekans	Transistörler sayısy	Yazmaçların büyüklüğü	Adres alanın büyüklüğü	Veriyolların büyüklüğü	Ön - bellek
8086	1978	0,8	8MHz	29K	16	1M	16	/
İntel 286	1982	2,7	12.5MHz	134K	16	16MB	16	/
İntel 386	1985	6,0	20MHz	275K	32	4GB	32	/
İntel 486	1989	20	25MHz	1,2M	32	4GB	32	8KB L1
Pentium	1993	100	60MHz	3,1M	32	4GB	64	16KB L1
Pentium Pro	1995	440	200MHz	5,5M	32	64GB	64	16KB L1 256K ya da 512KB L1
Pentium II	1997	466	266MHz	7M	32	64GB	64	32KB L1 256/512KB L2
Pentium III	1999	1000	500MHz	8.2M	32GP 128 SIMD - FP	64GB	64	32KB L1 512KB L2

MIPS - Milions of instruction per second

Tablo 3.1. Mikroişlemcilerin kronolojik gelişmeleri

8080 işlemcisi yönergeleri 8008’dan on misli daha hızlı çalıştırıyor ya da başka sözlerle 50.000 yönerge saniyede (20 μs bir yönerge). 8008’den farklı olarak 8080 tamamiyle TTL uyumludur ve bunun sonucu olarak arabirim çok daha basit ve ucuzdur. 8080’in 64KB belleği var, bu bellek büyüklüğü 8008’den dört misli daha büyüktür. Ayrıca, bu mikroişlemci 1974 yılında MİTS Altair 8800 adıyla satışa çıkan ilk kişisel bilgisayarda dahil edilmiş.

1977 yılında “İntel” son 8 bitli mikroişlemci **8085**’i temsil etmiş. Günümüzde farklı elektronik aygıtlarda 200 milyon üzerinde böyle mikroişlemci kullanılıyor ve üretimi gelecekte de devam edecek. 8085’e benzer 500 milyon üzerine 8 bitli mikroişlemci satan bir diğer şirket “Zilog”’tur ve bu işlemci Z - 80 adıyla biliniyor.

1978 yılında **8086** mikroişlemcisi üretilmiştir, bir yıl sonra ise 8088 modeli üretilmiştir. Her iki yonga 16 - bitlik mikroişlemcidir ve çalıştırma hızı 2.5MIPs. Mikroişlemci belleği 16 misli kadar artıyor ve 1 MB büyüklüğündedir. Bu dönemde yenilik 4 ile 6 bayt arasında olan küçük yönerge önbelleğidir. Yönerge önbelleği, daha çalıştırılmadan önce, birkaç yönergenin bellekten mikroişlemciye önceden gönderilmisini sağlıyor. Yönergeler kümesi 20.000 farklı yönergeden oluşan büyük kümedir.

Onlardan bazıları komplekslidir ve bundan dolayı bu mikroişlemciler için CISC (Complex Instruction Set of Computers) teknolojiyle olduğunu diyoruz. 80286 mikroişlemcisi 8086 ve 8088'le yakın aynıdır, aralarındaki fark bu mikroişlemcide 16MB alanın adreslenebileceği olanağıdır.

Yeni yazılım uygulamaları daha hızlı mikroişlemciler, daha fazla bellek ve daha geniş veriyolları arıyormuş. Bu nedenden dolayı, 1986 yılında İntelin birinci 32 - bitli **80386** mikroişlemcisi üretilmiştir. Bu mikroişlemci 32 - bitli veri yolu ve 32 - bitli adres yolu içeriyor ve 4GB doğrudan adreslenmesine izin veriyor. Aynı zamanda, kaliteli grafik de hızlı ve güçlü mikroişlemci gerektiriyor. Çağdaş monitörler 300.000'den fazla piksel içeriyor. Onlarda grafik özel yazılımla düzenleniyor, GUI (Graphical User Interface) ve çağdaş VGA (Variable Graphics Array) gibi. 80386'da yenilik belleğin çalışmasını düzenleyen özel donanım bileşiğidir - MMU (Memory Management Unit). Önceki modellerde bu sorun yazılımla çözülüyormuş.

1989 yılında "İntel" **80486** mikroişlemciyi tanıtmış. Bu mikroişlemci aslında 80386 mikroişlemcinin, 80387 yardımcı işlemcinin ve 8KB önbelleğin bir tümleşik devrede birleşimidir. Bu mikroişlemcinin çalışma frekansı 66 MHz'tir. Ancak aktarım hala yavaşmış ve 33 MHz'te yapılıyormuş. Çalışma frekansı 100MHz, aktarım hızı 33MHz ve 16KB önbelleği olan 80486DX4 verziyonun performansları yakın 60MHz frekanslı Pentium mikroişlemcinin performanslarına yakın olduğunu da not etmek istiyoruz.

Pentiyum - işlemcisi piyasaya 1993 yılında çıkmış. O dönemde Pentium işlemcinin farklı verziyonları 110 ile 150MİPS arasında yönerge çalıştırabiliyormuşlar. Başka daha özel karakteristikleri veri önbelleğin ve yönerge önbelleğin ayrılmasıdır, 4GB RAM belleği, 60 - 66MHz veriyol hızı olmasıdır. Daha büyük çalışma hızı sanal yazılımın daha gerçek görünmesini sağlıyor. Yönergeler kümesi multimedya genişlemesi olarak adlandırılan yeni yönergelerle genişlenmiştir. Önceki bilgisayar nesillerine kıyasen Pentiumun tam sayılarla çalışmak için iki mikroişlemcinin bulunması görünür bir fark yaratıyor. Bu özellik aynı zamanda iki yönergenin birbirinden bağımsız olarak gerçekleimesini sağlıyor. Bu çalışma şekline süperskalar teknoloji denir.

Pentium Pro mikroişlemcide yenilik iki önbellektir: 8 KB veri belleği ve 8 KB yönerge belleğinden oluşan 16 KB L1 (level one) bellek. Diğer önemli değişiklik üç yönergenin paralel çalışması için üç çalıştırma birimin olmasıdır. Pentium Pro mikroişlemcisi için Windows NT işletim istemi Windows 95'ten daha uygundur. Bu arada Windows NT 32 - bitli kodun kullandığını Windows 95'in 16 - bitli kodun kullandığını not edelim.

Pentium II'nin ortaya çıkmasıyla, "İntel" şirketi mikroişlemci üretiminde yepyeni bir yön sunmuş. Mikroişlemcinin tümleşik devre olarak anakartına doğrudan bağlanması yerine, o özel tasarlanmış plastik kutuda yerleşmiştir. Aynı zamanda, mikroişlemcinin bulunduğu tümleşik devrede 4KB önbellek bağlanmıştır. Bununla çalışmanın daha etkili olmasını, daha doğrusu daha hızlı çalışması sağlanmıştır. 1998 yılında "İntel" veriyollarda aktarım hızını 66 MHz'ten 100 MHz'e artırdı. Bu şekilde veriyolunun mikroişlemciden daha yavaş olduğu zaman meydana gelen dar boğaz efekti yokolmuştur. 1998 yılında "İntel" Pentium II Xeon olarak adlandırılan, Pentium II'nin yeni verziyonunu tanıtmış. İki bilgisayar arasında en büyük fark L2 önbelleğin 512 KB'yan 1 ya da 2MB'a yükselmesidir. Ayrıca, Xeon mikroişlemcisi, Pentium Pro mikroişlemciye belli benzerlilik olarak, 4 mikroişlemciyle bir sistem içinde çalışabilmesi için tasarlanmıştır.

Pentium III mikroişlemcinin temel özellikleri şunlardır: en yüksek çalışma frekansı 1 GHz, 32 KB L1 önbellek ve 256KB ya da 512KB L2 önbellek, yeriyol hızı 100 MHz.

2000 yılında **Pentium IV** modeli piyasaya çıktı. 1,3 ya da 1.4 frekansla çalışabiliyordu. Pentium IV'te ilginç olan şey L1 önbelleğin 32 KB'tan 8 KB'a azalmasıdır. L2 önbelleği aynı büyüklükte kalmıştır. Aluminyum bağlantıları yerine bakır kullanılmaya başlamış. Bakır daha iyi iletkendir ve miroişlemcilerin hızını artırıyor. Veriyolun hızı 133'ten 200MHz'e artmış.

3.2. Veri Türleri

İnsanın kullandığı bilgiler, dijital bir cihaza geçmeden ya da işletilmeden önce dijital şekline dönüşmeleri gerekiyor. Dijitalleşmiş veriler, ikili kodlar, yani sıfırlardan ve birlerden oluşmuş diziler tanımlıyor. Bir verinin kodlandığı ya da ifade edildiği bitler grubuna kod sözü deniliyor. Veriler ve kod sözleri arasında tek yönlu uyuşma vardır, daha doğrusu iki verinin aynı kod sözü olamaz. Mikroişlemcilerin çalıştırdığı veriler 2 temel gruba ayrılabilir: karakterler ve sayılar.

Karakterler grubuna sayılar, harfler ve metin yazılması için kullanılan alfanumerik işaretler giriyor. Klavyede her tuşla iki ya da fazla karakter yazılabilir. Karakterler önceden belirlenmiş kurallarla kodlanıyor ve bu arada herbir karakter yedi sıfırdan ya da birden oluşan tek kombinasyonuyla ifade ediliyor. Karakterler için sıkça kullanılan kod **ASCII** kodudur. Bu kodun bütün adı "American Standard Code for Interchange Information"dır ve "Veri değişimi için Amerikan standart kodu" anlamına geliyor. Bu kod tüm karakterlerin bilgisayar içinde bir aygıttan başka bir aygıta aktarım için ya da dış aygıtlar aracılığıyla (örneğin, klavye giriş için ve yazı-

cı çıkış için) verilerin girmesi ve çıkması için kullanılıyor. ASCII kodu 7 - bitlidir ve ona göre bu kodla toplam 2^7=128 işaret ifade edilebilir. Tablo 3.2'de ASCII kodlar verilmiştir.

kod	işaret	kod	işaret	kod	İşaret	kod	işaret	kod	İşaret	kod	işaret
20	boş yer	30	0	40	@	50	P	60	‘	70	P
21	!	31	1	41	A	51	Q	61	A	71	q
22	“	32	2	42	B	52	R	62	B	72	r
23	#	33	3	43	C	53	S	63	C	73	s
24	$	34	4	44	D	54	T	64	D	74	t
25	%	35	5	45	E	55	U	65	E	75	u
26	&	36	6	46	F	56	V	66	F	76	v
27	‘	37	7	47	G	57	W	67	G	77	w
28	(	38	8	48	H	58	X	68	H	78	x
29	)	39	9	49	I	59	Y	69	I	79	y
2A	*	3A	:	4A	J	5A	Z	6A	J	7A	z
2B	+	3B	;	4B	K	5B	[	6B	K	7B	{
2C	’	3C	<	4C	L	5C	/	6C	L	7C	|
2D	-	3D	=	4D	M	5D	]	6D	M	7D	}
2E	.	3E	>	4E	N	5E	^	6E	N	7E	~
2F	/	3F	?	4F	O	5F	_	6F	O	7F	

Tablo 3.2. Alfanumerik işaretler için ASCII kodlar

0'dan 1F'e kadar (on altılı) olan kodlar metnin aktarılmasından önce düzenlemek için kullanılan kontrol işaretleridir.

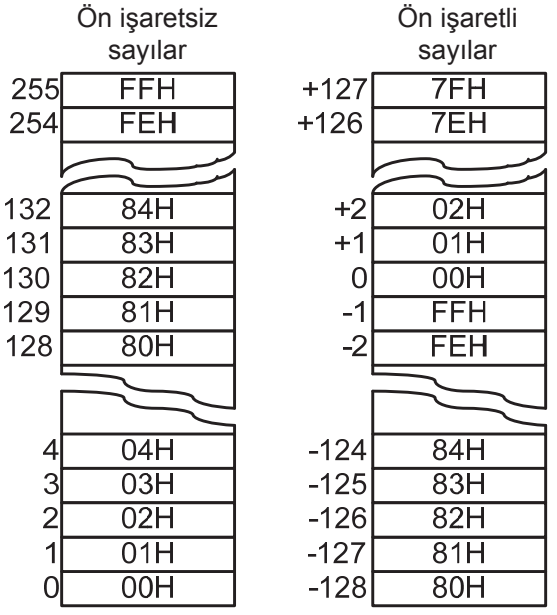
Örnek 3.1: Resim 3.1'de “Mikro” sözcüğü için ASCII kodu verilmiştir.

Alfanumerik işaretler	M	i	k	r	o
İkili onaltılı	01001101 4D	01101001 69	01101011 6B	01110010 72	01101111 6F

Resim 3.1 Mikro sözcüğünün ASCII kodla ifade edilmesi

Mikro sözcüğünü hafıza etmek için beş bellek konumu kullanılacaktır ya da sözcüğün her harfi için birer bellek konumu kullanılacak. Karakterlerden oluşan sözler aslında veri dizileri tanımlıyorlar. Kitabın devamında, yönergeler kümesini daha detaylı öğrendiğimizde, dizilerle çalışmayı büyük ölçüde basitleştiren, diziler için özel yönergelerin varolduğunu göreceğiz.

Sayıları ikili şekilde ifade etmek için iki kod türü kullanılıyor: ağırlıklı ve ardaşıl. Ağırlıklı kod için örnek, daha önce öğrendiğimiz doğal ikili sayı sistemidir. Onlu sayı sistemden ikili sayı sistemine dönüştürmeyi birinci konunun başında inceledik. Pozitif ikili sayının ikinci tümleyicinin hesaplanmasıyla negatif sayının ikili değeri elde ediliyordu. **Ön işaretli sayılar**la çalıştığımız zaman sayının en önemli biti işareti belirlemek için kullanılan bittir. Bu bit sıfır ise o zaman sayı pozitiftir, eğer bu bit bir ise o zaman sayı negatiftir. Ön işaretsiz sayılarla çalıştığımız zaman 8 bitle 256 sayı ifade edilebilir, 0'dan 255'e kadar. Ön işaretli sayılarla çalıştığımız zaman 8 bitle - 128'den +127'ye kadar sayılar ifade edilebilir. Resim 3.2'de ön işaretli ve ön işaretsiz 8 bitli sayılar verilmiştir. Tabloların satırlarında her sayının on altılı değeri verilmiştir, kenardan ise onların onlu sayı sisteminde değerleri yazılmıştır.



Resim 3.2. Ön işaretli ve ön işaretsiz sayıların onaltılı sayı sisteminde ifade edilmesi

Mikroişlemcinin işletirdiği veriler **büyüklüklerine** göre da ayrılabilir. 8 - bitli verilere bayt denir. 16 - bitli sayılar sözler adıyla biliniyorlar. Bir sözle ön işaretsiz sıfırdan +65535'e kadar sayılar ya da ön işaretli - 32768'den +32767'ye kadar sayılar ifade edilebilir. 32 - bitli veriler çift sözler olarak ifade ediliyor ve onların hafıza edilmesi için 4 bellek konumu gerekiyor.

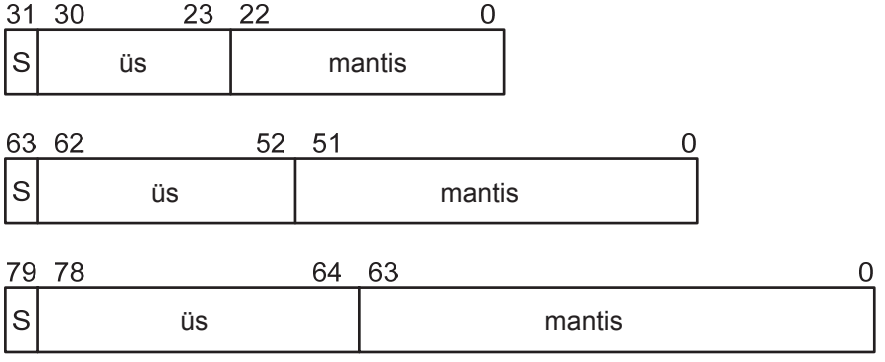
BCD kodu çok sıkça kullanılan ikili kodudur. Bu kodla kod sözleri ikiyle bölünerek kalanın yazılmasıyla elde edilmiyorlar. BCD kodunda onlu sayıdan her rakam 4 ikili rakamla ifade ediliyor. 4 bitle 2^4=16 kombinasyon yapılabilir, ancak onlardan sadece 10'unu, 0'dan 9'a kadar onlu sayıları ifade etmek için kullanıyoruz. Altı kombinasyon kullanılmadan kalıyor. Bu yöntem on altılı sayı sistemden ikili sayı sistemine

dönüştürmek için kullanılan yöntemle hemen aynı olur, sadece orada tüm 16 kombinasyon kullanılıyor. Örnek 3.2'de onlu sayı olan 1944 sayısının BCD ve ikili sayı sisteminde tanımlanması verilmiştir. Her iki ifadede 16 bit kullanılıyor

Örnek 3.2:
Onlu sayı sistemi: 1944
İkili sayı sistemi: 00000111100110000
BCD kodu: 0001 1001 0100 0100

16 bitle BCD'de 0'dan 9999'a kadar saylar tanımlanabilir, ikili sayı sisteminde ise 0'dan 65536'ya kadar sayılar temsil edilebilir. Buna göre ikili sayı sistemi BCD kodundan daha etkili olduğu açık görünüyor, ancak onlu sayı sisteminden BCD koduna dönüştüren elektrik devreleri, ikili istemine dönüştüren devrelerden çok daha basit ve hızlıdır.

Gerçek sayılar **kayan noktalı sayılar** adıyla da biliniyor. Tüm modern mikroişlemciler kayan noktalı sayılarla işlemler için özel sayısal yardımcı işlemci ve özel yönergeler kullanıyorlar. Resim 3.3'te ikili sayı sisteminde kayan noktalı sayıların üç formatı (şekli) gösterilmiştir. Onlar üç bölümden oluşuyorlar: işaret biti –S (sign), üs ve mantis.



Resim 3.3. Kayan noktalı sayıların formatı

Eğer sayı negatif ise o zaman işaret biti bir olacak, sayı pozitif ise o zaman işaret biti sıfır olacak. Üsü ve mantisi belirlemek için şu yöntem kullanılıyor:

1. Sayıyı ikili sayı sistemine dönüşmek;
2. İkili sayının normalizasyonu. Ondalık noktasından önce ikili sayının sadece en değerli biti gelene kadar nokta sola doğru taşınıyor. Üsü derecesi noktanın sola kaydığı pozisyonlar sayısına eşittir. Ondalık noktasından sonra gelen bitler mantisi tanımlıyor. Mantise sağ taraftan uygun formatı elde etmek için sıfırlar eklenebilir.
3. Üs, formatın temeline, dereceyi ikili sayı şeklinde katmayla elde ediliyor. Birinci kısa format için temel 7FH sayısıdır, ikinci format için 7FFH sayısıdır

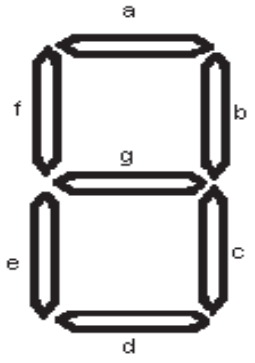
ve üçüncü format çin 7FFFH sayısıdır. Format ne kadar büyüksa o kadar gerçek sayı daha yükü doğrulukla ifade ediliyor.

Örnek 3.3: 100,25 gerçek sayısını kayan noktalı sayı şeklinde tanımlayacağız.

1. 100,25=1100100,01
2. 1100100,01=1,10010001x2^6
3. 110+01111111=10000101

Çözüm: işaret biti=0, üs=10000101,
mantis=10010001000000000000000

Ondalık sayı sistemin rakamlarını görsel şekilde göstemek için **yedi parçalı kod** kullanılıyor. Yedi parçalı görüntünün her alanı, resim 3.4.a'da gösterdiği gibi 7 bölümden oluşuyor.



a)

\	x	a	b	c	d	e	f	g	\
0	0	1	1	1	1	1	1	0	= 7EH
1	0	0	1	1	0	0	0	0	= 30H
2	0	1	1	0	1	1	0	1	\
3	0	1	1	1	1	0	0	1	\
4	0	0	1	1	0	0	1	1	\
5	0	1	0	1	1	0	1	1	\
6	0	1	0	1	1	1	1	1	\
7	0	1	1	1	0	0	0	0	\
8	0	1	1	1	1	1	1	1	\
9	0	1	1	1	1	0	1	1	\

b)

Resim 3.4. Yedi parçalı görüntü ve onlu sayı sistemden rakamların kodları

Her bölüm birçok LED diyottan oluşmuştur ve onların sayısı değişiyor. Bölümün (parçanın) yanması onun tüm LED diyotlarına mantıksal birimden bilgi gönderildiğinde meydana geliyor. Yedi parçanın pozisyonları a, b, c, d, e, f, g harfleriyle işaretleniyor. Resim 3.4.b'de onlu sayı sisteminden 0'dan 9'a kadar rakamların yediparçalı kodları verilmiştir.

Örnek 3.4: 2 ve 7 rakamlarının gösterilmesi için hangi bölümlerin yanması gerekiyor?
Çözüm: 2 rakamının gösterilmesi için a, b, g, e ve d bölümlerin yanması gerekiyor, 7 rakamı için a, b ve c bölümler yanmalıdır.

3.3. Mikroişlemcinin Genel Yapısı

Mikroişlemci dijital devrelerden oluşuyor: mantıksal devreler, kod çözücüler, yazmaçlar, çıkarma ve toplama devreleri, bellekler vb. Bu dijital devreler mikroişlemcinin yapısını oluşturuyor. Mikroişlemcinin yapısından onun işlevi, mikoişlemcinin farklı yönergelerin çalıştırabilme yeteneğine bağlıdır. Farklı üreticilerden ve farklı nesillerin mikroişlemcilerin farklı yapısı var, ancak donanım bileşenlerin 80%'i hemen tüm mikroişlemcilerde rastlanabilir. Mikroişlemci organizasyonu sadece oluştuğu parçaları adlandırmak değil, onların işlevini tanımak ve bir işlevi bütüne bağlandıkları şekli tanımak demektir. Bir genel mikroişlemcinin genel yapısını incelemeden önce **yönergelerin çalıştırmasını**, onları aşamalara ayırarak, açıklamamız lazım. Yönergeyi beş aşamaya ayıracağız:

- İşlem kodunun bellekten, mikroişlemcinin yönerge yazmaçına aktarılması.
- İşlem kodunun yönetim biriminde çözülmesi.
- Verileri bellekten mikroişlemcinin genel yazmaçlarından birine aktarılması.
- Yönergenin aritmetik mantık biriminde efektif çalıştırılması
- Elde edilen sonucun mikroişlemcinin bir genel yazmacında ya da bellek konumunda yazdırılması.

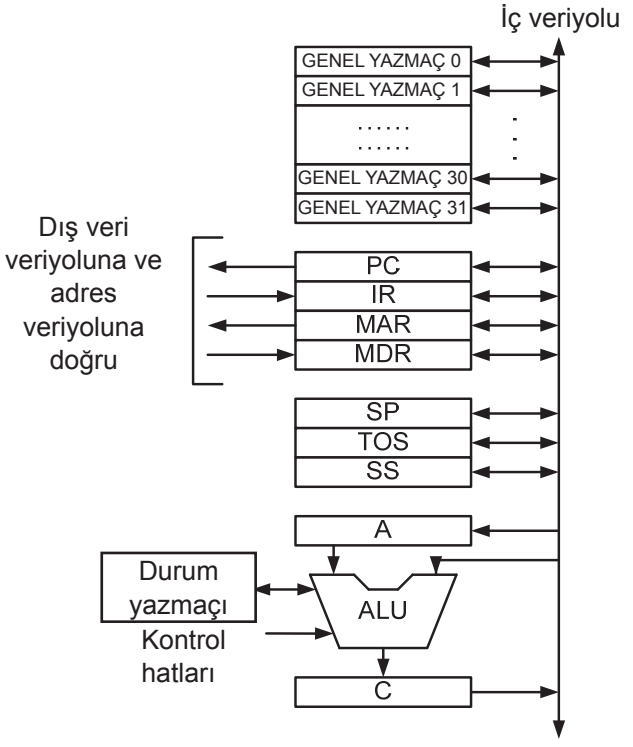
Beş aşamanın herbiri mikroişlemcinin donanımından bir parça etkinleştiriyor. Yani fonksiyonların donanım ayırımı yapılmıştır. Yönergelerin ayrışması devam edebilir, ancak en iyisi aralarında donanım karışması meydana gelmememsidir. O şekilde çalışma hızı yükseliyor, fabrikalarda üretim bandına benzer olarak.

Ancak yönergenin çalıştırılmasını birkaç aşamaya ayırdıktan sonra birçok soru ortaya çıkıyor. Mikroişlemci çalıştırılması gereken sıradaki yönergeyi nasıl buluyor? Hangi yönergenin söz konusu olduğunu nasıl biliyor? Yönergeyi aritmetik mantık biriminde nasıl çalıştıracak? İşlem sırasında meydana gelen ara sonuçlar verileri ve elde edilen sonuçları nerede yerleştireceğini nasıl biliyor? Bu ve diğer soruların cevabını mikroişlemcinin yapısını inceledikten sonra alacağız. Resim 2.3'te genel mikroişlemcinin temel blok modeli gösterilmiştir. Mikroişlemcinin oluştuğu parçaların yazmaçlar, aritmetik mantık birimi ve yönetim birimin olduğunu bir kez daha hatırlatıyoruz.

Mikroişlemcinin yapısını daha kolay anlamak için her bileşen parçanın yapısını ayrı ayrı göreceğiz ve onların bağlılığını açıklayacağız.

3.3.1.Mikroişlemcide Yazmaçlar

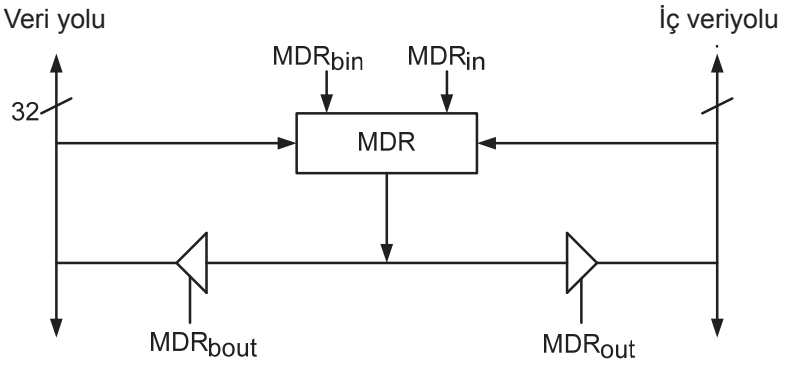
Yazmaçlar mikroişlemcinin içinde bulunan hızlı bellek konumlarıdır ve veriler, yönergeler ve mikroişlemicinin durumu için bilgiler saklıyor. Yazmaçların **genel ayırımı** genel yazmaçlar ve özel amaçlı yazmaçlar olmak üzere ikiye ayrılıyor. Resim 3.5.'te daha önemli yazmaçlar ve onların mikroişlemcinin yoluyla, dış veri veriyolu ve adres veriyoluyla bağlanma şekli gösterilmiştir.



Resim 3.5. Mikroişlemcinin genel modelinde yazmaçlar

Özel amaçlı yazmaçlar kişisel (programcının erişimi var) ya da sistemik olabilir. **Özel amaçlı yazmaçların** başka ayırımı işlevine göre yapılır. Bu anlamda üç grup vurgulayabiliriz: bellekte verilerin okunması ve yazılması için yazmaçlar (MAR, MDR), yönerge aktarım yazmaçları (PC, İR) ve yığın bellekle çalışma için yazmaçlar (SP, TOS, LV). Her yazmaç için iç veriyoluyla ya da dış veri veriyolu ya da adres veriyoluyla bağlanmasını sağlayan kontrol sinyaller bulunuyor. Veri veriyoluyla bağlanmak için kontrol sinyallerin, yazmaca verinin yazılmasına ya da yazmaçtan verinin okunmasına gerektiğine bağlı olarak in ya da out ön ekleri vardır. Bu ön eklerin yanında b harfi de bulunuyorsa, o zaman dış sistemik veri veriyolu (İngilizce bus=veriyolu) söz konusu olduğu demektir.

Resim 3.6.'da MDR yazmaçın veriyoluyla bağlanması ve onun kontrol sinyalleri gösterilmiştir. Veri hatlarına arabelleklenme yapıldığını görebiliriz ve bunun için üç durumlu arabellekler kullanılmıştır.



Resim 3.6. Bellek veri yazmaçın kontrol sinyalleri

Devamda daha önemli özel amaçlı yazmaçların işlevlerini açıklayacağız.

- İçeriğinde **program sayacı** (PC=Program Counter) olmayan mikroişlemci yoktur. Program sayacı sıradaki yönergenin adresini içeriyor. Yönergenin 32 bit büyük olduğunu tahmin edelim. Sıradaki yönergenin adresini elde etmemiz için verilen anda çalıştırılan yönergenin adresine dört değerini eklememiz gerekiyor. Bunun sebebi bir bellek konumunun bir baytlık bilgi hafıza etki ettiğidir ve ona göre 32 - bitlik yönergenin hafıza edilmesi için dört bellek konumu gerekecetir. Program sayacının üç kontrol sinyali vardır: $PC_{bout,}$ PC_{out} ve $PC_{in.}$ Mikroişlemci sıradaki yönergeyi aradığı zaman, program sayacın içeriği PC_{bout} sinyalin aktiflenmesiyle adres yoluna gönderiiliyor. Program sayacın içeriğini sadece yönetim birimi değiştirebilir ve böyle durumda yeni içerik iç veriyolu aracılığıyla aktarılıyor ve onun yazılması için PC_{in} sinyalin etkilenmesi gerekiyor. Program sayacın içeriği, PC_{bout} ve PC_{out} sinyallerin aktiflenmesiyle dış adres veriyoluna ve aynı zamanda iç veriyoluna taşınabileceğini de not ediyoruz.
- PC yazmacın içeriği sistem adres yoluna gönderiliyor, belleğin yönergeyi veri veriyoluna yerleştirmesi bekleniyor ve ondan sonra yönerge **yönerge yazmacına** IR (Instruction Register) yazdırılıyor. Veriyollarınla basit bağlanma şeklinden dolayı iki kontrol sinyali kullanılıyor IR_{bin} ve IR_{out}. Biri bellekten yazmaca yönerge yazılması gerekince kullanılıyor, diğeri ise yönergenin yönetim birimine taşınması gerektiğinde kullanılıyor.
- MAR (Memory Address Registar) ya da **bellek adres yazmacı** mikroişlemcide işlenmesi gereken verinin adresini saklıyor. Bazı yönergeler içinde, işlenmesi gereken verinin alınacağı bellek konumun adresini içeriyor. Bu adres MAR_{in} kontrol sinyalin etkinleştirilmesiyle bellek adres yazmacına giriyor, adresin belleğe aktarılması gerekince, bu adres MAR_{bout} kontrol sinyalin aktiflenmesiyle adres veriyolu aracılığıyla belleğe gönderiliyor.

Bellek adres yazmacın içeriği MAR_{out} sinyalın aktif edilmesiyle iç veriyoluna aktarılabilir.

- MDR (Memory Data Register) ya da **bellek veri yazmacı** bellekte yazılması ya da bellekten okunması gereken veriyi saklıyor. Bu yazmacın veriyollarıyla bağlanması iki yönlüdür ve bundan dolayı dört kontrol sinyali gerekecek: $MDR_{bin,}$ $MDR_{bout,}$ $MDR_{in,}$ $MDR_{out.}$ Bellek veri yazmacı resim 3.6'da gösterilmiştir.
- Yığın bellekle çalışma yazmaçlarını daha geç tanıyacağız ve detaylı olarak yığın bellek ile onda okuma ve yazma süreçleri açıklanacaktır.

Genel amaçlı yazmaçlar sadece iç veriyoluyla bağlıdırlar. Resim 3.5.'teki mikroişlemci modeli G_0'dan G_{31}'e kadar işaretlenmiş 32 genel amaçlı yazmaç içeriyor. İşaretlemede kullanılan G harfi genel anlamına gelen İngilizce general sözcüğünün ilk harfinden geliyor. Her genel yazmaç için G_{xin} ve G_{xout} olarak işaretlenen ikişer kontrol sinyali bulunuyor. Kontrol sinyalleri genel yazmaçların girişlerin ve çıkışların yetenek sağlamaları için kullanılıyor. Bu kontrol sinyallerin aktifleştirilmesiyle mikroişlemci hangi yazmaçtan veri okuyacağını ve hangi yazmaca veri yazacağını seçiyor. Kontrol sinyallerin kullanımını toplama yönergesiyle göstereceğiz.

add R_d,R_{s1},R_{s2}

Bu yönergeyle R_{s1} ve R_{s2} genel yazmaçların içerikleri toplanıyor ve toplam R_d yazmacında yazılıyor. s harfi kaynak anlamına gelen İngilizce source sözcüğünden geliyor, d harfi ise İngilizce amaç, hedef anlamına gelen destination sözcüğünden geliyor. Örneğin, 5 ve 7 sıra numaralı yazmaçların içeriklerini toplamak istiyoruz ve toplamı 9 numaralı yazmaca yerleştirmek istiyoruz. Demek ki şu yönergenin çalıştırılması gerekiyor:

add G9, G5, G7

Sadece bir iç veriyolu olduğundan dolayı, bir kaynak yazmacın içeriği geçici yazmaç A'ya taşınmalıdır. İkinci yazmacın içeriğini iç veriyolu A'ya geçiriyoruz ve onları topluyoruz. **Kontrol hatların etkinleştirilmesini** üç aşamaya ayırabiliriz:

1. G5 yazmacın içeriğini iç veriyoluna geçirmek için G_{5out} sinyalini aktifleştiriyoruz. Aynı zamanda iç veriyolundan içeriği A yazmacına geçirmek için A_{in} sinyalini aktifleştiriyoruz.
2. Şimdi G7 yazmacın içeriği G_{7out} sinyalın etkinleştirilmesiyle iç veriyoluna aktarılması gerekiyor. Sıradaki adım aritmetik mantık biriminin kontrol sinyallerin aktif edilmesi olacak. Kontrol sinyallerin durumu tablo 3.4'te beşinci satıra uygun olması gerekiyor. C_{in} sinyalın aktifleşmesiyle toplam C yazmacına giriyor.

3. Son adım, C yazmacının değerini G_9 yazmacına yazılmaktan oluşuyor. Bu aktarma C_{out} ve G_{9in} sinyalerin aynı zamanda etkinleşmsiyle sağlanıyor.

Tablo 3.3'te her adı için kontrol sinyallerin aktifleşme sıralaması verilmiştir.

adımlar	Kontrol sinyalleri
1	$G_{5out;}$ A_{in}
2	$G_{7out;}$ ALU:$F_{o,}$ $F_{1,}$ ENA, ENB; C_{in}
3	$C_{out;}$ G_{9in}

Tablo 3.3. İki genel yazmacın içeriklerini toplama işlemi için konrol sinyaller

3.3.2. Aritmetik Mantık Birimi

Mikroişlemci bilgisayarın "beyinidir", **aritmetik mantık birimi ise mikroişlemcinin "beyinidir".** Aritmetik mantık birimi aracılığıyla yönergeler kümesinden yönergeler çalıştırılıyor. Daha kompleksli işlemler doğrudan yapılmıyor, daha basit işlemlerin kombinasyonuyla yapılıyorlar.

Aritmetik mantık biriminin mikroişlemcinin yazmaçlarıyla bağlanma şekli, resim 3.5.'te gösterilmiştir. Resimde görüldüğü gibi aritmetik mantık birimi mikroişlemcinin dış veriyoluyla bağlı değildir, sadece iç veriyoluyla bağlıdır ve onun aracılığıyla yazmaçlardan veriler arıyor. Ayrıca, resimde aritmetik mantık birimlerin iki geçici, çalışma yazmaçları gösterilmiştir, A ve C. A yazmacı birimin üstünde bulunuyor ve verinin birime girmeden ve işletilmesi başlamadan önce geçici olarak korunması için kullanılıyor. C yazmacı elde edilen sonucun, mikroişlemcinin genel ve özel yazmaçlarda yerleşmeden önce, geçici olarak korunması için kullanılıyor.

Aritmetik mantık birimi doğrudan **durum yazmacı** olarak adlandırılan özel bir yazmaçla bağlıdır. Bu yazmacın bitlerine **bayraklar** deniyor (İngilizce flag). Bayraklar durumun göstericisidir, daha doğrusu elde edilen son sonuçta bitlerin durumu için bilgi veriyor. Bayrağın kaldırılması yani aktifleştirilmesi için belli bir koşulun yerine getirilmesi gerekiyor. Her yeni mikroişlemci nesiliyle bayrakların sayısı artıyor. Ancak hemen her mikroişlemcide rastlanan en önemli bayraklar şunlardır:

- Sıfır Z bayrağı (İngilizce – zero) sıfıra eşit sonuç elde edildiğinde aktifleşiyor.
- S bayrağı (İngilizce – sign) negatif sonuç elde edilince aktifleşiyor.
- C bayrağı (İngilizce - carry) ikili sayıda en değerli pozisyonda aktarma meydana geldiği zaman aktifleşiyor. Örneğin iki 8 - bitli sayı toplarsak, se-

kizinci pozisyondan dokuzuncu pozisyona aktarma olursa C bayrağı aktifleşecek. 8 - bitli verilerde dokuzuncu pozisyon yok ve aktarma biti C bayrağında yazılacak.

- O bayrağı (İngilizce - overflow) taşma sırasında aktifleşiyor, yani öngörülen sayıdan daha küçük ya da daha büyük değer elde edilince etkinleştiriliyor. O bayrağı sadece ön işaretli sayılarla çalıştığımız zaman kullanılıyor. Resim 3.2'ye baktığımız zaman bu bayrağın +127'den daha yüksek ya da - 128'den daha küçük sonuç elde edilince aktif edlidiği sonucuna varabiliriz. Ön işaretsiz sayılarda meydana gelen taşma için gösterge olarak aktarma bayrağı C kullanılıyor

Aritmetik mantık biriminin yapısıyla, birinci konuda bileşik devreleri incelediğimizde tanıdık ve aynısı resim 1.20'de gösterilmişti. Tüm mantıksal devrelerin, iki veri girişinin A ve B ve işlem seçimi için F0 ve F1 kontrol sinyalleri fonksiyonları açıklanmıştı. Ancak, bu iki **kontrol sinyalleri** dışında dört kontrol sinyali daha var: ENA (enable A), ENB (enable B), INVA (invert A) ve INC (increase). ENA ve ENB sinyalleri veri girişlerin çalıştırışmasını sağlıyor. INVA sinyaliyle A girişindeki bit eviriliyor. INC sinyali ikili sayıya bir eklemek için kullanılıyor. Altı kontrol sinyaliyle 64 farklı kombinasyon yapılabilir, ancak hepsi kullanılmıyor. Daha ilginç kombinasyonlardan bazıları tablo 3.4'te verilmiştir. Verilen fonksiyonlardan aritmetik toplama en çok kullanılan işlemdir. Örneğin, A+1 işleminde, ENB sıfıra eşittir çünkü B veri girişine ihtiyaç yok, ancak sinyal bir olacaktır, çünkü onunla bir ekliyoruz. INVA ve INC sinyalleri bire eşit olursa, o zaman A değerinin ikinci tamamlayıcısını hesaplıyoruz, yani A'nın negatif değerini –A'yı hesaplıyoruz. B veri girişinin etkin yaparsak o zaman B+(- A)=B - A fonksiyonunu elde edeceğiz, birinci veri girişini ikincisinden eksilteceğiz. Belki bazı kimse tablo 3.4'te ilk iki fonksiyona, artimetik mantık biriminin çıkışında, girişte ne varsa yine onu elde edecek fonkisyonlara ne gerek varsa onu sorabilir. Bu fonksiyon sıkça bir yazmaçtan başka yazmaca veri aktarımı sırasında kullanılıyor, çünkü yazmaçlar aralarında bağlı değildir. Örneğin, veri bir genel yazmaçtan başka genel yazmaca doğrudan geçemez. Aralarında aritmetik mantık birimi aracılık yapıyor. Birinci yazmacın içeriği aritmetik mantık birimin girişine getiriliyor, aritmetik mantık birimin çıkışından ise hedef olan ikinci yazmaca gönderiliyor. Tablonun ilk iki satırından, veri girişlerden birini aritmetik mantık biriminin çıkışına aktarmak için mantıksal toplama (YADA) işlemin kullanıldığı görülüyor ve sadece bir etkileştirme sinyali aktiftir.

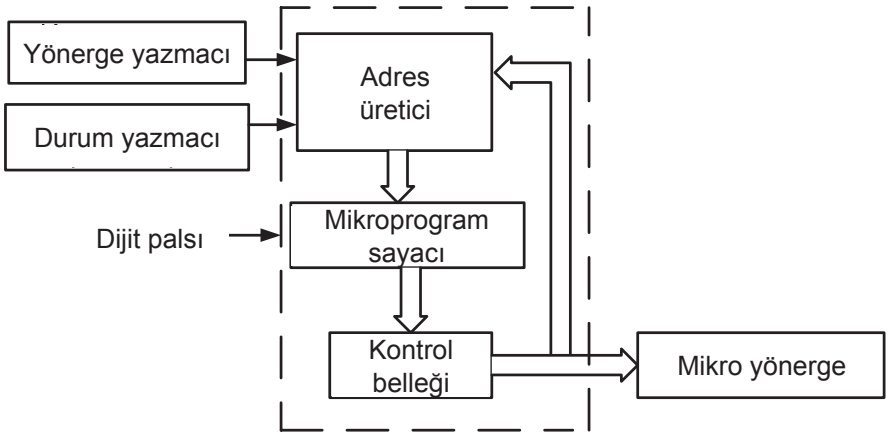
F0	F1	ENA	ENB	INVA	INC	Fonksiyon
0	1	1	0	0	0	A
0	1	0	1	0	0	B
0	1	1	0	1	0	Olumsuz A
1	0	1	1	0	0	Olumsuz B
1	1	1	1	0	0	A+B
1	1	1	1	0	1	A+B+1
1	1	1	0	0	1	A+1
1	1	0	1	0	1	B+1
1	1	1	1	1	1	B - A
1	1	0	1	1	0	B - 1
1	1	1	0	1	1	- A
0	0	1	1	0	0	A □ B
0	1	1	1	0	0	A □ B
0	1	0	0	0	0	0
0	1	0	0	0	1	1
0	1	0	0	1	0	- 1

Tablo 3.4. 1 - bitli aritmetik mantık birimin doğruluk tablosu

3.3.3. Yönetim Birimi

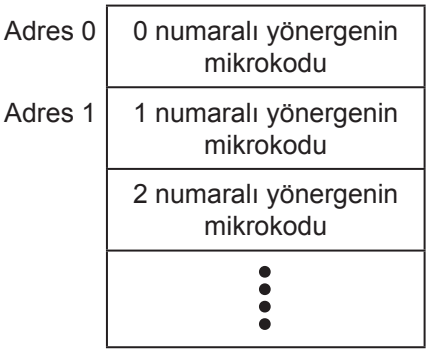
Yönetim biriminin en önemli işlevi, başlatılan program bitene kadar yönergeleri birer birer çalıştırmasıdır. Yönetim birimi yönergelerin çalıştırılmasını yazmaçlar, aritmetik mantık birimi, bellek ya da dış aygıtlar için kontrol sinyalleri yaratarak yapıyor. Resim 3.7.'de yönetim biriminin temel blok diyagramı verilmiştir.

Yönetim birimin girişine yönerge yazmacın içeriği, durum yazmacı ve dijit palsı getiriliyor. Yönergenin ilk sekiz biti işlem kodunu tanımlıyor ve bu kod yönergenin tanıtılması için kullanılıyor. Yönergelerin dört gruba ayrıldığını hatırlayalım: aktarma yönergeleri, aritmetik, mantıksal yönergeler ve program akışı kontrol yönergeleri. Kontrol yönergeleri koşullu olabilir. Onların çalıştırılması için durum yazmaçlardaki bayrakların durumu için bilgi gerekiyor. Yönerge yazmacın içeriği ve durum yazmacı aslında mikroprograma erişim için adres üreten adres üreticinin girişleridir.



Resim 3.7. Mikroişlemci genel modelin yönetim birimi

Yönetim birimin en önemli parçası **mikroprogramdır**. Mikro program mikroişlemcinini içinde, kontrol belleği olarak adlandırılan özel ROM belleğinde bulunuyor. Mikroprogram mikrokodlardan oluşuyor. Mikrokodların sayısı, mikroişlemcinin yönergeler kümesindeki yönergelerin sayısına eşittir. Her yönerge için birer mikrokod vardır. Mikrokodlar mikro yönergelerden oluşuyor. Bir mikrokotta mikro yönergelerin sayısı, yönergenin çalıştırılmasının ayrıldığı aşamalar, adımlar sayısına eşittir. Her adım için birer mikro yönerge vardır. Mikro programın böyle doğrusal organizasyonu resim 3.8'de gösterilmiştir.



Resim 3.8. Mikroprogramın organizasyonu

Mikroişlemci kontrol sinyalleri üretiyor. Her kontrol sinyali için **mikro yönergede** özel bit bulunuyor. Kontrol sinyalin ve onun mikro yönergeki bitin aynı değerleri vardır, mantıksal sıfır ya da bir. 32 genel yazmacımız varsa ve her yazmaç için ikişer kontrol sinyali gerekirse, o zaman mikro yönergeden 64 bit, genel sinyalin kontrolu için olacak. Mikro yönergenin içeriğinde aritmetik mantık biriminin kontrol bitleri ve özel amaçlı yazmaçlar da giriyor.

Mikro yönerge 90'dan fazla bit içerebiliyor. Tüm mikro yönergelerin aynı büyüklükte olduğunu vurgulamak gerekiyor. Bir mikroişlemci nekadar büyükse o kadar yönergelerin çalıştırılması daha hızlıdır. Ancak, aynı zamanda, daha büyük mikro yönerge daha fazla sayıda kontrol hatları ya da donanım harcaması demektir.

Tablo 3.3.'te add G9, G5, G7 yönergenin çalıştırılmasında tüm adımlar için kontrol sinyalleri verilmişti. Üç adım olduğu için bu yönergenin mikrokodu üç mikro yönerge içerecek. Birinci mikro yönerge sırasında yüksek seviyede sadece G_{5out} ve A_{in} bitleri olacak, ikincide $G_{7out,}$ ALU, $F_{o,}$ $F_{1,}$ ENA, ENB, C_{in} ve üçüncüde C_{out} ve $G_{9in.}$ Mikro yönergenin tüm kalan bitleri sıfıra eşit olacak.

Bir mikroişlemcinin süresi 1 dijit palsı kadardır. Bundan dolayı, dijit palsı yönetim biriminin girişinde getiriliyor. Onun aracılığıyla aslında mikroprogramda mikro yönergelerin sayması yapılıyor.

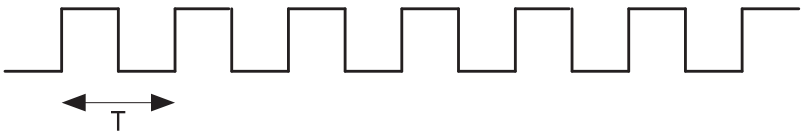
3.4. Mikroişlemcilerin Ortak Özellikleri

Donanım açısından mikroişlemci çok basit işlemler yapabilir ve onları çok büyük hızla gerçekleştiriyor. Kopmleksli yönergeler fazla basit yönergelerin kombinasyonuyla elde edilir. Mikroişlemcinin hızı MIPS (bir saniyede milyonlar yönergeler) ile ifade ediliyor. Ancak MIPS, bilgisayarın toplam hızına etkileyen sadece bir bileşendir. Mikroişlemcinin yönergeleri hızlı çalıştırması için, yönergeleri işlemciye hızlı aktarılması gerekiyor, aktarma hızı ise veri yolun ve belleklerin çalışma hızına bağlıdır. Mikroişlemcinin hızını etkileyen birkaç etken (faktör) sayacağız, ancak onların dışında başka faktörler de var ve onları daha geç inceleyeceğiz.

Çalışma frekansı

Mikroişlemcinin hızı 1 saniye içinde yapılan işlemler sayısı olarak ifade edilebilse, mikrobilgisayarın hızını en basit şekilde belirleyebiliriz. Ancak, farklı işlemlerin farklı zaman sürdükleri sorun yaratıyor. Mikroişlemcinin hızı herzlerle [Hz] ifade ediliyor.

Mikroişlemcinin içinde (ancak onun dışında da olabilir) kuvars kristali yerleştirilmiştir ve bu kristal devamlı aynı hızla dörtgen tetikleri yaratıyor. Bu kristale pals üretici deniyor ve mikroişlemcide dijit palsları veriyor. Bu tetiklerin frekansından mikroişlemcinin çalışma hızı bağlıdır, onunla beraber bilgisayarın hızına da etkiliyor.



Resim 3.9. Dijit palsın üreticiden çıkış sinyali

Periyodik dörtgen tetikler dizisinin temel özelliği periyottur (bir salınımın sürdüğü zaman). Periyod frekansın evrik değeri olarak hesaplanıyor. Frekans aslında bir saniyede kaç periyodun olduğunu gösteriyor.

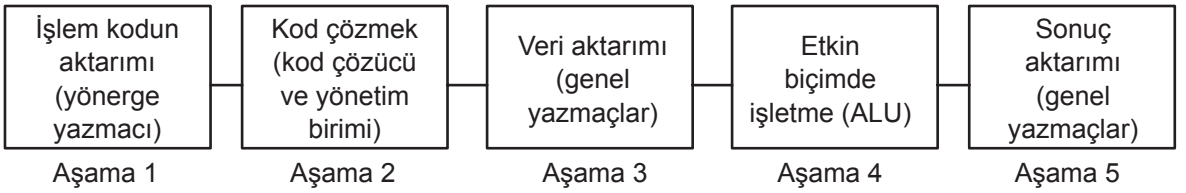
Sekizbitli mikroişlemcide kuvars kristalı saniyede 1.000.000 ile 4.000.000 arasında tetik yayıyor (1MHz'ten 4MHz'e kadar). Pentium 3'te bu frekans 1GHz değerindedir. Bugünkü mikroişlemcilerin çalışma frekansı 2 ile 4 GHz arasındadır. Farklı işlemler farklı zaman sürdüğünü önceden söylemiştik. Örneğin, bir sekizbitli verinin RAM belleğinden mikroişlemciye aktarımın gerçekleşmesi için üç pals gerekiyor, iki sayının aritmetik toplamı için dört pals gerekiyor. Aklımızda olması önemli olan tüm işlemlerin sürdüğü zaman tam sayı dijit palsıyla ifade edilmesidir. Verilen bir işlem 3.2 pals sürebilir, ancak mikroişlemci bu işlem için 4 palsın harcandığını sayıyor. Demek ki 0.8 pals borcuna harcanmıştır. Bu aktarım şekline senkron aktarım denir. Eğer her aygıtın işlemlerin sürdüğü zaman için ayrı zaman ölçmesi olsayadı (palsların farklı sürdüğü zamana düşünülüyor), o zaman çok kolay karışıklık meydana gelebilir. Mikroişlemci her aygıtın hızını hafıza etmesi gerekiyor ve onları senkrone etmesi gerekiyor. Bu da büyük zaman kaybına yol açıyor.

Verilerin Büyüklüğü

İşlenilen verilerin büyüklüğüne göre, mikroişlemciler 8, 16, 32, 64 ve 128 - bitlik mikroişlemcilere ayrılıyor. Daha büyük veriler işletebilen mikroişlemciler daha güçlüdür. Ancak, tabii ki daha güçlü mikroişlemci elde etmek için donanım kaynaklarına daha fazla yatırım gerekiyor. Örneğin, 8 - bitli mikroişlemcilerde veri aktarımı için 8 pin bulunuyor, anakartında 8 veri hatları var, mikroişlemcide her genel yazmaç sekizer flip flop içeriyor. 16 - bitli mikroişlemcide tüm bu büyüklükler 2 misli daha büyüktür. Örneğin, eğer 16 - bitli mikroişlemcinin sadece 8 veri pini varsa, o zaman 16 - bitli verilerin iki kezden aktarılmaları gerkecek, bu da zaman kaybı anlamına geliyor.

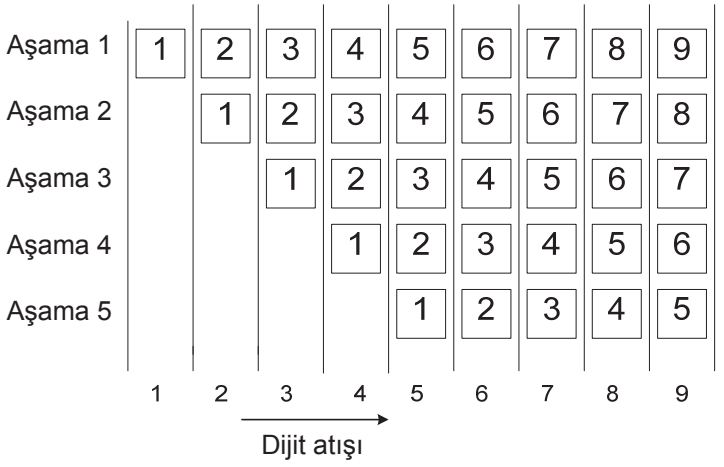
Akış Kavramı (PIPELINE)

Pipeline kavramı, mikroişlemcide birçok yönergenin akışlı, ardaşıl çalıştırılması demektir, ancak her yönergenin farklı aşamada olması gerekiyor. Resim 3.10'da bir yönergenin çalıştırılması için gereken beş aşama ve her aşama için gereken donanım bileşenleri gösterilmiştir.



Resim 3.10. Mikroişlemcide yönergenin çalıştırılması aşamaları

Resim 3.11.'de karelerle yönergeler işaretlenmiştir, kare içinde bulunan sayı yönergenin sıra numarasıdır. Bu şekilde yönergelerin çalıştırılmasında ardaşıllık kolayca görünebiliyor.



Resim 3.11. Akışlı kavramda aşamaların zamana bağlılığı

2 numaralı yönergenin çalıştırılması için, önce 1 numaralı yönergenin çalıştırılması gerekiyor. 3 numaralı yönergeden önce 2 numaralı yönerge çalıştırılıyor vs. Birinci dijit palsında 1 numaralı yönerge aşama 1'de dir, yani yönerge bellekten mikroişlemciye geçiriliyor. Yönergeler son aşamaya - elde edilen sonucun yazılmasına kadar, bir aşamadan sonraki aşamaya geçiyor. Resim 3.11'den her aşamanın bir pals kadar sürdüğü görünüyor (pals üreticisine bak). Buna göre 9 pals içinde 5 yönerge çalıştırılacak. Pipeline olmasaydı bir yönerge için 5 pals gerekecekti, diğer yönergeler ise bekleyecekti. 5 yönergenin çalıştırılması için toplam olarak 5 pals · 5 yönerge = 25 pals gerekecekti.

CISC ve RISC

İlk mikroişlemciler CISC'te (Complex Instructions Set Computer) üretilmiştir. Bu anlayış çok büyük yönergeler kümesi ve mikroişlemcinin çok kompleksli işlemlerin yapması gereğiyle karakterize ediliyor. Mikroişlemcinin çalıştırabileceği yönerge sayısına göre mikroişlemcinin gücü belirleniyormuş.

Mikroişlemcinin ortaya çıkmasından bugüne kadar onların hızı sürekli artıyor. Ancak, hızın artması ve işlemler sayısının artması birbirine aykırı olan gereklerdir. Şu soru ortaya çıkıyor: daha iyi mikroişlemci elde etmek için hangisinden vazgeçmeliyiz. Detaylı araştırmalarla çoğu mikroişlemcilerde bütün yönergele kümesinden en çok 20 %'si uygulandığı sonuca varılmış. Böyle araştırmadan sonra küçük yönerge kümeli çok hızlı mikroişlemcilerin yapılması fikri getilirmiştir. Bu şekilde RISC (Redused Instruction Set Computer) anlayışına temellenen mikroişlemci yapılmasında yeni yön yaratılmıştır. Bu anlayışa göre basit mikroişlemcilerle (genelde az sayıda transistörlerle) çok büyük hız elde ediliyor. RISC kavramı programcıya yaratıcılık açısından geniş alan bırakıyor – daha basit yönergelerden daha kompleksli yönergeler yaratsın. Basit şekilde yönerge sayısının azalması söz konusu değil, sadece onların rasyonelleşmesi yapılıyor.

Bugün RISC mikroişlemcilerin yapısının temelinde birkaç prensibin bulunduğu sayılıyor:

- Bir yönergenin çalıştırılması için palslar sayısının azalması.
- Kod çözülmenin basitleşmesi için yönergenin sabit formatta kullanılması;
- Ana belleğe erişim için yönergeler sayısının azalması;
- Mikroişlemcinin tanıyabileceği kodlar sayısının azalması.

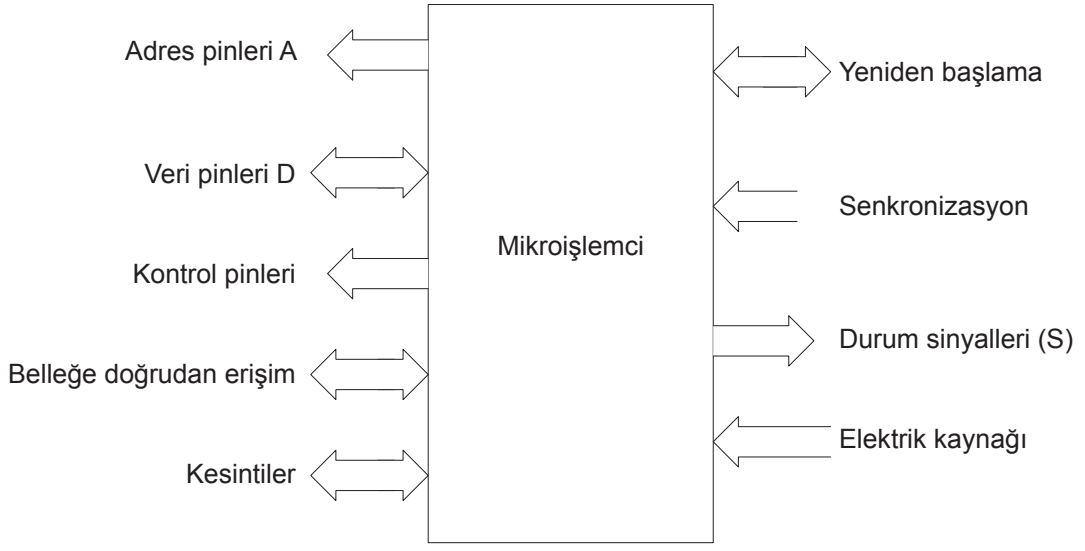
Bu prensiplerin büyük kısmı donanım seviyesinde uygulamıştır. RİSC kavramı paralel hatların daha büyük boyutta kullanılmasını sağlıyor ve kod çözümü için devrelerin büyük payı var.

3.5. Mikroişlemcinin Pin - diyagramı

Pinler (iğneler) tümleşik devrenin iç hatların uzantılardır ve ondan sonra ana kartın hatlarıyla bağlanıyorlar. Resim 3.12'de mikroişlemcinin genel modelin pin - diyagramı gösterilmiştir. Pin - diyagramı, pinlerin sıralamasını ve onların işaretlerini veriyor. Pinlerin işaretleri, kısaca işlevlerini betimleyen İngilizce sözcüklerin kısaltmalardan geliyor.

Pinler birkaç gruba ayrılıyor.

- A işaretli pinler **adres pinleridir**. Onlar her zaman mikroişlemci için çıkış pinleridir, bellek yongaları ise giriş pinleridir. Adresin bellekten bir konumun ya da dış aygıtın seçimi için tek ikili kombinasyonu olduğunu hatırlayalım. Adres hatların toplam sayısından mikroişlemcinin adres alanının büyüklüğüne bağlıdır. Adres alanın büyüklüğü, adres pinlerin toplam sayısı n ise, 2^n ilişkisiyle hesaplanıyor.



Resim 3.12. Mikroişlemci genel modelin pin diyagramı

- D işaretli pinler **veri pinleridir**. D harfi DATA İngilizce sözcüğünün ilk harfidir ve veri demektir. Veri pinleri mikroişlemcinin veri yazdığına ya da veri okuduğuna bağlı olarak giriş - çıkış pinleridir. Veri pinlerin sayısı 8, 16, 32 ya da 64 olabilir ve büyüklükleri işletilen verilerin büyüklüğüne bağlıdır.

- Kontrol pinlerin farklı isimleri olabilir ve herbirinin ayrı işlevi vardır. $\overline{WR}$ (write) ve $\overline{RD}$ (read) pinleri yazma ve okuma işlemleri arasında seçim yapıyorlar. Onların olumsuzlanması mantık sıfırda aktif olduklarını gösteriyor. IO/$\overline{M}$ (input output/$\overline{memory}$) pini giriş - çıkış aygıtların ve bellek aygıtı arasında iletişim aygıtın seçimi için kullanılıyor.

- Kontrol pinleri dışında, **durum pinleri** de mevcuttur ve S_0 ve S_1 ile işaretleniyorlar. Durum pinleri aktarılan verilerin ve aktarma için kullanılan aygıtın önemi için bilgi veriyorlar.

- **Kesinti** pinleri INTR (interrupt) ve INTA (interrupt acknowledge) pinleridir. Kesintiler dış aygıtların ve mikroişlemcinin iletişimi için özel düzenlenmiş mekanizmalardır. Dış aygıtın çalışmasında hata meydana gelirse, o zaman INTR pinin aracılığıyla mikroişlemciye kesinti araması gönderiliyor. Eğer mikroişlemci o anda çalışan programın kesilmesine izin verirse, o zaman kesinti onaylama pini INTA etkinleşiyor. Ondan sonra meydana gelen hatanın onarımı yapılıyor.

- **HOLD ve HLDA** (Hold Acknowledge) pinleri belleğe doğrudan erişim (DMA - Direct Memory Access) arayan ve onaylayan pinlerdir. Çok büyük sayıda yö-

netim fonkisıyonlardan serbestlenmek için mikroişlemci özel bir denetleyiciye, veriyolların tahkimini yönetmeye ve belleğe doğrudan erişim sağlamaya izin veriyor.

- Mikroişlemciyle yönetim sinyalleri. Bu sinyal grubuna RESET sinyali aittir ve onun etkisi altında mikroişlemci verilen anda yapılan işlemi durdurarak tüm yazmaçların yeniden başlatılmasını gerçekleştiriyor, yani onların içeriklerinin siliyor.

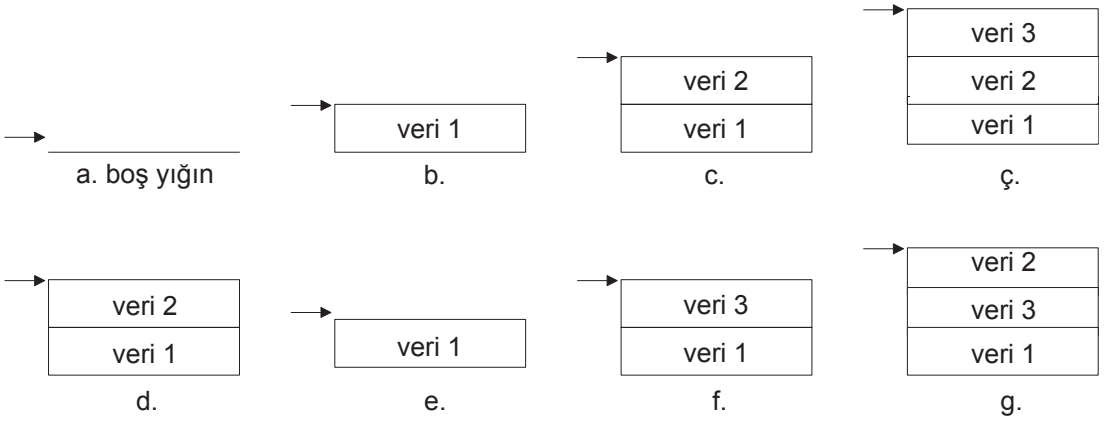
- Senkronizasyon sinyalleri. Sinkronizasyon çalışma hızların uyum sağlanmasını tanımlıyor. Genelde, mikroişlemci en hızlı bileşendir ve bellek gibi daha yavaş bileşenleri beklemek zorundadır. Bu grup sinyallere dijit palsını tanımlayan CLK (Clock) sinyali aittir. Mikroişlemcide gerçekleşen her işlem tam sayı dijit palsı kadar sürdüğünü vurgulayalım.

- Elektrik kaynağı. Bu grup sinyallere tekyönlü gerilim Vcc ve GND (Ground) ile işaretlenen topraklama sinyalleri aittir.

3.6. Yığın Belleğin Kullanımı

Yığın mikroişlemcinin içinde bulunan ve en üst konumdan, tepeden dolan ve boşalan özel düzenlenmiş bellek alanıdır. Yığın belleği LIFO (Last In First Out) kısaltmasıyla da tanınıyor. LİFO (Last In First Out) **son giren veri, ilk çıkıyor** anlamınma geliyor. Yığın belleği fazla tabakalı bellektir, çünkü yeni verilerin yazılması tabakaların yığılmasıyla kıyaslanabilir. Her yeni veri yığının tepesinde yazılıyor. Ancak, aynı zamanda mikroişlemci sadece yığının en üst yerde (tepede) bulunan veriyi okuyabiliyor. Yığının ortasında yerleşmiş bir veri almak istersek, aranan veriye ulaşmak için onun üstünde bulunan tüm bellek konumlarını boşaltmamız gerekiyor ve bu şekilde otomatik olarak gereken veri (bellek konumu) yığının tepesinde bulunacak. Resim 3.13.'te yığın belleğin verilerle dolması ve boşalması gösteriliyor. Başlangıçta yığın boştur (resim3.13.a.). Ondan sonra birinci veriye giriyoruz (resim 3.13.b.). Kenarda bulunan ok mikroişlemciye verildiği anda açık olan veriye ya da yığın belleğinde en üst bellek konumuna gösteriyor. İkinci veri birinci veri üstüne yerleşiyor ve ok bir yer için yukarı çıkıyor (resim 3.13.c.). Ondan sonra üçüncü veriyi de ekliyoruz (resim 3.13.ç.).

Şimdi ikinci verinin ve üçüncü verinin yerlerini değiştirmek istiyoruz. Bunu sadece üçüncü veriyi birinci veri üstüne yerleştirerek yapamıyoruz, önce üçüncü veriyi



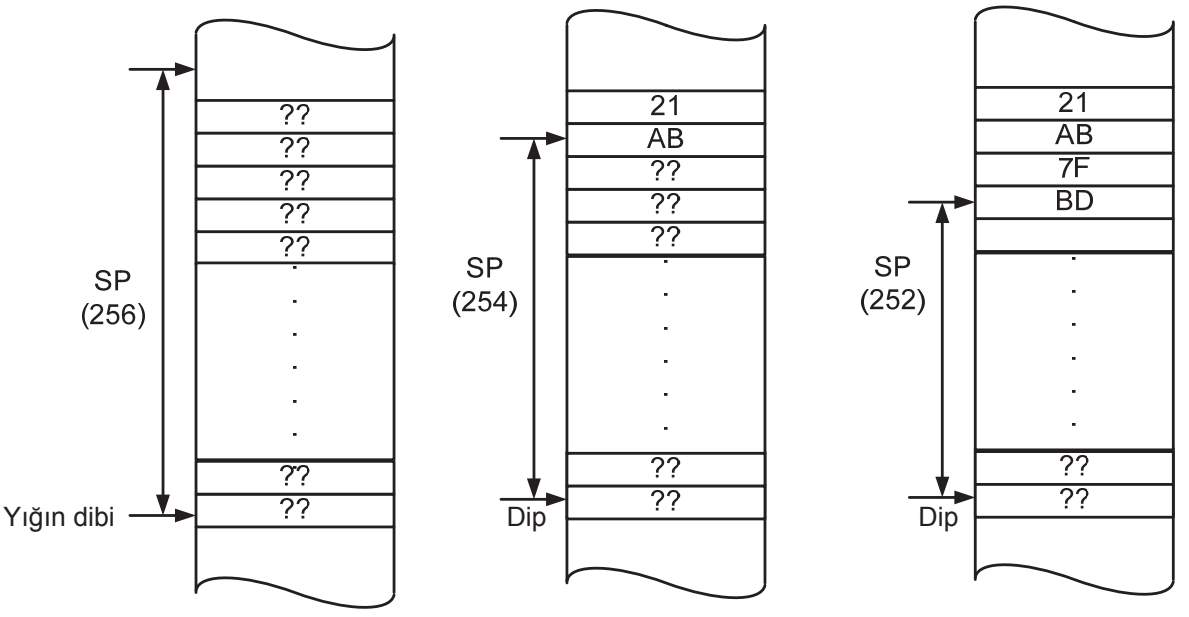
Resim 3.13. Yığın belleğin doldurulması ve boşaltılması

çıkarmamız gerekiyor (resim 3.13.d.), sonra ikinci veriyi çıkarıyoruz (resim 3.13.e.), ondan sonra ise ikinci ve üçüncü veriyi yeniden girmemiz (eklememiz) gerekiyor ancak ters sıralamayla, önce üçüncü veriyi ekliyoruz (resim3.13.f.), ondan sonra ise ikinci veriyi ekliyoruz (resim3.13.g.).

Yığın bellekle çalıştığımız zaman en önemli yazmaç **yığın göstergecidir, SP (Stack Pointer).** Bu yazmaç yığın belleğinde son girilen verinin adresini içeriyor, yani yığından verilen anda erişebilen tek verinin adresini içeriyor. TOS (Top of Stack) yazmacı aslında yığın göstergecinin gösterdiği veriyi içeriyor. Bazı mikroişlemcilerde yığın belleğin en alt yerin, yığın dibinin adresini hafıza eden özel yazmaç bulunuyor. Öyle yazmaç, ilk kez 8086 mikroişlemcide rastlanan yığın bölütü yazmacıdır (SS - Stack Segment),

Her yeni verinin eklenmesiyle yığın göstergecinin değeri artması yerine,onun değeri azalıyor. Aynısı yığının boşalması sırasında da oluyor, sadece ters yönde. Yığının boşalması sırasında yığın göstergecin değeri azalmıyor, hemen artıyor. Buna göre doğru durumu görmemiz için resim 3.13'ü ters çevirmemiz gerekiyor. Yığının daha kolay açıklanması için resim 3.13 verilen şekilde gösterilmiştir. Yığın göstergecinin değeri, yığının dolması ve boşalması sırasında nasıl değiştiği resim 3.14. aracılığıyla açıklanmıştır.

Yığının dibi ve tepesi sabittir. Yığının her dolması ve boşalmasıyla, yığının tepesinden son girilen verinin yerine kadar ölçülen "derinliği" değişiyor. Yığın göstergecin değeri "derinliği" tanımlamıyor, göstergecin değeri yığının dibinden son girilen veriye kadar olan mesafeyi ölçüyor. Yığın göstergecin pozisyonuna uyan ok, yığın doldurulunca açağıya iniyor, yığın boşanınca ise yukarıya çıkıyor.



Resim 3.14. Yığın göstergeç yazmacın değerinin değişmesi (SP - Stack Pointer)

Yığın bellek verilerin korunması için ve alt programların çalışması sırasında kullanılıyor. Genel yazmaçların sayısı her zaman yeterli değildir ve programın çalıştırılması sırasında tüm genel yazmaçların meşgul olma durumu meydana gelebilir. Genel yazmaçların bazısı, yakın zamanda kullanılmayacak, ancak daha geç kullanılabilecek sabit içerirse, o zaman bu yazmacın içeriğini, yığın bellekte taşıyarak korunabilir. Mikroişlemci sabiti arayınca, sabit yığın bellekten bazı genel yazmaca geri götürülebilir.

Yığın belleği ana programdan alt programa geçiş yapılması sırasında program sayacın ve durum yazmacın değerini korumak için kullanılıyor. Alt programın çalışması başlatıldığı zaman bu iki özel amaçlı yazmaç alt program tarafından dikte edilen başka değerlerle dolacak. Sorun alt programdan ana programa döndüğümüz zaman yaşanıyor. Öncede program sayacının değeri korunmamışsa, mikroişlemci ana programda hangi adrese döneceğini bilmeyecek.

Sonuçlar:

Önişaretsiz sayılarla çalıştığımız zaman, 8 - bitle 256 sayı tanımlanabilir, sıfırdan 255'e kadar. Önişaretli sayılarla çalışırsak, o zaman 8 - bitle - 128'den +127'ye kadar sayılar tanımlanıyor.

BCD kodu sıkça kullanılan ikili kodudur. Bu kodta kod sözcükleri ikiyle bölerek kalanı yazmakla elde edilmiyor. BCD kodunda onlu sayıdan her rakam 4 ikili rakamla ifade ediliyor.

Modern işlemciler tam sayıların işletimesi için aritmetik mantık birimi ve gerçek sayılarla ya da kayan noktalı sayılar olarak adlandırılan sayıların işletilmesi için birim içeriyorlar.

Karakterlerin ikili şekilde ifade edilmeleri için ASCII kodu kullanılıyor. 7 - parçalı kod onlu sayı sisteminden rakamları 7 - parçalı ekranda görsel şekilde gösterilmesi için kullanlıyor.

Mikroişlemcide yönergelerin çalıştırılması birkaç aşamaya bölünüyor. Daha önemeli aşamalar şunlardır: işlem kodunun aktarımı, kod çözümlenmesi, operantların bellekten genel yazmaçlara aktarılması, aritmetik mantık biriminde efektif çalıştırılma ve elde edilen sonucun yazdırılması.

Özel amaçlı yazmaçlar kişisel (programcının erişimi var) ya da sistemik olabilir. Özel amaçlı yazmaçların başka ayırımı işlevine göre yapılır. Bu anlamda üç grup vurgulayabiliriz: bellekte verilerin okunması ve yazılması için yazmaçlar (MAR, MDR), yönerge aktarım yazmaçları (PC, İR) ve yığın bellekle çalışma için yazmaçlar (SP, TOS, LV). Her yazmaç için iç veriyoluyla ya da dış adres veriyolu ya da veri veriyoluyla bağlanması için kontrol sinyalleri vardır.

Bayraklar durum göstergesidir, yani elde edilen son sonuçtan bitlerin durumu için bilgi veriyorlar. Sıfır bayrağı Z (İngilizce - zero) sıfıra eşit sonuç elde edilince etkinleştiriliyor. S bayrağı (İngilizce - sign) negatif sonuç elde ettiğimizde aktif oluyor. C bayrağı (İngilizce - carry) ikili sayıda en yüksek pozisyonda taşıma (aktarma) olduğu zaman aktifleşiyor. O bayrağı (İngilizce – overflow) taşma sırasıda ya da öngörülen değerden daha düşük ya da daha yüksek değer elde edildiğinde aktifleşiyor. Bu bayrak sadece önişaretli sayılarla çalıştığımız sırada kullanılıyor.

Mikroişlemci, uygun kontrol sinyaller üreterek aritmetik mantık biriminin çalışmasını yönetiyor. F_0 ve F_1 kontrol sinyalleri işlem seçimi için kullanılıyor. ENA ve ENB sinyalleri veri girişlerin yetenek sağlanması için kullanılıyor. INVA sinyaliyle A girişideki bir eviriliyor. INC sinyali ikili sayıya bir eklemek için kullanılıyor

---

Yönetim birimin en önemli parçası mikroprogramdır. Mikroprogram mikrokodlardan oluşuyor. Her yönerge için birer mikrokod vardır. Mikrokodlar mikro yönergelerden oluşuyor. Bir mikrokotta mikro yönergelerin sayısı, yönergenin çalıştırılmasının ayrıldığı aşamalar, adımlar sayısına eşittir. Her kontrol sinyali için mikro yönergede özel bit bulunyor.

Yönergelerin sürdüğü zaman tam sayı dijit palsla ölçülüyor. Dijit pals üreticisi mikroişlemci yongasında yerleştirilmiş kuvars kristalı tanımlıyor, ancak mikroişlemci dışında da bulunabiliyor.

akış kavramında çalıştırılmaları aynı zamanda birkaç yönergenin çalışması demektir, ancak her yönerge farklı aşamadadır ve her aşama için farklı donanım bileşeni gerekiyor.

CİSC kavramı içinde kompleksli yönergeler de içeren çok büyük yönergeler kümesiyle karakterize oluyor. RISC kavramı yönergelerin sürdüğü zamanın kısalması için, yönergeler kümesinin rasyonelleşmesi demektir. Adres pinleri mikroişlemcinin çıkış pinleridir ve çalışma bellekten bir konumun seçimi için kullanılıyor. Veri pinleri mikroişlemcinin veri okuması ya da yazmasına bağlı olarak giriş - çıkış pinleridir.

IO/M (input output/memory) pini giriş - çıkış aygıtları ya da bellek aygıtı arasında iletişim aygıtın seçimi için kullanılıyor.

Kesinti pinleri INTR (interrupt) ve INTA (interrupt acknowledge) pinleridir. Kesintiler dış aygıtların mikroişlemciyle iletişim için özel düzenlenmiş mekanizmalardır.

Yığın mikroişlemcinin içinde bulunan ve en yüksek konumdan, tepeden doldurulan ve boşaltılan özel düzenlenmiş bellek alanıdır. Yığın belleği çok tabakalı bellektir, çünkü yeni verilerin yazılması tabakaların yığılmasıyla kıyaslanabilir. Yığın belleği verilerin korunması için ve alt programlarla çalışma sırasında kullanılıyor.

Yığın göstergeci, SP (Stack Pointer) yığın belleğinde son girilen verinin adresini içeriyor, yani yığından erişebilen tek verinin adresini içeriyor.

Sorular ve Ödevler:

1. “İntel” şirketinin ürettiği mikroişlemcileri onların kronoloji gelişimine göre say.
2. Processor sözcüğünü ASCII kodla ifade et!
3. 16 - bitli veriler kullanılırsa, ikili sayı sisteminde ifade edilebilen önişaretli ve önişaretsiz onlu sayıların kapsamını hesapla!
4. Yedi parçalı ekranda dört rakam gösterilirse yanan bölümleri say!
5. Yönerge yazmacında hangi kod saklanıyor?
6. Mikroişlemcinin içinde yazmaçlar nasıl ayrılıyor?
7. Bellek adres yazmacın ve bellek veri yazmacın işlevlerini açıkla!
8. 1 - bitli aritmetik mantık biriminin kontrol sinyallerini say ve açıkla!
9. Durum yazmacında bayrakların işlevlerini say ve açıkla!
10. Mikroişlemcinin mikroprogramı nerede bekleniyor ve ne için kullanılıyor?
11. Yönerge ile mikro yönerge arasında fark nedir?
12. Mikroişlemcide çalıştırılan bir yönergenin, çalıştırılmanın ayrıldığı aşamaları say!
13. 8086 mikroişlemcisi 16 - bitli mikroişlemcidir. Bu ifadenin ne demek olduğunu açıkla!
14. RISC kavramının CISC kavramında göre hangi avantajları ve dezavantajları var?
15. Mikroişlemcide yönergelerin akışlı çalıştırılmasını açıkla?
16. Mikroişlemcide adres ve veri pinlerin işlevlerini açıkla!
17. Mikroişlemci bazı dış aygıta veri yazdırırısa, $\overline{WR}$, $\overline{RD}$, $IO/\overline{M}$ kontrol pinleri nasıl mantıksal durumda olacaklar?

18. HOLD ve HLDA pinlerin ne için kullanıldıklarını açıkla!

19. 8085 mikroişlemcide genel yazmaçta yazdırılan çalıştırıldığı yönergenin sonucunun kaybolması neden meydana geliyor?

20. Program sayacının nasıl işlevi var?

21. RAM belleğinde sıradaki konumun adresi, program sayacının içeriğine bir ekleyerek elde edilmediği iki yönerge say.

22. Mikroişlemcinin çalışmasında CLK sinyalin nasıl fonksiyonu olduğunu açıkla.

23. Yığın belleği nasıl düzenlenmiştir?

24. Yığın - göstergeç yazmacı ne için kullanılıyor?

25. Yığın belleğin nasıl kullanımı olduğunu açıkla.

4. Mikroişlemci Sistemleri

4.1. Mikroişlemcinin Bağlanması Sırasında Temel Terimler

Arabirim terimi altında bir sistemin içeriğinde aygıtlar arasında tam uyumluluğun sağlanması amacıyla bağlanma şekli, uyarlama tanımlanıyor. Bir aygıtın performansları oldukça tatmin edici olabilir, ancak giriş - çıkış özellikleri, sistemde önceden yerleştirilmiş aygıtların özelliklerine uyumlu olmadığı için bu aygıt sistemde yerleştirilemez. Uyum sağlamak, bağlanmak yani arabirim sadece elektronik,donanım bileşenlere değil, çoğu zaman aygıtlar arasındaki farkları aşmak için özel yazılım da gerekli olabilir.

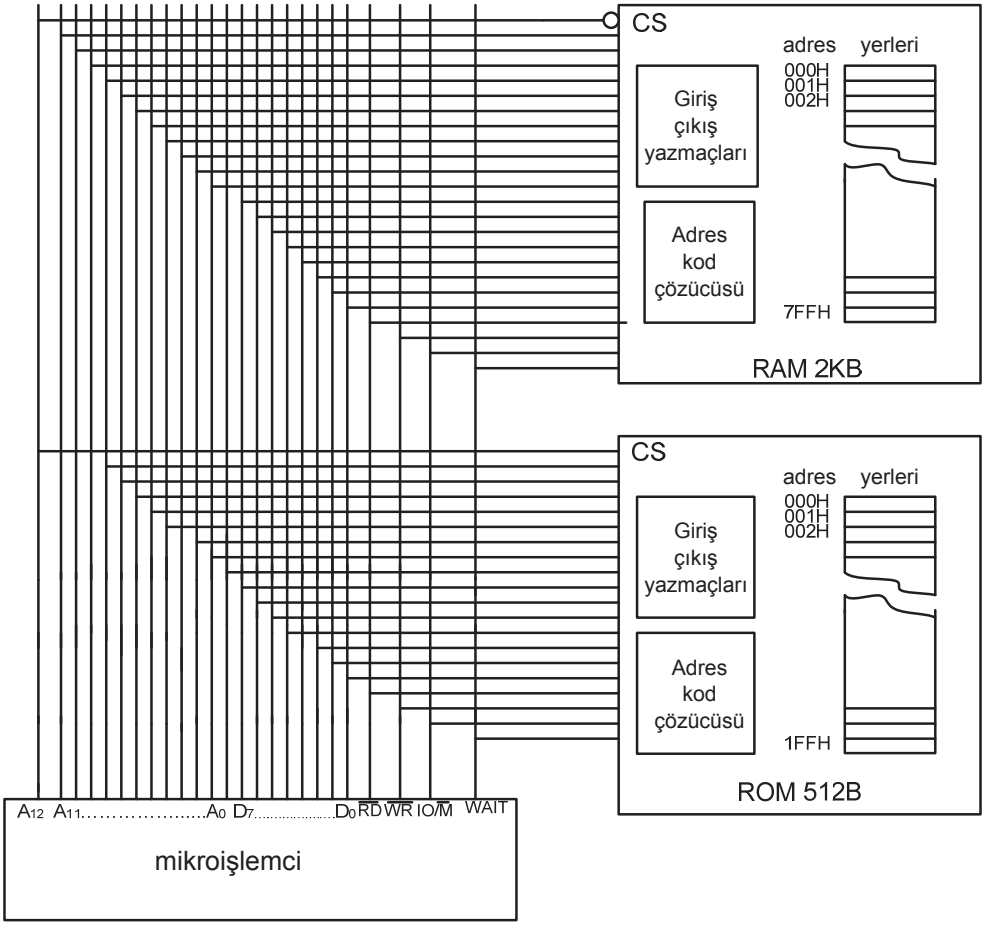
Senkronizasyon (eşzamanlama) başarılı arabirimin gerçekleşmesi için çok önemlidir. Senkronizasyon iki aygıtın hızı arasında uyum sağlamak demektir. Bu amaçla farklı sayaç türleri kullanılıyor. Örneğin, veriler gönderen aygıt, verileri alan aygıttan dört misli daha hızlıdır. Birinci aygıta verileri göndermek için dört dijit atışı gerekiyor, ikinci aygıta ise verileri almak için 4 atış gerekiyor. Dijit palsı göndericiden sayacın girişine getiriliyor. Sayaç çıkışta girişin 4 atışına bir atış veriyor. Şöyle ki, sayaç her giriş palsıyla içeriği bir için artan yazmaçtır. Bu durumda sayaçın $100_2=4_{10}$ içeriği olunca, sayaç kendi çıkışını etkinleştiriyor ve sinyal veriyi alan alıcıya gönderiliyor. Mikroişlemci, bellekten daha hızlı aygıttır. Senkronizasyon olmazsa, bellek mikroişlemci tarafından gönderilen tüm verileri alamayacak ve veri kaybı meydana gelecek. Onun için, mikroişlemcinin zamanlama sisteminde birkaç bekleme dijit palsı eklenmelidir ve öylece belleğe tüm gönderilen verileri saklaması için yeterince zaman veriliyor.

Sinyalleşme (sinyalizasyon) kullanıcı bilginin aktarılması başlamadan önce, güvenilir aktarımın sağlanması için, kontrol sinyallerin değişimi olarak tanımlanıyor. Örneğin, hızlı dış aygıtı mikroişlemciye veriler gönderiyor, ancak mikroişlemci o anda daha yüksek öncüllüğü olan program çalıştırıyor ve kullanıcı bilginin kaybı meydana geliyor. Bundan dolayı, dış aygıtı verileri göndermek için izin almak için beklemelidir, ya da böyle bir imkân yoksa, gönderilen verilerin geçici olarak bazı bellek modülünde yerleştirilmelidir ve mikroişlemcinin serbest kalması bekleniyor.

Mikroişlemci ve mikroişlemciye bağlanan tüm tümleşik devreler **TTL uyumluluk** koşullarını yerine getirmelidir, yani giriş ve çıkış mantık seviyeleri uyumlu olmalıdır. Tetikleme sinyali çok zayıfsa, giriş pinin aktiflenmesi gerçekleşmeyecek, çok kuvvetliyse tümleşik devrenin hasar görmesine yol açabilir.

4.2. RAM ve ROM Bellekleriyle Bağlanma

Resim 4.1.'de RAM ve ROM belleklerin mikroişlemciyle bağlanmaları gösterilmiştir.



Resim 4.1. Mikroişlemcinin RAM ve ROM bellekleri ile bağlanması

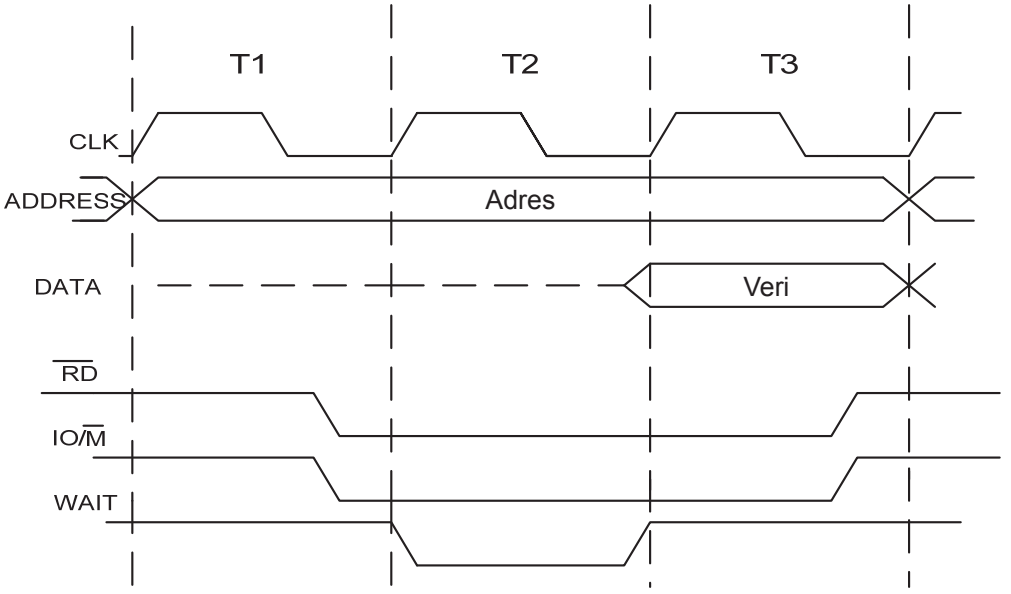
Bellek yongaların pinleri (iğneleri) anakartın hatları aracılı yoluyla mikroişlemcinin pinleriyle bağlıdır. Anakarttaki hatlara veriyolu denildiğini ve veriyolların üç gruba: adres, veri ve kontrol veriyoluna ayrıldığını bir kez daha hatırlayalım.

Adres hatları bellekler için her zaman giriş, mikroişlemci için ise çıkış hatlarıdır. Adres RAM ya da ROM belleğinden bir yerin seçilmesi için benzersiz ikili kombinasyonudur. Örneğimizde RAM belleğin kapasitesi 2KB'tır ve bunun için 11 adres hatı (2^{11}=2KB) gerekiyor, A_0'dan A_{10}'a kadar. Bu adres bitleri 000H'dan 7FFH'ya kadar adresleri oluşturuyor. ROM belleğin kapasitesi 512 B'tır ve onların adreslenmesi için 9 adres hatı gerekiyor, A_0'dan A_8'e kadar. ROM belleğin adres kapsamı 000H'dan (000000000B), 1FFH'ye (111111111B) kadar yayılıyor. Mikroişlemcinin RAM ya da ROM bellek arasından hangisinden okuyacağı en önemli adres hatı A_{12}'ye bağlıdır. Bu hat, iki bellekten birine erişim sağlayan cs pinlerini aktif ediyor. ROM ve RAM belleklerin yongalar kümesinde bir yonganın seçimiyle, adres kod çözümleme sürecini incelediğimiz bölümde tanışacağız. Veri hatları iki yönlüdür, bellekten veri okuduğumuza ya da yazdığımıza bağlı olarak, girişli ve çıkışlı olabilir. IO/$\overline{M}$, $\overline{RD}$ ve $\overline{WR}$ kontrol sinyalleri mikroişlemci için çıkış sinyalleridir, $\overline{WAIT}$ ise bellek için çıkış sinyalidir.

Resim 4.2.'de bellekten veriler okuma işlemi ve işlem sırasında bir bekleme atışı olduğu zaman diyagramı verilmiştir. Mikroişlemci senkron çalışma modunda çalışıyor. Veri aktarımında senkronizasyon çok önemli süreçtir. Senkronizasyon mikroişlemcinin dijit palsıya gerçekleşiyor – clock sinyali. Dijit palsın periyodu, mikroişlemcinin tanıdığı en küçük zaman kapsamıdır. Bellekten bir baytın okunması için üç dijit palsı gerekiyor. Sinyallerin kenarları yükselen ve alçalan kenarları dikey olmadıklarını görüyoruz, çünkü hiçbir elektrik sinyali, durumu bir anda mantıksal sıfırdan bire değiştiremez.

Birimci dijit palsı T1 başlangıcında, mikroişlemci adres hatları aracılığıyla gereken adresi belleğe gönderiyor. Adres birkaç bitten oluşan kombinasyondur, ancak onların durumu gösterilmiyor, sadece adres değişince kesişen iki hat kullanılıyor. Aynı şekilde veri hatların değişiklikleri de tanımlanıyor, tüm diğer sinyaller için ise birer hat kullanılıyor. Adresten sonra, mikroişlemci iki kontrol sinyali gönderiyor – IO/$\overline{M}$ ve $\overline{RD}$. Bellek bu sinyalleri daha birinci pals zamanında alıyor. Ancak, bellek verilen işlemi hemen gerçekleştirecek durumda değildir Belleğe veriyi aramak, bulmak ve aranan veriyi göndermek için daha fazla zaman gerekiyor. Demek ki, bu işlem bir pals süresi içinde gerçekleşmiyor ve bundan dolayı bekleme durumunu belirleyen $\overline{WAIT}$ sinyali aktifleşiyor.

Bellekten mikroişlemciye veriler, veri hatları aracılığıyla üçüncü palsta gönderilecek, kontrol sinyalleri etkinliği kaldırılacak. adres hatları ise serbest kalacak.



Resim 4.2. Bellekten okuma işlemin zaman diyagramı

Senkron aktarımın basitliğinden ve çalışma sırasındaki hassasiyeten dolayı büyük avantajı vardır. Ancak herşeyden önce çalışma hızı gibi büyük dezavantajları da var. Örneğin, her makine döngünün zaman süresi dijit palsın katsayısı (tamsayı) ile ifade ediliyor. Verilen işlemin tam olarak 3.1 pals süresinde gerçekleşebilmesine rağmen, bu işlem için 4 pals kullanılacaktır, çünkü pals ayırımı kesinlikle yasaktır. Belirli teknoloji ilerlemeleri de başlatılan çalışma zamanına uyum sağlayamazsa onların kullanıllması daha büyük dezavantaj ortaya koyuyor. Birkaç yıl sonra zaman erişimi 40ns yerinde 20ns olan bellek yongaları üretilebilir. Böyle ilerleme otomatik olarak bekleme palsın elenmesine neden olmalıdır, ancak bekleme palsın elenmesi meydana gelmeyecek, çünkü mikroişlemci ve diğer aygıtlar bellek konumundan verinin okunması üç palsın süreceğini bekliyorlar ve bu durum kolayca değişemez.

4.3.Sanal (Virtüel) Bellek

Yazılımın gelişmesiyle programlar gittikçe daha büyük ve geniş olmaya başlamış ve ana RAM belleği sistemin efektif şekilde çalışmasını sağlamak için küçük kalıyormuş. Sanal bellek kavramının gündeme gelmesiyle, ana (ilkel) belleğinin dış (ikincil) belleğini kullanarak genişlenmesi yapılabiliyormuş.

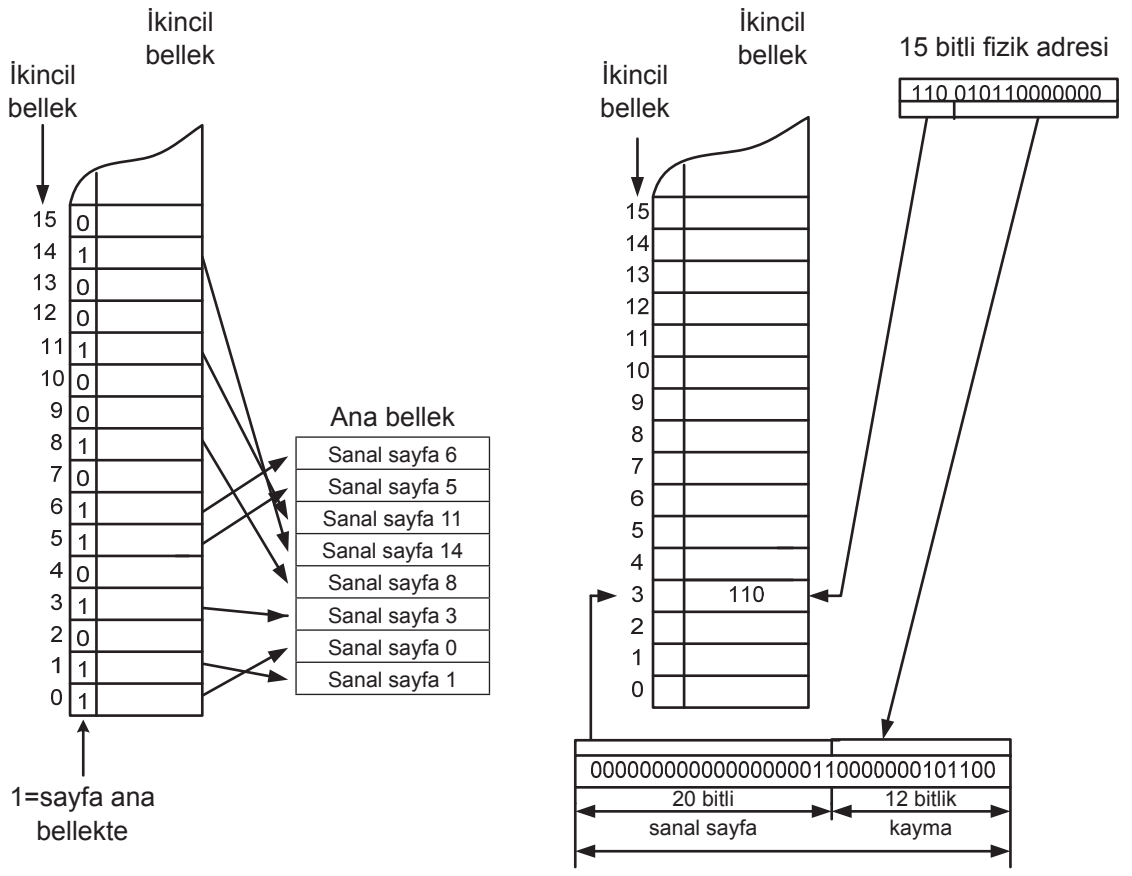
Mikroişlemcide çalışıtırılması gereken program bloklara ayrılıyor ve her blok birbirinden bağımsız olarak çalıştırılıyor. Programın birinci bloğu çalıştırıldıktan sonra, birinci blok ana bellekten ikincil belleğe geri dönüyor, ikincil bellekten ana belleğe ise sıradaki blok taşınıyor, ondan sonra sıraya üçüncüsü geliyor ve tüm programın çalıştırılmasına kadar bu şekilde devam ediyor. Başlangıçta programın bloklara ayrılması programcının göreviymiş, ancak bu işlem programcıların işini büyük ölçüde ağırlaştırıyormuş. Bugün programın bloklara ayrılmasını ve blokların bir bellek alanından başka bellek alanına taşınmasını, otomatik olarak Memory Menagment Unit (MMU) özel kontrol yongası tarafından gerçekleşiyor. Sanal belleğin ortaya çıkmasına kadar, adres alanı bellek alanından daha büyükmüş ya da ona eşitmiş. Sanal belleğin ortaya çıkmasıyla, durum değişmiş, daha doğrusu adres alanı bellek alanından daha küçük olmuş. Adres bölümünün adres alanında yer alan bitlerin sayısına ya da bu bitlerin geçtikleri adres veriyolunun genişliğine bağlı olduğunu hatırlayalım. Bellek alanı doğrudan yerleşmiş olan bellek yongaların sayısına ve yongaların kapasitelerine bağlıdır.

Resim 4.3'te ana ve ikincil bellek ile sanal bellek alanından fizik bellek alanına geçiş gösterilmiştir. Resim 4.3.a'dan ana belleğin 8 çerçeveye ayrılmış olduğunu görebiliriz. Bu arada her çerçeve 4KB=2^{12} sığdırabilir, ana belleğin tümü ise 2^{15}=32KB=84KB'tır (4K - çerçeve büyüklüğü, 8 - çerçeveler sayısı). İkincil ya da sanal bellek 4KB büyüklükteki sayfalara ayrılmıştır, tüm sanal belleğin büyüklüğü ise 4GB=2^{32}'dir.

Adres bitlerin anlamı resim 4.3.b.'de gösterilmiştir. Ana belleğin bellek yerlerin adreslerine fizik adresi deniyor ve onlar 15 bitten oluşuyor. Sabit - diskteki konumların adreslerine sanal adresleri deniyor ve onlar 32 bit içeriyor. Mikroişlemcinin gereklerine göre, sanal bellekten istenilen sayfalar alınıyor ve ana belleğin çerçevelerinde yerleştiriliyor. Çerçevelerin ve sayfaların aynı büyüklükte olduğunu vurguluyoruz, ancak çerçeveler sabittir, sadece çerçevelerdeki sayfalar değişiyor. Sanal adres programcı tarafından verilen adrestir ve bu adres MMU yonganın giriş yazmacında saklanıyor ve sanal adresinden mikroişlemci RAM'dan istenilen veriyi buluyor. Sanal adresin ilk 20 biti sanal belleğinde sayfanın numarasını veriyor. 20 bitle toplam 2^{20} farklı kombinasyon yapılabilir, onlar da aslında ikincil belleğin içerebildiği sayfalar sayısıdır.

Sanal adresin son 12 biti seçilen sayfa çerçevesinde bellek konumun tanımlanması için kullanılıyor. Fizik adresinde son 12 bitin anlamı sanala adresindeki son 12 bitin anlamıyla aynıdır, ilk 3 bit ise seçilen veri ana belleğin hangi çerçevesinde yerlemiş olduğunu gösteriyor.

3 bitle 2^3=8 kombinasyonlar yapılabilir ve bu sayı ana bellekte çerçeveler sayısına uyuyor. MMU yongasında iki bilgi içeren özel sayfalar tablosu bulunuyor: mevcut sayfa ana belleğin bazı çevreçevinde bulunuyor mu (0 - hayır, 1 - evet) ve çerçevenin sıra numarası.



a. Ana bellekten ikinci belleğe aktarma

b. Fizik adresten sanal adresin elde edilmesi

Resim 4.3.

Ana belleğine yeni sayfaların eklenmesi için farklı logaritmalar vardır. Ana bellekte tüm çerçeveler sayfalarla doluysa, yeni sayfanın eklenmesi için hangi sayfanın çıkarılması gerektiği soru ortaya çıkıyor. En iyi yakın gelecekte mikroişlemci hangi sayfalardan gerek duyacağını bilmek olur ve en erken aranacak sayfalar, çerçeveye eklensin, daha geç aranacak sayfalar ise çerçeveden çıkarılsın ve onların yerine yeni sayfalar için yer açılsın. Ancak böyle ön görmeler çok zor yapılabilir. Sıkça kullanılan algoritma LRU (Least Recently Used) algoritmasıdır. LRU algoritmasına göre ana bellekten yakında kullanılmış sayfa çıkarılıyor. Bu yöntemde belli kuralsızlıklar ortaya çıkıyor ve böyle bir durum tablo 4.1'de verilmiştir.

Sanal sayfa 7	Sanal sayfa 7	Sanal sayfa 7
Sanal sayfa 6	Sanal sayfa 6	Sanal sayfa 6
Sanal sayfa 5	Sanal sayfa 5	Sanal sayfa 5
Sanal sayfa 4	Sanal sayfa 4	Sanal sayfa 4
Sanal sayfa 3	Sanal sayfa 3	Sanal sayfa 3
Sanal sayfa 2	Sanal sayfa 2	Sanal sayfa 2
Sanal sayfa 1	Sanal sayfa 1	Sanal sayfa 0
Sanal sayfa 0	Sanal sayfa 8	Sanal sayfa 8
a	b	c

Tablo 4.1. sayfalar değişimi için LRU algoritması

Şöyle ki, mikroişlemcinin sanal bellekten ilk sekiz sayfayı girmekten ihtiyacı var (0 - 7). Ondan sonra 8 sıra numarası olan dokuzuncu sayfayı alması gerekiyor. Ancak, onun için yer yok. Yakında kullanılan sayfa 0 sıra numaralı sayfadır ve o sayfa çıkarılıyor ve onun yerine 8 numaralı sayfa yerleşiyor. Sıraya gelen adımda mikroişlemcinin yeniden 0 numaralı sayfayı yeniden araması manasızdır ve 0 numaralı sayfayı yerleştirmek için 1 numaralı sayfayı çıkarmamız gerekiyor. Demek ki, önceki adımda 0 numaralı sayfanın çekilmesi bir taraftan hatalı hamledir, ancak LRU algoritması bunu gerektiriyor. Diğer kullanılan algoritma FIFO (First In First Out) algoritamsıdır ya da “ilk giren ilk çıkar”. FİFO algoritmasına göre sayfaların değişimi için kriter son kullanıştan en çok zaman geçen değil, ana belleğe girişten en uzun zaman kalan safya değiştiriliyor.

Programcı için çok önemli etken ana bellekte sayfaların değişme hızıdır. Ayrıca, sayfaların büyüklüğü, yani çerçevenin büyüklüğü de önemli parametredir. Sayfalar çok küçük, aynı zamanda çok büyük olmamalıdır. Çok büyükse, o zaman RAM belleğinde az sayıda sayfa yerleşebilecek ve mikroişlemcinin gereklerini yerine getirmek için sayfaların daha sıkça değiştirilmeleri gerekecek. Diğer taraftan, sayfalar çok küçükse, o zaman tüm RAM belleğin doldurulması için gerçekleşmesi gereken aktarmaların (geçişlerin) sayısı artıyor, ancak RAM belleğinde daha büyük sayıda çerçeveler varsa, o zaman yeni sayfanın girmesi halinde, çıkarılacak eski sayfalar arasında daha büyük seçim olacak. Herşeye rağmen, sayfanın büyüklüğü ve onların RAM belleğindeki sayısı ters olantılı olduğundan dolayı uzlaşmalı bir çözümün bulunması gerekiyor.

4.4. Önbelleğin Organizasyonu

Önbellek mikroişlemci ve ana bellek arasında aracılık yapıyor. Amacına göre önbellek yönerge ve veri önbelleği olabilir. Yönerge önbelleğinde ana bellekten önceden getirilmiş yönergeler saklanıyor ve mikroişlemci tarafından çalıştırılmaları için çağırlmalarını bekliyorlar. Veri önbelleği mikroişlemcinin daha sıkı çağırması, ancak mikroişlemcinin genel yazmaçlarında bu veriler için yer olmayan verilerin saklı olduğundan geliyor.

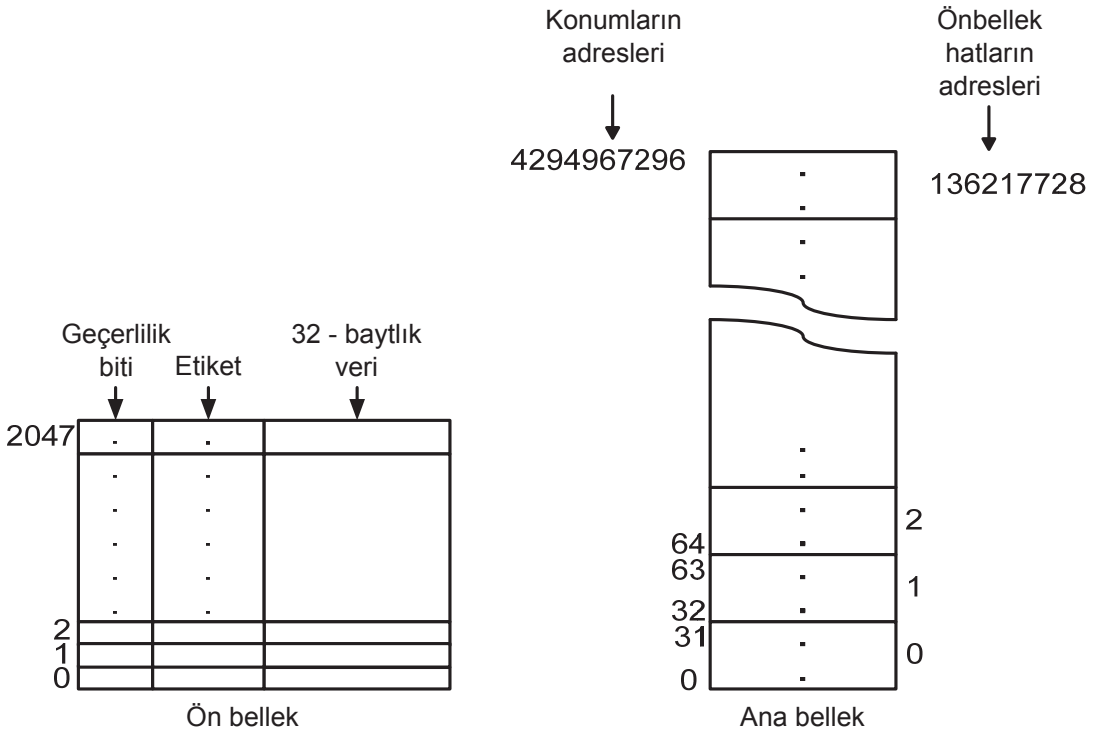
Daha gelişmiş bellek organizasyonlarında birkaç önbellek seviyesi vardır. Üç önbellek seviyesi mevcuttur. Birinci seviyeli önbellek mikroişlemcinin yongasında bulunuyor, ikinci seviye önbelleği mikroişlemcinin paketlemesinde bulunuyor ve üçünci seviye önbelleği anakartında yer alıyor.

Tüm önbellek çeşitlerin uygulamaları şu şekilde gerçekleşiyor. Ana bellek, sabit büyüklükte olan ve önbellek hatları denilen bloklara ayrılıyor. Önbellek hatları 4 ile 64 ardaşıl baytlardan oluşuyorlar. Hatlar 32 bayttan oluşmuşsa, o zaman sıfır hattı 0'dan 31'e kadar adresli baytları içeriyor, ikinci hat 32'den 63'e kadar adresli olan baytları, ikinci hat 64'ten 95'e kadar vs. Bu hatlardan bazıları önbellekte bulunuyorlar. Mikroişlemci belleğe erişmeden önce, önbeleğin denetimcisi aranan verinin önbelekte olup olmadığını konrol ediyor. Aranan veri önbellekte bulunuyorsa, o zaman mikroişlemci ana belleğe ulaşım yapmayacak. Bu durumda adres ve veri veriyolları serbest kalacak ve veriyollar başka bazı işlem için kullanılabilecek. Örneğin mikroişlemcinin aracılığı olmadan ana belleğe doğrudan veri aktarımı. Ancak, aranan veri önbellekte bulunmuyorsa, o zaman önceden girilmiş önbellek hatlarından biri çıkarılmalıdır ve onun yerine aranan hat yazılmalıdır.

Resim 4.4.'te 32 baytlık blokara ayrılmış olan 4 GB kapasiteli ana bellek gösterilmiştir. Aynı resimde 2048 girişli önbellek de gösterilmiştir.Önbelleğin her girişi 32'şer baytlık önbellek hatı sığdırabilir. Buna göre önbelleğin kapasitesi 2K32B=64KB olacak.

Önbelleğin her girişi üç parçadan oluşuyor:

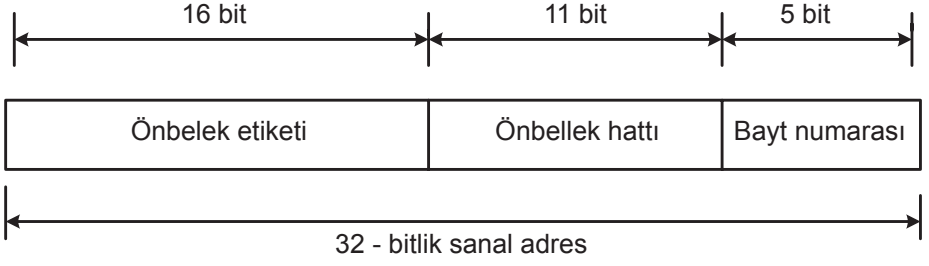
1. Geçerlilik biti sistemin başlatılmasından sonra önbelleğin girişinin dolu olup olmadığını gösteriyor. Sistem başlatıldığı zaman önbelleğin tüm girileri geçersiz olarak işaretlidir.
2. Etiket (tag) alanı denilen (İngilizce tag=işaretleme) verinin alındığı ana bellekten önbellek hatın sıra numarasını içeriyor.
3. Veri alanı önbellekten önbellek girişine aktarılan veriyi içeriyor.



Resim 4.4. Ana belleğin (bloklara ayrılmış) ve 2048 girişli önbelleğin görünümü

Resim 4.5.'te önbellek denetimcisinin arana veriyi bulmasına yardım eden sanal adres gösterilmiştir. Sanal adresin, önbellek girişlerin olduğu gibi üç bölüme ayrıldığını görüyoruz

1. Etiket (tag) alanı önbellek girişinde aynı isimli alana uyuyur
2. Hat alanı aranan verinin bulunduğğu önbellek girişinin sıra numarasını veriyor
3. Bayt alanı seçilen önbellek girişindeki baytın sıra numarasını veriyor



Resim 4.5. Önbellekte veri bulmak için sanal adres

Sanal adresten her alanın büyüklüğü şu şekilde belirleniyor. Ana bellek 4GB kapasiteli olduğu için sanal adres 32 bitli olacak. Bayt alanında bitlerin sayısı, önbellek girişlerin büyüklüğüne eşit olan önbellek hatlarının büyüklüğüne bağlıdır. Örneğin, önbellek hatlarının büyüklüğü 32 baytsa, o zaman bayt alanı 5 bitlik büyüklüğünde olacak (2^5=32). Hat için alanın büyüklüğü önbelleğin kaç girişi olduğuna bağlıdır. Bizim örneğimizde bu sayı 2048=2^{11} değerindedir ve buna göre hatlar için alan 11 bit içerecek. 32 bitlik sanal adresin kalan 16 bit etiketin alanını temsil ediyor.

Mikroişlemci adres ürettiği zaman, hat alanı ayrılıyor ve bu alan 2048 önbellek girişinden birini bulmak için kullanılıyor. Geçerlilik biti bire eşit olursa, o zaman sanal adresten etiket alanı ve önbellek girişinin etiket alanı kıyaslanıyor. Bu iki etiket alanı aynıysa, o zaman önbellek aranan veriyi içeriyor diyoruz ve bu duruma önbellek isabeti deniliyor. Aranan veri önbellekten okunacak ve ana bellekten aktarılmadan kaçınılacak. Bayt alanı önbellek içeriğinin tamamını aktarmadan sadece bir baytın ayrılmasını sağlıyor. Geçerlilik biti sıfıra eşitse ya da etiket alanları birbirine uymuyorsa, o zaman aranan veri önbellekte bulunmadığını diyoruz. Böyle durum önbellek isabetsizliği deniyor. Böyle durumda ana belleğe erişmek gerekiyor, oradan aranan önbellek hattı alınıyor ve önbelleğe yerleştiriliyor.

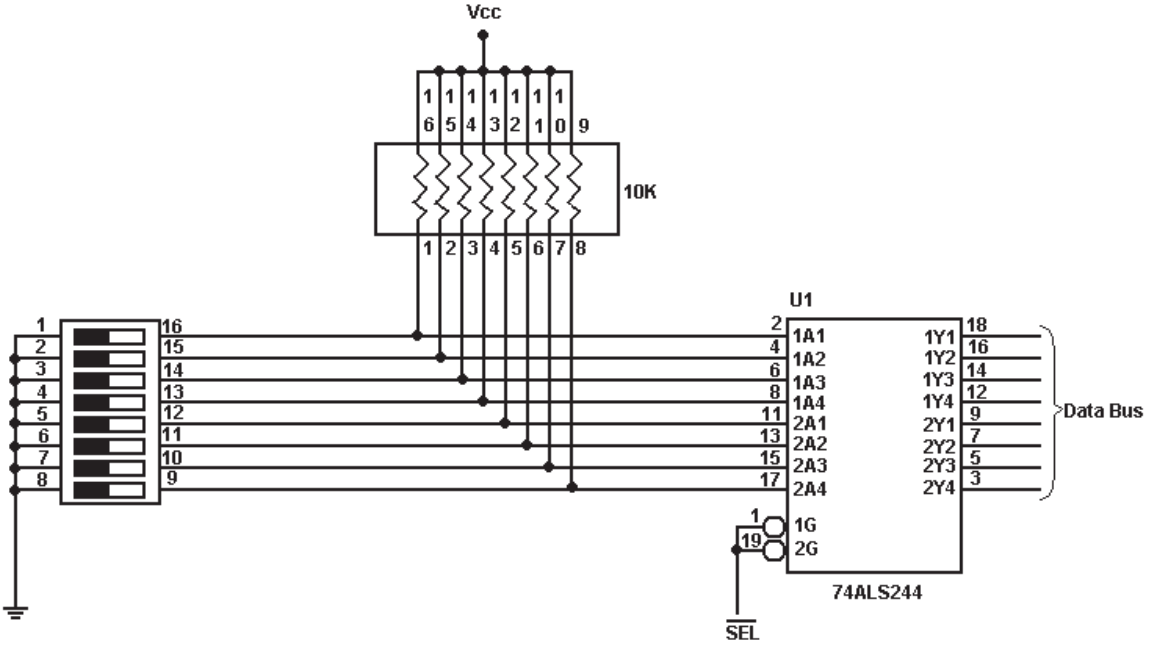
Önbellekte bu şekilde arama süreci uzun olmasına rağmen, çok kısa erişim zamanı var ve nanosaniye büyüklüğündedir. En büyük sorun önbellek isabetsizlikleri tanımlıyor, çünkü o zaman iki bellekte arama yapması gerekiyor, hem önbellekte hem de ana bellekte.

4.5. Giriş/Çıkış Aygıtarla Bağlanmak

Giriş çıkış aygıtları bilgisayarın dış dünyayla iletişim kurmak için kullanılıyor. Bu aygıtlar giriş aygıtları olabilir (klavye, fare), çıkış aygıtları olabilir (monitör, yazıcı) ya da girişli çıkışlı olabilir (CD diski). Giriş çıkış aygıtları dış aygıtlar adıyla da biliniyor. Dış dünyayla iletişim kurmak dışında, giriş çıkış aygıtlardan bazıları veriler saklamak için de kullanılıyor. Kötü taraf, giriş çıkış aygıtların veri aktarımı sırasında bellek aygıtları gibi adres ve veri veriyollarını kullanmalarıdır. Giriş çıkış aygıtları ve mikroişlemci arasında veri aktarımını daha iyi anlamak için çok basit iki örnek göreceğiz.

Resim 4.6.'da sekiz paralel anahtarla bağlanmak için örnek verilmiştir. Anahtarlar veri veriyoluyla 74ALS244 ara bellek aracılığıyla bağlıdırlar. Ara belleğin gi-

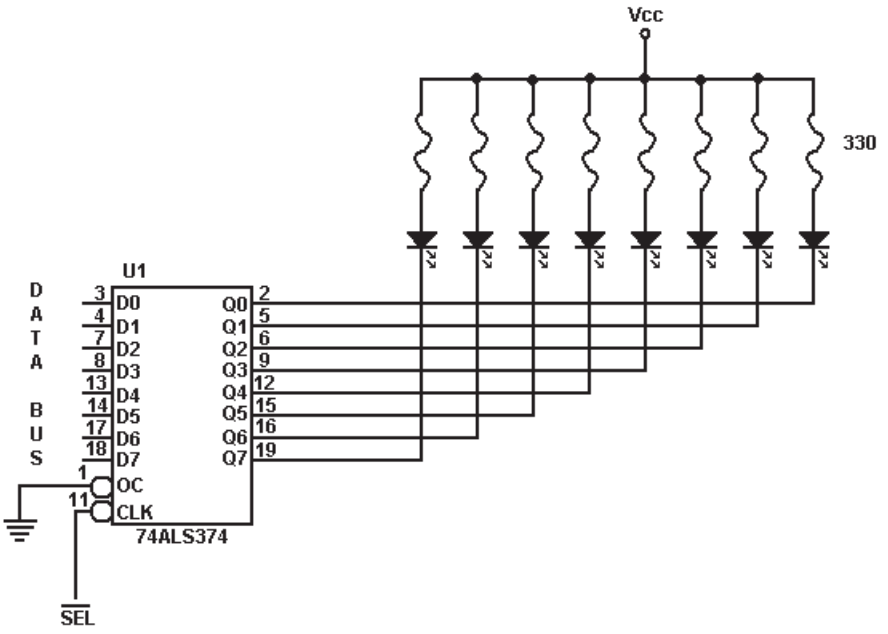
rişinde anahtarlar eklenmiştir, çıkışta ise veri veriyolunun sekiz hatı bulunuyor. Anahtarlardan bazısı kapalı olduğu zaman veri hatından mantıksal birim gönderiliyor, çünkü anahtar ara belleğin giriş pinini tabloya bağlıyor. Anahtar açık olunca mantıksal biririm gönderiliyor. Ders kitabın birinci konusunda, temel mantıksal devreleri incelerken, ara belleğin sinyallerin işletilmesini yapmadığını, ara bellek dış aygıtlarını veri veriyoluyla donanımsal bağlama yaptığını vurgulamıştık.



Resim 4.6. Mikroişlemcinin 8 paralel anahtarla bağlanması

Ara bellek anahtarların gönderdiği bitleri geçirip geçirmediği $\overline{SEL}$ sinyalın durumuna bağlıdır. Bu sinyal yüksek seviyedeyse, ara bellek yüksek empendans durumunda olacak. Sinyal düşük seviyedeyse, o zaman ara belleğin iki seçim pini 1G ve 2G (G - gate) aktif oluyor. $\overline{SEL}$ sinyalın değeri, daha geç inceleyeceğimiz dış aygıtın arabirim adresinin kod çözülmesiyle elde ediliyor.

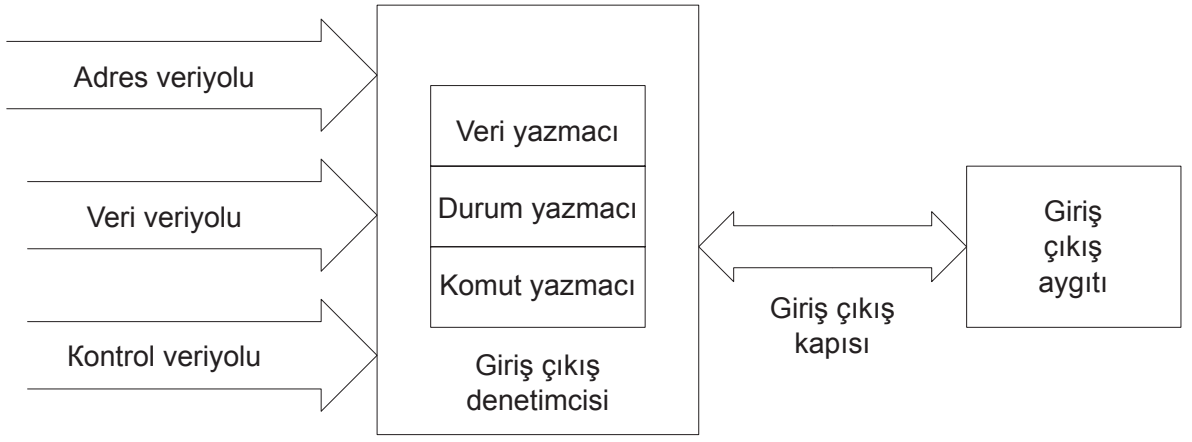
Resim 4.7.'de çıkış aygıtıyla bağlanma için örnek verilmiştir. Sekiz paralel yerleştirilmiş LED diyotlar, mikroişlemci mantıksal sıfır gönderdiğinde yanıyorlar. Giriş arabiriminde ara bellek olduğu gibi, veri veriyolu ve LED diyotlar arasında mandal (latch) olarak adlandırılan özel bir aygıt yerleştiriliyor. Mandal, görevleri tetikleme sinyalın süresini devem ettirmek olan palslı flip floplardan oluşan aygıttır. Mikroişlemcinin gönderdiği sinyaller sadece 1 µs sürüyorlar. Bu süre led diyotların tetiklenmeleri için yeterli değil. Bu yüzden palslı flip flopun girişinde sinyalın gelmesinden hemen sonra dijit palsı alçak seviyeye düşüyor, girişler devre dışı bırakılıyor, ancak gönderilen sinyal flip flopun çıkışında kalıyor.



Resim 4.7.Mikroişlemcinin 8 paralel led diyoduyla bağlanması.

Gördüğümüz iki basit örnekten, giriş çıkış aygıtların sistem veriyoluyla doğrudan bağlı olmadıkları görünebiliyor. Dış aygıtlar ve sistem veriyolunun iletişimi arasında aracılık yapan devrelere giriş/çıkış denetimcileri ya da arabirim denetimcileri denir. Bu denetimcilerin kullanımı için birkaç faydalı nedeni vardır. Herşeyden önce, denetimciler mikroişlemcinin yönetim işlevini kolaylaştırıyor. Mikroişlemci daha serbest kalıyor ve kullanıcı programlarını daha çabuk çalıştırıyor. Arabirim denetimcileri dış aygıtlarını yönetmek için basit kontrol sinyalleri üretiyorlar. Sinyalleri sistem veriyolundan göndermek için çok az enerji harcanıyor. Arabirim denetimcileri kullanılmazsa ve bu denetimciler gönderilen sinyallerle güçlendirilmezse, o zaman dış aygıtları bilgisayarla bağlamak için kullanılan kablolar sadece birkaç santimetre uzunluğunda olurdu.

Giriş/çıkış denetimcileri üç çeşit iç yazmaç içeriyor: veri, komut ve durum yazmaçları. Giriş çıkış denetimcinin yazmaçları resim 4.8.'de gösterilmiştir. Veri yazmaçları dış aygıtlardan bazısından okunan verileri ya da dış aygıtların bazısına yazılması gereken verileri saklıyorlar. Durum yazmaçları dış aygıtların mevcut durumları hakkında bilgi içeriyorlar. Durum yazmaçların önemini, bilgisayarı yazıcıyla bağlanma örneğiyle açıklayacağız. Mikroişlemci yazıcıya karakter göndermeden önce, yazıcıyı yöneten denetimcinin durum yazmaçları durumu kontrol ediliyor. Durum yazmacın dördüncü biti, yazıcının bilgisayara bağlı olup olmadığını hakkında bilgi veriyor, yedinci bit yazıcının meşguliyeti hakkında bilgi veriyor ve beşinci bit yazıcıda yazdırma için kağıt olup olmadığını kontrol ediyor. Veri yazmacı, yazdırılması ge-



Resim 4.8. Giriş çıkış denetimcinin blok diyagramı

reken karakteri saklıyor, komut yazmacı aracılığıyla ise mikroişlemci, yazıcının gerçekleştirmesi gereken işlemi veriyor.

Giriş/çıkış denetimcilerde tüm yazmaçların kendi adresleri var ve bu adreslere arabirim adresleri denir. Arabirim adresleri bellek adresleri gibi, mikroişlemcinin adres pinlerinden gönderiliyor ve şu soru ortaya çıkıyor, adres bitleri ne zaman bellek adresi ne zaman da arabirim adresi tanımlıyor.

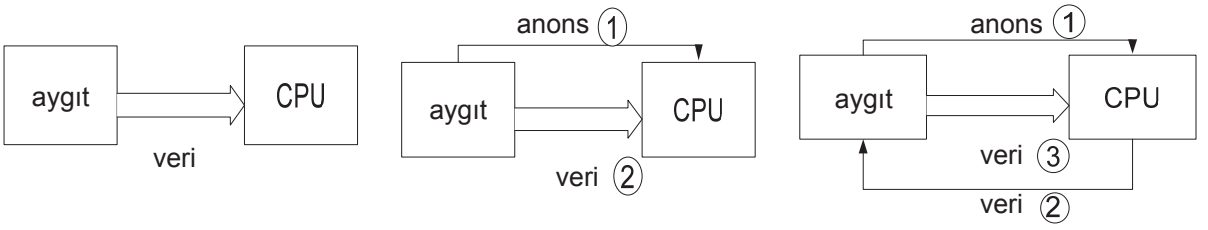
Dış aygıtların adreslenmesi iki şekilde gerçekleşebilir:

- izole giriş/çıkış
- belleksel kopyalanmış (çoğaltılmış) giriş/çıkış

İzole giriş/çıkışta belleği giriş - çıkış aygıtlardan ayırtmak için İO/$\overline{M}$ (input output/$\overline{memory}$) sinyali kullanılıyor.

Belleksel kopyalanmış giriş - çıkışta İO/$\overline{M}$ sinyali kullanılmıyor. Bu yaklaşımda belleği dış aygıtlardan ayırtmak için adres biti, genelde en değerli bit kullanılıyor.16 adres hatımız varsa, o zaman A_{15}=0 olunca, adres bellek adresidir. Eğer A_{15}=1 ise, o zaman mikroişlemciden adres dış aygıt için arabirim adresi olacak. Belleksel kopyalanmış giriş/çıkış'ın avantajları dış aygıtlara ya da dış aygıtlardan aktarım ve bellek için kullanılan aynı aktarma yönergelerin kullanılmasıdır. Bu şekilde yönergeler kümesi büyük ölçüde genişleniyor. Buna farklı olarak izole giriş/çıkış'ta mikroişlemci ve dış aygıt arasında iletişim için sadece özel aktarma yönergeleri kullanılıyor. Ancak, kopyalanmış giriş/çıkış'ın dezavantajları da vardır. Herşeyden önce, adres bitlerin giriş/çıkış aygıtlar için harcanması. Bellek alanı otomatik olarak azalıyor. Diğer bir dezavantajı alınan yönergelerin daha uzun yönerge kodlarından dolayı daha uzun zaman sürmeleridir.

Mikroişlemci ve bazı dış aygıt arasında veri iletimi başlamadan önce, sağlam iletim sağlamak için birbirine kontrol sinyalleri gönderiyorlar. Kontrol sinyallerin türüne göre **dört paralel iletim düzen** fark ediyoruz: basit, anonslu, tokalaşma düzeni ve çift tokalaşma. Resim 4.9.'da ilk üç düzenin kontrol sinyalleri ve onların blok modelleri verilmiştir. Çemberlerle sinyalleri gönderme sıralaması işaretlenmiştir.



a) Basit düzen b) Anonslu düzen c) Tokalaşma düzeni

Resim 4.9. Paralel iletim düzenlerin blok diyagramları

Veriler gönderen aygıt anons sinyalli gönderebiliyor (İngilizce – strobe), veriler kabul eden aygıt ise anonsu almış onayı gönderebilir (İngilizce – acknowledge). Basit iletim düzeninde, mikroişlemci ve giriş çıkış aygıtları kulanıcı bilginin aktarımı başlamadan önce birbirine kontrol sinyalleri göndermiyorlar. Basit iletim düzeni en az belirsiz iletimdir, çünkü kullanıcı bilgisi kolayca kaybolabilir. Basit düzen en belirsiz düzendir, ancak aynı zamanda en ekonomiktir, çünkü donanımda tasarruf yapılıyor. Anonslu düzende veriler gönderen aygıt iletimi anons (strobe) sinyalin aktıfleştirilmesiyle iletimi anons ediyor. Anonslu düzen mikroişlemci ve klavye arasındaki iletişimde kullanılıyor. Tokalaşma (handshaking) düzeninde aygıtlar birbirine kontrol sinyalleri gönderiyorlar: anons sinyali (strobe) ve onaylama sinyali (acknowledge). Çift tokalaşma düzeni en sağlamdır, ancak aynı zamanda en pahalıdır, çünkü her dört bakır hatı kontrol sinyali göndermek için kullanılacak. Ayrıca, ne kadar fazla sinyal varsa, iletim o kadar daha uzun sürecek

4.6. Pratik Giriş Çıkış Bağlantı Noktaları

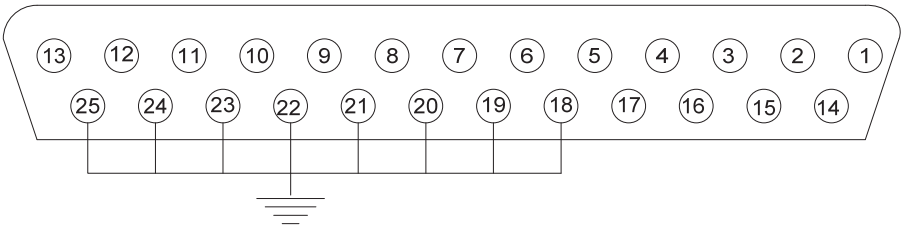
En çok kullanılan bilgisayar bağlantı noktaları şunlardır: paralel bağlantı noktası, dizisel (seri) bağlantı noktası ve USB bağlantı noktaları. Dış aygıtlarını bağlayabilmemiz için bağlanma noktalarda pinlerin sıralamasını bilmemiz ve onların sinyallerini anlamamız önemlidir.

Paralel bağlantı noktaları yazıcıların bağlanması için kullanılıyor, önümüzdeki derslerde ise mikrodenetimcilerin programlanması için de kullanıldığını göreceğiz. Paralel bağlantı noktasında en yüksek sayıda pinler vardır. Bilgisayarın üç paralel bağlantı noktası olabilir: LPT1, LPT2 ve LPT3, ancak genelde bir bağlantı noktası var LPT1. Paralel bağlantı noktası üç yazmaçla kontrol ediliyor: veri, komut ve durum yazmacı. Bu yazmaçlar üç ardaşıl arabirim adresi kullanıyorlar. Bu yazmaçların standart on altılı adresleri tablo 4.2.'de verilmiştir.

Bağlantı noktası	Veri bağlantı noktası	Durum bağlantı noktası	Komut bağlantı noktası
LPT1	378H	379H	37AH
LPT2	278H	279H	27AH
LPT3	3BCH	3BDH	3BEH

Tablo 4.2. Paralel bağlantı noktanın yazmaçları için standart arabirim adresleri

Bilgisayarın yazıcıyla bağlanma kablosunun iki bağlantı ucu vardır: DB25 ve CENT39. DB25 bağlantı ucu bilgisayarın çıkışını tanımlıyor, CENT39 bağlantı ucu ise yazıcının girişini tanımlıyor. Bağlantı uclarda pinlerin sıralaması resim 4.10'da gösterilmiştir



Resim 4.10. DB25 paralel bağlantı noktası

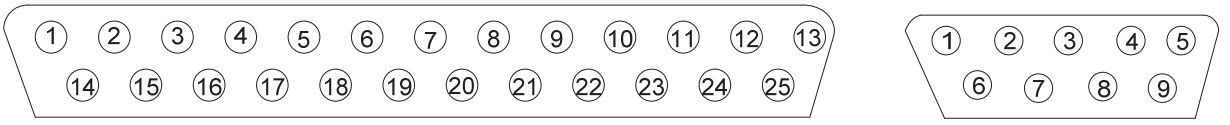
Tablo 4.3.'te pinlerin herbiri için sinyaller verilmiştir.

Pin numarası		sinyal	işlev
DB25	CENT39		
1	1	$\overline{Data\ strobe}$	Mikroişlemci anons sinyali gönderiyor
2	2	Data 0	Verinin birinci biti
3	3	Data1	Verinin ikinci biti
4	4	Data2	Verinin üçüncü biti
5	5	Data3	Verinin dördüncü biti
6	6	Data4	Verinin beşinci biti
7	7	Data5	Verinin altıncı biti
8	8	Data6	Verinin yedinci biti
9	9	Data7	Verinin sekizinci biti
10	10	$\overline{Ack}$	Yazıcı veriyi aldığı onaylama sinyali gönderiyor
11	11	Busy	Yazıcı meşguldur
12	12	Paper empty	Kağıt yok
13	13	Select	Yazıcı bağlıdır
14	14	Afd	Yazdırmanın bir sıra aşağıya kayması

15	32	$\overline{Error}$	Yazdırılma sırasında hata
16	-	Int	Yazıcı karakerin yazdırılması için başlatılıyor, ondan sonra da sıfırlanıyor
17	31	Select in	Yazıcı seçiliyor
18 - 25	19 - 30	Topraklanma	/
-	17	Topraklanma	/
-	16	Topraklanma	/
-	33	Topraklanma	/

Tablo 4.3. DB25 paralel bağlantı nokta pinlerinin işlevsel açıklaması

Bilgisayarın **dizisel bağlantı noktası** (COM) dış modem, fare, çizici ve benzer aygıtları bağlamak için kullanılıyor. Ayrıca dizisel bağlantı Windows işletim sistemi kullanan küçük yerel ağ yapmak için de kullanılabilir. Dizisel bağlantı noktası iki tür bağlantı ucu kullanılyor: D - SUB (25 pinli) ve D - SUB (9 pinli). Bağlantı ucların şekli ve pinlerin sıralaması resim 4.11'de gösterilmiştir.



DSUB25 DSUB9

Resim 4.11. Dizisel bağlantı noktaları

Tablo 4.4.bu iki bağlantı uçların daha önemli sinyallerin anlamını açıklıyor.

D - SUB 25	D - SUB 9	Sinyal	İşlev
Pin 2	Pin 2	TD - Transmit Data	Verilerin dizisel girişi
Pin 3	Pin 3	RD - Receive Data	Verilerin dizisel çıkışı
Pin 4	Pin 7	RTS - Request To Send	Modemin veri değişimi için hazırlık başlatıyor
Pin 5	Pin 8	CTS - Clear To Send	CD sinyalının alınmasından sonra aktiflştiriryor
Pin 6	Pin 6	DSR - Data Set Ready	Çalışmak için hazırlık başlatıyor
Pin 7	Pin 5	SG - Signal Ground	Tablo
Pin 8	Pin 1	CD - Carrier Detect	Bağlantı kurulmuştur
Pin 20	Pin 4	DTR - Data Terminal Ready	Sinyalın alnmasından sonra DTR'nin hazırlamasını başlatıyor
Pin 22	Pin 9	RI - Ring Indicator	Telefon hat sinyalı bulununmuştur

Tablo 4.4. Dizisel bağlantı pinlerinin işlev açıklaması

Veri gönderme protokolunu iki bilgisayar A ve B arasında veri değişimini, örneğin yardımıyla açıklayacağız. Aktarma protokolunu üç aşamaya ayırabiliriz: bağlantı kurmak, veri iletimi ve bağlantı kopması.

Bağlantı kurmak aşamasını üç adıma ayırabiliriz:

1. A bilgisayarı DTR sinyalini aktifleştirerek A modemine veri değişimi istediğini gösteriyor. Ondan sonra telefon numarasını TD pinin aracılığıyla B modemine iletiyoruz.
2. A modemi B modemiyle bağlantı kurunca, B modemi Rİ pinini aktifleştirerek bilgisayarına gelen çağrıyı duyuru ediyor. B bilgisayarı verileri kabul etmek için hazırsa, o zaman DTR sinyalini aktifleştirecek. İzin alındıktan sonra B modemi DSR sinyalini aktifleştirerek A bilgisayarında veri almaya hazır olduğunu söylüyor.
3. Geri bilgiyi aıdıktan sonra, A modemi CD pinini etkinleştirerek A bilgisayarına bildiriyor, aynı zamanda DSR pinini aktifleştirerek bağlantı kurma aşaması sona eriyor.

Veri aktarımı aşaması için tokalaşma düzeni kullanılıyor ve bu arada kontrol sinyalleri RTS ve CTS pinlerin aracılığıla karşılıklı olarak değişiyor. Veriler TD pinlerin yardımıyla gönderiliyor.

Bağlantının kopması, sadece RST (Request To Send) sinyalini iki taraftan devre dışı bırakarak gerçekleşiyor ve bu işlem diğer taraftan CTS sinyallein devre dışı kalmalaıa yol açıyor.

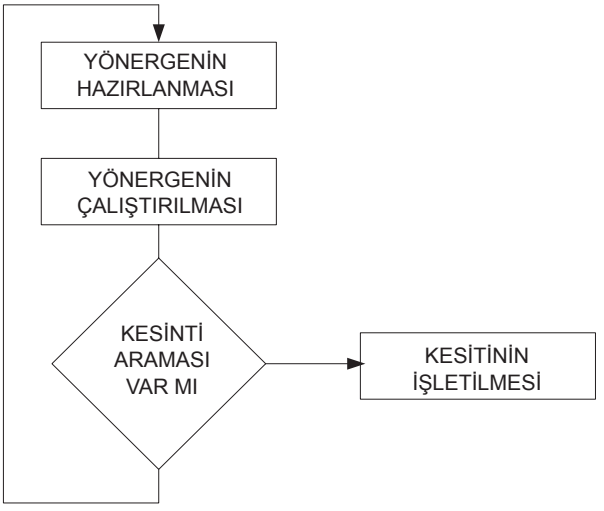
4.7. Kesintili (interrupt) Sistemler

Dış aygıtlardan bazısı mikroişlemciden mevcut programın çalıştırılmasının kesilmesini ve bu dış aygıtın onarmasını yapan alt programın çalıştılmasına geçmeyi araması sıkça meydana gelen bir olaydır. Kesintiler sadece dış aygıtlardan kaynaklanmayabilir. Kesinti, mikroşlemcinin içinde kuraldışı bir durum elde edilince de meydana gelebilir. Bu tür kesintilere iç kesinti denir. İç kesintilerine karşı mikroişlemci dış kesintilere karşı davrandığı gibi davranıyor. Dış aygıtları mikroişlemciye özel kesinti aramaları gönderiyor. Bu amaçla INTR (interrupt) özel pini aktifleştiriliyor. Mevcut programın her yönergenin çalıştırılmasından sonra, mikroişlemci kesinti aramanın olup olmadığını kontrol ediyor. Böyle bir arama varsa, o zaman mikroişlemci mevcut işlemi kestirerek, kesintinin çalıştırılmasına geçebilir.

Bu arada INTA (Interrupt acknowledge) pini aktifleştiriliyor.Bu şekilde mikroişlemci dış aygıtını kesinti aramasını kabul ettiğini bildiriyor. Kesinti durumu resim 4.12.'de gösterilmiştir. Kesinti acil değilse, o zaman reddedilebilir ya da bekleme durumuna koyulabilir.

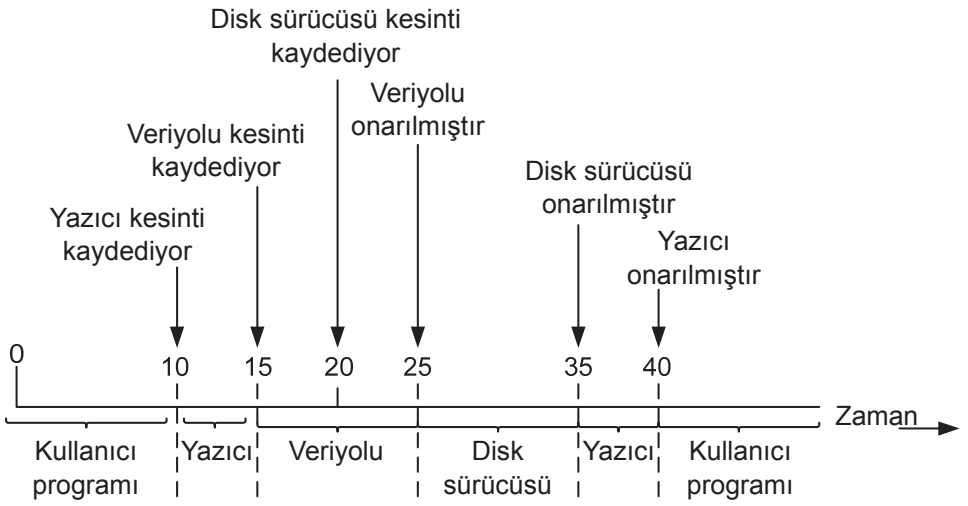
Mikroişlemci kesinti aramasını kabul ederse, o zaman şu etkinlikler alınacaktır:

1. Dış aygıt tanımlama için mikroişlemciye kesinti vektörü adlı kod (8 bitten oluşan kod) gönderiyor.
2. Kesinti vektörü mikroişlemciye kesintinin onarımı için programı (interrupt service routine) daha kolay bulmasına yardım ediyor.
3. Kesinti işlendikten sonra, mikroişlemci mevcut programı kesintiden önce kaldığı yerden çalıştırmaya devam ediyor.



Resim 4.12. Kesinti onarımının blok diyagramı

Eğer aynı zamanda mikroişlemciye birkaç dış aygıtan birkaç kesinti araması varsa, ne olacağı sorusu ortaya çıkıyor. Kesintilerin işletilmesi sırasında karışıklık olmasın diye, bazı dış aygıtlara diğerlerine kıyasen daha büyük öncüllük veriliyor. Resim 4.13.'te öyle bir örnek verilmiştir. Mikroişlemciye 3 aygıttan kesinti araması geliyor. Veriyolunun en yüksek öncüllüğü vardır, ondan sonra disk sürücüsü geliyor ve yazıcının en alçak öncüllüğü vardır. Mikroişlemci bazı mevcut program çalıştırıyor ve birdenbire t=10 zaman anında yazıcı tarafından başlatılan kesinti meydana geliyor. Mikroişlemci mevcut programı durduruyor ve bu kesintinin işletilmesi için program başlatılıyor. Bu program 10 zaman birimi sürüyor. Ancak, t=15 anında veriyolu kesinti araması gönderiyor. O zaman mikroişlemci yazıcı için alt programı durduruyor ve veriyoldan kesintinin işletilmesine geçilecek (veriyolun yazıcıya kıyasen daha yüksek öncüllüğü vardır).



Resim 4.13. Farklı öncüllüklü kesintilerin işletilmesi

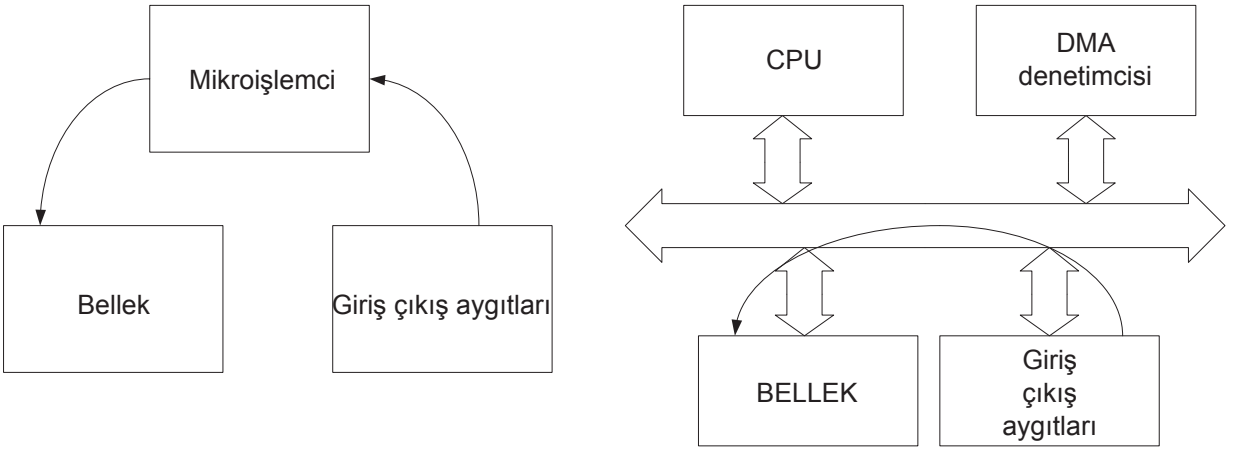
Veriyolu onarılırken, disk sürücüsü tarafından yeni, üçüncü kesinti araması gönderiliyor. Mikroişlemci önce veriyolundan gelen kesintinin işletilmesi için programı sona erdiriyor ve bu işlem t=25'e kadar sürüyor. Aynı anda mikroişlemci disk sürücüden kesintinin işletilmesi için programı başlatıyor. Disk sürücünün kesintisini işlettikten sonra t=35 zaman biriminde mikroişlemci yazıcıdan kaynaklanan kesintiyi işleten programı sona erdirmek için dönüyor. Demek ki, yazıcı ilk olarak kesinti araması göndermesine rağmen, bu arama son olarak işletilecek, çünkü yazıcının en alçak öncüllüğü vardır. Mikroişlemci aldığı tüm kesinti aramalarını işlettikten sonra, mevcut programa geri dönüyor.

Mikroişlemcilerin en büyük bölümünde iki kesinti türü fark ediyoruz – maskeli ve maskesiz kesintiler. Maskelemenin anlamı alınan kesintinin işletilmesi için mikroişlemcinin mevcut programın kestirilmesinin yasaklanması demektir. Maskeli kesintilerde mikroişlemci önce maskenin olup olmadığını tespit ediyor ve maske yerleşmiş ise, mikroişlemci gelen kesinti aramasını önemsemeden sıradaki yönergenin çalıştırılmasıyla devam ediyor. Kesintilere maskenin yerleştirilmesi için özel yönerge vardır - SIM (Set Interrupt Mask). Kesinti maskenin okunulması için RIM (Read Interrupt Mask) yönergesi kullanılıyor.

4.8. DMA Aktarımı

Bazan mikroişlemci çalışmasında zamanın %20'ni verilerin aktarılmasında harcıyor. DMA (Direct Memory Access) kısaltması aslında, mikroişlemcinin aracalığı olmadan belleğe doğrudan erişim demektir. DMA aktarım kavramı, mikroişlemciyi bellek ve giriş çıkışların arasında gerçeklenen veri değişiminden serbestlemek tanımlıyor.

Bu kavramla mikroişlemcinin yüklenmesinin meydana gelmemesi yanısıra, daha güvenli veri aktarımı sağlanıyor, yani verilerin kaybolması azalıyor. Güvenli veri aktarımı özellikle bazı çok hızlı dış aygıtından veri aktarımı için geçerlidir. DMA konsepti özel DMA denetimcinin kullanımıyla gerçekleşiyor. DMA kavram sistemi resim 4.14.'te gösterilmiştir.



Resim 4.14. DMA kavramlı ve DMA kavramsız sistemler arasında kıyaslama

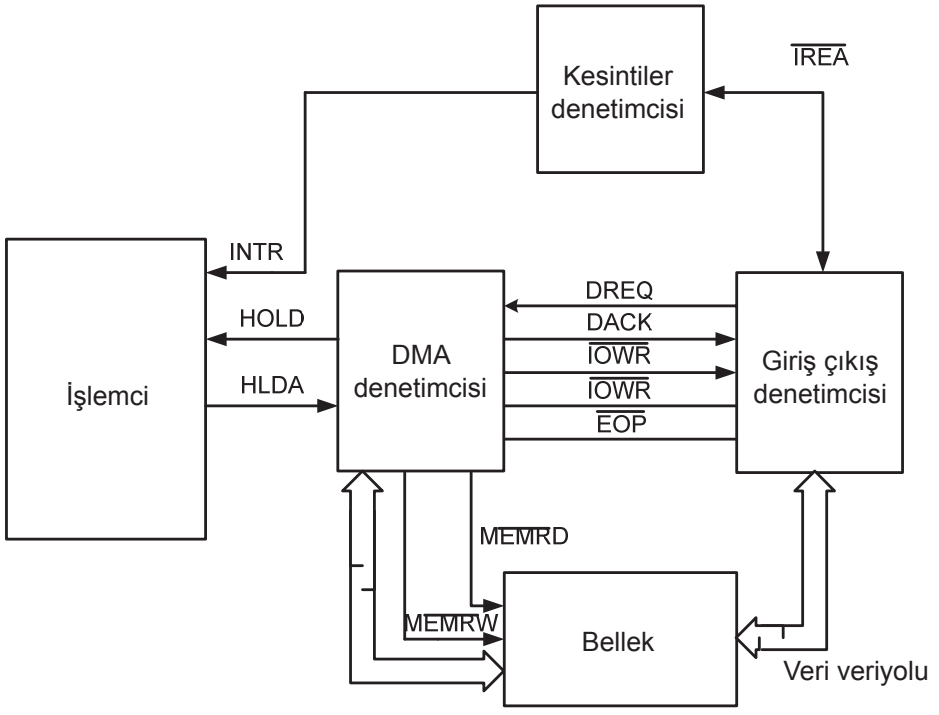
DMA denetimcisi mikroişlemci için köle (slave) olarak tanımlanıyor. Bazı giriş çıkış aygıtından belleğe ya da ters yönde veri bloğun aktarılması gerektiği zaman, mikroişlemci: aygıtın arabirim adresini, bellek adresini, aktarılması gereken baytların sayısını ve aktarım türü için kontrol sinyalini (giriş ya da çıkış aktarımı) gönderiyor. Aktarma yönergesini aldıktan sonra, DMA denetimcisi veriyolların hakimi (master) oluyor ve aktarımı gerçekleştirmek için kontrol sinyalleri üretmeye başlıyor. DMA denetimcisi veriyolların ayırımını (tahkimini), bellek adresinin alınmasını ve RD ve WR kontrol sinyallerin üretimini gerçekleştiriyor. Gerçeklenen aktarımın ardından veriyollarını serbestleştiriyor, bellek adresinin ve bayt sayısının yenilenmesini yapıyor ve eğer aktarmak için daha baytlar varsa tüm süreci yeniden tekrarlıyor. Aktarımın sonunda mikroişlemci yeni kesinti alıyor ve aktarımın durumunu kontrol ediyor (başarılı ya da başarısız). Eğer daha büyük miktar verini hızlı aktarımı söz konusu olursa, o zaman DMA denetimcisi bir baytın ya da sözcüğün her sona eren aktarımdan sonra veriyolların kullanımını aramaya gerek yoktur.

DMA aktarımın göstermek için resim 4.15.'te verilen basit bir örnek açıklayacağız. Resim 4.15.'te DMA akımında yer alan tüm aygıtların: mikroişlemci, DMA denetimcisi, bellek ve giriş/çıkış denetimcisinin birbirine bağlanması gösterilmiştir. Adres ve veri veriyoluna mikroişlemci ve DMA denetimcisinin erişimi vardır, ancak örneğin basitleştirilmesi amacıyla mikroişlemcinin veriyollarıyla bağlantısı gösterilmemiştir. Bir DMA kanalı sadece bir giriş/çıkış aygıtına veri aktarımı için kullanılıyor. Bir DMA

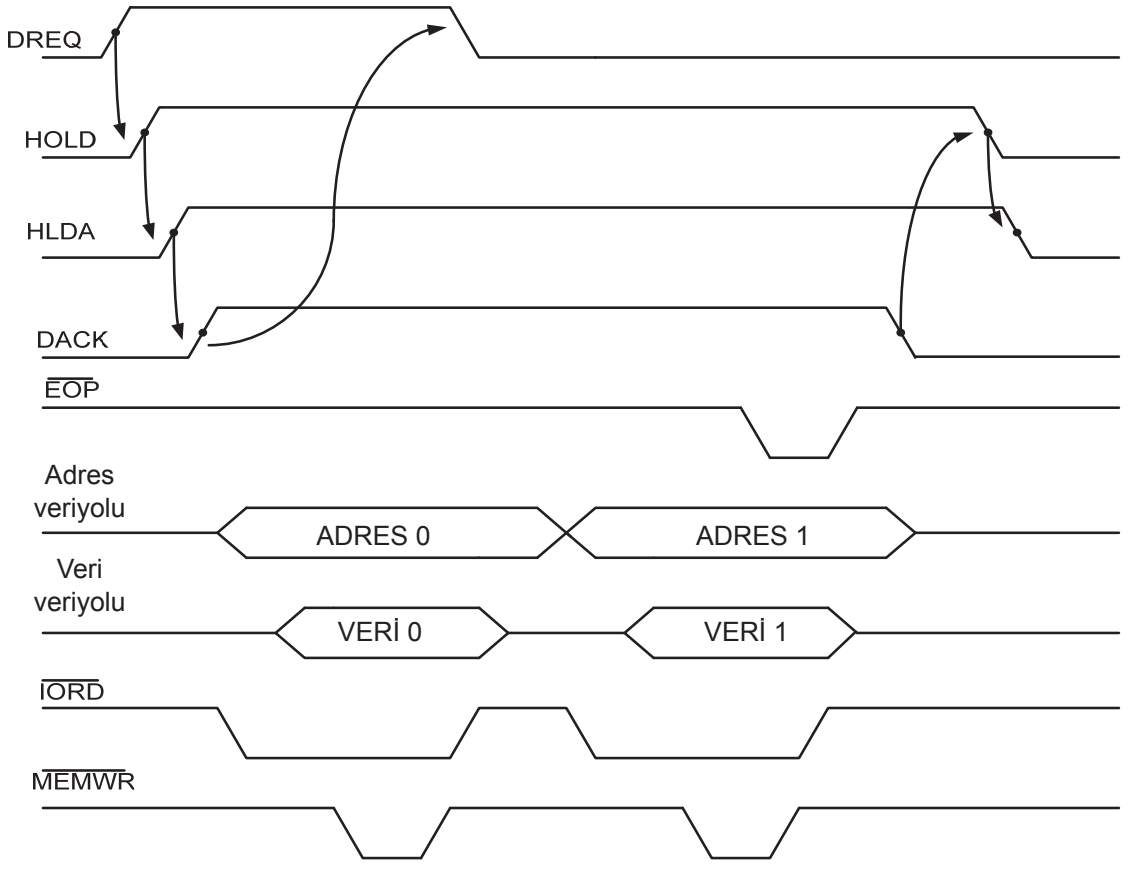
kanalını fazla aygıtın ayırması ya da aktarılan verinin fazla aygıta ulaşması gibi olaylar meydana gelemez.

Okuma işleminin gerçekleşmesini tahmin edelim (giriş aygıtı) ve iki sözlük (iki 16 - bitli veri) bilginin aktarılması gerektiğini de tahmin edelim.

Resim 4.15. DMA denetimcisinin bağlanma şekli



Resim 4.16.'da DMA aktarma işleminin zaman diyagramı verilmiştir. Giriş aygıtı aktarım için hazır olunca DMA denetimcisine DREQ (DMA request) hattı aracılığıyla arama gönderiyor. Bu sinyalın alınmasından sonra DMA denetimcisi HOLD sinyalini yüksek seviyeye çıkararak mikroişlemciye veri aktarımı yapmak istediğini anons ediyor. Mikroişlemci HLDA (Hold Acknowledge) sinyalini aktifleştirerek izin veriyor ve bununla DMA denetimcisi veriyolların hakimi oluyor. Devamda DMA denetimcisi, giriş aygıtını DMA aktarımı başlayabileceğini bildirmek için DACK (DMA Acknowledge) sinyalini gönderiyor. DACK sinyali gönderiler aramaya DREQ cevap, onaylama tanımlıyor. DMA denetimcisi giriş aygıtından $\overline{IORD}$ okuma sinyalini ve bellekte yazdırma $\overline{MEMWR}$ sinyalini üretiyor. Giriş aygıtı kontrol sinyallere cevap veriyor ve verileri veri veriyoluna yerleştiriyor. Veri aktarıldıktan sonra bellek adresinin değeri artıyor, baytların sayısı azalıyor ve bayt sayısı sıfır değilse yeni süreç başlayabilir. Tüm baytların aktarımı sona erdiğinde DMA denetimcisi EOP (End Of Process) sinyalini gönderiyor. Aynı zamanda DACK ve HOLD sinyalleri sönüyor ve mikroişlemci yeniden veriyolların hakimi oluyor, DMA denetmcisi ise köle oluyor.



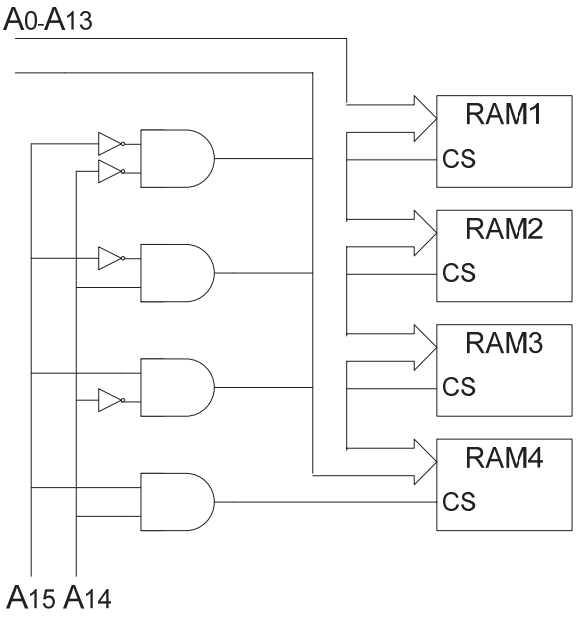
Resim 4.16. DMA denetimcinin kontrol sinyallerinin zaman diyagramı

Yavaş giriş aygıtı söz konusu olursa, o zaman mikroişlemciden veriyolları tamamiyle alınmasına gerek yok. DMA aramanın aktarılması gereken tüm baytlar için geçerli olması yerine, sadece bir bayt için geçerli olacaktır ve gönderilen her veriden sonra DREQ sinyalı aktifleştirilecek.

4.9. Adresli Kod Çözümlenmesi

Bir bilgisayarda büyük sayıda bellek ve dış aygıtlar bulunuyor. Mikroişlemci aynı zamanda iki ya da fazla aygıtla iletişim kuramaz, verildiği anda sadece bir aygıtla iletişim kurabilir. Adresli kod çözümlenmesi birçok aygıttan sadece birinin seçilme sürecidir. RAM belleğinden bir veri okurken, önce bellek yongası seçiliyor, ondan sonra da seçilen yongadan bir bellek yeri seçiliyor. Hatırlayalım, bellek yonganın seçimi, CS, CE ve S seçme pinlerinin aktifleştirilmesiyle gerçekleşiyor. Mikroişlemcinin adres hatlarının bir bölümü bir bellek yongasından seçim sinyali almak için kullanılıyor, diğer hatlar ise aranan bellek yerine ulaşmak için kullanılıyor.

Adresli kod çözümleme sürecini resim 4.17.'de gösterilen örnek yardımıyla açıklayacağız



Resim 4.17. İki adres girişli adresli kod çözücüsü

Bu örnek çok basittir ve gerçek dışıdır, ancak adresli dok çözümleme sürecini anlamak için yardım edecek. Adres hatların toplam sayısı 16'dır, A_0'dan A_{15}'e kadar. Buna göre 2^{16}=64KB büyüklükte adres alanı elimizde var. Adres alanı dört RAM yongası arasında ayrılmıştır. Her yonganın kapasitesi 8KB=64KB:4 değerindedir. Herhangi bir yongadan bir yerin seçilmesi için A_0'dan A_{13}'e kadar adres hatları kullanılıyor. İki en değerli adres hatları A_{14} ve A_{15} hatları dört yongadan birini seçmek için kullanılıyor. Yonga seçimi tablo 4.5'e göre yapılıyor.

	A_{15} A_{14}
RAM1	00
RAM2	01
RAM3	10
RAM4	11

Tablo 4.5. İki adres çıkışlı adres çözücünün doğruluk tablosu

Birinci yonganın tüm adresleri 00 ile başlıyor, ikincinin 01 ile, üçüncünün 10 ve dördüncünün 11 ile başlıyor. Kalan adres bitleri değişiyor. A_{13} 'dan A_0'a kadar olan bitler başlangıç adresleri için sıfırdır, son adresler için birdir. Resim 4.18.'de her dört yonganın başlangıç ve son adresleri, yani her dört yonganın bellek haritası verilmiştir. A_{15} ve A_{14} bitlerin değerlerine bağlı olarak, belli bir adresin hangi bellek yongasına ait olduğunu belirleyebiliriz.

		A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
RAM1	başlangıç	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	son	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
RAM2	başlangıç	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	son	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
RAM3	başlangıç	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	son	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
RAM4	başlangıç	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	son	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Resim 4.18. 16KB'lı dört yongadan oluşan RAM belleğin bellek haritası

Örnek 4.1. Mikroişlemci 16 - bitli adres kullanıyor A_{15} - $A_{0.}$ Bellek modülü seçmek için, A_{15} ve A_{14} adres hatları şu mantıksal seviyede ayarlanmıştır:

$A_{15} = 1, A_{14} = 0$

Verilen on altılı adreslerden hangileri yukardaki bellek modülüne aittir?

A) 6100H B) 9100H C) C100H Ç) F100H

Çözüm: Önce bir satırda A_{15}'ten A_0'a kadar tüm adres bitlerini yazıyoruz. Ondan sonra, on altılı adresleri ikili adreslere dönüştüyoruz ve onları adres bitlerinin satırı altında yazıyoruz. Son olarak, A15 v A14 adres bitlerini çevreliyoruz ve adreslerden hangisi yukarıdaki koşullu yerine getirdiğini bakıyoruz.

	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
6100H	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0
9100H	**1**	**0**	0	1	0	0	0	1	0	0	0	0	0	0	0	0
C100H	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
F100H	1	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0

Aradığımız adres 9100H adresi olduğunu görüyoruz.

Adres hatların sayısı ve yongaların sayısı değişen büyüklüklerdir.

Örnek 4.2: Belleğin 2MB toplam kapasitesi var ve 8 birbirine aynı yongadan oluşuyor. Yonganın seçilmesi için kaç adres hattın kullanıldığını belirle ve seçilen yongadan bellek yerin seçilmesi için hangi adres hatları kullanılıyor?

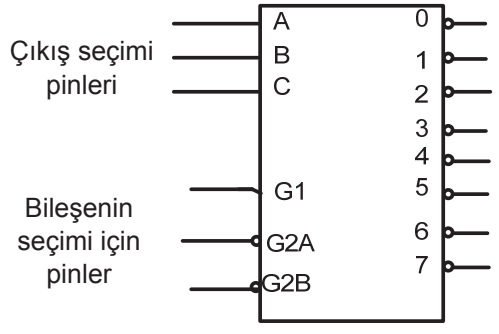
Çözüm: Toplam bellek kapasitesi adres hatların toplam sayısını belirlemizi sağlıyor. 2MB= $2^1 \cdot 2^{20} = 2^{21}$ 21 adres hatların toplam sayısını tanımlıyor ve onlar A_{20}'den A_0'a kadar işaretleniyor.

	$A_{20}A_{19}A_{18}$
RAM1	000
RAM2	001
RAM3	010
RAM4	011
RAM5	100
RAM6	101
RAM7	110
RAM8	111

Tablo 4.6. 256 KB kapasiteli 8 yonganın adres çözümlenmesi için adres bitleri

8 yonga olduğu için, 8 farklı kombinasyon elde edebilmemiz için üç adres biti gerekecek. Her zaman en değerli bitler seçiliyor ve bu durumda en değerli bitler A_{20}, A_{19} ve A_{18} bitleridir. Kalan adres hatları, A_{17}'den A_0'a kadar, bir bellek yonganın kapasitesi olduğu toplam 256KB'tan bir baytın seçilmesi için kullanılacak. Tablo 4.6.'da yonga seçimi için üç adres bitin değerleri verilmiştir.

Mikroişlemci tekniğinde kod çözücü tümleşik devreler çok sıkı kullanıyorlar. Resim4.19'da 74LS138 adres kod çözücüsü gösterilmiştir. Onun üç adres girişi vardır ve buna göre sekiz çıkışı olacaktır. G1, G2A, G2B girişleri bileşenin seçilmesi için kullanılıyorlar. G1 pini yüksek seviyede olmalıdır, G2A, G2B pinleri ise alçak seviyede olmalıdırlar. A, B ve C pinleri çıkış pinlerden birinin seçilmesi için kullanılıyorlar. Tablo 4.7.'de doğruluk tablosu verilmiştir.

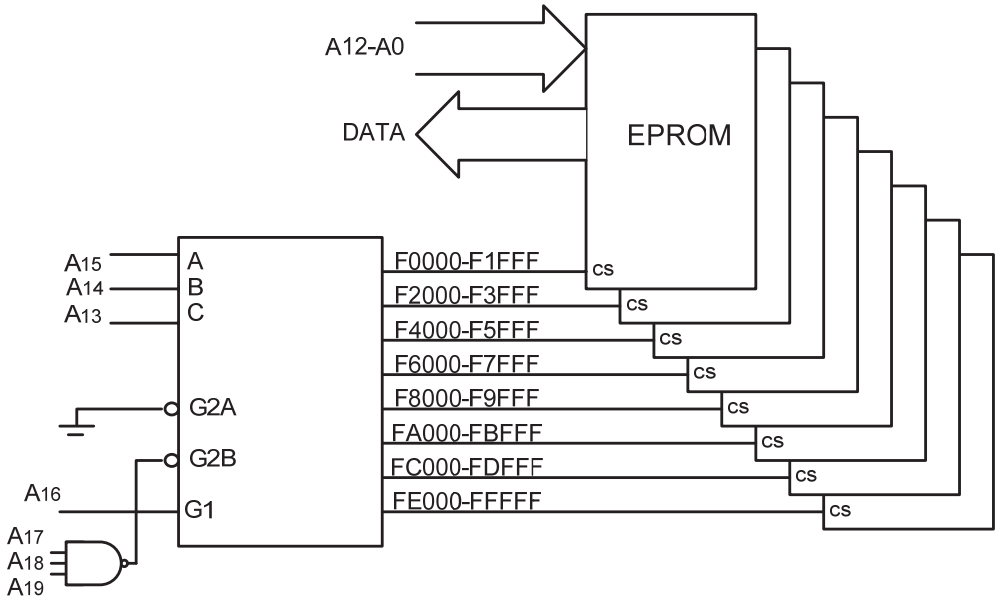


Resim 4.19. 74LS 138 kod çözücünün pin diyagramı

ABC	Çıkış
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Tablo 4.7. 74LS138 kod çözücünün doğruluk tablosu

74LS138 kod çözücüsü bellek adreslerin kod çözümlenmesi için ve arabirim adreslerin kod çözümlenmesi için kullanılabiliyor. Resim 4.20.'de 74LS138 kod çözücüden ve 8 EPROM yongadan oluşan basit kod çözümleme devresi gösterilmiştir.



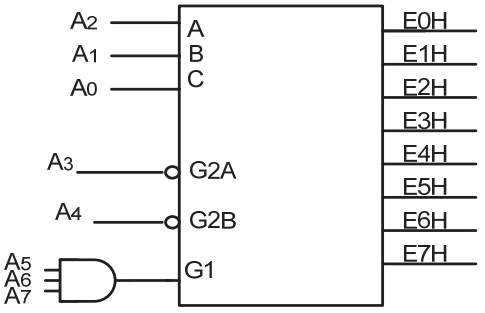
Resim 4.20. 74LS38 kod çözücünün bellek yongalarıyla bağlanması

Her yonganın 8KB kapasitesi var ve buna göre toplam kapasite 64KB değerindedir. Resim 4.20.'de her yonganın adres kapsamı verilmiştir. A13, A14 ve A15 adres bitleri kod çözücüsünden bir çıkışın seçilmesi için kullanılıyor, yani toplam sekiz bellek yongasından birini seçmek için. Bu bitlerin değerleri tablo 4.7.'ye göre değişiyor ve tabloda $A_{15}A_{14}A_{13}$=ABC olarak tanımlanıyor. A_{16} her zaman birdir, çünkü G1 pini mantıksal birde aktiftir. OVE devresinin çıkışında mantıksal sıfır elde etmek için A_{17}, A_{18} ve A_{19} bitlerin hepsi bir değerinde olmalıdır. G2B pinini aktifleştirmek için OVE devresinde mantıksal sıfır olmalıdır. A_{17}, A_{18}, A_{19} bitlerinde herhangisi sıfıra eşitse, o zaman OVE devresinin çıkışında mantıksal bir (1) elde edeceğiz, G2B pini aktifleştirilmeyecek ve sonuç olarak 74LS138 kod çözücüsü etkinsiz kalacak.

74LS138 kod çözücüsünün 8 çıkışı var ve bu yüzden 74LS138 kod çözücüyle 8 arabirim adresi çözümlenebilir. Hatırlayalım, arabirim adresi 8 bitten tekrarlanmayan kombinasyondur ve dış aygıtın seçilmesi için kullanılıyor. Resim 4.21'de gösterildiği gibi arabirim - adresi 74LS138 kod çözücünün girişine getiriliyor.

A_7'den A_4'e kadar adres bitleri, bileşenler seçen pinlerin aktifleştirilmesi için gereken koşulları yerine getirmelidirler. Yerine getirilmesi gereken koşullar G2A, G2B ve G1 pinleri önünde yerleşmiş olan mantıksal devrele bağlıdır. A7'den A3'e kadar beş bit sabittir ve 74LS138 kod çözücünün çıkışlarına bağlı olan her sekiz dış aygıt için geçerlidir. A_2, A_1 ve A_0 bitleri dış aygıtın seçilmesi için kullanılıyor. G2A ve G2B pinlerinin aktifleştirilmesi için A_3=0 ve A_4=0 olmalıdır. G1 pini için mantıksal bir (1) ge-

rekiyor. VE devrenin çıkışında mantıksal bir (1) elde etmemiz için, tüm onun girişleri bir olmalıdır, yani $A_5=A_6=A_7=1$ olmalıdır.



A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
1	1	1	0	0	0	0	0	=E0H
1	1	1	0	0	0	0	1	=E1H
1	1	1	0	0	0	1	0	=E2H
1	1	1	0	0	0	1	1	=E3H
1	1	1	0	0	1	0	0	=E4H
1	1	1	0	0	1	0	1	=E5H
1	1	1	0	0	1	1	0	=E6H
1	1	1	0	0	1	1	1	=E7H

Resim 4.21. E0H'dan E7H'ye kadar adres kapsamalı 74LS138 kod çözücüsü için doğruluk tablosu

Yorum: G1 pinin girişinde O - YADA mantıksal devresi bulunuyorsa, o zaman O - YADA devrenin çıkışında mantıksal bir (1) elde etmek için A_7, A_6 ve A_5 bitleri sıfır olmalıdır. O zaman adresler 00H'dan 07H'ye kadar olacaktı.

Sonuçlar:

Bellekten okuma işlemi üç dijit palsı kadar sürüyor. Birinci palsta mikroişlemci belleğe adresi gönderiyor. İkinci pals bekleme palsıdır ve bu sürede bellek aranan bellek konumunu arıyor ve buluyor. Üçüncü palsta bellek aranılan veriyi mikroişlemciye gönderiyor. Eğer belleğe erişim zamanı daha uzunsa, o zaman bekleme için iki ya da fazla pals eklenebilir.

Sanal bellek kavramına göre program bloklara ayrılıyor ve her blok mikroşlemcide diğerlerinden bağımsız olarak çalıştırılıyor. Programın birinci bloğun çalıştırılması tamamlanınca, birinci blok ana bellekten ikincil belleğe geri dönüyor, ikincil bellekten ana belleğe ise sıradaki blok gönderiliyor, ondan sonra üçüncü ve tüm programın çalıştırılmasına kadar o şekilde devam ediyor.

Sanal bellek teriminin kullanıma girmesiyle ana (birincil) bellek dış (ikincil) bellek sayesinde genişleniyor. İkincil bellek sayfalara ayrılıyor, ana bellek ise çerçevelere ayrılıyor. Çerçevelerin sayısı sayfalar sayısından çok daha azdır. İhtriyaçlara göre çerçevelerde yerleştirilmiş sayfalar değişebilir.

Amacına göre önbellek, yönerge önbelleği ya da veri önbelleği olabilir. Yönerge önbelleğinde ana bellekten önceden getirilmiş yönergeler saklanıyor ve mikroişlemci tarafından çalıştırılmaları için çağrılmalarını bekliyorlar. Veri önbelleği mikroişlemcinin daha sıkı çağırdığı ancak mikroişlemcinin genel yazmaçlarında bu veriler için yer olmayan verileri saklıyor.

Daha gelişmiş bellek organizasyonlarda birkaç önbellek seviyesi vardır. Üç önbellek seviyesi mevcuttur. Birinci seviyeli önbellek mikroişlemcinin yongasında bulunuyor, ikinci seviye önbelleği mikroişlemcinin paketlemesinde bulunuyor ve üçünci seviye önbelleği anakartında yer alıyor.

Arabellek sinyalleri işletmiyor, arabellek sadece veri aktarımı sırasında dış aygıtları veri veriyoluyla donanım bağlanmasını yapıyor. Mandal (latch) işlevi tetikleme sinyalının süresini uzatmak olan, palslı flip floplardan oluşan aygıttır.

Dış aygıtlar ve sistem veriyolu arasında iletişiminde aracılık yapan devrelere giriş/çıkıs denetimcileri ya da arabirim denetimcileri denir.

Dış aygıtların adreslenmesi iki şekilde gerçekleşebilir: izole giriş/çıkış ve belleksel kopyalanmış (çoğaltılmış) giriş/çıkış. İzole giriş/çıkışta, belleği giriş - çıkış aygıtlardan ayırmak için IO/M (input output/$\overline{memory}$) sinyali kullanılıyor. Belleksel kopyalanmış giriş/çıkışta, belleği dış aygıtlardan ayırmak için adres biti, genelde en değerli adres biti kullanılıyor. Eğer 16 adres hatımız varsa, o zaman A_{15}=0 olunca mikroişlemciden adres, bellek adresidir. Eğer A_{15}=1 olursa, o zaman mikroişlemciden adres dış belleğin arabirim adresi olacak.

Veri aktarımı başlamadan önce giriş aygıtı anons ya da duyuru (strobe) sinyali gönderiyor ve mikroişlemci aktarıma izin verirse o zaman mikroişlemci onaylama sinyali (Acknowledge) gönderiyor.

Paralel bağlantı noktaları yazıcıların bağlanması için kullanılıyor, önümüzdeki derslerde ise mikrodenetimcilerin programlanması için de kullanıldığını göreceğiz. Bilgisayarın dizisel bağlantı noktası (COM) dış modem, fare, çizici ve benzer aygıtları bağlamak için kullanılıyor. Ayrıca dizisel bağlantı Windows işletim sistemi kullanan küçük yerel ağ yapmak için de kullanılabilir.

Dış aygıtları mikroişlemciye özel kesinti aramaları gönderiyor. Bu amaçla INTR (interrupt) özel pini aktifleştiriliyor. Mevcut programın her yönergenin çalıştırılmasından sonra, mikroişlemci kesinti aramanın olup olmadığını kontrol ediyor. Böyle bir arama varsa, o zaman mikroişlemci mevcut işlemi kestirerek, kesintinin çalıştırılmasına geçebilir ve bu arada INTA (Interrupt acknowledge) pini aktifleştiriliyor.

DMA aktarım kavramı, mikroişlemciyi bellek ve giriş ilçıkış aygıtların arasında gerçeklenen veri değişiminden serbestlemek tanımlıyor. DMA konsepti özel DMA denetimcinin kullanımıyla gerçekleşiyor

HOLD pini mikroişlemci için giriş pinidir ve onun aracılığıyla DMA denetimcisi DMA aktarımı için arama gönderiyor. Mikroişlemci aktarım aramasını kabul ederse mikroişlemci HLDA (Hold Acknowledge) çıkış pinini aktifleştiriyor. DMA denetimcisi dış aygıttan DREQ pini aracılığıyla DMA aktarımı için aramayı alıyor, DACK sinyalini aktifleştirerek onay veriyor. DMA aktarımı EOP sinyalinin aktifleştirilmesiyle sona eriyor.

Adresli kod çözümlenmesi mikroişlemcinin iletişim kurması gereken sadece bir bellek ya da dış aygıttan seçilme sürecidir. Bellek yonganın seçimi, $\overline{CS}$, $\overline{CE}$ ve $\overline{S}$ seçme pinlerinin aktifleştirilmesiyle gerçekleşiyor. Mikroişlemcinin adres hatlarının bir bölümü bir bellek yongasından seçim sinyali almak için kullanılıyor, diğer hatlar ise aranan bellek yerine ulaşmak için kullanılıyor.

74LS138 adres kod çözücüsünün üç adres girişi ve sekiz çıkışı vardır. Onunla 8 arabirim adresi ya da bellek adresi çözümlenebilir. G1, G2A ve G2B girişleri bileşenin seçilmesi için kullanılıyor.

Sorular ve Odevler

1. Şu terimlerin anlamlarını açıkla: senkronizasyon, sinyalleşme, TTL uyumluluğu ve arabelleme.

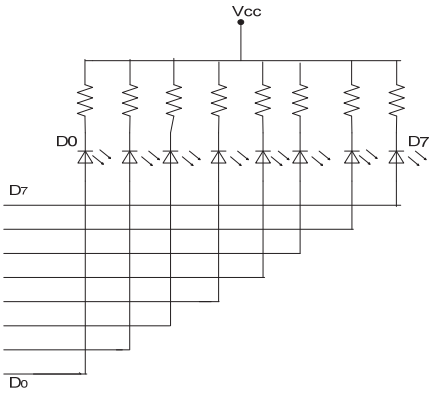
2. Mandal (latch) tetikleme sinyalin zaman süresini arttıran devredir. Bunu nasıl yaptığını açıkla.

3. Belleklerin genel işlemciyle bağlanması için hangi pinler kullanılıyor?

4. Wait sinyalin işlevini açıkla? Bu sinyali hangi aygıt gönderiyor ve hangi aygıt kabul ediyor?

5. Bellekten okuma işlemi için zaman diyagramında her üç dijit atışı sırasında hangi kontrol sinyalleri aktifleştiriliyor?

6. Mikroişlemci veri veriyolu aracılığıyla led diyodlara 54H bilgisini gönderirse, hangi led diyodlar yanacak? Nedenini açıkla!



7. Sanal bellek yaklaşımında bellek alanının genişleme şeklini açıkla?

8. Mikroişlemci 32 - bitli sanal adres kullanıyor. Bir sayfanın büyüklüğü 8 KB'tır. Sanal adres alanına kaç sayfa sığdırılabilir?

9. Sanal bellek 16 sayfa içeriyor, ancak sayfalar için sadece dört çerçevesi vardır. Başlangıçta ana bellek boşmuş. Program şu sayfalar için arama gönderiyor: 0, 7, 2, 7, 5, 8, 2, 4. LRU ve FİFO kavramları kullanılırsa hangi aramalar sayfa değişimine yol açacak?

10. Sanal bellek 1024 sözcük büyüklüğünde sekiz sayfa kullanılıyor ve dört sanal çerçeve içeriyor. Aşağıda sanal belleğin durumu gösterilmiştir.

Sanal sayfa	Sayfanın çerçevesi
0	3
1	1
2	yoktur
3	yoktur
4	2
5	yoktur
6	0
7	yoktur

Hangi sanal adresler eski bellek sayfanın yenisiyle değişmesini başlatacak?

11. Önbellekte hangi veriler saklanıyor?

12. Önbellek hattı ve önbellek giriş terimlerini açıkla.

13. Önbellek girişindeki bitlerin anlamını açıkla ?

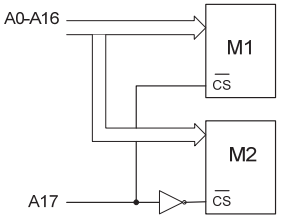
14. DB25, DSUB25 ve DSUB9 bağlantı noktaların kaç pinleri vardır?

15. Bilgisayarın paralel bağlantı noktaları nerede kullanılıyor?

16. Adresli kod çözümleme süreciyle hangi sinyaller üretiliyor?

17. Mikroişlemci 16 - bitli adres kullaniyor: A_{15} - $A_{0.}$ Bellek modülün seçilmesi için, A_{15} ve A_{14} adres hatları şu mantıksal seviyelerde ayarlanmıştır: A_{15}=0 A_{14}=1. Verilen on altılı adreslerden hangisi verilen bellek modülüne aittir?
A) 6500H B) 5400H C) A700H Ç) 0100H

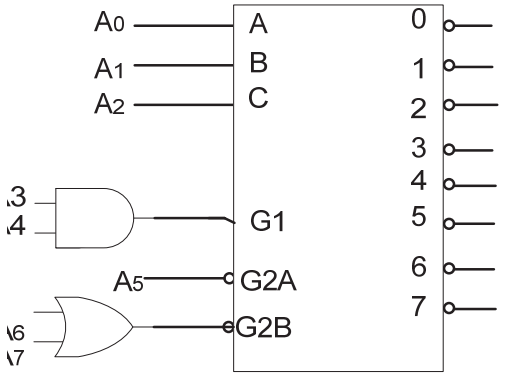
18. Aşağıdaki resimde iki bellek modülün adresli kod çözümlenmesi verilmiştir. M2 bellek modülü hangi adresleri kapsıyor?



19. 74LS138 kod çözücüsünde, bileşiğin seçimi ve çıkışın seçimi için pinler nasıl işaretleniyor?

20. 16 - bitli adres veriyolu olan mikroişlemci şu bellek modülleri kullanıyor:
0000h - 3FFFH adres kapsamalı RAM1,
4000H - 7FFFH adres kapsamalı RAM2,
8000H - BFFFH adres kapsamalı RAM3 ve
C000H - FFFFH adres kapsamalı EPROM.
En değerli adres bitleri şu mantıksal seviyelerde ayarlanmışsa A_{15}=0, A_{14}=1, verilen modüllerden hangisi adreslenmiştir?

21. Resimde sekiz arabirim adresin kod çözümlenmesi için 74LS138 adres kod çözücüsü gösterilmiştir. Altıncı çıkış pinin adresi nedir?



22. Mikroişlemci ve dış aygıtlar arasında paralel aktarım için dört düzeni say!

23. Anonslu düzende ve çift tokalaşma düzeninde STROBE sinyalın ne için kullanıldığını açıkla?

24. Belleksel kopyalanmış giriş/çıkışlı adres kod çözümlenmesinde, hangi sinyal belleği dış aygıtlardan ayırıyor?

25. Belleksel kopyalanmış giriş/çıkış sisteminin belleksel izole giriş/çıkış sisteminde kıyasen avantajları ve dezavantajları nedir?

26. $\overline{INTR}$ ve İNTA pinlerini hangi aygıtlar ve ne zaman aktifleştirdiğini açıkla? Bu pinler hangi mantıksal seviyede aktifleştiriliyor?

27. Kesintinin kabul edilmesinden sonra, dış aygıtı veri veriyolu aracılığıyla arabirim adresini mikroişlemciye gönderiyor. Mikroişlemci arabirim adresini ne için kullanacağını açıkla?

28. Kesintilerin maskelenme sürecini açıkla!

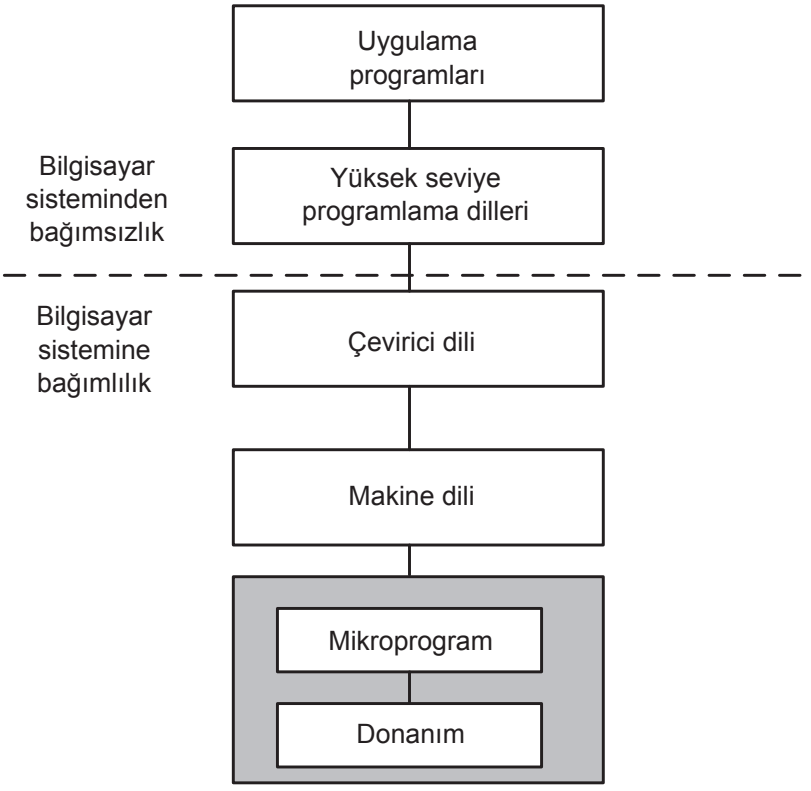
29. Verilerin doğrudan aktarma yaklaşımı ne zaman kullanılıyor?

30. DMA denetimcide HOLD - HLDA ve DREQ - DACK pinlerinin işlevlerini açıkla?

5. Mikroişlemcinin Programlanması

5.1. Programlama Dillerin Ayırımı

Programlama dilleri insan ve bilgisayarlar arasında iletişim sağlıyor. Tüm programlama dilleri üç gruba ayrılıyor: makine dilleri, çevirici (asembler) dili ve yüksek seviye programlama dilleri. Resim 5.1.'de farklı programlama diller türlerinin hiyerarşisi ve onların bilgisayarlarda uygulamaları verilmiştir.



Resim 5.1. Programlama dillerin hiyerarşisi

Çevirici (asembler), makine dili ve yüksek seviye programlama dilleri arasında aracılık yapan programlama dilidir. Çevirici alçak seviye programlama diller grubuna aittir. Çevirici dili giriş - çıkış aygıtlar için program yordamları yazmak için, farklı taşınır aygıtlara yazılım ve yeni yazılımın yazılması için geliştirici programlara uygundur. Yüksek seviye programlama dilleri, BASIC, FORTRAN, C++, PASCAL ve başkaları gibi, uygulama programların yazılması için kullanılıyorlar.

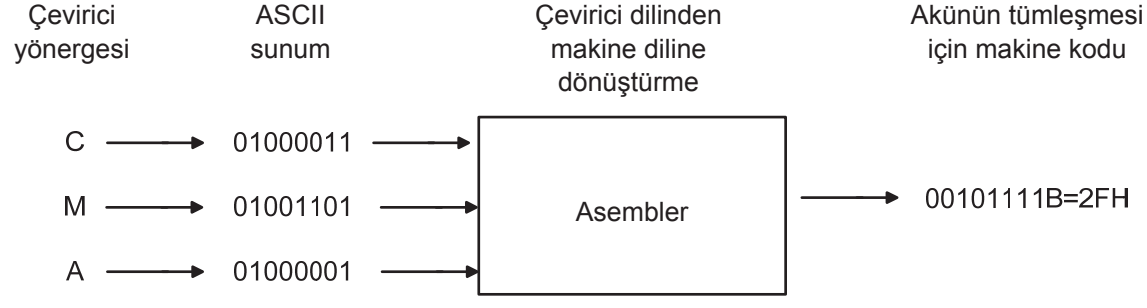
Programcı mikroişlemciye yönergeler vererek erişiyor. Makine dilinde yönergeler, sıfırlar ve birler dizisi olarak ikili şeklinde tanımlanıyor. Mikroişlemci yönergeleri sadece bu şekilde kabul ediyor. Her yönerge çerçevesinde iki ana parça fark ediliyor: işlem kodu ve adres bölümü. İşlem kodu, mikroişlemcinin gerçekleştirmesi gereken işlem türünü belirliyor, adres bölümü ise gereken verilerin alınacak yerlerin ya da verilecek yerlerin adreslerini belirliyor. Aşağıda makine kodunda bir yönerge verilmiştir. Ilk sekiz bit işlem kodunu tanımlıyor, diğerleri ise adres bölümünü tanımlıyor.

01110111000011100001

Makine dilleri, donanım kaynaklarından daha iyi şekilde faydalanmak istendiği zaman ve çalışma hızının artmasını istediğimiz sırada daha uygundur. Yönergelerin ikili şekli insanlar için uygunsuzdur. İnsan, yönergeleri tanımlayan tüm bitler kombinasyonlarını aklına tutamaz. Çeviricide işlem kodları, yönergenin anlamını açıklayan İngilizce sözcüklerin kısaltılmış şekliyle tanımlanıyorlar, adresler ise sembolik adlarla değiştiriliyor. Dolayısıyla, yukarıda verilmiş makine yönergesini sembolik şeklinde şöyle yazılabilir:

Add address

Çevirici ve makine yönergeleri arasında tek anlamlı ilişki vardır. Çevirici dilinde makine koduna dönüştürülmesini, asembler (çevirici) olarak adlandırılan özel program - çevirici aracılığıyla bilgilsayar gerçekleştiriyor. Resim 5.2. akümülatörün içeriğini tümleştiren yönergesini çeviriciden makine koduna dönüştürülmesini gösteriyor.



Resim 5.2. Çevirici dilinden makine diline dönüştürme

Yüksek seviye programlama dillerinin çevirici diline kıyasen olduğu avantajları şunlardır:

- Yönergeler insanların konuşmasına ve düşünme şekline çok yakındır. Yüksek seviye programlama dilleri donanımı yeterince tanımadan başarılı programlamayı sağlıyor. Çevirici dillerde böyle durum yoktur
- Programlama daha az zaman arıyor, programlar görevli kısadır ve yapılan hatalar kolay bulunuyor. Basit programlama zengin program kütüphanesinden koşulludur, program kütüphanesi ise hem basit, ancak kompleksli yönergelerden de oluşuyor ve programcıya bu yönergeleri yazımsal olarak yapmasına gerek kalmıyor.
- Yüksek seviye programlama dillerin en büyük avantajı, onların farklı bilgisayar sistemlerinde fazla değişiklik yapılması gerekmeden uygulanabilmesidir. Diğer taraftan çevirici farklı mikroişlemciler için farklıdır ve farklı sistemlerde uygulanamıyor.

Ancak, çeviricinin yüksek seviye programlama dillerine kıyasen saydiğimiz tüm dezavantajlara rağmen, hala kullanılıyorlar. Çeviricilerin günümüzde de kullanılmasının nedeni, çeviricinin iki en önemli özelliğidir:

- Bellek alanının tasarrufu. Çeviricide yazılan program, aynı programın yüksek seviye programalama dilinde yazılmış programdan daha çok yönergeler içeriyor. Ancak, çeviriciden makine dilinde dönüşüldüğünde çevirici programın hafıza edilmesi için çok az program belleği gerek olduğuna fark edebiliriz. Bu özellik eskiden daha çok önemliymiş, çünkü eski bilgisayar sistemlerin çok düşük kapasiteli sabit bellekleri varmış.
- Çeviricinin kullanılması için ikinci iyi neden programların büyük hızla çalıştırılmalarıdır. Çeviriciden makine diline dönüşturmek çok kolay ve hızlıdır. Programı yazmak için belki fazla zaman harcanabilir, ancak kullanıcı için programın çalıştırma hızı çok önemlidir.
- Çeviricinin belki de en büyük avantajı, onun donanıma doğrudan erişimidir. Çevirici yönergeleriyle mikroişlemcinin içinde bazı yazmaça ya da RAM belleğinde bazı bellek konumuna ulaşıyoruz. Yönergelerin her çalıştırılmasıyla mikroişlemcinin ya da bellek yongaların bazı pinleri tetikleniyor. Çeviriciyi öğrenirken biz aslında bilgisayarın donanımını daha iyi tanıyoruz.

5.2. Çevirici Yönergeyi Oluşturan Parçalar

Çevirici yönergesi 4 alandan oluşuyor:

- Etiket alanı
- Anımsatıcı alanı
- İşlenenler alanı
- Yorum alanı

Bu alanların sıralaması şöyledir:

[ETİKET] ANIMSATICI [İŞLENENLER] [YORUM]

Sıralamadaki orta parantezler demek ki bu alanların var olmasını şart değildir. Bazi yönergeler bu alanları içeriyor, bazıları ise içermiyor.

Etiket programda bir yönergenin ya da sabitin adlandırılması için kullanılıyor. Bu alanın olması şart değildir. 8 - bitli mikroişlemcilerde etiket 1'den 5'e kadar alfanumerik işaretlerden oluşuyor. Etiket olarak yazmaçların adları ve işlem kodları kullanılamaz. Yönergenin adlandırılması için kullanıldığı zaman, etiket anımsatıcıdan iki nokta işaretiyle (:) ve boş yerle ayrılıyor. Etiket alanı dallanma yönergelerinde, alt program yönergelerinde ve döngüçlerde kullanım görüyor. Şöyle ki tüm bu yönergelerin içeriğinde bellek konumun adresini içeriyor. Etiket aslında, atlamasi gereken ya da alt programin başlayacağı bellek konumun adresini değiştiriyor.

Anımsatıcı yönerge işlevinin açıklamasını sembolik şekilde yapan İngilizce sözlerin kısaltmasını tanımlıyor. Anımsatıcı alanı her yönergenin içeriğinde olmalıdır ve bazan yönergenin tek alanıdır. Resim 5.2.'de gösterilen örnekte anımsatıcı CMA'dır. CMA Complement Accumulator sözcüklerin kısaltmasıdır ve anlamı birinci tümleştiricinin hesaplanması, akünün içeriğinin olumsuzluğudur.

Işlenenler (operantlar) verilen yönergenin gerçeklesmesinde yer alan verilerdir. Örneğin, toplamada işlenenler birinci ve ikinci toplanandır, üssüde işlenenler temel ve üsüdür vs. Yönerge türüne göre işlenenler alanı boş olabilir, bir ya da iki işlenen içerebilir. Işlenenler alanı iki işlenen içeriyorsa, o zaman onlar virgul işaretiyle ayrılıyorlar. Böyle durumda birinci işlenen hedef, diğeri ise verinin kaynağıdır. İşlenenler alanında yer alan veriler farklı sayı sistemlerinde ifade edilebilir:

- H ile on altılı sayı işaretleniyor
- Onlu sayi D ile işarteleniyor ya da işaret kullanılımıyor
- B ve Q ile sekizli veriler işaretleniyor
- B ile ikili veriler işaretleniyor

İşlenenler her zaman sabit değildir. İşlenen bazan değişken büyüklük de olabilir. Böyle durumda, işlenen yönergenin içeriğinde değil de bazı genel yazmacın ya da bellek konumunun içeriği olabilir. İşlenenlerin bulunma şekillere adresleme şekilleri denir ve onları daha geç tanıyacağız.

Yorum, çevirici yönergenin sonunda yazılıyor ya da program içinde bağımsız ifadeler olarak yazılıyor. Çevirici yorumları her zaman; (nokta virgül) işaretiyle ayrılıyorlar. Mikroişlemci nokta virgül işaretine gelince devamdaki metni analize etmiyor. Ancak, kullanıcılar ve programcılar için yorumlar çok önemli alanlardır, çünkü onlar programın anlaşılmasına çok yardım ediyorlar.

Aşağıda belli bir yönerge için örnek verilmiştir

```
TEKRARLA:  INC SONUÇ ; sonucun değerini bir için arttır
```

TEKRARLA etikettir ve onunla INC SONUÇ yönergesi adlandırılıyor. Programci için bu yönergenin bulunduğu yerin adresiyle çalışacağına, etiketi kullanması çok daha kolaydır. Büyük bir ihtimalle programin devamındaki metinde programcı tam bu yönergeye INC SONUÇ yönergesine atlaması gereken atlama yönergesi kullanacak. INC anımsatıcıdır ve arttır anlamına gelen, İngilizce increase sözcüğünün kısaltmasıdır. SONUÇ işlenendir. SONUÇ bazı genel yazmacın ya da bellek konumunun sembolik adıdır. Sembolik adların verilmesi özel çevirici komutlarla gerçekleşiyor. Çevirici komutlarla, daha geç, çevirici programın yazılma sürecini incelerken tanıyacağız. Tabii ki, yukarıdaki yönergenin açıklaması, yönergenin sonunda yorum olmasaydı çok daha zor olacaktı.

5.3. Adresleme Şekilleri

Makine diline dönüştürülmüş çevirici yönergelerinin incelemesini yaparsak, yönergenin işlem kodunu ifade etmek için elde edilen diziden az sayıda bitlerin kullanıldığı sonucuna varacağız. Bitlerin büyük kısmı yönergenin tanımlamasından fazla yönergenin gerçekleşmesinde yer alan işlenenlerin tanımlanması için kullanılıyor.

ADD toplama yönergenin analizini yapacağız. ADD yönergesi üç işlenenin tanımlanması gerektiriyor: iki toplanan ve bir toplam. Toplananlar için veri kaynakları olduklarını diyoruz (source), toplam için ise elde edilen sonucun hedef noktası (destination) olduğunu diyoruz. Bu işlem resim 5.3.'te gösterilmiştir.

ADD	Kaynak 1	Kaynak 2	hedef

Resim 5.3. Üç adresli çevirici yönerge

Örneğin, bellek adresi 32 - bitli ise, o zaman işlem kodundan sonra yönerge üç 32 - bitli adres içerecek, yani toplam 96 adres biti içerecek. Adresler işlem kodundan çok daha fazla bitler gerektiriyor.

Genel anlamda, işlenenleri tanımlayan bitler sayısını azaltmak için iki yöntem vardır. İşlenen daha sıkı kullanılırsa, o zaman bu işlenen bellek konumu yerinde bazı genel yazmaçta yerleşebilir. Yazmaçları işlenenleri saklamak için kullanmak çift kazanç getiriyor: işlenenlere daha hızlı erişim ve işlenenleri tanımlamak için daha az bit sayısı gerekiyor. Örneğin, eğer mikroişlemci 32 genel yazmaç içerirse, o zaman tüm yazmaçları kodlayabilmek için beş bit gerekecek (2^5=32). Buna göre, toplama işlemi için işlenenler yazmaçlarda bulunuyorsa, o zaman onların tanımlanması için 15 bit gerekecek, işlenenler bellek konumlarında bulunursa o zaman 96 bit gerekecek.

Ancak, buna rağmen, yazmaçları işlenenleri saklamak için kullanmak her zaman ideal bir çözüm değil. Eğer yazmaçlardaki işlenenlerin önce bellekten aktarılması gerekirse, hatta durum daha kötü olabilir. Böyle durumda hem yazmaçların hem de bellek konumların tanımlanması gerekecek. İyi ki analizler, işlenenlerin en büyük kısmını birçok kez kullanıldıklarını gösteriyor. Bu yüzden yeni mikroişlemci yapılar, belleğe erişim sayısının azalması amacıyla daha büyük sayıda genel yazmaçların olmasını öngörüyor.

İşlenenlerin tanımlanması için gereken bitler sayısının azalması için kullanılan ikinci yöntem fazla işlenenin tek şekilde görünüm yöntemidir. Örneğin, aynı bir yönergede aynı bir işlenen hem veri kaynağı olacak hem de elde edilen sonucun hedef noktası olacak. Böylece ADD yönergesinin üç adres içereceği yerine, iki adres içerecek. Üç - adresli yönerge durumunda toplama süreci aşağıdaki ilişkiyle gösteriliyor:

HEDEF=KAYNAK 1+KAYNAK 2

İki - adresli toplama yönerge durumunda şu ilişki geçerli olacak:

YAZMAÇ 2= YAZMAÇ2 + KAYNAK 1

İki - adresli yönerge durumunda yazmaç 2'nin aynı zamanda hem hedef hem kaynak 2'yi tanımladığını görüyoruz. İki - adresli yönergenin dezavantajı, yönergenin çalıştırılmasından sonra yazmaç 2 işlenenin eski içeriğinin kaybolmasıdır. Yazmaç 2 işlenenin eski içeriğini korumak istersek, onu yönergenin çalıştırılmasından önce başka bazı genel yazmaçta yerleştirmemiz gerekiyor.

Daha eski nesillerden işlemciler sadece bir, akümülatör adıyla tanımlanan genel yazmaç kullanıyorlar. Örneğin, ADD yönergesinle akümülatörün içeriğine her zaman bazı bellek konumun içeriği ekleniyor. Bu durumda, sadece bir işlenen, yani sadece işlenenin bulunduğu bellek konumun tanımlanması gerekiyor. İşlenenlerin sayısını üçten bire indirmek büyük ilerleme olarak gözüküyor, maalesef bunu sadece basit hesaplamalarda, büyük sayıda ara sonuçların olmadığı hesaplamalarda kullanabiliriz.

Yukarıda belirtilenlerden, işlenenlerin birkaç şekilde tanımlayabileceğimizi sonucuna varabiliriz:

- yönergenin içinde;
- mikroişlemcinin genel yazmaçlarında birinde;
- bazı bellek konumunda;
- ya da bazı giriş aygıt aracılığıyla girilebilir;

Aranan işlenen nasıl bulunağıcına bağlı olarak birkaç adresleme şekilleri var. Bunlardan daha önemlileri şunlardır: yazmaçların adreslenmesi, dolaysız (direkt) adresleme, doğrudan adresleme ve yazmaçlı dolaylı (inderekt) adresleme.

Yazmaçları adresleme şeklinde işlenenler mikroişlemcinin genel yazmaçlarında bulunuyor. Resim 5.4.'te yazmaçların adreslenmesini kullanan yönerge verilmiştir.

anımsatıcı	yazmaç 1	yazmaç 2

Resim 5.4. Yazmaçların adresleme formatı

Yönerge iki - adreslidir. İşlem kodundan sonraki işlenen (yazmaç 1) veri kaynağı ya da elde edilen sonucun hedef noktası olabilir, yönergedeki son işlenen (yazmaç 2) ise sadece veri kaynağıdır. Bu adresleme şeklin avantajı belleğe erişim sayısının azalmasıdır. Gereken işlenenler mikroişlemcide bulunuyor ve onları bellekte aramak gerekmiyor. Bu şekilde bellekte arama ve verilerin mikroişlemciye aktarılması için harcanan zaman kurtarılıyor, bu da yönergelerin daha hızlı çalıştırılması demektir. Ayrıca, işlenenler bellek konumları yerinde genel yazmaçlarda bulundukları zaman yönergenin kodu daha kısadır. Yazmaçların adresleme şeklin dezavantajı, tüm işlenenleri yazmaçlarda yerleştirmek istersek daha büyük sayıda yazmaçlardan gerek olmasıdır.

Doğrudan adresleme şeklini kullanan yönergeler biraz daha uzun yönergelerdir. Bu adresleme şeklinde işlenen yönergenin içinde bulunuyor. Doğrudan terimin anlamı verinin hemen yönerge kodundan sonra olduğu demektir. Resim 5.5.'te doğrudan adresleme kullanan genel yönerge verilmiştir.

anımsatıcı	yazmaç 1	veri

Resim 5.5. Doğrudan adresleme formatı

Yönergenin sonunda veri bulunduğunu görüyoruz. Bu veri herhangi bir sayı sisteminden sayı olabilir: ikili, onlu ya da on altılı. Doğrudan adresleme şekli özellikle sabitlerle çalışmak için uygundur. Örneğin, bir genel yazmacın sabit değerle doldurulması gerektiğinde ya da bir genel yazmacın içeriğine bir sayının eklenmesi gerektiğinde uygulanıyor. Doğrudan adresleme değişkenle çalışma olanağı vermiyor.

Değişkenlerle çalışmak için **dolaylı adresleme** kullanılıyor. Dolaylı adresleme şeklinde yönerge işlenenin bulunduğu bellek yerinin adresini içeriyor. Resim 5.6.'da bu adreslem şeklinin kullanan genel yönerge verilmiştir.

anımsatıcı	yazmaç 1	bellek yerin adresi

Resim 5.6. Dolaylı adresleme formatı

Dolaylı adresleme şekli sıkça kullanılıyor, çünkü belleğin tüm yerlerine erişim sağlıyor. Diğer taraftan yazmaç adreslemesi çok az sayıda yazmaçlarla erişimimiz var. Değişkenlerle çalışma olanağı, bellek yerin adresi aynı kalıp içeriği bellek ye-

rine eriştiğimizde her zaman değişebilmesinden oluşuyor. Dolaylı adreslemenin kötü tarafı yönerge formatın uzunluğudur. Çünkü yönergeler, içinde yazmaçların kodlarından ve verilerden daha uzun kodu olan bellek adreslerini içeriyor. Dolaylı adreslemenin diğer dezavantajı yönergelerin uzun sürmesidir. Örneğin, yazmaçlar adreslemede yönerge bir bayttan oluşuyorsa, o zaman dolaylı adreslemede aynı yönerge en az üç bayttan oluşacak. O, demek ki dolaylı adreslemeli yönergenin aktarımı üç kez daha uzun sürecek. Yönergenin tüm baytları aktarıldıktan sonra, yönergenin içeridiği adresle yeniden belleğe ulaşmak gerekiyor ve aranan işlenenenin bulunması gerekiyor.

Yazmaçlı dolaylı adresleme istenilen işlenenenin bulunması için bellek yerinin adresi kullanıyor. Ancak, bu adresleme şeklinde bellek yerin adresi yönergenin içinde bulunmuyor, adres mirkoişlemcide bazı yazmaçta bulunuyor. Adresler korumak için kullanılan yazmaçlara gösterge (pointer) denir. Bu şekilde bellekte tüm konumlara ulaşma olanağı kalacak ve aynı zamanda yönergelerin uzunluğu ve süresi azalacak.

5.4. Genel Mikroişlemcinin Yönergeler Kümesi

Herhangi mikroişlemci sözkonusu olursa, mikroişlemcilerin programlama yönergeleri birbirine benzerdir. Tabii ki aralarında belirli farklar da vardır. Örneğin, iki sayının toplama süreci tüm mikroişlemci türlerinde aynıdır, o farkla ki bazıları iki - adresli toplama yönergeleri, bazıları ise üç - adresli toplama yönergeleri kullanıyor. Tüm yönergeler birkaç gruba ayrılabilir: veri aktarma yönergeleri, aritmetik yönergeler, mantıksal yönergeler, dallanma yönergeleri, alt programların geçekleşmesi için yönergeler ve yığıt bellekle çalışmak için yönergeler.

5.4.1. Veri Aktarma Yönergeleri

Verilerin aktarılması için yönergelerde aranan veriyi kaynağından alıyoruz ve hedef noktasında kopyalıyoruz. Kaynaklar ya da hedefler mikroişlemcide yazmaçlar ya da bellekten konumlar olabilir. Buna göre dört farklı veri aktarma şekli olabilir: yazmaçtan yazmaca, yazmaçtan bellek yerine ve bellek yerinden yazmaca. Dördüncü aktarma şekil için bir bellek yerinden başka bellek yerine veri aktarımı için yönergeler yoktur. Verinin doğrudan da verilebilmesini unutmayalım.

Aktarım için en çok kullanılan anımsatıcı, İngilizce Move sözcüğünün kısaltması olan MOV anımsatıcısıdır. MOV anımsatıcısı yer değiştir anlamına geliyor,

halbuki verilerin aktarılmasını açıklamak için daha uygun sözcük kopyala (copy) sözcüğü olabilir. Verinin yerini değiştir deyince, verinin alındığı kaynağın boşandığını, hedefin ise o veriyle doldurulduğunu düşünüyoruz. Ancak bu süreç, düşündüğümüz gibi gerçekleşmiyor. MOV yönergesinden sonra, kaynaktaki veri hedefe kopyalanıyor ve o şekilde kaynağın ve hedefin aynı içerikleri var. MOV yönergesinden sonra hedefte kaynaktaki verinin kopyası olacak. Bu sürecin kötü tarafı, MOV yönergesinden önce, hedefte bulunan verinin kaybolmasıdır. Farklı mikroişlemcilerde kaynağın ve hedefin çevirici yönergelerinde farklı yerleri var. Örneğin, İntel şirketinin mikroişlemcilerinde, aktarma yönergelerinde anımsatıcıdan sonra hedef yazılıyor, ondan sonra ise veri kaynağı yazılıyor. Bu yönerge şekli aşağıdaki yönergede gösterilmiştir.

MOV hedef, kaynak.

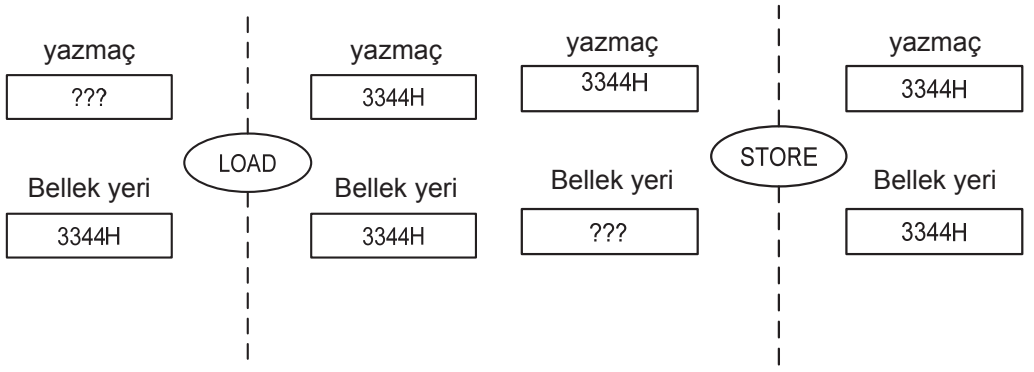
Hedef ve kaynak virgül işaretiyle birbirinden ayrılmıştır. Aşağıda aktarma için çevirici yönergeleri ve onların yorumları verilmiştir.

```
MOV yazmaç, adres       ;Veriyi adresi yazılmış bellek
                        ;konumundan yazmaçta kopyalıyoruz.

MOV adres, yazmaç       ;Yazmaçtan veriyi adresi verilmiş
                        ;bellek konumuna kopyalıyoruz.

MOV yazmaç              ;65H;65H verisi yazmaçta yazılıyor.
```

Büyük sayıda mikroişlemcilerin bellekten ya da belleğe veri aktarmak için özel yönergeleri vardır:

- LOAD - Bu İngilizce sözün anlamı doldur demektir ve adresi verilmiş bellek konumdan akümülatöre veri aktarmak için kullanılıyor.
- STORE - Bu İngilizce söz depola anlamına geliyor ve akümülatörden adresi verilmiş bellek yerine veri aktarmak için kullanılıyor.

Yukarıdaki iki yönergenin kullanımı grafiksel olarak resim 5.7.'de gösterilmiştir. İki yönergenin, LOAD ya da STORE, çalıştırılmasından sonra yazmacın ve bellek yerinin içerikleri aynı olduğunu görüyoruz, ancak 3344H verisinin kaynağı farklıdır. LOAD yönergesinde veri kaynağı bellek yeridir, STORE yönergesinde ise kaynak yazmaçtır. Ayrıca, LOAD yönergesinde yazmaçın eski içeriği kayboluyor, STORE yönergesinde ise bellek yerinin içeriği kayboluyor.



Resim 5.7. LOAD ve STORE yönergelerin çalıştırılması

Verilerin aktarımı için yönergeler grubuna, giriş çıkış aygıtlarla çalışma yönergeleri de aittir. Mikroişlemcilerin çoğunda giriş aygıtından mikroişlemcide yazmaca doğru veri aktarımı için bir yönerge ve yazmaçtan çıkış aygıtına doğru veri aktarımı için bir yönerge vardır. Giriş yönergesi için anımsatıcı İN'dir, çıkış yönergesi için ise anımsatıcı OUT'tur. Aşağıda giriş çıkış aygıtlardan veri aktarımı için iki yönerge verilmiştir.

```
IN yazmaç, arabirim adresi          ;Giriş aygıtından yazmaca aktarma
OUT arabirim adresi, yazmaç         ; Yazmaçtan çıkış aygıtına aktarma.
```

Arabirim adresi dış aygıtına tanımak için ikili kod tanımlıyor. Her giriş ya da çıkış aygıtın kendi arabirim adresi var ve arabirim adresi yardımıyla mikroişlemci hangi aygıtla iletişim kurduğunu anlıyor. Konumların bellek adresleri olduğu gibi, giriş çıkış aygıtların arabirim adresleri vardır. Ayrıca, verilen İN ve OUT yönergelerinden, MOV yönergesinde olduğu gibi, verinin hedefi yönergenin anımsatıcısından sonra yazıldığını, verinin kaynağı ise virgül işaretinden sonra yazıldığını görebiliyoruz.

Sonunda, aktarımı yönergelerinde, aktarılan verilerin büyüklüğüne dikkat edilmesi gerektiğini söyleyelim. Örneğin, 16 bitli veri 8 bitli yazmaça aktaramayız, çünkü veri için yeterince yer olmayacak. Aynı şekilde 8 bitli verinin 16 - bitli yazmaça aktarılması tavsiye edilmez, çünkü yazmaçın tüm kapasitesi kullanılmış olmayacak.

5.4.2. Aritmetik Yönergeler

En çok kullanılan aritmetik yönergeleri, toplama va çıkarma yönergeleridir. Toplamak için kullanılan yönerge ADD yönergesidir, çıkarma işlemi için ise yönerge, SUB yönergesidir. SUB aslında İngilizce Substract sözcüğün kısaltmasıdır ve çıkarma anlamına geliyor. Bu yönergelerin birkaç türevleri vardır.

Bazılarında mikroişlemcilerde toplama yönergesi iki işleneni içeriyor, bazı mikroişlemcilerde ise üç işlenen içeriyor. Bunu aşağıda verilen iki yönerge ile görebiliriz:

```
ADD hedef, kaynak              ;Kaynak hedefe ekleniyor.

ADD kaynak1, kaynak 2, hedef   ;Hedef=kaynak1+kaynak2
```

Toplama yönergeleri aktarma bayrağının (Carry) dikkate alınıp alınmayacağına bağlı olarak aralarında fark olabilir. Aşağıda iki yönerge verilmiştir:

```
ADD kaynak, hedef              ;Hedefe sadece kaynak ekleniyor.
ADC kaynak hedef               ;(Add Carry) Hedefe kaynak
                               ;ve aktarma bayrağının değeri
                               ;ekleniyor.
```

Toplama yönergesi için geçerli olan şeyler, çıkarma yönergesi için de geçerlidir.

Toplama ve çıkarmadan sonra çarpma ve bölme yönergeleri geliyor. Çarpma için kullanılan yönerge MUL yönergesidir. MUL, İngilizce çarpma, multiply sözcüğünün kısaltmasıdır. Bölme için DIV yönergesi kullanılıyor ve İngilizce bölme - divide sözcüğün kısaltmasıdır. Çarpma işlemi sırasında çarpımın çarpanlardan iki misli daha uzun olduğunu, bölme işleminde ise bölüm için yer dışında, kalan için de yer ayrılması gerektiğini not edelim. Örneğin, iki 8 bitli sayı çarparsak, o zaman çarpım 16 bitli olabilir. Aşağıda çarpma için iki yönerge verilmiştir.

```
MUL kaynak1, kaynak 2, hedef   ;hedef =kaynak1 · kaynak 2

MUL kaynak                     ;Çalışma yazmacın içeriği
                               ;kaynakla çarpılıyor ve çarpım yeniden
                               ;çalışma yazmacında yerleşiyor.
```

Üç işlenenli yönergelerde, bölen, bölünen ve bölüm herhangi yazmaçta ya da bellek konumunda bulunabilir. Bir işlenenli yönergede bölünen ve bölüm her zaman çalışma yazmacında bulunuyorlar, bölen ise herhangi yazmaçta ya da bellek yerinde bulunabilir. Tabii ki bölen doğrudan, sabit olarak da verilmiş olabilir. Ön işaretli sayılarla çarpma ve bölme işlemleri için özel yönergelerin var olduğunu analım ve onların anımsatıcıları IMUL ve IDIV'tir.

Bir (1) için arttırmak ya da azalmak için yönergeler çok pratik kullanımı görüyorlar. Bu yönergeler bir büyüklüğün sayılması gerektiği zaman kullanılıyorlar.

Bir için artma yönerge INC (İncrease) yönergesidir, bir için azalma için ise DEC (Decrease) yönergesi kullanılıyor. Bu iki yönergenin birer işlenenleri vardır, yazmaç ya da bellek konumu.

5.4.3. Mantıksal Yönergeler

En önemli mantıksal yönergeler: AND (mantıksal çarpma), OR (mantıksal toplama) ve NOT (olumsuzluk ya da birinci tümleyici) yönergeleridir. Bazı mikroişlemciler XOR (dışlamalı ya da), NOR (OYADA devresi) ve NAND (OVE devresi) yönergelerini de kullanıyor. Mantıksal fonksiyonların anlamlarını birinci konuda, Temel Mantıksal Devreler konusunda tanıdık. Aritmetik yönergelere benzer olarak, mantıksal yönergelerin de ikişer işlenen kaynağı ve elde edilen sonuç için hedef noktaları vardır. Bazı mikroişlemcilerde kaynaklardan biri hedef olarak da kullanılıyor. Aşağıda birkaç mantıksal yönerge ve onların yorumları verilmiştir.

```
AND hedef,kaynak          ;Kaynağın ve hedefin mantıksal çarpımı
                          ;ve sonuç hedefte yazılıyor.

NOT hedef                 ;Hedefin birinci tümleyicisiyle
                          ; değiştirilmesi.

OR kaynak1, kaynak 2, hedef   ;İki kaynağın mantıksal toplaması ve
                              ;sonucun hedefte yazılması.
```

Kaynakların ve hedefin genel yazmaçlarda ya da bellek konumlarında bulunabildiklerini ve kaynaklardan birinin doğrudan ikili, onlu ya da onaltılı sayı siseminde sayı olarak verilebileceğini hatırlayalım.

Çoğu kez mantıksal yönergelerde işlenenlerden biri maske tanımlıyor. Maske önceden belirlenmiş sıfırlardan ve birlerden oluşan kombinasyon tanımlıyor. Bitlerin sıralaması nasıl olacağı maskenin amacına bağlıdır. Örneğin, AND mantıksal yönergesi, 16, 32 ya da 64 - bitli veriden bir baytın ayrılması gerektiği zaman kullanılıyor. Aşağıda böyle bir örnek verilmiştir, 32 - bitli verilden, sağdan ikinci baytın ayrılması gerekiyor. Bunu yapmak için maske kullanılacak ve bu maske kaybolması gereken baytların yerlerinde sıfırlardan oluşacak, sağdan ikinci baytın yerine ise sadece birler yerleşecek.

Önek 5.1:

```
        00110010 00011100 10111001 11100010 --- 32 - bitli veri
AND     00000000 00000000 11111111 00000000 --- maske
        ---------------------------------------------------
        00000000 00000000 10111001 00000000 --- ayrılan bayt
```

OR mantıksal yönergesi veriden bir bitin test edilmesi gerektiği zaman kullanılıyor. Maskede o bitin pozisyonunda bir olacak, tüm diğer pozisyonlar mantıksal sıfıra ayarlanacak. Mantıksal toplamanın gerçekleşmesinden sonra tüm pozisyonlarda sonuç bir elde etmezsek, demek ki incelenen bit mantıksal sıfırmış. Aşağıdaki örnekte sağdan birinci bit test ediliyor.

Örnek 5.2:

```
        xxxxxxxx xxxxxxxx --- incelenen veri
OR      11111111 11111110 --- maske
-------------------------------------------
        11111111 1111111x --- Sonuç
```

5.4.4. Döndürme ve Kaydırma Yönergeleri

Döndürme va kaydırma yönergeleri çok faydalıdırlar ve birkaç türevlerde vardır. Şimdiye kadar incelediğimiz yönergelerden farklı olarak, döndürme ve kaydırma yönergelerin sadece bir işlenenleri vardır, yani veri kaynaktan alınıyor, işleniyor ve aynı yere geri veriliyor. Bu kaynak ve hedefin aynı olduğu anlamına geliyor.

Döndürme sürecini örnekle açıklayacağız. 16 - bitli 10000000 verisini sola bir yer için döndüreceğiz, ondan sonra da bir yer için sağa döndüreceğiz. Sola doğru döndürme sırasında verinin tüm bitleri bir yer için sola taşınıyor, sıfır bit birinci bitin yerine geliyor, birinci bir ikincinin yerine,…, altıncı bit yedinci bitin yerine. Yedinci bit sola gidemiyor, veriden çıkıyor, döndürülecek ve sıfır bitin yerine gelecek. Sağa doğru döndürme sırasında tüm bitler bir yer için sağa taşınacak, yedinci bit altıncı bit yerine, altıncı bit beşincinin yerine,…, birinci bit sıfır bitin yerine gelecek. Sıfır bit veriden çıkacak, sağdan sola dönecek ve yedinci bitin yerine yerleşecek.

```
10000000 --- veri
-------------------------------
00000001 --- sola döndürme
01000000 --- sağa döndürme
```

Bitlerin döndürülmesi birkaç kez yapılabilir, döndürme sayısı da yönergenin sonunda yazarak vurgulanıyor. Aşağıda birkaç döndürme yönergeleri gösterilmiştir.

```
ROL hedef, 4            ;Hedefteki bitler dört yer için
                        ;sola döndürülüyor.

ROR hedef, 3            ;Hedefteki bitler üç yer için
                        ;sağa döndürülüyor

ROL kaynak, hedef, 4    ;Kaynaktaki bitler sola doğru dört kez döndürülüyor
                        ;ve sonuç hedefte yerleşiyor.
```

Kaydırma yönergeleri (shift) döndürme yönergelerine çok benzerdir. Kaydırma yönergelerinde verinin bitleri sola ya da sağa doğru birkaç yer için kayıyor (taşınıyor), ancak veriden çıkan bitler yok oluyor, verinin diğer tarafına geçmiyor. Boşalan yerler sıfırlarla dolduruluyor. Kaydırma yönergesi aşağıdaki örnekte gösterilmiştir. Bitler üç yer için sola ya da sağa kaydırılmıştır.

1110001010011101 ----- veri

0001110001010011 ----- üç yer için sağa kaydırmak
0001010011101000 ----- üç yer için sola kaydırmak

Kaydırma yönergeleri için anımsatıcı sola kaydırmak için SHL (Shift Left) ve sağa kaydırmak için SHR (Shift Right)'dir. Devamda birkaç örnek verilmiştir.

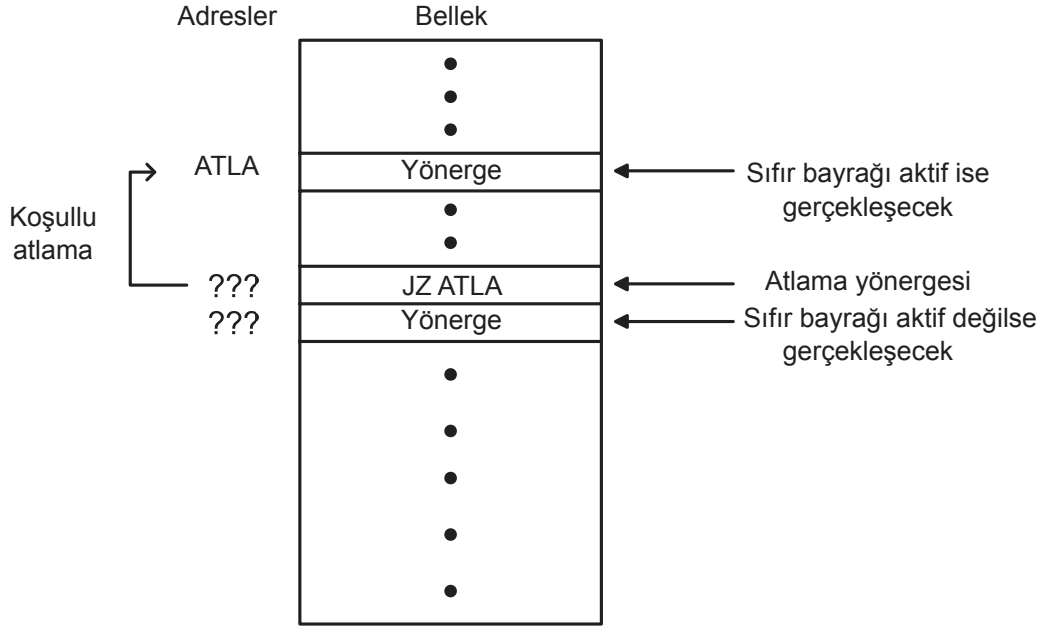
```
SHL hedef, 5              ;Hedefteki bitlerin beş yer için
                          ;sola kaydırmak.

SHR hedef,4               ;Hedefteki bitlerin dört yer için
                          ;sağa kaydırmak.

SHR kaynak, hedef, 2      ;Kaynaktan bitler iki kez sağa kaydırılıyor
                          ;ve sonuç hedefte yerleştiriliyor.
```

5.4.5. Atlama ve Dallanma Yönergeleri

Programın çalıştırılması sırasında çok kez elde edilen sonucun kontrol ediltmesi, test edilmesi gereği ortaya çıkıyor. Elde edilen sonucun değerine bağlı olarak sıradaki yönergenin ya da başka yönergenin çalıştırılması gerçekleşiyor. Hatırlayalım, gerçekleşen aritmetik mantık yönergelerin sonuçları durum yazmaçtaki bayrakların durumuna doğrudan etkiliyor. Bayrakların anlamlarını üçüncü bölümde, Aritmetik mantık birimi bölümünde tanıdık. Yönergenin çalıştırılması bayrakların durumuna bağlı olduğundan dolayı, bu yönergeler koşullu yönergeler olarak da biliniyor. Aynı zamanda programın akışı bu yönergelelere bağlı olduğu için dallanma ve atlama yönergeleri, kontrol yönergeler adıyla da tanımlanıyor. Atlama yönergeleri çok sık kullanılıyor. Atlama yönergelerin anımsatıcısı JMP'dir. JMP İngilizce atlamak anlamına gelen Jump sözcüğün kısaltmasıdır.

Koşullu atlamalar dışında koşulsuz atlamaların da olabileceğini vurgulayalım. Koşulsuz atlamaların gerçekleşmesi bayrakların durumuna bağlı değildir. Atlama yönereglerin içeriğinde hiçbir işlenme bulunmuyor, bu yönergeler atlasması gereken bellek konumun adresini içeriyorlar. Bazan adres sayı olarak değil, etiket olarak ve-

riliyor. Resim 5.8.'de Koşullu atlama JZ (Jump Zero) yönergesinin çalıştırılması gösterilmiştir. Adres olarak etiket kullanılmıştır, atlanması gereken yerin ATLA sembolik isimli adresi var. Atlama sıfır bayrağın aktifleştirilmesiyle gerçekleşecek, yani önceki çalıştırılan yönergenin sonucu sıfıra eşit olursa, atlama gerçekleşecek. Bu koşul yerine gelmezse, o zaman atlama yönergesinde verilen adrese atlanmayacak, program kendi akışıyla devam edecek ve atlama yönergenin altında gelen sıradakı yönergeyi çalıştırılacak.



Resim 5.8. Atlama yönergenin çalıştırılmasının grafiksel görünümü

Aşağıda birkaç atlama yönergesi ve onların yorumları verilmiştir.

```
JMP adres          ; Koşulsuz atlama

JZ adres           ; Atlama sıfır bayrağı aktif ise gerçekleşecek
                   ; (sonuç sıfıra eşittir)

JS adres           ; İşaret bayrağı aktif ise atlama gerçekleşecek
                   ; (negatif sonuç)

JC adres           ; Aktarma bayrağı aktif ise atlama gerçekleşecek

JO adres           ; Aşma bayrağı aktif ise atlama
                   ; gerçekleşecek.
```

Dallanma yönergeleri (branch) atlama yönergelerin benzeridir. Dallanma yönergelerinde iki işlenen arasına kıyaslanma yapılıyor ve elde edilen sonuca bağlı olarak mevcut yönergede atlamanın gerçekleşmesi ya da gerçeklenmemesi oluyor. Kıyaslanan işlenenler birbirine eşit olabilir, birinci işlenen ikinci işlenenden daha büyük ya da daha küçük olabilir. Devamda birkaç dallanma yönergesi verilmiştir.

```
BEQ kaynak1,kaynak2, adres    ;(Branch Equal) Kaynak 1 ve kaynak 2 eşitse
                              ;verilen adrese atlanıyor.

BEQZ kaynak, adres            ;(Branch Equal Zero) Kaynak sıfıra eşitse
                              ;verilen adrese atlanıyor.

BGT kaynak1, kaynak2, adres   ;(Branch Greater Than) Kaynak1 kaynak 2'den
                              ;daha büyükse verilen adrese atlanıyor.
```

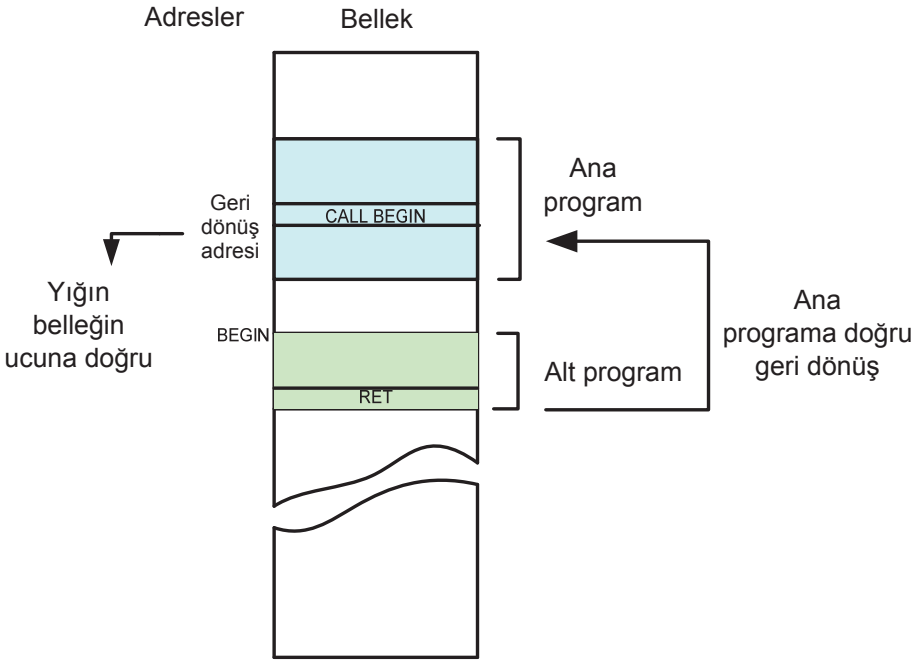
5.4.6. Alt Programlarla Çalışma Yönergeleri

Alt program ana programdan bağımsız olarak çalışıyor ve belleğin herhangi bir yerinde yerleşebilir. Alt program bir değişken büyüklüğün sinüs, kare ya da logaritma değerinin hesaplanması için program ya da belli bir dış aygıttan veri girmek ya da çıkarmak için program olabilir. Ana program alt program içerdiği zaman, alt programın tamamlanmasına kadar ana programda kesinti meydana geliyor. Bu durumda, program sayacında alt programın birinci yönergenin adresi yerleşiyor ve alt programda son yönergenin çalıştırılmasına kadar program sayacının değeri bir için artıyor. Şu soru ortaya çıkıyor: ana programına nasıl geri dönülüyor? Bu amaçla, alt programa atlamadan önce, ana programda sıradaki yönergenin adresin hafıza edilmesi gerekiyor. Program sayacı bunun için kullanılamaz, çünkü sayaç alt programla meşgüldür. Ana programın sıradaki baytın adresi yığın belleğin (tabağın) ucunda yazılıyor. Mikroişlemci alt programın çalıştırılması tamamlanınca, yığıtın ucundaki adresi program sayacına taşıyor ve bu şekilde ana programa geri dönülüyor.

Sıkça alt programın içinde başka alt program olabilir. Böyle durumda, birinci alt programdan ana programa dönüş, ana programın sıradaki baytın adresini yığıtın ikinci yerinde (yukardan aşağıya bakarken) yerleştirmekle gerçekleşiyor, ikinci alt programdan birinci alt programa dönüş için ise yığıtın en üst bellek konumu kullanılacak.

Kaç alt programımız varsa, yığıtın o kadar bellek konumları meşgul olacak.

Resim 5.9'da alt programın çağrılması durumunda geri dönüş adresini koruma süreci ve alt programdan ana programa dönüş gösterilmiştir.



Resim 5.9. Alt programın çalıştırma süreci

Alt programlarla çalışmak için iki yönerge vardır. Bir yönerge altoprogamı çağırmak içindir ve anlamı çağır olan Call anımsatıcısı var. Çağırma yönergesi sadece alt programın başladığı bellek yerinin adresini içeriyor (Call adresi). İkinci yönerge alt programdan ana programa dönmek içindir ve bu yönergenin İngilizce geri dönmek anlamına gelen Return sözcüğün kısaltması olan RET anımsatıcısı var. Call meydana gelince program sayacının değeri otomatik olarak yığıtın ucuna götürülüyor, RET yönergesi meydana gelince geri dönüyor. Bu işlemi programcı yapmıyor.

5.5. Programların Yazılması

Çevirici dilde yazılmış programa kaynak programı denir. Önce kaynak programının bellekte girilmesi gerekiyor (bazı dış aygıt aracılığıyla, genelde klavyeyle) ve bazı dosyada belli bir adla hafıza edilmesi gerekiyor. Programın (yazılması) editör olarak adlandırılıan özel program aracılığıyla yapılıyor.

Kaynak programını dış aygıtlardan birine yazdırdıktan sonra, çeviriciyi çağırıyoruz ve programın çeviri yapmasını emrediyoruz. Çeviri çevirici dilinden makine diline yapılıyor. Çevirici, çeviriye başlıyor ve hiçbir hata yoksa, adı nesne - program olan program yaratıyor ve onu yine ayrı bir dosyada yazdırıyor.

Nesne - program makine dilinde programdır, ancak bu program da çalıştırılmak için hazır değildir. Programın kullanışlı olması için, bellekte ayrı bir yerde yerleşmesi gerekiyor ve olası başka programlarla bağlanabilir. Bu işlemi bağlayıcı (linker) olarak adlandırılan özel program yapıyor. Bağlanma sırasında bir sorun (örneğin, gereken modül bulunmuyor) çıkmazsa, bağlayıcı çalıştırılabilir program yaratıyor. Çevirici programda hataları gidermek için hata ayıklayıcı (debager) denilen özel bir program kullanılıyor.

Program yüksek seviye programlama dillerden birinde yazılmışsa, o zaman onun makine diline çevirisı yapılması için adı derleyici olan özel çevirici - program kullanılıyor. Editör, çevirici, bağlayıcı, hata ayıklayıcısı ve derleyici alet programları tanımlıyor. Programcının tüm bu yardımcı programlardan herkesin ne zaman kullanması gerektiğini bilmelidir, yani alet programları hangi komutlarla kullanılması gerektiğini bilmelidir.

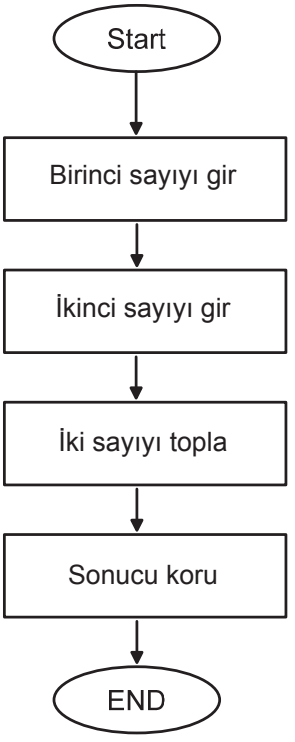
Programın yapısına göre programlar birkaç gruba ayrılabilir: doğrusal programlar, dallanmalı programlar, döngülü programlar, alt programlı programlar ve kompleksli programlar. Devamdaki metinde tüm bu yapılarla tanışacağız.

5.5.1. Doğrusal Programlar

Doğrusal programlar aktarma, aritmetik, mantısal yönergelerden ve döndürme ile kaydırma yönergelerden oluşuyorlar. Yönergeler birbiri üstüne sıralanıktır ve o sıralamada çalıştırılıyorlar. Duğrusal programlarda bir yönergenin atlatırılması ya da alt programın çalıştırılması için programın kesilmesi olanaksızdır. Ayrıca programda geri dönmek de yapılamaz. Demek ki, doğrusal programın çalıştırılması başladıktan sonra, programın sonu gelene kadar sürekli aşağıya doğru gidiyoruz. Program sayacının değeri doğrusal şekilde artıyor ve sıradaki yönerge her zaman mevcut yönergenin altında yer alıyor.

Doğrusal program örneği olarak iki sayının toplamasını yapan programı açıklayacağız. İki toplanan bellekte bulunuyor ve onların mikroişlemciye, yani mikroişlemcinin iki yazmacına aktarılmaları gerekiyor. Aynı zamanda sonuç da belleğe aktarılarak korunuyor.

Programı daha kolay anlamak için programın blok diyagramı verilmiştir. Blok diyagramda programın yaratılışı sırasında tüm adımlar açıklanmıştır.



Resim 5.10. Doğrusal programın blok diyagramı

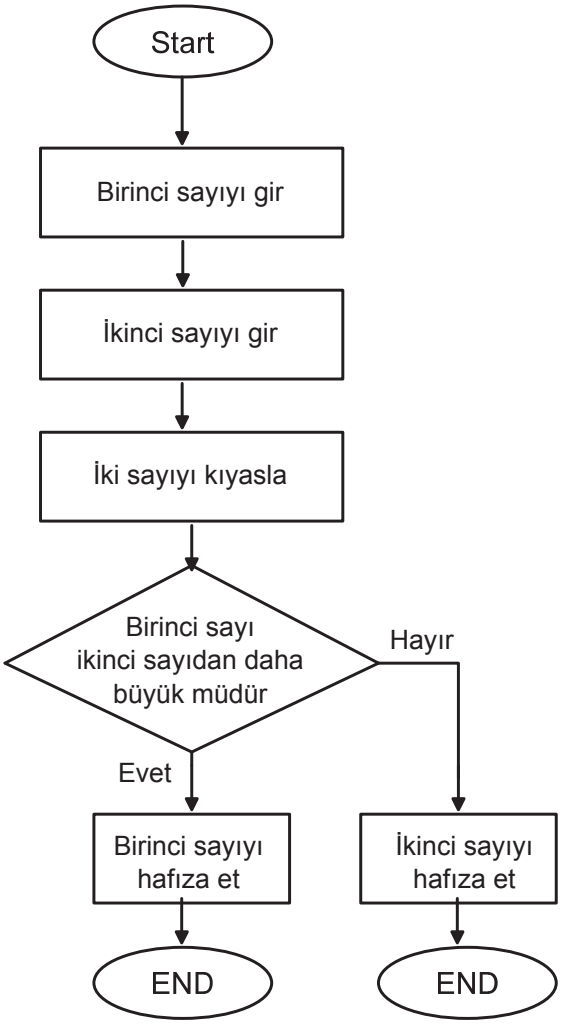
```
MOV R1, adres1      ;Adresi verilmiş konumun değeri R1 yazmacında
                    ;kopyalanıyor.

MOV R2, adres2      ;Adresi verilmiş konumun değeri R2 yazmacında
                    ;kopyalanıyor.

ADD R1,R2           ;R2 yazmacın değeri R1 yazmacın değerine
                    ;ekleniyor.

MOV adres3, R1      ;Toplama sonucu R3 adresli bellek
                    ;konumunda yazdırılıyor.
```

5.5.2. Dallanmalı Programlar

Dallanmalı programlar yapısında atlama (Jump) yönergeleri ve dallanma (Branch) yönergeleri içeriyor. Hatırlayalım, atlama ve dallanma yönergelerine kontrol yönergeleri deniyor, çünkü onlar programın akışını değiştirebiliyor, yani sıradaki yönerge her zaman mevcut yönergenin altında bulunmuyor. Programın hangi yön-

de gerçekleşeceği yönergede verilen koşulun yerine getirilip getirilmemesine bağlıdır. Koşul, alternatif adıyla da biliniyor ve blok diyagramda romb (eşkenar dörtgen) ile işaretleniyor.

Örnek olarak, iki pozitif sayının kıyaslamasını yapan ve daha büyük sayıyı hafıza eden programı açıklayacağız.



Resim 5.11. Dallanmalı programın blok diyagramı

İki sayını kıyaslanması çıkarma işlemi kullanılıyor. Çıkarma sırasında aktarma bayrağı (carry) aktiflenirse, o zaman ikinci sayı birinci sayıdan daha büyük demektir, bayrağın aktiflenmemesi birinci sayının ikinci sayıdan daha büyük olduğu demektir.

```
MOV R1, adres1          ;Adresi verilmiş konumun değeri
                        ;R1 yazmacında kopyalanıyor

MOV R2, adres2          ;Adresi verilmiş konumun değeri
                        ;R2 yazmacında kopyalanıyor
```

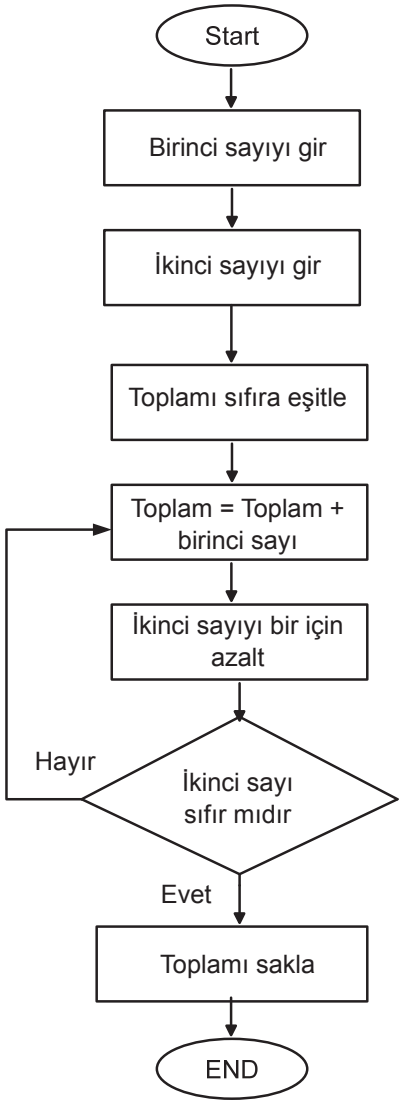
```
SUB R1, R2              ;R1 yazmacın değerinde R2 yazmaçın
                        ;değeri çıkarılıyor.

JC İKİNCİ               ;Taşıma (Carry) bayrağı aktif ise
                        ;İKİNCİ adresi olan konuma atlanıyor.

MOV adres3, R1          ;Taşıma bayrağı aktif değilse
                        ;bellek konumuna R1 yazmaçın
                        ;değeri taşınıyor.

JMP SON                 ;R2 yazmaçın bellek konumunda
                        ;yazılmasın diye sıradaki
                        ;yönerge atlatılıyor.

İKİNCİ: MOV adres3, R2  ;R2 yazmaçı bellek konumunda
                        ;yazdırılıyor.

SONUÇ:          /
```

Görüldüğü gibi, İKİNCİ ve SON etikelerin kullanılması programcının işini büyük ölçüde kolaylaştırıyor, çünkü etiketlerin kullanımıya atlanılması gereken uygun konumların adreslerinin hafıza edilmesi gerekmiyor.

5.5.3. Döngülü Programlar

Programların yazılması sırasında, çok sıkça bir yönerge grubunun fazla kez çalışıtırılması gereği ortaya çıkıyor. Bazı mikroişlemci türlerinin hatta belli program bölümlerin tekrarlanması için özel yönergeleri vardır. Bu tür yönergeler için LOOP anımsatıcısı kullanılıyor ve bu sözcüğün Türkçe anlamı düğüm ya da döngüdür. Döngüler genelde programda geri dönmek demektir, hem de birkaç kez. Döngüler sayacın kullanılmasını gerektiriyor ve sayaç aslında döngünün kaç kez tekrarlanacağını gösteriyor. Sayaçlar çoğu zaman yazmaçlardır ve onlar ileriye ya da geriye sayabilirler. Örnek olarak iki sayının çarpmasını gerçekleştiren programı açıklayacağız.

Çarpma birinci sayıyı, ikinci sayının değeri olduğu kez toplanması şeklinde gerçekleşiyor. Sayacı aslında ikinci sayı tanımlıyor. Birinci sayının her eklenmesinde, ikinci sayının değeri bir için azalacak. Döngü, ikinci sayının sıfır olmasına kadar tekrarlanıyor. Program için çok önemli olan şey birinci sayının eklemesi başlamadan önce sonucun sıfıra eşitlenmesidir. Bu programın çalıştırılması resim 5.12.'deki blok diyagramda gösterilmiştir.



Resim 5.12. Döngülü programın blok diyagramı

Aşağıda döngülü program verilmiştir. Önceki programlarda olduğu gibi bu programda da sayılar bellekten yazmaçlara aktarılıyor.

```
        MOV R1, adres1    ;Verilen adresteki konumun değeri
                          ;R1 yazmacına kopyalanıyor

        MOV R2, adres2    ;Verilen adresteki konumun değeri
                          ;R2 yazmacına kopyalanıyor.

        SUB R3, R3        ;R3 yazmacını kendikendinden çıkartarak
                          ;sıfırlandırıyoruz.

BACK:   ADD R3, R1        ;R3 yazmacına R1 yazmacını ekliyoruz.
```

```
DEC R2          ;R2 yazmacını bir için azaltıyoruz.

JNZ BACK        ;Önceki yönergenin sonucu sıfır
                ;değilse geri dönüyoruz, bir toplamaya
                ;daha. Sonuç sıfır ise aşağıdaki
                ;yönergeye gidiyoruz.

MOV adres3, R3  ;R3 yazmacın değerini adres3'ün gösterdiği
                ;bellek adresine yazdırıyoruz.
```

Doğrusal programlar, dallanmalı ve döngülü programların bileşimini tanımlatan programlara kompleksli programlar denir.

Sonuçlar:

Çeviricinin programlama dili olarak şu avantajları var: bellek alanın tasarrufu, programların büyük hızla çalıştırılmaları, donanıma doğrudan ulaşım. Dezavantajarı ise: küçük program kütüphanesi, bilgisayar sistem türünden bağlılığı, programlar uzundur ve hatalar daha zor bulunuyor.

Çevirici yönergesi 4 alandan oluşuyor ve bu alanlar şu sıralamayla verilmişler: etiket alanı, anımsatıcı alanı, işlenenler alanı ve yorumlar alanı. Etiket, programda bir yönergenin ya da sabitin adlandırılması için kullanılıyor. Anımsatıcı yönerge işlevini sembolik olarak açıklayan İngilizce sözcüklerin kısaltmasıdır ve bu alanın herhangi çevirici yönergede yer alması şarttır. İşlenenler verilen yönergenin çalışmasında kullandığı verilerdir. Yorum çevirici yönergenin sonunda yazılıyor ve programda bağımsız ifadeler olarak yer alıyor. Çevirici yorumları her zaman; (nokta vürgül) işaretiyle ayrılıyorlar.

İşlenenleri tanımlayan bitlerin sayısını azaltmak için iki yöntem kullanılıyor. Birinci yönteme göre daha fazla kullanılan - işlenen bellek konumu yerine bazı genel yazmaca yerleştiriliyor. İkinci yöntem fazla işlenenin tek şekilde görünüm yöntemidir. Örneğin, aynı bir yönergede, aynı bir işlenen hem veri kaynağı hem elde edilen sonucun hedef noktası olabilir.

Adresleme şekilleri aranan işlenenin bulunma şeklini açıklıyorlar. Yazmaçların adreslenmesi sırasında işlenenler mikroişlemcinin genel yazmaçlarında bulunuyorlar. Doğrudan adreslemede işlenen yönergenin içinde yer alıyor. Doğrudan sözcüğünün anlamı verinin hemen yönerge kodunun devamında olduğu demektir. Dolaylı adreslemede yönerge işlenenin bulunduğu bellek konumunun adresini içeriyor.

Aktarma için en çok kullanılan yönerge MOV hedef, kaynak yönergesidir. Hedef ve kaynak: bellek konumları, yazmaçlar ya da doğrudan verilmiş işlenenler olabilir.

Belleğe ya da bellekten aktramak için özel yönergeler vardır. LOAD adres yönergesi, verileri adresi verilmiş bellek konumundan akümülatöre aktarmak için kullanılıyor. STORE adres yönergesi, verilerin akümülatörden verilen adresli bellek konumuna aktarılması için kullanılıyor.

Mikroişlemcilerin çoğunda verilerin dış aygıttan mikroişlemciye aktarılması için bir yönerge vardır (İN) ve verilerin yazmaçtan dış aygıtlara aktarılması için bir yönerge vardır (OUT).

En çok kullanılan aritmetik yönergeler toplama ve çıkarma yönergeleridir. Toplama için ADD yönergesi, çıkarmak için ise SUB yönergesi kullanılıyor. Çarpma yönergesi MULL'dur ve İngilizce çarpma, multiply sözcüğünün kısaltmasıdır. Bölme yönergesi DIV yönergesidir ve ingilizce bölme sözcüğü olan divide'in kısaltmasıdır.

En önemli mantıksal yönergeler şunlardır: AND (mantıksal çarpma), OR (mantıksal toplama) ve NOT (olumsuzluk ya da birinci tümleyici) yönergeleridir. Bazı mikroişlemciler XOR (dışlamalı ya da), NOR (OYADA devresi) ve NAND (OVE devresi) yönergelerini de kullanıyor.

OR mantıksal yönergesi veriden biri bitin test edilmesi gerektiği zaman kullanılıyor. Maskede, test edilen bitin pozisyonunda bir koyuluyor, tüm diğer pozisyonlarda mantıksal sıfır duruyor. 16, 32 ya da 64 bitli veriden bir baytın ayrılması için AND yönergesi kullanılıyor ve maske yok olması gereken pozisyonlarda sıfırlardan oluşuyor.

ROL sola döndürme yönergesidir, ROR ise sağa döndürme yönergesidir. Bitlerin döndürülmesi birkaç kez gerçekleşebilir ve döndürülmesinin gerçekleşek olan sayısı yönergenin sonunda yazılmış olan sayıyla belirleniyor. Kaydırma yönergelerinde verideki bitler birkaç pozisyon için sola ya da sağa doğru kayıyorlar, ancak kayarken veriden çıkan bitler kayboluyorlar, verinin diğer tarafından geri dönmüyorlar.

Atlama yönergelerin anımsatıcısı JMP'dir. Atlamalar koşullu ya da koşulsuz olabilir. Atlama yönergeleri kendilerinde hiçbir işlenmeyi içermiyor, sadece atlanması gereken bellek yerinin adresini içeriyorlar. Dallanma yönergelerinde iki işlenen arasında kıyaslama yapılıyor ve elde edilen sonuca bağlı olarak verilen yönergeye atlama gerçekleşecek ya da gerçekleşmeyecek.

Ana program alt program içerdiği zaman, alt programın tamamlanmasına kadar ana programda kesinti meydana geliyor. Böyle durumda, program sayacında alt programdaki ilk yönergenin adresi yerleşiyor ve sayacın değeri alt programın son yönergesi çalıştırılana kadar bir için artıyor. Alt programlarla çalışmak için iki yönerge vardır. Biri alt programı çağırmak için kullanılıyor (Call), diğer yönerge ise alt programdan ana programa dönmek için kullanılıyor (RET).

Doğrusal programlar aktarma, aritmetik mantıksal yönergelerden ve döndürme ve kaydırma yönergelerinden oluşuyor. Yönergeler bir bir üzerine sıralanıktır ve o sıralamada çalıştırılıyor.

Dallanmalı programlar içinde atlama yönergeleri (Jump) ve dallanma yönergeleri (Branch) içeriyor. Bu yönergelere kontrol yönergeleri denir, çünkü programın akışını değiştirebiliyorlar, yani sıradaki yönerge her zaman mevcut yönergenin altında bulunmuyor.

Döngüler programa geri dönmek demektir ve dönmek birkaç kez tekrarlanabilir. Döngüler sayaçların kullanmasını gerektiriyor ve sayaç aslında döngünün kaç kez tekrarlanacağını gösteriyor. Sayaçlar genelde yazmaçlardır ve ileriye ya da geriye sayabilirler.

Sorular ve ödevler:

1. Çevirici makine diline en yakın olan programlama dilidir. Açıkla!
2. Çeviricinin programlama dili olarak avantajları ve dezavantajları nedir?
3. Bir çevirici yönergenin bileşen parçalarını say ve onların kullanışlarını açıkla!
4. Farklı sayı sistemlerinde gösterilen işlenenler nasıl işaretleniyor?
5. Etiketler hangi çevirici yönergelerinde kullanılıyor?
6. Üç - adresli ve iki - adresli çevirici yönergelerini kıyasla!
7. Çevirici yönergenin tüm dört adresleme şeklini say!
8. Yazmaçların adreslemenin avantajların ve dezavantajların hangilerinin olduğunu açıkla!

9. Değişkenlerle çalışmak için hangi adresleme şekli kullanılıyor?

10. Bellek konumların adreslerini saklayan yazmaçlarına ne denir?

11. Genel mikroişlemcinin yönergeler kümesinin ayrıldığı yönerge gruplarını say!

12. İntel şirketinin aktarma yönergelerinde veri kaynağının ve hedefin pozisyonları nedir?

13. Bellekten mikroişlemciye ve ters yönde veri aktarımı için özel yönergeler hangileridir?

14. “IN yazmaç, arabirim adresi” yönergesini yorumla!

15. MUL kaynak yönergesinde işlenenler nerede bulunuyor?

16. Mantıksal yönergelerde maskelerin kullanımını açıkla!

17. Birkaç döndürme ve kaydırma yönergesi say!

18. 1000110011010011 verisini
    a) sola doğru üç yer için döndür
    b) dört yer için sağa kaydır

19. Dallanma yönergeleri ve atlama yönergeleri arasındaki fark nedir?

20. “BEQZ kaynak, adres” yönergesini yorumla!

21. Alt programlarla çalışma yönergelerinde yığın belleğin kullanışını açıkla!

22. Hangi programlara kaynak programlar, nesne programlar ve çalıştırılır programlar denir?

23. Doğrusal programların, dallanmalı programların ve döngülü programların yapılarını açıkla!

24. Bir bellek konumunun içeriğini diğer bellek konumdan çıkarma işlemini gerçekleştiren program bölümü yaz!

25. Aşağıdaki program bölümünün nasıl işlevi vardır?

```
MOV R1,80H
TEKRARLA: ROL R1,1
JMP TEKRARLA
```

6. Mikrodenetimciler

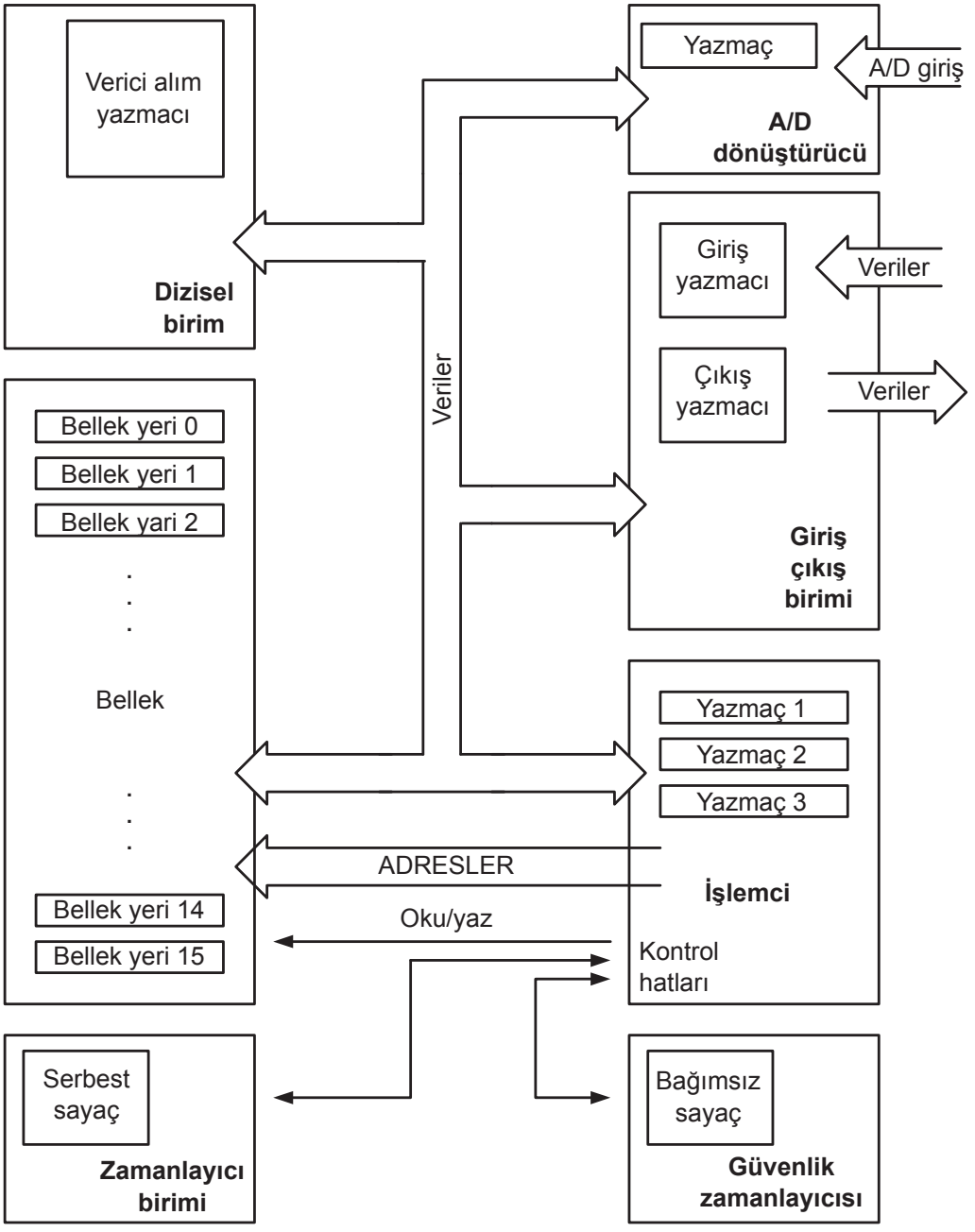
6.1. Mikrodenetimcilere Giriş

Mikrodenetimci ve mikroişlemci arasında büyük sayıda farklar vardır. Birinci ve en önemli fark onların işlevleridir. Mikroişlemcinin çalışması için bellekler ve giriş çıkış aygıtlar gibi diğer bileşenlerin eklenmeleri gerekiyor. Bunun anlamı aslında mikroişlemcinin bilgisayarın sadece "kalbi" olduğu demektir. Diğer taraftan **mikrodenetimci herşeyin bir yerde** olması için tasarlanmıştır. Böylece aygıtların tasarımı için gereken alan ve zamanda tasarruf yapılıyor. Mikrodenetimci **tümleşik devre şeklinde mikrobilgisayardır**. Mikrodenetimcinin yapısı bir bilgisayarda olması gereken herşeyi içeriyor: mikroişlemci, bellekler ve veri aktarımı için bakır hatlar. Tabii ki, mikrodenetimcinin donanımı mikroişlemcinin donanımına göre kıyaslanamaz kadar küçüktür. Örneğin, PIC16f84 mikrodenetimcinin sadece 18 pini var, RAM belleğinin kapasitesi sadece 68B'tır, çalışma frekansı yaklaşık 4 MHz değerindedir. Ancak bu büyüklükler mikrodenetimcinin işlevini sınırlandırmıyor. Mikrodenetimcinin birçok kez programlanabilme olanağı, düşük fiyatı ve alan tasarrufu, mikrodenetimcilerin çok sayıda elektronik kurgularda kullanılmasına destek veriyor. Mikrodenetimcilerin kullanımını etrafımızda her yerde bulabiliriz: ev aletlerinde, sanayide, otomobillerde, oyuncaklarda vs.

Bugün işlemsel ve otomatik yönetim, mikrodenetimcilerin kullanımı olmadan düşünülemez. Mikrodenetimciler etrafındaki ortam hakkında bilgileri, onların giriş pinlerine takılmış olan algılayıcılar (senzörler) aracılığıyla alıyor. Ondan sonra alınan veriler mikrodenetimcide işleniyor ve programın çalıştırılmasından sonra mikrodenetimcinin çıkış pinlerine takılmış yönetim birimler aktifleştiriliyor (elektromotorlar, pompalar, görüntü birimleri vb). Elektroniğin tarihsel gelişimine bakarsak, analog elektronikten sonra dijital elektroniğin geldiğini, dijital elektroniğin mikroişlemci tekniğinin gelişimine yol açtığını, ondan mikrodenetimcilerin meydana geldiklerini, mikrodenetimcilerin de robotiğin gelişimini koşullandırdığını söyleyebiliriz.

Resim 6.1.'de Mikrodenetimci, onun tüm oluşturan parçalarla ve onların bağlanmasının blok modeli gösterilmiştir.

Bilgisayarlarda olduğu gibi, mikrodenetimcilerde **bellek birimi** için en önemli terimler adresleme ve bellek yerleridir. Bellek yerleri bir bir üstüne sıralanıktır ve her bellek yerin sıra numarası var. Bu sıra numarasına bellek adresi denir.



Resim 6.1. Genel mikrodenetimcinin blok modeli

Adresleme, bir bellek yerin seçiminden başka birşey değil. R/$\overline{W}$ kontrol sinyali, hangi işlemin gerçekleşeceğini karar veriyor, bellekten okumak ya da belleğe yaz-

mak. Bellekte saklanan verileri ve yönergeleri merkez işlemci birimi işletiyor. Merkez işlemci birimin içeriğinde yazmaçlar, aritmetik mantık birimi ve kontrol (denetim) birimi bulunuyor.

Belleğin merkez işlemci birimiyle bağlanması için **veriyolu** denilen paralel bakır iletkenlerden oluşan bağlamlar kullanılıyor. İletkenlerin sayısı 8, 16 ya da fazla olabilir. Veri veriyolu işlenen veriler kadar geniştir, adres veriyolun genişliği ise bellek konumların sayısına bağlıdır.

Mikroişlemci, bellekler ve veriyolları bir işlevsel bütün oluşturuyorlar, ancak onun dış dünyayla hiçbir bağlantısı yoktur. Bu yüzden, bu işlevsel bütüne birkaç bellek konumundan oluşan bir blok daha ekleniyor. Bu bloğun bir tarafı veri veriyoluyla bağlıdır, diğer taraftan ise gözlerle görünür mikrodenetimcinin pinleriyle bağlıdır. Bu eklenen konumlara **kapılar** denir. Kapılar, özeliklerine göre giriş kapıları, çıkış kapıları ya da iki yönlü kapılar olabilir. Kapıyla çalışmaya başlamadan önce gereken kapıyı seçmek gerekiyor, ondan sonra da veri okunuyor ya da yazılıyor. Şöyle ki, kapılar sıradan bellek konumları gibi davranıyor, şu farkla ki onlar mikrodenetimcini pinlerinin mantıksal durumuna doğrudan etkiliyorlar.

Ancak bu iletişim şeklinin de kendi dezavantajları var, özellikle büyük mesafede veri akatarımı söz konusu olduğu zaman. Sekiz bitin aktarımı için sekiz aktarım hattı gerekecek ve böylece tutumluluğu azalıyor. Bundan dolayı **dizisel işletişim birimi** gerekiyor.Dizisel iletişim birimi üç hattan veri aktarımı sağlıyor: alıcı, verici ve kontrol hattı. Böyle bir konseptin çalışması için veri değişim (aktarım) kurallarının belirlenmeleri gerekiyor. Bu koşullara protokol denir. Örneğin, verici hattında aktarımın başlatılmasına kadar mantıksal birimi yerleşiyor. Aktarım başladığı zaman, verici hattı mantıksal sıfıra iniyor ve alıcı taraf verinin gönderildiğini bilecek. Verici taraf ikili verinin bitlerini gönderiyor ve her bit için bir dijit palsı zaman süresinin gerektiğini tahmin ediyoruz. Sekizinci bitin gönderilmesinden sonra, ya da sekizinci dijit palsından sonra verici hattı yeniden yüksek seviyeye yükseliyor ve onunla aktarımın tamamlandığı işaret ediliyor. İki hattın olduğu için, verici ve alıcı, mikrodenetimci aynı zamanda hem veri gönderebilir hem de veri alabilir.

Zamanlayıcı bloğu mikrodenetimcinin diğer bir çok önemli parçasıdır. Zamanlayıcı geçirilmiş zaman için bilgi veriyor. Zamanlayıcı aslında bir **serbest sayaçtır** ve değeri aynı zaman aralıklarda artıyor. Zamanlayıcının T1 ve T2 anlarında değerleri biliniyorsa, onların farkını hesaplayarak geçirilmiş zaman hakkında bilgi alınabilir.

Mikrodenetimcinin en önemli özelliklerinden biri onun hassasiyetidir. Yönetim sürecinde bir engelin meydana geldiğini tahmin edelim ve bizim denetimci programdan çalıştırmasını durdurmuş ya da ondan daha kötüsü yanlış çalışmaya devam etmiş olabilir. Bu sorunu aşmak için mikrodenetimci sıfırlanıp yeniden başlatıl-

ması gerekiyor. İnsan, mikrodenetimcinin çalışmasını sürekli olarak gözetleyemez. Bu yüzden dolayı ek olarak, bekçi köpeği anlamına gelen watchdog olarak adlandırılan **güvenlik sayacı** ekleniyor. Programın her doğru çalıştırılmasından sonra güvenlik sayacında sıfır yazılıyor. Programın bir yerinde "takılma" meydana gelirse, sıfır yazılmayacak, sayacın değeri artmaya başlayacak ve en yüksek maksimum değerine geldiği zaman kendisi mikrodenetimciyi yeniden başlatacak. Bu şekilde güvenlik sayacı programın yeniden çalıştırılmasına yol açacak, ancak bu kez doğru şekilde çalıştırılmasını sağlayacak.

Dış sinyalleri mikrodenetimcinin kullandığı sinyallerden (sıfırlar ve birler) büyük ölçüde farklı olduğundan dolayı, onların ikili şekilde dönüştürülmeleri gerekiyor. Bu görevi analog - dijital dönüştürme birimi yapıyor.

Mikrodenetimcinin programlanması için kullanılan programlama dillerden çeviriciyi, C ve Beyzik programlama dillerini vurgulayabiliriz. Çevirici daha düşük seviye programlama dilidir ve programlama yavaş yapılıyor, ancak bellekte en az yer alıyor ve programın çalıştırma hızı açısından en iyi çözümler sunuyor.

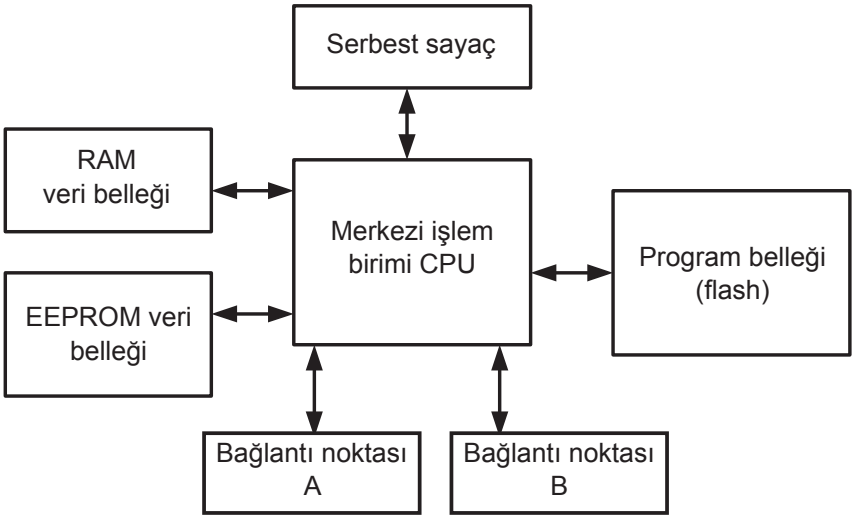
6.2. PIC 16f84 Mikrodenetimcisi

Pratik olmamız için, mikrodenetimcilerin incelemesini bir gerçek ve çok sıkı kullanılan PIC16f84 denetimcisini inceleyerek yapacağız. PIC kısaltmasının iki anlamı olabilir. PIC, **Programmable Intelligent Computer**'in kısaltması olabilir, ancak bazı yayınlarda PIC **Programmable Interrupt Controller** sözlerin kısaltması olarak tanımlanıyor ve anlamı programlanabilir kesinti denetimcisi demektir. Mikrodenetimcilerde kesintilerin onarımı 8 - bitli mikroişlemcilerde kesintilerin onarımına çok benzerdir. Kesit türleri ve onların kontrolü daha geç açıklanacaktır.

PIC 16f84 **üç** farklı **modta** (kipte) çalışıyor: program modu, çalışma modu ve bekleme (dinlenme) modu. Program modunda PIC 16f84 programlanıyor, yani program belleğinde kullanıcı programı giriliyor. PIC 16f84'in bir elektrik kurguda yerleşmişken kendi yatağından çıkaralmadan programlama olanağı var. Bu süreç ISCP (In - Circuit Serial Programming) adıyla biliniyor. Çalışma modunda denetimci amacı olduğu görevleri yerine getiriyor, verileri işletiyor ve farklı süreçleri yönetiyor. Mikrodenetimci bekleme modunda (sleep), onun aktif çalışmadan gerek olmadığı durumda enerji tasarrufu amacıyla ayarlanıyor. Gereğe göre, PIC 16f84 uyanabilir ve başlayan programın çalıştırılmasıyla devam edebilir ya da komple olarak sıfırlanabilir.

Resim 6.2'de PIC 16f84 mikrodenetimcinin basitleşmiş blok - modeli gösterilmiş. Kısaca oluştuğu daha önemli bileşen parçalarını açıklayacağız.

Merkezi işlem birimi denetimcinin "kalbidir". Onun bilgisayarda işlemcinin olduğu gibi aynı işlevi var ve bellek konumlarının adreslerini hesaplamak ve programda verilen yönergelerin kod çözümlülüğünü yapmak ve yönergeleri çalıştırmak için kullanılıyor. PIC16f84 iki adresleme türü kullanıyor, dolaysız (direkt) ve dolaylı (enderekt) adresleme, ancak onları PIC16f84 denetimcinin bellek organizasyonu inceyeceğimiz bölümde daha yakından tanıyacağız. **W çalışma yazmacıdır** ve işletilen verilerin kaynağı olarak kullanılıyor, ancak aynı zamanda gerçekleşen aritmetik mantık işleminden sonra elde edilen sonucun hedef noktası olarak da kullanılıyor. W harfi İngilizce Working sözcüğünün ilk harfidir ve çalışma anlamına geliyor. Tüm işlemler, toplama, çıkarma, kaydırma ve diğerleri gibi aritmetik mantık biriminde gerçekleşiyor.



Resim 6.2. PIC16f84 denetimcinin blok modeli

Mikrodenetimcinin çalıştırdığı kullanıcı programı, program belleğinde bulunuyor. Yönergeler birer birer önce yönerge yazmacına, ondan sonra da yönetim birimine götürülüyor ve burada kod çözümlenmesi yapılıyor. Kod çözücüsü ve yönetim birimi aslında aritmetik mantık birimi için ve uygun yazmaçları devreye getirmek için kontrol sinyalleri üretiyorlar. Bazı yönergelerin çalıştırılması durum yazmaçta bayrakların durumuna etkileyebilir. Bir bayrağın aktifleşip aktifleşmeyeceği yönergenin çalıştırılmasından sonra elde edilen sonuca bağlıdır. Durum yazmacında bitlerin anlamını daha geç, PIC16f84 mikrodenetimcinin tüm yazmaçların detaylı inceleyeceğimiz bölümde daha iyi tanıtacağız.

EEPROM **belleği,** elektrik kaynağının kapatılmasından sonra da verileri koruyor. EEPROM belleğinde çalıştırılan programlardan elde edilen veriler saklanıyor, ancak işletmek için önemli olan sabitler de burada hafıza ediliyor. Örneğin, böyle bir veri, sıcaklık düzenleyicilerde sıcaklık değeri olur. RAM geçici bellektir ve orada ara sonuçlar ve elektrik kaynağının kapatılmasından sonra gerekmeyen, büyük önemi olmayan veriler saklanıyor.

PIC16f84 mikrodenetimcinin **iki giriş çıkış kapısı A ve B** kapıları vardır ve onların aracılığıyla dış dünyayla iletişim kuruyor. Daha geç A ya da B kapıların bazı pinlerinin kesintisini yaratmak için ya da mikrodenetimcinin serbest sayacın girişi için nasıl kullanıldığını göreceğiz.

Mikrodenetimcinin **pals üreticisiyle ve zamanlayıcılarla,** daha geç, PIC16f84 mikrodenetimcinin zaman sistemini inceleyeceğimiz bölümde tanıyacağız. Bu bölümde sadece PIC16f84 mikrodenetimcinin dış dijit pals kaynağı (kristal salıngaç ya da RC salıngaç) kullandığını anacağız. Serbest sayaç 8 - bitlidir, kristal salıngaçın her dördüncü atışını sayıyor ve ulaşabildiği en yüksek değer 255'tir. Ayrıca, zamanlayıcı biriminin yapısında watchdog güvenlik sayacı da bulunuyor.

PIC16f84 mikrodenetimcisi **dört şekilde sıfırlanabilir**. Elektrik kaynağının açıldığından hemen sonra, tüm yazmaçların başlangıç durumuna getirilmesi amacıyla sıfılanabilir. Mikrodenetimci MCLR (microcontroller clear) pinine mantıksal sıfırın getirilmesiyle sıfırlanabilir. Sıfırlama güvenlik sayacın en yüksek değerinin aşılmasıyla ya da bekleme moduna geçilmesiyle meydana gelebilir. Sıfırlamanın en önemli etkilerinden biri program sayacın değerini sıfıra (0000H) ayarlamaktır ve bu durumda program birinci yazılmış yönergeden çalışmaya başlanması sağlanıyor.

6.3. PIC16f84'ün Bellekleri

Belleklerin kapasiteleri çok düşüktür: RAM=68B, EEPROM=64B ve program belleği=1KB. Resim 6.3.'te PIC16f84 mikrodenetimcinin bellek organizasyonu ve farklı bellek türlerin bellek haritası verilmiştir.

Şimdi tüm bellek türlerin işlevlerini ve onlara ulaşma şekillerini daha detaylı tanıyacağız.

RAM belleği

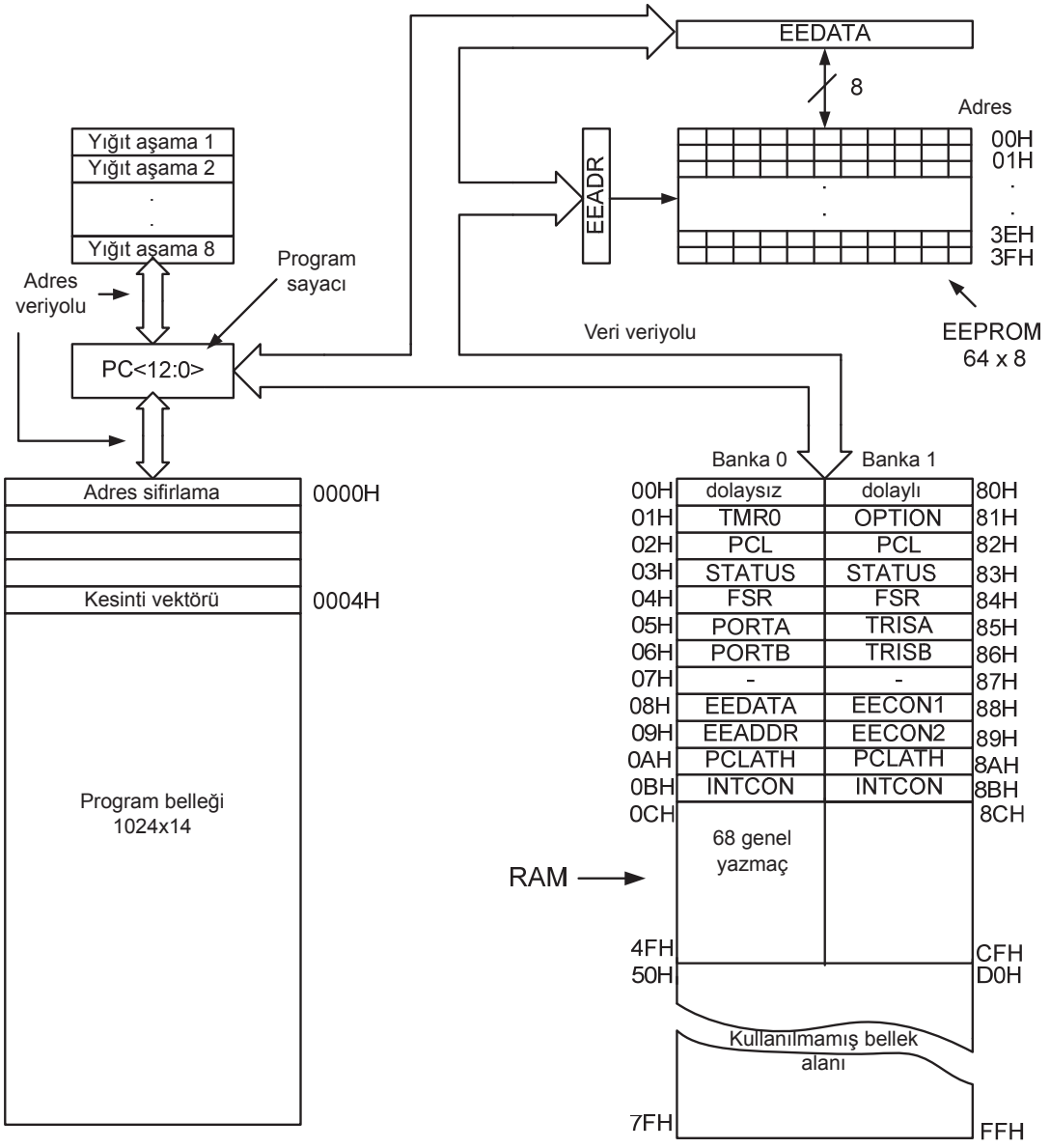
Mikrodenetimcilerde **RAM** belleğin, tüm diğer bilgisayarlarda RAM belleği gibi aynı işlevi var. RAM geçici bellektir ve elektrik kaynağının kapatılmasından sonra tüm verileri kayboluyor. RAM **çalışma belleğidir**, yani RAM belleğinde işlemci verilen programın çalıştırılması için gereken tüm verileri saklıyor. İşlemci bellek konumlarına uygun adresler göndererek erişiyor.

Ancak organizasyon açısından PIC16f84'ün RAM belleği genel mikroişlemcinin RAM belleğinden büyük ölçüde farklıdır. **RAM belleği iki bankaya ayrılıyor** ve banka 0 (sıfırıncı banka) ve banka 1(birinci banka) olarak işaretleniyor. Banka 0'a, adresleri sıfırla başlayan konumlar aittir (00H'dan 4F'e kadar), banka 1'e ise adresleri birle başlayan konumlar aittir (80H'den CFH'ye kadar). Adresleri 0CH'den 7FH'ye kadar ve D0H ile FFH arasında olan konumlar, şimdilik kullanılmıyor ve onların okunması her zaman sıfır sonuç veriyor. Bellek bankasının seçimi için durum yazmacı kullanılıyor ve onu devamda inceleyeceğiz. Sıfırıncı banka seçilmiş olurken birinci bankadan konumlara erişim yapılamaz.

Bilgisayarlarda **genel ve özel amaçlı yazmaçlar** mikroişlemcide bulunuyor, PIC 16f84'te ise bu yazmaçlar **RAM belleğinde yerleşiktir**. RAM belleğin her iki bankasında ilk 12 bellek yeri özel amaçlı yazmaçları tanımlıyor. Özel amaçlı yazmaçlar kısaca SFR (Special Function Registers) olarak yazılıyor. Örneğin mikrodenetimcinin serbest sayaçı (TMR0) adresi 01H olan banka 0'dan bellek konumu tanımlıyor. Durum yazmacı iki bellek bankasında bulunabiliyor, yani sıfır bankasında durum yazmacı adresi 03H olan konumu tanımlıyor, birinci bankada bulunuyorsak o zaman yazmaç adresi 83H olan yerdedir.

Program sayaçı 13 bitli yazmaçtır. Onun sekiz daha değersiz PCL (Program Counter Low) bitleri banka sıfırda 04H adresli yerde bulunuyorlar ya da banka birde 84H adresli yerde bulunuyorlar. Diğer beş daha değerli bitler sıfırıncı bankada 0AH adresli yerde ya da birinci bankada 8AH adresli yerde bulunuyor. Program sayacın gerçekleşmesi gereken sıradaki yönergenin adresini içerdiğini hatırlayalım. Verilen anda gerçekleşen yönergenin adresini bir için büyüterek, sıradaki yönergenin adresi elde ediliyor. Atlama yönergesi söz konusu olunca o zaman program sayacı atlama yönergesinde verilen adrese eşittir. Durum yazmacı ya da program sayacı gibi yazmaçlar iki bellek bankasında bulunduğu zaman, hangi bankada bulunduğumuz önemli değil.

Daha geç tüm özel amaçlı yazmaçların işlevlerini ve onların kontrol bitlerinin anlamlarını tanıyacağız.



Resim 6.3. PIC16f84'ün bellek organizasyonu

Özel amaçlı yazmaçların altında genel amaçlı yazmaçlar bulunuyor (GPR - General Purpose Registers). Toplam **68 genel amaçlı yazmaç** vardır. Genel yazmaçlara ulaşmak için hangi bankada bulunduğumuz önemli değil. Sıfırıncı bankada bulunursak, o zaman genel yazmaçları adreslenmeleri için 0CH'den 4FH'ye kadar adresler bulunuyor, birinci bankada bulunursak o zaman 8CH'den CFH'ye kadar adresler kullanılıyor. Bazan genel yazmaçlara ulaşmak için sembolik isimler kullanılıyor. Şöyle ki, özel komutlar vardır, direktifler (emirler) olarak adlandırılıyorlar ve onlarla her genel yazmaca sembolik isim veriliyor. Gerçekten, isim hafıza etmek, on altılı sayı sisteminde adresin değerini hafıza etmekten daha kolaydır. Emirleri, daha

geç PIC16f84 denetimci için program yazma sürecini açıklayacağımız bölümde tanışacağız.

Program belleği

Program belleği kalıcı fleş belleğidir ve onda **kullanıcı programı** bulunuyor. Program beleğinde program, B kapısının yedinci pini aracılığıyla, dizisel, birer birer bit olarak giriliyor. PIC16f84'te programın girilmesi için bilgisayar, programatör, özel yazılım (MPLAP, FPP...), programatörü bilgisayarın paralel bağlantı noktasıyla bağlanmak için kablo ve kullanıcı programı gerekiyor.

EEPROM belleği

İkinci veri ve kalıcı belleği **EEPROM belleğidir.** EEPROM belleğinde **kullanıcı programının çalıştırılmasıyla sonuç olarak elde edilen veriler** saklanıyor. Örneğin, bu tür veriler, uzak mesafede bulunan meteoroloji istasyonundan alınan veriler olabilir (sıcaklık, dış basınç, yağış miktarı) ya da bir gün içinde bir park yerinde park edilen otomobil sayısı hakkında veri olabilir vs. Program belleğinden farklı olarak, EEPROM'da veriler 8 bit birden paralel olarak okunuyor ve yazılıyor. EEPROM belleğine dolaylı şekilde erişiliyor ve EEPROM belleğinde rastgele, istenmeyen verilerin bellekte yazılmasından korunmak için özel yordam kullanılıyor. EEPROM belleğinde okumak ya da yazmak için üç özel yazmaç kullanılıyor. EEADR okunan ya da yazılan bellek yerin adresini koruyor. EEDATA, yazılması ya da okunması gereken veriyi içeriyor. EECON 5 - bitli yazmaçtır ve EEPROM belleğinden okunması ya da yazılmasıyla ilgili kontrol bitleri içeriyor. Bu bitlerin anlamı daha geç açıklanacak.

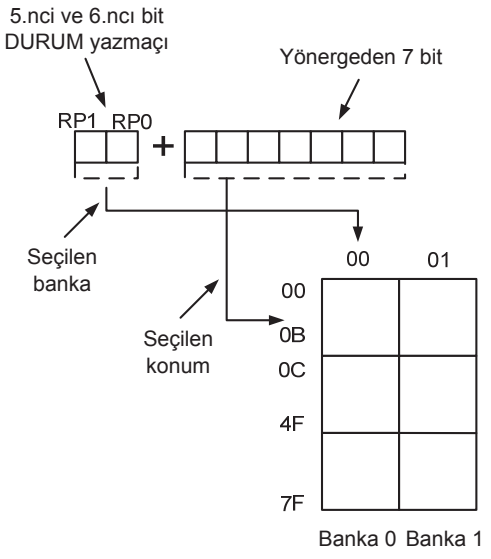
Yığıt bellek

PIC16f84'ün 8 aşamalı 13 - bitli yığıt belleği vardır. Onun genel işlevi, mikrodenetimcinin alt programın nerede çağrıldığının yerini bilmek için ana programdan alt programa geçildiği zaman program sayacının değerini korumaktır. Alt programdan ana programa geri dönüldüğü zaman yığıt bellekten değer program sayacına geri dönüyor.

6.4. PIC16f84'te Adresleme Şekilleri

RAM belleğinin yerlerine erişmek için iki adresleme şekli kullanılıyor: dolaysız ve dolaylı adresleme.

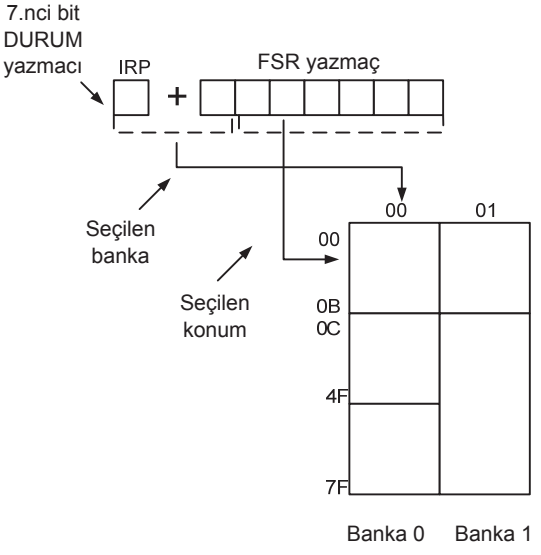
Dolaysız adresleme şeklinde RAM konumuna erişim adresi, genel mikroişlemcide olduğu gibi yönergenin içinde içeriktir. Resim 6.4.'te dolaysız adresleme süreci gösterilmiştir.



Resim 6.4. Dolaysız adreslemede adresin hesaplanması

Dolaysız adresleme ile 9 - bitli adresler elde ediliyor. İlk iki en değerli bit aslında durum yazmacından RP1 Ve RP0 bitleridir ve onlar iki bellek bankasından birini seçmek için kullanılıyor. Kalan yedi bit yönerge bitleridir ve seçilmiş bankadan bir yerin seçilmesi için kullanılıyor.

Dolaylı adreslemede bellek adresi üretim süreci resim 6.5'te gösterilmiştir. Resim 6.4. Dolaysız adreslemede adresin hesaplanması

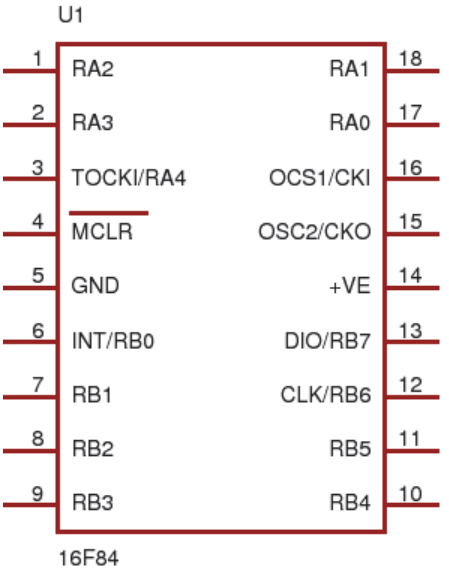


Resim 6.5. Dolaylı adreslemede adresin hesaplanması

Dolaylı adreslemede, adres yönergede içerik değildir. Adres FSR özel yazmaçta bulunuyor. Bu yazmaç, durum yazmacın yedinci biti IRP ile beraber RAM bellek-

ten bir yerin seçilmesi için adrese biçim veriyor. İstediğimiz veriyi okumamız için, seçilen bellek yerine gitmek gerekmiyor, o bellek yerindeki veri, sıfırıncı bankadan adresi 00H ya da birinci bankadan adresi 80H olan İNDF yazmacından okunabilir. Örneğin, 0FH adresli genel yazmaç 20H veriyi içersin. Buna göre, dolaylı adreslemenin uygulanmasıyla FSR yazmacı 0FH değerini içerecek, İNDF yazmacı ise 20H değerini içerecek. Dolaylı adresleme dizilimlerle çalışma sırasında kullanım görüyor.

6.5. PIC16f84'ün Pin - Diyagramı



Resim 6.6.'da PIC16f84'ün pin diyagramı gösterilmiştir.

Resim 6.6. PIC16f84'ün Pin - diyagramı

Mikrodenetimcinin veri pinleri yoktur. Ancak, dış dünyayla iletişim kurmak için A ve B bağlantı noktaların pinlerini kullanıyor. Bağlantı nokta terimi, aynı zamanda sıfırlar ve birler kombinasyonuna erişim ve ayarlanma olanağı veren, mikrodenetimcinin pinler grubunu tanımlıyor. Fiziksel olarak, bağlantı noktaları, denetimcinin pinlerinyle iletkenler aracılığıyla bağlanmış olan ve denetimcinin içinde bulunan yazmaç tanımlıyor. Bağlantı noktaları merkezi işlem sistemi ve dış dünya arasında fiziksel bağıdır. Pinlerin mantıksal durumu, PORTA ve PORTB özel amaçlı yazmaçlardan okunabilir. Resim 6.6.'da bu pinler RA ve RB ile işaretlenmiştir. R harfi, İngilizce yük anlamına gelen, *rack* sözünün ilk harfidir. **B bağlantı noktası sekizpinlidir, A bağlantı noktası ise beşpinlidir**. Onlar ya giriş bağlantı noktaları ya da çıkış bağlantı noktaları olabilir, iki yönlü olamazlar. A ve B bağlantı noktaların giriş ya da çıkış noktaları olmaları, TRİSA ve TRISB yazmaçların bitlerine bağlıdır. **TRİSA ve TRİSB** 8 - bitli yazmaçlardır. Örneğin, eğer TRİSB= 00001111 ise, o zaman B bağlantı noktasından dördüncü, beşinci, altıncı ve yedinci pin çıkış pinleridir, 3, 2, 1 e sıfırıncı pin ise giriş pinleridir. Bir değeri pinin giriş pini olduğu, sıfır ise pinin çıkış pini olduğu de-

mektir. Daha kolay anlamak için sıfır değeri İngilizce output sözünün ilk harfine hatırlarıyor, bir sayısı ise input sözünün ilk harfine hatırlatıyor.

PIC16f84'te A ve B bağlantı noktalarından bazıların çift işlevi var.

- **RA4/TOCKI** pini, A bağlantı noktasının dördüncü pinidir, ancak aynı zamanda serbest sayaçın giriş atışı olarak da kullanılabilir. Bu pinin A bağlantı noktasının dördüncü pini ya da serbest sayacın dış girişi olması, OPTION, TOSC yazmaçtan beşinci pinin mantıksal durumuna bağlıdır.

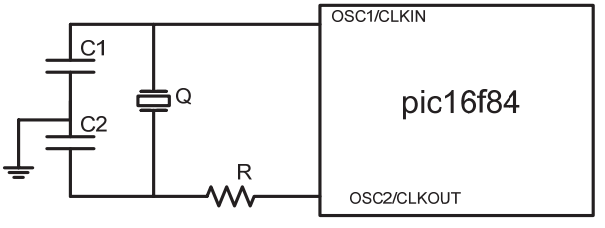
- **RBO/INT,** B bağlantı noktasının sıfırıncı pinidir, ancak aynı zamanda dış kesinti kaynağı olarak da kullanılabilir. Tüm kesinti pinleri, INTCON (Interrupt Control) yazmacın işlevinin incelenmesi sırasında açıklanmıştır.

- **RB4, RB5, RB6, RB7** pinlerin üçer işlevi var. Onlar B bağlantı noktasının giriş - çıkış pinleridir, ancak aynı zamanda mantıksal seviyenin değişmesi sırasında kesinti kaynağı olarak da kullanılabilir. RB6 ve RB7 program modunda şu şekilde kullanılıyorlar: RB6'ya tetiklmeme palsları getiriliyor, RB7 ise programın mikrodenetimciye dizisel girilmesi için kullanılıyor.

PIC16f84 mikrodenetimcisinden diğer pinlerin şu işlevleri vardır:

- **MCLR** alçak seviyeye ayarlanma sırasında iç yazmaçların sıfırlanması için kullanılıyor. Bu pin çalışmanın program modunda kullanılıyor ve +12 V ile + 14 V gerilimi arasında ayarlanıyor.

- Osilatörleri (salıngaçları) pals kaynağı olarak takmak için **OSC2/CLOCKOUT ve OSC1/CLKIN** pinleri kullanılıyor. Resim 6.7.'de PIC 16f84'ün kristal salıngaçla bağlanması gösterilmiştir.



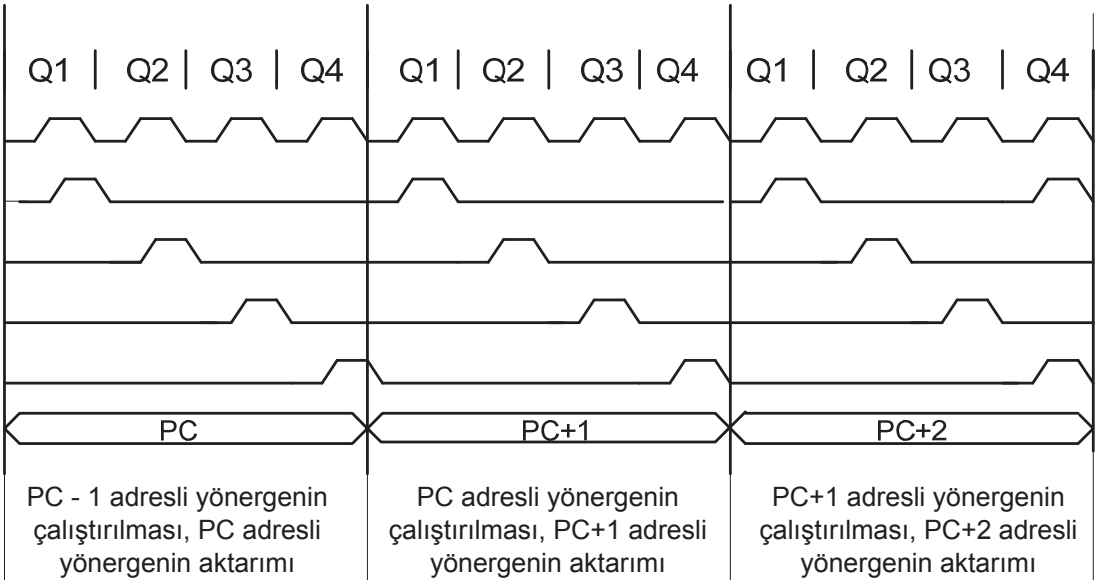
Resim 6.7. PIC16f84'ün kristal salıngaçla bağlanması

Mikrodenetimci birçok salıngaç türü kullanabilir: LP (low power), XT (crystal resonator), HS (high speed). Dijit pals kaynağı olarak RC salıngaçları da kullanılabilir.

- GND tablo, topraklamadır, +Ve ise elektrik kaynağıdır

6.6. PIC16f84'ün Zamanlama Sistemi

Mikrodenetimci eşzamanlı (senkron) çalışma modunda çalışıyor. Yönergeleri çalıştırmak için dijit palsı gerekiyor. Dijit palsı, mikrodenetimcinin pinlerine takılmış kristal salıngaçtan elde ediliyor. **Bir yönergenin çalıştırılması için dört dijit palsı gerekiyor Q1, Q2, Q3 ve Q4**. Bu resim 6.8.'de gösterilmiştir.

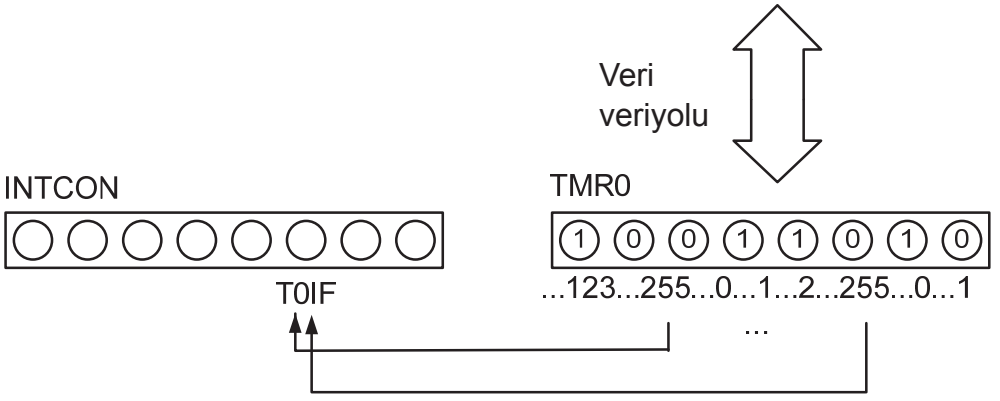


Resim 6.8. Yönerge çalıştırılmanın zamanlama diyagramı

Birinci dijit atışı Q1 sırasında, yönerge program belleğinden yönerge yazmacına aktarılıyor. İkinci dijit palsı Q2 yönergenin kod çözümlenmesi yapılıyor, sıradaki iki atışta ise yönerge etkin şekilde çalıştırılıyor. Resim 6.8.'de, bellekten işlemciye aktarılması gereken sıradaki baytın adresini gösteren program sayaç yazmacının içeriğindeki değişmeler gösterilmiştir.

PIC - 16f84 mikrodenetimcisi sıkça zamanlayıcı ya da sayaç olarak kullanılıyor. **Zamanlayıcı olarak çalıştığı zaman mikrodenetimci kristal salıngaçla bağlanıyor**. Periyod, dijit pals frekansının evrik değeridir (T=1/f). Zamanlayıcının sayması gereken dijit atışların sayısı, geçen zaman kristal salıngaçın periyoduyla bölerek elde ediliyor. Örneğin, kristal salıngaçın frekansı 1KHz ise, o zaman periyod 1ms olacak. Böylece 1s gecikme zamanı elde etmek için, kristal salıngacın 1000 periyodun geçmesi gerekiyor, 2s gecikme zamanı için 2000 periyodun sayılması gerekiyor vs.

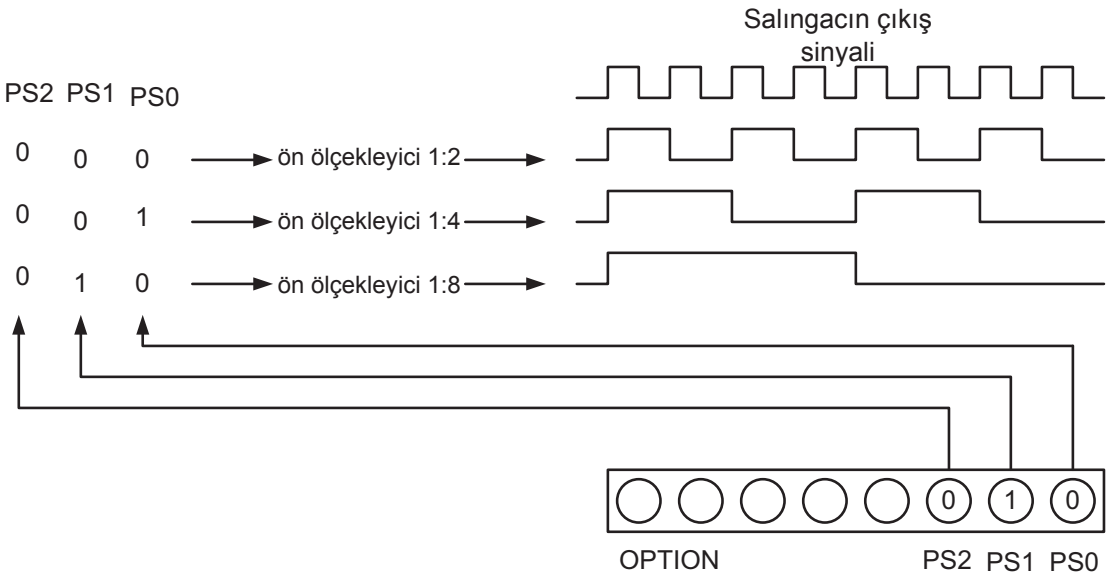
Mikrodenetimcinin zamanlayıcısı **TMR0** kısaltmasıyla işaretleniyor ve bu aslında timer0'ın kısaltmasıdır. TMR0 8 - bitli yazmaçtır ve 00H'den FFH'ye kadar sayabiliyor. En yüksek değere ulaşınca, zamanlayıcı tekrar sıfırdan saymaya başlıyor. Sayma sırasında bu tür geçişe **serbest sayacın aşması** deniliyor. Devamda TMR0 sayacın aşması mikrodenetimcinin çalışmasında kesinti yaratabilir mi göreceğiz. Böyle durumda, INTCON kesinti kontrol yazmacından, T0IF (Timer0 Interrupt Flag) biti aktifleştiriliyor. Serbest sayacın aşmalarının sayılmasıyla, en yüksek sayılan dijit palsların sayısı artıyor. Örneğin, sayaç en yüksek değeri 64 kez aşmışsa, o zaman mikrodenetimcinin 64·256=16384 dijit atışın saydığını diyoruz. Resim 6.9.'da TMR0 ve İNTCON yazmaçları arasında olan bağlantı gösterilmiştir.



Resim 6.9. Kesinti – sayacın aşması

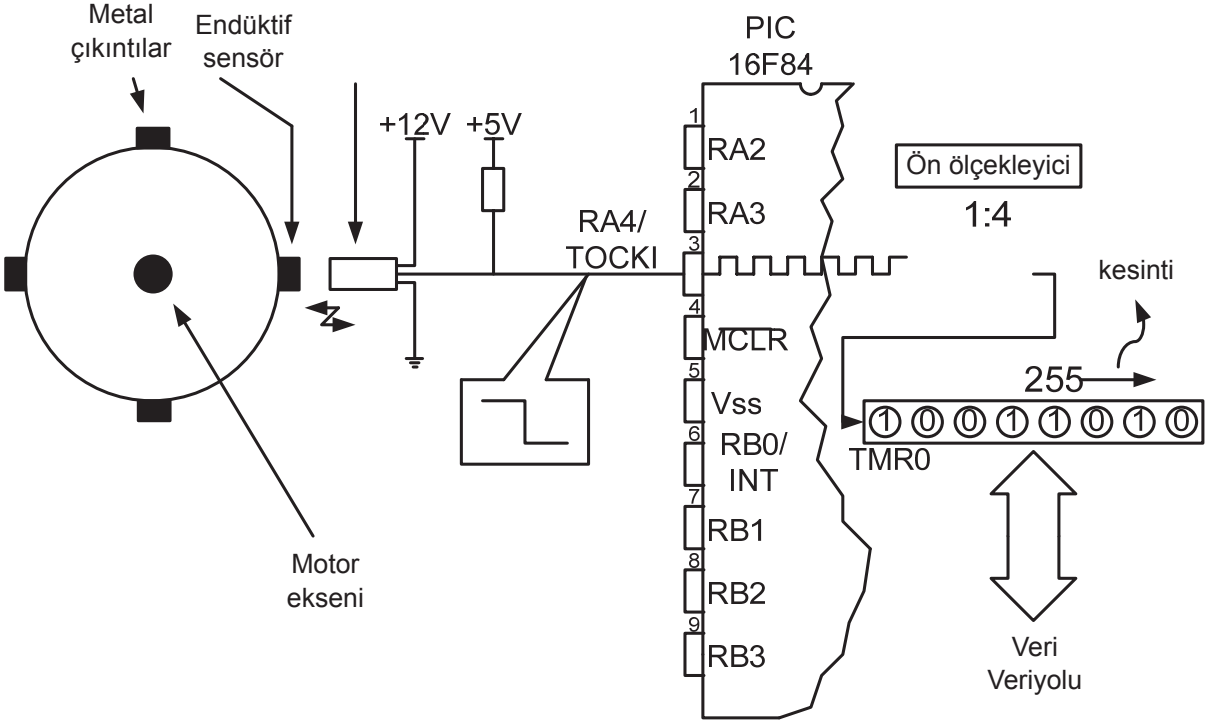
Serbest sayacın, kristal salıngaçın her dördüncü tetiğinin sayılması ilginçtir. Öylece, serbest sayacın frekansı kristal salıngaçın frekansından dört misli daha düşüktür. TMR0 serbet sayacın frekansı, ön ölçekleyici olarak adlandırılan özel frekans bölücünün kullanımıyla daha da çok azalabilir. Ön ölçekleyici aslında OPTION sayacın control yazmacının içeriğinde bulunan 3 - bitli kodtur. Bu 3 - bitli kodun değerine bağlı olarak bölücü 2 ile 256 arasıda değer alabilir. Ön ölçekleyicinin, serbest sayacın frekansına etkisi resim 6.10.'da gösterilmiştir.

Kristal salıngacın dijit atışları dışında, TMR0 yazmacı **RA4/TOCKI** pininde durumun değişmelar sayısını da sayabilir ve o zaman mikrodenetimcinin **dış rastgele değişmelerin sayacı** olduğunu diyoruz. Rastgele değişmeler, meydana gelmesi öngörülemeyen değişmelerdir. Örneğin, bir dakika içinde RA4/TOCKI pinin giriş geriliminde binlerce değişme meydana gelebilir, ancak ondan sonraki dakikada hiçbir değişme olmayabilir. Örneğin, RA4/TOCKI pinine optik sensör bağlayabiliriz ve bu sensör ne zaman aktifleşse, TMR0 yazmacın değeribiriçin artıyor. Bu şekilde park yerinde giren ve çıkan otomobillerin sayısı sayılabilir, klavye tuşuna basma sayısı belirenebilir vs.



Resim 6.10. Ön ölçekleyicinin TMR0 sayacın üzerindeki etkisi

Resim 6.11.'de PIC16f84 mikrodenetimcinin dönmelerin sayacı olarak kullanma örneği verilemiştir. RA4/TOCKI pinine endüktif sensor bağlanmıştır. Bu sensor dönen dingilden birkaç milimetre yakınlığında takılmıştır.



Resim 6.11. PIC16f84'ün girişine endüktif sensörün bağlanması

Motorun ekseninde dört tane metal çıkıntı (yumru) takılmıştır. Bu çıkıntılardan biri sensöre yaklaşınca, sensörün endüktansı değişecek ve RA4/TOCKI giriş pininde gerilim düşüşü meydana gelecek. Resim 6.11.'den bu pin için etkinleştirici iniş kenarı, yani giriş gerilimin birden sıfıra düşmenini olduğu görünüyor. Her iniş kenarı serbest sayacın değerinin bir için artmasına yol açacak. Serbest sayaç alabildiği en yüksek değere ulaşınca, yani 256 sayma yaptıktan sonra, kesinti - sayacın aşması meydana gelecek. Bir saniyede dönme sayısının elde edilmesi için, serbest sayacın değeri geçen zaman dört'le bölünmesine gerekiyor (dört metal çıkıntıdan dolayı).

Serbest sayaç dışında, zamanlayıcı birimin içeriğinde, adı **güvenlik sayacı** olan bir sayaç daha bulunuyor. Watchdog zamanlayıcısı aslında mikrodenetimci yonganın içinde olan RC salıngaçı tanımlıyor. Buna göre, güvenlik sayacı, OSC2/CLOCKOUT ve OSC1/CLKIN pinlerinde salıngaç takılmış olmadığı zaman bile çalışacak. Hatırlayalım, **güvenlik sayacı her yönergenin çalıştırılmasından sonra sıfırlandırılıyor**. Bazı yönerge çalıştıramılıyorsa, güvenlik sayacın değeri artmaya başlayacak ve en yüksek değerine ulaşınca, PIC16f84 mikrodenetimcisi sıfırlandırılıyor. Programın yeniden çalıştırıldığında, yönergenin tamamlanmamasına yol açan aynı engelin meydana gelmeyeceğini tahmin ediyoruz. Güvenlik sayacın açılma ya da kapanmasını, yapılandırma (konfigürasyon) kelimesinden, WDTE (Watchdog Timer Enable) sıfırıncı bitin aracılığıyla yapma olanağı vardır. Güvenlik sayacın periyodu 18ms'dir ve bu değer sıcaklığa ve denetimcinin elektrik kaynağına bağlı olarak değişebilir. Periyodun değeri, OPTİON sayaçlar denetim yazmacından ön ölçekleyicinın kullanımıyla artabilir. Watchdog ile TMR0 arasında, Ön ölçekleyicinin kullanacağı sayaçın seçimi, OPTİON yazmacında dördüncü bit olan PSA'ya bağlıdır.

6.7. PIC16f84'te Kesintilerin Tanınması

Her günlük hayatta kesintiyi, başlattığımız işin tamamlanmasını engelleyen olay olarak tanımlıyoruz. Kesintiler (İnterrupt) merkezi işlem birimi ile dış dünya arasında özel iletişim şeklidir.

Bazı **kesintilerin durumunu** işlemci **sürekli kontrol ediyor**. Böyle kesinti için örnek olarak klavye ile iletişimi verebiliriz. Programın çalıştırılması sırasında, eşit zaman aralıklarla, işlemci klavyeden herhangi bir tuşun basılmış olup olmadığını kontrol ediyor. Kesintinin (basılmış tuşun) meydana geldiği tespit edilince, kalvyeden hangi tuşun basıldığını belirleyen alt program çağrılıyor.

Bazı kesintiler tamamiyle rastgele olaylardır. İşlemci mevcut programı çalıştırıyor ve öngörülen kesintilerden biri meydana gelirse, işlemci mevcut programın çalıtırmasını durduruyor ve kesintiyi işletiyor. Kesintinin işletilmesi, kesintinin onarımı için özel alt programın çalıştırılmasıyla yapılıyor. Kesintinin işletilmesinden sonra, işlemci başlatılan programı tamamlıyor. Burada önemli olan şu ki kesinti meydana gelmeden önce, işlemci kesintilerin durumunu kontrol etmek için hiçbir etkinlik göstermiyor. Böyle bir kesinti için örnek olarak çamaşır makinenin çalışmasını alabiliriz. Çamaşır makinesi, kapısını kapamadan çalışmaya başlamayacak. Ancak çalışmaya başlayınca, kapının durumu artık göz önüne alınmıyor. Fakat, kapı bir kişi tarafından istemeyerek açılırsa ne olacak? Tabii ki böyle durumda makinenin çalışmanın durdurmasına (kesmesine) neden olacak. Ama meydana gelen engel ortadan kalktıktan sonra, kapı tekrar kapatılır ve makine çalışmaya durduğu yerden (kesintinin oluştuğu yerden) devam etmelidir. Demek ki, çamaşır makinesi kesintiyi başlamadan önce, programının çalıştırılması nereye kadar geldiğini hafıza etmelidir.

PIC16f84 mikrodenetimcinin **dört kesinti çeşiti** vardır:

- EEPROM belleğinde veri yazılmasının sonu.
- Serbest sayacın değeri aşmasından kaynaklanan TMR0 kesintisi.
- B bağlantı noktasından, yedinci, altıncı, beşinci ve dördüncü pinin durumu değişmesi sırasında meydana gelen kesinti.
- RB0/INT pinin dış kesintisi.

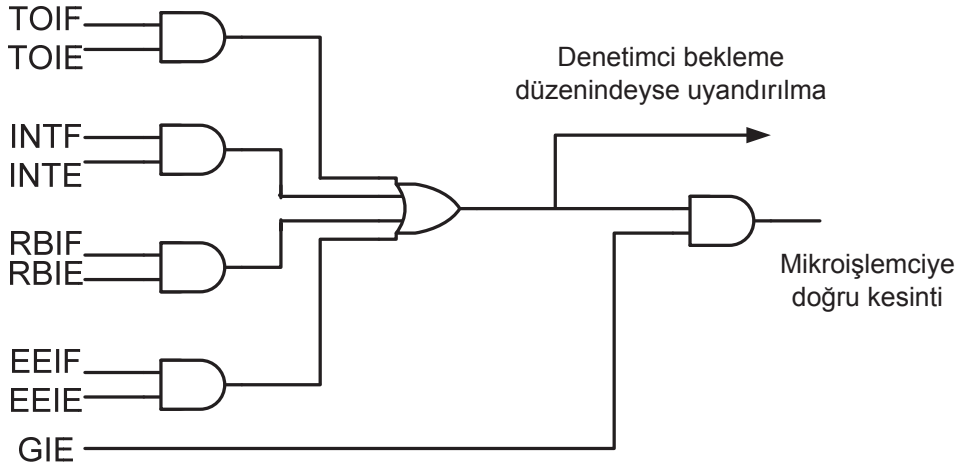
Her kesinti türü için ikişer biti vardır.

- Interrupt Enable bit (IE) – 4 tür kesinti olanağı vermesi için yazılım yoluyla yüksek seviyeye ayarlanıyorlar. Yazılımsal ayarlama, biti yüksek seviyeye getirmek ve bu şekilde programcıya dört olası kesintiden çalışması için birini seçmek olanağı vermesi demektir. Bu bitlerden bazısı alçak seviyede ayarlanırsa, o zaman öyle bir kesinti meydana gelse de görünmeyecek, işletilmeyecek.

- Interrupt Flag bit (IF) donanımlı olarak ayarlanıyor. Kesintinin meydana geldiğinde kullanıcıyı bildirmek için, mikrodenetimci bitleri yüksek seviyeye ayarlıyor. Kesintinin onarımı için alt programı çalıştırdıktan sonra, programcının kesintiyi sıfırlamaya gerekiyor.

Kesinti bayraklarıyla devreye koyan bitlerle, İNTCON kesinti kontrol yazmacını açıklayacağımız bölümde tanıyacağız. Her kesinti çeşidi için ortak bir bit var - GIE (General Interrupt Enable). Bu ortak bit ile tüm kesinti türleri birden yasaklanıyor ya da devreye giriyorlar. Bu bitin kullanışlıdır çünkü çok önemli bir programın çalışması kesilmesin diye tüm kesintiler için geçici olarak yasak getirilebilir. GİE biti sıfırlanırsa, o ana kadar çözülmemiş kesintiler yoksayılıyor. Çözülmemiş kalan kesintiler, GIE bitin yeniden ayarlanmasından sonra işletiliyor. Ayrıca, bir kesintiye cevap verildiği zaman, başka bir kesintinin oluşmaması için GİE biti tekrar sıfırlandırılıyor. PIC16f84 mikrodenetimcisi aynı zamanda fazla kesinti türleri işletemiyor ve kesinti onarımı için bir alt program içinde başka alt program çalıştıramaz. PIC16f84 mikrodenetimcinin kesinti mantığı resim 6.12.'de gösterilmiştir.

Kesinti zamanında, program sayacın değeri yığıtın ucunda korunuyor. Program sayacının çalıtırlması gereken yönergenin adresini içerdiğini, ancak kesintinin meydana geldiğinden dolayı bu yönergenin çalıştırılmadığını hatırlayalım. Program ancak sayacının korunması yeterli değil, çünkü kesinti ve ana programda kullanılan bazı yazmaçlar kullanılabilir. Bu yazmaçların içerikleri korunmazsa, kesinti programın tamamlanmasından sonra, ana program bambaşka değer içeren yazmaçlar elde edecek ve yanlış sonuçların elde edilmesine yol açılıyor. Bundan dolayı, program sayacın değeri dışında, W çalışma yazmaç değerin ve durumun yazmacın korunması gerekiyor.



Resim 6.12. PIC16f84 için kesinti mantığı

Ne yazık ki, yığıt bellekten veri alma ve veri verme yönergeleri yoktur. Yığıt bellekle çalışma yerine, yazmaçların eski değerlerinin korunması için yeni geçici yazmaçları tanımlayan özel koruma alt programı yapılıyor.

6.8. PIC16f84'te Yazmaçlar

OPTION yazmacı

Sayaçların ve zamanlayıcının kontrolu için OPTİON yazmacı kullanılıyor. Resim 6.13.'te içerdiği bitlerin kısaltmalarıyla OPTİON yazmacı gösterilmiştir.

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
RBPU	INTEDG	TOCS	TOSE	PSA	ön ölçekleyici		

Resim 6.13. OPTION yazmacında bitlerin sıralaması

Tablo 6.1.'de OPTION yazmacından tüm bitlerin kısaltmaları ve onların işlevleri verilmiştir. Önce TMR0 sayacı için atış kaynağının seçiminden başlayabiliriz. Onun için **TOCS** biti kullanılıyor. TOCS biti bir olunca, RA4/TOCKI pinin seviye değişiklikleri sayılıyor. TOSC biti sıfır olunca, salıngacın tetikleri sayılıyor. Pals kaynağı olarak RA4/TOCKI pinine bağlı olan dış kaynak seçilirse, hangi değişikliklerin sayılacağı belirlenmelidir. **TOSE** biti bire eşit olursa düşüş kenarları sayılacak, sıfıra eşit olursa yükselen kenarlar sayılacak. Ondan sonra sırada ön ölçekleyicinin seçimi geliyor.

Bit	Kısaltma	İşlev
yedinci	RBPU (RB Pull Up)	B bağlantı noktası için Pull up rezistörü 1 - kapalı rezistörler, 0 - açık rezistörler
altıncı	INTEDG (Interrupt Edge)	RA4/TOCKI pini için etkinleştirici seçimi 1 - düşen kenar, 0 - yükselen kenar
beşinci	TOCS (Timer0 Clock Select)	TMR0 için giriş sinyalın seçimi 1 - RA4/TOCKI pini 0 – kristal salıngaç
dördüncü	TOSE (Timer0 Select Edge)	TMR0 için etkinleştirici seçimi 1 - düşen kenar, 0 - yükselen kenar
üçüncü	PSA (Prescaler Select Assignment)	Ön ölçekleyici için sayaç seçimi 0 - TMR0, 1 - Watchdog
2, 1, 0	3 - bitli ön ölçekeleyici	Periyod çarpanı ya da frekans bölücü

Tablo 6.1. OPTİON yazmacında bitlerin işlevsel açıklamaları

OPTİON yazmacının en az değerli üç bitine ön ölçekleyici denir. Ön ölçekleyici çalışma frekancının bölücüsünü içeriyor. Şöyle ki, kristal salıngaçın çalışma frekansının KHz sıradan değeri varsa, demek ki periyodun değeri ms sıradandır. Bu değer insan için gerçek dışıdır ve artması gerekiyor. Bu amaç için ön ölçekleyici kullanılıyor. Frekans bölücünün, yani periyod çarpanın değeri tablo 6.2.'ye göre elde ediliyor.

Ön ölçekleyici= PS2 PS1 PS0	Serbest sayaç için frekans bölücüsü	Güvenlik sayacı için frekans bölücüsü
000	2	1
001	4	2
010	8	4
011	16	8
100	32	16
101	64	32
110	128	64
111	256	128

Tablo 6.2. Frekans bölücünün ön ölçekleyiciden bağımlılığı

PSA biti ön ölçekleyiciye hangi sayacın verileceğine karar veriyor. Serbest sayaçla (TMR0) çalışmak istiyorsak, PSA bitin değeri sıfır olmalıdır. OPTİON yamacından iki bit zamanlayıcının kontrolü için kullanılmıyor. Onlar altıncı ve yedinci bittir. **INTEDG** biti, RBO/INT pinin kesinti kenarının seçimi için kullanılıyor.Bu bit bir olduğu zaman, kesinti RBO/INT pinin yükselen kenarı sırasında oluşuyor, INTEDG=0 olduğu zaman kesinti iniş kenarı sırasında meydana geliyor. RBPU pini ayarlanıp/sıfırlanıp, B bağlantı noktasında Pull - Up rezistörün kapanması/açılması için kullanılıyor

INTCON yazmacı

Kesintilerin kontrolü için INTCON yazmacı kullanılıyor. INTCON yazmacı kesintilerin kontrolü için iki bit içeriyor. Resim 6.14.'te INTCON yazmacı bitlerinin kısaltmalarıyla verilmiştir.

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
GİE	EEİE	TOIE	INTE	RBIE	TOIF	INTF	BRIF

Resim 6.14. INTCON yazmacında bitlerin sıralaması

Tablo 6.3.'te INTCON yazmacının tüm bitlerin kısaltmaların anlamları ve onların işlevleri verilmiştir.

RBIF biti B kesinti noktasından RB4, RB5, RB6, RB7 pinlerinde kesinti meydana gelince ayarlanıyor. Bu pinlerin kesintilere duyarlı olmaları için, onların TRISB yazmacının kullanımıyla giriş pinleri olarak tanımlanmaları gerekiyor. INTF biti RB0/INT pininde kesinti meydana gelince ayarlanıyor. Bu kesintinin ana programa döndükten sonra yeniden meydana gelmemesi ve kesinti onarımı için alt programın çalıştırılması sırasında **INTF** bitin sıfırlandırılması gerekiyor. Bunu programcının unutmaması gerekiyor, çünkü mikrodenetimci durmadan kesinti onarım alt

programını çalıştıracak. **TOIF** biti TMR0 sayacında değerin aşması meydana gelince ayarlanıyor.

bit	kısaltma	işlev
yedinci	GIE (General Interrupt Enable)	Tüm kesinti türlerine izin veriyoruz
altıncı	EEIE (EEPROM Interrupt Enable)	EEPROM'da yazmak için kesintiye izin veriyoruz
beşinci	TOIE (TMR0 Interrupt Enable)	TMR0'ın değer aşması için kesintiye izin veriyoruz
dördüncü	INTE (Interrupt Enable)	RB0/INT pininde kesintiye izin veriyoruz 1 - izinli 0 - izinsiz
üçüncü	RBIE (Port B Interrupt Enable)	B bağlantı noktasından 7, 6, 5, 4 pinin kesintisine izin veriyoruz 1 - izinli 0 - izinsiz
ikinci	TOIF (TMR0 Interrupt Flag)	Kesinti - serbest sayacın değer aşması meydana gelmesi 1 - TMR0 değerini aşmış 0 - aşmamış
birinci	INTF (Interrupt Flag)	B bağlantı noktasında sıfırıncı pininde (RBO/INT) kesintinin meydana gelmesi 1 - kesinti var, 0 - kesinti yok
sıfırıncı	RBIF (Port B Interrupt Flag)	B bağlantı noktasından 7, 6, 5, 4 pinlerinde kesinti 1 - kesinti var, 0 - kesinti yok

Tablo 6.3. INTCON yazmacından bitlerin işlevsel açıklaması

EEIE biti EEPROM belleğinde verilerin yazılması gerekince ayarlanıyor. Verinin bellekte yazılması yaklaşık 10ms sürüyor ve bu zaman mikrodenetimci için oldukça uzun zamandır. Bilginin kaybolması meydana gelmemesi ya da mevcut programda hatanın meydana gelmemesi için EEPROM'da veri yazılması bir kesinti şekli olarak sayılıyor.

GIE (General Interrupt Enable) biti tüm kesintilerin global olarak sağlanması için kullanılıyor. Mikrodenetimci kesintiye cevap verirse, yani kesintinin onarımı için alt program çalıştırıyorsa, herhangi yeni bir kesintinin meydana gelmesini engellemek için GIE bitin sıfırlandırılması gerekiyor.

EECON yazmacı

Bu yazmaç EEPROM belleğinde okumanın ve yazılmanın kontrolü için kullanılıyor. Resim 6.15.'te EECON yazmacı gösterilmiştir.

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
/	/	/	EEIF	WRERR	WREN	WR	RD

Resim 6.15. EECON yazmacında bitlerin sıralaması

Aşağıdaki tabloda EECON yazmacından tüm bitlerin anlamları verilmiştir

bit	Kısaltma	işlev
yedinci	X	anlamı yok
altıncı	X	anlamı yok
beşinci	X	anlamı yok
dördüncü	EEIF (EEPROM Interrupt Flag)	1 - yazma başarılı tamamlanmıştır, 0 - hala okuyor
üçüncü	WRERR (Write Error)	1 - yazma sırasında hata, 0 - hata yok
ikinci	WREN (Write Enable)	1 - yazma olanağı sağlanmıştır, 0 - yazma olanağı yok
birinci	WR (Write)	1 - yazma başlatıyor, 0 - yazma başlatmıyor
sıfırıncı	RD (Read)	1 - okuma başlatıyor, 0 - okuma başlatmıyor

Tablo 6.4. EECON yazmacında bitlerin işlevsel açıklaması

RD ve **WR** bitleriyle iki seçenek,okuma ve yazma, arasından biri seçiliyor. Bu iki bit yazılımla ayarlanıyor, okumanın ya da yazmanın tamamlanmasından sonra donanımla sıfırlandırılıyor. **WREN** (Write Enable) biti EEPROM belleğinde yazılmaya izin veren (WREN=1) ya da izin vermeyen, yasaklayan (WREN=0) bittir. **WRERR** (Write Error) biti yazma sırasında hata meydana gelince ayarlanıyor. **EEIF** biti EEPROM'da yazma işleminin sona erdiğini gösteren gösterge - bitidir.

Durum yazmacı

Durum yazmacı iki bellek bankadan birini seçmek için kullanılıyor, aritmetik mantık birimim durumunu takip ediyor ve durumu sıfırlandırıyor. Resim 6.16.'da durum yazmacında bitlerin sıralabması ve onların kısaltılmaları gösterilmiştir. Bitlerin ayarlanması ve sıfırlandırılması, daha geç inceleyeceğimiz yönergelerle, yazılımla da gerçekleşebilir. Bu olanak sadece TO ve PD bitleri için geçerli değildir. Onların durumu sadece okunabilir, yazma olanağı yoktur.

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
IRP	RP1	RP0	TO	PD	Z	DC	C

Resim 6.16. Durum yazmacında bitlerin sıralaması

Tablo 6.5.'te durum yazmacından her bitin işlevi açıklanmıştır

Bit	Kısaltma	İşlev
yedinci	IRP	Dolaylı adresleme şekline sekizinci bit olarak kullanılıyor 1 - birinci banka 0 - sıfırıncı banka
6,5	RP1:RP0	Dolaysız adresleme şeklinde banka seçimi için kullanılıyorlar 01 - birinci banka 00 - sıfırıncı banka
dördüncü	TO (Time Out)	Güvenlik sayacında değer aşma göstergesi 1 - Elektrik kaynağının açılmasından, güvenlik sayacın ayarlanmasından ve bekleme moduna geçişinden sonra 0 - güvenlik sayacın değerinin aşılmasından sonra
üçüncü	PD (Power Down)	1 - elektrik kaynağının açılmasından, her sıfırlandırılmadan ve güvenlik sayacın sıfılranırılmasından sonra 0 - bekleme moduna geçildikten sonra
ikinci	Z (Zero)	Aritmetik toplama ya da çıkarma sırasında aktifleştiriliyor 1 - elde edilen sonuç sıfırdır, 0 - elde edilen sonuç sıfırdan farklıdır
birinci	DC (Digit Carry)	1 - toplama sırasına üçüncü pozisyondan dördüncü pozisyona aktarma ya da çıkarma sırasında borçlanma, 0 - aktarma yok
sıfırıncı	C (Carry)	1 - toplama ya da çıkarma sırsında sonuçtan en değerli bite aktarma ya da borçlanma, 0 - aktarma yok

Tablo 6.5. Durum yazmaçtan bitlerin işlevsel açıklaması

6.9. PIC 16f84'ün Yönergeler Kümesi

8085 mikroişlemcinin yönergeler kümesini incelerken, yönergeleri işlevlerine göre birkaç gruba ayırmıştık: aktarma yönergeleri, mantık, aritmetik yönergeler, döndürme, atlama yönergeleri vb. PIC 16f84'te bu ayırımı uygulamayacağız. Yönergele-

ri dört gruba ayıracağız: sabitlerle çalışma yönergeleri, WF (Working register - File) yönergeler, bit - yönergeler vo kontrol yönergeler.

Sabitle çalışma yönergeleri

Sabitlerle çalışma yönergeler en kolay yönergelerdir. Bu yönergelerde hedef her zaman W çalışma yazmacıdır. Örneğin, **MOVLW K** yönergesinin anlamı K sabitini W çalışma yazmacına yanaştırmak (kopyalamak) demektir. L harfi, sabit anlamına gelen İngilizce Literal sözcüğünün ilk harfinden geliyor. Bu grup yönergelere şunlar aittir: ADDLW K, SUBLW, ANDLW, IORLW, XORLW. Bir işlenen W yazmacında bulunuyor, diğer işlenen K sabitidir, sonuç ise W yazmacında yazılıyor.

Örnek 6.1:

```
SUBLW 80H          ;80H - W “W
IORLW 10010111B    ;10010111B mantıksal olarak W’nin içeriğiyle toplanıyor ve
                   ;sonuç W’de yazılıyor
```

WF yönergeler (Working registar - File yönergeler)

WF yönergeler hedef seçimi konusunda anlaşmazlık yaratabilirler. WF yönergelerden sadece **MOVWF f** yönergesinin hedefi her zaman f yazmacıdır.

Örnek 6.2:

```
MOVWF FILE         ;W’nin içeriği FILE adıyla bilinen yazmaçta
                   ;kopyalanıyor.
```

Ters yönde veri aktarma istersek, FILE yazmacından W çalılşma yazmacına doğru, o zaman yönergesi **MOVF FILE, W** kullanılıyor. Tüm diğer WF yönergelerde (ADDWF, ANDWF, SUBWF, IORWF, XORWF), FILE yazmacından sonra virgül ve belirlenmiş hedefin olup olmadığına bağlı olarak, hedef W çalışma yazmacı ya da başka bazı FILE yazmacı olabilir.

Örnekler 6.3:

```
SUBWF FILE, W      ;FILE - W → W, hedef W’dir e W.
SUBWF FILE         ;FILE - W → W. Hedef FILE yazmacıdır.
XORWF GOSTER       ;GOSTER yazmacın içeriğine ve W yazmacına dışlamalı
                   ;YADA yönergesi gerçekleşiyor ve
                   ;sonuç GOSTER yazmacında yazılıyor
```

Değişken hedefle diğer yönergeler tablo 6.6.’da verilmiştir.

D yazmacı verilmemişse, o zaman sonucun f yazmacında yazıldığını bir kez daha hatırlatalım.Sonucun, yönergede verilmiş son yazmaçta yazıldığını söyleyebiliriz.

INCF f,d	(Increase File) f yazmacının içeriği bir için artıyor ve sonuç d yazmacında yerleşiyor.
INCFSZ f,d	(Increase File Skip Zero) Sonuç sıfır elde edilirse sıradaki yönerge atlatılıyor.
DECF f,d	(Decrease File) f yazmacının içeriği bir için azalıyor ve sonuç d yazmacında yerleşiyor.
DECFSZ f,d;	(Decrease File Skip Zero) Sıfır sonuç elde edilirse sıradaki yönerge atlatılıyor.
COMF f,d	(Complement File) f yazmacının içeriği tümleşiyor ve sonuç d yazmacında yazdırılıyor.
SWAP f,d	f yazmacının 4 daha yüksek değerli biti f yazmacının 4 daha alçak değerli bitlerle yer değiştiriyor ve sonuç d yazmacında yazdırılıyor.
RLF f,d	(Rotate Left File) f yazmacının bitleri, carry bayrağın aracılığıyla, bir yer için sola döndürülüyor ve sonuç d yazmacında yerleşiyor.
RRF f,d	(Rotate Right File) f yazmacının bitleri, carry bayrağın aracılığıyla, bir yer için sağa döndürülüyor ve sonuç d yazmacında yerleşiyor.

Tablo 6.6. Değişken hedefli yönergeler

Bit - yönergeler

Bit - yönergelerle bazı yazmaçtan bir bitin ayarlanması ve sıfırlandırılması yapılıyor. Bundan dolayı, yönergede yazmacın ve yazmaçtaki bitin sıra numarasının belirlenmesi gerekiyor.

Örnek 6.4:
BSF FILE, 3;(Bit Test File) FILE yazmacından üçüncü bit ayarlanıyor

Bit - yönergeler grubuna şu yönergeler aittir:

BCF Bit Clear File (Bitin sıfırlandırılması)

BSF Bit Set File (Bitin ayarlanması)

BTFSC Bit Test File Skip if Clear (Bit sıfır ise, sıradaki yönergeyi atlat.)

BTFSS Bit Test File Skip if Set (Bit bire eşitse, sıradaki yönergeyi atlat.)

Kontrol yönergeler

Kontrol yönergeler programın akışını değiştirebiliyorlar. Sıradaki yönergenin çalıştırılması yerine, alt program çağrılabilir ya da bazı bitin veya bayrağın mantıksal durumuna bağlı olarak koşullu ya da koşulsuz atlamalar gerçekleşebilir.

yönerge	yorum	örnek
CLRF f	yazmacı temizle (sıfırlandır)	CLRF PORTA
CLRW	W çalışma yazmacını sıfırlandır	/
CALL address	Alt program çağır (address alt programın başlangıcıdır)	CALL DELAY
GOTO address	Başlangıç adresi *address* olan yere atla	GOTO START
RETFIE	Kesinti onarım alt programından geri dönüş	/
RETLW k	Çalışma yazmacında sabitin girilmeisyle alt programdan geri dönüş	RETLW 00110110B
RETURN	Alt programdan geri dönmek	/
SLEEP	Bekleme moduna (düzenine) geç	/

Tablo 6.7. Kontrol yönergeleri

6.10. PIC16f84 Mikrodenetimcinin Bağlanması

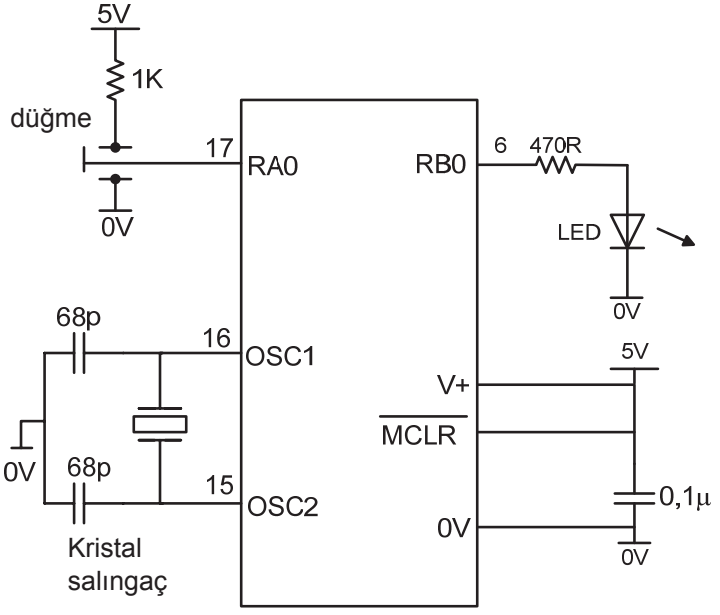
PIC16f84 mikrodenetimcinin işletim yönetiminde geniş kullanımı olduğunu önceki derslerde vurgulamıştık. Ancak, başlangıç için çok basit bir örnek açıklayacağız. **A bağlantı noktasının sıfırıncı pinide düğme (tuş) yerleştiriyoruz, B bağlantı noktasının sıfırıncı pininde led diyod takıyoruz. Düğmeye basılınca, LED diyodu 1 saniye yanacak, ondan sonra sönecek**. Önce mikrodenetimcinin bağlanma şeklini açıklayacağız, ondan sonra bu örnek için programın yazılmasını açıklayacağız.

Resim 6.17.'de bu örnek için gereken donanım gösterilmiştir. Kristal salıngaç ve yoğunlaştırıcı (kondansatör) mikrodenetimciye dijit palsı vermek için gerekiyor. 68gF'lık iki yoğunlaştırıcı sisteme denge sağlamaları gerekiyor, yani tek yönlü bileşenleri gidermeleri gerekiyor. Elektrik aygıtın tasarlanması sırasında, mikrodenetimcinin dijit palsı aldığı hatlarda hışırtının meydana gelmemesi için salıngaç mümkün olduğu kadar mikrodenetimciye yakın yerleştiriliyor. Denetimcinin elektrik kaynağı ve tablo arasında yerleştirilen yoğunlaştırıcının da buna benzer işlevi var. $\overline{MCLR}$ sıfır-

landırma pinin yüksek seviyede ayarlandığını görüyoruz. Böylece mikrodenetimcinin bu pinin aracılığıyla sıfırlandırma olanağı engellenmiştir.

Mikrodenetimicin pinlerinde çıkış elektrik cereyanın yüksekliği 20mA - 25mA arasındadır ve bu güç, led diyodun ya da daha küçük bir rölenin tetiklenmesi için yeterlidir.

Daha büyük yükler için, tek yönlü ceryan akımı için transistörler ve alternatif ceryan akımı için triyak kullanarak, çıkışta ceryan akımının güçlendirilmesi gerekiyor. LED diyodun yanması (ışıklanması) için anodun katottan daha yüksek potansiyelde olması gerekiyor. LED diyodun, hiçbir hasar yaşanmadan, en yüksek güçle yanması için, onun doğru şekilde bağlanması gerekiyor. Bu amaçla, diyodla dizisel olarak rezistör (direnç elemanı) bağlanıyor. Dirençin değeri, diyotta üreticiler tarafından tafsiye edilmiş en yüksek ceryan akımın geçebilmesi için hesaplanıyor.



Resim 6.17. PIC16f84'ün düğme ve LED diyodla bağlanması

Diyodlar mikrodenetimcilerle farklı şekilde bağlanabiliyorlar. Resim 6.17'de gösterilmiş durumda, LED diyodu katodu tabloya bağlanmış bulunuyor ve diyodun yanması için RB0 çıkış pininde mantıksal sıfır gerekiyor. Bu şekilde anod, katottan daha yüksek potansiyelde bulunacak. Diyod ters tarafta dönük olursa, anodun tablo tarafında bulunmasıyla, o zaman diyod mantıksal sıfır durumunda yanacak. Led diyodu, tablonun dışında 5V'luk elektrik kaynağıyla da bağlanabilir.

Genelde mikrodenetimcinin girişinde anahtar (switch) bağlanıyor. Bağlanma, mikrodenetimcide giriş ceryan akımını ayarlayabilen **pull up direnç** elemanı aracılı-

ğıyla yapılıyor. Anahtar kapalı olduğu zaman, mikrodenetimcinin giriş pinine mantıksal sıfır gönderiliyor, çünkü o zaman pin tabloya duğrudan bağlanıyor. Anahtar açık olunca pinin giriş gerilimi yüksek seviyede bulunuyor, yani pin, elektrik kaynağın gerilimiyle, koruyucu direnç (pull up rezistör) aracılığıyla bağlanıyor. Anahtarın iki dengeli (stabil) durumu var, açık ya da kapalı. Anahtardan farklı olarak, klavyedeki tuşların bir dengeli ve bir de geçici durumları olabilir. Kullanıcı bir tuşa bastığı zaman, tuş mikrodenetimciye, yani mikroişlemciye bilgi gönderiyor. Ancak tuş, basılık halde 1 saniye ya da daha az zaman kalıyor, ondan sonra tuş (düğme) serbest bırakılıyor. Bir saniyelik zaman mikrodenetimci için çok uzun bir süredir. Mikrodenetimci bir saniye içinde düğmeyi çok kez kontrol edebilir. Bu yüzden, düğmenin iki kontrolu arasında **gecikme zamanı** girmelidir ve bu şekilde düğmenin bir kez basılmasını mikrodenetimci tarafından birçok kez saymasından kaçınılıyor.

6.11. PIC16f84'ün Programlanması

PIC16f84 mikrodenetimcinin donanımını ve yönergeler kümesini tanıdıktan sonra, PIC16f84'ün programlama sürecini açıklayacağız. Mikrodenetimcinin programlanması sırasında, uyması gereken kesin olarak tanımlanmış kurallar uygulanıyor. Programın incelemesini basitleştirmek için programı birkaç bölüme ayıracağız:

- Değişkenlerin ve sabitlerin tanımlanması.
- Mikrodenetimcinin bildirimi ve program başlangıcı.
- Mikrodenetimcinin yapılandırması
- Gecikme alt programı
- Bağlanma noktaların ilklendirmesi (hazırlanması)
- Ana programın yazılması

Metinin devamında bütün bu bölümler açıklanacak.

Değişkenlerln ve sabitlerin tanımlanması

Değişkenlerin ve sabitlerin tanımlanması için **EQU çevirici emri** kullanılıyor. Değişkenler ve sabitler RAM belleğinin genel ve özel amaçlı yazmaçlarında bulunuyorlar. EQU emrinde bu yazmaçlardan hepsine sembolik isim veriliyor. Bildirimde yazmacın adın dışında onun RAM belleğinde adresi de veriliyor. Örneğin, NIVO EQU 0DH emriyle 0DN adresli bellek yerine NIVO sembolik ismi verilyor. Bu şekilde verilen yerde bulunan değişkene NIVO ismini kullanarak ulaşıyoruz. Bu şekil programcı için bellek yerin onaltılı adresin aklına tutması yerine NIVO ismin ezberlemesi çok daha kolaydır. Aşağıda LED diyodun düğmeyle yanması ve kapanması için programda kullanılan tüm yazmaçlar tanımlanmıştır:

```
TMRO        EQU     O1H     ;Serbest sayaç 01H adresli yazmaçtır.
STATUS      EQU     O3H     ;Durum yazmacı 03H adresli yazmaçtır.
PORTA       EQU     O5H     ;A bağlantı noktası 05H adresli yazmaçtır.
PORTB       EQU     O6H     ;B bağlantı noktası 06H adresli yazmaçtır.
TRISA       EQU     85H     ;A bağlantı noktasından giriş çıkış pinlerin
                            ;seçimi için yazmaç 85H adresli yazmaçtır.
TRISB       EQU     86H     ;A bağlantı noktasından giriş çıkış pinlerin
                            ;seçimi için yazmaç 86H adresli yazmaçtır.
OPTION_R    EQU     81H     ;Sayaçları kontrol eden yazmacın 81H adresi
                            ;vardır.
COUNT       EQU     OCH     ;COUNT genel amaçlı yazmacın 0CH adresi
                            ,vardır.
```

Verilen tüm yazmaçlardan sadece COUNT genel amaçlı yazmaçtır. Özel amaçlı yazmaçların isimlerini koruduklarını görüyoruz, genel yazmaçların isimlerini ise amaçlarına bağlı olarak programcı seçiyor. Ayrıca, özel amaçlı yazmaçların adresleri, resim 6.3.'te göstrerilmiş adreslere uyumludur ki böyle bir şey doğaldır çünkü yazmaçların hepsi için mikrodenetimcinin özel mantığı vardır.

Mikrodenetimcinin bildirimi (deklarasyonu) ve programın başlangıcı

Farklı PIC mikrodenetimci türleri vardır. Onlardan birkaçını sayabiliriz: PIC1f83, PIC16f84, PICf872, PIC16C923, PIC17C42A, PIC18C242 ve başka. Onlar aralarında belleklerin ve sayaçların büyüklüğüne göre, pinleri sayısına ve diğer özelliklere göre farklıdır. Programın yazılması sırasında mikrodenetimcinin türü de yazılmalıdır. Bunun için **LİST emri** (direktifi) kullanılıyor.

ORG emriyle program belleğinde ana programın başlangıcı belirleniyor.

```
LIST    P=16f84     ;Mikrodenetimci türünü seçiyoruz.
ORG     0           ;Ana programın başlangıç adresi 0'dır.
GOTO    START       ;START sembolik ismi olan yere atla.
```

Mikrodenetimcinin yapılandırması

Mikrodenetimcini seçiminden sonra, onun yapılandırmasına geçebiliriz. **Yapılandırma sözü** 14 - bitlidir ve onunla salıngaç, güvenlik sayacın kullanımı ve elektrik kaynağın açılması için zamanlayıcı tanımlanıyor. 13.ncü yerden 4.ncü yere ka-

dar bulunan bitlere program belleğin kod güvenlik bitleri denir. Üçüncü bit PWRTE ile işaretleniyor ve anlamı elektrik kaynağın açılma zamanlayıcısı olan Power - Up Timer Enable sözlerinin kısaltılmasıdır. Bu zamanlayıcı 72ms'lik gecikme meydana getiriyor, yani bu süre içinde elektrik kaynağının denge sağlamasına kadar denetimciyi sıfırlandırılmış durumda tutuyor. Bu bitin değeri sıfır olduğu zaman, zamanlayıcı devre dışındadır, bir olduğu zaman ise zamanlayıcı etkindir. İkinci bit WDTE ile işaretleniyor ve güvenlik sayacın etkinleştirilmesi anlamına gelen Watchdog Timer Enable sözlerinin kısaltmasıdır. Bu bitin değeri bir olunca, o zaman güvenlik sayacı çalışıyor, sıfır olunca Watchdog zamanlayıcısı devre dışındadır. Son iki bit salıngaç türünü belirliyor ve onların durumuna göre **salıngaç türü** tablo 6.8.'de verilmiştir.

00	RC salıngaç
01	HS salıngaç
10	XT salıngaç
11	LP salıngaç

Tablo 6.8.

Mikrodenetimcinin yapılandırması için kullanılan emir **_CONFIG** emridir. Aşağıda düğmenin basılmasıyla LED diyodun yanıp sönmesi için programımıza ilişkin yapılandırma gösterilmiştir.

```
_CONFIG H'3FF0'        ;LP salıngaçı seçilmiştir, güvenlik sayacı
                       ;devre dışıdır, elektrik kaynağın açılması için
                       ;zamanlayıcı çalışıyor, kod güvenliği yoktur.
```

Gecikme alt programları

Dijit pals kaynağı olarak kristal salıngaç kullanıldığı zaman, TMR0 serbest sayacı her dördüncü tetiği sayıyor. 32KHz=32768Hz frekanslı salıngaç kullanıyorsak, zamanlayıcının hızı 32 KHz'in bir çeyreği olacak, yani 8KHz=8192Hz olacak. 1 saniyelik zaman elde etmemiz için TMR0 sayacın 8192 tetiğin sayması gerekiyor. Bu da oldukça çoktur. Bu yüzden, bu durumda OPTİCON yazmacının ön ölçekleyicisini kullanıyoruz. Ön ölçekleyici 111'e eşitse, o zaman 8KHz frekanslı dijit palsını 256 ile bölüyoruz ve 32 değerini elde ediyoruz. Demek ki bir saniyelik gecikme elde etmemiz için zamanlayıcının 32 kez sayması gerekiyor. Metnin devamında 1 saniyelik gecikme için örnek verilmiştir.

```
GECIKME   CLR    TMRO          ;TMRO=000000000B

LOOP      MOVF   TMRO, W       ;TMRO W'ye kopyalanıyor

SUBLW     32.
```

```
BTFSS     STATUS, ZEROBIT

GOTO      LOOP
RETLW     0
```

Programın başlagıcında, saymanın sıfırdan başlaması için TRM0 sayacı sıfırlanıyor. Bu program sıralaması sürdüğü zaman içerisinde, sayacın değeri devamlı artıyor. TMR0 sayacın sayması 32 değerine ulaşıncaya kadar sürüyor. TMR0 sayacının 32 değerine ulaşıp ulaşmadığını, içeriğinden 32 sayısını çıkararak kontrol ediyoruz. BTFSS yönergesiyle ZERO bayrağını incelememiz gerekiyor. Çıkarma işleminden sonuç 0 elde edersek, ZERO bayrağının değeri 1 (set) olacak. O zaman GOTO yönergesi atlatılıyor ve sıradaki RETURN (alt programın kapatılması) yönergesine gidiyoruz. Çıkarma sonucu 0'a eşit değilse, bayrağın değeri 0 olacak. O zaman GOTO yönergesi çalıştırılacak, yani LOOP adlı döngü tekrarlanıyor.

5 saniyelik gecikme nasıl yapabiliriz? Bir saniye için 32 kez saymamız gerekiyorsa, 5 saniye için 160 saymak gerekecek. Örnek, önceki örnek gibi aynı olacak, sadece çıkarma yönergesinde SUBLW 160 duracak.

Yazmaçların ilklenmesi (başlatılması)

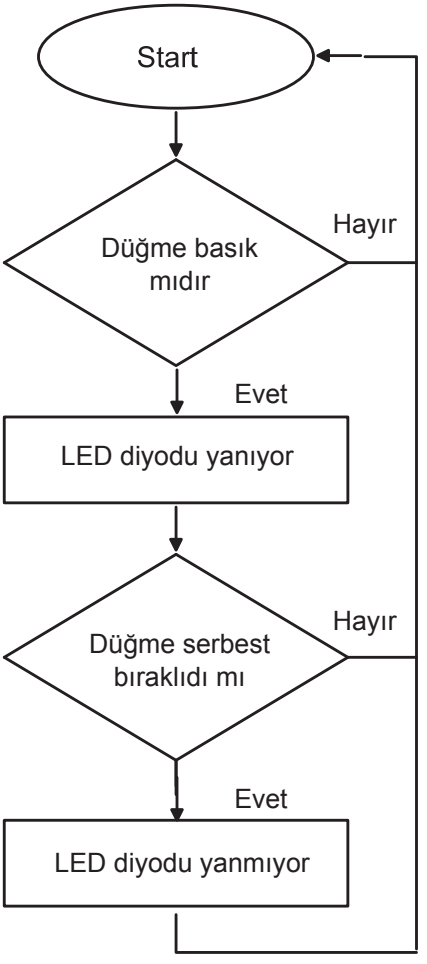
Yazmaçların ilklenmesi, alt programların ve ana programın başarılı çalıştırılmaları gerekçesiyle, yazmaçları başlangıç durumuna ayarlamak demektir. Önce **A ve B bağlantı noktaların pinlerinde bitlerin hareket etme yönünün tanımlanması** gerekiyor. Hatırlayalım, bu işlevi TRISA ve TRISB yazmaçları gerçekleştiriyor. PORTA ve PORTB yazmaçlarının bazı eski içerikleri mikrodenetimcinin çalışmasında hataya yol açmaması için, PORTA ve PORTB yazmaçları sıfırlanmalı, yani silinmelidir. **Gecikme zamanın programlanması sırasında,** değeri 111 olan **ön ölçekleyici kullandık**. Bu amaçla OPTION yazmacının çağrılması gerekiyor ve Bu yazmacın üç en az ağırlıklı bitleri 111 değerine ayarlanmalıdır. Devamda, bağlantı noktaların ve yazmaçların ilklenmesini gerçekleştiren program sıralaması verilmiştir.

```
START   BSF     STATUS,5        ;Birinci bellek bankası çağrılıyor.

        MOVLW   B'00011111'     ;Veri, W çalışma yazmacı aracılığıyla
                                ;TRISA yazmacına giriliyor.

        MOVWF   TRISA           ;A bağlantı noktanın tüm pinleri giriş pinleridir

        MOVLW   B'000000000'    ;Veri, W çalışma yazmacı aracılığıyla
                                ;TRISB yazmacında yerleştiriliyor.

        MOVWF   TRISB           ;B bağlantı noktanın tüm pinleri çıkış pinleridir.
```

```
MOVLW   B'00000111'  ;Veri, W çalışma yazmacı
                     ;aracılığıyla OPTION_R yazmacına
                     ;yerleştiriliyor.

MOVWF   OPTION_R     ;Ön ölçekleyici=111.

BCF     STATUS,5     ;sıfırıncı bellek bankası
                     ;çağrılıyor.

CLRF    PORTA        ;PORTA yazmacı sıfırlanıyor.

CLRF    PORTB        ;PORTB yazmacı sıfırlanıyor.
```

Ana program

Resim 6.18.'de LED diyodun düğmeyle açılma programın algoritması gösterilmiştir.



Resim 6.18. LED diyodun düğmeyle açılma programının blok - diyagramı

```
BEGIN     BTFSC PORTA,0    ;Düğmenin basık olup olmadığını kontrol
                           ; ediyoruz. Eğer basıksa sıradaki yönerge
                           ;atlatılıyor.

          GOTO BEGIN       ;Eğer düğme basılmış değilse BEGİN adresine
                           ;geri dönüyoruz ve tekrar kontrol ediyoruz.

          BSF PORTB, 0     ;LED diyodu açılıyor.

RELEASE   BTFSS PORTA, 0   ;Düğme serbest bırakılmışsa sıradaki yönerge
                           ;atlatılıyor.

          GOTO RELEASE     ;Düğmenin serbest bırakıldığını yeniden kontrol
                           ;etmek için RETURN adresine geri dönüyoruz.

          CALL GECIKME     ;Bir saniyelik gecikme ayarlıyoruz.

          BCF PORTB,0      ;LED diyodu kapanıyor

          GOTO BEGIN       ;Süreci tekrarlıyoruz

END
```

Sonuçlar

Mikrodenetimci tümleşik devre şekilli mikrobilgisayardır. Mikrodenetimcinin içeriğinde bir bilgisayarda olan her şeyi var: mikroişlemci, bellekler ve veri aktarımı için bakırlı hatlar.

PIC kısaltamın iki anlamı vardır. PIC, Programmable Intelligent Computer sözlerinden kısaltma olabilir. Ancak, bazı kitaplarda PIC, Programmable Interrupt Controller sözlerinin kısaltması olarak da tanımlanıyor ve anlamı programlanabilen kesintiler denetimcisi'dir.

PIC 16f84 üç farklı modta (düzende) çalışabiliyor: program modu, çalışma modu ve bekleme modu. Program modunda, program belleğinde kullanıcı programı giriliyor. Çalışma modunda, mikrodenetimci verileri işletiyor ve farklı süreçleri yönetiyor. Mikrodenetimici bekleme ya da uyuma (sleep) moduna elektrik enerjini tüketimin tasarrufu yapılması amacıyla ayarlanıyor.

PIC16f84 üç tür bellek içeriyor: RAM, EEPROM ve program belleği.

EEPROM belleği programın çalıştırılmasıyla elde edilen verileri, ancak aynı zamanda işletim için önemli olan sabitleri saklıyor. EEPOM belleğinden okumak ve EEPROM belleğinde yazmak için üç özel yazmaç kullanılıyor. EEADR yazmacı okunduğu ve yazı yazıldığı bellek yerin adresini saklıyor. EEDATA yazılması gereken ya da okunan veriyi saklıyor. EECON 5 - bitli yazmaçtır ve EEPROM belleğinden okunmasıya ya da yazılmasıya ilgili kontrol bitleri içeriyor.

RAM belleği iki bankaya ayrılıyor ve onlar banka 0 ve banka 1 olarak işaretleniyor. Banka 0'a (sıfırıncı bankaya) adresi sıfırla başlayan belek konumlara aittir (00H'dan 4F'e kadar), banka 1'e (birinci bankaya) adresleri birle başlayan bellek yerleri aittir (80H'dan CFH'ye kadar).

PIC 16f84 mikrodenetimcisinde, genel ve özel amaçlı yazmaçlar RAM belleğinde bulunuyor. RAM belleğinden her iki bankanın ilk 12 bellek yeri, özel amaçlı yazmaçlardır.

Dış dünyayla iletişim için PIC 16f84 mikrodenetimcisi, A ve B bağlantı noktaları pinlerini kullanıyor. B bağlantı noktası sekiz pinlidir, A bağlantı noktası ise beş pinlidir. Onlar giriş pinleri ve çıkış pinleri olabilir, ancak iki yönlü olamazlar. A ve B bağlantı noktaların girişli ya da çıkışlı olmaları, TRISA ve TRISB yazmaçlardaki bitlere bağlıdır.

PIC16f84 mikrodenetimci sıkça zamanlayıcı ya da sayaç olarak kullanılıyor. Mikrodenetimcinin zamanlayıcısı TMR0 kısaltmasıyla işaret ediliyor ve aslında timer0'dan kısaltmadır. TMR0 8 - bitli yazmaçtır ve 00H'den FFH'ye kadar sayabilir. Alabildiği en yüksek değere ulaşınca, zamanlayıcı yeniden sıfırdan saymaya başlıyor. Zamanlayıcı olarak kullanıldığı zaman, mikrodenetimci kristal salıngaç olarak bağlanıyor. TMR0 yazmacı, kristal salıngaçın dijit atışları dışında, RA4/TOCKI pinin durum değişikliklerini de sayabilir ve o zaman mikrodenetimcinin dış rastgele değişikliklerin sayacı olduğunu diyoruz.

PIC 16f84 mikrodenetimcinin dört tür kesintisi olabilir: EEPROM belleğinde veri yazılmasının sonu, serbest sayacın değerin aşmasından kaynaklanan kesinti, B bağlantı noktasından yedinci, altıncı, beşinci ve dördüncü pinlerin durum değişikliği sırasında meydana gelen kesinti ve RB0/INT pinin dış kesintisi.

Her kesinti türü için iki tür bit vardır: Interrupt Enable bit (IE) ve Interrupt Flag bit (IF). Birincisi, kesintinin etkinleşmesi için yazılım yoluyla yüksek seviyeye ayarlanıyor. Kesinti bayraklar donanım yoluyla ayarlanıyor. Mikrodenetimci kullanıcıyı kesinti meydana geldiğini bildirmek için bayrakları yüksek seviyede ayarlıyor.

Sayaçların ve zamanayıcıların kontrolü için OPTION yazmacı kullanılıyor. OPTION yazmacı aracılığıyla sayaç türü, atış kaynağı ve frekans bölücüsü (ön ölçekleyici) tanımlanıyor.

PIC 16f84 mikrodenetimcide sabitlerle çalışma yönergeleri LW harfleriyle sonuçlanıyor. Bir işlenen W çalışma yazmacında bulunuyor, diğer işlenen ise sabitin ta kendisidir. Çalıştırılan yönergenin sonucu W yazmacında yazılıyor.

MOVWF f veri aktarma yönergesidir. Kaynak W çalışma yazmacıdı, hedef ise f yazmacıdır. Ters yönde, FILE yazmacında W çalışma yazmacına doğru veri aktarımı için MOVF FILE, W yönergesi kullanılıyor

Bit - yönergesiyle herhangi bir yazmaçtan bir bitin ayarlanması ya da sıfrılanması yapılıyor. Yönergede, yazmaç ve bu yazmaçta bitin sıra numarası verilmelidir.

Değişkenlerin ve sabitlerin tanımlanması için EQU çevirici emri kullanılıyor. EQU emrinde yazmacın ismi dışında onun RAM belleğinde adresi de veriliyor. LIST emriyle mikrodenetimcinin türü belirleniyor. ORG emriyle program belleğinde ana programın başlangıcı belirleniyor. Yapılandırma kelimesi 14 - bitlidir ve onunla salıngaç, güvenlik sayacın ve elektrik kaynağının açılma zamanlayıcının kullanımı tanımlıyor.

PIC 16f84 mikrodenetimcisi LED Diyodlara, ikili şeklinde birkaç milisaniye süren sinyaller gönderiyor. Gecikme programları tetikleme sinyallerin süresinin devam edilmesi için kullanılıyorlar. Gecikme programları mikrodenetimciyi düğmelerle (tuşlarla) bağlamak için de kullanılıyor. Bu şekilde düğmelere yeniden başlamadan önce serbest bırakılmaları için yeterli zaman veriliyor.

Yazmaçların ilklenmesi onları başlangıç durumunda ayarlamak için kullanılıyor. Bitlerin, A ve B bağlantı noktaların pinlerinden hareket etme yönünü TRISA ve TRISB yazmaçları belirliyor. PORTA ve PORTB yazmaçların eski içerikleri mikrodenetimcinin çalışmasında hataya yol açmasın diye, bu yazmaçların sıfırlanmaları yani silinmeleri gerekiyor.

Sorular ve ödevler

1. PIC 16f84'ün pratik kullanımı için birkaç örnek say!

2. PIC16f84 üç farklı mod'ta çalışabilir: program, çalışma ve bekleme modu. Her mod için anlamını açıkla!

3. PIC 16f84 mikrodenetimcinin hangi bölümünde özel amaçlı yazmaçlar bulunuyor?

4. PIC 16f84'ün program belleğinde girebilen en büyük kullanıcı programın büyüklüğü ne kadardır?

5. Bir mikrodenetimcinin programlanması için neyin gerektiğini açıkla!

6. EEPROM belleğinde hangi veriler saklanıyor?

7. Sıfırıncı ve birinci bellek bankalarında konumların adresleri arasında fark nedir?

8. PIC 16f84'te dolaysız ve dolaylı adreslemede bellek yerinin adresi nerede bulunuyor?

9. PIC 16f84 mikrodenetimcinin A ve B bağlantı noktalarını açıkla! Onların ne için kullanımları olduğunu açıkla?

10. OSC1 ve OSC2 pinleri ne için kullanılıyor?

11. TMR0 yazmacının ne işlevi olduğunu açıkla?

12. RA4/TOCKI pinin çift işlevi var. Açıkla!

13. Ön ölçekleyici nedir ve onun ne işlevi vardır?

14. Güvenlik sayacın işlevini açıkla!

15. PIC16f84 mikrodenetimcinin 4 kesinti türü vardır. Onları say!

16. TRISA ve TRISB yazmaçlarının işlevleri ne olduğunu açıkla?

17. Kesintiler kontrolü için hangi yazmaç kullanılıyor?

18. Kontrol yazmaçlarında bitlerin donanım yoluyla ve yazılım yoluya sıfırlamak terimlerini açıkla?

19. INTEDG biti Interrupt Edge sözlerinin kısaltmasıdır. Bu bitin ne için kullanıldığını açıkla!

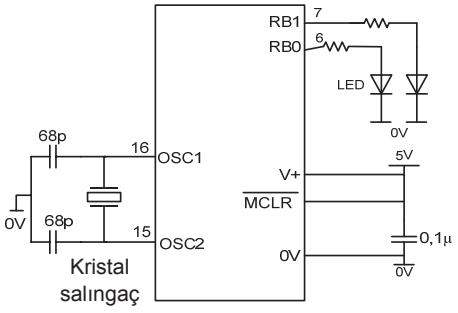
20. EEIF biti EEPROM Interrupt Flag sözlerinin kısaltmasıdır. Bu bitin nasıl ayarlandığını açıkla.

21. MOVF FILE ve MOWF FILE yönergelerini kıyasla!

22. Verilen yönergeleri yorumla:
    CLRF
    BCF
    INCF

23. Verilen yönergeleri yorumla:
    RLF COUNT,W
    DECSZ COUNT
    ADDWF COUNT
    BTFSS PORTB,3
    INCFSZ TMRO

24. B bağlantı nokatsının sıfırıncı ve birinci pininde, anodları pinlere doğru yönelmiş iki LED diyodu bağlanmıştır. Verilen program sıralamasının etkisini açıkla:




```
BEGIN BSF PORTB, 0
      BCF PORTB, 1
      CALL DELAY5
      BCF PORTB, 0
      BSF PORTB, 1
      GOTO BEGIN
```

25. PIC 16F84 mikrodenetimci, frekansı f=200 KHz olan kristal salıngaca bağlanırsa ve ön ölçekleyicinin 011 değeri varsa, sayacın çalışma frekansı ne kadardır? 0,5 saniyelik gecikme elde etmek için kaç atışın sayılması gerekiyor?

26. Düğmenin iki kez art arda basılması arasında gecikme zamanının eklenmesi neden gereklidir?

27. EQU emri ne için kullanılıyor?

28. Mikrodenetimci türünün seçilmesi için hangi emir kullanılıyor?

29. Yapılandırma kelimesi ile ne tanımlanıyor?

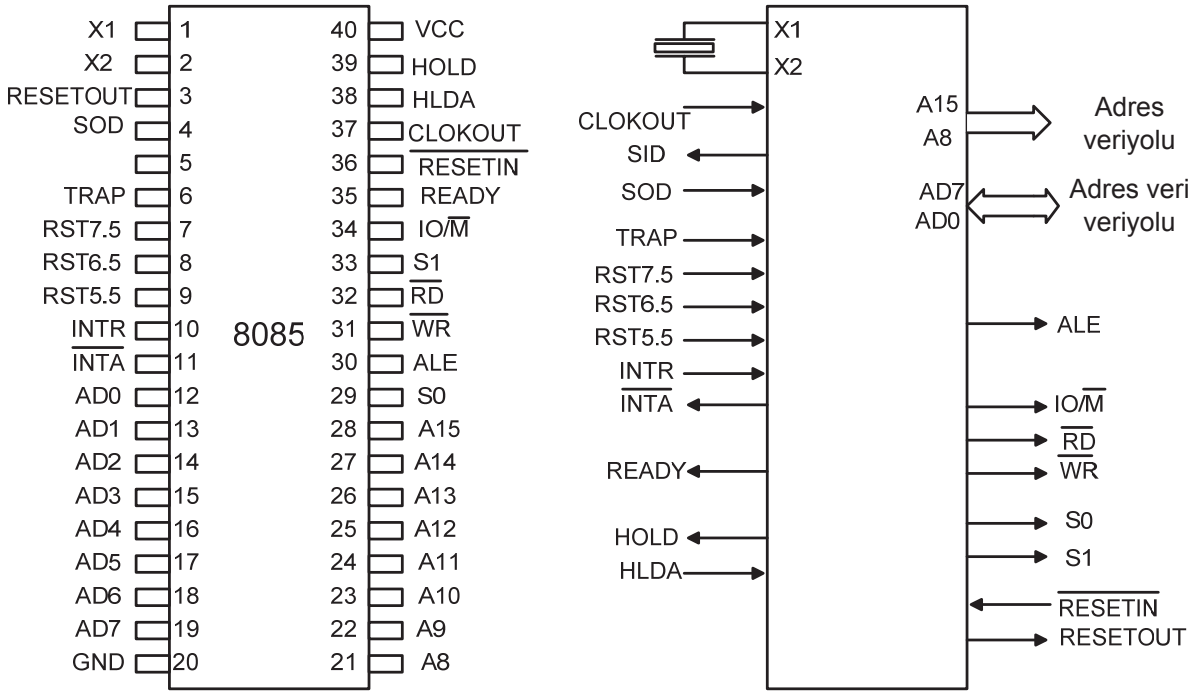
30. B bağlantı noktasının sekiz pininde, anodları pinlere doğru yönelik sekiz LED diyodu bağlanıktır. Aşağıdaki program sıralamasının etkisi ne olduğunu açıkla:

```
BASLANGIC  MOVLW 1111000B
           MOVWF PORTB
           CALL DELAY 5
            MOVLW 00001111B
            MOVWF PORTB
            CALL DELAY 5
            GOTO BEGIN
```

7. 8 - bitli Mikroişlemciler (8085 Mikroişlemci)

7.1. 8085 Mikroişlemcinin Pin - Diyagramı

Resim 7.1.'de 8085 mikroişlemcinin pin diyagramı tanımlanmıştır. Pine giren oklar giriş sinyalleridir, çıkan oklar ise çıkış sinyalleridir (mikroişlemciden bilgisayarın diğer bölümlerine doğru).



Resim 7.1. 8085 mikroişlemcinin pin diyagramı

Bazı pinler yüksek seviyede aktiftir, bazıları ise alçak seviyede aktiftir. Alçak seviyede aktif olan pinler, kısaltmaları üzerinde olumsuzluk işaretiyle işaretlen-

miştir. Bu iki durum dışında, yüksek empedans olarak adlandırılan üçüncü durum da olabilir. Yüksek empedans durumunda, pin hiçbir tetiklemeye karşılık vermiyor, yani pin engelli (bloke) durumundadır.

A_{15} - A_8 Adres veriyolu
Adres veriyoluyla16 - bitik adresin sekiz daha değerli bit aktarılıyor. Adres veriyolu **HOLD** durumu ve **RESET** yönergesi sırasında yüksek empedans durumunda bulunuyor**.**

AD_{0-7} Çoğullamalı adres/veri veriyolu Birinci atış sırasında veriyolu ilk 8 en değerli adres bitinin aktarımı için kullanılıyor, makine döngüsünün ikinci ve üçüncü atışında ise veriler gönderiliyor.

ALE Address Latch Enable - AD (adres - veri) veriyolu için kontrol sinyali Eğer ALE=1 ise, o zaman adres - veriyolu adres bitleri aktarıyor. Eğer ALE=0 ise, o zaman AD veriyolu veri pinleri aktarıyor.

S0,S1, IO/$\overline{M}$ Durum - pinleri
Makine döngüsü terimi altında, bir baytlık bilginin mikroişlemciden belleğe ya da ters yönde aktarılması için gereken zaman tanımlanıyor. Her makine döngünün başlangıcında durum - sinyalleri aktifleştiriliyor. Üç durum hattı aracılığıyla, mikroişlemci nasıl makine döngüsünün gerçekleşeceği hakkında bilgi veriyor. Altı farklı makine döngüsü ayırıyoruz.

IO/$\overline{M}$	S1	S0	Makine döngüsü
0	0	1	Bellekte yazdırma
0	1	0	Bellekten okuma
1	0	1	Çıkış aygıtına yazdırma
1	1	0	Giriş aygıtından okuma
0	1	1	İşlem kodunun aktarımı
1	1	1	Kesintinin işletilmesi

Durum hatları makine döngüsünün başlangıcında aktifleştiriliyor ve döngünün sonuna kadar aktif durumda kalıyorlar. Yönerge döngüsü, yönergenin tamamıyla gerçekleşmesi için aktarılması gereken baytların sayısına bağlı olarak, üç ya da dört makine döngüsünden oluşuyor.

$\overline{RD}$ Okuma pini (Read)
Mikroişlemci, bellekten ya da dış aygıtlardan veri okuması gerektiği zaman onlara $\overline{RD}$ kontrol sinyalı gönderiyor. Bu pin mantıksal sıfır durumunda aktiftir.

$\overline{WR}$ Yazma pini (Write)
RD'den ters anlamı var ve mikroişlemcinin belleğe ya da dış aygıta verinin yazılması gerekince aktifleştiriliyor

READY READY kontrol sinyalı yüksek seviyede olduğu zaman, belleğin ya da dış aygıt verinin okunması ya da yazılması için hazır oldukları demektir. READY sinyalı alçak seviyede ise, o zaman merkez mikroişlemcisi READY sinyalın bellekte ya da dış aygıtta okunmanın ya da yazılmanın tamamlanmasından önce yüksek seviyeye çıkması için tüm sayı atışların olmasını bekleyecek

HOLD Belleğe doğrudan erişim arama pini Bu giriş, herhangi bir dış aygıtı, belleğe doğrudan erişim (Direct Memory Access) aradığı zaman aktifleştiriliyor (yüksek seviyeye çıkıyor). O zaman merkezi işlemci hükümdar değil, köle oluyor. Merkezi işlemci veriyollarını veri değişimi için kullanamıyor, ancak bu zamanı bazı iç çalışmayı tamamlamak için kullanabilir. Böyle durumun sürdüğü zaman içinde adres, veri, RD, WR ve IO/M pinleri yüksek empedans durumuda bulunuyorlar

HLDA Hold Acknowledge - Belleğe doğrudan erişim için izin pini Mikroişlemci belleğe doğrudan erişim aramasını kabul etmiş ve HLDA sinyali yüksek seviyeye çıktığı zaman, mikroişlemci verilerin çok hızlı aktarımından dolayı, bazı dış aygıtın veriyollarını serbestleştirecek.

INTR Kesinti araması (Interrupt Request)
Her yönerge döngünün tamamlanmasından sonra, mikroişlemci INTR pinin aktif olup olmadığını kontrol ediyor. Bu pin mikroişlemci için giriş pinidir.
Sıfırlama adresi, aktif kesintinin onarımı için başlatılan alt programın başladığı bellek yerinin adresini tanımlıyor. INTR'nin sabit sıfırlama adresi yoktur. Mikroişlemcinin hangi kesinti alt programın çalıştıracağı, kesinti arayan dış aygıtın gönderdiği kesinti vektörüne bağlıdır. Kesinti vektörlerin sayısı, kesinti başlatabilen dış aygıtların sayısına eşittir.

$\overline{INTA}$ Kesinti için izin pini (Interrupt Acknowledge)
Bu pin, mikroişlemcinin kesinti aramasını kabul edince aktifleştiriliyor. Bu arada, 8259 kesinti yonganın ya da başka bir kesinti bağlantı noktasını aktifleştiriliyor.

TRAP En yüksek öncüllüklü kesinti.
TRAP kesintisi en yüksek öncüllüklü kesintidir. Bu pinin aktifleştirilmesinden sonra, mikroişlemci kesinti aramasını gerçekliştirmelidir. TRAP maskelenemez, ne de atılabilir. Trap sözcüğü tuzak, pusu anlamına geliyor. TRAP çok büyük hatalar durumunda kullanılıyor, örneğin, elektrik kaynağıyla ilgili ya da veriyolundan karışık veri aktarımı sırasında yaşanan sorunlar. Mikroişlemci TRAP kesintisini, sadece alçak seviyeden yüksek seviyeye atlayıştan (yükselen kenar) sonra etkinleştirici en az 160 mikrosaniye aynı durumda kalırsa tanıyor. Etkinleştirici yüksek seviyeye çıkarsa ve yeniden alçak seviyeye inerse ondan sonra da yeniden ikinci kez yüksek seviyeye çıkarsa, mikroişlemci kesinti aramasını kabul etmiyor.

RST 5.5 Sıfırlama kesintileri (Restart)
RST 6.5 Kesinti onarımı için alt programı çalıştırdıktan sonra,
RST 7.5 mikroişlemci kesilen programa geri dönmüyor, hemen programı yeniden başlatıyor. Sıfırlama adresi TRAP ve RESET kesintileri için sabittir. RST 6.5 ve RST 5.5 maskelenen kesintilerdir. Onların maskelenmesi ve okunması için SIM (set interrupt mask) ve RIM(read interrupt mask) yönergeleri kullanılıyor. Mikroişlemcinin RST 6.5 ve 5.5 pinlerinin kesinti aramalarını tanıması için etkinleştirici aramaların kabul edilmesine kadar yüksek seviyede olmalıdır.

$\overline{RESET\ IN}$ Bu pinin aktifleştirilmesinde sonra, program sayacının sıfır durumuna ayarlanması ve HLDA flip - flopun kesinti bayrakların sıfırlanması meydana geliyor. Belirli durumlarda durum - yazmacın ve genel amaçlı yazmaçların durumlarının değişmesi de meydana gelebilir.

RESET OUT Reset out sinyaliyle, mikroişlemci bilgisayarın diğer aygıtlarına sıfırlandığını haber veriyor.

CLK Dijit palsı - atışı (periyodik dörtgen tetikleme dizisi) pals üretici tarafından üretiliyor. Mikroişlemcinin çıkışında elde edilen çalışma frekansı

X1 ve X2 pinlerinde bağlanmış kristal salıngacın frekansından iki misli daha düşüktür.

X1,X2 Kristal salıngacın giriş pinleri X1 ve X2 seviye açısından birbirine ters sinyallerdir (biri yüksek seviyedeyken, diğeri alçak seviyededir) ve mikroişlemcinin içinde iç zamanlayıcı için kullanılıyor.

SID Dizisel iletişim için giriş pini (Serial Input Data)

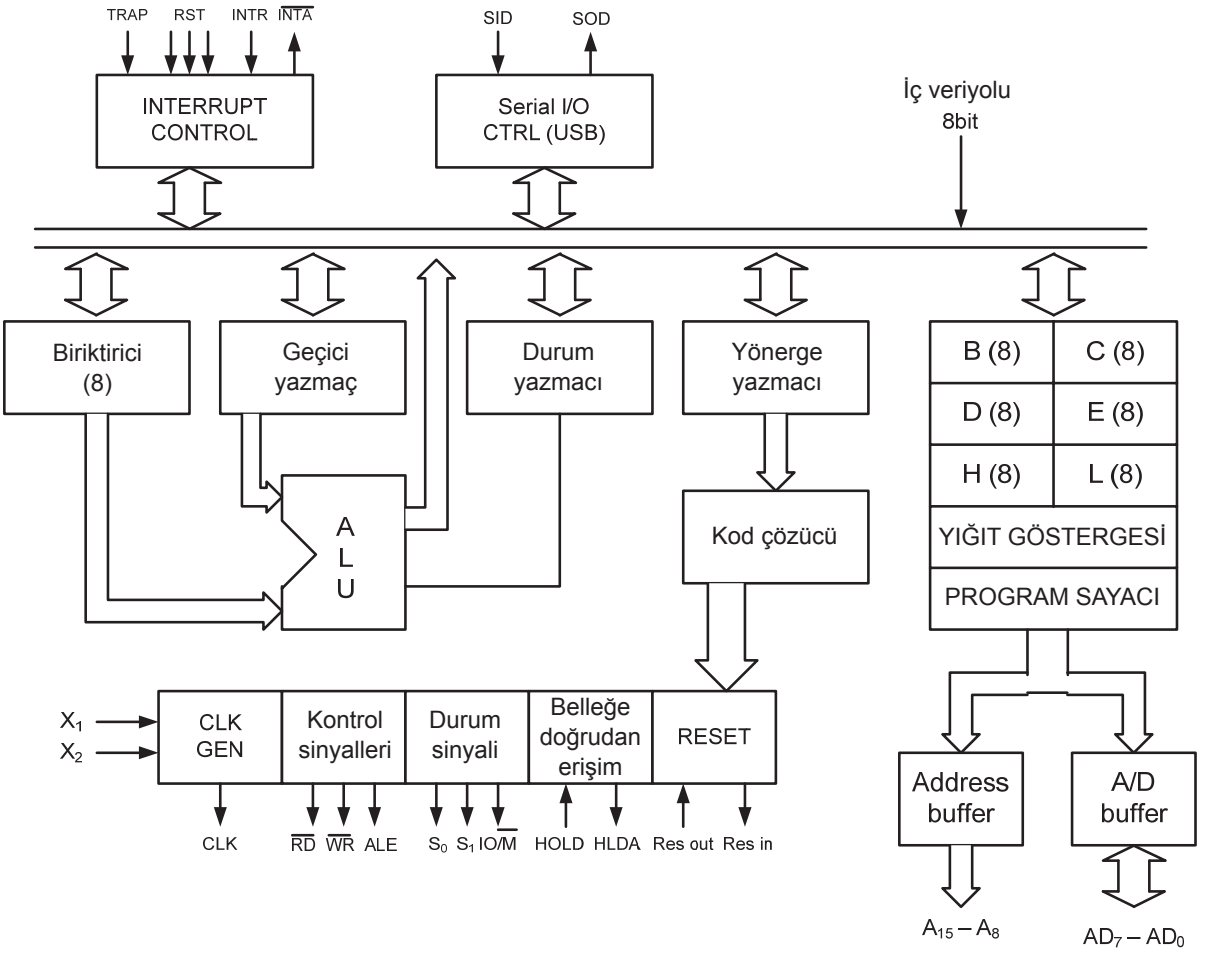
SOD Dizisel iletişim için çıkış pini (Serial Otput Data)

7.2. 8085 Mikroişlemcinin Yapısı

İntel şirketinin 8085 mikroişlemcisi piyasaya 1977 yılında çıktı. Bu mikroişlemci çok az sayıda donanım bileşeni gerektiriyor, ancak bu bileşenlerin yerleşme şekli ve işlevi, çalışma sırasında büyük etkililik sağlayabiliyor.

„İntel" şirketi 200 milyon üzerine bu tür yongadan üretmiştir. "Zilog" şirketi 500 milyondan fazla mikroişlemci satan başka bir şirkettir. "Zilog" şirketinin Z - 80 adlı mikroişlemcisi, 8085 mikroişlemciyle hemen aynıdır. 8085 mikroişlemcisi organizasyon açısından, PIC 16f84 gibi işletim yönetiminde büyük kullanım gören mikrodenetimcilere çok benzerdir. Güçlüğü açısından 8085, pentiyum mikroişlemcisiyle kıyaslanamaz, ancak 8085 mikroişlemcinin daha uzun yıllar üretileceği öngörülüyor çünkü çok güçlü mikroişlemciler aramayan, çok sayıda basit elektronik aygıtlarında geniş çapta kullanılıyor. Resim 7.2.'de 8085 mikroişlemcinin işlevi blok modeli, onun tüm genel ve özel yazmaçlarıyla beraber göstrerilmiştir. 8085 mikroişlemci 8 - bitli mikroişlemcidir ve buna göre 1 bayt büyüklüğünde veriler işletiyor. İçeriğinde **8MHz frekanslı** dijit palsı üreten pals üreticisi, kesintilerle çalışma ve zamanlama kontrolu için özel denetimci bulunuyor. **64KB** kapasiteli **belleğe** erişimi var, yani bellek alanının dolaysız adrelenmesi için 16 adres pinine sahiptir 2^{16} =64K).

Yazmaçlar mikroişlemcinin iç belleğidir, ancak az kapasiteleri vardır. Mikroişlemcinin işlettiği veriler ardaşıl bellek yerlerinde yerleşiktir. Mikroişlemci birinci verinin adresini belleğe gönderiyor, bellek ise gereken veriyi mikroişlemciye gönderiyor. Mikroişlemci birinci veriyi işlettikten sonra, sıradaki veriyi çağırıyor vs. Çalıştırılan yönergenin sonucu farklı yerlerde yerleşebilir. Elde edilen sonucun bir kez daha işletilmesi gerekirse, o zaman sonuç biriktiricide yerleştirilebilir. Elde edilen sonucun, hemen sıradaki adımda değil, ancak yakın zamanda yeniden işletilmesi gerekirse, o zaman sonuç mikroişlemcinin yazmaçlarından birine yerleşebilir. Ayrıca, son sonuç ya da çok ileride işletilmesi gereken sonuç söz konusu olursa, o zaman sonuç bellekte yerleştirilebilir.



Resim 7.2. 8085 mikroişlemcinin yapısı

Biriktirici (akümülatör) mikroişlemcinin çalışma yazmacıdır ve aritmetik mantık birimin girişinde bulunuyor. Biriktirici dışında **genel amaçlı yazmaçlar** grubuna daha altı başka yazmaç giriyor ve onlar B, C, D, E, H, L harfleriyle işaretleniyor. Bu yazmaçlar 8 bit biriktirebilir ve veri kaynağı ya da elde edilen sonucun hedefi olabilirler. Yazmaçların hepsi, tablo 7.1.'de gösterildiği gibi 3 bitin kombinasyonu olarak tanımlanıyor.

A	111
B	000
C	001
D	010
E	011
H	100
L	001

Tablo 7.1. 8085 mikroişlemcinin genel yazmaçlarının kodları

Çoğu kez, 16 - bitli adresleri yerleştirmek ve saklamak için genel yazmaçlar çift olarak kullanılıyor. Böyle durumda şu işaretler kullanılıyor: BC, DE, HL. B, D ve H yazmaçları daha yüksek ağırlıklı bitleri, C, E ve L yazmaçları ise daha düşük ağırlıklı bitleri saklıyorlar.

Program bayt çoğunluğu olarak tanımlanıyor. Baytların yerleşik olduğu bellek yerleri tüm bellekte rastgele olarak dağılmamıştır, hemen ardaşıl bellek yerlerinde sıralanıktır. Örneğin, beşinci baytın adresini, dördüncü baytın adresine bir ekleyerek elde ediyoruz. Bir programın çalışması zamanında, mikroişlemci adresleri bellekte sıralanmış olduğu şekilde birer birer çağırıyor. **Program sayacı**, bellekten aktarılması ve işletilmesi gereken sıradaki baytın adresini hafıza eden, mikroişlemcide bulunan yazmaçtır. Mikroişlemci, bellekten mikroişlemciye her yeni baytın getirildiğinden sonra, program sayacının değerine bir ekliyor. Program sayacı, mikroişlemcide verilen anda işletilen baytın adresini değil, sıradaki baytın adresini içeriyor. Bu kural sadece atlama yönergelerinde ve alt programların çağrılma yönergelerinde geçerli değildir. Atlama yönergesi gerçekleştirilirse, o zaman program sayacının değeri bir için artmayacak. Onun yerine program sayacı, atlama adresinde verilmiş olan adresin değerini alacak. Atlama yönergelerinde mikroişlemci atlamanın gerçekleştiği yere artık geri dönmüyor.

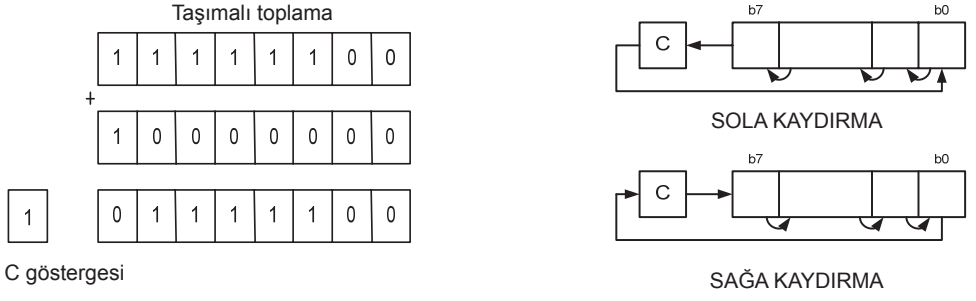
Yığıt bellek alt programlarla çalışmak için kullanılıyor. Yığıt özel olarak organize edilmiş bellek alanıdır. Yığıt büyük sayıda bellek yerlerinden oluşuyor. Yığıt bellek tepeden dolduruluyor ve boşaltılıyor. Her yeni veri yığıtın tepesinde yazılıyor. Yığıtın ortasında yerleşmiş olan veriyi almak istersek, aradığımız verinin üstünde bulunan tüm bellek yerlerini boşaltmamız gerekiyor ve bu şekilde aradığımız veri (bellek yeri) otomatik olarak yığıtın tepesinde yetişecek. Mikroişlemci adı **yığıt göstergesi** (stack pointer) olan özel yazmaç içeriyor. Yığıt göstergesi 16 - bitli yazmaçtır ve yığıtın tepesinde bulunan bellek yerinin adresini içeriyor.

Mikroişlemcinin gerçekleştirebileceği her yönergenin, kendi 8 - bitli işlem kodu vardır. 8 bit ile 256 farklı kombinasyon yapılabilir, yani 256 işlem kodu tanımlanabilir. Yönerge bellekten mikroişlemciye aktarıldıktan sonra, aktarılan yönerge, adı **yönerge yazmacı** olan özel bir yazmaçta yerleştiriliyor. Yönerge bu yazmaçta işlemsel olarak çalıştırılana kadar kalıyor. Yönerge yazmacında yönerge kod çözücüsüne gönderiliyor. Kod çözücünün 8 girişi ve 256 çıkışı vardır. Bu 256 çıkıştan her biri belli bir etkinlik teşvik ediyor. Kod çözücüsü mikroişlemcinin yönetim birimiyle bağlıdır. Aslında, işlem kodu kod çözücüsüne giriyor ve yönetim biriminde uyartıya yol açıyor, yönetim birimi ise çıkışta yönetim sinyalleri veriyor. Hangi yönetim sinyallerin aktif olacağı, işlem kodundan ve aynı zamanda kristal salıngacın (atış üreticinin) dijit atışlarına bağlıdır.

Adres yazmacı, **verilen işlemin uygulandığı işletenin yerleşmiş olduğu bellek yerindeki adresini içeriyor**.

Aritmetik - mantık birimi, ikili sayılara aritmetik - mantıksal yönergeleri gerçekleştiren, merkezi işlemcinin bir parçasıdır. Her aritmetik - mantık birimi, ikili aritmetiğe uyumlu şekilde, iki yazmacın içeriklerine toplama gerçekleştiren toplayıcı içermelidir. Sadece temel toplayıcıyı kullanarak, temel aritmetik işlemler olan çıkarma, çarpma ve bölme işlemlerini gerçekleştiren programlar yapılabilir. Pratikte, aritmetik - mantık birimi toplama dışında, çıkarma, bu mantıksal işlemler ve dizinlerin kaydırılması gibi donanımsal ve diğer fonksiyonlar gerçekleştirebilir. Aritmetik - mantık birimine **durum - yazmacı** bağlıdır. Bu yazmaç durum göstergeleri, bayraklar (flags) içeriyor. Bayrakların farklı rolü olabilir, ancak genelde bir yönergenin çalıştırılmasından sonra aritmetik - mantık biriminin durumunu gösteriyorlar. Tipik bayraklar şunlardır:

- **C (carry)** – Taşıma göstergesi. Bu bit, toplama işleminde ikili sayılarda sekizinci pozisyondan dokuzuncu pozisyona taşıması olduğu zaman ya da çıkarma işleminde dokuzuncu pozisyondan sekizinci pozisyona ödünç alma durumu olduğu sırada, donanım yoluyla ayarlanıyor. Ayrıca C bayrağı dizinleri kaydırma işlemlerinde de kullanılıyor. Resim 7.3.'te C bayrağının kullanımıyla ilgili iki örnek verilmiştir:

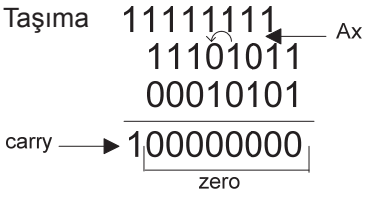


Resim 7.3. Carry - taşıma gösterge bayrağının pratik kullanımı için örnekler

- **Z (zero)** – gerçekleşen işlemin sonucu sıfır olunca, sıfır bayrağı mantıksal bire eşittir, elde edilen sonuç ise sıfırdan farklı ise, o zaman Z bayrağı mantıksal sıfıra eşittir
- **S (sign)** – işaret bayrağı. İşlem sonucu negatif değer ise bu bayrak bire eşittir, aksi taktirde mantıksal sıfırdır.
- **P (parity)** – Çift değer bayrağı. İşlem sonucu çift sayı birli içerirse, P bayrağı birdir, aksi taktirde ise P bayrağı mantıksal sıfırdır.

- **Ax –** Ax yardımcı taşıma bayrağıdır ve ikili sayıların toplanması sırasında dördüncü pozisyondan beşinci pozisyona taşıma ya da çıkarma sırasında beşinci pozisyondan dördüncü pozisyona ödünç vermesi olduğu zaman aktif oluyor.

Örnek 7.1: $11101011_{(2)}$ ve $00010101_{(2)}$ sayılarının toplanması sırasında hangi bayraklar etkinleştiriliyor?



C,Z ve Ax bayrakları aktifleştiriliyor.

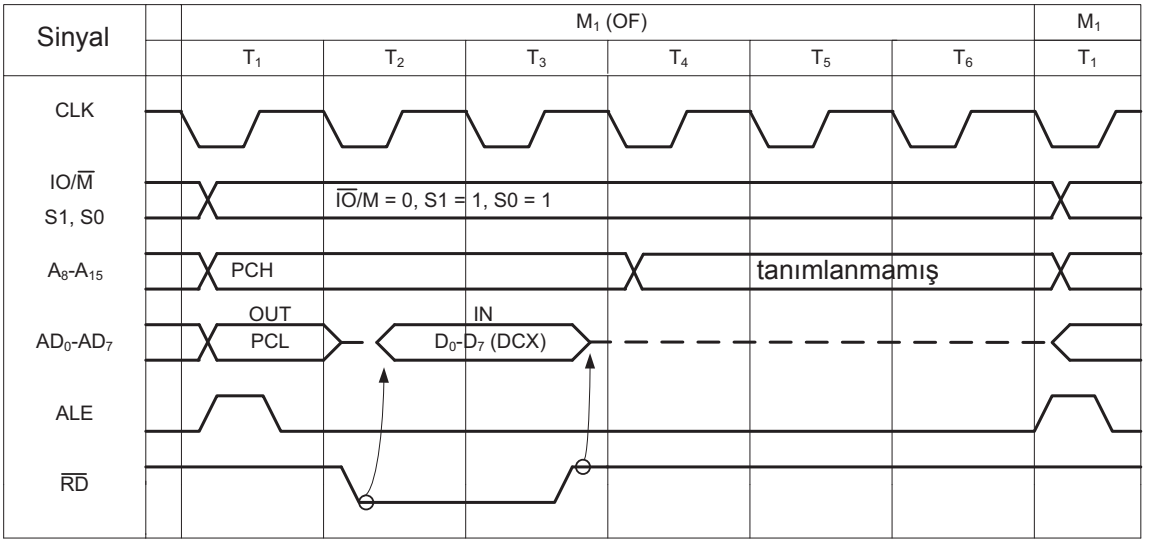
7.3. 8085 Mikroişlemci için Makine Döngüsü

Her yönerge döngüsü, **işlem kodunun aktarımı için makine döngüsüyle** başlıyor. 8085 mikroişlemcisinde makine kodunun, verilen yönerge için tek olan 8 - bitli kombinsyon tanımladığını önceden vurgulamıştık. Makine döngüsü diğer döngülerden, üç dijit palstan daha uzun sürdüğüne göre farklıdır, çünkü mikroişlemcinin ne yapması gerektiğini bilmesi için kod çözümlenmesi yapılması gerekiyor.

Resim 7.4.'te makine döngüsü - işlem kod aktarımı için zamanlama diyagramı verilmiştir. Hatırlayalım, zamanlama diyagramı yönergenin gerçekleştiği sırada aktifleştirilen kontrol sinyallerin arasında bağımlılığın grafiksel gösterişidir. Her makine döngüsünün başlangıcında, 8085 mikroişlemcisi, makine döngü türünün belirlenmesi için durum hatlarını etkinleştiriyor. İşlem kodunun bellekten mikroişlemciye taşındığı zaman, durum hatların şu değerleri vardır:

IO/$\overline{M}$=0 – işlemci bellekle iletişim kuracak (dış aygıtla değil)
S_1=1 – okuma işlemi gerçekleşek (yazma değil)
S_0=1 – aktarılan bayt işlem kodudur (veri,işletilen değil)

Birinci dijit atışı sırasında, 8085 mikroişlemci bellek yerinin ya da hitap edildiği giriş - çıkışın kapısının tanımlanması için 16 - bitli adres gönderiyor. İşlem kodu aktarımı durumunda, adres veriyolundan program sayacın değeri gönderiliyor. Daha değerli bayt (PCH Program Counter High) A_8 - A_{15} hatlarından geçiriliyor ve dördüncü dijit atışın tamamlanmasına kadar bu bayt onlarda kalıyor. Program sayacının daha az değerli bayt (PCL - Program Counter Low), AD_o - AD_7 hatlarından geçiriliyor ve onlarda sadece bir dijit atışı içinde kalıyor. Şöyle ki, birinci dijit palsından sonra, AD_0 - AD_7 hatları, belli bir bellek yerinden okunan işlem kodonun aktarılması için serbest olmalıdır. ALE sinyalı sadece birinci dijit atışı sırasında, adres - veri veriyolunun adresi bitlerinin kullanıldığı sürede, yüksek seviyededir. ALE (Adress Latch Enable) sinyali 8212 adres mandalının etkinleştirilmesi için kullanılıyor. Mikroişlemci durum sinyallerini ve adresi belleğe gönderdikten sonra, belleği okumaya hazırlamak için $\overline{RD}$ kontrol sinyali alçak seviyeye düşüyor.



Resim 7.4. işlem kodu aktarımı için makine döngüsünün zamanlama diyagramı

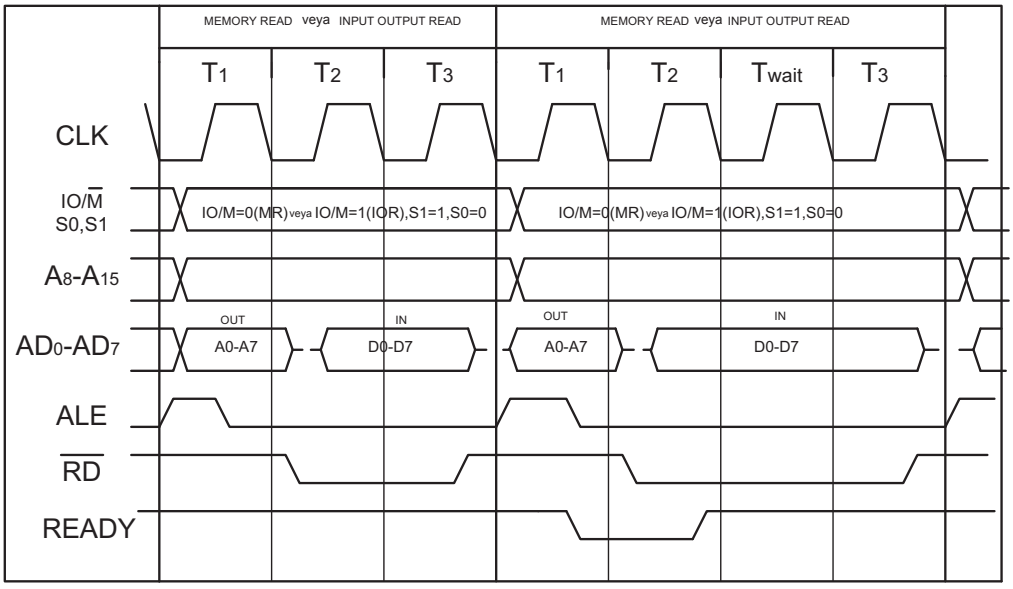
İkinci dijit palsında, bellek araştırılacak ve belli bir gecikmeden sonra aranan baytı–işlem kodunu adres - veri veriyoluna getirecek. Gecikmenin ne kadar süreceği, belleğe erişim zamanına bağlıdır.

Üçüncü dijit palsı, T_3 zamanında, 8085 mikroişlemcisi işlem kodunu AD veriyolundan alıyor ve yönerge yazmacıya yerleştiriyor. Mikroişlemci RD sinyalini yüksek seviyeye çıkarıyor ve onunla bellek aygıtını etkinleştirecek.

Dördüncü dijit atışı, T_4 sırasında işlem kodunun çözümlenmesi yapılıyor ve mikroişlemci sıradaki dijit palsıyla T_5'e mi, sıradaki yönergenin T_1'ine mi gireceğine karar veriyor. T5 ve T6 zamanlarında, A_0 - A_7'den adres bitleri değişebilir ve bundan dola-

yı bu iki dijit palsı sürünce, $\overline{RD}$ sinyalini yüksek seviyeye getirerek tüm bellek ve giriş - çıkış aygıtlarını etkisiz hale getirmek önemlidir. Aksi taktirde, A_0 - A_7 adres bitleriyle bazı aygıt seçilerek hataya yol açılabilir.

Resim 7.5.'te bellekten okuma için **iki ardışıl makine döngüsü** gösterilmiştir. Birinci makine döngüsünde bekleme (T - wait) durumu yok, diğerinde ise bir bekleme durumu var. Bellekten okuma makine döngüsü için durum sinyallerinin şu değerleri var: IO/M=0, S1=1, S0=0.



Resim 7.5. bellekten okuma makine döngüsü için zamanlama diyagramı

İşlem kodunun aktarımı makine döngüsüne kıyasen, sadece S_0 sinyali farklıdır. İşlem kodunun aktarımı makine döngüsünde aktarılan bayt işlem kodudur, bellekten okuma makine döngüsünde ise taşınan bayt belli bir yönergenin uygulandığı veri ya da işletilen olduğunu vurgulayalım. İşlem kodunun aktarımı ve bellekten okuma makine yönergelerin arasında ikinci fark, birincisinin en az 4 atış, ikincinin ise en az 3 atış sürmesidir. Bu durum doğaldır, çünkü bellekten okuma işleminde kod çözümlenmesi gerçekleşmiyor. Üçüncü fark ise şudur: işlem kodunun aktarılmasında adres her zaman program sayacından alınıyor, bellekten okuma döngüsünde bellek adresi farklı yerlerden kaynaklanabilir. Bellek adresi yönergenin içinde olabilir (MOV addr) ya da HL yazmaçlar çiftinde bulunabilir.

İşletilenler, başka bir şekilde vurgulanmamışsa, genel yazmaçlardan birinde ya da biriktiricide yerleşiyorlar. Makine döngüsü bekleme durumu (Twait) içerirse, o zaman READY sinyali mecburen aktifleştiriliyor. İkinci atışta mikroişlemci her zaman READY sinyalini kontrol ediyor. READY sinyali yüksek seviyede bulunuyorsa, mikroişlemci T3 durumuna devam edecek ve döngü tamamlanacak. READY sinyalı alçak

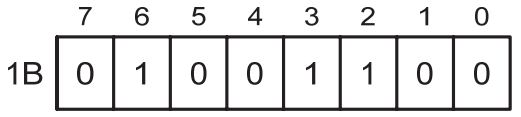
seviyedeyse, o zaman mikroişlemci Twait bekleme durumu ekleyecek. Mikroişlemci, belleğe erişim zamanına bağlı olarak birkaç böyle bekleme durumu ekleyebilir. Giriş - çıkış aygıtlardan okuma (İ/O read) makine döngüsü için, kontrol sinyalerin bağımlılığı için zamanlama diyagramı, bellekten okuma makine döngüsünün zamanlama diyagramıyla aynıdır, sadece IO/M durum sinyali 1'e eşittir. Bellekte veri yazması sırasında RD kontrol sinyali yerinde, WR sinyali kullanılıyor ve yazılan veriler T3'ün sonuna kadar AD veriyolunda kalıyorlar ve bellekten okumada RD ile olduğu gibi WR sinyalinin seviyesine bağlı değiller.

7.4. 8085 Mikroişlemcide Adresleme Şekilleri

İşletmek için verilerin (işlenenlerin) nasıl bulunduğuna bağlı olarak, 8085 mikroişlemcide **4 adresleme şekli farkediyoruz**: doğrudan, yazmaçlı dolaysız, dolaysız ve yazmaçlı dolaylı.

Yazmaçlı dolaysız adreslemede, işletilmesi gereken veriler, mikroişlemcinin genel yazmaçlarında bulunuyor. Bu adresleme şeklini kullanan yönergeler bir bayt içeriyorlar ve bu bayt yönergenin işlem kodunu tanımlıyor. Yönerge işlenenleri içermiyor ve mikroişlemcinin nasıl etkinliğini gerçekleştirmesi gerektiğini bilmesi için işlem kodu yeterlidir. Yazmaçlı adreslemeyi kullanan yönergelere birinci biçim yönergeler deniyor. Onlar bir baytlık bellek alanı kaplıyor, yani bir bellek yeri kaplıyor.

Örnek 7.2: Resim 7.6.'da MOV H,L yönergesi makine dilinde ifadesi tanımlanmıştır ve bu yönerge birinci biçim yönergesidir. Yönergenin anlamı L yazmacından H yazmacına veri aktarımıdır. Her iki yazmaç mikroişlemcinin içinde bulunuyor.



Resim 7.6. MOV H, L yönergesi için birinci yönerge biçimi

Altıncı ve yedinci bit yönergeyi tanımlıyorlar. Makine dilinde 01 bitleriyle MOV yönergesi dışında hiçbir başka yönerge başlamıyor. Beşinci, dördüncü ve üçüncü bit H yazmacını kodunu tanımlıyorlar ve H yazmacı aktarılan işlenenin hedefini tanımlıyor. İkinci, birinci ve sıfırıncı bit L yazmacı için kodu tanımlıyorlar. L yazmacı aktarılan verinin kaynağıdır. Genel yazmaçların kodları tablo 7.1.'de gösterilmişti.

Doğrudan adreslemede, işlenen yönergenin içinde bulunuyor. Bu adresleme şeklinde,işlem kodunun hemen devamında veri geliyor. Bu yönergeler için ikinci yönerge biçimleri olduklarını diyoruz. İkinci yönerge biçim iki bayt içeriyor. Birinci bayt her zaman işlem kodudur, ikinci bayt ise işletilmesi gereken 8 - bitli veridir. Çeviricide yazılan bu yönergeler 8 - bitli sayı içeriyorlar, yani onlatılı sayı sisteminde iki rakamlı sayı tanımlıyorlar.

Örnek 7.3: Resim 7.7.'de MVI A, 9EH yönergesinin yönerge biçimi gösterilmiştir. Bu yönerge ile 9EH=10011110B verisi A biriktiriciye giriliyor.

	7	6	5	4	3	2	1	0
1B	0	0	1	1	1	1	1	0
	7	6	5	4	3	2	1	0
2B	1	0	0	1	1	1	1	0

Resim 7.7. MVI A,9EH yönergesinin ikinci yönerge biçimi

Birinci bayttan beşinci, dördüncü ve üçüncü bit A yazmacının kodunu tanımlıyor. Birinci baytın diğer bitleri MVI yönergesinin kodudur. İkinci bayt biriktiriciye girilmesi gereken ikili sayıyı tanımlıyor 10001110B=9EH.

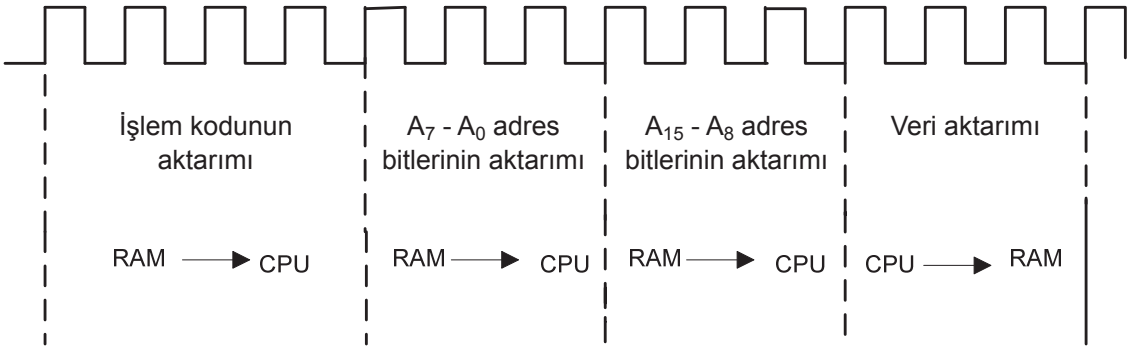
Dolaysız adreslemede veriyi bilmiyoruz, ancak yönerge aradığımız veriyi okuyabileceğimiz bellek yerin adresini veriyor. Yukarıda incelediğimiz iki adresleme şekline kıyasen, dolaysız adreslemede işlenenlerin mikroişlemciye ulaşması için belleğe en çok sayıda erişim gerekiyor. Dolaysız adresleme şeklini kullanan yönergeler üçüncü yönerge biçiminden yönergelerdir. **Üçüncü yönerge biçimi** üç bayt içeriyor. Birinci bayt işlem kodudur, ikinci ve üçüncü bayt ise beraber olarak bellek yerin 16 - bitlik adresini tanımlıyorlar. Bu yönerge biçimi, işletilmesi gereken verinin RAM belleğinden bir bellek yerinde okunması ya da yazılması gerektiğinde kullanılıyor. Üçüncü biçimden çevirici yönergeleri 16 - bitli sayı ya da on altılı sayı sisteminde dört rakamlı sayılar içeriyor.

Örnek 7.4: Resim 7.8.'de STA 8FFFH yönergesinin yönerge biçimi gösterilmiştir. Bu yönergeyle, 8FFFH adresli bellek yerinden biriktiriciye aktarılıyor. Birinci bayt, 32H=00110010B, STA yönergesinin işlem kodunu tanımlıyor. İkinci bayt, 11111111 B=FFH, bellek adresinin daha az değerli bitleridir A_7 - A_0. Üçüncü bayt, 8FH=10001111B daha yüksek değerli adres bitleridir A_{15} - A_8..

	7	6	5	4	3	2	1	0
1B	0	0	1	1	0	0	1	0
	7	6	5	4	3	2	1	0
2B	1	1	1	1	1	1	1	1
	7	6	5	4	3	2	1	0
3B	1	0	0	0	1	1	1	1

Resim 7.8. STA 8FFFH yönergesi için üçüncü yönerge biçimi

Resim 7.9.'da STA 8FFH yönergesinin çalıştırılması için her dört makine döngüsü gösterilmiştir.



Resim 7.9. STA yönergesi için makine döngüsü

Birinci makine döngüsünde işlem kodu (birinci bayt) bellekten mikroişlemciye aktarılıyor. İşlem kodu yönerge yazmacında yerleşiyor ve kod çözümlenmesi gerçekleşiyor. İkinci makine döngüsünde A_0'dan A_7'ye kadar adres bitleri (ikinci bayt) bellekten mikroişlemciye taşınıyor. Onlar adres yazmacında (address latch) yerleşiyorlar. Üçüncü makine döngüsünde A_8'den A_{15}'e kadar daha yüksek adres bitleri (üçüncü bayt) için aynı süreç tekrarlanıyor. En sonunda dördüncü makine döngüsünde biriktiricinin içeriği belirlenmiş bellek yerine aktarılıyor.

Yazmaçlı dolayı adresleme şeklinde, HL yazmaçlar çift aradığımız verinin bulunduğu bellek yerinin adresini içeriyor. MOV A,M yönergesi yazmaçlı dolaylı adresleme için örnektir. Bu yönergeyle HL yazmaçlar çiftinde adresli bellek yerinden veri alınıyor ve birikicide kopyalanıyor.

7.5. Yönergeler Kümesi

8085 mikroişlemcinin yönergeler kümesi 246 yönergeden oluşuyor, ancak başlangıç için sadece daha önemli 40 yönergeyi tanıtacağız. Her verilen yönerge için şu bilgiler verilmiştir: yönergenin yaptığı etki, işlem kodu, bayt sayısıyla ifade edilmiş yönerge büyüklüğü ve uygulanan adresleme şekli. Ayrıca, yönergenin açıklamasından sonra örnek olarak bir çevirici programdan bir bölüm verilmiştir. Bu örneklerde şu bilgiler vardır:

- on altılı sayı sisteminde verilmiş bellek konumun adresi (on altılı sayı sisteminde bir rakam, ikili sayı sisteminde 4 rakama denk geliyor).
- önceden adreslenmiş bellek yerinin içeriği.
- çeviricide verilmiş yönerge
- yorum

7.5.1. Veri Aktarma Yönergeleri

Bir aktarmayla sadece bir baytlık bilgi taşınabilir. Veri aktarma yönergelerinde kaynak (source) ve hedef (destination) terimlerine rastlanıyor. Kaynaktan veri alınıyor ve hedefte yerleşiyor. Veri aktarımı yazmaç - yazmaç, yazmaç - bellek yeri ya da bellek yeri - yazmaç yönünde yapılabilir. Çevirici yönergesiyle bir bellek yerinden başka bir bellek yerine, doğrudan aktarma yapılamaz, yani başka sözlerle, birinci bellek yerinden alınan veri, önce mikroişlemcide bir yazmaçta geçiriliyor, ondan sonra da mikroişlemciden ikinci bellek yerinde gönderiliyor. Aktarma sırasında veri değişmiyor. Aktarımın gerçekleştirilmesinden sonra veri kaynağın içeriği, hedefin içeriğiyle aynı olmasını not etmek gerekiyor. Aslında aktarma yerine verinin kopyalanmasının gerçekleştiğini söyleyebiliriz. Herşeye rağmen kopyalamak terimi kullanılmıyor, terim olarak aktarma kullanılıyor.

MVI r, veri

İşlev	Veri r yazmacında yerleşiyor
İşlem kodu	00DDD110
Bayt sayısı	2 bayt
Adresleme şekli	Doğrudan adresleme

D harfi İngilizce destination sözcüğün ilk harfidir ve anlamı amaç, hedef demektir. Yukardaki yönergede veri hedefi bir genel yazmaçtır ve onun DDD üç - bitli kodunu tablo 7.1.'in yardımıyla belirliyoruz.

Örnek 7.5:

```
8200 3E MVI A,9EH      ;9EH on altılı sayıyı biriktiricide
                       ;yerleştir
8201 9E                ;Veri
```

MOV r2, r1 ya da MOV destination, source

İşlev	r1 yazmacın içeriği r2 yazmacına kopyalanıyor
İşlem kodu	01DDDSSS (D - destination, S - source)
Bayt sayısı	1 bayt
Adresleme şekli	Yazmaçlı dolaysız

Örnek 7.6:

```
8150  26    MVI H, 00H  ;H yazmacını sıfır değeriyle dolduruyoruz.
8151  00                ;Veri
8152  6C    MOV L, H    ;H yazmacın içeriği L yazmacında
                        ;kopyalanıyor.
```

LDA adr (LOAD - doldurma)

İşlev	*adr* adresli yerin içeriği biriktiriciye taşınıyor
İşlem kodu	00111010
Bayt sayısı	3 bayt
Adresleme şekli	Dolaysız adresleme

Örnek 7.7:

```
8200  3A  LDA  81C0H   ;81C0(16) adresli bellek yerin içeriği
                       ;biriktiricide kopyalanıyor.
8201  C0                ; A7 - A0 adresi
8202  81                ; A15 - A8 adresi
8203  6F  MOV  L,A      ;biriktiricinin içeriği L yazmacında
                       ;kopyalanıyor.
```

STA adr (STORE - depolamak, **LDA**'nın tersi)

İşlev	Biriktiricinin içeriği adr adresli bellek yerine kopyalanıyor
İşlem kodu	00110010
Bayt sayısı	3 bayt
Adresleme şekli	Dolaysız adreslemee

Örnek 7.8:

```
8200  3E   MVI A,55H     ;Biriktirici 55H verisiyle dolduruluyor.
8201  55                 ;Veri
8202  32   STA 8330H     ;Biriktiricinin içeriği 8330 adresli
                         ;yerde kopyalanıyor.
8203  30                 ;A7 - A0
8204  83                 ;A15 - A8
```

IN adr

İşlev	„adr" ara birim adresli giriş aygıtından bayt biriktiriciye aktarılıyor
İşlem kodu	11011011
Bayt sayısı	2 bayt
Adresleme şekli	Doğrudan

Örnek 7.9

```
8000  DB   IN 01H      ;01H ara birim adresli giriş aygıtından veriyi
                       ;biriktiriciye kopyala.
8001  01               ;Arabirim adresi
8002  67   MOV H,A     ;Biriktiricinin içeriğini H yazmacında
                       ;kopyala.
```

OUT adr (**IN**'in tersi)

İşlev	Biriktiricinin içeriği „adr" ara birim adresli çıkış aygıtına kopyalanıyor
İşlem kodu	11010011
Bayt sayısı	2 bayt
Adresleme şekli	Doğrudan adresleme

Örnek 7.10:

```
8200  3E       MVI A,55H   ;55H verisi biriktiricide yerleşiyor.
8201  55                   ;Veri
      8202  D3  OUT 02H    ;Biriktiricinin içeriği 02 ara birim adresli
                           ;aygıtta kopyalanıyor.
8203  02                   ;Arabirim adresi
```

7.5.2. Aritmetik Yönergeler

Donanım yoluyla iki aritmetik işlemi gerçekleşebilir: toplama ve çıkarma. Bu işlemlerin gerçekleşmesi sırasında işlenenlerden biri biriktiricide, diğer işlenen ise mik-

roişlemcinin başka bir yazmacında bulunuyor. Ayrıca, yönergenin çalıştırılmasından sonra, elde edilen sonuç birikitiricide yazılıyor ve bu şekilde işlenenlerden biri kayboluyor.

ADD r (aritmetik toplama)

İşlev	r yazmacının içeriği biriktiricinin içeriğine ekleniyor
İşlem kodu	10000SSS
Bayt sayısı	1 bayt
Adresleme şekli	Yazmaçlı dolaysız
Bayraklar	Zero, carry

Örnek 7.11:

```
8200  7C  MOV A,H     ;H yazmacının içeriği biriktiricide
                      ;kopyalanıyor.
8201  85  ADD L       ;L yazmacının içeriği biriktiricinin
                      ;içeriğine ekleniyor.
8202  67  MOV H, A    ;Biriktiricinin içeriği H yazmacında
                      ;kopyalanıyor.
```

SUB r (**substract** - çıkarma)

İşlev	r yazmacının içeriği biriktiricinin içeriğinden çıkarılıyor
İşlem kodu	10010SSS
Bayt sayısı	1 bayt
Adresleme şekli	Yazmaçlı dolaysız
Bayraklar	Zero, carry

INR r (increment - arttır) ve **DCR r (decrement** - azaltır)

İşlev	Yazmacın içeriği bir için artıyor/azalıyor
İşlem kodu	000DDD0100/00DDD101
Bayt sayısı	1bayt
Adresleme şekli	Yazmaçlı dolaysız

Örnek 7.12:

```
8000  3A   LDA 81D8H   ;81D8H adresli konumun içeriği
                       ;biriktiricide kopyalanıyor.
8001  D8                ;A7 - A0
8002  81                ;A15 - A8
8003  3C   INR A        ;Biriktiricinin içeriği
                       ;bir için artıyor.
```

```
8004  32 STA 81D8H     ;Biriktiricinin içeriği 81D8K
                       ;adresle götürülüyor
8005 D8                ;A7 - A0
8006 81                ;A15 - A8
```

7.5.3. Mantıksal İşlemler

ANA r (AND ACCUMULATOR)

İşlev	Birikitiricinin içeriğine ve „r" yazmacının içeriğine „VE" mantıksal işlemi uygulanıyor ve sonuç biriktiricide yazılıyor.
İşlem kodu	10100SSS
Bayt sayısı	1 bayt
Adresleme şekli	Yazmaçlı dolaysız

ORA r (OR ACCUMULATOR)

İşlev	Birikitiricinin içeriğine ve „r" yazmacının içeriğine „YADA" mantıksal işlemi uygulanıyor
İşlem kodu	10110SSS
Bayt sayısı	1 bayt
Adresleme şekli	Yazmaçlı dolaysız

CMA (Complement accumulator)

İşlev	Biriktiricide tüm bitler eviriliyorlar
İşlem kodu	00101111
Bayt sayısı	1 bayt
Adresleme şekli	Yazmaçlı dolaysız

Örnek 7.13:

```
8000  3A  LDA 8250H    ;8250 adresli bellek yerin içeriği
                       ;biriktiricide kopyalanıyor
8001  50                ;A7 - A0
8002  82                ;A15 - A8
8003  6F  MOV L,A       ;Biriktiricinin içeriği L yazmacında
                       ;kopyalanıyor
8004  3A  LDA 8251 H    ;8251 adresli bellek yerin içeriği
                       ;biriktiricide kopyalanıyor
8005  51                ;A7 - A0
```

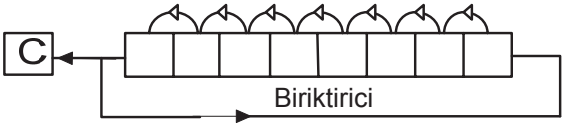
```
8006  82                ;A15 - A8
8007  A5   ANA L       ;biriktiricinin ve L yazmacının içeriklerinin üzerine
                        ;„VE” mantıksal işlemi uygulanıyor
8008  32   STA 8252H   ;biriktiricinin içeriği 8252 adresli
                        ;bellek yerinde yerleştiriliyor.
800A  52                ;A7 - A0
800B  82                ;A15 - A8
```

Aslında, yukarıdaki yönergeler sıralarıyla 8250H ve 8251H adresli bellek yerlerin içerikleri üzerine „OVE” mantıksal işlemini gerçekleştiriyoruz ve elde edilen sonucu 8252H adresli bellek yerinde yazıyoruz.

7.5.4. Döndürme Yönergeleri

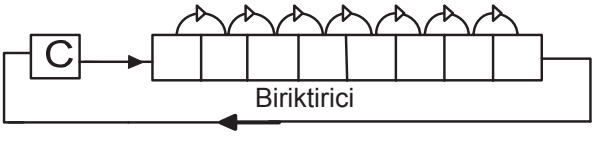
Döndüğü yönüne göre, iki tür döndürme olabilir: sola doğru döndürme ve sağa doğru döndürme. C bayrağının açık olup olmadığına bağlı olarak, döndürme C bayrağının aracılığıyla ya da onsuz yapılabilir. Tüm döndürme yönergelerinde biriktiricin içeriği döndürülüyor ve onun dışında hiçbir başka yazmaca döndürme uygulanmıyor. Sağa döndürmek için iki yönerge vardır.

ROL(Rotate Left) – Sola doğru döndürme tüm bitlerin sola kayması demektir ve en solda bulunan en değerli bit dönerek en az değerli, sağ taraftaki bitin yerine yerleşiyor. Bu yönerge resim 7.10.'da gösterilmiştir.



Resim 7.10. Biriltiricinin sola döndürülmesi

RRC (Rotate Right Carry) - C (carry) bayrağı aracılığıyla sağa doğru döndürme, bitlerin bir pozisyon için sağa kayması ve en sağda bulunan en az değerli bit C bayrağına girmesi, C bayrağının değeri ise sol tarafta en değerli biti yerine götürülmesi demektir. Sağa döndürme resim 7.11.'de gösterilmiştir.



Resim 7.10. Biriltiricinin sağa döndürülmesi

Örnek 7.14:

```
8140  DB   IN 01H       ;baytı giriş aygıtında biriktiriciye
                        ;aktar
```

```
8141  01                ;Giriş bağlantı noktanın ara birim adresi
8142  0F    RRC         ;Biriktiricinin içeriğini sağa döndür.
8143  D3    OUT 02      ;Biriktiricinin içeriğini 02H arabirim
                        ;adresli aygıta aktar.
8143  02                ;Ara birim adresi
```

7.5.5. Atlama Yönergeleri (Jump - atlama)

Atlamalar koşullu ve koşulsuz olabilir. Mikroişlemci koşulsuz atlama yönergeyi aldığı zaman, yönergede yazılmış olan bellek yerinin adresine doğrudan atlıyor. Koşullu atlamalarda, atlamanın gerçekleşmesi için atlamadan önce bazı koşulun yerine getirilmesi gerekiyor (örneğin, bayraklardan biri yüksek seviyede olsun).

JMP adr

İşlev	Yönergede verilmiş adres program sayacında kopyalanıyor
İşlem kodu	11000011
Bayt sayısı	3 bayt
Adresleme şekli	Dolaysız adresleme

Örnek 7.15:

```
8100  3E    MVI A, 01H  ;Biriktirici 01(16)=00000001(2) verisiyle
                        ;dolduruluyor
8101  01                ;Veri
8102  D3    OUT 02H     ;Biriktiricinin içeriği çıkış
                        ;aygıtına kopyalanıyor.
8103  02                ;Giriş bağlantı noktanın ara birim adresi
8104  07    ROL         ;Biriktiricinin içeriği sola doğru
                        ;döndürülüyor
8105  C3    JMP 8102 H  ;8102 adresli konumuna atlama
                        ;(geri dönüyoruz)
8106  02                ; A7 - A0
8107  81                ; A15 - A8
```

Yukardaki program sırası 8 alandan oluşan görüntü biriminde „hareketli ışık” etkisini elde etmek için kullanılıyor.

JZ adr (JUMP if ZERO - sıfır bayrağının değeri bir ise atla - koşullu atlama)

İşlev	Yönergede verilen adres, sıfır bayrağın değeri 1 ise program sayacına kopyalanıyor.
İşlem kodu	11001010
Bayt sayısı	3 bayt
Adresleme şekli	Dolaysız adresleme

Örnek 7.16:

```
8200  7C    MOV A, H      ;H yazmacının içeriğini biriktiricide
                          ;kopyala.
8201  95    SUB L         ;L yazmacın içeriği biriktiriciden
                          ;çıkarılıyor.
8202  CA    JZ 8266H      ;Sonuç sıfır ise, 8266 adresine
                          ;atla.
8203  66                  ;A7 - A0
8204  82                  ;A15 - A8
8205  XX
```

Yukardaki program dizisinde yazmaçların içerikleri arasında kıyaslama yapılıyor ve iki yazmacın içeriği aynıysa, atlama gerçekleşiyor.

JNZ adr (jump if not zero)

İşlev	Sıfırdaki bayrak alçak seviyedeyse atlama gerçekleşiyor
İşlem kodu	11000010
Bayt sayısı	3 bayt
Adresleme şekli	Dolaysız adresleme

Örnek 7.17:

```
8300  26    MVI H,47H     ;H yazmacını 47(16) değeriyle doldur
8301  47                  ;veri (değer)
8302  25    DCR H         ;yazmacın içeriğini bir için azalt.
8303  C2    JNZ 8302H     ;Elde edilen sonuç hâlâ sıfır değilse,
                          ;8302H adresli bellek yerine geri dön
8304  02                  ;A7 - A0
8305  83                  ; A15 - A8
```

Yukardaki program dizisi, aslında her adımda değeri bir için azalan program sayacıdır.

7.5.6. Alt Programlarla Çalışma Yönergeleri

Alt program belli bir işlev gerçekleştiren bağımsız bir program bütünüdür. Bir alt program birkaç ana programda kullanılabilir.

RET – RET yönergesi çağrılan alt programın sonunu ve ana programa geri dönmeyi belirtiyor.

HLT (halt) – Bu yönerge, mikroişlemcinin programın çalışması için doğru şekilde hazırlanması için programda son yönerge olmalıdır. Bu yönerge olmasa, mikroişlemci

sıradaki bellek yerlerinin de içeriklerini yorumlamaya çalışacak ve böylece karışıklığa yol açılabilir.

CALL adr

İşlev	Bu yönerge yardımıyla ana program alt programı mikroişlemcide çalıştırılması için çağırıyor. Yönergede verilen adres alt programın başladığı ilk bellek yerin adresini tanımlıyor.
İşlem kodu	11001101
Bayt sayısı	3 bayt
Adresleme şekli	Dolaysız adresleme

Örnek 7.18:

```
Ana program
8000  26      MVI H,33H       ;H yazmacını 33H verisiyle doldur
8001  33                      ;veri
8002  2F      MVI L,8FH       ;L yazmacını 8FH verisiyle doldur
8003  8E                      ;veri
8004  CD      CALL 8081H      ;başlangıç adresi 8300H olan
                              ;alt programı çağır
8005  10                      ;A7 - A0
8006  80                      ;A15 - A8
8007  32      STA 8300H       ;Elde edilen sonucu 8300H yerinde
                              ;yazdır
8008  00                      ;A7 - A0
8009  83                      ;A15 - A8
800A  76 HLT                  ;Programın sonu
800B  XX

Alt program
8010  7C      MOV A,H         ;yazmacın içeriği biriktiricide
                              ;kopyalanıyor.
8011  85      ADD L           ;Biriktiricinin içeriğini L yazmacının
                              ;değerini ekle.
8012  C9      RET             ;Ana programa geri dön.
```

7.5.7. Yığıt Bellekle Çalışma Yönergeleri

Yığıt bellekle çalışma yönergeleri ana programdan alt programa geçiş yapmak istediğimiz zaman kullanılıyor. Biriktirici ve durum yazmacı, alt program tarafından kullanılmaya sunuluyor. Alt programın çalışmaya başlamadan önce, bu iki yazmacın değeri korunmalıdır.

Çünkü mikroişlemcinin ana programa döndüğü zaman bu yazmaçların değerleri gerekli olacak. Biriktirici ve durum yazmacının değerleri yığıtın tepesinde yerleşiyor. Mikroişlemci alt programın çalışıtırmasını tamamlayınca, onlar yeniden mikroişlemciye geri verilecek.

PUSH PSW – Biriktiricinin ve durum yazmacının değerleri yığıtın tepesinde yerleşiyor

POP PSW – Biriktiricinin ve durum yazmacını korunan değerleri yığıtın tepesinden mikroişlemcinin yazmaçlarına gönderiliyor.

Örnek 7.19:

```
823C  F5     PUSH PSW        ;Biriktiricinin ve durum yazmacın
                             ;içeriklerini yığıtın tepesinde koru;
823D  CD     CALL 82E0H      ;Başlangıç adresi 82E0H olan
                             ;alt programı çağır.
823E  E0                     ;A7 - A0
823F  82                     ;A15 - A8
8240  F1     POP PSW         ;Biriktirici ve durum yazmacın
                             ;içeriklerini geri ver.
```

7.6. 8085 Mikroişlemci için Programların Yazılması

Her yönergeden sonra, **genel yazmaçlardan biri kendi içeriğini değiştiriyor**. Yönergelerin çoğunda, biriktiricinin (LDA,STA, ADD, SUB, ANA, ORA, CMA, ROL, RRC) içeriği değişiyor, ancak başka bir yazmaç (INC r, DCR r, MOV r1,r2 MVI r,data) da olabilir. Aşağıda verilmiş örneklerde, her çalıştırılan yönergeden sonra, kullanılan genel yazmaçların içerikleri tespit ediliyor.

Örnek 7.20: A ve B yazmaçları için içerikleri verilmiştir, A=76H ve B=D4H. Aşağıda verilmiş yönergelerin çalıştırılmasından sonra, yazmaçların içeriklerini hesapla:

```
AND B
CMA
DCR B
```

Çözüm:
A=01110110B ve B=11010100H yazmaçların içeriklerine VE mantıksal işlemi uygulanıyor.

AND B
```
01110110 — A
11010100 — B
--------
01010100 —  Yeni A
```

Bu işlemden sonra yazmaçların içerikleri şöyle olacak: A=01010100B, B=11010100B (B yazmacı aynı kalıyor)

A=01010100B yazmacının içeriği için birinci tamamlayıcı hesaplanıyor.

CMA $\overline{01010100}$=10101011 =A

Bundan sonra yazmaçların içerikleri: A=10101011B ve B=11010100B olacak

---

B=11010100B yazmacının değeri bir için azalıyor

DCR B
```
 11010100 — B
-        1
--------
 11010011 —  Yeni B
```

Son çözüm: A=100101011B, B=11010011B

Örnek 7.21: Verilen yönergelerin çalıştırılmasından sonra A ve E yazmaçlarının içeriklerini hesapla!

```
MVI A,A3H
MVI E,9EH
SUB E
```

Çözüm:
Başlangıçta yazmaçların içerikleri bilinmiyor. MVI yönergesinin yardımıyla yazmaçlara belli değerler veriliyor.

```
MVI A, A3H
MVI E,9EH
```

A=A3H=10100011B, E=9EH=10011110B

A yazmacının içeriğinden E yazmacın içeriği çıkarılıyor.

SUB E
```
 10100011 — A
-10011110 — E
--------
 00000101 —  Yeni A
```

Çözüm: A=00000101B, E=10011110B (aynı kalıyor)

Örnek 7.22: Aşağıdaki program bölümünün çalıştırılmasından sonra program sayacının değerini hesapla. Verilen program bölümün başlamasından önce program sayacının değeri 8000H'dır.

```
MVI A,65H
MVI B,D8H
ORA B
STA 1234H
```

Çözüm:
Bu örnekte uygulanan yönergelerin, yönerge biçimlerin bilinmesi önemlidir. Mevcut yönerge birinci biçimden ise, o zaman sıradaki yönergenin adresinin elde edilmesi için program sayacının değerine bir ekleniyor, Mevcut yönerge ikinci biçimden ise, o zaman iki ekleniyor, üçüncü biçimdense, üç ekleniyor.

Program sayacının içeriği	Yönerge	Yorum
8000H	MVI A,65H	Bu yönerge ikinci biçimdemdir. Program sayacının içeriğine iki eklşyoruz. 8000+2=8002H
8002H	MVI B,D8H	8002+2=8004H
8004H	ORA B	Bu yönerge birinci biçimdendir. Bir ekliyoruz. 8004+1=8005H
8005H	STA 1234H	Bu yönerge üçüncü biçimdendir. Üç ekliyoruz. 8005+3=8008H
8008H		

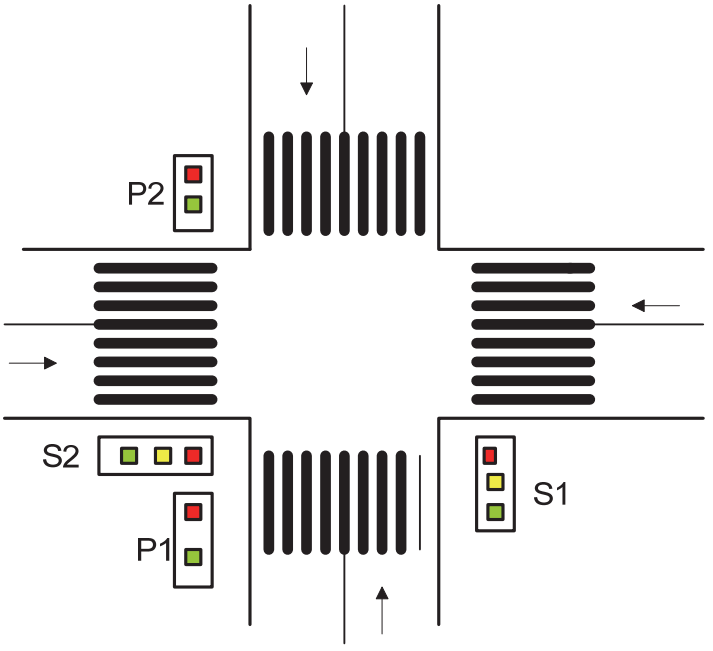
Verilen program bölümünün çalıştırılmasından sonra, program sayacının değeri 8008H olacak.

7.6.1. Basit Bir Kavşakta Trafiğin Düzenlemesi için Yazılım

Trafik ışıklarda ışıkların hangi dinamikle yandığını hepimiz görmüşüz. Kırmızı ışıktan sonra beraber olarak kırmızı ve sarı ışık yanıyor, ondan sonra ise yeşil ışık yanıyor. Yeşil ışıktan sonra sadece sarı ışık yanıyor,ondan sonra da kırmızı ışık yanıyor. Yayalar için trafik ışıklara gelince, yeşil ışık sadece otomobiller için kırmızı ışık yanarken yanıyor. Tüm kavşağın trafik ışıklarla sinyalleşmesi için resim 7.12.'de gösterilmiş sisteme aynı bir sistem çapraz tarafta yerleşiyor.

Kavşakta trafiği düzenleyen yazılım için kullanılan program iki bölümden oluşuyor. Birinci bölüm 6B büyüklüğündedir ve **çıkış bağlantı noktasında taşınması ge-**

reken verileri içeriyor. Çıkış bağlantı noktaların pinleri trafik ışığın ışıklarıyla bağlıdır ve pine mantıksal 1 değeri gelince, ışık yanıyor. LIGHT_LOC sembolik ismi ile tanımlanan, birinci verinin bulunduğu birinci konumun adresinin etiketidir.



Resim 7.12. Trafik ışığında ışıkların görünümü ve onların işaretlenmesi

Not: Eğer sizin simülasyonunuz etiketler kullanımını desteklemezse, o zaman adresi 16 - bitli sayı şeklinde ya da dört rakamlı on altılı sayı olarak verebilirsiniz.

Trafik ışıkları ve yayalar	T_1	T_2	Y1	Y2	
Işık renkleri	KSY	KSY	K	K	
Light_loc:	001	100	1	0	(1min)
	010	100	1		(3sec)
	100	110	1		(3sec)
	100	001	0		(1min)
	100	010	1		(3sec)
	110	100	1		(3sec)

Programın **ikinci bölümü**, birinci bölümden verilerin çıkış bağlantı noktalarına aktarılması için **yönergeleri içeriyor**. Başlangıçta, iki MVI yönergeyle veriler bölümünden birinci baytın adresi HL yazmaç çiftinde giriliyor. MOV A,M yönergesiyle HL çiftindeki adres, belleğe gönderiliyor, aranan bayt bulunuyor ve biriktiricide yerleşiyor. CALL GECİKME 1 dak. yönergesiyle, biriktiricinin içeriği bir dakika değişmeden kalıyor ve o kadar zaman kırmızı ve yeşil ışığın yanması gerekiyor. Bu alt program çağrılmazsa, biriktiricinin içeriği çok çabuk değişecek (birkaç ns'den daha az zaman içinde) ve trafik ışığında ışıkların yanması için zaman yeterli değildir. Sonraki adım-

da (İNX H), yazmaç çiftin içeriği bir için artıyor ve bu şekilde veri bölümünden ikinci bayt 01010011'in adresini elde ediyoruz. Bellekten biriktiriciye ve ondan sonra çıkış bağlantı noktasına veri aktarma süreci altı kez tekrarlanıyor (veriler bölümünden her bayt için ayrı ayrı). Sonunda JUMP CEVIR yönergesiyle tüm bunlar yeniden tekrarlanıyor, yani T1 trafik ışığında yeşil ışığı yanacak, T2 trafik ışığında kırmızı ışık yanacak.

```
çevir:  MVI     H,LIGHT
        MVI     L,LOC
        MOV      A,M
        OUT     PORT A
        JNX     H
        MOV     A,M
        OUT     PORT A
        CALL    GECIKME_3sec
        .
        _______
        INX     H
        MOV     A,M
        OUT     PORT A
        CALL    GECİKME_3sec
        JUMP    CEVİR
```

7.6.2 Kodların Kıyaslanması

Bilgisayarın çıkış bağlantı noktasında 8 led diyot bağlanmıştır. Diyotlar yan yana dönüşümlü olarak yeşil ve kırmızıdır.

```
RGRGRGRG        G - green
                R - red
```

Diyotlar, pinlerine mantıksal birler gönderilince yanıyorlar. Bilgisayarın giriş bağlantı noktasında paralel olarak 8 anahtar bağlıdır. Anahtar kapalı olunca, pin tabloyla bağlanıyorlar ve mantıksal sıfır durumumuz var, anahtar açık olunca ise pin elektrik kaynağıyla bağlanıyor ve mantıksal bir durumu var. Giriş bağlantı noktasından 8 - bitli kombinasyon, 8250 adresi olan bellek yerinin içeriğiyle kıyaslanıyor. İki bayt eşitse, o zaman yeşil ışıklar yanıyor, eşit değilse kırmızı ışıklar yanıyor. Kıyasma iki baytınm çıkarmasıyla gerçekleşiyor. İki sayı eşitse, o zaman sonuç sıfırdır ve ZERO bayrağı aktifleştiriliyor.

```
LDA 8250        ;8250 adresli yerden veri
                ;biriktiriciye gönderiliyor.
```

```
MOV H,A          ;biriktiriciden veri koruma amacıyla
                 ;H yazmacına aktarılıyor. Sıradaki yönergeyle
                 ;biriltiricinin içeriği değişebilir ve
                 ;veri kaybolabilir.
8204   IN 01H    ;Veri biriktiriciden 01H adresli
                 ;bağlantı noktasına gönderiliyor.
       SUB H     ;Kıyaslama başarılıysa ZERO
       JZ 8211   ;bayrağı aktifleştiriliyor ve 821 adresli
       MVI A,AA  ;yere atlanıyor. Yeşil diyotlar yanıyor ve program
       OUT 02    ;tamamlanıyor.
       JMP 8204  ;Kıyaslam başarılı değilse, bayrak alçak
8211   MVI A,55  ;seviyede kalıyor, atlama gerçekleşmiyor
       OUT 02    ;ve atlama yerine sıradaki yönerge MVI A,AAH
                 ;yönergesi çalıştırılıyor. AAH=10101010B sayısı
                 ;biriktiriye,ondan ise çıkış bağlantı noktasında aktarılıyor.
                 ;10101010 sayısında birlerin, çıkış bağlantı noktalarındaki
                 ;kırmızı diyotlarla aynı pozisyonları var. Kırmızı Led diyotlar
                 ;yanınca 8204 koşulsuz atlama gerçekleşiyor ve giriş
       RST7      ;anahtarlarından yeni kombinasyon getiriliyor.
```

7.7. 8085 Mikrobilgisayar için Tümleşik Bileşenler

8085 mikroişlemcinin piyasaya çıkmasıdan sonra, “İntel” şirketi 8085 tümleşik mikroişlemciyle tamamen uyumlu olan yeni tümleşik bileşenler tasarladı. Bu tümleşik bileşenler mikroişlemci ve dış aygıtlar arasına iletişimde aracılık yapıyorlar. Tümleşik bileşenler aynı zamanda denetimci olarak da biliniyor, çünkü dış aygıtlardan mikroişlemciye doğru ve ters yönde veri aktarımını kontrol ediyorlar. Mikroişlemci ve dış aygıtların doğrudan bağlanması son derece kullanışsız olurdu. Arabirim bileşenleri programlanamaz ya da programlanabilir olabilir. **Programlanabilir** bileşenler birkaç baytlık bellek içeriyor (bir ya da fazla) ve onların içeriğini programcı belirliyor. Örneğin, arabirim bileşenden A bağlantı noktasının sekiz pini giriş ya da çıkış pini olacağını belirleyen bit vardır (giriş için bir, çıkış için sıfır). Giriş aygıtların mikroişlemciyle bağlanması sırasında veri arabelleklemeyi sağlaması gerekiyor. Arabellekleme sadece IN yönergesi sırasında dış aygıtın mikroişlemcinin veri veriyoluyla bağlanması demektir. Arabellekler programlanabilir arabirim bileşenlerin parçası olabilir ya da ayrı tümleşik devreler olabilir. Çıkış aygıtların mikroişlemciyle bağlanması sırasında, verilerin süresini OUT yönergesinin süresinden çok daha fazla uzatmasını sağlayan mandallar kullanılıyor. Yönergenin 100ms’den daha az sürdüğü biliniyorsa bu süre-

çin gerekli olduğu ortaya çıkıyor. Mandal genelde arabirim denetimcilerin içerisinde bulunuyor.

7.7.1. 8212 Arabirim Bileşeni

8212 basit, geniş kullanımlı, programlanamaz bileşendir ve her 8085 mikroişlemci sisteminde kullanılıyor. Bu tümleşik devre başka sistemlerde de yerel gerekçeler için kullanabiliyor. 8212 bileşeni tek yönlüdür, yani veriler sadece soldan sağa aktarabilir.

8212 bileşenin pinleri

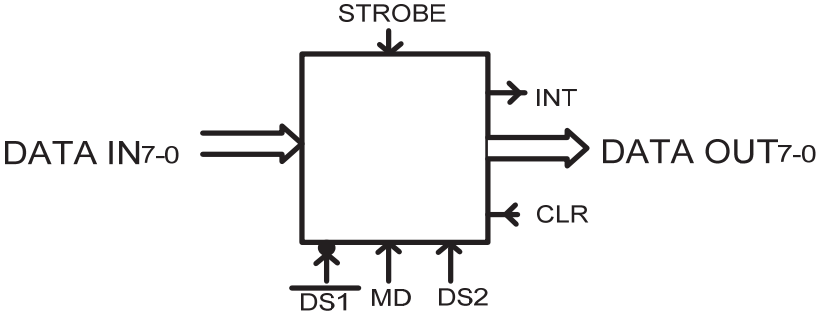
Resim 7.13.'te 8212 bileşenin pin - diyagramı gösterilmiştir.

- Sekiz DI (Data In) pini mikroişlemciden ya da dış aygıtlardan verilerin girmesi için kullanılıyor, DO (Data Out) pinleri ise 8212 bileşenden verilerin çıkması için kullanılıyor.

- STB (Strobe) giriş pinidir ve onun aracılığıyla mikroişlemci ya da dış aygıtı 8212 bileşeni için veri aktarımını anons ediyor. Eğer bileşen anonslu düzende çalışıyorsa, o zaman STB=1 olmadan giriş veri hatları veri alamayacak.

- CLR(Clear) 8212 bileşenin içeriğinde sekiz atışlayan flip - flopların sıfırlanamsı için kullanılıyor.

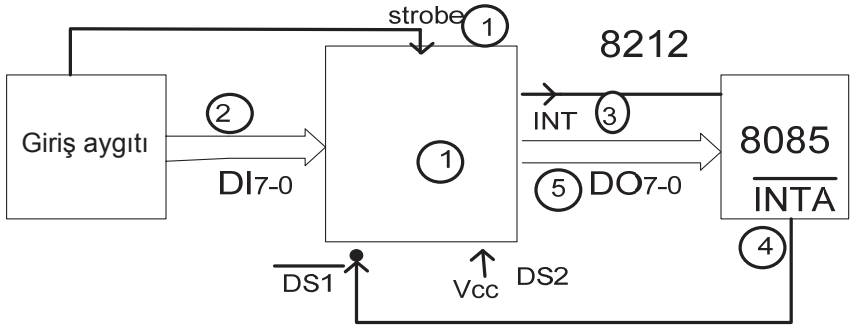
- INT çıkış pinidir ve 8085 mikroişlemcide kesinti üretmek için kullanılıyor.



Resim 7.13. 8212 bileşenin pin diyagramı

- MD giriş pinidir ve çalışma düzeninin seçimi için kullanılıyor. MD pini yüksek seviyede olduğu zaman, 8212 bileşeni tipik **arabellek** olarak çalışıyor. Veriler bileşenden geçişli olarak hiç orada kalmayarak doğrudan geçiyor. Veriler 8212 bileşeninde DATA İN aracılığıyla girer girmez DATA OUT'tan çıkabilirler. MD alçak seviyede olduğu zaman, 8212 bileşeni **mandal** olarak çalışıyor. Veriler geçici olarak D flip - floplarında korunuyor. STB sinyalin yardımıyla, verilere DI_7 - DI_0 pinleriyle erişiliyor, DS1 ve DS2 pinlerin aktifleştirilmesinden sonra ise veriler okunuyorlar.

- Çıkış very bağlantı noktaların etkinleştirilmesi için DS1 ya da DS2 pinlerden biri kullanılıyor. DS1 pini mantıksal sıfır durumunda aktiftir, DS2 pini ise mantıksal bir durumunda aktiftir. Mikroişlemci kesinti onayını DS1 pinine gönderirse, o zaman DS2 V_{CC} elektrik kaynağıyla bağlanmalıdır, eğer mikroişlemci kesinti onayını DS2 pinine gönderirse, o zaman DS1 tabloya bağlanmalıdır.



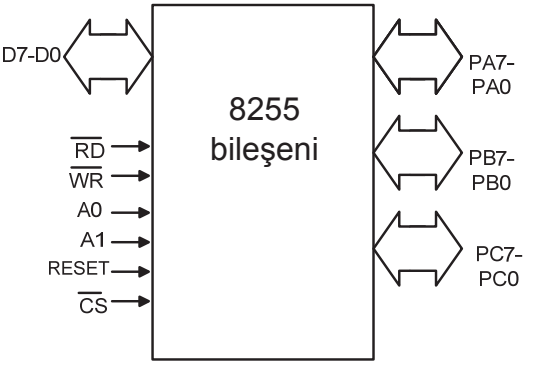
Resim 7.14. 8212 bileşenin veri gönderme sıralaması

8212 giriş bağlantı noktası olarak çalıştığı zaman (bir giriş aygıttan veriler alarak, mikroişlemciye gönderiyor), sinyallin gönderilmesi şu sıralamayla gerçekleşiyor. Giriş aygıtı anons (strobe) pinini aktifleştiriyor. Alınan anonstan sonra, 8212 DATA IN giriş veri kapısını açıyor ve veriler 8212'ye giriyor. 8212 'de veriler, INT pininin aktifleşerek 8212'nin mikroişlemiden kesinti aramasına kadar kalıyor. Mikroişlemci kesintiye izin verirse, o zaman çıkış veri kapısı (DATA OUT) etkinleştiriliyor ve veriler mikroişlemciye ulaşıyor. Resim 7.14.'te 8212 denetimcinin giriş kapısı olarak çalıştığı zaman bağlanması gösterilmiştir. Çemberlerle sinyallerin **gönderilme sıralaması işaretlenmiştir.**

7.7.2. 8255 Programlanabilir Bileşeni

8255 programlanabilir arabirim bileşenidir. Onun yazılım programları çalıştırması için mikroişlemcisi ve RAM belleği yoktur, ancak içeriği programcı tarafından belirlenen komut yazmacı vardır. 8255 bileşeni, Pentium mikroişlemci sistemler gibi, birçok mikroişlemci sistemlerinde kullanım buluyor. 8255 denetimcisi kişisel bilgisayarların klavye ve yazıcıyla bağlanması için kullanılıyor.

Resim 7.15.'te 8255 denetimcinin **pin - diyagramı** verilmiştir.



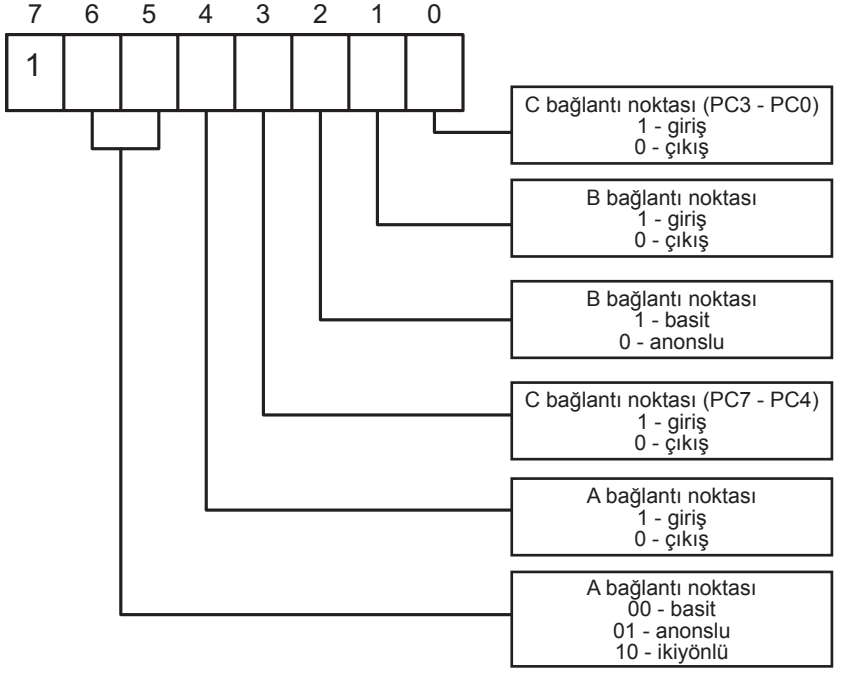
Resim 7.15. 8255 bileşenin pin diyagramı

PA, PB ve PC 8 - pinli bağlantı nokatalarıdır ve 8255'in dış aygıtlarla bağlanması için kullanılıyorlar. 8255'in mikroişlemciyle bağlanması için veri pinleri kullanılıyor (D_7 - D_0). İşlem seçimi için (okuma ya da yazma) $\overline{RD}$ ve $\overline{WR}$ kontrol pinleri kullanılıyor.

Mikroişlemcinin 8255 bileşenine erişmesi için $\overline{CS}$ pinin aktifleştirilmesi gerekiyor. Bu sinyalın adres kod çözümleme süreciyle üretildiğini hatırlayalım. 8255 bileşenine erişildiğinden sonra iç yazmaçlardan birinin seçimi yapılıyor. İç yazmaçlar deyince, denetimcinin üç bağlantı noktasına ve komut yazmacına düşünülüyor. Bağlantı noktaları dış aygıtlardan okumak ya da onlarda yazmak için kullanılıyorlar, komut yazmacı aracılığıyla ise denetimcinin programlanması gerçekleşiyor. İç yazmacın seçimi A_1 ve A_0 adres pinlerinin yardımıyla yapılıyor. Buna göre, mikrodenetimci dört adres kullanacak. 8255 bileşeni 8085 mikroişlemciyle bağlanırsa, o zaman A_{15}'ten A_2'ye kadar bitler her dört adres için aynı olacak ve onlar aslında, $\overline{CS}$ sinyalinin üretiminde yer alacak. İç yazmaçların ayırımı yapılsın diye, A_1 ve A_0 bitleri her dört adreste farklı olacak. Örneğin, mikroişlemcinin veri veriyoluyla gönderdiği bitlerin nerede yazılacağı, A_1 ve A_0 bitlerin durumuna bağlıdır.

A1	A0	
0	0	A bağlantı noktası
0	1	B bağlantı noktası
1	0	C bağlantı noktası
1	1	komut yazmacı

8255 bileşeni komut yazmacı aracılığıyla programlanıyor. **Komut yazmacı** 8 - bitli yazmaçtır ve programcı onun içeriğini belirleyerek, denetimcinin bağlantı noktaların hangi düzende çalışacaklarına karar veriyor. Dört aktarma düzenin varolduğunu hatırlayalım: basit, anonslu, tokalaşma düzeni ve çift tokalaşma düzeni.



Resim 7.16. 8255 bileşeninden komut yazmaç bitlerinin işlevsel açıklaması

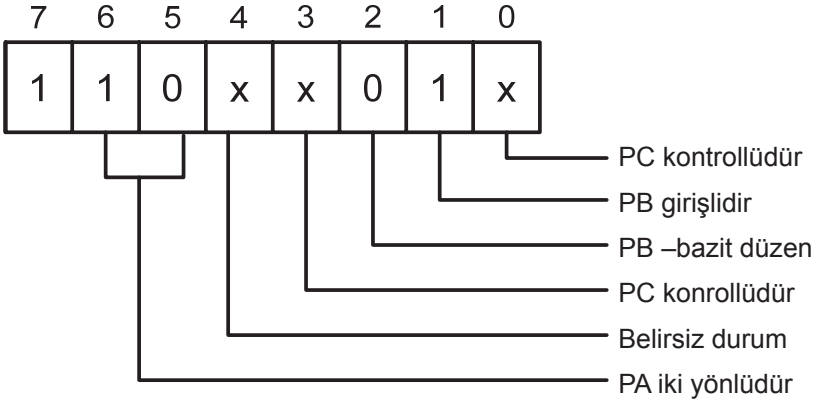
A bağlantı noktası giriş, çıkış ya da iki yönlü bağlantı noktası olabilir ve basit ya da anonslu düzende çalışabilir. B bağlantı noktası giriş ya da çıkış bağlantı noktası olabilir (iki yönlü olamaz) ve basit ya da anonslu düzende çalışabilir.

C bağlantı noktası kullanıcı ya da kontrol bağlantı noktası olabilir. C bağlantı noktası sadece A ve B bağlantı noktaları basit düzende çalıştığı zaman kullanıcı bağlantı noktası olabilir. O zaman PC pinlerin nasıl olacağını, yani giriş ya da çıkış pinleri olacağını seçebiliriz. A ya da B bağlantı noktalardan biri basit düzende çalışmıyorsa, o zaman C bağlantı noktası kontrol bağlantı noktası olabilir ve onun pinleri anons ve onay sinyallerin aktarımı için kullanılıyor. Resim 7.16.'da komut yazmacın içeriği ve her bitin anlamı gösterilmiştir. C bağlantı noktası kontrol bağlantı noktası olursa, o zaman komut yazmacında üçüncü ve sıfırıncı bitin 0 ya da 1 olması önemli

değil. O zaman onları X ile işaretliyoruz ve belirsiz durumda olduklarını diyoruz. Aynı zamanda PA iki yönlüyse dördüncü bit de X olacak.

Örnek 7.23: A bağlantı noktası iki yönlü, B bağlantı noktası girişli ise ve basit düzende çalışıyorsa, 8255 bileşenin komut yazmacının içeriğini belirle.

Çözüm:



Resim 7.17. A bağlantı noktası ikiyönlı ve B bağlantı noktasının girişli ve anonslu olduğu zaman komut yazmacın içeriği

C bağlantı noktası kontrol bağlantı noktasıdır, çünkü A bağlantı noktası basit düzende çalışmıyor. C bağlantı noktası kontrol **bağlantı noktası** gibi kullanıldığı zaman, pinleri, kullanıcı bilgi yerine kontrol sinyalleri gönderecek. Bu sinyaller, A ve B bağlantı noktalarından ve veri veriyolundan kullanıcı verilerin aktarılacağını anons eden sinyallerdir. Aktarma anonsu için kontrol sinyalleri dışında, gönderilen verilerin alındığını onaylayan kontrol sinyalleri de vardır. Tablo 7.2.'de A ve B bağlantı noktaların tüm çalışma düzeni için C bağlantı noktasının tüm kontrol sinyallerin işaretleri verilmiştir.

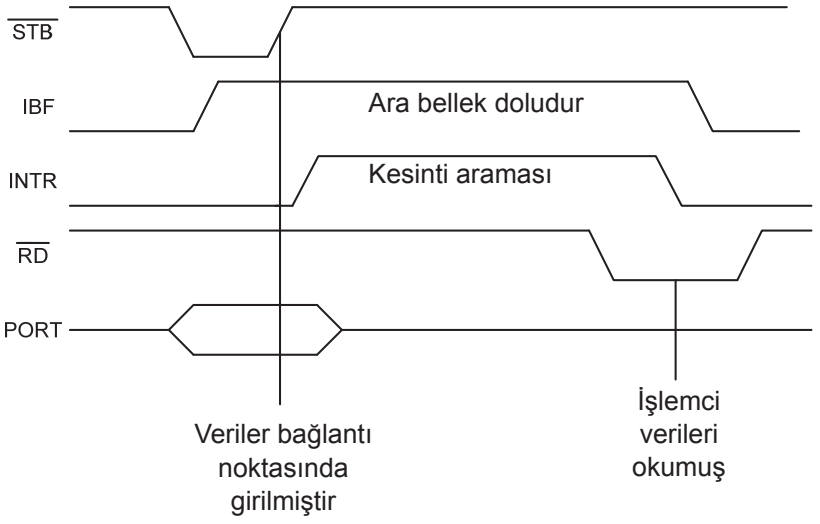
Pin	Anonslu düzen		İki yönlü
	Giriş	Çıkış	
0	$INTR_B$	$INTR_B$	$INTR_B$
1	IBF_B	OBF_B	IBF_B
2	$\overline{STB}_B$	$\overline{ACK}_B$	$\overline{STB}_B$
3	$INTR_A$	$INTR_A$	$INTR_A$
4	IBF_A	/	STB_A
5	$\overline{STB}_A$	/	IBF_A
6	/	$\overline{ACK}_A$	ACK_A
7	/	$\overline{OBF}_A$	$\overline{OBF}_A$

Tablo 7.2. C bağlantı noktasından kontrol bağlantı noktası gibi çalıştığı zaman pinlerin işaretlenmesi

A endeskli sinyaller A bağlantı noktasıyla ilgilidir, B endeksliler ise B bağlantı noktasıyla ilgilidir.

- INTR (Interrupt) sinyaliyle 8255 denetimcisi, mikroişlemcinin mevcut çalışmasında kesinti arıyor. Bu kesinti denetimcinin dış aygıtlardan aldığı veriler mikroişlemciye göndermek için gerekiyor. Mikroişlemci kesinti aramasını kabul ederse, 8255 bileşenin $\overline{RD}$ pini aktifleştiriyor.
- STB (strobe) verilerin aktarımını anons eden ve dış aygıtı 8255 bileşenine kadar gönderen sinyaldır. $\overline{STB}$ sinyali gönderilen verilerin alınması için A ya da B bağlantı noktasının pinlerini etkinleştiriliyor.
- IBF (Input Buffer Full), verilerin 8255 bileşende ulaşınca aktifleşen sinyaldır. IBF sinyalini 8255 bileşeni dış aygıtına gönderiyor.
- $\overline{ACK}$ (Acknowledge) sinyali A ya da B bağlantı noktasının çıkış bağlantı noktası olduğu zaman kullanılıyor. $\overline{ACK}$ pini, aygıtın 8255 bileşeninden verileri aldıktan sonra aygıtı aktifleştiriyor.
- OBF (Output Buffer Full) sinyali 8255 bileşenin veri aktarımını anons etmek için dış aygıtına kadar gönderdiği sinyaldır. Bu veriler 8255 bileşeni mikroişlemciden almıştır, ancak çıkış aygıtına kadar göndermelidir.

Resim 7.18.'de A ya da B bağlantı noktalarından biri **giriş bağlantı noktası** olarak çalıştığı zaman sinyallerin zamanlama diyagramı verilmiştir.



Resim 7.18. 8255 giriş bağlantı noktası olarak çalıştığı zaman zamanlama diyagramı

Sinyallerin akifleşme sıralaması şöyledir:

1. Giriş aygıtı $\overline{STB}$ pinini aktifleştiriyor ve A ya da B bağlantı noktasında veri alımı için pinleri etkinleştiriyor.

2. Veriler, A ya da B bağlantı noktaların pinleri aracılığıyla 8255 bileşenine giriyor.

3. 8255 verileri aldıktan sonra, IBF pini aktifleştiriyor ve giriş aygıtına verilerin 8255'e ulaştığını onaylıyor.
4. Ancak, veriler 8255'te kalmamalıdır. Onlar mikroişlemciye gönderilmelidir. Bundan dolayı 8255 INTR pinin aracılığıyla mikroişemciye kesinti araması gönderiyor.
5. Mikroişlemci kesinti aramasını kabul ederse, $\overline{RD}$ kontrol pini aktifleştiriliyor.
6. Veriler veri veriyolu yardımıyla mikroişlemciye ulaşıyorlar.

Sonuçlar:

8085 mikroişlemcisi 8 bitlidir, 8MHz frekansla dijit palsı üreten pals üreticisine, kesintilerle çalışması ve zamanlama kontrolü için özel denetimciye sahiptir. 64KB kapasiteli belleğe erişimi var, yani bellek alanını dolaysız adreslemek için kullanılan 16 adres pini var (2^{16} =64K).

ALE adres - veri veriyolu için kontrol sinyalidir. Eğer ALE=1 ise, o zaman adres - veri veriyolu adres bitleri aktarıyor. Eğer ALE=0 ise, o zaman veriyolu veri pinleri aktarıyor.

8085 mikroişlemcide 5 kesinti türü var: TRAP, RST7.5, RST6.5, RST5.5 ve INTR. TRAP en yüksek öncüllüğü olan kesintidir. Üç sıfırlandırıcı kesintinin aynı, sabit sıfırlandırma adresi var. INTR kesintinin sabit sıfırlandırma adresi yok ve hangi kesinti alt programın çalıştırılacağı, kesinti üreten dış aygıtın türüne bağlıdır.

Üç durum hatı olan SO,S1 ve IO/$\overline{M}$ aracılığıyla mikroişlemci nasıl makine döngüsünün gerçekleşeceği hakkında bilgi veriyor. Altı farklı makine döngüsü ayırıyoruz: bellekte yazma, bellekten okuma, çıkış aygıtına yazma, çıkış aygıtından okuma, işlem kodunun aktarımı ve kesintileri işletilmesi.

Biriktirici mikroişlemcinin çalışma yazmacıdır ve aritmetik mantık biriminin girişinde bulunuyor. Biriktirici dışında, genel amaçlı yazmaçlar grubuna, B, C, D, E, H, L harfleriyle işaretlenen altı yazmaç daha giriyor.

Program sayacı, bellekte geçirilmesi ve işletilmesi gereken sıradaki baytın adresini hafıza eden, 8085 mikroişlemcisinde bulunan yazmaçtır.

Mikroişlemci, bellekten mikroişlemciye getirilen her yeni bayt için program sayacının içeriğine bir ekliyor.

Mikroişlemci, 16 bitli yazmaç olan ve yığıtın tepesinde bellek yerinin adresini içeren özel yazmaç olan yığıt - göstergeci (stack pointer) içeriyor.

Durum yazmacı bayraklar (flags) adı olan durum göstergeler içeriyor. Bayrakların farklı rolleri olabilir ve genelde bazı yönergenin çalıştırılmasından sonra aritmetik - mantık birimin durumunu gösteriyorlar.Tipik bayraklar şunlardır: aktarma bayrağı, çift değer bayrağı, sıfır bayrağı, işaret bayrağı ve yardımcı aktarma bayrağı.

Yazmaçlı dolaysız adreslemede işletilmesi gereken veriler mikroişlemcinin genel yazmaçlarında bulunuyorlar. Yazmaçlı adreslemeyi kullanan yönergelere birinci biçim yönergeleri denir. Onlar bir baytlık bellek alanı, yani bir bellek yeri kapsıyorlar.

Doğrudan adreslemede işlenen yönergenin içinde bulunuyor. Bu yönergeler için ikinci yönerge biçiminden olduklarını diyoruz. İkinci yönerge biçimi iki bayt içeriyor. Birinci bayt her zaman işlem kodudur, ikinci bayt ise işletilmesi gereken sekizbitli veridir.

Dolaysız adresleme şeklinde yönerge istediğimiz veriyi okuyabileceğimiz bellek yerin adresini içeriyor. Bu adresleme şeklini kullanan yönergeler üçüncü yönergeler biçimindendir. Birinci bayt işlem kodudur, ikinci ve üçüncü bayt beraber bellek yerin 16 - bitli adresini tanımlıyor.

Veri aktarımı yazmaç - yazmaç (MOV), biriktirici - bellek yeri (STA), bellek yeri - biriktirici (LDA), biriktirici - çıkış aygıtı (OUT) ve giriş aygıtı - biriktirici (IN) arasında gerçekleşebilir. Çevirici yönergeyle bir bellek yerinden başka bellek yerine doğrudan aktarma gerçekleşemez.

Toplama ve çıkarma aritmetik yönergelerinde bir işlenen biriktiricide bulunuyor, ikinci işlenen ise 8085 mikroişlemcinin genel yazmaçlarından birinde bulunuyor. Elde edilen sonuç biriktiricide yazılıyor ve bu şekilde biriktiricinin eski içeriği kayboluyor. Aynı durum AND ve ORA mantıksal yönergeler için de geçerlidir.

Dönme yönüne göre iki döndürme türü vardır: sola döndürme ve sağa döndürme. C bayrağının açık olup olmadığına bağlı olarak, döndürme C bayrağının aracılığıyla ya da onsuz gerçekleşebilir.

Atlama ve alt program yönergeleri her zaman atlaması gereken ya da alt programın başlatılması gereken yerlerin adreslerini içeriyor.

8212 basit, geniş kullanımlı, programlanamaz bileşendir ve her 8085 mikroişlemci sisteminde kullanılıyor. Arabirim bileşeni giriş bağlantı noktası olarak çalışınca, o zaman anons (strobe) pinin aracılığıyla dış aygıttan aktarma araması alıyor. 8212 bileşeni, INT kesinti pinin aktifleştirilmesiyle mikroişlemcinin çalışmasında kesintiye yol açabilir. 8085 mikroişemcisi kesinti aramasını kabul ederse, o zaman 8212 bileşenin seçim pinlerini, $\overline{DSI}$ ve DS2'yi aktifleştiriyor.

8255 bileşeni komut yazmacı yardımıyla programlanıyor. Komut yazmacı 8 - bitli yazmaçtır ve programcı onun içeriğini belirleyerek, denetimcinin bağlantı noktaları hangi düzende çalışacaklarına karar veriyor. A bağlantı noktası girişli, çıkışlı ya da iki yönlü olabilir ve basit veya anonslu düzende çalışabilir. B bağlantı noktası girişli ya da çıkışlı olabilir (iki yönlü olamaz) ve basit veya anonslu düzende çalışabilir. C bağlantı noktası kullanıcı ya da kontrol bağlantı noktası olabilir.

Sorular ve Ödevler

1. 8085 işlemcisi 8-bitlidir. Açıkla!

2. 8085 mikroişlemcide adres ve veri veriyolun genişliği ne kadardır?

3. ALE (Address Latch Enable) sinyalının adres-veri veriyolun çoğullamasını nasıl kontrol ettiğini açıkla?

4. Kontrol sinyalleri verilen mantıksal seviyede olursa: $\overline{RD}$=1, $\overline{WR}$=0, IO/$\overline{M}$=1, 8085 mikroişlemcisi hangi işlemi gerçekleştirecek?

5. Makine döngüsünün tanımlanması için hangi sinyaller kullanılıyor?

6. Hangi aygıtlar INTR pinini aktifleştiriyor? Hangi aygıt kesinti için izin veriyor?

7. Belleğe doğrudan erişim (Direct Memory Access) kavramını açıkla!

8. 8085 mikroişlemcinin dijit pals pini nasıl işaretleniyor? 8085 mikroişlemcinin dijit palsının frekansı ne kadardır?

9. 8085 mikroişlemcinin tüm özel amaçlı yazmaçlarını say!

10. 8085 mikroişlemcinin genel yazmaçları nasıl işaretleniyor?

11. 8085 mikroişlemcisinde makine döngü türlerini say!

12. Aşağıdaki makine döngüler arasında farklar nedir:
    A) işlem kod aktarımı ve bellekten okuma
    B) Bellekten okuma ve dış aygıttan okuma

13. Sıfırlama kesintisi terimi nerden geldiğini açıkla!

14. Yönerge bellek yerin adresini içerirse hangi adresleme şekli uygulanmıştır?

15. 8085 mikroişlemcisinde, üçüncü yönerge biçiminden ikinci bayt neyi tanımlıyor?

16. Mevcut yönerge ikinci yönerge biçiminden ise ve 4500H adresiyle başlarsa, sıradaki yönergenin başlangıç adresini hesapla.

17. ROR ve RRC yönergeleri arasında farkı açıkla!

18. A=D6H ve C=2AH yazmaçlarının içerikleri verilmiştir. ADD C yönergesinin çalıştırılmasından sonra hangi bayraklar aktifleştirilecek?

19. A= EDH ve L=669H yazmaçların içerikleri verilmiştir. SUB L yönergesinin çalıştırılmasından sonra biriktiricinin içeriğini hesapla!

20. Biriktiricinin içeriği, A = 3AH ve aktarma için durum bayrağının içeriği verilmiştir, C=1. Aşağıdaki yönergelerin çalıştırılmasından sonra onların değerlerini hesapla:
    A) ROL
    B) RLC

21. A=12H ve E=DEH yazmaçlarının içerikleri verilmiştir. Aşağıdaki program sırasının çalıştırılmasından sonra onların içeriklerini hesapla:
    ADD E
    CMA
    MOV E,A

22. 1234H adresi olan bellek yeri 7CH verisini, A=9EH biriktiriciyi ve B=38H yazmacını içeriyor. Aşağıdaki yönergelerin çalıştırılmasından sonra biriktiricinin içeriğini açıkla:
    SUB B
    INC A
    LDA 1234H

23. 8FFFH adresli bellek yeri 4CH verisini, A=12H yazmacını ve L=4DH yazmacını içeriyor. Aşağıdaki yönergelerin çalıştırılmasından sonra biriktiricinin içeriğini hesapla:

```
LDA 8FFFH
ANA L
MOVA,L
STA 8FFFH
```

24. Mikroişlemci sistemlerinde arabirim bileşenlerin ne işlevleri var?

25. 8212 programlanamaz, tek yönlü bileşendir. Açıkla! olarak çalışırsa, 8212 denetimcide hangi pinler DATA IN ve DATA OUT bağlantı noktalarını etkinleştiriyor?

27. 8212 denetimcide MD pini ne için kullanıldığını açıkla!

28. 8255 denetimcinin A, B ve C bağlantı noktaların ve komut yazmacının adreslenmelerini açıkla?

29. 8212 denetimcisinde denetimci ve mikroişlemci arasında veri aktarımı için ve denetimci ile dış aygıtlar arasında veri aktarımında hangi pinler kullanılıyor?

30. 8212 denetimcide C bağlantı noktası ne zaman konrol bağlantı noktası olarak, ne zaman ise kullanıcı bağlantı noktası olarak kullanılıyor?

31. 8255 denetimcide, A bağlantı noktası tokalaşma düzeninde çalışıyorsa ve giriş bağlantı noktasıysa ve B çıkış bağlantı noktası ise ve basit düzende çalışıyorsa komut yazmacının içeriğini belirle!

32. 8255 denetimcisi giriş bağlantı noktası olarak kullanılırsa, DATA, INTR, RD, PORTA, IBF, STB pinlerin aktifleşme sıralamasına göre sırala!

8. 16 ve 32 Bitli Mikroişlemciler

8.1. 8086 Mikroişlemcinin Temel Özellikleri

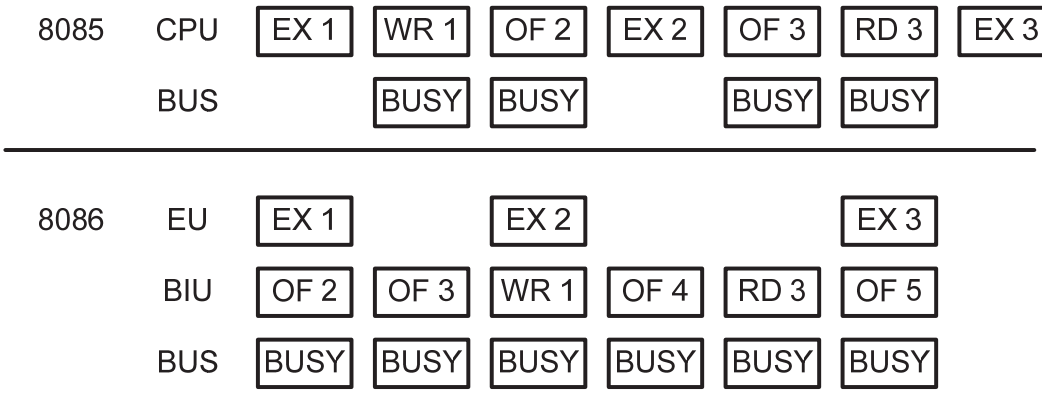
8086/8088 mikroişlemcileri üçüncü nesil mikroişlemcileridir. 8088 belleğe ve giriş - çıkış aygıtlara doğru **8 - bitli veri veriyoluyla** tasarlanmıştır, 8086 ise **16 - bilti veri veriyoluna** sahiptir. Tüm diğer özelliklere göre bu iki mikroişlemci türü aynıdır ve bir mikroişlemci için yazılan yazılım diğerinde de çalışacak. 8086/8088 mikroişlemcilerin bellek kapasiteleri 1MB büyüklüğündedir ve bellek yerlerin adreslenmesi için 20 - bitli adresler kullanılıyor.

Tüm mikroişlemciler programları birer birer yönergenin çalıştırılmasıyla gerçekleştiriyor ve aşağıdaki aşamalardan devamlı geçiyorlar:

- bellekten mikroişlemciye işlem kodunun aktarımı (Opcode fetch)
- gerekirse işlenenlerin aktarımı (READ)
- etkili işletme - çalışma (EXECUTION)
- sonucun belleğe geri dönmesi(WRITE)

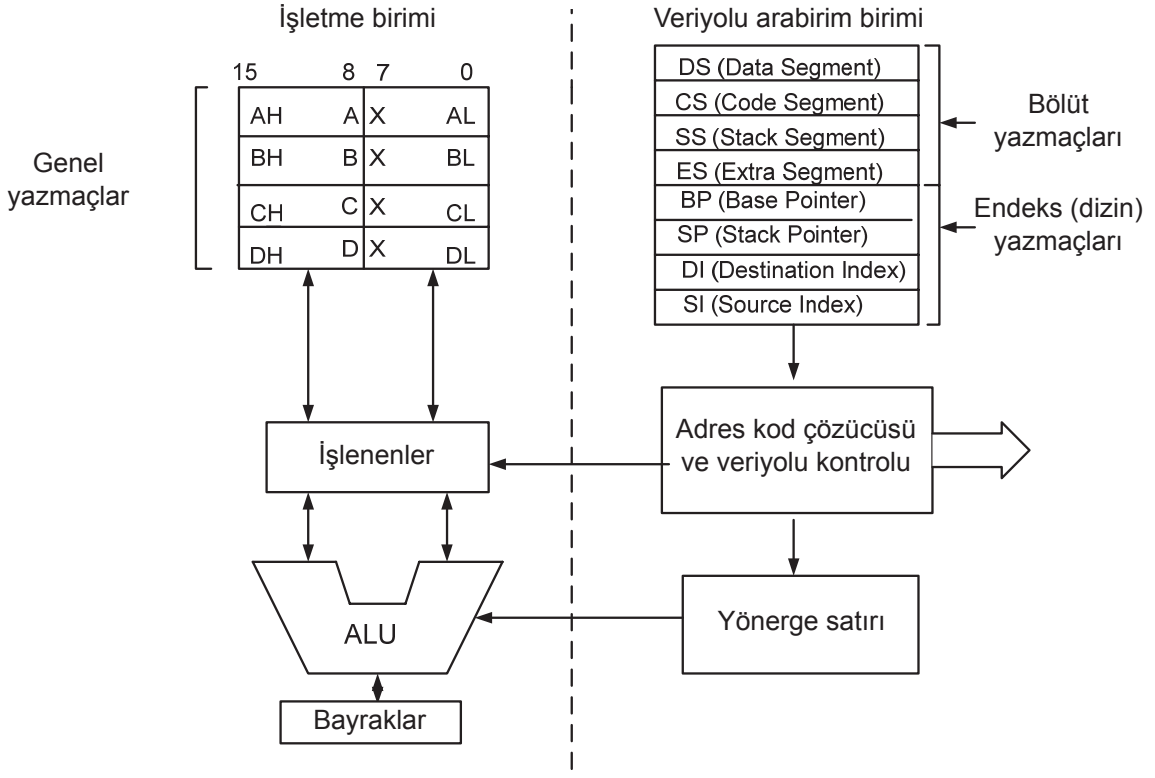
8086 işlemci yukardaki aşamaları paralel şekilde yürüten birimleri içeriyor. İşletme birimi (execution unit) yönergeleri çalıştırıyor, veriyolu arabirim birimi (bus interface unit) ise işlem kodların, işlenenlerin ve sonuçların aktarımını gerçekleştiriyor. Her iki birim aynı zamanda çalışıyor.

Resim 8.1'de 8085 mikroişlemcinin ve 8086/8088 mikroişlemcilerin çalışmaları arasında kıyaslama yapılmış. 8086 mikroişlemcide iki bileşen birimin paralel şekilde çalışması, donanım kaynakların geniş çapta kullanılması ve daha büyük hız sağlanmıştır.



Resim 8.1. 8085 ve 8086 mikroişlemcileri için yönerge aşamaların gerçekleşmeleri arasında kıyaslama

Resim 8.2.'de iki birimin **işlevsel blok modeli** gösterilmiştir.



Resim 8.2. 8086 mikroişlemcinin işlevsel blok modeli

İşletme Birimi

16 - bitli aritmetik - mantık birimi durum ve kontrol bayrakların durumuna etkiliyor ve genel yazmaçları ve yönerge işlenenleri işletiyor. İşletim biriminde tüm yazmaçlar ve iç veriyolları, daha hızlı aktarma hızını sağlamak için 16 bitlidir. İşletim bi-

riminin dış dünyayla hiçbir şekilde bağı yoktur ve yönergeleri veriyolu arabiriminden alıyor. Herhengi bir yönerge belleğe ya da dış aygıtlara erişim aradığında, işletim birimi veriyolu arabirim birimine veri almak ya da göndermek için istek gönderiyor. İşletim birimi 16 - bitli adresler kullanıyor. **Veriyolu arabirim birimi adres yerdeğişimi gerçekleşiyor (fiziksel adres üretiliyor) ve bu şekilde işletim birimine 1MB'a kadar belleğe erişim sağlıyor**.

Veriyolu Arabirim Birimi

Veriyolu arabirim birimi işletim biriminin dış aygıtlarla ve bellekle iletişimi gerçekleştiriyor. İşletim birimi meşgul olunca, veriyolu arabirim birimi ileriye bakıyor ve önceden 4 bayt'a kadar hafıza ediyor. İşletim biriminde çalıştırılması gereken yönergeler, **yönergeler bekleme sırasında** koyuluyor. Veriyolu arabirim birimi yeni yönergenin aktamasını, yönergeler sırasında en az iki bayt serbest olduğu zaman başlatıyor. İşletim birimi program akışını değiştiren yönerge çalıştırırsa (CALL ve JUMP yönergeleri), o zaman veriyolu arabirim birimi yönergeler sırasını sıfırlandırıyor, yeni adresten işlem kodunun aktarımını yapıyor ve işletim birimine sunuyor ve sırayı yeni adreslerle doldurmaya devam ediyor. Giriş - çıkış birimin isteği üzere, veriyolu arabirim birimi yönergeler sırasının doldurulmasını durduruyor.

Yazmaçlar

AX, BX, CX, DX yazmaçlarına **genel yazmaçlar** deniyor. Genel yazmaçlar veri yazmaçları olarak kullanılıyor, yani işletilmesi gereken verileri geçici olarak saklamak için kullanılıyor. Veri yazmaçları, 8 ya da 16 bit farklı büyüklükte verileri saklamak için kullanılabilme özelliği onları benzersiz yapıyor. 8 - bitli veriler saklamak istersek, 16 - bitli yazmaçlar yarıya bölünüyor. 0'dan 7'ye kadar ilk daha az değerli bitler AL, BL, CL, DL olarak işaretleniyor (L - low alçak), 8'den 15'e kadar daha değerli bitler ise AH, BH, CH, DH olarak işaretleniyor (H - high yüksek).

Gösterge ve **endeks yazmaçları** aritmetik ve mantık işlemleri sırasında kullanılıyorlar, ancak aynı zamanda daha geç anlatacağımı fizik adreslerin hesaplanması için de kullanılıyorlar. Devamda genel ve gösterge yazmaçları için kullanılan kısaltmaların anlamları verilmiştir.

A - biriktirici
B - temel (base)
C - sayaç (counter)
D - veri (data)
DI – Destination Index - Hedef endeksi
SI - Source Index – kaynak endeksi
BP - Base Pointer – temel göstergesi
SP - Stack Pointer - yığıt göstergesi

Bölütlendirmek, belleği bölütler olarak adlandırılan parçalara bölmektir. Bölüt ve **bölüt yazmacı** terimleri arasında fark vardır. Bölüt belleğin parçasıdır, bölüt yazmacı ise aranan bölütün başlangıcını bulmaya yardım eden mikroişlemcide bellek konumudur. Bölütün başlangıç adresi, bölüt yazmacın içeriğini 16 sayısıyla çarparak elde ediliyor, yani bölüt yazmacın içeriğine sağ taraftan 4 sıfır ekleniyor. Bölütün en yüksek değeri 2^{16} = 64 KB büyüklüğündedir. **Dört tür bölüt** vardır. Bölütlerin kısaltmaları ve onların anlamları şöyledir:

DS - Data Segment - Veri bölütü
CS - Code Segment - Kod bölütü
ES - Extra Segment - Ekstra bölüt
SS - Stack Segment - Yığıt bölütü

Kod bölütü programın yönergelerini içeriyor. Veri bölütü işletilmesi gereken verileri, işlenenleri içeriyor.Ekstra bölüt dizilerle çalışıldığı zaman kullanılıyor ve bu bölüt tüm diziyi içeriyor. Yığıt bölütü onun en yüksek yerinden doldurulуyor ve boşaltılıyor.

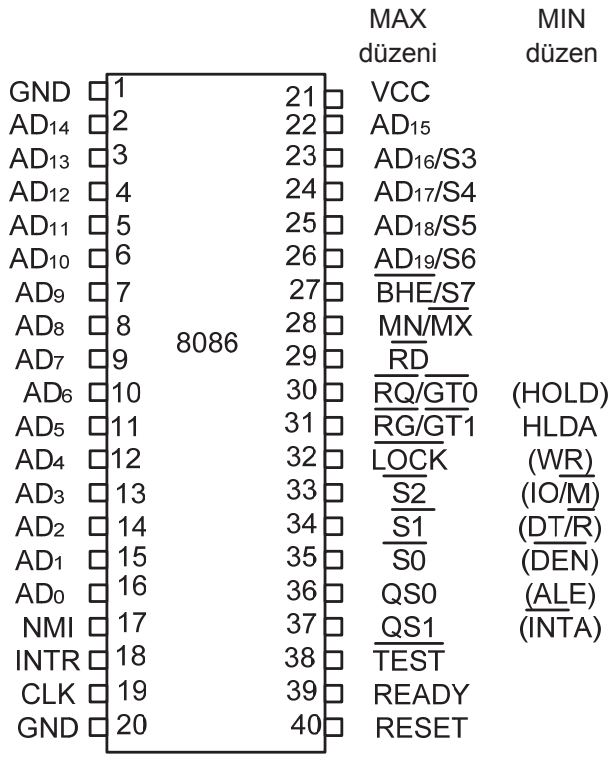
Her programın kendi bölütleri vardır. Kaç programımız varsa, o kadar kod bölütümüz de olacak. Ancak verilen anda sadece mevcut programın kod bölütü aktiftir ve onun başlangıç adresini, kod yazmacın içeriği yardımıyla elde ediyoruz. İşletilen program değişirse, o zaman bölüt yazmaçların içerikleri de değişecek.

8.2. 8086 Mikroişlemcinin Pin Diyagramı

8086 mikroişlemcinin **40 - pinli kasası** var ve +5V'luk elektrik gücü kullanıyor. Resim 8.3.'te 8086 mikroişlemcinin pin diyagramı gösterilmiştir. Programların büyüklüğü ve kompleksliğine bağlı olarak, 8086 mikroişlemci **iki düzende çalışabilir: minimum (enküçük) ve maksimum (en büyük).** Bu iki düzeni ilerideki bölümlerde inceleyeceğiz. 40 pinin bir bölümü, minimum ve maksimum düzeninde farklı işlevleri vardır. Önce iki düzende aynı işlevleri olan pinleri tanıyacağız.

AD15 - AD0 Çoğullu adres veri veriyolu

A19/S6 - A16/S3 Çoğullu adres durum veriyolu. S0 durum biti her zaman yüksek seviyededir, S1 biti kesinti bayrağın (IF - Interrupt Flag) durumunu veriyor. S2 ve S3 bitleri programın dört bölütünden hangisinin verilen anda aktif olduğunu gösteriyor.



Resim 8.3. 8086 mikroişlemcinin pin diyagramı

RD Bu çıkış pini mantıksal sıfırda aktiftir ve bazı dış aygıttan ya da bellekten okuma işlemini başlatıyor.

READY Bu pin alçak seviyedeydse, o zaman mikroişlemci bekleme durumuna giriyor ve pinin durumu değişene kadar bu durumda kalıyor.

INTR mikroişlemci bu giriş pini aracılığıyla donanım kesintileri algılıyor.

$\overline{TEST}$ bu pin WAIT yönergesi sırasında tespit ediliyor. Bu pin genelde 8087 sayısal yardımcı işlemci ile bağlıdır.

NMI bu pinin kısaltması Non Maskable Interrupt sözlerinden kaynaklanıyor ve maskelenmeyen kesinti anlamına geliyor.

RESET bu pin mikroişlemcinin sıfırlandırılması için kullanılıyor ve ondan sonra mikroişlemci FFF0H adresli konumlardan yönergeleri çalıştırmakla devam ediyor.

MN/$\overline{MX}$ bu pin alçak seviye durumundaysa, o zaman 8086 mikroişlemcisi maksimum düzeninde çalışıyor, yüksek seviyedeyse o zaman minimum düzeninde çalışıyor.

$\overline{BHE}$/S7 Bu pin Bus High Enable sözlerinin kısaltmasıyla işaretleniyor ve sıfır değeri varsa, o zaman veri 16 - bitli veri ya da özel bayt olması önemli olmadan, mikroişlemcinin D8'den D15' e kadar olan daha değerli veri bitlerine erişimi vardır

Minimum düzeninde mikroişlemci kontrol sinyallerini üretiyor ve onları bellekten dış aygıtlara aktarıyor.

M/$\overline{IO}$ Bu pin yüksek seviyedeyken, mikroişlemci bellekle iletişim kuruyor, alçak seviyede ise o zaman dış aygıtlarla iletişim kuruyor.

$\overline{WR}$ WR çıkış pinidir ve yazma işlemini başlatıyor.

$\overline{INTA}$ Bu sinyal kesinti aramasına onay olarak kullanılıyor ve INTR sinyaline cevap tanımlıyor.

ALE ALE giriş pini aktif olduğu zaman adres - veri veriyolundan adres bitleri aktarılıyor.

DT/$\overline{R}$ Bu pinin işareti Data Transmit /Receive sözlerinin kısaltmasıdır ve gönderilen/alınan veriler anlamına geliyor. Bu sinyal verilerin hareket yönünü tanımlıyor, mikroişlemciye doğru ya da ters yönde.

$\overline{DEN}$ DEN işareti Data Enable sözlerinin kısaltmasıdır ve anlamı verilerin etkinleşmesidir. Veriyolundan veri aktarımının gerçekleşmesi gerektiğinde, bu pin her zaman alçak seviyededir.

HOLD HOLD giriş pinidir ve onun aracılığıyla DMA denetimcisi mikroişlemciden veriyollarını devralmak için istek gönderiyor.

HLDA HLDA çıkış pinidir ve onun aracılığıyla mikroişlemci, verilerin doğrudan aktarma isteğine cevap veriyor.

$\overline{SSQ}$ SSQ durum sinyalı IO/M ve DR/R sinyalleriyle beraber, minimum çalışma düzeninde makine döngü türünün tanımlanması için kullanılıyor.

Maksimum çalışma düzeninde, aynı pinler mikroişlemcinin matematik yardımcı işlemciye bağlanması için kullanılıyorlar. Böyle durumda kontrol işlevlerini yeriyolların denetimcisi gerçekleştiriyor.

$\overline{S2}$, $\overline{S1}$ **ve** $\overline{S0}$ durum bitleri maksimum çalışma düzeninde makine döngüsünün türünü tanımlıyor. Bu sinyaller mikroişlemci için çıkış sinyalleridir, veriyolu denetimcisi için ise giriş sinyalleridir. Bu denetimciyi aşağıda daha yakından tanıyacağız.

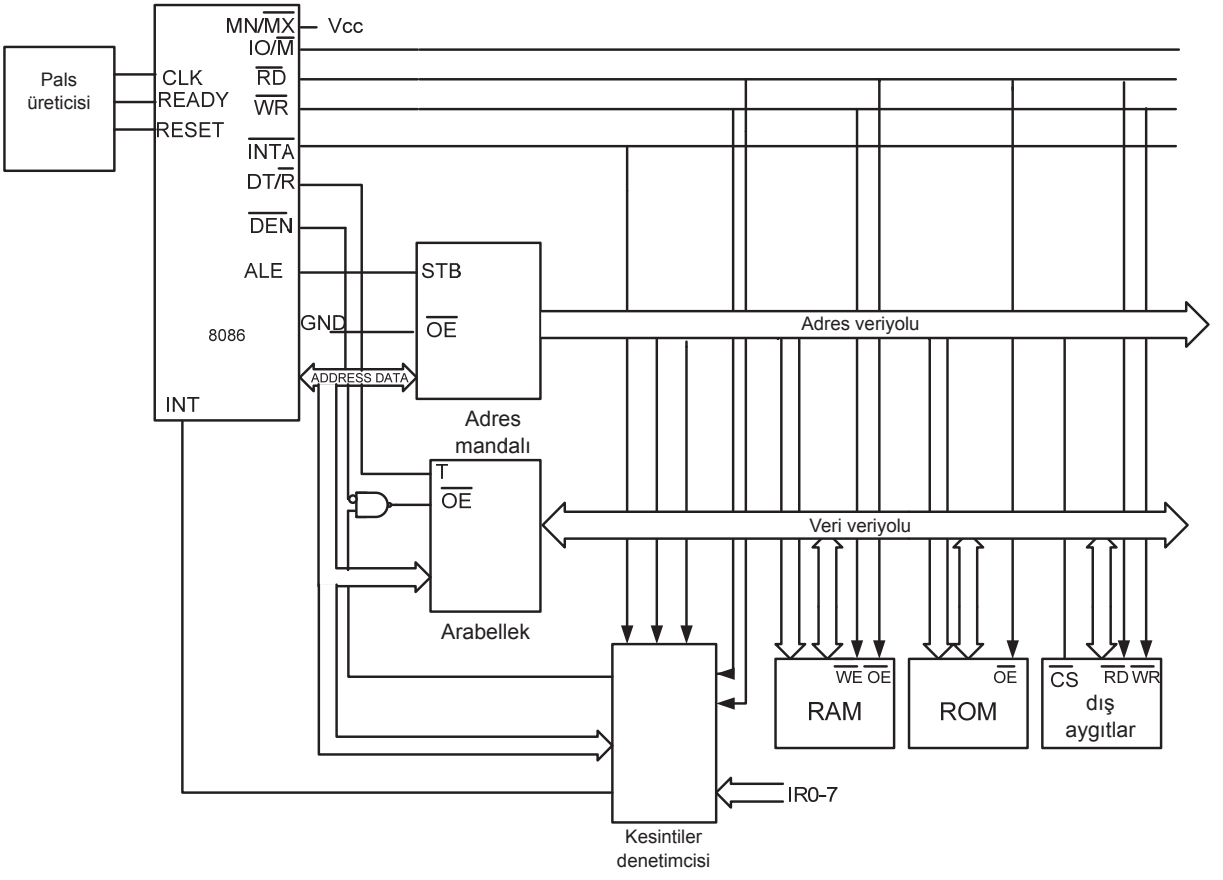
RO/GT1 ve RO/GT0 Bu pinlerin kısaltması Request/Grant sözlerinden geliyor ve istek/onay anlamına geliyor.Bu pinler ikiyönlüdür ve belleğe doğrudan erişimini (DMA) yönetmek için kullanılıyorlar

$\overline{LOCK}$ Bu pin sistemde dış aygıtlarını kilitlemek için kullanılıyor. Bu pin yönerge önünde LOCK önekini ekleyerek aktifleştiriliyor.

QS1 ve QS0 Bu pinler queue status sözlerinin kıslatmasıyla işaretleniyor ve yönerge sırasının durumunu tanımlamak için kullanılıyorlar. Yönergeler sırası devre dışı ya da boş olabilir, birinci baytı işlem kodu ya da ondan sonra sıraya gelen bayt olabilir.

8.3. Minimum ve Maksimum Çalışma Düzeni

8086 mikroişlemcide pinlerin işlevlerini açıklarken, mikroişlemcinin iki farklı düzende çalışabileceğini söylemiştik, minimum ve maksimum. **Çalışma düzeninin seçimi MN/MX pinin aracılığıyla yapılıyor**. MN/MX pini alçak seviyedeyse, o zaman mikroişlemci maksimum düzeninde çalışıyor, yüksek seviyedeyken ise mikroişlemci minimum düzeninde çalışıyor. Minimum çalışma düzeni 8085 8 - bitli mikroişlemcinin çalışma şekline çok benzerdir ve 8085 mikroişlemcinin kullandığı tüm programlar, 8086 mikroişlemcisi için de uygulanabilir. Resim 8.4.'te 8086 mikroişlemcinin **minimum mikrobilgisayar sistemi** gösterilmiştir. Bu şekil çok daha hesaplıdır, çünkü **mikroişlemci kontrol sinyallerini kendisi üretiyor**. IO/M, RD ve WR sinyalleri iletişim için bellek ya da dış aygıtı seçiyorlar ve işlem türünü seçiyorlar, okumak ya da yazmak. Adres - veri veriyolundan adres bitleri aktarıldığı zaman, ALE adres mandalın etkinleştirilmesini gerçekleştiriyor. DEN ve DT/R sinyalleri verilerin hareket etme yönünü belirliyorlar ve veri önbelleğini etkili hale getiriyorlar. Burada tabii ki INTA sinyalidedir ve onun aracılığıyla kesinti denetimcinin aradığı kesintiye izin veriliyor. 8086 mikroişlemcinin kullandığı sinyaller, 8085 mikroişlemcinin kullandığı sinyallerle neredeyse aynıdır.



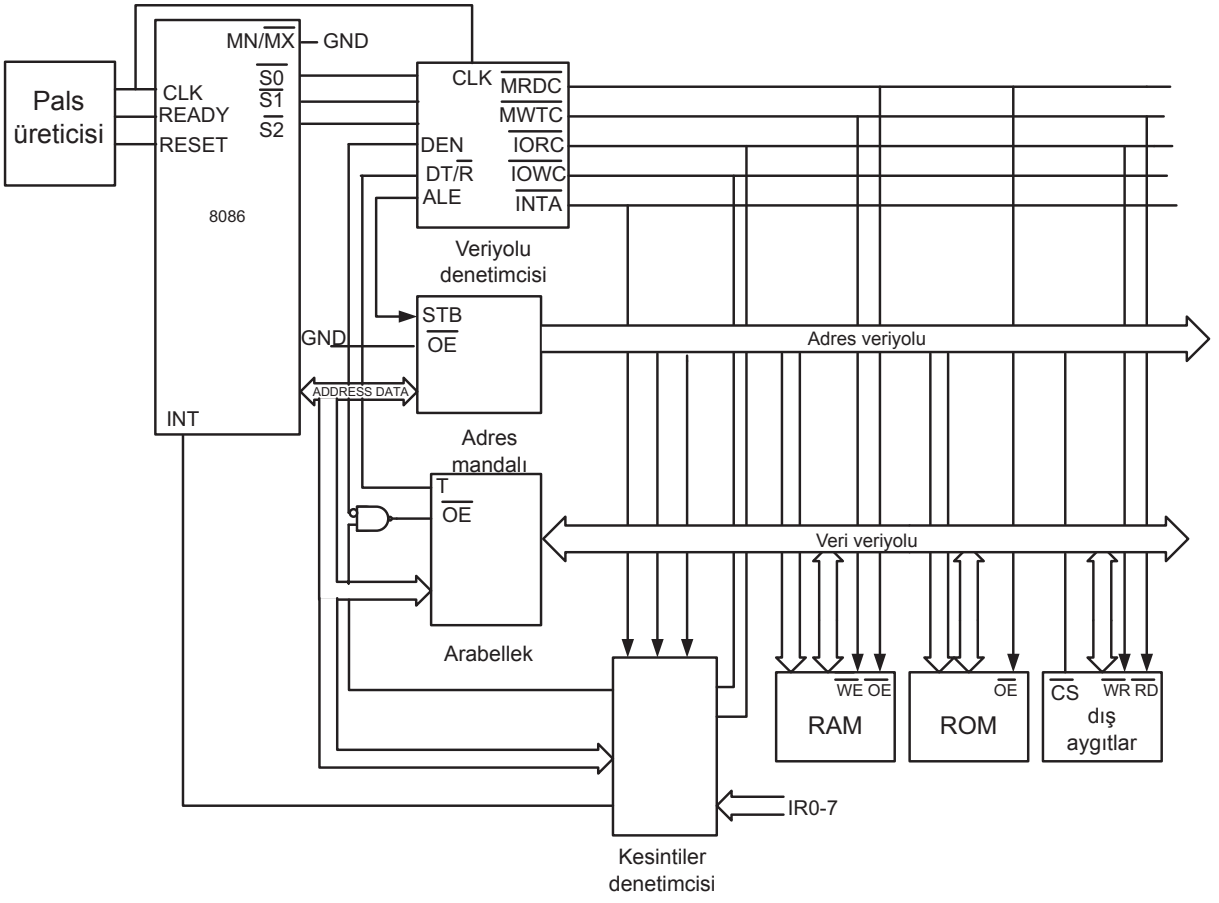
Resim 8.4. 8086 minimum mikrobilgisayar sistemi

Maksimum düzeni, sistemde bir ya da fazla aritmetik yardımcı işlemcinin dahil olduğu zaman kullanılıyor. 8086 mikroişlemcinin veriyollarıyla yönetmek için yeterli pini olmadığı için bu işlevi 8288 seri numaralı veriyolu denetimcisi gerçekleştiriyor. Resim 8.5.'te, mikroişlemcinin maksimum düzeninde çalıştığı zaman, 8086 mikroişlemci sistemi gösterilmiştir. Veriyolu denetimcisine bakarsak, 8086 mikroişlemcinin minimum düzeninde ürettiği aynı kontrol sinyalleri ürettiğini göreceğiz.

Minimum düzeninde S0 durum hattı hep alçak seviyedeydi, S1 kesinti bayrağının durumunu gösteriyordu ve S2 mevcut programda hangi bölütün aktif olduğunu gösteriyordu. Maksimum düzeninde aynı bu **durum hattları**, makine döngü türünün belirlenmesi için kullanılıyor: işlem kodun aktarımı, bellekten okuma ya da yazma, dış aygıttan okuma ya da yazma, kesintinin işletilmesi. $\overline{S0}$, $\overline{S1}$ ve $\overline{S2}$ bitlerin kodları çözülüyor ve onların kombinasyonuna bağlı olarak çıkış kontrol sinyallerinden bazıları aktifleştiriliyor. $\overline{RD}$, $\overline{WR}$ ve IO/$\overline{M}$ sinyalleri yerine benzer işlevi ancak farklı işaretlenen kontrol sinyalleri kullanılıyor:

- $\overline{IOWC}$ - Input Output Write Control, dış aygıta yazma işlemini başlatıran sinyaldir.

- $\overline{IORC}$ - Input Output Read Control dış aygıttan verinin okunması gerektiği zaman aktifleştiriliyor.
- $\overline{MWTC}$ - Memory Write Control bellekte yazmak için kullanılıyor
- $\overline{MRDC}$ - Memory Read Control bellekten okuma sırasında aktifleştiriliyor.



Resim 8.5. Maksimum 8086 mikrobilgisayar sistemi

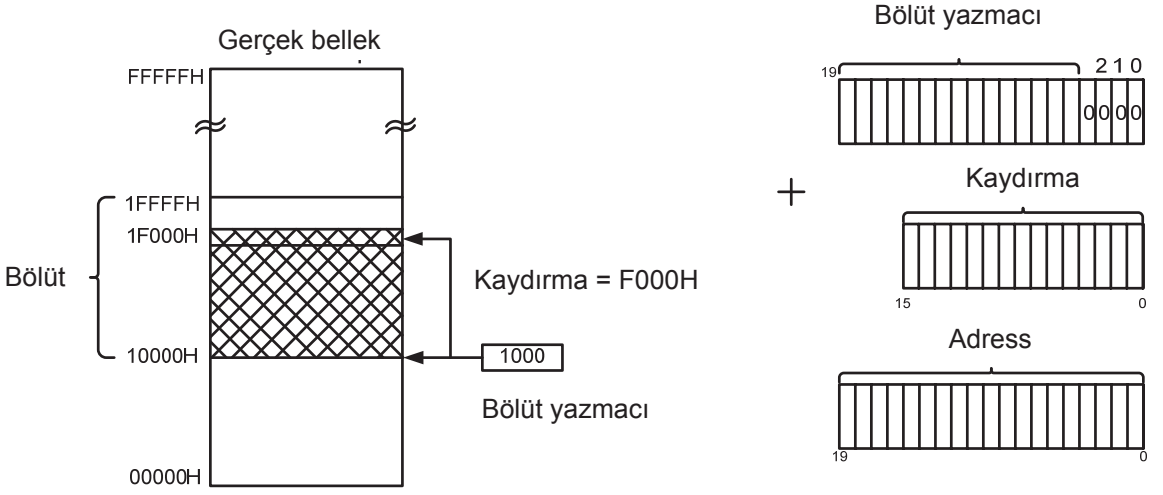
Maksimum düzenin kullanımı, 16 bitli 80286 mikroişlemcinin ortaya çıkmasıyla bitiyor.

8.4. Gerçek Çalışma Düzeni

80286 ve ondan sonra ortaya çıkan tüm mikroişlemciler iki çalışma düzeninde çalışabiliyorlar: korumalı ve gerçek düzen. Tek bir 8086 mikroişlemcisi sadece gerçek çalışma modunda çalışıyor. Gerçek çalışma düzeni mikroişlemciye 1MB bellek alanına erişim sağlıyor. Bellek alanının ilk 1MB'ı gerçek bellek adıyla bilinmektedir. DOS işletim sistemi, gerçek çalışma modunda çalışıyor. Gerçek çalışma modu, herşeyden önce eski yazılımın yeni mikroişlemcilerle uyum sağlanması için hala

kullanılmaktadır. Gerçek düzende yazılan programların korumalı düzende çalışmaları için onların işletilmesi gerekiyor. Bu da büyük çaba girişimi arıyor. 1MB belleğin adreslenmesi için 20 adres biti gerekiyor (2^{20} = 1MB).

Gerçek çalışma modunda fizik adresin oluşması için, iki bölüm gerekiyor: bölütün başlangıç adresi ve kaydırma. Bölütün başlangıç adresi, bölüt yazmacın 16 - bitli değerini 4 yer için sola kaydırarak ve sağ taraftan 4 sıfırın eklenmesiyle elde ediliyor. Aynı etki, bölüt yazmacının içeriğini $16_{16}=1000_{10}$ ile çarpmakla elde ediliyor. Bölüt yazmacın içeriği on altılı sayı sisteminde verilmişse, bölütün başlangıç adresi, bölüt yazmacın on altılı içeriği bir yer için sola kaydırarak, sağ taraftan bir sıfırın eklenmesiyle elde ediliyor. Örneğin, eğer bölüt yazmacı 1000H değerini içerirse, o zaman bu bölütün birinci baytın adresi 10000H olacak. Kaydırma başlangıç adresine eklenen 16 - bitli değerdir ve toplam 64KB'tan 1 baytın tanımlanması iiçin kullanılıyor. **64KB** değeri **bellek bölütünün en büyük değeridir** ve 2^{16} = 64 KB denklemi yardımıyla elde ediliyor. Denklemde 16 sayısı, kaydırmada bitlerin sayısıdır. Kaydırmanın en küçük değeri 0000H'dır ve bu değeri olduğu zaman bölütün ilk baytına yönlediriyor. Kaydırmanın en büyük değeri FFFFH'dir ve o zaman bölütün son baytına yönlendiriyor. Gerçek çalışma modunda bölütün uzunluğu 64KB uzunluğunda olduğundan dolayı, bölütün bitiş adresi, başlangıç adresine FFFFH değerin eklenmesiyle elde ediliyor. Resim 8.6.'da 8086 mikroişlemcinin RAM belleğine nasıl eriştiği gösterilmiştir.



Resim 8.6. 8086 mikroişlemcide fiziksel bellek adresin hesaplanması

Yazmacın başlangıç adresine eklenen kaydırma, bazı yazmaçta bulunabilir ya da 16 - bitli sayı şeklinde verilmiş olabilir. Kaydırmanın tanımlanması için sıkı (sert) kurallar vardır. Örneğin, kod bölütüyle çalıştığımız zaman, kaydırma her zaman yönerge göstergesinde içeriktir, yığıt bellekle çalıştığımız zaman ise kaydırma SP ve

BP yazmaçlarından birinde içerik olabilir. Tablo 8.1.'de **bölüt yazmacı - 16 - bitli kaydırmak için tüm geçerli kombinasyonlar verilmiştir**.

Bölüt	Kaydırma	Kullanım
CS	IP	Sırakadi yönergenin bulunması
SS	SP ya da BP	Yığıtta konuma erişim
DS	BX,DI,SI ya da 16 bitli sayı	İşlenenin bulunması
ES	DI dizilerle çalışmak için	Hedef adresinin tanımlanması

Tablo 8.1.

Gerçek çalışma modunda her bölütün en çok 64 KB belleği olabilir. Bazı bölütler eşleşebilir. Bazı bölüt ona ait tüm 64KB'ı kullanmazsa, o zaman başka bir bölüt onunla eşleşebilir, yani ikinci bölüt, birinci bölütün kullanmadığı baytları kullanabilir. Bu şekilde programcı, programın gerçekleşemesi için gereken fiziksel belleği azaltabilir.

8.5. 8086 Mikroişlemcide Adresleme Şekilleri

Adresleme türlerini MOV yönergenin yardımıyla açıklayacağız. MOV yönergesiyle veriler bir yerden başka bir yere taşınıyor. Taşınan veriler bayt olabilir (8 bit), sözler olabilir (16 bit), çift sözler olabilir (32 bit) vs. Aktarma bir yazmaçtan başka bir yazmaca ya da yazmaçtan belleğe ve ters yönde olabilir. Verilerin alındığı yere **işlenenler kaynağı** denir, İstenen verilerin gittiği yere **hedef** denir. Resim 8.7.'de MOV yönergesi verilmiştir ve farklı adresleme şekillerinde verilerin hareket etme yönü verilmiştir. İşlem kodunun hemen yanında hedef verilmiştir, devamda ise kaynak bulunuyor. Kaynak ve hedef birbirinden virgül işaretiyle ayrılıyor.

8086'dan başlayarak, 80286'ya kadar şu adresleme şekilleri kullanılıyor: yazmaçlı adresleme, doğrudan, dolaysız, yazmaçlı dolaylı, temel dolaylı, yazmaç göreceli ve temel görece endeksli. 80386 ve ondan sonra gelen mikroişlemcilerde bir adresleme türü daha var, o da sayısal (skalar) endeksli adreslemesidir.

Yazmaçlı Adresleme

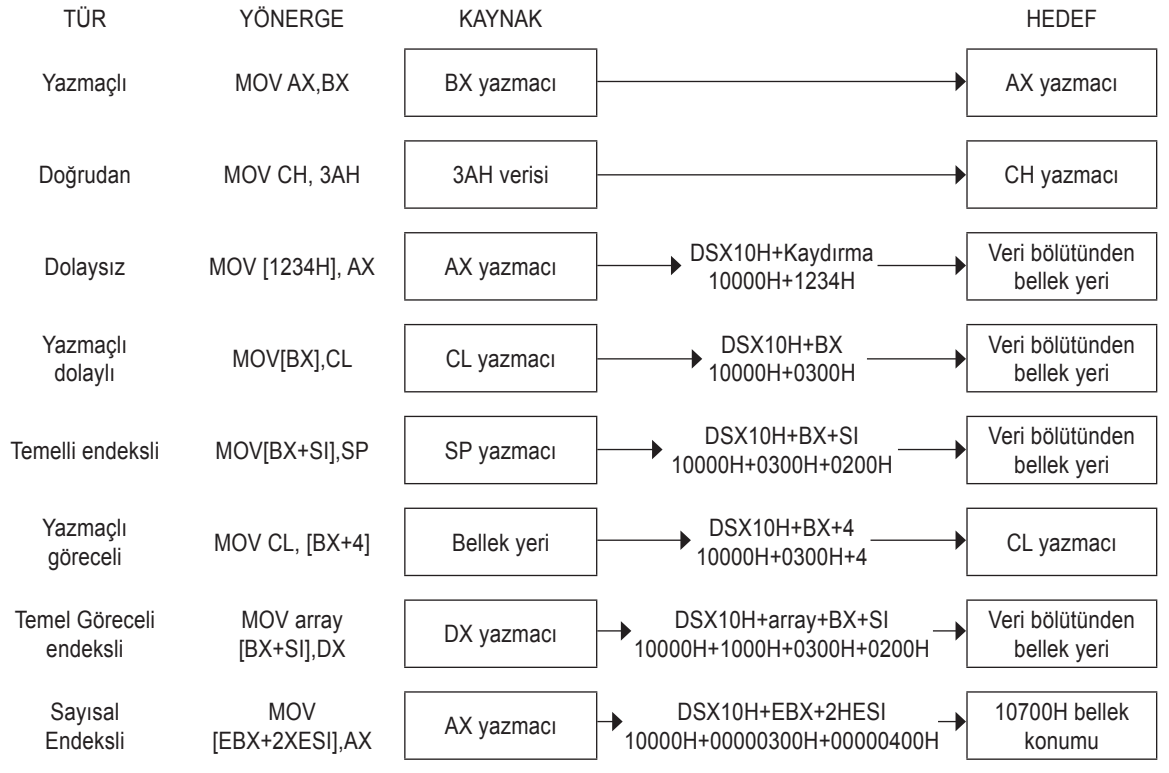
Veri, mikroişlemciye terketmeden bir yazmaçtan başka yazmaca aktarılıyor. Bir bölüt yazmacından başka bölüt yazmacına, faklı büyüklükte yazmaçlar arasında aktarım gerçekleşemez ve kod bölütü hedef olamaz.

Doğrudan adresleme

Aktarılması gereken veri kaynağın yerinde, hedefe doğrudan yazılıyor. Veri ikili sayı sisteminde verilmişse, o zaman sayıdan sonra B işareti duruyor, veri on altılı sayı sisteminde verilmişse, o zaman sayıdan sonra H harfi yazılıyor. Veri onlu sayı sisteminde ifade edilmişse, o zaman sayıdan sonra D harfi yazılabilir, ancak onlu sayı sistemide harfin yazılması şart değil.

Dolaysız Adresleme

Dolaysız adresleme şeklinde, yönerge bellek yerin adresini içeriyor. Bellek konumunun fiziksel adresi, bölüt yazmacının değerini 10H on altılı sayıyla çarparak ve bu çarpıma kaydırmayı ekleyerek elde ediliyor. Çevirici yönergede kaydırma değer olarak orta parantezlerde yazılıyor [ ].



Resim 8.7. 8086 mikroişlemcide adresleme şekilleri

Örneğin, MOV AL, DS:[1234H] yönergesinin anlamı şudur: bölüt yazmacında bir baytın 1234 H kaydırmayla AL yazmacına kopyalamak. Bu yönerge şöyle de ya-

zılabilir: MOV AL, [1234H]. Eğer başka bir şekilde vurgulanmazsa, yönergede her zaman bölüt yazmacı kullanılıyor. Programcı veri bölütünü tanımladığı zaman, bildirimler (direktifler) olarak adlandırılan özel sözde - yönergeler yardımıyla bellek yerlerine sembolik isimler verebilir. Programcı için bellek yerlerin on altılı adresleri ezberlemesinden, sembolik isimleri ezberlemek çok daha kolaydır. Örneğin, MOV AL,NUMBER yönergesi, NUMBER sembolik ismi olan veri bölütündeki bellek yerinden baytı, AL yazmacına kopyalamak demektir.

Yazmaçlı Dolaylı Adresleme

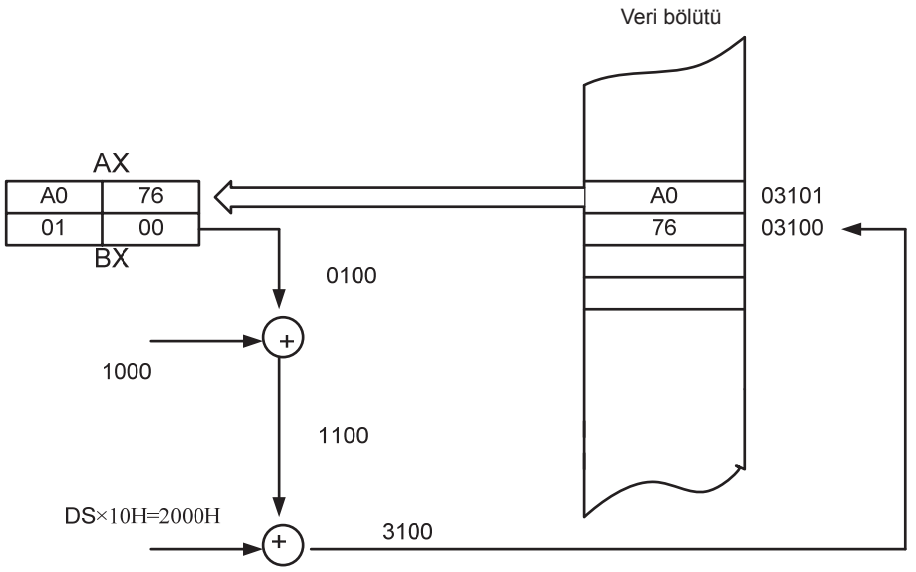
Yazmaçlı dolaylı adresleme şekliyle herhangi bir bellek yerine erişim edilebilir ve bu arada kaydırma şu yazmaçlardan birinde içeriktir: BP, BX, DI ya da SI. Hangi bölütün söz konusu olduğu, resim 8.5'teki tablosunu kullanarak, endeks - yazmaçtan görebiliriz. Örneğin, MOV CX, [BX] yönergesiyle, BX yazmacında, veri bölütün kaydırmayla toplamı CX yazmacına kopyalanıyor. Bu arada, DSH10H + BX adresli birinci bayt CL yazmacında yerleştiriliyor, DSH10H + BX + 1 adresli ikinci bayt ise CH yazmacında yerleştiriliyor. Bir bellek yerinden başka bellek yerine veri kopyalamak olanağına izin verilmiyor. Örneğin MOV [DI], [BX] yönergesi geçerli değildir.

Temel Endeksli Adresleme

Bu adresleme şeklinde, kaydırma bir temel yazmacın (BX ya da BP) ve bir endeksli yazmacın (DI ya da SI) toplamı olarak elde ediliyor. Temel Endeksli adresleme şekli dizilerle çalışma sırasında kullanılabilir. Temel yazmacı bölütün başlangıcından, dizi başlangıcına kadar kaydırmak için kullanılıyor, endeks - yazmacı ise dizi başlangıcından istenilen yere kadar kaydırmayı içeriyor. Örneğin, MOV [BP + DI], AH yönergesi, AH yazmacındaki sözü, yığıt bölütünden bellek yerine kopyalamak için kullanılıyor ve bu arada, kaydırma BP·10H ve DI içeriklerinin toplanmasıyla elde ediliyor. Önceki adresleme şeklinde olduğu gibi, temelli endeksli adresleme şeklinde de BP yazmacını kullanırsak, o zaman verinin yığıt bölütünde olduğu demektir, BX yazmacı kullanılırsa, o zaman verinin veri bölütünde bulunduğu demektir.

Yazmaçlı Göreceli Adresleme

Bu adresleme şeklinin, temel endeksli ve doğrudan adresleme şekilleriyle benzerlikleri var. Yazmaçlı - göreceli adreslemede, veri - bölütünün herhangi bir yerine erişmek için orta parantezdeki sayının, bazı temel ya da endeks - yazmacın içeriğine eklenmesi gerekiyor (BP, BX, DI ya da SI). Resim 8.8.'de yazmaçlı göreceli adresleme şekli için örnek verilmiştir. MOV AX, [BX+1000H] yazmacıyla, BX·10H temel yazmacıyla adreslenen ve ona 1000H sayısı eklenerek, veri bölütünden bir söz kopyalanıyor. Bu adresleme şeklinde de [ ] parantezlerin kullanıldığını not edelim. Bu parantezler yoksa, BX yazmacının içeriği ve 10000H değerinin toplamı veri bölütünden bellek adresi değil, verinin tanımlandığı hatalı sonucuna varılabilir. DS yazmacının değeri 0200H olduğunu vurgulayalım.



Resim 8.8. Yazmaçlı - Göreceli adresleme şeklinde adresin hesaplanması

Temel Göreceli Endeksli Adresleme

Temel göreceli endeksli adresleme, göreceli endeksli adreslemeye benzerdir. Sadece temel göreceli endeksli adreslemede bölütün başlangıcı ve istenilen konum arasındaki mesafenin elde edilmesi için endeks yazmacın içeriği 16 - bitli sayıyla toplanması gerekiyor. Böyle adresleme şekilli yönerge için örnek, MOV AX, [BX+SI+1000H] yönergesidir.

8.6. 8086 Mikroişlemcide Yönergeler Kümesi

8086 işlemcisi yüzlerce yöneregeye sahiptir ve bu yönergeler birkaç gruba ayrılabilir:

- Veri aktarma yönergeleri
- Aritmetik yönergeler
- Kaydırma ve döndürme yönergeleri
- Mantıksal yönergeler
- Kontrollu atlama yönergeleri
- Dizilerle çalışma yönergeleri
- Alt programlarla ve kesintilerle çalışma yönergeleri
- Mikroişlemcinin kontrol için yönergeleri

8.6.1. Veri Aktarma Yönergeleri

MOV yönergenin anlamını, 8086 mikroişlemcinin adresleme şekillerini incelerken tanıdık. Hatırlama amacıyla, birkaç özel örnek vereceğiz:

Örnek 8.1.

MOV BX, AX - verinin bir yazmaçtan başka yazmaca doğrudan aktarımı
MOV BX, 1234H - 1234H verinin BX yazmacına aktarım

MOV BX, [1234H] - başlangıçtan 1234H mesafelik uzaklıkta olan veri bölütünden bellek yerindeki BX yazmacına veri aktarımı

MOV [1234H], BX - BX yazmacından belleğe veri aktarımı

MOV yönergenin kullanımında belli sınırlamalar var. Bu yönerge ile bir bellek yerinden başka bellek yerine aktarma yapılamaz, ancak genel yazmaçların aracılığıyla yapılır. Bölüt yazmacına veri aktarılamaz ne de bir bölüt yazmacından başka bölüt yazmacına aktarma olanağı yoktur.

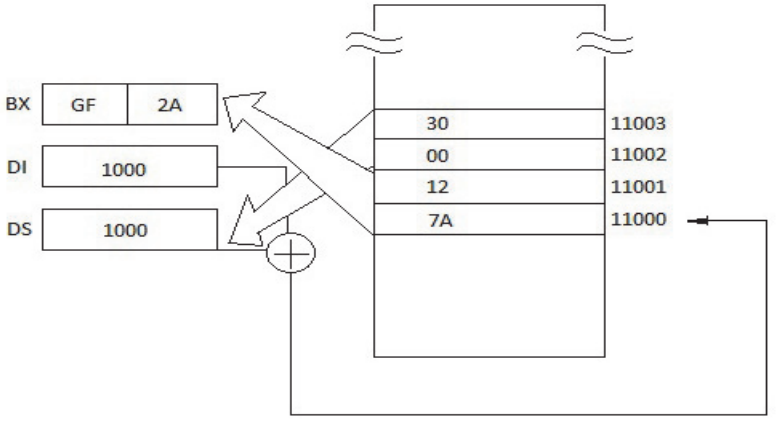
8086 mikroişlemcinin, **PUSH** ve **POP** yönergelerin 6 farklı türü vardır. Yığıt belleğinde aktarılan veri farklı yerlerde yerleşebilir.

Örnek 8.2.

PUSH BX - veri bazı genel yazmaçtadır
PUSH [BX] - veri, veri bölütün başlangıcından BX değerine eşit mesafede uzak olan bellek yerinde bulunuyor
PUSH 1225H - veri doğrudan verilmiştir
PUSH DS - veri bazı bölüt yazmacında bulunuyor
PUSH A - tüm 16 - bitli genel yazmaçlarda bulunan veriler korunuyor
PUSH F - durum yazmacı bayraklarıyla beraber korunuyor

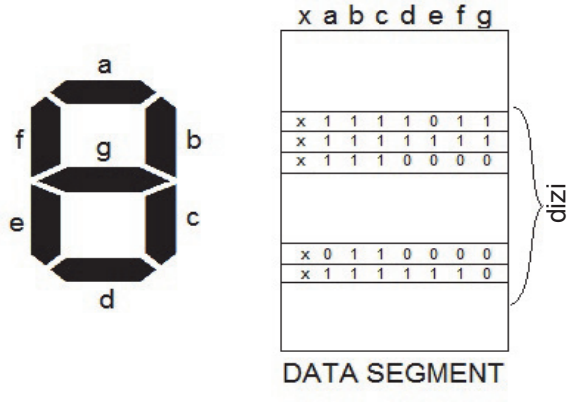
8086 mikroişlemci, efektif adreslerin aktarımı için özel yönergelere sahiptir. Efektif adresler verinin bölütün başlangıcında nekadar uzak olduğunu gösteren kaydırmayı (mesafeyı) tanımlıyor**. LEA** (Load Effective Address) yönergesiyle, verinin efektif adresi 16 - bitli yazmaca aktarılıyor. Örneğin, LEA BX,[DI] yönergesiyle DI yazmacın içeriği BX yazmacına aktarılıyor. MOV BX, [DI] yönergeyle ise DI yazmacından efektif adresli veri BX yazmacına aktarılıyor.

LDS ve **LES** yönergeleri, belleğe 32 bitin aktarılması için kullanılıyor. Bu arada, ilk iki daha değerli bayt DS ya da ES bölüt yazmaçlarına aktarılıyor, son iki daha az değerli bayt bazı 16 - bitli genel yazmaçta aktarılıyor. LDS BX, [DI] yönergesiyle BX yazmacı 11000 ve 11001 adresi yerlerden dolduruluyor, DI yazmacı ise 11002 ve 11003 adresli yerlerinden dolduruluyor. Yönergenin çalıştırılmasından sonra DS 3000H sayısını, BX ise 127AH sayısını içerecek. Bu işlem resim 8.9.'da gösterilmiştir.



Resim 8.9. LDS BX, [DI] yönergesinin görüntüsü

XLAT (Translate) Bu yönerge bir kodtan başka koda dönüştürme sırasında kullanılıyor, örneğin BCD'den yedibölütlü koda dönüştürmesi gibi. Veri bölütünde, 0'dan 9'a kadar rakamların yedibölütlü kodlarını içeren 10 bayttan oluşan dizi hazırlanmıştır. Resim 8.10'da yedi bölütlü görünüm birimi ve onun kodları verilmiştir. XLAT yönergesiyle, veri bölütünde konumun bellek adresinin elde edilmesi için AL yazmacın içeriği çarpıma ekleniyor, ondan sonra ise o bellek yerin içeriği AL yazmacında kopyalanıyor.



Resim 8.10. Yedi bölütlü görünüm biriminin aktifleştirilmesi için veri dizisi

XCHG (Exchange) Bu yönerge bir yazmaçın diğer yazmaçla ya da bir bellek yeriyle, içeriklerinin takası için kullanılıyor. Bu yönerge iki bellek yerin ya da bölüt yazmacın içeriklerinin takası için kullanılamaz.

İN ve **OUT** yönergeleri, giriş çıkış bağlantı noktalarla çalışmak için kullanılıyor. İN yönergesiyle veri giriş bağlantı noktasından biriktiriciye aktarılıyor, OUT yönergesiyle ise biriktiriciden veri çıkış bağlantı nokasına aktarılıyor. Bağlantı noktalarıyla çalışmak için iki yönerge türü vardır. **Birinci yönerge türünde** bağlantı noktaların sabit adresleri var. Arabirim - adresi 8 - bitli kombinasyon tanımlıyor ve sol taraftan sekiz sıfır ekleyerek 16 bite kadar genişlenebilir. Örneğin, OUT 19H, AX yönergesiyle, AX yazmacının içeriği, 0019H sabit adresli çıkış bağlantı noktasına aktarılıyor. Bağlantı noktalarla çalışmak için **ikinci tür yönergelerde**, bağlantı noktaların değişken adresleri vardır. Bağlantı noktanın adresi DX yazmacında saklanıyor, aktarılması ya da bağlantı noktasından alınması gereken veri ise AL veya AX yazmacında saklanıyor. Programın çalıştırılması sırasında DX yazmacının içeriği, yani bağlantı noktasının adresi değişebilir.

Örnek 8.3:

```
OUT DX, AL      ;AL yazmacında veri DX yazmacındaki,
                ;adresli çıkış bağlantı noktasına aktarılıyor.

IN AL, DX       ;DX yazmacındaki adresli giriş bağlantı noktasından veri
                ;mikroişlemcide AL yazmacına aktarılıyor
```

8.6.2 Dizilerle (Dizgilerle) İşlemler

Dizi, veri ya da ekstra bölüttünden ardaşıl bellek yerlerinde yerleşmiş veriler kümesidir. Bir bölütte işletilmesi gereken veriler bulunuyor, diğer bölütte ise gerçekleşen işlemler sonucu olarak elde edilen veriler yer alıyor. DI ve SI endeks - yazmaçları, kaydırmayı içeren yazmaçlardır. DI yazmacı ES bölüt yazmacının eşliğinde kullanılıyor, SI ise DS'in eşliğinde kullanılıyor. Dizilere işlemlerin gerçekleşmesi için **D bayrağı** çok önemlidir. Eğer D=0 ise, DI ve SI yazmaçların değerleri otomatik olarak bir için azalıyor, eğer D=1 ise o zaman DI ve SI yazmaçların değerleri bir için artıyor. **CLD** (Clear D) yönergesi bu bayrağı sıfırlandırıyor, **STD** (Set D) yönergesi ise D bayrağını ayarlıyor.

LODS ve STOS

LODS yönergesiyle AX ya da AL yazmacı Si endeks yazmacın kaydırılmasıyla veri bölütünden veriyle dolduruluyor. Aslında, bu yönergenin iki türevi vardır. **LODSB** (Load Byte) yönergesi 8 - bitli verilerin aktarımı için kullanılıyor, **LODSW**

(Load Word) yönergesi ise 16 - bitli verilerin aktarılması için kullanılıyor. STOS yönergesinin LODS yönergesinden ters etkisi var, yani AX ve AL yazmacından veri, DI yazmacında kaydırmayla beraber, ekstra bölütünden bellek yerine aktarılıyor. **STOS** yönergesine REP öneki de eklenebilir. Rep öneki İngilizce tekrarla anlamına gelen Repeat sözünden geliyor. REP önekiyle STOS yönergesi CX yazmacınada değerin olduğu kadar tekrarlanıyor. **MOVS** yönergesi çok kullanışlıdır, çünkü bir bellek yerinden başka bellek yerine veri aktarımı sağlıyor. Böyle olanağın, verilerin sadece sıralanmış dizide olduğu zaman geçerli olduğunu öne sürelim. MOVS yönergesiyle, SI yazmacına göstergeli veri - bölütünden veri, DI göstergeli ekstra bölütüne aktarılıyor.

SCAS (String scan instruction) yönergesiyle AL yazmacın içeriği ile ekstra bölütten baytlar bloğu (SCASB) arasında ya da AX yazmacın içeriği ile ekstra bölütten sözler bloğu (SCASW) arasında kıyaslama yapılması gerçekleşiyor. Bu iki yönerge önünde REPNE (repeat while not equal) ön eki eklenebilir ve böyle durumda SCAS yönergesinin başarılı kıyaslamanın elde edilmesine kadar (eşit içerik) ya da CX'in değeri sıfır olana kadar tekrarlanması demektir. Buna benzer diğer bir ön ek REPE (repeat while equal) ön ekidir.

CMPS (compare string) yönergesiyle, iki veri bloğu kıyaslanıyor. Bunlardan bir blok, SI'de kaydırması olan veri bölütünde bulunuyor, diğer blok ise DI'de kaydırması olan ekstra bölütünde yer alıyor. Bu yönergenin iki türevi vardır: CMPSB – bayt blokların kıyaslanması için ve CMPSW – sözler blokların kıyaslanması için. Bu yönergelerin önünde de REPNE ve REPE önekleri eklenebilir.

8.6.3. Aritmetik Yönergeler

8086 işlemcisi şu aritmetik yönergelere sahiptir: toplama, çıkarma, çarpma, kıyaslama, olumsuzluk, bir için azaltmak ya da arttırmak.

Toplama ve çıkarma

Birkaç toplama türü var. Şöyle ki, yönergelerde uygulanan tüm adres modları **ADD** yönergesinde de uygulanabilir. Birkaç örnek vereceğiz:

Örnek 8.4.

```
ADD AL, BL          ;Yazmaçlı toplama, AL=AL+BL.

ADD CL, 44H         ;Doğrudan toplama CL=CL+44H.
```

```
ADD [BX], AL            ;AL yazmacın içeriği, BX'te kaydırması olan
                        ;veri - bölütünden bellek yerine ekleniyor.

ADD DL, [ BX + DI]      ;BX +DI kaydırmalı veri - bölütünden
                        ;bellek yerin içeriği DL içeriğine ekleniyor
                        ;ve toplam yine DL yazmacına yazılıyor

ADD AL, ARRAY [SI]      ;AL yazmacın içeriğine, dizi
                        ;başlangıcından SI mesafelik
                        ;uzaklıkta olan ARRAY dizisinden
                        ;bir baytın içeriğine ekleniyor.

ADC AL, AH              ;Carry aracılığıyla toplamak, AL=AL+AH+Carry.
```

Aritmetik toplama durum yazmaçların durumuna daha doğrusu Z, C, A, S, P ve D bayraklarına etkiliyor.

Aritmetik çıkarma **SUB (Substract)** yönergesi yardımıyla gerçekleşiyor. ADD yönergesi hakkında tüm söylenenler SUB yönergesi için de geçerlidir. Virgül işaretinden sonra yazmacın ya da bellek yerin içeriği, virgül işaretinden sonra yazmacın ya da bellek yerin içeriğinden çıkarılıyor ve onda sonuç, fark yazılıyor. Ödünçlü çıkarma SSB (Subtraction with borrow), elde edilen farktan Carry bayrağının değeri çıkarılarak gerçekleşiyor.

CMP (Compare) Kıyaslama

Bu yönerge bayrakların durumuna etkiliyor ve bu arada işlenenlerin değeri aynı kalıyor. Genelde, CMP yönergesinden sonra JA (jump above) ve JB (jump below) yönergeleri gibi koşullu atlama yönergesi bulunuyor.

Örnek 8.5:

```
CMP AL, 10H             ;AL yazmacın içeriğinden
                        ;10H değeri çıkarılılyor. AL'ın
                        ;içeriği değişmiyor, sadece
                        ;C bayrağıın değeri değişebilir.
                        ;Eğer AL>10H ise, o zaman C=1,
                        ;Eğer AL<10H ise, o zaman C=0.

JA  START               ;Aktarma bayrağı aktif değilse
                        ;atlama gerçekleşiyor, yani eğer
                        ;AL>10H.
```

8085 mikroişlemciden farklı olarak 8086 mikroişlemcinin yönergeler kümesi aritmetik **çarpma ve bölme** için yönergeler içeriyor. 8 - bitli ve 16 - bitli çarpma ayırıyoruz. 8 - bitli çarpmada bir çarpan her zaman AL yazmacında bulunuyor, diğer çarpan ise bazı genel yazmaçta veya bellek yerinde bulunuyor. Çarpım 16 - bitlidir ve AX yazmacında yerleşiyor.

16 - bitli çarpmada bir çarpan AX yazmacında yer alıyor, diğer çarpan ise bazı 16 - bitli yazmaçta ya da iki ardaşıl bellek yerinde bulunuyor. Çarpım 32 - bitli sayıdır ve bu arada ilk 16 daha az değerli bitler AX yazmacında yazılıyor, ikinci 16 daha değerli bit ise DX yazmacında yazılıyor. Çarpmak için çevirici yönergesi olan **MUL** yönergesi multiply sözcüğünün kısaltmasıdır.

Örnek 8.6:

```
MUL CL              ;AL ve CL yazmaçların içerikleri
                    ;çarpılıyor ve sonuç AX yazmacında
                    ;yazılıyor.

MUL CX              ;AX ve CX yazmaçların içerikleri
                    ;çarpılıyor ve sonuç AX ve DX
                    ;çiftinde yazılıyor.

MUL BYTE PTR [BX]   ;AL yazmacın içeriği ve veri bölütünden
                    ;veri bölütün başlangıcından BX
                    ;mesafede bulunan baytın
                    ;içeriği çarpılıyor.
```

Çarpma işlemi olduğu gibi, **bölme** işlemi de 8 - bitli ve 16 - bitli olabilir. 8 - bitli bölmede bölünen 16 bitli sayıdır ve AX yazmacına bulunuyor, bölen ise yönergenin içeriğinde verilmiş 8 bitli genel yazmaçta ya da bellek yerde bulunuyor. Bölme işleminin gerçekleştirilmesinden sonra bölüm AL yazmacında, kalan ise AH yazmacında yerleşiyor. Çeviricide bölme yönergesi **DIV** yönergesidir.

Örnek 8.7:

```
DIV CL              ;AX:CL = AL (AH kalan)
```

16 - bitli bölmede bölünen AX ve DX yazmaçlar çiftinde yerleşmiş olan 32 bitli sayıdır, bölen ise yönergenin içinde verilmiş bazı yazmaçta ya da bellek yerinde bulunuyor. Bölüm AX yazmacında, kalan ise DX yazmacında yerleşiyor.

8.6.4. Mantıksal Yönergeler

8086 mikroişlemcinin yönergeler kümesi şu mantıksal yönergeleri içeriyor: AND, OR, dışlamalı OR, NOT, TEST ve NEG.

AND yönergesi mantıksal çarpma gerçekleşiyor ve tüm adres modları için geçerlidir.

Örnek 8.8:

```
AND AL, BL          ;AL=AL AND BL
AND DI, 4FFFH       ;DI=DI AND 4FFFH
AND AX, [DI]        ;AX yazmacın içeriği veri yazmacının
                    ;başlangıcından DI'de mesafe uzaklığında
                    ;olan veriyle mantıksal olarak çarpılıyor.
AND [BX+SI], AL     ;AL yazmacın içeriği ve veri bölütünün
                    ;başlangıcında BX+SI mesafelik uzaklıkta
                    ;olan yerin içeriği arasında mantıksal çarpma.
                    ;Sonuç aynı bellek yerinde yazılıyor.
```

AND işlemi, bir sayının bazı bitlerini sıfırla çarparak bu **bitlerin maskelenmesi** için kullanılıyor. Maskelenen bitler sıfır oluyor. Maskelenmeyen bitler (birle çarpılan bitler) kendi değerlerini koruyor.

Örnek 8.9:

```
      x x x x x x x x - Bilinmeyen sayı
AND 0 0 0 0 0 0 0 1  Maske
    0 0 0 0 0 0 0 x  Sonuç
```

OR yönergesi mantıksal toplama işlemini gerçekleştiriyor.

Örnek 8.10:

```
OR AX,BX            AX=AX OR BX
OR AX, [BP]         ;AX yazmacın içeriği ve BP'de
                    ;mesafe için yığıt bölütünün başlangıcından
                    ;uzaklıkta olan söz mantıksal olarak toplanıyor.
                    ;Sonuç AX yazmacında yazılıyor.
OR CL, [2555H]      ;CL yazmacının içeriği ve veri
                    ;bölütün başlangıcından 2555H
                    ;mesafelik uzaklıkta olan veri
                    ;mantıksal olarak toplanıyor.
                    ;Sonuç CL yazmacında yazılıyor.
```

OR işlemiyle de maskelemek yapılabiliyor. Bu arada maskelenen bitler bir ile toplanıyor ve bu şekilde ayarlanıyorlar.

Örnek 8.11:

```
  x x x x x x x x  Bilinmeyen sayı
+ 1 1 1 1 1 1 1 0  Maske
  1 1 1 1 1 1 1 x  Sonuç
```

OR ve dışlamalı OR ya da **XOR** işlemleri arasında tek fark, ikinci işlemdeki iki birin toplamı sıfıra eşit olmasıdır. Dışlamalı OR işleminin gerçekleşmesi için mantıksal devreye karşılaştırıcı denir, çünkü iki giriş biti aynıysa çıkışta sıfır veriyor, giriş bitleri farklı ise çıkışta bir veriyor.

Bu işlem, bilinmeyen sayının bazı bitlerini evirmek gerektiği zaman maskelemek için kullanılıyor.

Örnek 8.12:

```
  XXXXXXXX  Bilinmeyen sayı
+ 0 0 0 0 1 1 1 1  Maske
  XXXXXXXX  Sonuç
```

Örnek 8.13:

```
XOR CH, CL      ;CH yazmacın içeriği CL yazmacının
                ;içeriğiyle dışlamalı olarak toplanıyor
                ;ve sonuç CH yazmacında yazılıyor.
XOR AH, EEH     ;Yazmaç AH=yazmaç AH+EEH.
XOR DX, [SI]    ;DX yazmacın içeriği veri bölütün
                ;başlangıcından,SI yazmacın değerine
                ;eşit olan mesafelik uzaklıkta olan veriyle
                ;dışlamalı olarak toplanıyor. Sonuç
                ;DX yazmacında yazılıyor.
```

TEST yönergesi AND yönergesine benzerdir, sadece TEST yönergesinde işlenenlerin değeri değişmiyor, bayrakların durumu değişiyor.

CMP yönergesi iki yazmacın içeriklerini kıyaslamak için kullanılıyor, TEST yönergesi ise bazı yazmacın bazı bitlerini denetlemek için kullanılıyor. Örneğin, TEST AL, 01H yönergesiyle en düşük ağırlıklı bit (sağ taraftan birinci bit) denetleniyor. Bu bit bir ise, o zaman Z bayrağı sıfır olacak, çünkü AND işlemiyle sonuç 01H olacak. Denetlenen bit sıfır ise, o zaman Z=1 olacak (sonuç=00H). Örneğin, TEST AL,80H yönergesiyle AL yazmacından en yüksek ağırlıklı biti (soldan birinci bit) test ediliyor.

NOT yönergesi bir yazmacın veya bellek yerin içeriğinin birinci tümleyiciyi hesaplamak için kullanılıyor, **NEG** yönergesi ise ikinci tümleyiciyi, yani işaretin değişmesini hesaplamak için kullanılıyor. Bu iki yönerge sadece birer işlenen içeriyor.

Örnek 8.14:

```
NOT CH              ;CH yazmacının birinci tümleyicisi
NEG AX              ;AX yazmacın ikinci tümleyicisi
NOT BYTE PTR [BGX]  ;Veri bölütünün başlangıcından
                    ;BX değerine eşit uzaklıkta olan
                    ;bayt tümleniyor.
```

8.6.5. Kaydırmak ve Döndürme

Kaydırma yönergeleri **Shift** yönergeler olarak biliniyor (Shift – kaydırmak, taşımak). Shift yönergelerle bazı yazmacın ya da bellek yerin bitleri birkaç yer için sola veya sağa kayıyor, sağ ve sol tarafta oluşan boşluklarda ise sıfırlar giriyor. **Dört tür kaydırma yönergesi** ayırıyoruz: mantıksal sol Shift yönergesi (SHL), mantıksal sağ Shift yönergesi (SHR), aritmetik sol Shift yönergesi (SAL), aritmetik sağ Shift yönergesi (SAR). Son yönergede (SAR'da) işaret bitin değerinin korunması önemlidir.

Örnek 8.15:

```
SHL AX, 1       ;AX yazmacındaki bitlerin bir yer
                ;sola mantıksal kaydırılması.
SHR BH, 12      ;BX yazmacındaki bitlerin 12 yer için
                ;sağa mantıksal aydırılıması.
SAR SI, 2       ;SI yazmacındaki bitlerin iki yer için
                ;sağa aritmetik kaydırılması.
```

Döndürme yönergeleri, bir yazmaçtan ya da bellek yerin bitlerini CARRY bayrağı aracılığıyla bir taraftan başka tarafa döndürüyorlar. **Dört tür döndürme yönergesi** vardır: sola döndürme (ROL), sağa döndürme (ROR), Carry aracılığıyla sola döndürme ve Carry aracılığıyla sağa döndürme.

Örnek 8.16:

```
ROL SI, 14      ;Yazmaçtaki bitler 14 kez sola döndürülüyor.
RRC BL, 6       ;BL yazmacındaki bitler Carry aracılığıyla
                ;6 kez sola döndürülüyor.
```

8.6.6. Atlama ve Alt Program Yönergeleri

8085 mikroişlemcide olduğu gibi, 8086 mikroişlemcide de, atlamalar koşullu ya da koşulsuz olabilir. 8085 mikroişlemciden farklı olarak 8086 mikroişlemcide fazla koşullu atlama türleri vardır. Tablo 8.2.'de tüm koşullu atlama türleri, bayrakların gerekli durumları ve onların işlevi verilmiştir.

Yönerge	**Bayraklar**	**İşlev**
JA (Jump if above)	Z=0, C=0	üstünse atla (CMP yönergesinden sonra geliyor)
JAE (Jump if above or equal)	C=0	Daha üstün ya da eşitse atla
JB (Jump if below)	C=1	Daha alttaysa atla
JBE (Jump if below or equal)	Z=1 or C=1	Daha alta ya da eşitse atla
JC (Jump if carry)	C=1	Taşıma bayrağı ayarlanmışsa atla
JE or JZ	Z=1	Sıfır bayrağı ayarlanmışşa (sonuç=0) atla
JG (Jump if greater than)	Z=0, S=0	Daha büyükse atla
JGE (Jump if greater than or equal)	SF=OF	Daha biyik ya da eşitse atla
JL (Jump if less than)	SF≠OF	Daha küçükse atla
JLE (Jump if less or equal)	Z=1 ya da SF≠OF	Daha küçük ya da eşitse atla
JNC (Jump if not carry)	C=0	Taşıma bayrağı aktif değilse atla
JNE or JNZ (Jump if not equal or jump if not zero)	Z=0	Eşit değilse ya da sıfır bayrağı aktif değilse atla
JNO (Jump if no overlow)	OF=0	Aşma yoksa atla
JNS (Jump if no sign)	S=0	İşaret bayrağı aktif değilse (pozitif sonuç) atla
JNP or JPO (Jump if not parity or jump if parity odd)	P=0	Sonuç tek sayıdır
JO (Jump if overlow)	O=1	Aşma varsa atla
JP or JPE (Jump if parity or jump parity even)	P=1	Sonuç çift sayıdır
JS (Jump if sign set)	S=1	Negatif sonuç

Tablo 8.2. 8086 mikroişlemci için koşullu ve koşulsuz atlamalar

JG, JL, JGE, JLE, JE ve JNE yönergelerin önişaretli sayılara, **JA, JB, JAE, JBE, JE ya da JNE** yönergelerin önişaretsiz sayılara uygulandığını vurgulayalım. Sayılar 8 - bitliyse, o zaman önişaretli sayılar - 128'den +127'ye kadar olacak. Öni-

şaretsiz sayılar ise 0 ile 255 arası kapsamında olacak. Örneğin, +127(7FH) sayısına bir (01H) sayısı eklenirse 80H (- 128) sonucu elde ediliyor. İki bayrak, OF ve SF bayrakları aktifleştiriliyor ve ikisi de ayarlanıyor. Belli ki bu işlemin gerçekleştirilmesinden sonra, JGE (+127>+1) yönergesi uygulanabilir.

Üç tür koşulsuz atlama yönergeleri vardır: **kısa, yakın ve uzak atlama**. Kısa atlama yönergeleriyle +127 ve - 128 bayttan daha büyük olmayan atlamalar yapılıyor. Yakın atlama yönergleriyle +/ - 32 KB'a kadara atlamalar yapılıyor. Yakın atlamada kaydırma 16 - bitlidir. Uzak atlama yönergeleri mikroişlemcide herhangi bellek yerine kadar atlamaya izin veriyor.

Döngü terimi, verilen koşulun yerine getirilmesine kadar birkaç yönergenin periyodik tekrarlanması olarak tanımlanıyor. Döngü, koşullu atlama yönergelerin yardımıyla gerçekleşebilir. 8085'ten farklı olarak, 8086'da döngülerin gerçekleşmesi için özel yönergeler vardır. **LOOP** yönergesi CX yazmacını sayaç olarak kullanılıyor. LOOP yönergenin her çalıştırılmasıyla, CX yazmacının değeri bir için azalıyor. CX yazmacının değeri sıfır olunca, döngüden çıkıyor.

Örnek 8.17:

```
        MOV CX,50          ;Sayaca başlangıç değerin verilmesi.

DONGU:.....
        .....              ;Döngüden yönergeler
        LOOP DONGU             ;CX sıfır değilse DONGU'ye atlanıyor, aksi durumda
                           ;LOOP'tan sonraki yönergeyle devam ediliyor.
```

LOOPE ve LOOPNE yönergelerinde CX yazmacı dışında, sıfır bayrağın (Z) durumu da tespit ediliyor. LOOPE yönergesinde döngünün gerçekleşmesi için koşul şudur: CX≠0 ve Z=1. LOOPNE yönergesinde döngünün gerçekleşmesi için CX≠0 ve ZF≠0 olmalıdır.

Alt programlarla çalışmak için kullanılan yönergeler **CALL** ve **RET** yönergeleridir. CALL yönergesiyle ana programdan alt programa geçiş yapılıyor, RET yönergesiyle ise alt programdan ana programa geri dönülüyor. CALL yönergesi, JMP ve PUSH yönergelerin kombinasyonunu tanımlıyor. Alt programın çalıştırılmasından önce, yönerge göstergesi yazmacının içeriği yığıtta yazılıyor. Yönerge göstergesi alt programın çalıştırılmasından sonra sıradaki yönergenin adresini içeriyor. Bu adrese dönüş adresi deniyor ve alt programın tamamlanmasından sonra, ana programda alt programı çağırmadan önce kaldığımız yerden devam etmek için dönüş adresi gereklidir. Eğer program yakınsa (NEAR), o zaman yığıtta IP yazmacın içeriği saklanıyor. Eğer alt program uzaksa (FAR), o zaman yığıtta IP ve CS yazmaçların içerikleri saklanıyor.

8.7. 8086 Mikroişlemcide Programların Yazılması

8086 mikroişlemcide programlama, 8085 mikroişlemcide programlamasından, herşeyden önce **bölütlendirmesinden** dolayı farklıdır. Programda kullanılan tüm **bölütler tanımlanmalıdır**. Veri bölütünde tüm sabitleri ve değişkenlere giriyoruz, yığıt bölütün büyüklüğünü tanımlıyoruz, yönergeler ise kod bölütünde içeriktir. Bölütlerin tanımlanması için bildirim ya da emir olarak adlandırılan özel yönergeler kullanılıyor. Yönergelerle verilerin işletilmesini gerçekleştiriyoruz, bildirimler ise komutlardır, çevirici programın yönetimini yönlendiriyorlar. En çok kullanılan bildirimler şunlardır: DB (Define Byte), DW (Define Word), DD (Define Doubleword), DUP (Duplicate), EQU (Equate), SEGMENT, ASSUME, PROC, ENDP, USES

DB, DW ve DD bildirimleriyle değişkenlere sembolik isimler veriliyor. Bu bildirimlerle 8 ve 16 bitli veriler için veri bölütünde gerekli alan ayrılıyor. Aranan verinin bulunduğu bellek adresini hafıza etmek yerinde, verinin sembolik ismi çok daha kolay hafıza edilebilir. Aşağıda bu bildirimlerin kullanımıyla ilgili örnek verilmiştir. Bildirimler dışında, sol taraftan bellek yerlerin içerikleri ve onların adresleri verilmiştir. Adresler birinci sütundadır ve onlar aslında bellek yerin veri bölütünün başlangıcından ne kadar uzak olduğunu gösteren 16 bitli kaydırmadır. 8 - bitli veriler için alan ayırdığımız zaman, sıradaki adresin mevcut adresine bir ekleyerek, 16 bitli verilere ise mevcut adrese 2 ekleyerek elde edildiğini görüyoruz.

Örnek 8.18:

```
0000 FE DATA1 DB 254        ;Veri bölütünün birinci yerinde
                            ;256D=FEH onlu verisini giriyoruz
                            ;ve onu DATA 1 adıyla adlandırıyoruz.

0002 (??) DATA2 DB (?)      ;DATA 2 değişkeni için bir baytlık
                            ;bellek alanın ayırtılması
                            ;Soru işareti, bilinmeyen içerik tanımlıyor.
0004 09F0 DATA 3 DW 2544    ;Adresleri 0004H ve 0005H olan bellek yerleri,
                            ;sembolik ismi DATA4 olan, 2544D=09F0H
                            ;16 - bitli veriyi içeriyorlar
```

```
0008 (??) LIST DB 100 DUP (?)   ;DUP bildirimi ayrılan bellek
                                ;yerin ikiye katlanması demektir.
                                ;İkiye katlamaların sayısı DUP bildiriminden
                                ;sonra veriliyor. Verilen bildirimle içeriği
                                ;bilinmeyen 100 baytlık bellek alan ı ayrılıyor.
                                ;DUP bildirimi genelde dizilerle çalışmada kullanılıyor.
```

EQU bildirimi etiketler yaratmak için kullanılıyor. Etiketler bellek yerlerin sembolik isimleridir ve genelde atlama yönergelerde ve alt programlarda kullanılıyorlar. **SEGMENT** bildirimi bölütün başlangıcını işaretliyor, **ENDS** ise bölütün sonunu tanımlıyor. **ASSUME** bildirimi, programda kullanılacak veri, yığıt ve ekstra bölütünü işaretlemek için kullanılıyor. Aynı bir bölütün birkaç programda kullanılabileceğini hatırlayalım. Alt programın başlangıcın ve sonunu işaretlemek için **PROC** ve **ENDP** bildirimleri kullanılıyor. **USES** bildirimiyle yığıt belleğin tepesinde bulunan yazmaçların içerikleri otomatik olarak korunuyor.

Aşağıda yazılmış alt program genel yazmaçların içeriklerini ve 100 sayısının toplamını gerçekleştiriyor. Sonunda toplamanın sonucu, veri bölütünden TOPLAM ismi verilen bellek yerinde korunuyor.

```
TITLE TOPLA                     ;TITLE sözcüğü başlık demektir.
                                ;Programcı, program ismini kendi
                                ;isteğine göre seçiyor.
SSEG SEGMENT STACK              ;Yığıt bölütünü tanımlıyoruz ve
     DB 256 (?)                 ;büyüklüğünü 256 bayta ayarlanıyor.
SSEG ENDS

DSAG SEGMENT                    ;Veri bölütünü tanımlıyoruz.
     SAYI DB 100                ;İçinde 100 sabitini içeriyor
     TOPLAM DB ?                ;ve içeriği bilinmeyen veTOPLAM
DSEG ENDS                       ;olarak adlandırılan bir değişken içeriyor.


CSEG SEGMENT                    ;Kod bölütünü tanımlıyoruz.
ASSUME CS:CSEG, DS:DSAG,        ;Programdan hangi bölütlerin
SS:SSEG                         ;kullanılacağını vurguluyoruz.

TOPLA PROC FAR                  ;Alt programı adlandırıyoruz.

          PUSH DS               ;Veri bölüt yazmacını ve
          MOV AX,0              ;biriktiriciyi yığıtın tepesinde
```

```
        PUSH AX          ;koruyoruz.
        MOV BX, DSEG     ;Yeni veri bölütünü
        MOV DS, BX       ;giriyoruz.

    MOV AL, SAYI         ; Al yazmacının içeriğine önce 100 sayısını
    ADD AL, BL           ; ekliyoruz, ondan sonra ise diğer yazmaçların,
    ADD AL, CL           ; BL, CL ve DL'nin değerlerini ekliyoruz.
    ADD AL, DL
    MOV TOPLAM, AL       ;Elde edilen sonucu TOPLAM sembolik isimli
                         ;bellek yerinde yazdırıyoruz

RET                      ;Alt programdan ana programa
                         ;dönme yönergesi
TOPLA ENDP               ;Alt programın sonu.
CSEG ENDS                ;Kod bölütün sonu.
        END TOPLA        ;Programın sonu
```

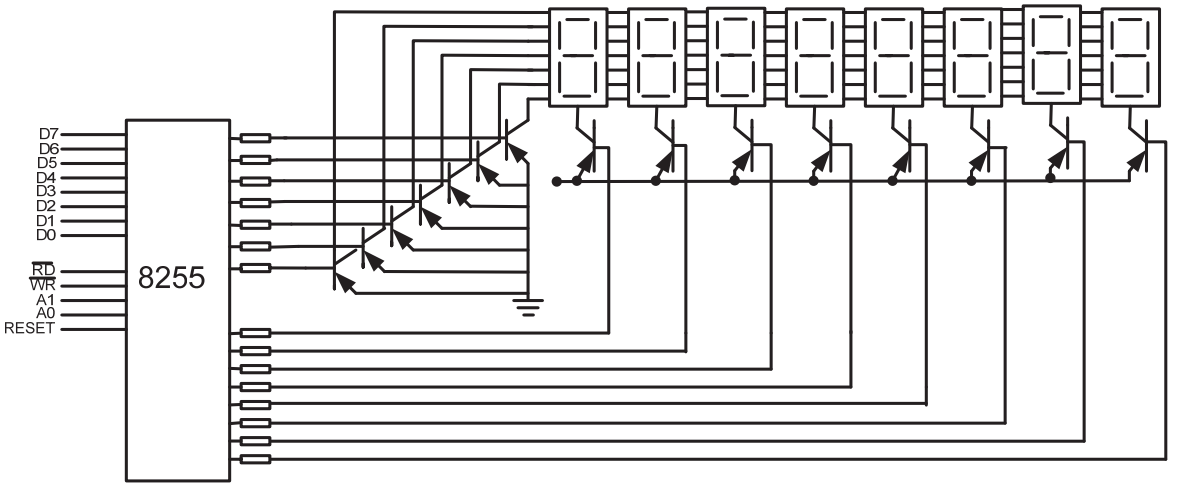
8.8. 8086 Mikrobilgisayar Sistemini Oluşturan Tümleşik Devreler

8.8.1. 8255 Tümleşik Devrenin Kullanımı

8086 mikroişlemcinin giriş - çıkış aygıtlarla bağlanması için programlanabilir 8255 arabirim - bileşeni kullanılıyor. Bu bileşenin pin diyagramını, programlanmasını ve çalışma düzenlerini, 8085 mikroişlemci için söz ettiğimiz bölümde tanımıştık. **8255 devresi Pentium mikroişlemcilerinin mikrobilgisayar sistemlerinde de kullanım görüyor**. Bu tümleşik devrenin işlevinin anlamak için iki örnek inceleyeceğiz: yedi bölütlü görünüm birimiyle ve klavyeyle bağlanmak.

Resim 8.11'de **8086 mikroişlemcinin yedi bölütlü görünüm birimiyle bağlanma devresi** tanımlanmıştır. Yedi bölütlü görünüm birimi sekiz alandan oluşuyor. Görünüm birimine verilerin gönderilmesi için çoğullama süreci kullanılıyor. Arabirimin tüm alanlarına **veriler A bağlantı noktasının pinler aracılığıyla gönderiliyor**, ancak veriler aynı zamanda değil, farklı zaman kapsamalarında gönderiliyor. **Alanın seçimi B bağlantı noktası aracılığıyla yapılıyor**. B bağlantı noktaların pinleri anahtar şeklinde çalışan PNP transistörlerin temelleriyle bağlıdır.

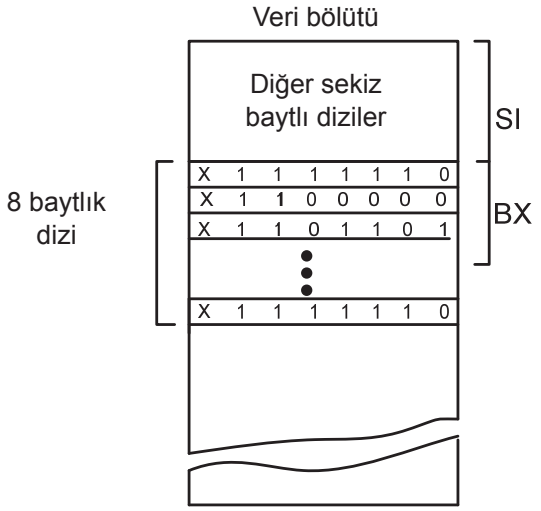
B bağlantı noktasında sadece bir pin alçak seviyede olacak, tüm diğer pinler yüksek seviyede olacak. Mantıksal sıfır seviyeli pin anahtarı (PNP transistörü) açıyor ve görünüm birimin o alanını seçiyor. Arabirimin diğer yedi alanı etkinsizdir. B bağlantı noktasına bir sıfır ve yedi birden oluşan veri gönderiliyor. Verideki sıfır sola veya sağa doğru döndürme yönergelerin yardımıyla devamlı olarak döndürülüyor. Görünüm biriminde alanlar ardaşıl şekilde, uygulanan döndürme yönergeye bağlı olarak soldan başlayarak sola doğru ya da ters yönde birer birer, aktifleştiriliyor. Biz görünüm biriminin tüm alanların aynı zamanda yanmadığını göremiyoruz, çünkü alanların aktifleştirilme hızı o kadar çabuktur ki, tüm alanların aynı zamanda yandığı izlenimi vardır.



Resim 8.11. 8086 mikroişlemcinin yedi bölütlü görünüm birimiyle bağlanması

Sekiz alanlı yedi bölütlü görünüm biriminin çoğullamasını gerçekleştiren alt programın incelemesine başlamadan önce, yedi bölütlü kodları hatırlayalım. Çünkü bu kodlar A bağlantı noktası aracılığıyla görünüm birimine gönderiliyor.

Veri bölütünde, görünüm biriminin sekiz alanına aktarılması gereken, **7 - bölütlü kodları** içeren 8 baytlık dizi yaratılıyor. Veri bölütündeki diziyi, SI yazmacın yardımıyla buluyoruz. SI aslında, veri bölütünün başlangıcından dizinin başlangıcına kadar mesafeyi veriyor. Birinci bayt, dizideki birinci kodun, birinici sol alana girilmesi gerekiyor, ikinci bayt birincinin yanındaki alana, üçüncü bayt soldan üçüncü alanda vs. BX yazmacı 8 baytlık diziden, dipten başlayarak tepeye doğru hareket etmek için kullanılıyor. BX yazmacının değeri sıfır olana kadar bir için azalıyor. 8 baytlı dizili veri bölütü resim 8.12.'de gösterilmiştir.



Resim 8.12. Yedi bölütlü kodlarla veri bölütü

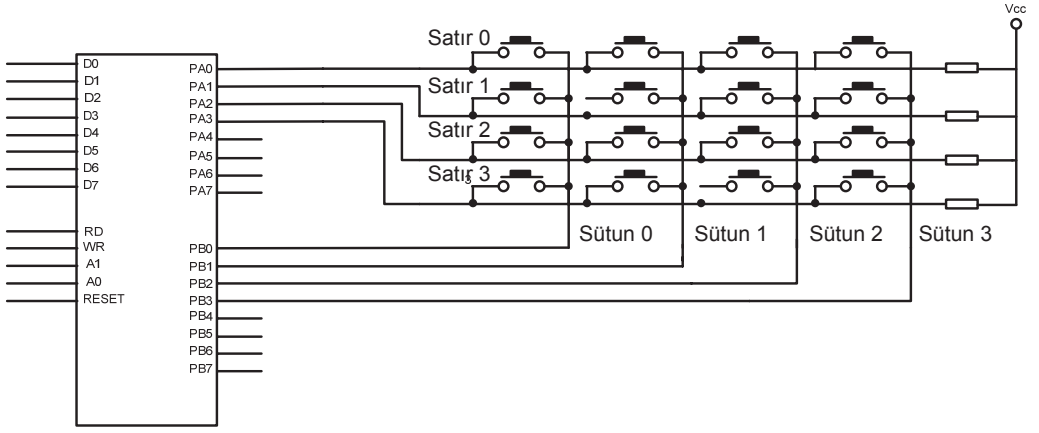
Görünüm biriminin çoğullama alt programında yeni alt program olan **DELAY alt programı** çağrılıyor. Bu alt programla, görünüm biriminde yan yana olan iki alanın aktifleştirilmesi arasında 1 ms gecikme giriliyor. Gecikme zamanın seçimi, bazı LED ekran üreticilerin önerilerine göre uyumludur.

Çoğullamak alt programında, A bağlantı noktasının 700H adresi var, B bağlantı noktanın ise 701H adresi var.

```
DISP    PROC
        PUSHF               ;Durum yazmacın içeriği yığıt
                            ;belleğin tepesinde yerleşiyor.
        MOV BX,8            ;Sayaç 8 değeriyle dolduruluyor.
        MOV AH,7FH          ;Alanların seçimi için 01111111B
        ;maskesi seçiliyor
        MOV DX,701H         ;B bağlantı noktasının adreslenmesi
        MOV SI, BASLANGIC   ;Veri bölütünde 8 baytlık dizinin
                            ;bulunması
DISP1   MOV AL, AH          ;Alanın seçimi
        OUT DX,AL
        DEC DX              ;A bağlantı noktasının adreslenmesi
        MOV AL,(BX+SI)      ;7 - bölütlü kod gönderiliyor
        OUT DX,AL
        CALL DELAY          ;1ms bekleniyor
        ROR AH,1            ;Sağdan sıradaki alana erişim
        INC DX              ;B bağlantı noktanın adreslenmesi
        DEC BX              ;Sayacın bir için azalması
        JNZ DISP            ;Sekiz kez tekrarla
```

```
        POPF            ;Durum yazmacının dönmesi
        RET             ;Alt programın sonu
DISP    ENDP
```

Şimdi **klavyeyle bağlanma sırasında, 8255 bileşenin kullanımını** açıklayacağız. Klavyeler farklı büyüklükte olabilir. Sıradan (Standart) klavye 101 tuş (düğme) içeriyor. Resim 8.13.'te gösterilen örnekte, klavye 16 tuş içeriyor ve bu 16 tuş 4 sırada ve 4 sütunda sıralanmıştır. Bu şekilde 4x4 boyutlu matris oluşuyor. **A bağlantı noktası** giriş bağlantı noktasıdır ve **satırlardaki tuşların durumunu okumak için kullanılıyor**.



Resim 8.13. 8086 mikroişlemcinin klavyeyle bağlanması

Dört satır olduğu için, A bağlantı noktasından dört pin kullanılacak (PA_0 - PA_3). Her satır, 10 KΩ pull up direnç aracılığıyla, V_{CC}=5V gerilime bağlıdır. Gözetlenen satırdan hiçbir tuş basılık değilse, o zaman A bağlantı noktasından pin mantıksal bir seviyesinde olacak. **B bağlantı noktası çıkış bağlantı noktasıdır ve sütunun seçimi için kullanılıyor**. Dört sütun olduğu için B bağlantı noktasından dört pin kullanılacak (PB_0 - PB_3). Sütunun seçimini örnek ile açıklayacağız. 1110, PB_3 - PB_0 dört pini için çıkış verisi ise, o zaman sıfırıncı sütun seçilecektir. B bağlantı noktasının dört pininden, sadece PB_0 pini mantıksal sıfır seviyesinde olacak.0'dan 3'e kadar bir tuş bastırılmışsa, o zaman otomatik olarak, A bağlantı noktasından uygun pine, mantıksal sıfır götürülüyor. Diğer satırlardan tuşlar da basılmış olabilir, ancak onlar A bağlantı noktasında pinlerin durumuna etkilemeyecek. B bağlantı noktası için çıkış verisi 1101 ise, o zaman 1 numaralı sütuna ait olan tuşlar seçilmiş olacaktır. Klavyedeki düğmeler anahtar değildir, düğmeler tuşlardır. Düğme basıldıktan sonra, düğme o durumda kalmıyor. Basılan düğme geri dönüyor, yani serbest bıraklılıyor. Tuşun bırakılması saniyenin birkaç bölümü kadar sürüyor. Bizim için bu zaman süresi gerçek dışı olabilir, ancak mikroişlemci için bu süre uzun zaman aralığı olabilir.

Mikroişlemci basılan düğmenin, serbest bırakılmasından önce durumu tespit edebilir. Mikroişlemci bunu aynı tuşun ikinci kez basılması gibi yorumlayabilir ve bu

tabii ki doğru değildir. Bundan dolayı bu gibi hataların meydana gelmemesi için, tuşların iki ardaşıl kontrolu arasında gecikme eklenmelidir.

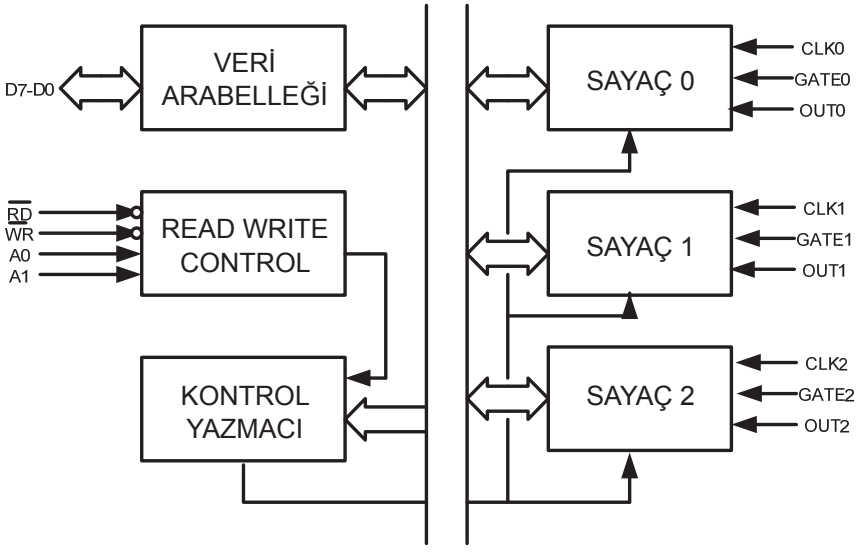
8.8.2. Programlanabilir 8254 Zamanlayıcı

İşletim yönetimi, mikroişlemci sisteminin kullandığı dijit palslardan, çok daha düşük frekanslı dijit palsları gerektiriyor. Giriş frekansı 8MHz'tir. **Programcı dijit palsın frekansını ve tetikleme şeklini değiştirebilir**. Programlanabilir 8254 zamanlayıcı INTR (18.2 Hz) pinin kontrolü ve DRAM belleğin yenilenmesi için gereken dijit palsın yaratılması için kullanılıyor. Resim 8.14.'te 8254 zamanlayıcının blok - diyagramı verilmiştir.

8254 zamanlayıcının dört iç yazmacı var: sayaç 0, sayaç 1, sayaç 2 ve komut yazmacı. Bu yazmaçların adreslenmesi için A1 ve A0 adres girişleri kullanılıyor. Adresleme tablo 8.3.'te gösterildiği şekilde yapılıyor.

A1	A0	Yazmaç
0	0	Sayaç 0
0	1	Sayaç 1
1	0	Sayaç 2
1	1	Kontrol yazmacı

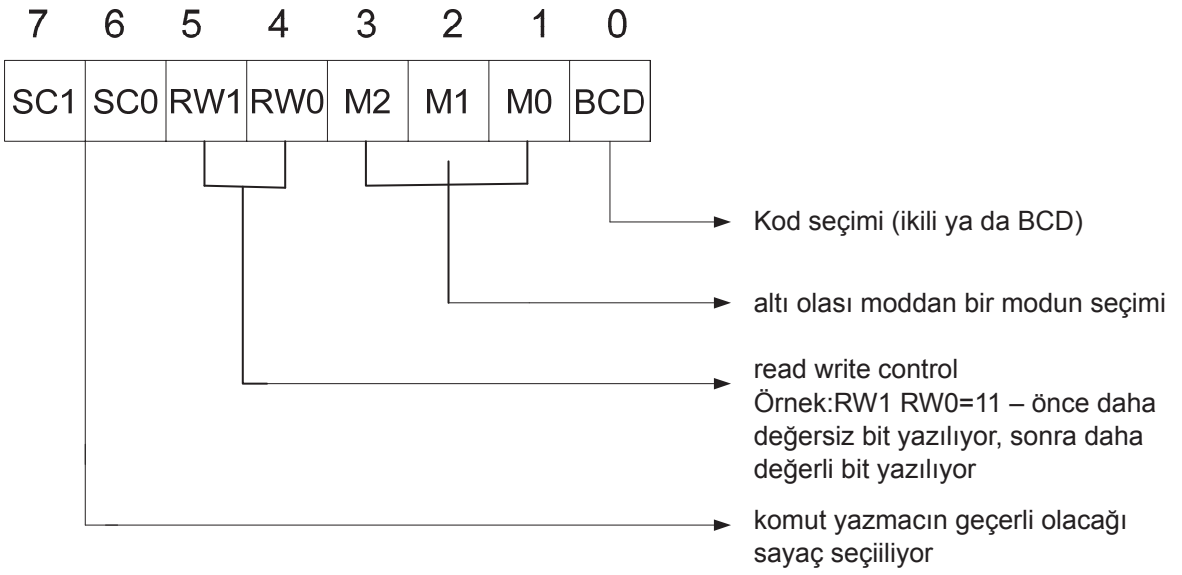
Tablo 8.3. Programlanabilir 8254 zamalayıcı iç yazmaçların adreslenmesi



Resim 8.14 Programlanabilir 8254 yazmacının blok modeli

Her sayacın üçer pini var. CLK giriş dijit palsı içindir, OUT çıkış dijit palsı içindir ve G(gate) kontrol ve sayacın etkinleştirilmesi için kullanılan pindir. Sayaçlar 16 - bitlidir. Bir sayacın programlanabileceği en yüksek değer FFFFH'dır.

Programlama komut yazmacın yardımıyla gerçekleşiyor. Komut yazmacı aracılığıla mod (düzen) ve sayaç seçiliyor. Mod giriş sinyaline bağlı olarak çıkış sinyalin şeklini tanımlıyor. Resim 8.15.'te komut yazmacındaki her bitin anlamı verilmiştir.



Resim 8.15. Programlanabilir 8254 zamanlayıcında komut yazmacındaki bitlerin işlevsel açıklaması

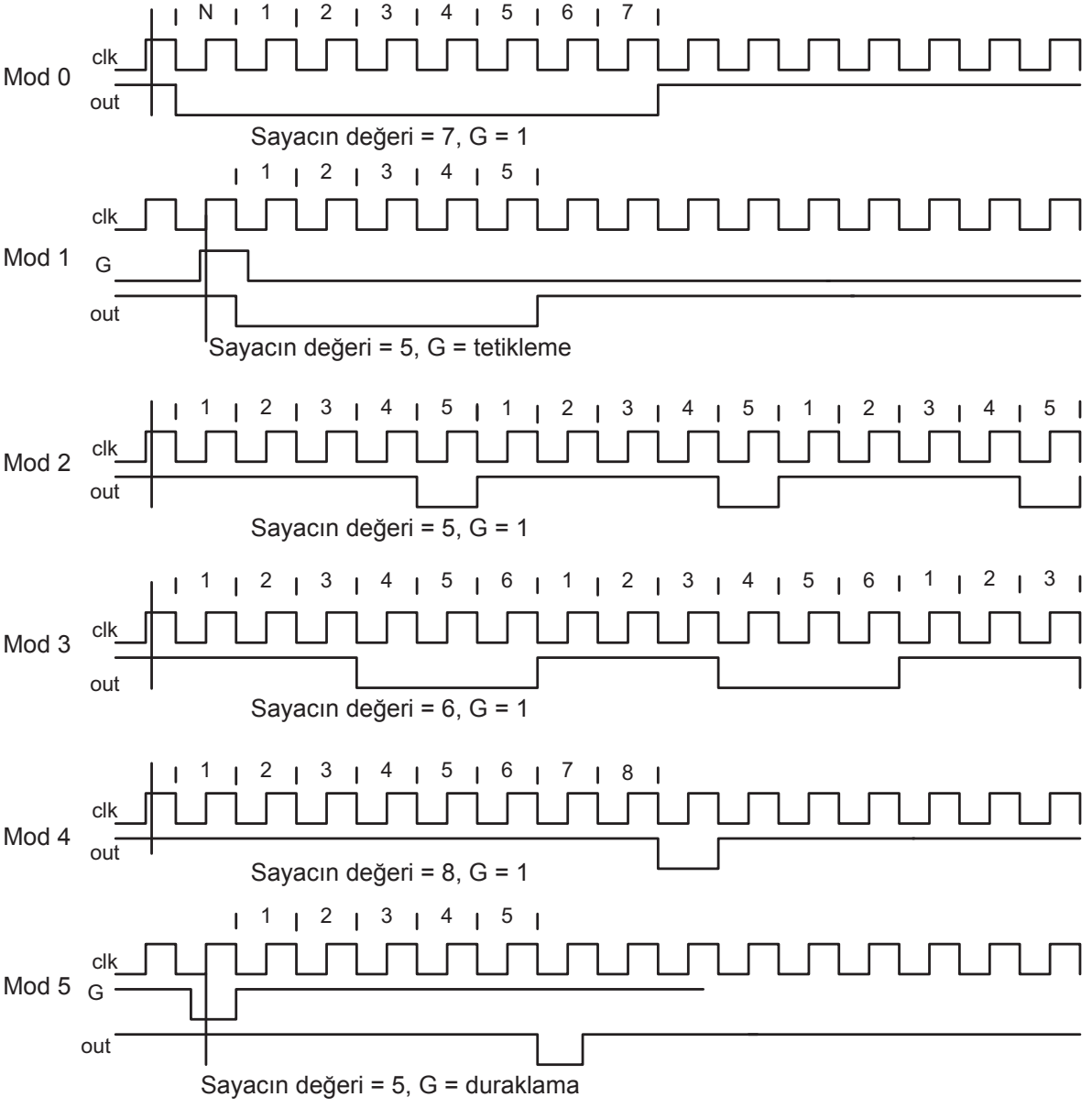
8254 zamanalayıcı 6 farklı mod'ta çalışabilir ve onlar zamanlama diyagramlarıyla beraber resim 8.16.'da verilmiştir.

Mod 0 – Bu çalışma modunda, OUT pini, sayma sürünce mantıksal bir olacak. Sayma, komut yazmacının değeri yazıldığı andan başlıyor. Sayma sırasında, G pini yüksek seviyede olmalıdır. Sayma sırasında G pini alçak seviyeye düşerse, o zaman sayma durdurulacak ve G pinin sinyalı yeniden mantıksal bir olunca sayma devam edecek.

Mod 1 – Sayaç programlanıyor ve saymaya başlaması için G pininden tetikleme bekliyor. Sayma sırasında G pinine bir tetikleme daha meydana gelirse, o zaman mevcut sayma tamamlanınca, yeni sayma başlayacak.

Mod 2 – Bu çalışma modunda, G pinin sinyali devamlı yüksek seviyede olmalıdır. Örneğin, sayaç 10 değerine programlanırsa, o zaman OUT pininde sinyal dokuz dijit atışı boyunca mantıksal bir olacak, onuncu atışta ise sıfır olacak. Tetikleme dizisi, yeniden programlama gerçekleştirilene kadar ya da G pinin sinyali mantıksal sıfır olana kadar devam ediyor. Bu diziye sürekli tetikleme dizisi denir.

Mod 3 – Önceki mod'ta olduğu gibi, bu çalışma modunda da geçit (G) yüksek seviyede olmalıdır. Örneğin, eğer COUNT = 4, o zaman OUT pininde sinyal iki atış boyunca mantıksal bir olacak, diğer iki atışta mantıksal sıfır olacak. Bu sinyale kare sinyalı deniyor.

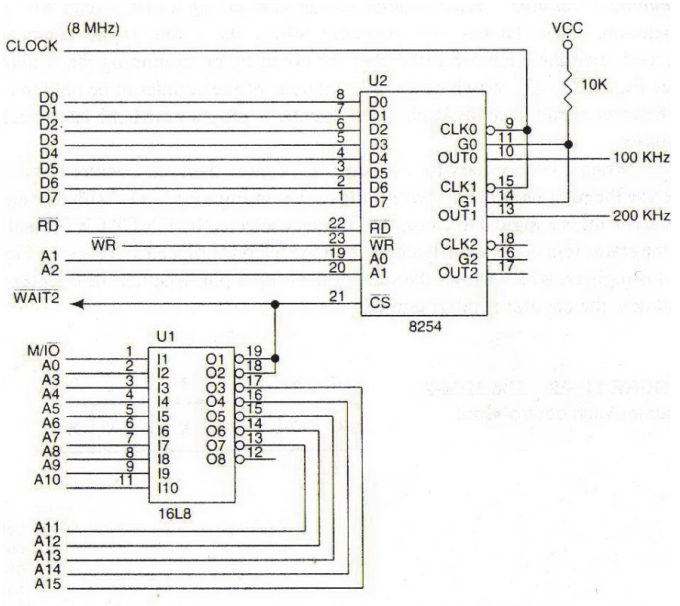


Resim 8.16. Programlanabilir 8254 zamanlayıcıda çıkış sinyalinin giriş sinyalinden bağımlılığın zamanlama diyagramı.

Mod 4 – Önceki iki çalışma modunda olduğu gibi, bu çalışma modunda da geçid (G) yüksek seviyede olmalıdır. OUT pininde, sayacın programlandığı kadar atışların sayılmasınla sadece bir tetikleme meydana gelecek.

Mod 5 – Bu mod, mod 4 gibidir, sadece burada saymanın başlaması için G pininde tetiklemenin meydana gelmesi gerekiyor, yani sinyalin 1'den sıfıra düşmesi ve ondan sonra yeniden 1'e çıkması gerekiyor.

Resim 8.17.'de 8254 zamanlayıcının, çalışma frekansı 8MHz olan 8086 mikroişlemciyle bağlanma modeli gösterilmiştir.



Resim 8.17. 8254 zamanlayıcının 8086 mikroişlemciyle bağlanması

Örnek 8.20:

8254 zamanlayıcındaki komut yazmacın ve sayaçların içeriklerini belirle, öyle ki sayaç 0'ın çıkışında 100KHz frekanslı kare sinyalı, sayaç 1'in çıkışında 200 KHz frekanslı sürekli tetikleme dizisi elde ediliyor.

Çözüm:

Komut yazmacın içeriği, her farklı sayaç için farklı olacak. Sayaç 0 için mod 3'ü uyguluyoruz ve sayaç 0'ı 80 değerine programlıyoruz (8MHz: 80 = 100KHz). Sayaç 1 için mod 2'yı uyguluyoruz ve sayacı 40 değerine programlıyoruz (8MHz: 40 = 200KHz).

Açıklanan hedefin gerçekleşmesi için alt programa TIME ismi verilmiştir. 8254 zamanlayıcı 0700H, 07002H, 0704H ve 0706H arabirim adreslerini kullanıyor.

```
TIME    PROC    NEAR
        PUSH    AX              ;yazmaçların içeriklerini
        PUSH    DX              ;koruyoruz

        MOV     DX, 706H        ;komut yazmacın adresi
        MOV     AL,00110110B    ;komut yazmacın içeriği
```

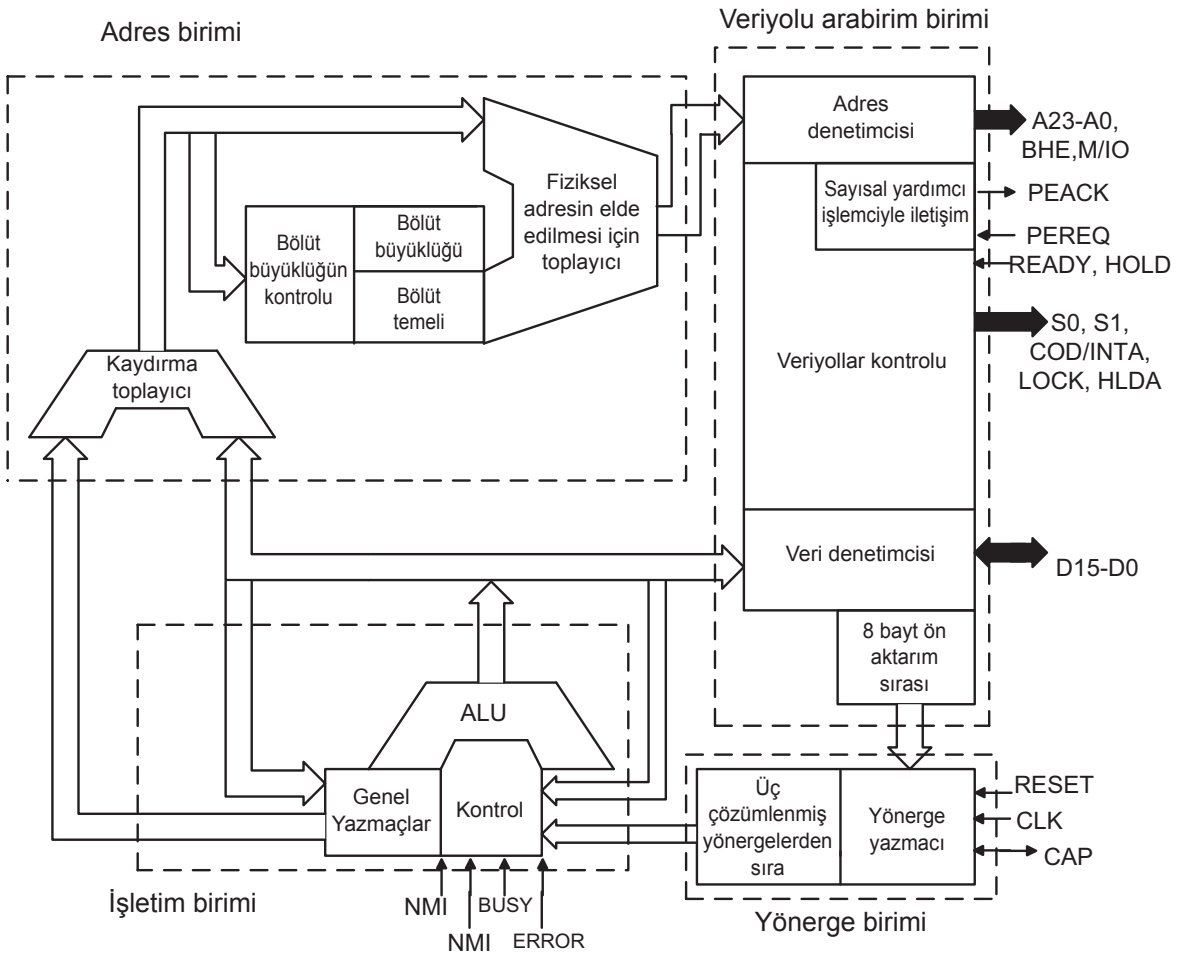
```
        OUT     DX,AL           ;sayaç 0 için
        MOV     AL,01110100B    ;komut yazmacın içeriği
        OUT     DX,A L          ;sayaç 0 için

        MOV     DX,700H         ;sayaç 0'ın adresi
        MOV     AL,80           ;sayaç 0'ın programlanması
        OUT     DX,AL

        MOV     DX,702H         ;sayaç 1'in adresi
        MOV     AL,40           ;sayaç 1'in programlanması
        OUT     DX,AL

        POP     DX
        POP     AX
        RET
TIME    ENDP
```

8.9. 80286 Mikroişlemcinin Temel Özellikleri

80286 işlemcisi, 8086 mikroişlemcinin geliştirilmiş şeklidir ve birden fazla ödev ve kullanıcı hizmetleri paralel olarak gerçekleştirme olanağı var. 80286 mİkroişlemci 16 MB'a kadar fiziksel belleği ve özel bellek yönetim birimi aracılığıyla 1GB sanal belleğe sahiptir. Yönergelerin gerçekleştirilmesi için dijit palsların sayısı azalmıştır. 80286 mikroişlemcisi artık kişisel bilgisayarlarda kullanılmıyor, ancak işletim yönetimi için bilgisayar sistemlerinde hala kullanım görüyor.

Resim 8.18.'de bu mikroişlemcinin **yapısı** gösterilmiştir.

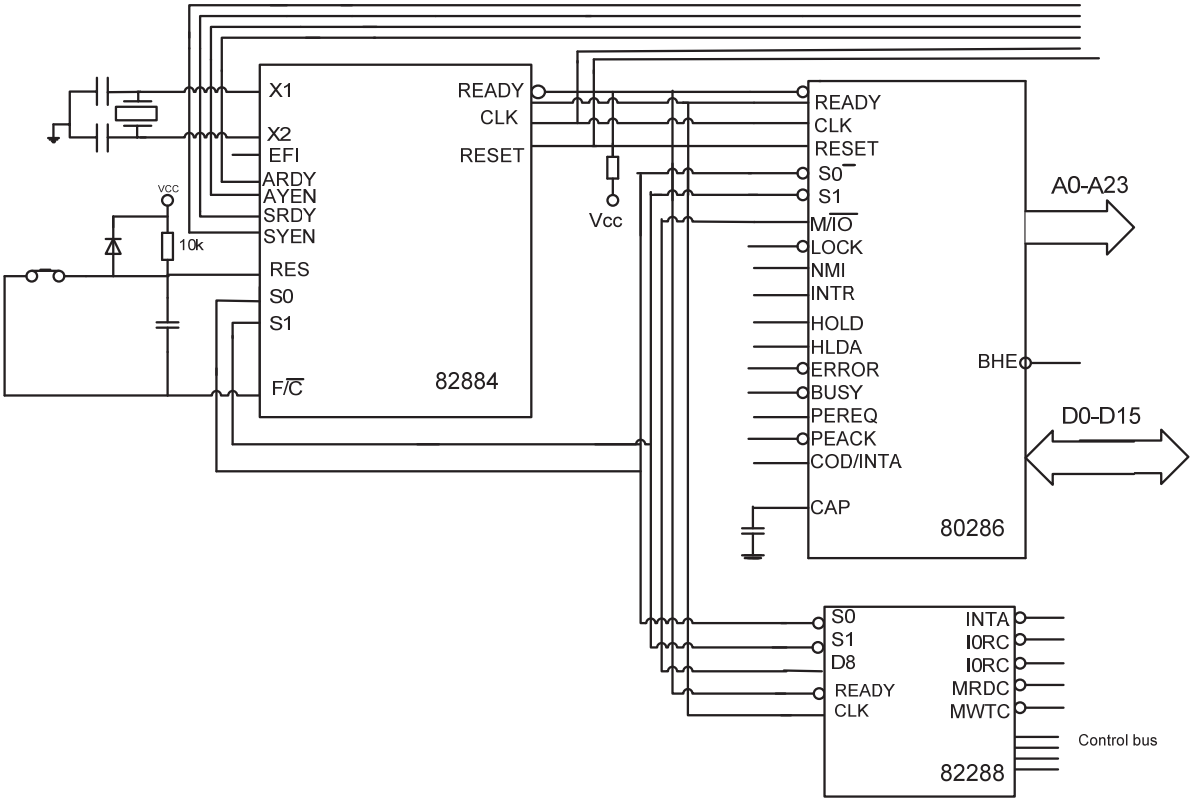
Bellek yönetim birimi adres birimi olarak adlandırılıyor. 8086 mikroişlemcide benzer şekilde, fiziksel adresleri hesaplamak için, bölüt yazmaçları ve kaydırma (bölütün başlangıcından istenen yere kadar olan mesafe). Aritmetik mantık birimi ve genel yazmaçlar 16 - bitlidir. Veriyollarla yönetim birimi, makine döngüsü seçimi için (S0, S1), DMA denetimcinin yönetimi için (HOLD, HLDA) ve kesintileri için (INTR, COD/INTA) durum sinyallerini üretiyor. 8086 mikroişlemciden farklı olarak, 80286 mikroişlemcisi iki yönerge sırası içeriyor, biri yönerge kod çözücüsü önünde ve diğeri kod çözümlenmiş yönergelerle. Adres veriyolu 24 - bitlidir ve 16MB'lık gerçek bellek alanın adreslenmesi için kullanılıyor. 80286 mikroişlemcinin çoğullanmış adres veri veriyolu kullanmadığını vurguluyoruz.



Resim 8.18. 80286 mikroişlemcinin yapısı

BUSY, ERROR, PEREQ ve PEACK **pinleri** yenidir ve onlar 80286 mikroişlemcinin matematik işlemci yardımcıyla iletişim için kullanılıyor. BUSY pinin, 8086 mikroişlemcide TEST pinin olduğu aynı işlevi var ve beklemek için tam sayı atışların eklenmesi için kullanılıyor. ERROR pini çift sayı kontrolünde hata meydana gelince aktifleştiriliyor.PEREQ ve PEACK pinlerin aracılığıyla matematik yardımcı işlemcisi çalışma için istek ve izin veriyor.

Resim 8.19.'da 80286 mikroişlemcinin **82288 veriyolu denetimcisiyle ve 82884 pals üreticisiyle** bağlanma şekli gösterilmiştir. Bu iki **tümleşik devre** 80286 mikroişlemcinin mikrobilgisayar sisteminin parçasıdır. Bu sistemi daha iyi anlamak için, 8086 mikroişlemci sistemiyle maksimum düzende çalışınca kıyaslanması gerekiyor (resim 8.5.). Bu resimde 8288 veriyolu denetimcisi kullanılmıştır ve resim 8.19.'da gösterilmiş 82288 denetimciye çok benzerdir. Kontrol sinyallerin üretim şekli aynıdır. Mikroişlemci S0, S1 ve IO/$\overline{M}$ durum bitlerini gönderiyor, onların kodu çözümleniyor ve denetimcinin çıkışında bellek kontrol sinyalleri (MRDC, MWTC) ve dış aygıtlar kontrol sinyalleri (IORC, IOWC) elde ediliyor.



Resim 8.19. 80286 mikroişlemcinin 82884 dijit pals üreticisiyle ve 82288 veriyolu denetimcisiyle bağlanması

82284 dijit pals üreticisi 8284 pals üreticisiyle çok benzerdir. Çıkış pinleri şunlardır:

33. CLK (clock) - Bu sinyal mikroişlemci ve sistemde diğer aygıtlar için giriş sinyalidir.
34. READY - Bu pin alçak seviyede olduğu zaman mikroişlemci bekleme durumuna giriyor. Bu pin yüksek seviyede olunca, mikroişlemcinin çalışmasına etkilemiyor.
35. RESET - Bu pin mikroişlemcinin sıfırlandırılması için kullanılıyor.
36. PCLK (Peripheral Clock) - Bu çıkış dijit palsın, EFI pinin dijit palsının frekansından iki kat daha düşük frekansı var.

82284 pals üreticisinin giriş pinleri şunlardır:

- X1, X2 - Bu iki pin kristal salıngaçın takılması için kullanılıyor
- EFI (External Frequency Input) - Bu pin dış dijit pals kaynağın eklenmesi için kullanılıyor.
- RES (Reset) - RES pinin aracılığıyla elektrik kaynağın sıfırlandırılması gerçekleşiyor ve RC devresine bağlıdır.

- F/C (Frequency Crystal Select) – Bu pinin yardımıyla dijit palsın kaynağı seçiliyor: X1, X2 ya da EFI.
- ARDY, SRDY (Asynchronous Ready Enable, Synchronous Ready Enable) - Bu iki pin READY sinyalın olası iki kaynağından birini seçmek için kullanılıyor.
- AYEN, SYEN (Asynchronous Ready, Synchronous Ready) - Bu iki pinden biri alçak seviyede olduğu zaman READY çıkış sinyali aktifleştiriliyor.
- S0, S1 - Durum pinlerini mikroişlemci gönderiyor ve makine döngü türünün tanımlanması için kullanılıyor.

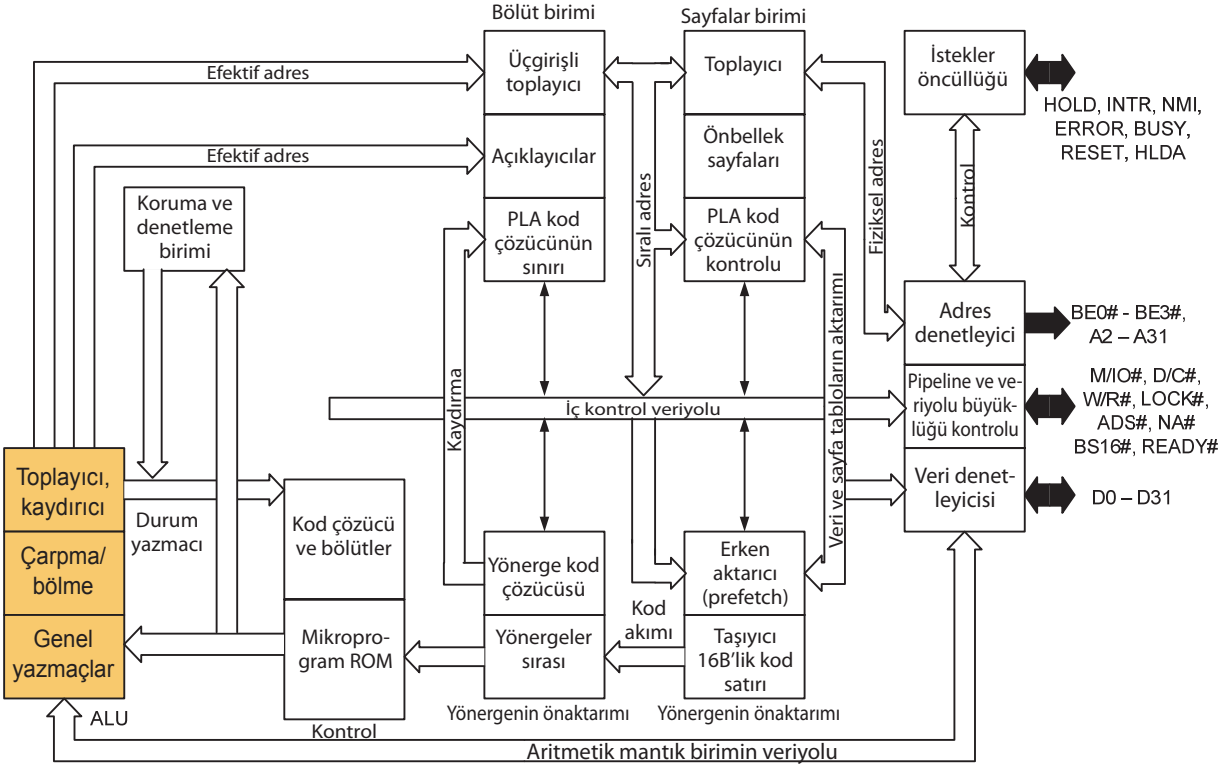
8.10. 80386 Mikroişlemcinin Temel Özellikleri

80386 işlemcisi, 8086 mikroişlemcinin 32 - bitli şeklidir. Verilerin büyüklüğünü iki kat artırtması dışında, 80386 mikroişlemcide çoklu işlemede, bellek organizasuonunda, sanal bellekte, yazılımın korumasında vs. birçok bakımdan gelişmiştir. Korumalı ve sanal çalışma moduyla bir sonraki konuda tanışacağız. **Veri ve adres veriyolu 32 - bitlidir**. Bellek alanın büyüklüğü 4GB'tır. 80386 mikroişlemcisi, mikroişlemciyi sıfırlandırmadan, gerçek çalışma modundan korumalı ve çalışma moduna geçiş sağlıyor. Bu seçenek önceki mikroişlemci nesillerde yoktu.

Resim 8.20.'de 80386 mikroişlemcinin yapısı ve onun **giriş ve çıkış sinyalleri** gösterilmiştir.

80386DX mikroişlemcinin 30 adres hatı var, A2'den A32'ye kadar ve toplam 1GB konumdan 32 - bitli yerin seçimi için kullanılıyor. Seçilen yerde,dört bayttan bir baytın seçimi için A0 ve A1 adres hatları yerinde, $\overline{BE3}$ - $\overline{BE0}$ dört kontrol sinyalleri kullanılıyor. M/IO pini aygıt seçimi için kullanılıyor (1 - bellek, 0 - dış aygıt), W/R pini ise işlem seçimi için kullanılıyor (1 - yazma, 0 - okumak). $\overline{ADS}$ (Address Data Strobe) pini, her geçerli bellek ya da arabirim adresi gönderildiği zaman aktifleştiriliyor. CLK2 pini, mikroişlemciden iki misli daha yüksek frekanslı sinyal veriyor. RESET pinin aktifleştirilmesiyle, mikroişlemci FFFFFFF0H adresli yerinden yazılımın çalıştırmasını başlatıyor. $\overline{READY}$ sinyali bekleme durumların sayısını kontrol ediyor. D/$\overline{C}$ (data/control) pini yüksek seviyede bulunuyorsa, o zaman very aktarılıyor, alçak seviyede olduğu zaman ise mikroişlemci kesinti işletimi için makine döngüsünü çalıştırıyor. $\overline{BS16}$ (Bus Select 16) pini, 16 ile 32 - bitli veriyolu arasında seçim yapıyor.

$\overline{NA}$ (Next Address) sinyali, sıradaki yönergenin adresinin *pipeline* kavramında gönderilmesi için kullanılıyor. HOLD ve HLDA pinleriyle DMA aktarımı kontrol ediliyor. Aritmetik yardımcı işlemcisi, 80386 mikroişlemciden, $\overline{PEREQ}$ (Coprocessor Request) pini aracılığıyla çalışma izni arıyor. $\overline{BUSY}$ pini mikroişlemci için giriş pinidir ve sadece yardımcı işlemci meşgul olduğu zaman aktiftir. Yadımcı işlemcinin çalışmasında hata meydana gelirse $\overline{ERROR}$ pini aktifleştiriliyor. INTR ve NMI pinleri donanımsal kesinti ya da maskelenmemiş kesinti durumunda aktifleştiriliyorlar.

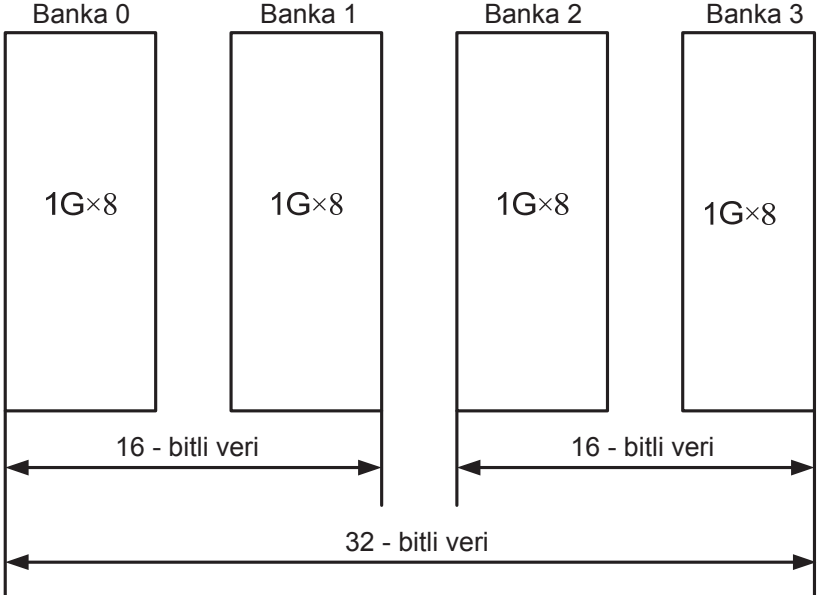


8.20. 80386 mikroişlemcinin yapısı

80386 mikroişlemcide yazmaçlar birkaç gruba ayrılıyor: genel yazmaçlar, yönerge göstergeleri, durum yazmaçları, bölüt yazmaçları, bölüt açıklayıcı yazmaçlar, kontrol yazmaçları, sistem adres yazmaçları ve denetleme ve hata giderme yazmaçları (hata ayıklama – İngilizce debug yazılım hataların gidermesi demektir). Genel yazmaçlar, yönerge göstergesi, durum yazmaçlar ve bölüt yazmaçların 8086 mikroişlemcide gibi aynı işlevleri var. **Genel yazmaçlar** 8, 16 ve 32 - bitli olabilir. 8 - bitli yazmaçların L (low) ve H (high) sonekleri var. 16 - bitli yazmaçların X sonekleri var: AX, BX, CX, DX. 32 - bitli yazmaçlar 16 - bitli yazmaçların önünde E (extended) harfini ekleyerek işaretleniyor: EAX, EBX,ECX, EDX. Yönerge ve durum yazmaçları aynı öyle 32 bitlidir, bölüt yazmaçları ise 16 - bitlidir. **Bölütlerin açıklayıcıları**, bölütlerin başlangıç fiziksel adresleri, bölütlerin büyüklüğü ve di-

ğer özellikleri hakkında bilgi veren yazmaçlardır. Onlar programlanamaz, programcı için görünmezdir, ancak 80386 mikroişlemcinin adreslenmesini anlamak için son derece önemlidir. **Üç kontrol yazmacı** vardır: CR0, CR2 ve CR3. CR1 kontrol yazmacı da vardır ancak kullanılmıyor ve ilerdeki İNTEL mikroişlamcileri için ayrılmıştır. CR0 yazmacın 16 daha değersiz bitleri makine durum sözünü içeriyor. Makine durum sözü, makine döngü türü hakkında bilgi içeriyor. CR2 kontrol yazmacı son çağrılan sayfanın 32 - bitli sıralı adresini içeriyor, CR3 yazmacı ise terminal dosyanın fiziksel temel adresini içeriyor. Sistem adres yazmaçları bölütlerin desteklenmeleri için dört özel tabloya erişim için kullanılıyor. Bu tablolar şunlardır: genel (global) açıklayıcılar tablosu, açıklayıcıların kesinti tablosu, yerel açıklaycılar tablosu ve verilen anda gerçekleşen görevin durum bölütü. Bu tabloların işlevlerini devamda tanıyacağız. Aynı tabloların Pentium işlemcilerin adreslenmesinde kullanıldığını not edelim.

80386 mikroişlemcinin belleği **dört bellek bankasına** ayrılıyor ve her bankanın 1GB kapasitesi var. 8, 16 ve 32 - bitli verilere doğrudan erişim mümkündür. 32 - bitli verinin okunması için, 80386 mikroişlemciye bir makine döngüsü gerekiyor.



Resim 8.21. 80386 mikroişlemcisi için RAM belleğin organizasyonu

8086 mikroişlemciye dört makine döngüsü gerektiğini hatırlayalım. Bellekte her yerin kendine ait 32 - bitli adresi var, 00000000H'dan FFFFFFFFH'ye kadar. Resim 8.21.'de dört bellek bankası gösterilmiştir. Banka seçimi için $\overline{BE3}$ - $\overline{BE0}$ (Bank Enable) kontrol sinyalleri kullanılıyor. Bir bayta kadar erişim için bir bankanın seçilmesi gerekiyor, 16 - bitli veriye erişim için ise iki bankanın seçilmiş olma-

sı gerekiyor. Sözler (16 - bitli veriler) genelde yan yana olan iki bankada yerleşiyor, banka 0 ve 1 veya banka 2 ve 3. 00000000H adresli konum banka 0'da bulunuyor, 00000001H adresli konum banka 1'da bulunuyor, 00000002H adresli konum banka 2'da bulunuyor vs.

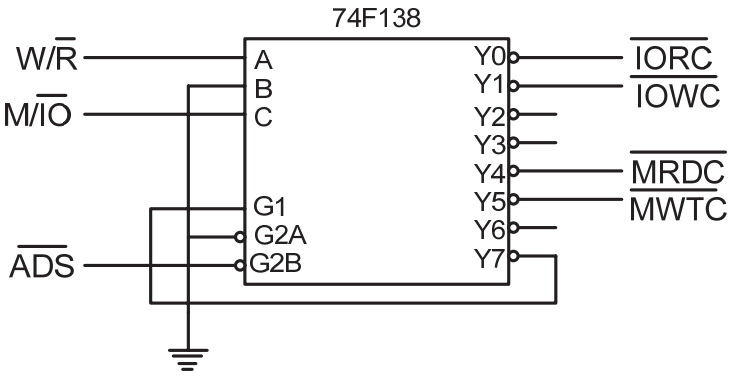
80386 mikroişlemcinin bellekle senkronize edilmesi için, zamanlamada bekleme atışların eklenmesi ya da veri aktarımını hızlandıracak yeni teknikler uygulamak gerekecek. Kullanılan bazı teknikler şunlardır: pipeline, önbellekler ve iç izinli bellek (interleaved memory).

Önbellek mikroişlemcinin daha sıkça kullandığı verileri saklamak için kullanılıyor. Önbellek SRAM belleğidir, erişim samanı 25ns'dir ve DRAM bellek ve mikroişlemci arasında aracılık yapıyor. 80386 mikroişlemcide, önbellek mikroişlemci yonganın belleğine takılmış değildir ve bu tür önbellek ikinci seviye önbellek adıyla biliniyor (level two cache). Devamda, mikroişlemci yongasında tümleşik, birinci seviye önbelleklerin (level one cache) de varolduğunu göreceğiz.

İç izinli bellek (interleaved memory) mikroişlemci sistemlerin hızlandırılması için kullanılan başka bir yöntemdir. Ancak bu yöntem daha pahalıdır, çünkü iki adres veriyolu ve onları yöneten özel denetimci gerektiriyor. İç izinli bellekte, bellek iki özel bölüme ayrılıyor. Birinci bölüm 00000000H - 00000001H, 00000004H - 00000005H vs. adresli konumları içeriyor, ikinci bölüm ise 00000002H - 00000003H, 00000006H - 00000007H vs. adresli konumları içeriyor. Bu yöntem, mikroişlemci 00000000H - 00000001 H adresli 16 - bitli veriyi işletirken, 00000002H - 00000003H adresli 16 - bitli veriye erişim sağlıyor.

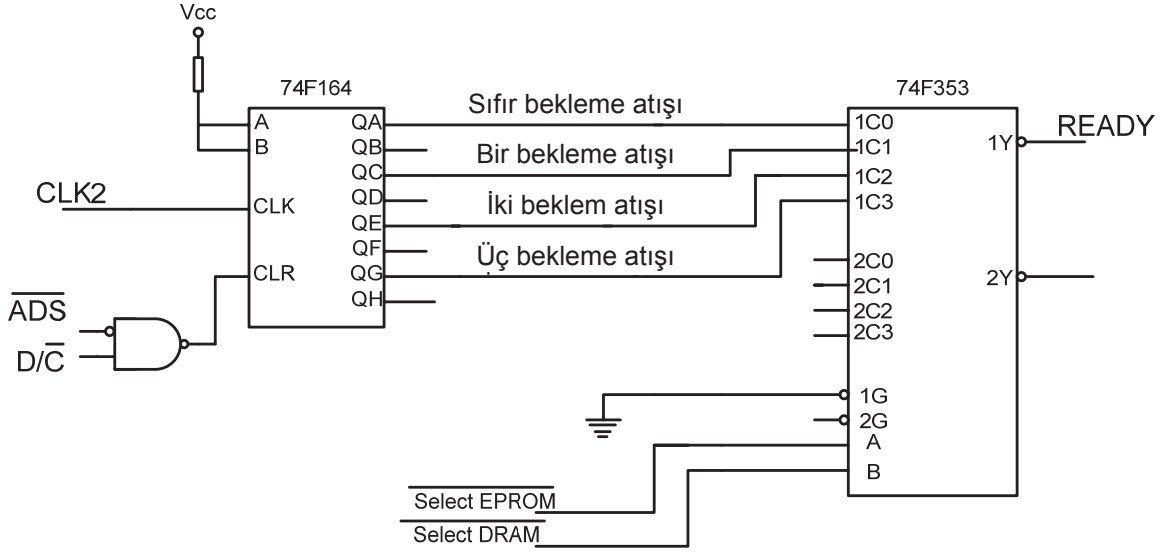
Giriş çıkış sistemine gelince, 8086 mikroişlemcinin giriş çıkış sistemiyle aynıdır.80386 mikroişlemci, giriş çıkış aygıların adreslenmesi için amaçlı 64KB bellek alanına sahiptir. Giriş çıkış sistemi bellek açısından izole edilebilir ve veri aktarımı için IN ve OUT yönergeleri kullanlıyor. Giriş çıkış birimlerin adreslenmesi için A15'ten A2'ye kadar adres hatları kullanılıyor ve BE3 - BE0 kontrol sinyalleri veriden bir bayt, 16 bit ya da 32 bitin seçilmesi için kullanılıyor.

Resim 8.22.'de bellek ve dış aygıtlar için **kontrol sinyallerin üretimi için basit tümleşik devre** gösterilmiştir. Bu devrenin giriş sinyallerini mikroişlemci gönderiyor, çıkış sinyalleri ise belleğe ve dış aygıtlara aktarılıyor. Kontrol sinyallerin işaretlenmesi, 80286 mikroişlemcide veriyollar denetimcisinin kontrol sinyallerin işaretlenmesiyle aynıdır.



Resim 8.22. 80386, 80486 ve Pentium mikroişlemci için bellek ve giriş çıkış birimine konrol sinyallerini üreten devre

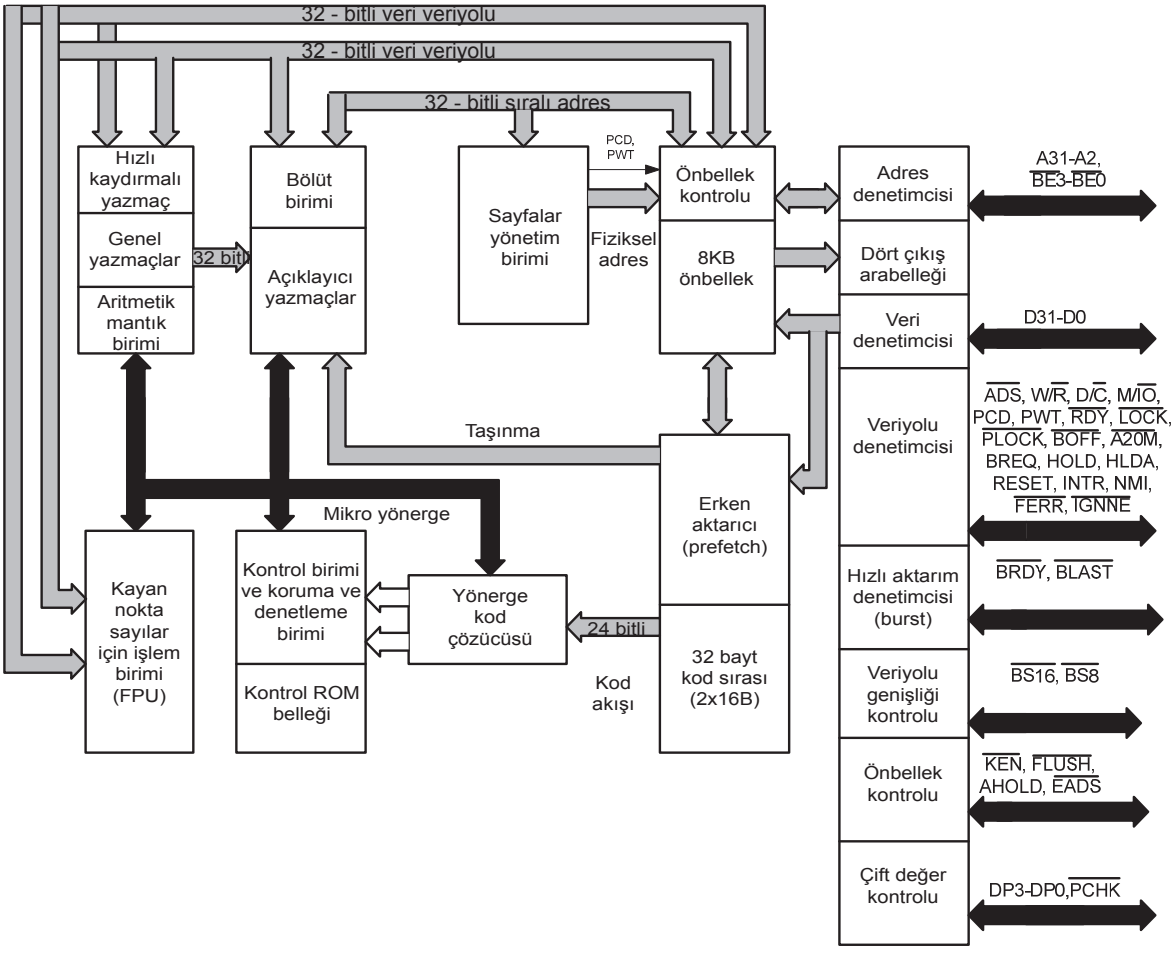
Bekleme durumları, belleğe erişim uzun sürdüğünden dolayı gereklidir. 80386 mikroişlemcide, 33MHz çalışma frekansında, erişim zamanı 46ns değerinde olmalıdır. DRAM'ın 60 ns erişim zamanı olduğundan dolayı bir bekleme durumun girmesi gerekiyor. EPROM belleğinin 100ns erişim zamanı var, buna göre iki bekleme durumun girmesi gerekiyor. READY sinyali mikroişlemcinin zamanlamasında, bekleme durumların sayısını kontrol ediyor. Resim 8.23.'te **READY sinyalın üretimi içim tümleşik devre** verilmiştir. 74164 devresi kaymalı yazmaçtır. Bu devre ADS sinyali alçak seviyede, D/C sinyali ise yüksek seviyede olunca sıfırlanıyor. Kaydırma, ADS sinyalin mantıksal bir olduğu zaman başlıyor ve devamda QA çıkışından başlayarak QK'ye kadar mentıksal birler meydana gelmeye başlıyor. Kaydırma yazmacın her dört çıkışı evirici çoklayıcıyla bağlıdır. READY sinyalın aktifleştirilmesi, bellek yongalarından (DRAM ya da EPROM) birinin seçimi için kullanılan sinyallere bağlıdır.



Resim 8.23. READY sinyalinin üretimi için tümleşik devre

8.11. 80486 Mikroişlemcinin Temel Özellikleri

80486 mikroişlemci yüksek tümleştirme seviyeli devredir ve 1,2 milyon transistörden oluşuyor. Mikroişlemcinin çalışma hızı farklı olabilir ve şu değerde olabilir: 25MHz, 33MHz, 50MHz, 66MHz ya da 100MHz. Bu mikroişlemcinin iki türü var: 80486DX ve 80486 SX. Bu iki mikroişlemci arasında fark, 80486SX mikroişlemcisinin yapısında sayısal yardımcı işlemcinin olmasıdır. Resim 8.24.'te 80486 mikroişlemcinin yapısı gösterilmiştir. 80486'nın yapısı şunları içeriyor: bellek yönetim birimi, 8KB birinci seviye önbelleği, aritmetik mantık birimi, **sayısal yardımcı işlemci** ve konrol birimleri.



Resim 8.24. 80486 mikroişlemcinin yapısı

Bellek yönetim birimi MMU (Memory Menagement Unit) iki bölümden oluşuyor: bölüt birimi ve sayfalarla çalışma birimi. Bölüt birimin içeriğinde, program bölütlerin büyüklüğü ve adresi hakkında bilgi saklayan bölüt açıklayıcılar bulunuyor.

Önbellek, bölüt birimini ve yönerge kod çözücüsünü bilgilerle,,besliyor". Yönerge kod çözücüsü önünde, kodları çözümlenmeyen işlem kodlarını koruyan 32B'lık yönerge sırası bulunuyor.

İki etkili çalıştırma birimi var, arimetik mantık birimi ve sayısal yardımcı işlemci. Aritmetik mantık birimi tam sayılarla çalışmak için kullanılıyor, sayısal yardımcı işlemci kayan noktalı sayılarla (gerçek sayılarla) çalışmak için kullanılıyor. 80386 mikroişlemcide sayısal yardımcı işlemci mikroişlemcinin içinde bulunmuyor, ana kartında bulunuyor.

Resimin sağ tarafında, kontrol birimleri gösterilmiştir. Kontrol birimlerin işlevini, onların **kontrol sinyalleri** yardımıyla açıklayacağız:

- Adres denetimcisi adres hatlarını yönetiyor, A31'den A2'ye kadar (dört bellek bankasından konumun seçimi için) ve $\overline{BE3}$ - $\overline{BE0}$ sinyaller – 32 biti veriden bir, 2 ya da 4 baytın seçimi için. Çıkış arabelleği verileri, 32 veri pini D31 - D0 aracılığıyla, belleğe ya da dış aygıtların gönderiyor.
- Veriyollarla yönetim için kontrol sinyaleri en büyük sayıdadır. 80486 mikroişlemcide $\overline{ADS}$, W/$\overline{R}$, D/$\overline{C}$, M/$\overline{IO}$, $\overline{LOCK}$, HOLD, HLDA, RESET, INTR, NMI pinlerinin, 8086, 80286 ve 80386 mikroişlemcilerde olduğu gibi aynı anlamları var. $\overline{A20M}$ (Address bit 20 Mask) sinyalı, 000FFFFFH adresli yerden 00000000H adresli yere kadar belleğin ayrılmasına yol açıyor. A20M sinyalı, 80486 mikroişlemcinin 1MB bellekli gerçek düzenide, tıpkı 8086 mikroişlemci gibi çalışması gerektiğinde kullanılıyor. PCD (Page Cache Disable) ve PWT (Page Write Through) pinleri sayfa aktarımı sırasında kullanılıyorlar. $\overline{RDY}$ pinin, READY piniyle aynı anlamı var, yani bekleme durumların sayısını kontrol ediyor. $\overline{PLOCK}$ (Pseudo Lock) çıkış pinidir ve sayısal yardımcı işlemcinin 64 ya da 80 bitli veri aradığı zaman aktifleştiriliyor. $\overline{BOFF}$ (back off) giriş sinyalıdır ve bu sinyal yardımıyla mikroişlemcinin veriyolları yüksek empedans durumuna koyuluyorlar. $\overline{BREQ}$ (Buss Request) sinyalı, 80486 mikroişlemcinin sadece kendi iç veriyollarını kullandığı zaman aktifleştiriliyor. $\overline{FERR}$ (Floating Point Error) sayısal yardımcı işlemcinin hata bulduğu zaman aktif oluyor. Bulunan hataya önem verilmesse $\overline{IGNNE}$ (Ignore Numeric Error) pini aktifleştiriliyor.
- $\overline{BRDY}$ (Burst Ready) ve $\overline{BLAST}$ (Burst Last Output) sinyalleri, veriyolundan hızlı veri aktarımı sırasında kullanılıyor.
- Önbellekle çalışmak için $\overline{KEN}$, $\overline{FLASH}$, $\overline{AHOLD}$ ve $\overline{EADS}$ pinleri kullanılıyor. $\overline{KEN}$ (Cache Enable) verilen anda veriyolundan aktarılan verinin önbellekte yerleştirilmesi gerektiği zaman aktifleştiriliyor. $\overline{FLUSH}$ (Cache Flush) önbellek içeriğinin silinmesi için kullanılıyor. $\overline{AHOLD}$ (Address Hold) sinyaliyle, adres veriyolu yüksek empedans durumuna koyuluyor, diğer veriyollar ise ak-

tiftir. Bu sinyal veriyolların bazı başka hükümdarı önbelleğe erişim yapmak istediği zaman uygulanıyor. $\overline{EADS}$ (External Address Strobe) $\overline{AHOLD}$ sinyaliyle aynı zamanda aktifleştiriliyor ve önbelleğe dışardan erişim hakkında sinyal veriyor.

- Çift değer kontrolu için dört pin, DP3 - DP0 (Data Parity) pinleri kullanılıyor ve hata bulunursa $\overline{PCHK}$ (Parity Check) pini aktifleşiyor.

Sonuçlar:

8086 mikroişlemcisinin 1MB kapasiteli belleği var ve bellek yerlerini adreslemek için 20 - bitli adresler kullanıyor. 8086 mikroişlemci iki özel birim içeriyor. İşletim birimi (execution unit) yönergeleri çalıştırıyor, veriyolu arabirim birimi (bus interface unit) ise işlem kodlarının, işlenenlerin ve sonuçların aktarımını gerçekleştiriyor.

AX, BX, CX, DX yazmaçları 8086 işlemcinin genel yazmaçlarıdır. 8 - bitli verileri saklamak istersek, 16 - bitli yazmaçlar yarıya ayrılıyor. 0'dan 7'ye kadar daha değersiz bitler AL, BL, CL, DL (L - low, alçak) ile işaretleniyor, 8'den 15'e kadar daha değerli bitler ise AH, BH, CH DH (H - high, yüksek) olarak işaretleniyor.

Bölütleştirme, belleğin bölütler olarak adlandırılan parçalara ayrılmasını tanımlıyor. Dört bölüt türü var: veri, kod, yığıt ve ekstra bölüt. Bölüt ve bölüt yazmacı terimleri arasında fark var. Bölüt belleğin parçasıdır, bölüt yazmacı ise mikroişlemcide bulunuyor ve aranan bölütün başlangıcının bulunmasını yardım ediyor.

Gerçek çalışma modunda, fiziksel adresin oluşması için iki bölüm gerekiyor: bölütün başlangıç adresi ve kaydırma. Bölütün başlangıç adresine eklenen kaydırma bazı yazmaçta bulunabilir ya da 16 - bitli sayı olarak verilebilir. Çevirici yönergelerde kaydırma, değer olarak orta parantezler [ ] içinde veriliyor.

8086 mikroişlemcisi iki düzende çalışabilir: minimum ve maksimum. Düzenin seçimi MN/$\overline{MX}$ pinin yardımıyla yapılıyor. Minimum düzende mikroişlemci kontrol sinyalleri kendisi üretiyor. Maksimum düzende yönetim işlevini 8288 seri numaralı veriyolu denetimcisi gerçekleştiriyor.

8086'dan başlayarak 80286 modeline kadar şu adresleme şekilleri uygulanmaktadır: yazmaçlı adresleme, doğrudan, dolaysız, yazmaçlı - dolaylı, temel - dolaylı, yazmaçlı - göreceli ve temel göreceli endeksli adresleme. Dolaysız adresleme şeklinde kaydırma 16 - bitli sayıdır, yazmaçlı - dolaylı adresleme şeklinde ise kaydırma bazı genel ya da endeks yazmacında içeriktir.

MOV BX,1234H yönergesiyle 1234H verisi BX yazmacına aktarılıyor. MOV BX, [1234H] yönergesinde veri kaynağı veri bölütün başlangıcından 1234H mesafelik uzaklıkta bulunan bellek yeridir, verinin hedefi ise BX yazmacıdır.

Dizi, veri ya da estra bölütünden ardaşıl, yan yana bulunan bellek yerlerinde yerleşmiş olan veriler kümesidir. Veri bölütünde işlenmesi gereken veriler yerleşiyor, ekstra bölütte ise gerçekleşen işlemlerden elde edilen sonuçlar yerleşiyor. Dizilerle çalışma yönergeleri S (string) harfiyle bitiyor.

CMP (Compare) yönergesi iki yazmacın içeriklerini kıyaslamak için kullanılıyor ve bayrakların durumuna etkiliyor. Bu arada işlenelerin değerleri aynı kalıyor. CMP yönergesinden sonra genelde, JA (jump above) ve JB (jump below) yönergeleri gibi koşullu atlama yönergeler geliyor.

TEST yönergesi bir yazmaçtan bitlerin denetlenmesi için kullanılıyor. Örneğin, TEST AL, 01H yönergesiyle, en az ağırlıklı bit (sağ taraftan birinci bit) test ediliyor.

JG, JL, JGE, JLE, JE ve JNE yönergeleri ön işaretli sayılar için koşullu atlama yönergeleridir, JA, JB, JAE, JBE, JE ya da JNE yönergeleri ise ön işaretsiz sayılar için kullanılıyor. Sayılar 8 - bitli ise, o zaman ön işaretli bitler - 128 ile +127 değerleri arasında olabilir, ön işaretsiz sayılar ise 0 ile 255 arasında olabilir.

Bildirimler komutlardır, çevirici programı düzenleyen yönlendirmelerdir. DB, DW ve DD bildirimleriyle değişkenler sembolik isimler kazanıyor. Bu bildirimlerle, 8 ve 16 bitli veriler için veri bölütünde bellek alanın ayrılması gerçekleşiyor. EQU bildirimi etiketlerin yaratılması için kullanılıyor. SEGMENT bildirimi bölütün başlangıcını, ENDS ise bölütün sonunu işaretliyor. Alt programların başlangıcını ve sonunu işaretlemek için PROC ve ENDP bildirimleri kullanılıyor.

8255 programlanabilir bileşeni, 8086 mikroişlemcinin sekiz alandan oluşan yedi bölütlü görünüm birimiyle bağlanması sırasında kullanılıyor. Görünüm birimine veriler göndermek için çoğullama süreci kullanılıyor. Alanın seçimi B bağlantı noktasında bulunan sekiz pin aracılığıyla yapılıyor. B bağlantı noktasından sadece bir pin aktif olacaktır ve bu pin görünüm biriminden bir alan seçecektir. Seçilen alandan hangi bölütlerin yanacağı, A bağlantı noktasından pinlerin durumuna bağlıdır.

82655 programlanabilir bileşeni, 8086 mikroişlemcinin, dört satırda ve dört sütunda sıralanmış, 16 düğmeden (tuştan) oluşan klavyeyle bağlanması sırasında kullanılıyor. B bağlantı noktası çıkış bağlantı noktasıdır ve bir sütunun seçilmesi için kullanılıyor. A bağlantı noktası giriş bağlantı noktasıdır ve satırlardaki düğmelerin durumunu okumak için kullanılıyor.

8254 zamanlayıcının dört iç yazmacı var: sayaç 0, sayaç 1, sayaç 2 ve komut yazmacı. Programlama komut yazmacı aracılığıyla gerçekleşiyor. Komut yazmacının yardımıyla mod ve sayaç seçiliyor. Mod giriş sinyaline bağlı olarak, çıkış sinyalin şeklini tanımlıyor.

80286 mikroişlemci 16 MB fiziksel belleğe ve 1GB sanal belleğe sahiptir. Dört işlevsel birimden oluşuyor: işletim birimi, adres birimi, yönerge birimi ve veriyolu arabirim birimi. 80286 mikroişlemcisi iki yönerge sırası içeriyor, biri yönerge kod çözücüsü önünde ve diğeri kodları çözümlenmiş yönergelerle.

BUSY, ERROR, PEREQ ve PEACK pinleri, 80286 mikroişlemcinin matematik yardımcı işlemciyle iletişim kurmak için kullanılıyor.

80386 işlemcisi, 8086 mikroişlemcinin 32 - bitli verziyonudur. Veri ve adres veriyolu 32 - bitlidir. Bellek alanın büyüklüğü 4GB değerindedir. 80386 mikroişlemcinin belleği dört bellek bankasına ayrılıyor ve her bankanın 1GB kapasitesi var. 8, 16 ve 32 - bitli verilere dorğudan erişim olanağı vardır. $\overline{BE3}$ - $\overline{BE0}$ dört kontrol sinyalı seçilen konumda dört bayttan bir baytın seçilmesi için kullanılıyor.

80386 mikroişlemcide yazmaçlar birkaç gruba ayrılıyor: genel yazmaçlar, yönerge göstergesi, durum yazmaçları, bölüt yazmaçlar, bölüt açıklayıcı yazmaçları, sistem adres yazmaçları ve denetleme ve hata giderme yazmaçları.

80486 mikroişlemci şu bölümlerden oluşuyor: bellek yönetim birimi, 8KB birinci seviye önbellek, aritmetik mantık birimi, sayısal yardımcı işlemcisi ve kontrol birimleri.Bellek yönetim birimi MMU (Memory Menagement Unit) iki bölümden oluşuyor: bölüt birimi ve sayfalarla çalışma birimi. Aritmetik mantık birimi tam sayılarla çalışmak için kullanılıyor, sayısal yardımcı işlemcisi ise kayan noktalı sayılarla (gerçek sayılarla) çalışmak için kullanılıyor. 80486 mikroişlemcide sayısal yardımcı işlemcisi mikroişlemcinin içinde yerleşmiştir, 80386 işlemcide olduğu gibi anakatrında bulunmuyor.

Sorular ve Ödevler:

1. 8088 mikroişlemcisi ve 8086 mikroişlemcisi arasında temel fark nedir?

2. 8086 işlemcisi minimum ve maksimum çalışma düzeninde çalışabilir. Açıkla!

3. 8086 mikroişlemcide işletim birimin ve veriyolu arabirim biriminin işlevleri nedir?

4. Veriyolu arabirim biriminde hangi yönergeler yönerge sırasının sıfırlandırılmasına yol açıyor?

5. 8086 mikroişlemcide hangi pin minimum ya da maksimum çalışma düzenin seçimi için kullanılıyor?

6. 8086 mikroişlemcide DT/$\overline{R}$ pinin nasıl işlevi vardır?

7. 8086 mikroişlemcide, minimum ve maksimum çalışma düzenlerinde durum bitlerinin anlamlarını açıkla!

8. 8086 mikroişlemcide sekizbitli ve onaltıbitli genel yazmaçların nasıl işaretlendiğini açıkla!

9. 8086 mikroişlemcide endeks yazmaçların hangi işlevleri var? Endeks yazmaçlarını say!

10. Bölütler nedir? Bölüt yazmaçları ne için kullanılıyor?

11. 8086 mikroişlemcinin gerçek çalışma modunda, bellek yerin fiziksel adresini hesaplamak için uygulanan süreci açıkla!

12. Verilen bölüt yazmacı - kaydırma kombinasyonları için bellek yerlerin adreslerini hesapla

    a) DS = 3400H BX = 2000H
    b) CS = 4900H IP = 1002H
    c) SS = 6500H SP = 3200H
    ç) ES = 7600H DI = 3A00H

13. 8086 mikroişlemcide uygulanan adresleme şekillerini say!

14. MOV BX, [1234] yönergesinde, orta parantezler içinde bulunun değer bize neyi gösteriyor?

15. C8H verinin CL yazmacına girmesini gerçekleştiren yönergeyi yaz.

16. Aşağıdaki tabloyu doldur.

Yönerge	Adresleme şekli	Verinin büyüklüğü	Verinin kaynağı	Verinin hedefi
MOV AL,[BP+DI]				
MOV CX,[DI]				
MOV [1000H],DL				
MOV AL,35				
MOV DL,[SP+2345]				
MOV [SI+1000H],AX				

17. 8086 mikroişlemcide, durum yazmaçlarında D bayrağının ne için kullanıldığını açıkla!

18. Verilen yönergelerin yorumlarını yaz:
    - RET
    - MOVSW
    - IN AL,DX
    - JE TOPLAM
    - POP [BX]
    - MOV CX,[SP]

19. TEST BL,AL yönergesi hatalıdır. Nedenini açıkla.

20. CL=98H,DL=6AH. Aşağıdaki yönergelerin çalıştırılmasından sonra yazmaçların içeriklerini hesapla:

```
AND CH,DL
SAR DL,2
```

21. AX=ABCDH,BX=2288H. Aşağıdaki yönergelerin çalıştırılmasından sonra yazmaçların içeriklerini hesapla:

```
XOR BH,79H
ROL AL,2
SUB AX,BX
```

22. Aşağıdaki örneklerde koşullu atlama yönergesi çalıştırılacak mi? Açıkla neden.

A)
```
MOV CL,80H
ADD CL,01H
JO TEKRARKA
```

B)
```
MOV CX,7589H
XOR CL,76H
NOT CL
```

C)
```
MOV AL,7FH
SUB AL,8AH
JC AJDE
```

23. CMP v TEST yönergeleri arasında fark nedir?

24. OUT 0002H, AX yönergesini açıkla.

25. 8086 mikroişlemcide, mikroişlemcinin bazı giriş aygıtından aldığı veriyi korumak için hangi yazmaç kullanılıyor?

26. 8255 arabirim bileşende iç yazmaçların adreslenmesinin nasıl gerçekleştiğini açıkla!

27. 8255 arabirim bileşeni, 8086 mikroişlemcinin ve yedi bölütlü görünüm biriminin bağlanması sırasında kullanılıyor. A ve B bağlantı noktalarının nasıl işlevleri olduğunu açıkla

28. Yedi bölütlü görünüm biriminde, PA=x1111001B ve PB= 10111111B ise, elde edilen görsel etkisi nasıl olacak?

29. 8255 arabirim bileşenin kontrol yazmacının 011H arabirim adresi var. A ve B bağlantı noktalarını o şekilde programla ki, yedibölütlü görünüm biriminin sağ taraftaki alan, 5 sayısıyla ışıklansın!

30. 8255 arabirim bileşeni ve klavyenin bağlanması sırasında, A ve B bağlantı noktalarındaki pinlerin nasıl işlevleri vardır?

31. Resim 8.12.'de gösterilmiş klavyede, düğmelerin iki ardaşıl çekimi arasında, 1ms gecikme zamanın neden girilmesi gerekiyor?

32. 8254 programlanabilir zamanlayıcı kaç sayaç içeriyor? Her sayaçtaki her üç pinin işlevlerini açıkla!

33. 8254 zamanlayıcının kontrol yazmacı ne için kullanılıyor?

34. 8254 zamanlayıcının giriş frekansı 8MHz'tir. 8254 zamanlayıcının çıkışında, 200KHz frekanslı sürekli tetikleme dizisi elde etmek için, sayaç ve kontrol yazmacı hangi değerlere programlanmaları gerekiyor?

35. 80286 mikroişlemciyi oluşturan birimleri say ve onların işlevlerini açıkla!

36. 82288 kontrol sinyaller üreticisinin hangi sinyalleri giriş, hangi sinyaller ise çıkış sinyalleridir?

37. 82884 dijit pals üreticisinde F/C ve EFI pinlerinin işlevlerini açıkla!

38. 80286, 80386 v3 80486 mikroişlemcilerde, adres ve veri veriyolun genişlikeri nekadardır?

39. 80386 - Pentium mikroişlemcilerde bellek bankanın seçimi için hangi pinler kullanılıyor?

40. 80386 mikroişlemcide genel yazmaçlar nasıl işaretleniyor?

41. 80386 mikroişlemcide, bölüt açıklayıcıları ve kontrol yazmaçlar ne için kullanılıyor?

42. İç izinli bellek kavramını açıkla!

43. 80286 ve 80386 mikroişlemcilerde, dijit pals üreticilerin tümleşik devreleri ve bellek ve dış aygıtlar için kontrol sinyalleri üreticileri arasında kıyaslama yap!

44. 80386 mikroişlemide READY sinyalın nasıl oluştuğunu açıkla!

45. Hangi mikroişlemci kendisinde sayısal yardımcı işlemcisini ilk olarak içeriyormuş? Sayısal yardımcı işlemci hangi yönergeleri çalıştırıyor?

46. 80486 mikroişlemcide, önbelleğin kontrolu için kullanılan pinleri say! Onların kullanımını açıkla!

9. Pentium Mikroişlemci

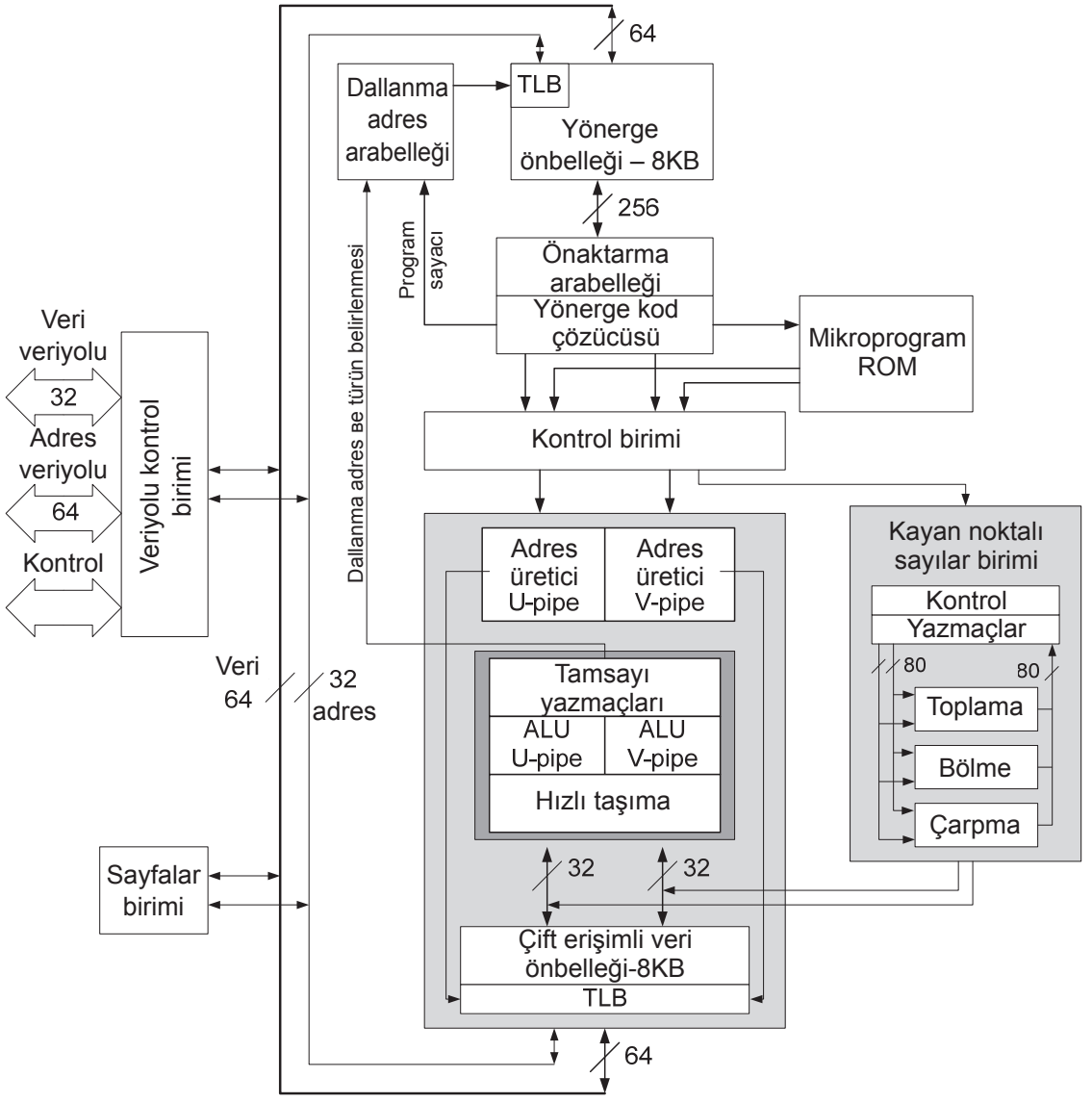
9.1. Pentium 1 Mikroişlemcisi

Pentium mikroişlemcide, 80486 mikroişlemciye kıyasen birçok açıdan ilerlemeler var: ilerlemiş önbellek, daha geniş 64 - bitli veri veriyolu, daha hızlı sayısal yardımcı işlemci, tam sayılar için çift mikroişlemci ve dallanmayı öngörme mantığı. Pentium mikroişlemcinin yapısı resim 9.1.'de gösterilmiştir.

Birinci seviyeden **önbellek** L1, 8KB'lık iki önbelleğe ayrılıyor. Birinci önbellek yönerge önbelleği, diğeri ise veri önbelleğidir. 80486 mikroişlemciden 32 - bitli veri veriyolu, 64 - bitliyle değişmiştir. **Sayısal yardımcı** işlemci kayan noktalı sayıların işletilmesi için kullanılıyor ve 80486 sayısal yardımcı işlemciden beş kat daha hızlıdır. **Tam sayılar** için **çift mikroişlemci** bir dijit atışta iki yönerge çalıştırabilir. Atlama yönergesi sıraya gelince, mikroişlemci atlama yönergesinde verilen adresten sıradaki yönergelerin erken aktarımını başlatacak. Çalıştırılması gereken yönergeler, yönerge önbellekte korunuyor ve atlama yönergesi sıraya gelince, bir dijit palsında çalıştırılacak. Atlama koşulu geçerli değilse, o zaman atlama yönergesi için üç dijit palsı daha gerekecek. Şansına, öngörüler genelde doğrudur.

Pentium mikroişlemcinin **Süpersayısal yapısı**, onun üç işlem biriminin olmasından geliyor: bir birim kayan noktalı sayılar için ve iki birim tam sayılar için (U pipe, V pipe). Buna göre paralel olarak üç farklı yönerge çalıştırılabilir.

Pentium mikroişlemcinin belleği 4 GB büyüklüğündedir ve 512 MB'lık 8 bellek bankasında ayrılıyor. Bellek yeri, bir baytlık veri ve bir çift değer kontrol biti saklayabilir. Hafıza edilen baytta, birlerin sayısı çift değerse, o zaman çift değer biti bir olacak, birlerin sayısı tek değerse, o zaman çift değer biti sıfır olacak.



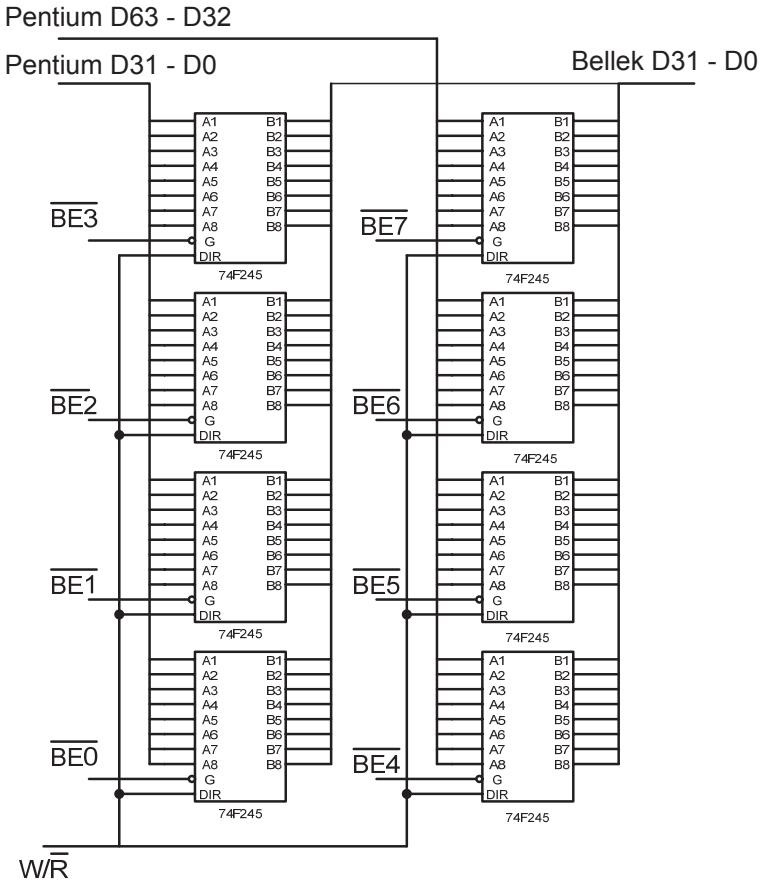
Resim 9.1. Pentium 1 miroişlemcinin yapısı

Bankaların seçimi BE7 - BE0 (bank enable) kontrol sinyallerin aracılığıyla yapılıyor. Bellek bankasına özel erişim 8, 16, 32 ve 64 - bitli verilere erişim sağlıyor. Örneğin, mikroişlemci bellekten 64 - bitli veri okumak isterse, o zaman tüm sekiz bankanın seçilmesi gerekiyor, yani tüm sekiz $\overline{BE7}$ - $\overline{BE0}$ pinleri aktif durumda olmalıdır. 64 - bitli veriden sadece birinci baytın okunması gerekirse, o zaman sadece BE0 pini aktif olacaktır. Kayan noktalı sayıların işletilmesi sırasında, 64 - bitlik bellek genişliği ve 64 - bitli veri veriyolu çok önemlidir. Kayan noktalı sayıların büyüklüğü 64 bit olduğu için, böyle bir sayının okunması için bir döngü yeterlidir.

80486 mikroişlemcide aynı işlem için iki döngü gerekiyordu. Verilerin çift değer kontrolu dışına, Pentium 1 mikroişlemcide, her işlem için adres bitleri (A31 - A0) üze-

rine de çift değer kontrolü yapılıyor. Bu amaçla AP (Address parity) pini kullanılıyor ve bu sırada hata meydana gelirse, APCHK (address parity check) pini aktifleştiriliyor. Hata durumunda sistem gereken etkinliği gerçekleştiriyor, yani kesinti meydana geliyor.

Belleğin veri veriyolu 32 - bitlidir, Pentium mikroişlemcinin veriyolu ise 64 - bitlidir. Bu **iki veriyolun bağlanması için,** 64 - bitli verileri 32 - bitli verilere ve 32 - bitli verileri 64 - bitli verilere dönüştüren çoklayıcı kullanılıyor. Resim 9.2.'de sekiz iki yönlü arabellekten oluşan böyle bir sistem verilmiştir.



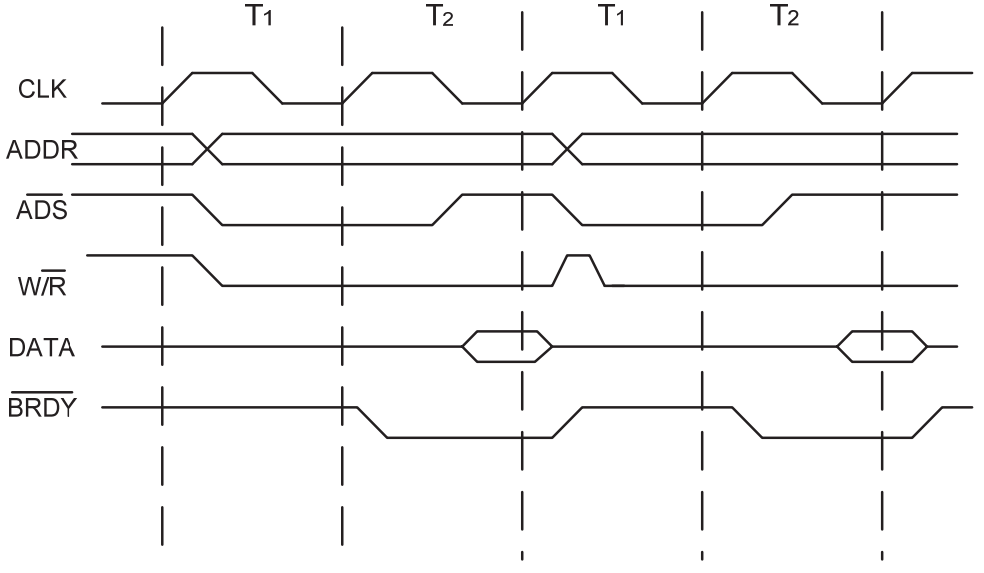
Resim 9.2. Pentium mikroişlemcinin 64 - bitli veri veriyolunun, belleğin 32 - bitli veri veriyoluyla bağlanması

Aktarma yönünü W/R sinyalı belirliyor. W/$\overline{R}$ sinyalı DIR (direction) pinlerin tüm sekiz arabelleğe gönderiliyor. W/$\overline{R}$=0 olduğu zaman okuma işlemi gerçekleşiyor ve veriler bellekten mikroişlemciye doğru hareket ediyor. W/R=1 olduğu zaman, yazma işlemi gerçekleşiyor ve bu durumda veriler mikroişlemciden belleğe doğru hareket ediyor. Bellek bankaların seçimi için kullanılan sinyaller (BE0'dan BE7'ye kadar), arabelleklerin etkinleştirilmesi için kullanılıyorlar.

Bu sinyaller G(Gate) arabellek seçim pinlerine kadar gönderiliyor. 64 bitli veriden dört daha değerli baytın $\overline{D63}$ - $\overline{D32}$ aktarılması gerekirse, o zaman $BE4$'ten $BE0$'a kadar sinyaller aktiftir, daha değersiz dört baytın $\overline{D31}$ - $\overline{D0}$ aktarılması gerekirse, o zaman $\overline{BE0}$'dan $\overline{BE3}$'e kadar hatlar aktif durumdadır. Bank Enable sinyalleri nasıl aktifleştirilirse, öyle veriler uygun bellek bankalarında yazılıyor.

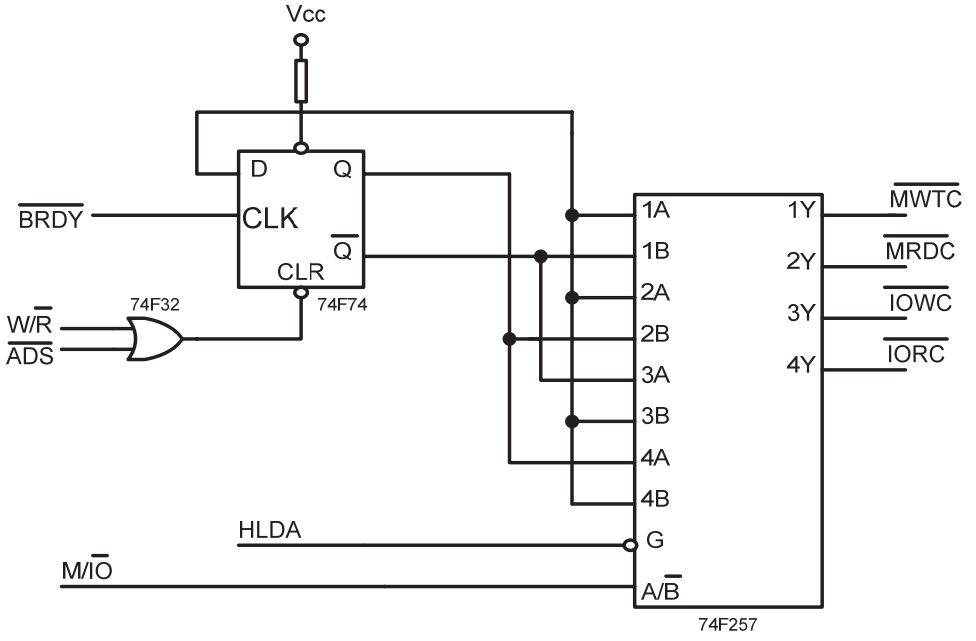
Bellekten okumak için çok daha etkili bir yöntem var, o da **akış (burst)** döngüsü aracılığıyla gerçekleşiyor. İngilizce burst sözcüğü, hızlı akış, şiddetli hava akımı,sel, püskürme anlamına geliyor. Bir akış döngüsünde, beş dijit palsı süresi içinde dört 64 - bitli sayı aktarılabilir. Bekleme palsı olmadan, bu döngüyle bellekten aktarma 15,2ns sürüyor. Bu yöntem özellikle ikinci seviye önbellekten okunduğu zaman uygundur.

Pentium mikroişlemcilerde **makine döngüsü** iki dijit pals kadar sürüyor. Çalışma frekansı 66MHz'tir, buna göre saniyede 33 milyon bellek aktarımı gerçekleşiyor. Resim 9.3.'te iki makine döngüsü gösterilmiştir. Birinci döngü bellekten okumak için (W/$\overline{R}$=0), ikinci döngü ise belleğe yazmak için (W/$\overline{R}$=1) kullanılıyor. Mikroişlemcinin her adres gönderdiğinde $\overline{ADS}$ (Address Data Strobe) pini aktifleştiriliyor. Bu sinyal olmadan, adres geçerli olmayacak. $\overline{BRDY}$ (Burst Ready) sinyalı mikroişlemci için giriş sinyalidir. İki makine döngüsünden birinci dijit palsın sonunda, $\overline{BRDY}$ sinyalı bire eşittir ve $\overline{BRDY}$ sinyalın bu durumu, belleğin veri aktarımı için henüz hazır olmadığı göstergesidir ve otomatik olarak bütün bir bekleme dijit palsı ekleniyor. Bekleme durumunun olmadığı için, $\overline{BRDY}$ sinyalın alçak seviyede olması gerekiyor. Böyle durum makine döngüleri ikinci dijit palsın sonunda meydana geliyor. İkinci atışın tamamlandıktan sonra mevcut döngü tamamlanıyor ve yeni döngü başlayabilir.



Resim 9.3.Pentium 1 mikroişlemcide okuma ve yazma için makine döngüsünün zamanlama diyagramı

W/$\overline{R}$, $\overline{ADS}$ ve M/$\overline{IO}$ kontrol sinayalleri, bellek ve dış aygıtlar için **kontrol sinyallerin üretimi** için kullanılıyor. Bu amaçla palslı flip flop ve sekiz girişli ve dört çıkışlı çoklayıcı kullanılıyor. Bu durum resim 9.4.'te gösterilmiştir.



Resim 9.4. Bellek ve dış aygıtlar için kontrol sinyalelrin üretimi

$\overline{BRDY}$ (Burst Ready) pini bellek ya da dış aygıtın, veri veriyolundan veri alırken ya da verirken aktifleştiriliyor. Okuma döngüsünün tamamlanması için ikinci dijit palsın sonunda $\overline{BRDY}$ sinyalı mantıksal sıfıra eşit olması gerekiyor. Aksi taktirde bekleme palsın eklenmesi gerekecek. $\overline{BRDY}$ sinyalı sıfıra eşit değilse en çok dört bekleme palsı eklenebilir.

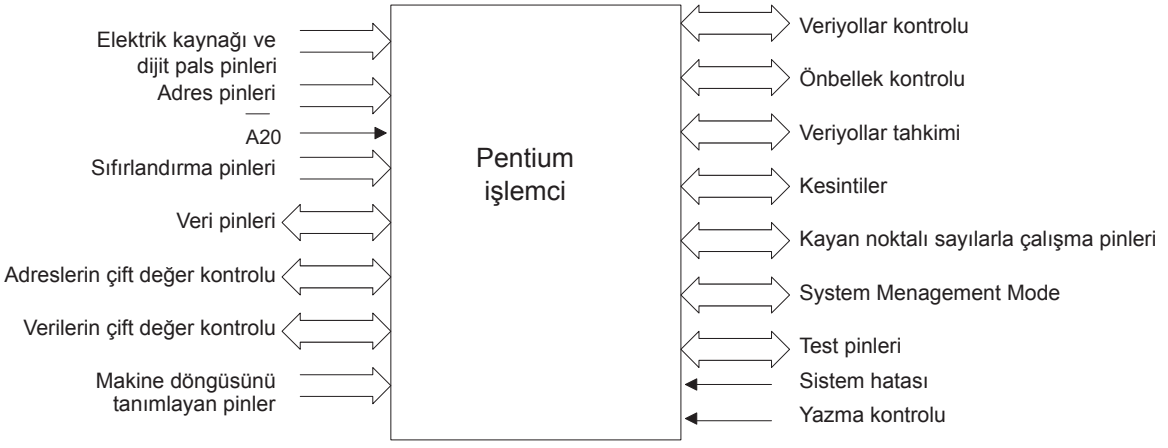
9.2. Pentium 1 Mikroişelmcinin Pin Diyagramı

Pentium 1 mikroişlemcinin 237 pini vardır. Pin diyagramın gösterildiği resim 9.5'te, pinler işlevlerine bağlı olarak pinler gruplara ayrılmıştır. Pentium 1 mikroişlemci 5V'luk elektrik kaynağı ve 3,3A'lik elektrik ceryan kuvveti kullanılıyor. Çalışma palsın frekansı 66MHz'tir.

Bellek yerin seçimi için **29 adres** pini vardır (A31 - A3) ve bellek bankanın seçimi için sekiz pin vardır $\overline{B57}$ - $\overline{B50}$.

$\overline{A20}$ pini, mikroişlemcinin gerçek çalışma modunda çalışması gerektiğinde aktifleştiriliyor. Mikroişlemcinin gerçek modunda çalışması, daha eski nesil mikroişlemcilerle uyum sağlanması için gereklidir.

İki **sıfırlandırma pini** var. RESET pini aktif olunca, tüm programlar kesiliyor ve mikroişlemci FFFFFFF0H adresli yerinden yazılımı çalıştırmaya başlıyor. INIT (initialization) giriş pini de sıfırlandırma tanımlıyor, ancak önbellekler, geri yazma arabellekler ve kayan noktalı sayılar yazmacına etkilemiyor.v



Resim 9.5. Pentium mikroişlemcinin pin diyagamı

Adreslerin çift değer kontrolu için iki pin, AP ve $\overline{APCHK}$ pinleri kullanılıyor. AP (address parity) pinin arcılığıyla çift değer biti gönderiliyor. Gönderilen adreste, çift sayıda birler varsa, o zaman bu bitin değeri birdir, aksi takdirde sıfırdır. Çift değerinde hata meydana gelirse, o zaman $\overline{APCHK}$ (address parity check) çıkış pini aktifleştiriliyor.

Çift değer kontrolu, bellekten ya da dış aygıttan okunan **veri bitlerine** de yapılıyor. Her bellek bankası için birer çift değer kontrol pini DP7 - DP0 (data parity) vardır. Çift değer kontrolünde hata meydana gelirse, o zaman $\overline{PCHK}$ (parity check) çıkış pini aktifleştiriliyor. $\overline{PEN}$ (parity enable) pini aracılığıyla mikroişlemci meydana gelen hatanın gidirelmesi için kesinti programı aktifleştiriliyor.

Makine döngü türünün tanımlanması için M/$\overline{IO}$ (memory/$\overline{input\ output}$), D/$\overline{C}$ (data/$\overline{control}$), W/$\overline{R}$ (write/$\overline{read}$), $\overline{CASHE}$, $\overline{LOCK}$ pinleri kullanılıyor. M/$\overline{IO}$ piniyle mikroişlemci iletişim aracı seçiyor, W/$\overline{R}$ piniyle işlem seçiliyor. D/$\overline{C}$ pini bire eşit olunca, o zaman bellekten ya da dış aygıtlardan veriler aktarılıyor, D/$\overline{C}$ pini sıfıra eşit olunca, o zaman kesinti işleten makine döngüsü çalışıyor. $\overline{CACHE}$ pini aktif olduğu zaman, mikroişlemci önbellekten veriler okuyor. $\overline{LOCK}$ pini en çok bellekten doğrudan aktarma (DMA kavramı) sırasında kullanılıyor.

Veriyolu kontrol pinleri $\overline{NA}$ (next address), $\overline{ADS}$ (address data strobe), $BRDY$ (burst ready) pinleridir. NA pini, dış bellek sıradaki makine döngüsü için hazır olduğu zaman aktif oluyor. $\overline{ADS}$ adres veriyolun etkinleştirilmesi için kullanılıyor ve bu pin mikroişlemci belleğe ya da dış aygıta adres gönderdiği her zaman aktif durumdadır. $\overline{BRDY}$ pini, belleğin verilen yere ulaşması için fazla zaman gerekirse, o zaman bekleme palsların eklenmesi için kullanlıyor.

Önbelleğin kontrolu için özel pinler vardır. $\overline{KEN}$ (cache enable) pini iç önbelleği etkin hale getiritiyor. WB/$\overline{WT}$ (writeback/write - through) pini önbellek için işlem seçiyor. WB/WT pini bir olduğu zaman, önbellek mikroişlemciden veri alıyor, bu pin sıfır olduğu zaman ise bellekten veriler alıyor. $\overline{FLUSH}$ pinin aktifleştirilmesiyle iç önbellek siliniyor.

Veriyollar tahkimi,verilerin aktarımı için veriyoları kullanacak aygıtın seçilmesini tanımlıyor. BREQ (bus request) pini mikroişlemcinin veriyollar kontrolüne isteme gönderdiği zaman aktifleştiriliyor. Veriyollarını doğrudan veri aktarımı (DMA) için bazı hızlı dış aygıt da kullanabilir ve böyle durumda HOLD ve HLDA pinleri aktif duruma getiriliyorlar.

Sayısal yardımcı işlemci kayan noktalı sayılar adıyla da bilinen gerçek sayıları işletiyor. $\overline{FERR}$ (floating - point error) pini, sayısal yardımcı işlemcinin çalışmasında hata meydana geldiği zaman aktifleştiriliyor. Ancak $\overline{IGNNE}$ (ignore numeric error) pini aktif durumda bulunuyorsa, o zaman bu hatalara önem verilmiyor.

Pentium 1 işlemcisi iki **özel düzende (modunda)** çalışabilir. $\overline{SMI}$ (system management interrupt) giriş pini aktif olduğu zaman, mikroişlemci sistem bellek yönetimi modunda çalışıyor. TMS (test mode select) pini aktif ise, o zaman mikroişlemci test modunda çalışacak.

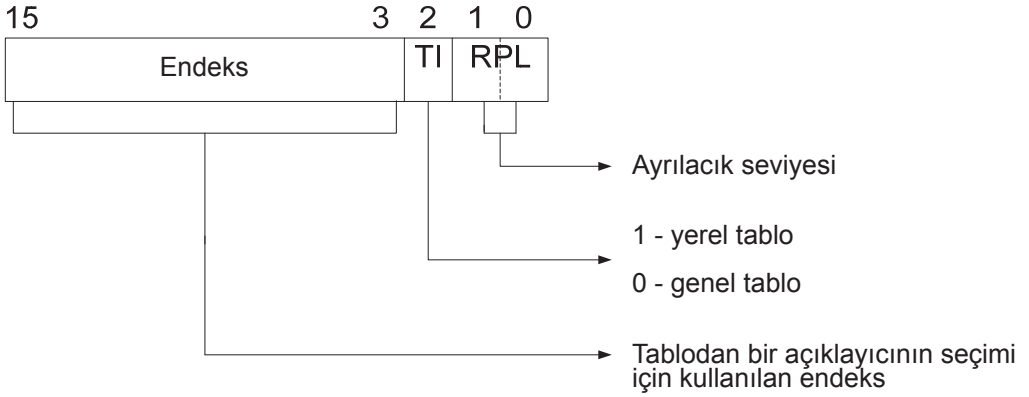
9.3. Pentium Bellek Yönetimi

Pentium mikroişlemcileri üç modda çalışabiliyor: gerçek modu, koruma modu ve sanal çalışma modu. Gerçek çalışma modunu 8086 mikroişlemciyi incelerken tanımıştık. Gerçek mod en basit çalışma modudur ve mikroişlemciye ilk 1MB bellek alanına kadar erişim sağlıyor. **Koruma modunda** adres alanı **4GB**'a (2^{32}) kadar genişliyor, **sanallaştırmayla** ise çok büyük, **64TB** (2^{46}) adres alanı elde ediyoruz.

Koruma çalışma modunda, bölüt yazmacı bölütün başlangıç adresini içermiyor. Onun yerine seçici (selektör) içeriyor. Seçici 8192 açıklayıcıdan birinin seçilmesine olanak veriyor. Her programın kendine ait yerel açıklayıcı tablosu var, ancak bilgisayarda tüm programları ayıran bir benzersiz genel açıklayıcı tablosu da var. Yerel tab-

lo bir programın kullandığı bölütleri açıklıyor (kod, veri, yığıt bölütü), genel tablo ise işletim sistemin kullandığı bölütler dahil, tüm sistem bölütlerini açıklıyor.

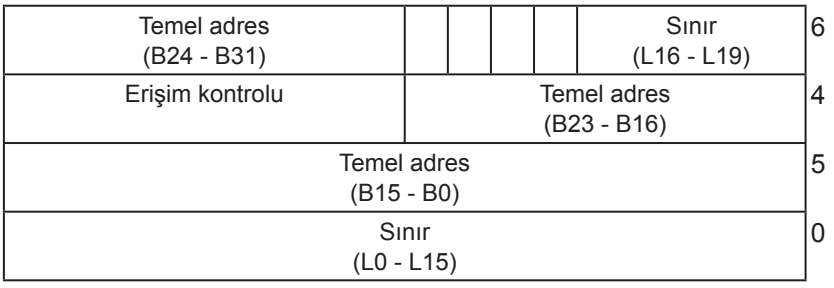
Bir bölüte ulaşmak için o bölütün seçicisi bölüt yazmaçlarından birinde yerleştirilmelidir. Her seçici 16 bitten oluşuyor. Seçici, açıklayıcıyı daha kolay bulabilecek şekilde oluşmuştur. Resim 9.6.'de seçici bitlerinin anlamı verilmiştir.



Resim 9.6.Seçici bitlerinin işlevsel açıklaması

Önce, seçicide bit 2'nin değerine bağlı olarak iki tablodan (GTD ve LTD) biri seçiliyor. Seçicideki endeks yardımıyla bir açıklaycının yeri belirleniyor. Endeks bir açıklayıcının, uygun açıklayıcılar tablosunun başlangıcından (dibinden) nekadar uzak olduğunu gösteriyor.

Açıklayıcının aracılığıyla bellekte aranan bölüt bulunuyor. Resim 9.7.'de bir açıklayıcı verilmiştir. Açıklayıcı 8 bitten oluşuyor.



Resim 9.7. Bölütte açıklayıcı bitlerinin işlevsel açıklaması

Temel adresi, aslında, bölütün başlangıç adresini tanımlıyor. Sınır (limit) bölütün son adresini bulmak için kullanılıyor. Bölütün büyüklüğü G bitine de bağlıdır. G (granularity) biti sıfırsa, o zaman sınır baytlarla ifade ediliyor ve bölütün en yüksek büyüklüğü 2^{20}=1MB'tır (20 - sınır için öngörülen bit sayısı). Eğer G=1 ise, o zaman sınır sayfalarla ifade ediliyor. 80386 mikroişlemcide sayfalar hiçbir zaman 4KB=2^{12}'den daha küçük değildir. Buna göre sınırdaki 20 bitle bölüt için toplam 2^{32}=$2^{20}2^{12}$ = 4GB büyüklüğü elde ediliyor.

Örnek 9.1:Bölütün temel adresi 1000 000H'dir ve sınır 001FFH'dir. Son yerin adresi yani bölütün sonu hesaplansın.

A) G=0

B) G=1

Çözüm:

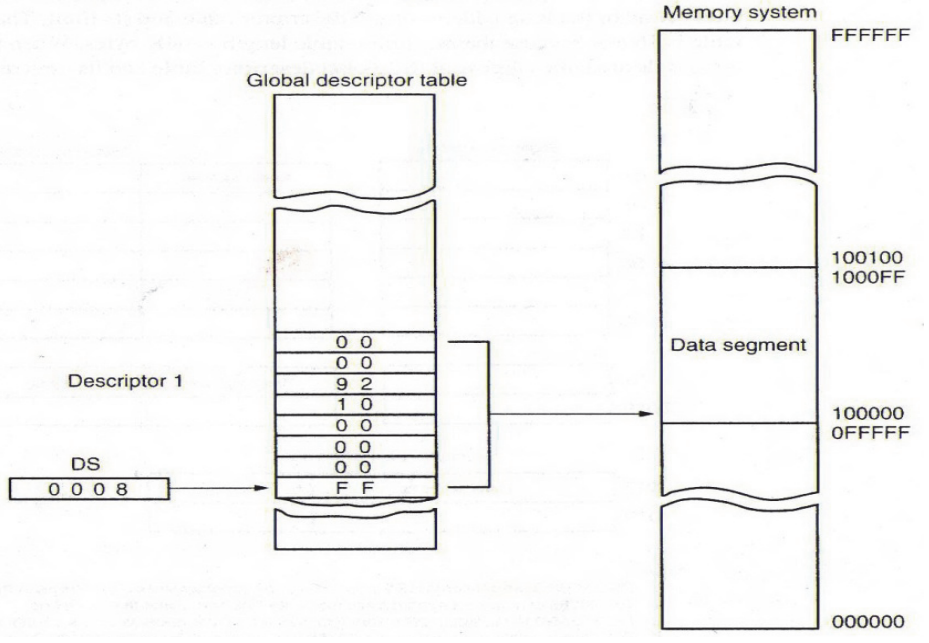
A) Eğer G=0 ise, o zaman sınır baytlarla ifade ediliyor ve bölütün sonu şu şekilde elde ediliyor:

Sonun adresi=başlangıç adresi + sınır =1000 000H+001FFH=1000 0 1FFH

B) Eğer G=1 ise, o zaman sınır sayfalarla ifade ediliyor. Bölüt 1000 000H adresiyle başladığı için demek ki bölütün ilk sayfası 1000 000H adresinden 1000 0FFFH adresine kadar yayılıyor. İkinci sayfa 1000 1000H'den 1000 1FFFH'ye kadar yayılıyor, üçüncü sayfa 1000 2000H'den 1000 2FFFH'ye kadar. Bu 01FF kez tekrarlanmalıdır. Son sayfa 101FF000H'den 101FFFFFH'ye kadar yayılıyor. Son sayfanın sonu bölütün sonuna uyduğu için, bölüt sonunun adresi 101FFFFFH olacak.

Erişim kontrol baytı bölütün bir sistem çerçevesinde nasıl çalıştığını açıklıyor. Örneğin, veri bölütü söz konusu olursa, verilerin değişebileceği ya da korunabileceğini vurgulayabilir. Ayrıca, bölüt kendi sınırını aşarsa, program kesilecek (TRAP kesintisi meydana gelecek)

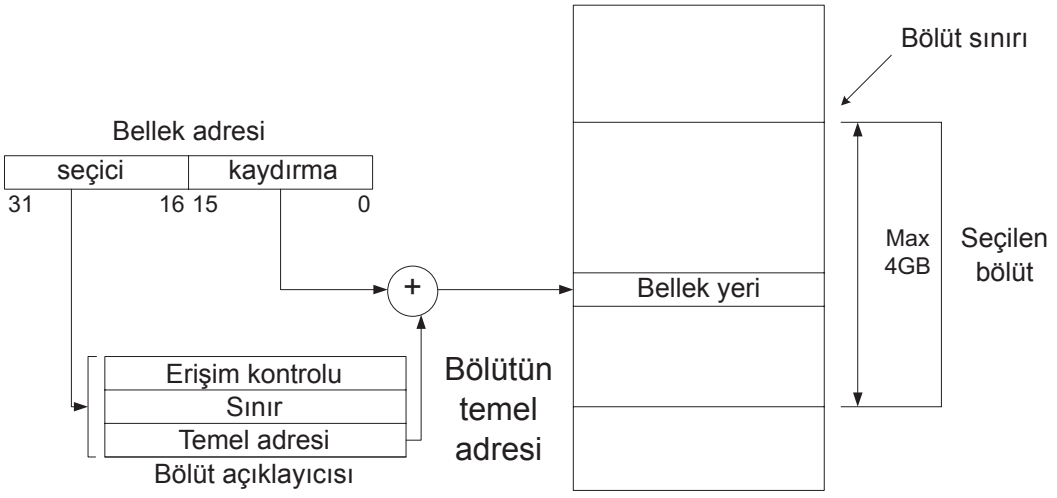
Resim 9.8.'de bölüt yazmacının değeri 0008H=0000000000001000B olan seçici içeridiği zaman bellekten **aranan** veri **bölütün bulunmasıyla** ilgili örnek verilmiştir.



Resim 9.8. Pentium mikroişlemcide belleğin veri bölütüne erişim

Resim 9.6.'ya göre 3 ile 15 arası bitler genel açıklayıcılar tablosundan birinci açıklayıcıya gösteriyor (bit 2 sıfırdır). Resim 9.7.'de açıklayıcının oluşmuş olduğu sekiz bayt verilmiştir ve onlar 0'dan 7'ye kadar numaralandırılmıştır. Açıklayıcının ikinci, üçüncü, dördüncü ve yedinci baytı, aslında, temel adresidir, yani bölütün başlangıç adresini veriyorlar. Resim 9.8.'den açıklayıcının bu baytları 10000000 değerini verdiklerini görebiliyoruz.

Aranan bölüt bulunduktan sonra, istenen **verinin bulunmasına** geçiliyor. 80386 32 - bitli adres kullanılıyor. Adresten daha değerli onaltı bit seçicinin değerini veriyor, daha değersiz on altı bit kaydırmanın değerini veriyor. 80386 mikroişlemcide kaydırmanın anlamı, 8086 mikroişlemcide kaydırmanın anlamıyla aynıdır. Kaydırma bölüt başlangıcından istenen bellek yerine kadar mesafeyi veriyor. Bellek yerine erişim resim 9.9.'da gösterilmiştir.



Resim 9.9. Seçilen bölütten bellek yerine erişim

Sonunda şu **sonuca** varabiliriz**:**

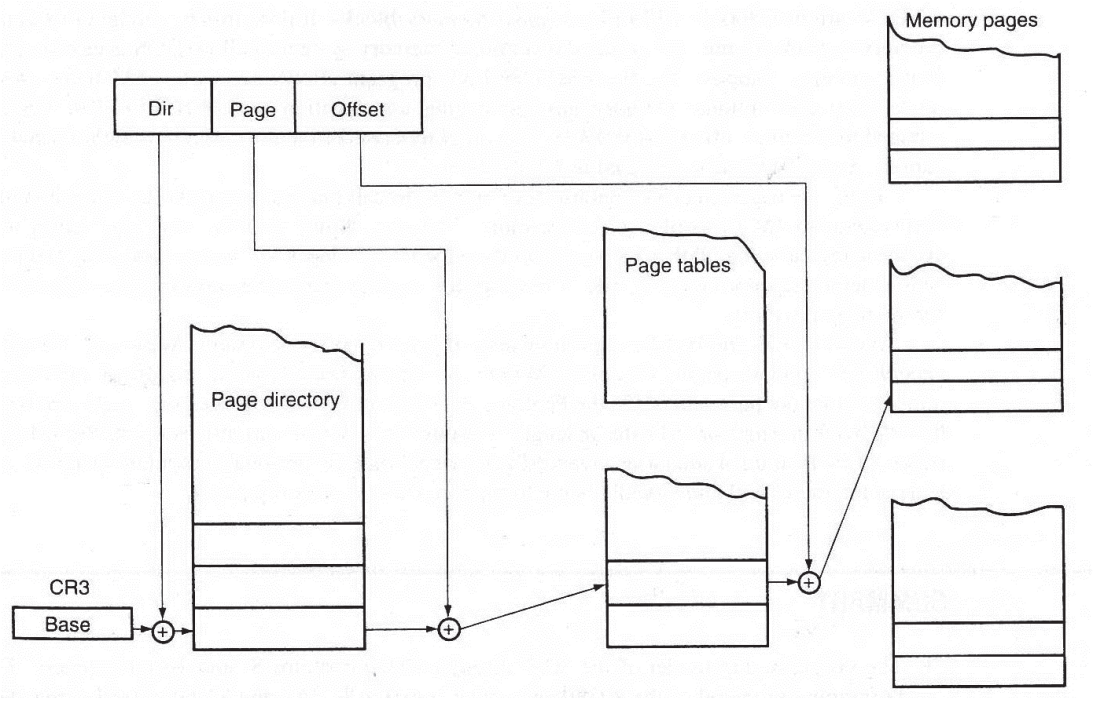
- Bellek adresinden daha değerli on altı bit seçicinin değerini veriyor.
- Seçici, açıklayıcılar tablosundan bölüt açıklayıcının bulunması için kullanılıyor.
- Açıklayıcı bölütün başlangıç ve sonuç adresinin hesaplanması için kullanılıyor.
- Kaydırma, seçilen bölütten bellek yerin bulunması için kullanılıyor.

9.3.1. Korumalı Sanal Çalışma Modu

Eğer sayfalar mekanizması kullanılıyorsa, o zaman 32 - bitli bellek adresi sanal adresi gibi yorumlanıyor. Sıralı adresten (şimdi sanal adresine dönüştürülmüş) fiziksel adresini elde etmek için üç bileşen kullanılıyor:

1. Terminaller dosyası.
2. Sayfalar tablosu.
3. Kullanıcı belleğinin parçası olan ve çerçeveye sınırlı olan sayfa.

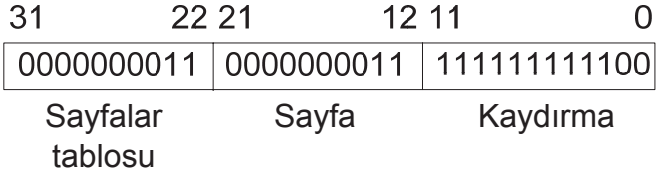
Resim 9.10.'da sanal adresten fiziksel adresin elde edilmesi için uygulanan mekanizma gösterilmiştir.



Resim 9.10. Sanal sayfaların kullanımıyla belleğe erişim

Sistemde sadece bir dosya var ve onun başlangıç adresi CR3 kontrol yazmacında bulunuyor. **Dosya** 1024 konumdan oluşuyor. Bu konumlardan hangisinin seçileceği, sanal adresten 22 ili 31 arası bitlerin değerlerine bağlıdır. Dosyadan seçilen yer, bizi 1024 **sayfalar tablosundan** birine yönlendiriyor. Dosyada konumların sayısı, sayfalar tabloların sayısına eşittir. Her sayfalar tablosu 1024 konumdan oluşuyor ve hangi yerin seçileceği sanal adresten 12 ile 21 arasındaki bitlerin değerine bağlıdır. Sayfalar tablosundan seçilen konum bir **sanal sayfanın** başlangıcına yönlendiriyor. Son olarak, sayfadan hangi yerin seçileceği, kaydırmaya yani sanal adresinden 1 ile 11 arasındaki bitlere bağlıdır.

Örnek 9.2:
00C03FFCH sanal adres değeri biliniyorsa, Sayfalar tablosunun sıra numarasını, sayfanın sıra numarasını ve seçilen sayfada veriyi belirle.
Çözüm:
Sanal adresi, ikili sayı sisteminde, resim 9.11'de gösterilmiştir.



Resim 9.11. Sanal adresindeki bitlerin anlamı

31 ile 22 pozisyon arasındaki bitler 3 sıra numaralı sayfalar tablosuna gösteriyor. 21 ile 12 arasındaki bitler üçüncü sayfaya yönlendiriyor. Aranan yer sayfanın başlangıcından FFCH mesafelik uzaklığındadır.

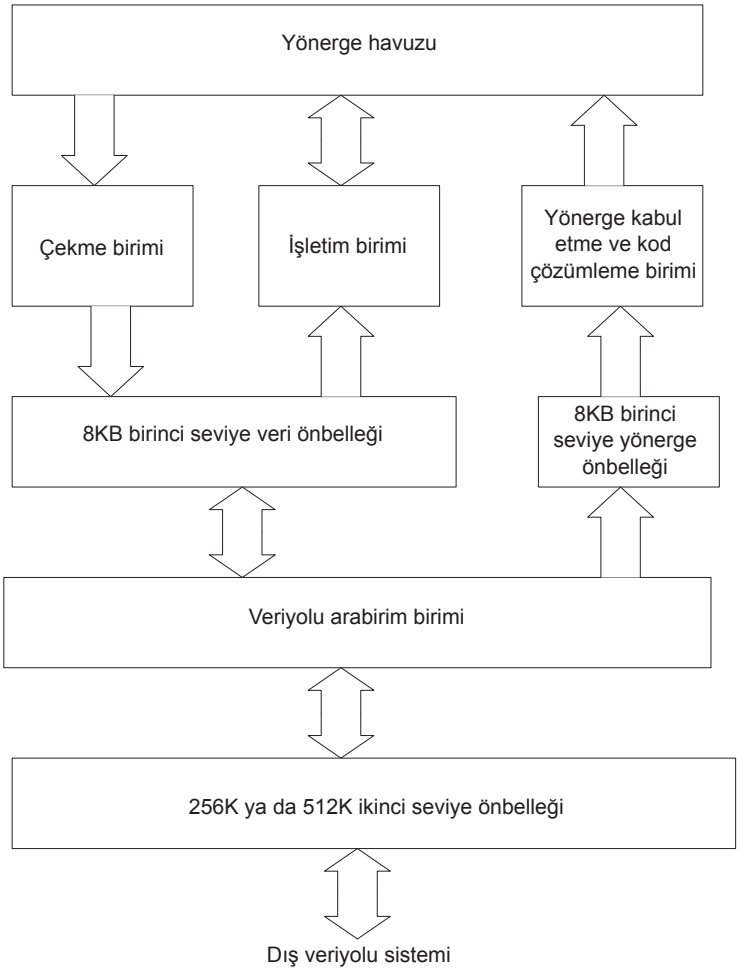
Çoğu kez mikroişlemci komşu bellek yerlerinden veri arıyor. Bellekte kontrol etme zamanın kısaltılması için 80386 mikroişlemcinin son kullanılan dosya - tablo - sayfa kombinasyonunu saklamak için donanım desteği var, kaydırmanın değeri (0'dan 11'e kadar bitler) değişiyor.

Şimdi sanal belleğin kapasitesini hesaplayacağız.
- Bir sayfanın büyüklüğü 2^{12} = 4 KB'tır (12 kaydırmada bitlerin sayısıdır)
- Sayfalar tablosunun 1024 sayfa x 4KB = 1KB x 4KB = 4 MB kapasitesi var.
- Sayfalar dosyası 1024 sayfalar tablosu içeriyor, buna göre sayfalar dosyasının kapasitesi 1024x4MB=1Kx4MB=4GB değerindedir.
- 4GB, aslında, bölütün sınırı sayfalarla ifade edildiği zaman bir bölütün kapasitesidir. Hatırlayalım, bellekte istenen verinin bulunması, bölüt yazmacında bölütün seçicisinin girilmesiyle başladı. Seçici, açıklayıcı seçmek için 13 - bitli endeks içeriyor ve iki açıklayıcı tablo arasından birinin seçilmesi için 1 bit içeriyor. Bu demek ki, seçiciyle 2^{14} =16K bölüt adreslenebilir. Bölüterin sayısı, bir bölütün kapasitesiyle çarpılırsa, 64TB kapasitesi elde ediliyor.

9.4. Pentium Pro Mikroişlemci

Pentium Pro mikroişlemcisi, yapısı açısından ondan önceki tüm mikroişlemcilerden farklıdır. Resim 9.12.'de Pentium Pro mikroişlemcinin yapısı gösterilmiştir.

Pentium Pro mikroişlemcisi farklı seviyede iki **önbellek** içeriyor. İkinci seviyede önbellek 256 KB ya da 512KB kapasitelidir ve anakartta takılmıştır. Birinci seviye önbellek mikroişlemci yongasında bulunuyor ve 8KB veri önbellek ve 8KB yönerge önbellek kapasitesi vardır.



Resim 9.12. PentiumPro mikroişlemcinin yapısı

Veriyolu arabirim birimi bellek adresleri ve kontrol sinyallerini üretiyor ve birinci seviyeden iki önbelleğe veriler ya da yönergeler gönderiyor.

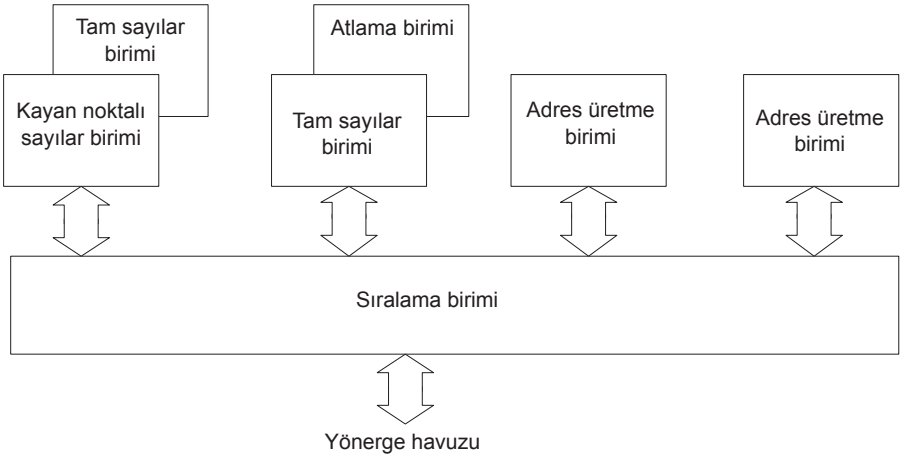
Yönerge kabul etme ve kod çözümleme birimi üç kod çözücüden oluşuyor. Onlar aynı zamanda üç farklı yönergenin kodunu çözebiliyor. Üç kod çözücünün çıkışları yönerge havuzuyla bağlıdır. Yönergeleri işletim birimi kabul edene kadar yö-

nergeler yönerge havuzunda kalıyorlar. Yönergeleri kabul etme ve kod çözümleme biriminde, programda atlama yönergesi bulunursa, programın akışını değiştiren özel mantık bölümü vardır. Mikroişlemcinin içinde bulunan yönerge havuz bellek yerlerin adreslerini saklayan küçük bellektir.

Yönergeleri **kabul etme ve yönergeleri çalıştırma** biriminin iç yapısı resim 9.13.'te gösterilmiştir. Üç işletim biriminin var olduğu görünüyor: tam sayılar için iki birim ve kayan noktalı sayılar için bir birim vardır.

Sıralama birimin çalıştırılması gereken yönergeleri sıralıyor ve onların listesini yapıyor. Sıralama birimi yönerge havuzuyla bağlıdır. Oradan çözümlenmiş ancak hala çalıştırılmamış yönergeleri alıyor.

Yönerge havuzuyla çekme birimi de bağlıdır. Çekme birimi çalıştırılan yönergeleri alıyor ve onları birinci seviye veri önbelleğine veriyor.



Resim 9.13. Yönergeleri kabul etme ve yönergeleri çalıştırma birimi

Pentium Pro mikroişlemcinin **belleği** sekiz bellek bankasına ayrılıyor. Bankalarda her konum bir bayt bilgi sığdırabilir ve her bankanın toplam 8GB kapasitesi var. Buna göre, Pentium Pro mikroişlemcinin 64GB kapasitelik belleği var. Adresleme, ek adres hatların, A_{32}'den A_{35}'e kadar hatların eklenmesiyle sağlanmıştır. Ek hatlarla beraber toplam 36 adres hattı var(2^{36} = 64GB). Pentium Pro mikroişlemci 64 - bitli veri veriyolu kullanıyor ve bu veriyolu sekiz bellek bankasından sekiz baytın aktarımı için kullanılıyor. Adres alanının artmasından dolayı, sayfaların büyüklüğü de 2MB'a artmış. 80486 ve Pentium I mikroişlemcilerde olduğu gibi, Pentium Pro da iç çift değer kontrolu kullanıyor, yani her bellek yeri için sekizer bit dışında, çift değer kontrolu için bir bit daha eklenmiştir.

Pentium Pro mikroişlmcide yenilik iç **hata onarma devresidir** (EEC Error Correction Circuit). Bu devre yardımıyla iki bit hatanın bulunması ve bir bit hatanın onarmasını sağlanıyor. Bunu başarmak için, sekiz ek bit gerekiyor ve onlar 64 - bitli sayılarda eklenmiştir. Ek olarak eklenmiş bu sekiz bit hatanın otomatik onarma kodunu içeriyor. EEC kodları çok güvenilirdir ve modern bilgisayarlarda sıkça kullanılıyorlar. Tek dezavantajı, 72 bitli genişliği olan (64 - bitli veri+9 - bitli kod) SDRAM belleğin pahalı olmasıdır.

9.5. Pentium 2 Mikroişlemci

Önceki mikroişlemcilere kıyasen, Pentium 2 mikroişlemcinin farkı, mikroişlemcinin ana kartında yerleşmiş olmamasıdır. Pentium 2 mikroişlemcide, mikroişlemci özel tasarlanmış plastik yatakta (slot - yuva) yerleşmiş bulunuyor. Çalışma frekansı 233 MHz ile 450MHz sınırları arasındadır, veriyolu hızı ise 66MHz ile 100MHz arasındadır. İkinci seviye önbelleğin kapasitesi 512KB, 1MB ya da 2MB olabilir. Pentium 2 mikroişlemcisi **64 - bitli veri veriyolu ve 36 - bitli adres veriyolu kullanıyor**. Birinci seviye önbellek kapasitesi 32KB'tır, ancak **ikinci seviye önbelleği** aynı paketle ve birinci seviye önbellekle birbirine çok yakın olmalarına rağmen, **artık mikroişlemcinin aynı tümleşik devresinde bulunmuyor**. Bu şekilde mikroişlemcinin etkisi memnun edicidir, fiyatı ise azalıyor. Önbelleğin hızı, Pentium 2 mikroişlemcinin hızından iki kat daha düşüktür. Mikroişlemcinin 400MHz çalışma frekansı varsa, o zaman önbellek 200MHz hızla çalışacak. Celeron mikroişlemcisi, aslında, Pentium 2 mikroişlemcinin bir sürümüdür, sadece onda ikinci seviye önbellek yoktur. Celeron mikroişlemcide ikinci seviye önbelleği özel olarak ayrılmış ve anakartında yerleşmiştir ve 66MHz hızla çalışıyor. Xeon mikroişlemcisi, Pentium 2 mikroişlemcinin en iyi sürümüdür, ikinci seviye önbellek içeriyor ve önbelleğin hızı mikroişlemcinin hızına eşittir.

Pentium 2 mikroişlemcinin 242 pini vardır. Pentium 2 mikroişlemcinin hemen tüm **pin** grupların **işlevlerini** kısaca açıklayacağız:

$\overline{A20}$ ⟶ A20 üzerinde daha değerli adres bitlerini maskeliyor. Bu şekilde 8086 mikroişlemciyle uyumluluk sağlanıyor.

$\overline{BE7}$-$\overline{BE0}$ ⟶ Sekiz bellek bankasından birini seçiyorlar.

$\overline{A35}$-$\overline{A3}$ ⟶ Belek bankasından bellek yeri seçiyorlar.

$\overline{ADS}$ ⟶ Geçerli adresin gönderildiğini işaretliyor.

$\overline{AERR}$ ⟶ Adresin çift değer kontrolunu aktifleştiriliyor.

$\overline{AP1}$-$\overline{AP0}$ ⟶ Adreste hata olduğunu sinyal veriyor

$\overline{BCLK}$ → Veriyolun frekansını belirliyor, 66MHz ya da 100MHz.
$\overline{BERR}$ → Veriyolu sisteminde hata sinyali veriyor.
$\overline{BINT}$ → Veriyolu sistemini sıfırlandırıyor.
$\overline{BNR}$ → (Bus not ready) Bekleme için dijital takt ekliyor.
BP[3:2] → (Breakpoints) Ayıklama yazmaçların çalışmalarında uyumluluk durumu sinyalı veriyor.
$\overline{BPRI}$ → (Bus Prioriy Request) Sistem veriyolunu kullanmak için arama göndermek için kullanılıyor.
$\overline{BP0}$-$\overline{BP1}$ → (bus Request) Pentium 2'nin veriyolu için arama gönderdiği sinyalı veriyor.
$\overline{D63}$-$\overline{D0}$ → Veri pinleri
$\overline{DEFER}$ → Dış sistemin makine döngüsünü tamamlamayacağı sinyalini veriyor.
$\overline{DEP7}$-$\overline{DEP0}$ → Veri aktarımında hata olduğunu sinyalleştiriyor.
$\overline{DRDY}$ → (Data Ready) Geçerli verinin gönderildiğini sinyalleştiriyor.
$\overline{FERR}$ → Sayısal yardımcı işlemcinin çalışmasında hata sinyali veriyor.
$\overline{FLUSH}$ → Okumak ve yazmak için önbellek hatlarını etkisiz duruma getiriyor.
$\overline{HIT}$ → Önbellekte başarılı veri bulunduğunu sinyalleştiriyor.
$\overline{HITM}$ → (Hit Modified) Önbelleği, onda yazıldığı sırada başka birimler tarafından kullanmasını engellemek amacıyla aktifleştiriliyor.

$\overline{IERR}$ → Sistemde iç hata sinyali veriyor ve kesinti başlatılıyor.

$\overline{IGNNE}$ → Sayısal yardımcı işlemcinin çalışmasında meydana gelen hatanın yoksayılmasına yol açıyor.
$\overline{INIT}$ → Önbellek, giriş çıkış arabirimi ve kayan noktalı yazmaçların dışında da sistem sıfırlanıyor.

$\overline{INTR}$ → Bazı dış aygıt kesinti arayınca aktifleştiriliyor.
$\overline{LOCK}$ → Böyle öneki olan yönergenin çalıştırılması sırasında mantıksal sıfır oluyor. Genelde belleğe doğrudan erişim sırasında kullanılıyor

NMI → Maskelenmeyen kesintiler
PICCLK → BCLK pinin frekansından, ¼ frekanslı dijit palsı üretiyor.

$\overline{PM1}$-$\overline{PM0}$ → (Performance Monitor) mikroişlemcinin performanslarını test eilmesi sırasında kullanılıyorlar.
PRDY → (Probe Ready) Ayıklamanın açıldığını sinyalleştiriyor.
$\overline{PREQ}$ → (Probe Request) Ayıklama aramasını sırasında aktifleştiriliyor.
$\overline{REQ4}$-$\overline{REQ0}$ → (Request Signal) Veriyollar denetimcinin ve mikroişlemci arasında iletişim için kullanılıyorlar.
$\overline{RESET}$ → Başlangıç adresi FFFFFFF0H olan 000FFFFF0H yazılımın çalıştırılmasını başlatıyor.

$\overline{RP}$ ⟶ (Request Parity) Çift değer kontrolunun aranması sırasında aktifleştiriliyor.

$\overline{RS2}$-$\overline{RS0}$ ⟶ (Request Status) Mikroişlemcinin durumununun gösterilmesi arandığı sırasında aktifleştiriliyor.

SLP ⟶ İşlemci bekleme (dinlenme) düzenine giriyor (Sleep).

$\overline{SMI}$ ⟶ (System Management Interrupt) Mikroişlemci sistem yönetim moduna giriyor.

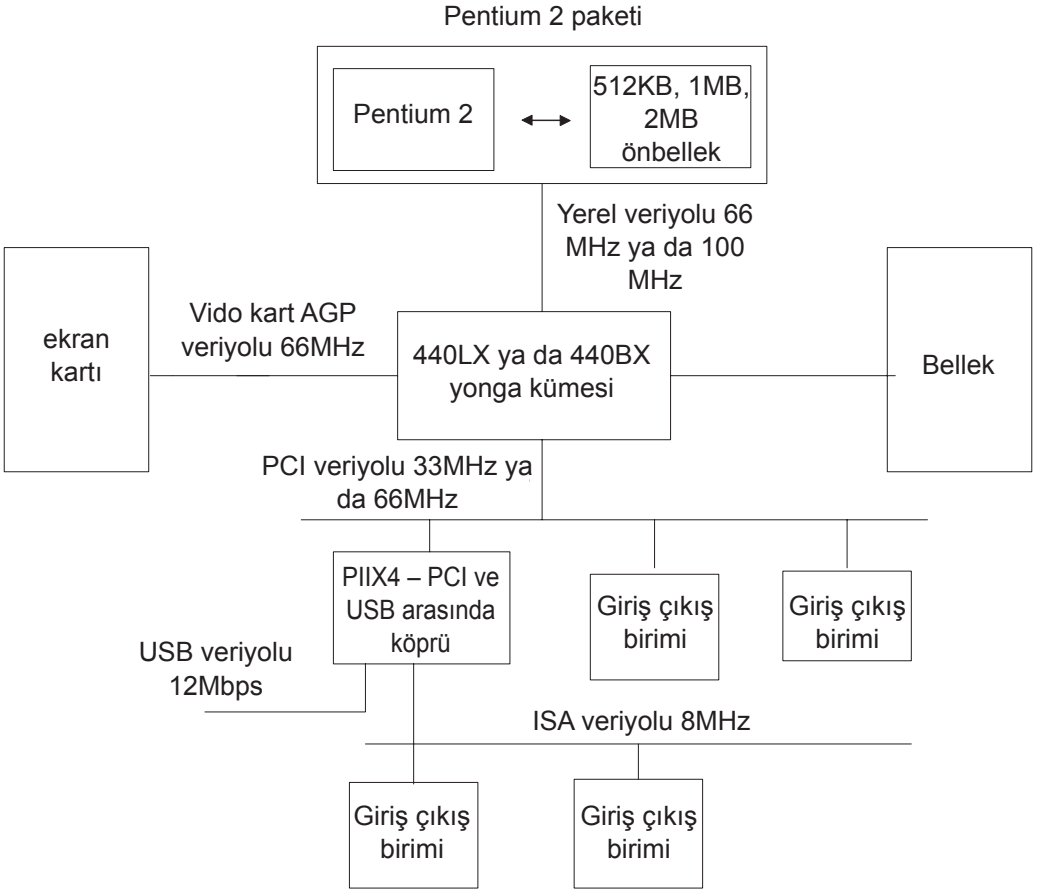
$\overline{THERMTRIP}$ ⟶ (Thermal Sensor Trip) Mikroişlemcinin sıcaklığı 130^{0}C üzerine artarsa aktifleştiriliyor.

TMS ⟶ (Test Mode Select) Test etme düzenin türü seçiliyor.

$\overline{TRDY}$ ⟶ (Target Ready) Mirkoişlemcinin dönmeli yazma işleminin başlamasına yol açıyor.

$\overline{VID4}$-$\overline{VID0}$ ⟶ (Voltage Data) Elektrik kaynağı ve topraklama pinleri

Pentium 2 mikroişlemci ve bellek arasında veri aktarımın kontrolü için 440LX ya da 440B yonga kümesi kullanılıyor. Yönga kümesi iletişimde aracılık yapıyor, eski geleneksel iletişim şeklinde ise mikroişlemci bellekle doğrudan bağlıdır. Resim 9.14.'te **Pentium 2 bilgisayar sistemin** yapısı tüm veriyolu türleriyle beraber gösterilmiştir.



Resim 9.14. Pentium 2 bilgisayar sistemin yapısı

ISA (Industrial Standard Architecture) en eski veriyolu türüdür, 8 ya da 16 bit genişliğindedir ve çalışma frekansı 8MHz'e kadar olan yavaş dış aygıtlar ya da belleklerin bağlanması için kullanılıyor. Devamda bu veriyolu, iki farklı geçiş kapsamı 1.5Mbps ya da 12Mbps (bir per second – bit saniyede) olan USB (Universal Serial Bus) veriyoluyla değişiyor. PCI (Peripheral Component Interconnect) 33MHz'e kadar çalışma frekanslı, mikroişlemci ve dış aygıtlar ya da bellek arasında 32 ve 64 - bitli aktarmalar destekliyor. AGP (Accelerated Graphics Port) veriyolun mikroişlemciyle vaynı çalışma freakansı var, geçiş kapsamı 528M bayttır ve sadece video kartın bellek sistemiyle bağlanması için kullanılıyor.

9.6. Pentium 3 Mikroişlemci

Pentium 3 mikroişlemcinin temel özellikleri şunlardır: 133MHz çalışma frekansı sırasında bir dijit palsı içinde dört 64 bitli verinin aktarımı, iki 16KB kapasiteli birinci seviye önbellek ve 15KB kapasiteli bir ikinci seviye önbelleği, SSE yönergelere destek ve programlarda dallanmaların öngörülmesi.

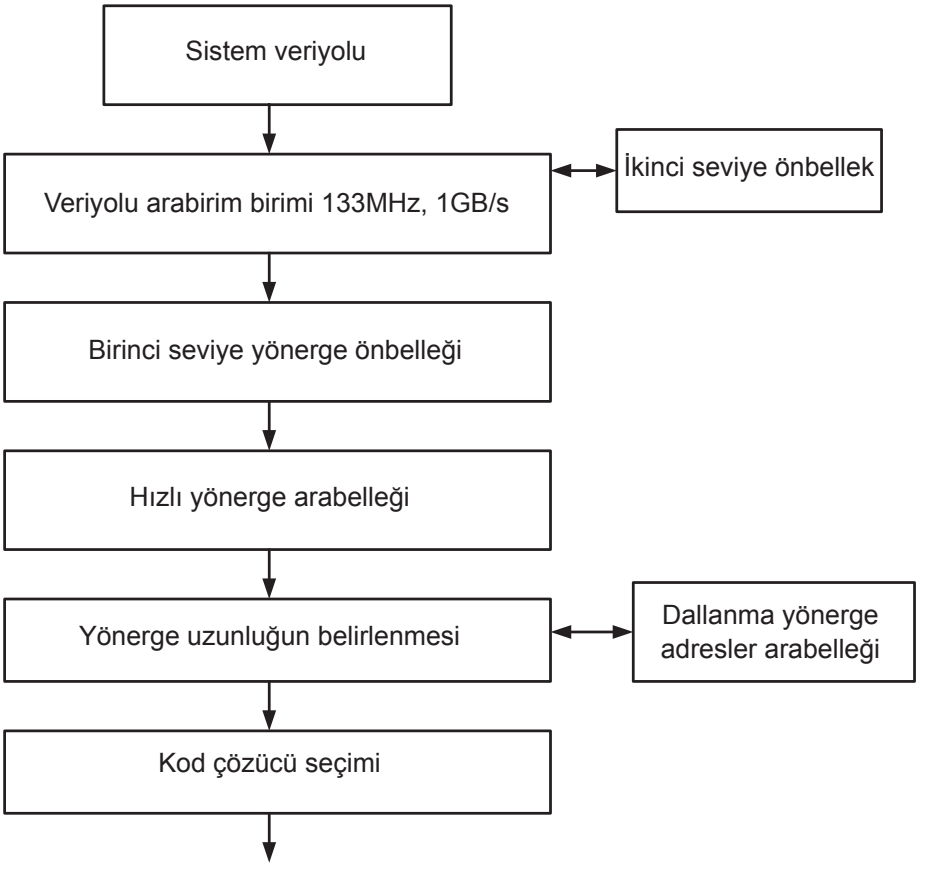
Pentium 3 mikroişlemcide, yönergelerin çalıştırılması **on bir aşamadan** geçiyor. Metnin devamında yönergenin çalıştırılmasından tüm aşamalar açıklanmıştır.

İlk üç aşama **yönergenin aktarılması ve alınması** için kullanılıyor. Resim 9.15.'de yönerge aktarma biriminin blok diyagramı gösterilmiştir.

- Aşama 1: Hızlı yönerge arabiriminde L1 yönerge önbelleğinden bir önbellek hattı giriyor. Önbellek hattı 256 bit büyüklüğündedir.
- Aşama 2: Önbellek hatından yönerge ayrılıyor. Yönergeler değişik büyüklükte oldukları için, ilk 128 bit ayrılıyor. Dallanma yönergesi (branch) söz konusu olursa, adres dallanma yönergelerin ara belleğinde yerleşiyor ve mikroişlemci daha geç oradan çağırıyor.
- Aşama 3: Mevcut yönerge için üç kod çözücüden biri seçiliyor.

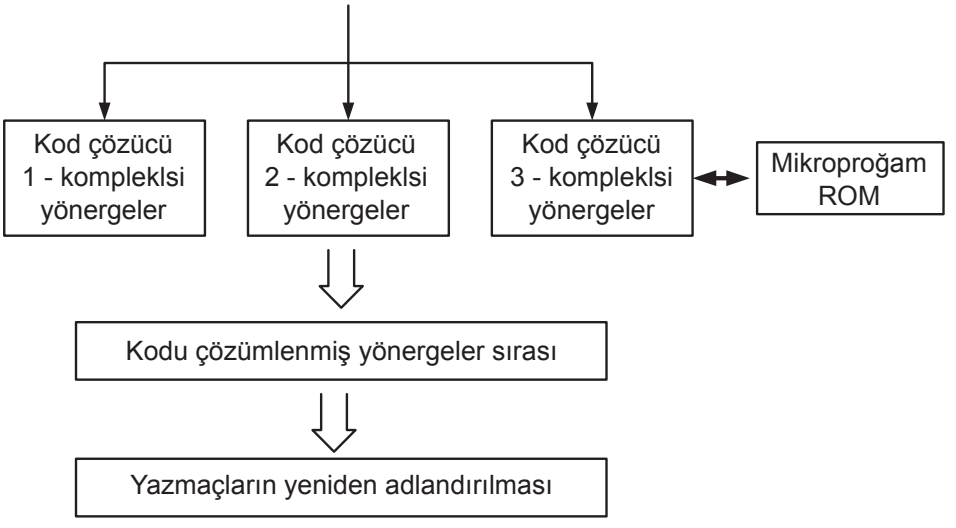
Yönerge aktarıldıktan sonra, yönerge kodunun çözümlenmesi gerekiyor ve bu işlem için iki aşama gerekiyor.

- Aşama 4: Yönerge kodunun çözümlenmesi için üç kod çözücüden biri kullanılıyor. Buna gore aynı anda üç yönergenin kodu çözülebilir. Kod çözümlenmesi sonucu olarak birkaç mikro yönerge elde ediliyor.
- Aşama 5: Mikro yönergeler, yönergeler sırasında giriliyor. Bir yönergeden altıdan fazla mikro yönerge elde edilirse bu aşamanın tekrarlanması lazım.



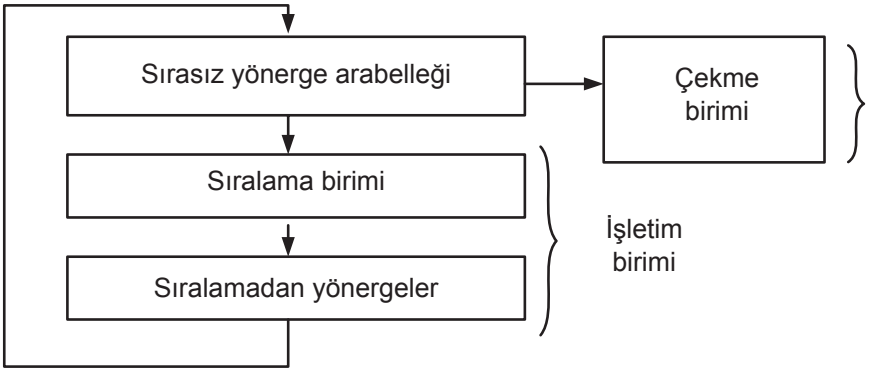
Resim 9.15. Pentium 3 mikroişlemcide yönerge aktarma birimi

Resim 9.16.'da **kod çözümleme ve yeniden adlandırma biriminin** blok diyagramı verilmiştir.



Resim 9.16. Pentium 3 mikroişlemcide kod çözümleme ve yeniden adlandırma birimi

- Aşama 6: Yönergeler programda yazılmış olduğu sıralamayla çalıştırılmadığından dolayı, yazmaçların durumunu takip etmek zordur. Bundan dolayı her mikro yönergeye 40 iç yazmaçtan birer yazmaç veriliyor.
- Aşama 7: Mikro yönergeler, onların sıralamasına dikkat verilmeden, arabelleğe (**reorder buffer** - sırasız arabellek) sıralamadan giriliyor. Mikro yönergeler sırasız arabellekten, mikro yönergelerin çalıştırılması için tüm veriler mevcut ise ve sıralama biriminde serbest giriş varsa, sıralama birimine getiriliyor.
- Aşama 8: Bu aşamada mikro yönerge sıralama biriminde bulunmalıdır ve mikro yönerge uygun işletim birimine gönderiliyor.Sıralama birimi işletim birimleriyle dört bağlantı noktası aracılığıyla bağlıdır.
- Aşama 9: Pentium 3 mikroişlemcinin **altı işletim birimi** vardır: tam sayılar için iki birim, kayan noktalı sayılar için iki birim ve bellekle iletişim için iki birim. Her yönergenin tamamlanması için bir dijit pals süresi gerekiyor.
- Aşama 10: **Çekme birimi** sırasız arabellelten hangi mikro yönergelerin çalıştırılmış olduğunu kontrol ediyor.
- Aşama 11: Çekme birimi sırasız arabellekten çalıştırılmış mikro yönergeleri çekiyor ve programda yazılmış şekilde sıralıyor. Bu işlem resim 9.17.'de gösterilmiştir.



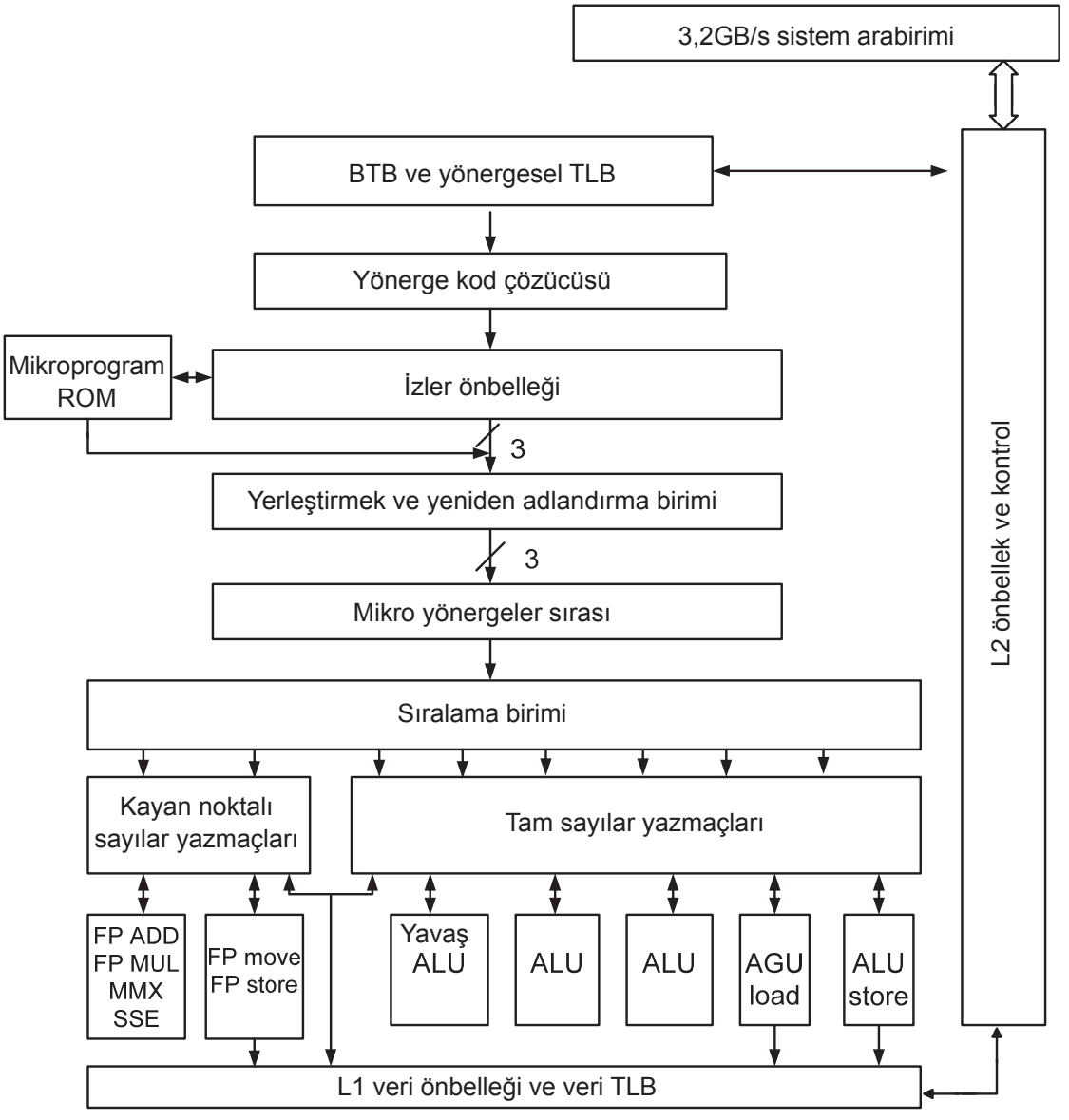
Resim 9.17. Çekme biriminin işlevi

9.7. Pentium 4 Mikroişlemci

Pentium 4 mikroişlemcisi, yapısı açısından Pentium Pro mikroişlemciyle çok benzerdir. İlk Pentium 4 mikroişlemcinin 1.3GHz çalışma frekansları varmış, bugünkü mikroişlemcilerin ise 3.2GHz çalışma frekansları var. Pentim 4 mikroişlemcinin iki paketi var: 423 - pinli PGA ve 478 - pinli FC - PGA2.

Resim 9.18.'de **Pentium 4 mikroişlemcinin yapısı** gösterilmiştir. Sistem veriyolun geçiş kapsamı 3.2GB/s'dir. Veriyolun genişliği 64 bit değerindedir ve çalış-

ma frekansı 100MHz'tir. Ancak onun hızı frekanstan dört kat daha büyüktür, çünkü bir dijit pals süresi içinde dört veri aktarılıyor ve gerçekte veriyolu 400MHz frekansı olduğu gibi aktaracatır. Bu teknik Quad Data Rate (QDR) adıyla biliniyor ve **dördünlü veri geçiş kapsamı** anlamına geliyor. İkinci seviye önbelleği 256 KB, 512KB, 1MB ya da 2MB kapasiteli olabilir, birinci seviye yönerge önbelleği ise 8KB ya da 16KB olabilir.



Resim 9.18. Pentium 4 mikroişlemcinin yapısı

Belki kimse Pentium 4 mikroişlemcinin yönerge önbelleği olmadığını düşünebilir. Yönerge önbelleği vardır, ancak yönerge kod çözücüden sonra bulunuyor ve **izler önbelleği** adıyla biliniyor. İzler önbelleği içinde 12K mikro yönerge sığdırılabilir.

Bir mikro yönerge 100 bit olduğundan dolayı, izler önbelleğin kapasitesi 150KB büyüklüğündedir. Yönerge önbelleğin kod çözücüden sonra yerleştirilmesi, kod çözücüyü, döngü yönergeleri gibi devamlı tekrarlanan yönergelerin kod çözümlemesinden "koruyor". İzler önbelleği önünde iki arabellek yerleşmiştir: BTB (Branch Target Buffer) ve TLB (Transition Lookside Buffer). **BTB arabelleği** çağrılan atlama ve dallanma yönergelerin adreslerini içeren küçük bellektir.

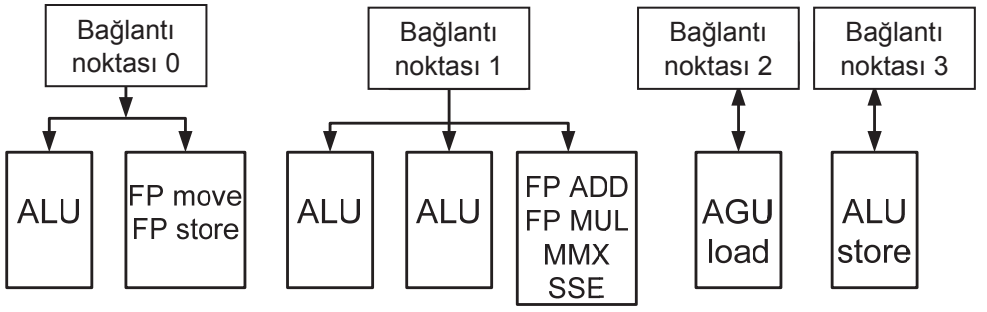
Kod çözücüsü bir dijit palsı süresi içinde bir yönerge işletebilir. Kodu çözümlenen yönerge kompleksli ise dörtten fazla mikro yönerge elde edilebilir. Mikro yönergeler, yönergenin çalıştılrılması için gereken kontrol sinyallerini veren ROM belleğine gönderiliyor. Önbellek bir dijit pals zamanı içine yeniden adlandırılma ve yerleştime birimine üçer mikro yönerge gönderebilir.

Yerleştirme birimi mikro yönergenin işletim birimine gelince herşeyin hazır olması için kaynakların hazırlığını yapıyor. Yerleştirme birimi her mikro yönerge için 126 arabellekten birini ayırıyor ve ayrılan arabelleğin aracılığıyla mikro yönergenin durumunu takip ediyor. Mikro yönergeler sıralamaya önem verilmeden çalıştırılabilir, çünkü mikro yönergelerin çalıştırılması tamamlandıktan sonra mikroişlemci arabelleklerin yardımıyla mikro yönergeleri doğru sıralamaya getiriyor. Yerleştirme birimi mikro yönergenin çalıştırılmasıyla elde edilen sonuçları yerleştirme için iç yazmaçlar ayırtıyor. **Yazmaçlar birkaç gruba ayrılıyor**: bellek yazmaçları, tam sayılarla çalışma yazmaçları ve kayan noktalı sayılarla çalışma yazmaçları. Resim 9.18.'de kayan noktalı sayılarla çalışma yazmaçları FP harfleriyle işaretleniyor ve İngilizce Floating Point sözlerinin kısaltmasıdır (anlamı - kayan nokta). LOAD ya da STORE yönergeleri söz konusu olursa, o zaman yerleştirme birimi, doldurmak (load) için kullanılan 48 arabellekten birini ya da saklamak (store) için kullanılan 24 arabellekten birini ayırtıyor.

Yer değiştirme birimin de ilgiinç işlevi vardır. Pentium mikroişlemcinin programcıya görünür 32 yazmacı var: EAX, EBX, ECX, EDX, EDI, ESI, EBP ve ESP. Bu sayı çok azdır, özellikle yönergelerin çalıştırılmasında ardaşıllık olmadığını göz önüne alırsak. Bundan dolayı bu yazmaçlar 256 iç yazmaçtan birine adını değiştiriyorlar (yeniden adlandırılıyorlar). Onlardan 128 yazmaç tam sayılar için ve 128 yazmaç kayan noktalı sayılar için. Yeniden adlandırma birimi bir dijit palsında üç mikro yönerge işletebiliyor. Adları değişmiş ve yerleşmiş mikro yönergeler, mikro yönergeler sırasına giriyor ve orada sıralama birimi tarafından alınmalarını bekliyorlar.

Sıralama birimi mikro yönergeleri kodları çözümlenmiş sırasıyla sıralıyor. İzler önbelleğinde ve yerleşme biriminde mikro yönergenin her kısmı devamlı çalışması amacıyla bu yönergeler karışmıştı. Sıralama birimi mikro yönergeleri birkaç gruba ayırıyor: bellek yönergeleri, hızlı (basit) yönergeler, kayan noktalı sayılarla çalışmak için daha basit yönergeler ve kompleksli (yavaş) yönergeler. Bu sıralama şekli

verilerin işletim birimine hızlı aktarımı için çok önemlidir. Sıralama birimi ve işletim birimi aralarında dört bağlantı noktasıyla bağlıdır. Onlardan iki bağlantı noktası bir dijit palsında ikişer mikro yönerge aktarıyor, iki bağlantı noktası ise birer mikro yönerge aktarıyor. Bağlantı noktalar ve işletim birimi resim 9.19.'da gösterilmiştir. Pentium 4 mikroişlemcinin 6 işletim birimi var, onlardan tam sayılarla çalışmak için iki birim ve kayan noktalı sayılarla çalışmak için iki birim ve RAM belleğinden veri aktarımı için iki birim (load, store). Resim 9.19'da son iki birim AGU (address Generator Unit) işaretiyle işaretlenmiştir ve **adres üretici** anlamına geliyor. Bağlantı noktaları 2 ve 3'ün sadece bellekle iletişim birimle bağlı olduklarını görüyoruz. Bunun amacı gereken verilerin aktarımının hızlandırılmasıdır.



Resim 9.19. Sıralama birimin ve işletim birimin bağlanması için bağlantı noktaları

Altı işletim birimi paralel olarak çalışıyor. Örneğin, kayan noktalı sayılarla çalışmak birimine bir mikro yönergeyi tamamlaması için birkaç dijit palsı gerekiyor. Ancak bu birim çalışırken bağlantı noktaları 0 ve 1, işletim biriminin tüm kapasiteyle durmadan çalışması için diğer daha basit aritmetik mantık birimine veriler gönderecek. Basit aritmetik mantık biriminden ikisi bir dijit palsında ikişer mikro yönerge çalıştırabilir ve o şekilde tüm işletim birimleri için bir dijit palsında yedi yönerge toplam kapasitesi elde ediyoruz.

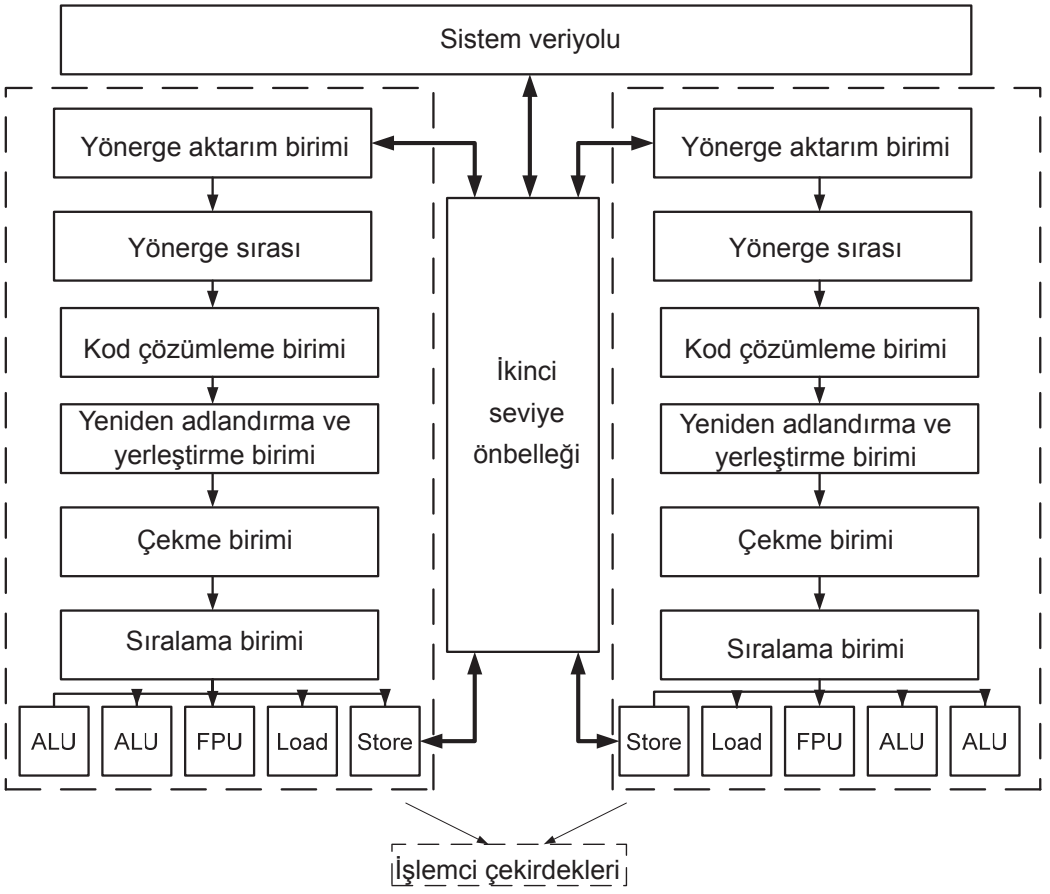
9.8. Pentium Dual Core Mikroişlemci

Bir çekirdekli geleneksel mikroişlemcilerde, merkezi işlem birimi aktarmak, çalıştırmak ve verilerin saklanmasının için sorumludur. Ancak, RAM'ın ve veriyolların hızı sorundur, çünkü onları hızı mikroişlemcinin hızından çok daha düşüktür ve mikroişlemcinin çalışmasını yavaşlatıyor. Bu sorun multişletmeyle, yani aynı zamanda fazla uygulamaların çalıştırılmasıyla daha da büyük oluyor. Başlangıçta mikroişlemci sistemler üreticileri, multiişletim sorununu mikroişlemcinin çalışma frekansını artırarak çözmeye denemiş.

Ancak bu çözüm çok zor ve pahalıymış, çünkü daha yüksek frekans daha fazla ısıtma demektir ve çalışmada hataya ve zarara yol açabilir. Bundan dolayı tümleşik devrede bir mikroişlemci daha takmaya düşünmeye başlamışlar. Daha basit sözlerle iki kafa bir kafadan daha iyi düşünüyor, dört el iki elden daha elverişlidir. İki çekirdek gerçekleşmesi gereken uygulamayı ayırıyor. Bir program çalıştığı zaman bile, bir mikroişlemci çekirdeği yönergeleri çalıştırıyor, diğer işlemci ise belleğe ulaşıyor ve aktarılan yönergelerin kodlarını çözüyor. İki mikroişlemci çekirdeği arasında alıp verilen veriler mikroişlemci yongasından çıkmadıkları için sinyallerin güçlenmesine gerek kalmıyor. Bu durum da daha az enerji harcamasına ve daha az ısıtmaya yol açıyor. Bu özellikle taşınır bilgisayarlar – dizüstü bilgisayarlarda (laptoplarda) önemlidir.

Sistemin iki çekirdeği tanıması için SMT (Simultaneous Multi - threading Technology) kısaltmasıyla bilinen özel yazılım gerekiyor. SMT aslında işletim sistemin bir parçasını tanımlıyor.

Resim 9.20.'de ikiçekirdekli mikroişlemcinin yapısı gösterilmiştir.



Resim 9.20. İkiçekirdekli mikroişlemcinin yapısı

İşlemci çekirdeklerin kendine ait birinci seviye veri ve yönerge önbellekleri, ayrı kod çözücüleri ve işletim birimleri var, ancak aynı veriyolu denetimciyi

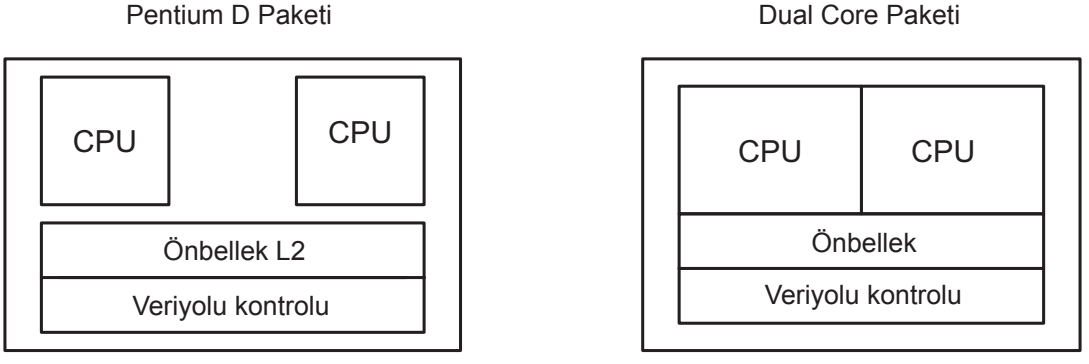
ve aynı ikinci seviye önbelleğini palyaşıyorlar. İki (çift) çekirdekli mikroişlemciler, bekleme zamanın azalması için birçekirdekli mikroişlemcilerden daha büyük önbellek arıyor. Bugün dört çekirdekli mikroişlemcilere de rastlanabilir, ancak onların çalışması için özel bellek denetimcisi gerekiyor.

İki (çift) çekirdekli mikroişlemciler multi işletim (çoklu işletim) sistemlerden farklıdır. Multiişletim sistemlerinde kendi kaynaklarına sahip olan iki ayrı mikroişlemci vardır, ikiçekirdekli mikroişlemcilerde ise kaynaklar paylaşılıyor. Multi işletim sistemleri, iki çekirdekli mikroişlemciler sistemlerinden daha hızlıdır, ancak harçlar da daha büyüktür.

Günümüzde, aralarında yapıları açısından faklı olan, büyük sayıda iki çekirdekli mikroişlemciler vardır. Pentium D mikroişlemcisi iki çekirdeklidir, ancak iki merkezi mikroişletim birimi bir yongada değildir. Onlar ayrı yongalarda, ancak birbirine çok yakın, bir pakette yerleşmiş bulunuyorlar. Yapısı açısından Pentium D mikroişlemcisi, Pentium 4 mikroişlemciye çok benzerdir. Şöyle ki, resim 9.20. hemen tamamiyle Pentium D mikroişlemcinin yapısına uygundur. Aynı pakette iki ikiçekirdekli mikroişlemci yanısıra, ikinci seviye önbelleği ve veriyolu arabirim birimi de yerleşmiştir. Bu mikroişlemcinin işletim gücü daha yüksek olabilir, ancak ısınma da çok daha yüksektir.

Bu eksikliğe **Pentium Dual Core** mikroişlemcide çözüm bulunmuştur. Pentium Dual Core mikroişlemcide tüm birimler aynı bir tümleşik devrede (pakette değil) yerleştirilmiştir. Bu iki ikiçekirdekli mikroişlemci türü resim 9.21.'de gösterilmiştir.

Pentium Core 2 Duo, Pentium Dual Core mikroişlemcinin ardılıdır. Pentium Dual Core mikroişlemciden farklı olarak, Core 2 Duo mikroişlemcide iki mikroişlemci çekirdeğin ayrı ikinci seviye önbellekleri ve veriyolu ulaşımı (FSB - Front Side Bus) vardır. Bu özellikler iki kat daha büyük hız ve daha iyi performanslar demektir.



Resim 9.21. Pentium D ve Dual Core mikroişlemci arasında kıyaslama

Aşağıda birkaç iki çekirdekli mikroişlemci türün performansları gösterilmiştir:

- Pentium Dual Core ⟶ 2GHz frekans, 1MB ikinci seviye önbellek, 200MHz veriyolu hızı
- Pentium Core 2 Duo ⟶ 1.86 GHz frekans, 2MB ikinci seviye önbellek, 266 veriyolu hızı
- Pentium Core 2 Extreme ⟶ 3.2GHz frekans, 2x48MB ikinci seviye önbellek, 333MHz veriyolu hızı

Her yeni çıkan ikiçekirdekli mikorişlemciyle, önbelleğin büyüklüğü artıyor, aynı zamanda sistem veriyolunun hızı da artıyor.

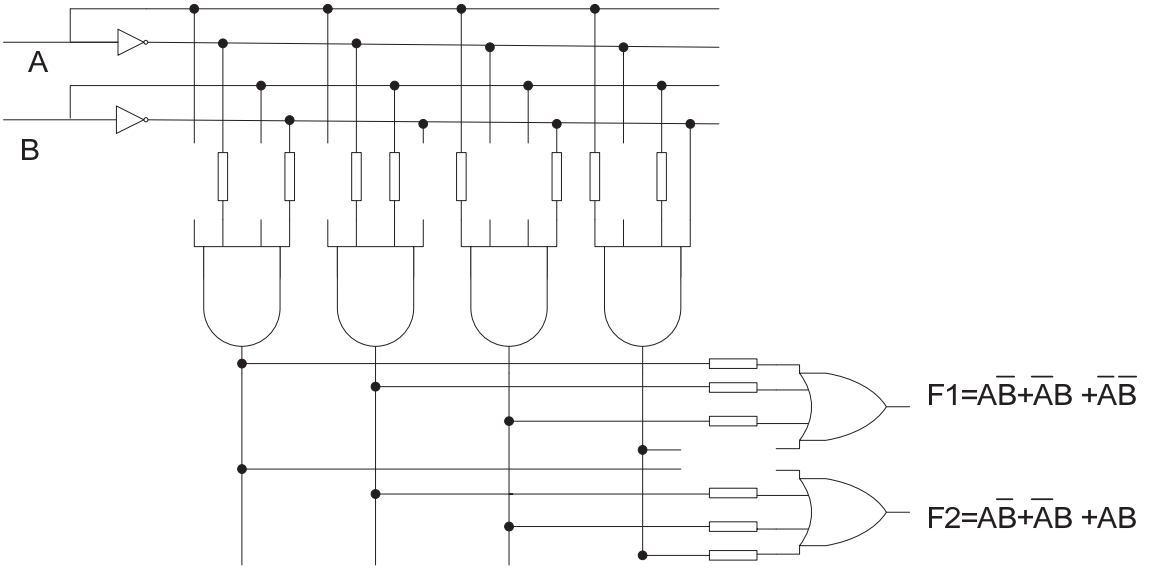
9.10. Pentium Mikrobilgisayar Sistemlerin Oluşması için Tümleşik Devre

Pentium mikroişlemcilerin geniş kullanımları var. Pentium mikroişlemci deyince aklımıza kişisel bilgisayar geliyor, ancak Pentium mikroişlemcileri cep telefonlarında, taşınabilir bilgisayarlarda, audio ve video donanımlarda vb. yerlerde kullanılıyor. İkinci konunun girişinde, 2.1. Mikrobilgisayar Sistemlerine Giriş konusunda mikrobilgisayar sisteminin merkezi işletim birimi, çalışma (geçici) ve kalıcı bellekten oluştuğunu söylemiştik. Dış aygıtların eklenmesiyle bilgisayar sistemleri elde ediliyor. Örneğin, klavye ve monitörün eklenmesiyle küçük bir bilgisayar sistemi elde ediliyor. Biz Pentium mikroişlemcinin bellekle ve dış devrelerle bağlanmasına fazla ağırlık vereceğiz. Adres kod çözümlenmesini ve programlanabilir kod çözücülerini açıklayacağız. Ayrıca, Pentium mikroişlemcinin bellek ve dış aygıtlarla bağlanması için özel olarak tasarlanmış tümleşik devreler olan yonga kümesinin yapısını ve pinlerini açıklayacağız.

9.10.1. Pentium Mikroişlemcinin Bellekle Bağlanması İçin Tümleşik Devreler

Programalanabilir kod çözücüleri, PLD (Programmable Logic Device) kısaltmasıyla biliniyorlar. Aynı şekilde çalışan ancan farklı isimleri olan çok sayıda PLD kod çözücüler türü vardır: PLA (Programmable Logic Array), PAL (Programmable Array Logic), GAL (Gated Array Logic). Kod çözücülerin programlanması, birinci konuda, yani 1.7.3. ROM bellekler konusunda incelediğimiz PROM bellekler programlanması yapıldığı gibi aynı şekilde yapılıyor. Bazı PLD kod çözücülerin içerikleri değişebilir. EPROM belleklerde olduğu gibi bazı PLD kod çözücülerin içerikleri silinip yeniden programlanabilir.

Resim 9.22.'de bir genel PLD kod çözücünün yapısı verilmiştir. Resimde gösterilmiş PLD kod çözücüsü iki sıradan oluşuyor: VE devrelerinden oluşan sıra ve YADA devrelerinden oluşan sıra. Her giriş mantıksal değişkeni tanımlıyor. Kod çözücünün her çıkışı girişlerin durumuna bağlıdır. Bir YADA devrede girişlerin sayısı VE devrelerin sayısına eşittir. Çıkışların girişleriden bağımlılığın nasıl olacağı hangi sigortanın yanmış olup olmadığına bağlıdır. VE devrenin girişinde yanmış sigortalar mantıksal sıfır durumu demektir ve onlar VE devrede çıkışların durumlarına etkilemiyor, YADA devresinin girişinde de yanmış sigortalar mantıksal sıfır durumu demektir. Fabrikada üretilmiş kod çözücülerin sigortaları yanmamış durumdadır. Sigortalar gereğe göre, onlardan yüksek elektrik ceryani geçirilerek yanıyorlar.



Resim 9.22. PAL prograla nabilir kod çözücünün yapısı

16L8 devresi, 10 sabit giriş, iki sabit çıkış ve giriş ya da çıkış pini olarak programlanabilen 6 pinden oluşan PAL kod çözücüsüdür. PAL 16L8 kod çözücünün girişinde mikroişlemciden adres sinyalleri geliyor, kod çözücünün çıkış sinyalleri ise ROM ya da RAM yongaların CE (Chip Enable) giriş pinlerine gönderiliyor ve bellek bileşenlerin seçimi için kullanılıyor.

Resim 9.23.'te küçük bir Pentium bellek sistemi gösterilmiştir. Bu sistem, 512K toplam kapasiteli sekiz 27512 EPROM bellek yongasından (64Kx8) ve FFF80000H'den FFFFFFFFH'ye kadar adres alanından oluşuyor. Kod çözümlenmesi için iki PAL16L8 kod çözücüsü kullanılmıştır.A19 ile A28 arası bitler ikinci kod çözücünün girişleridir. U2 çıkışının adres bitlerinde bağımlılığı şu denklemle verilmiştir:

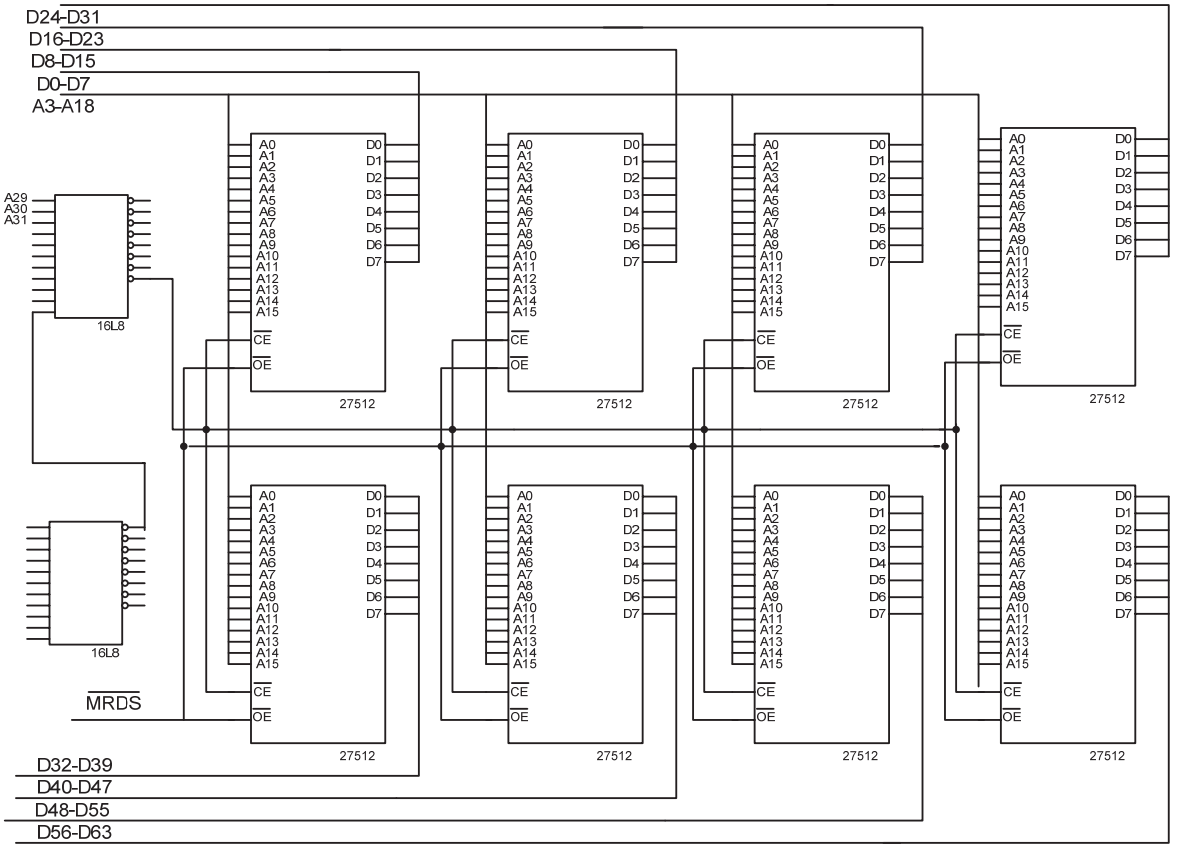
$\overline{U2}$ =A19 · A20 · A21 · A22 · A23 · A24 · A25 · A26 · A27 · A28

Birinci kod çözücünün girişleri A31, A30, A29 adres bitleri ve ikinci kod çözücünün çıkışlarıdır. Birinci kod çözücü CE pini için sinyalı üretiyor. Bu pinin aktifleştirilmesiyle tüm sekiz bellek yongası etkinleştiriliyor. CE çıkışının birinci PAL16L8'in girişlerinden bağımlılığı şu denklemle verilmiştir:

$$\overline{CE} = \overline{U2} \cdot A29 \cdot A30 \cdot A31$$

CE sinyalın aktifleştirilmesi için A31'den A19'dan adres bitlerinin mantıksal sıfır olması gerekiyor. Bellek yongaların seçimi için kullanılan ikinci pin, $\overline{OE}$ pini, $\overline{MRDC}$ kontrol sinyalini aktifleştiriyor ve onun üretilme şekli resim 9.4.'te gösterilmişti. A3 ile A18 arası bitler, seçilmiş bellek modüllerden bellek yerin seçilmesi için kullanılıyor.

Pentium mikroişlemcinin veri veriyolu 64 bitlidir. Tüm sekiz 27512 bellek yonga aktif olduğu zaman, 64 bitli verinin elde edilmesi için her yonga birer bayt verecek. Her bellek yonganın veri çıkışı, veri veriyolundan sekiz hatlı özel grupla bağlıdır: D7 - D0, D8 - D15,D16 - D23, D24 - D31,D32 - D39, D40 - D47, D48 - D55, D56 - D63.



Resim 9.23. Pentium mikroişlemcinin 64 bitli bellekle bağlanması

İntel 865G Yonga Kümesi Tümleşik Devresi

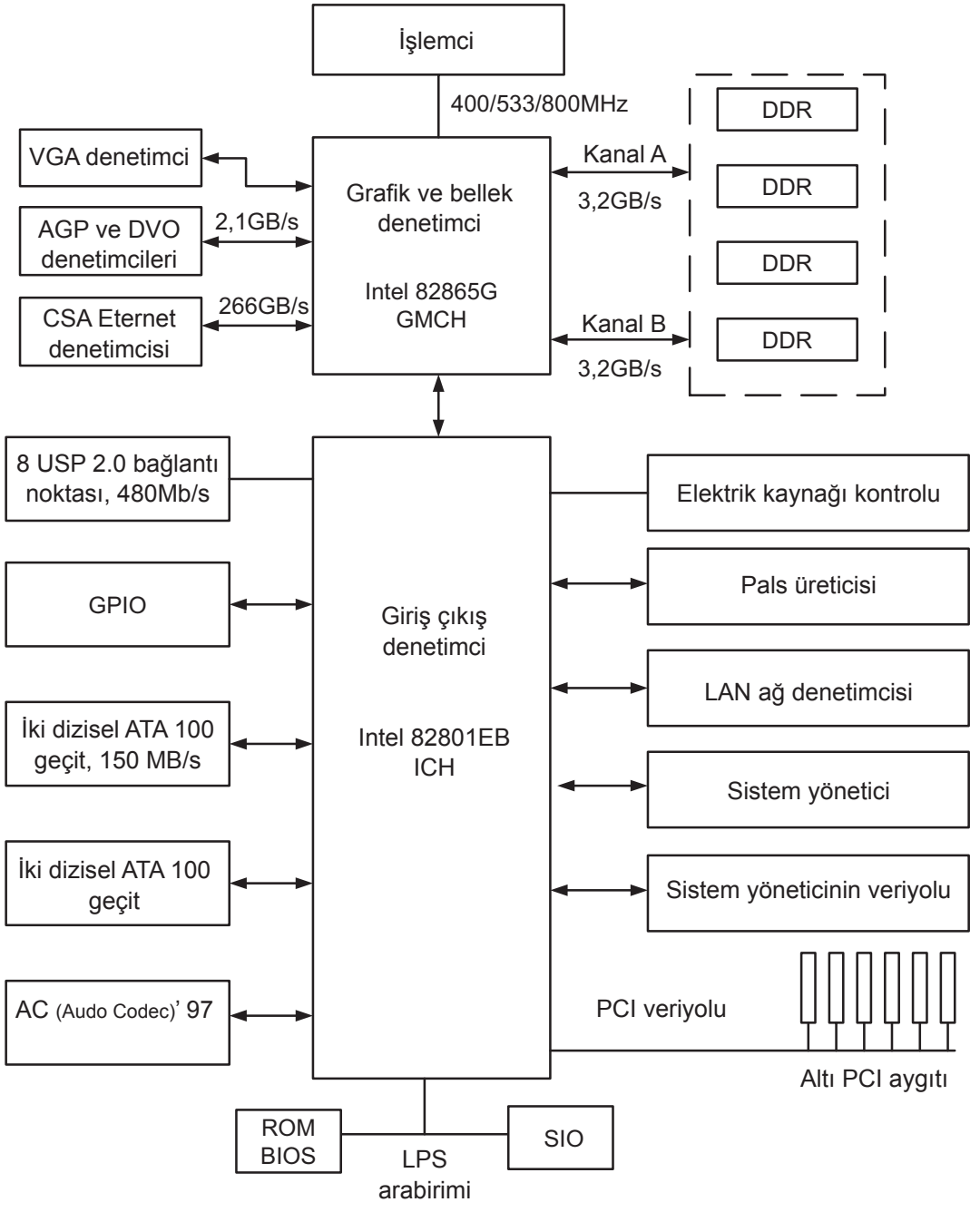
Pentium bilgisayar sistemlerinde, mikroişlemcilerin fazla yürütücü (işletme) işlevleri vardır, **konrol işlevini ise yonga kümesi** (Chip Set) **adıyla bilinen özel tümleşik devreler kümesi gerçekleştiriyor**. İngilizce set sözü küme anlamına, chip sözü ise yonga ya da tümleşik devre anlamına geliyor. Bu tümleşik devreler bellekle ve dış aygıtlarla yönetim için kontrol sinyalleri üretiyor. Yonga kümesi mikroişlemci ve anakartta diğer bileşenler arasında iletişim aracıdır. Hatta, ana kartta yerleştirilmiş yonga kümesi türüne göre, ana kartı da adlandırılıyor. Yonga kümesinin blok diyagramında ayrı birimlerin işlevini incelerken, ana kartının da daha önemli parçalarıyla tanışacağız.

Yonga kümelerini incelemekte İntel 865G yonga kümesini kullanacağız. İntel 865G yonga kümesi, 512 KB ikinci seviye önbelleği ve sistem veriyolun frekansı 400MHz, 533MHz ya da 800MHz olan Pentium 4 mikeoişlemci içeren masa üstü bilgisayar sistemlerinde kullanılıyor. Her yonga kümesi **iki temel bileşenden** oluşuyor: grafik ve bellek denetimcisi (82865G CMCH - Graphics Memory Controller Hub) ve giriş çıkış denetimcisi (82801EB ICH - Input/output Controller Hub). İnigilizce hub sözü merkez, bağlantı göbeği anlamına geliyor. Bunlar sistemde örneğin bir çıkıştan fazla çıkış elde edilen, genişleme, dallanma olduğu yerlerdir. Grafik ve bellek denetimcisi mikroişlemciyle, system belleğiyle, hızlı iletişim birimiyle, giriş - çıkış merkeziyle (hub) ve başka elemanlarla bağlıdır. Resim 9.24.'te 865G yonga kümesinin blok diyagramı gösterilmiştir. Şimdi kısaca 865G yonga kümesinden tüm bileşenlerin işlevlerini ve onların İngilizce terimlerini açıklayacağız.

- **VGA** (Video Graphics Array), ekran ve farklı analog görüntü birim türlerinin bağlanması için çok popüler standarttır. VGA standardı 640x480 piksel (imgecik) çözünürlük, 60 MHz yenileme frekansı ve aynı zamanda 16 rengin gösterilmesini sağlıyor. Daha düşük 320x200 piksel çözünürlükte, 256 renk gösterilebilir. VGA standardı günümüzde SVGA (Super VGA) standardıyla değiştirlmiştir. SVGA standardı 14 - inçli ekranda, 800x600 çözünürlükte ya da 20 - inçli monitörlerde 1200x600 çözünürlükte 16 milyon rengin gösterilmesini sağlıyor. VGA bağlayıcının ana kartında 15 pini vardır.

- **AGP** (Accelerated Graphics Port) grafik kartları ve 3D hızlandırıcıların (grafiksel hızlandırıcılar) takılması için kullanılan bağlantı noktasıdır. AGP bağlantı noktasının sistem belleğine doğrudan erişimi var. AGP veriyolu 32 - bitlidir, çalışma frekansı 66MHz'tir ve böylece 2.1GBps geçiş kapsamı veriyor. AGP bağlantı noktaların piyasaya çıkmalarında once, video kartların takılması için iki kat daha düşük geçiş kapsamı olan PCI bağlantı noktaları kullanılıyormuş.

• DVO (Digital Video Output) bağlantı noktaları tekkanallı ve çift kanallı video aygıtarın daha yüksek çözünürlüğü ve yenileme frekansı (refresh) olan aygıtlarla bağlanması için kullanılıyor, örneğin LCD ekranlar.



Resim 9.24. İntel 865G yonga kümesinin blok diyagramı

• **CSA** (Communication Streaming Architecture), grafik ve bellek denetimcinin (GMCH) Eternet denetimciyle bağlanması için özel olarak tasarlanmış kavramdır. Eternet bilgisayar ağların oluşması için bir protokoldur. Hızlı iletişim

denetimcisi (CSA) 66MHz çalışma frekansı sırasında saniyede 266 milyon transfer aktarımı sağlıyor.

- 865G yönga kümesinin sistem belleğiyle bağlanması için 3.2GB/s geşiç kapsamlı iki kanal kullanılıyor. Belleğin kapasitesi 4GB'tır, dört bankayla ayrılmıştır ve DDR SDRAM yongaları kullanılmıştır. DDR (Data Dual Rate) kısaltması, bir dijit palsı sırasında iki verinin aktarımı demektir.

82801EB giriş çıkış denetimcisi, içinde şu elemanları bağlıyor:

- **ATA 100 denetimci** – sabit disk belleğinin bağlanması için, paralel ATA geçidi 40 - pinli IDE (Integrated Drive Electronic) geçidini değiştirmiştir. IDE kablosu sadece birkaç santimetre uzunluğundadır ve bu özellik ATA bağlanma kablolarına kıyasen başka bir dezavantajdır.
- İki dizisel ATA denetimcisi – dizisel ATA geçidinin, paralel geçidinden daha yüksek frekansı var, ancak çalışma hışırtısı var. ATA geçidi yedi pinlidir ve 150MB/s geçiş kapsamı vardır.
- EHCI (Enhanced Host Controllor Interface), 480MB/s geçiş kapsamlı USB 2.0 geçitlerin denetimcisidir.
- UHCI (Universal Host Controllor Interface) 12MB/s geçiş kapsamlı USB 1,1 ve 1,0 geçitlerin denetimcisidir. 85G yonga kümesinin USB1,1 geçidi yok, sekiz USB 2.0 geçidi var ve onlar USB 1,1 geçidiyle uyumludur.
- Eski ve yavaş ISA veriyolunu değiştiren LPC (Low Pin Count) veriyolunun denetimcisi. LPC veriyolu, içinde BIOS programını içeren EEPROM yonganın ve dizisel, parallel geçitleri, fare ve kalvyenin gibi basit giriş çıkış aygıtların (SIO - Simple Input Output) bağlanması için kullanılıyor.
- **PCI** (Peripheral Component Interconnect) **veriyolun denetimcisi.** Örneğin, ses kartı PCI yuvası için aygıttır.
- AC'97 (Audio Codec) denetimcisi aynı zamanda ses sinyallerin analog dijital dönüştürmesi için kullanılıyor.
- Bilgisayar ağlarla (LAN - Local Area Network) **bağlanma dentimcisi.**
- GPIO (General Purpose Input Output) denetimcisi 8 pinli GPIO geçitlerini kontrol ediyor ve LED diyodlar ya da anahtarlar gibi basit aygıtların bağlanması için çok uygundur. GPIO geçitleri giriş veya çıkış geçitleri olarak programlanabilir.

865G yonga kümesinin daha önemli birimlerini tanıdıktan sonra, bu tümleşik devrenin daha karakteristik, daha genel pinlerinin işlevlerini açıklamaya deneyeceğiz. Yonga kümesi pinlerinin büyük kısmı Pentium mikroişlemcinin pinleriyle benzer işlevleri olduğunu göreceğiz. Bundan dolayı bazı pinleri açıklamayacağız çünkü onları 9.2.Pentium 1 Mikroişlemcinin Pin Diyagramı ve 9.5.Pentium 2 Mikroişlemci derslerinde inceledik. Resim 9.25.'de 865G tümleşik devrenin, işlevlerine göre ayrılmış pin grupları gösterilmiştir.

Grafik ve bellek denetimcisi (82865G) ve mikroişlemci arasında iletişim için şu sinyaller kullanılıyor:

HD → (Host Data) - Denetimcinin veri veriyolu
HA → (Host Address) - Denetimcinin adres veriyolu
HADSB → (Host Address Strobe) - adres gönderme anons sinyalı
DBSY → (Data Bus Busy) - Veri veriyolu aktarmayı daha tamamlamamış
DINV → (Dynamic Bus Inversion) - 64 - bitli veriden hangi sözün aktarılacağını belirliyor
DRDY → (Data Ready) - Veri gereken hedefe varmış

CPURESET → Mikroişlemcinin sıfırlandırılması
HIT,HITM → Önbelleğe erişim için pinler
HTRDY → (Host Target Ready) - Aygıt verileri kabul etmek için hazırdır
PROCHOT → Mikroişlemcinin sıcaklığın yükselmesi
AP → (Address Parity) - adresin çift değer kontrolu
DP → (Data Parity) - Verileri çift değer kontrolu
IERR → (Internal Error) - Sistemde iç hata
BINIT → (Bus Initialization) –Veriyolların sıfırlandırılması

DDR SDRAM sistem bellekle iletişim için şu sinyaller kullanılıyor:

SCMDCLOK → (Differential DDR Clock) - Adres ve veri sinyalının kabul edilmesi için bu sinyalın yükselen kenarı gerekiyor.

SCS → (Chip select) - Bir SDRAM yongası seçiliyor
SMAA → (Memory Address) - Bellek adresi
SRAS, SCAS → (Row Address Strobe, Column Address Strobe) - adres veriyolun çoklayıcısını kontrol ediyorlar

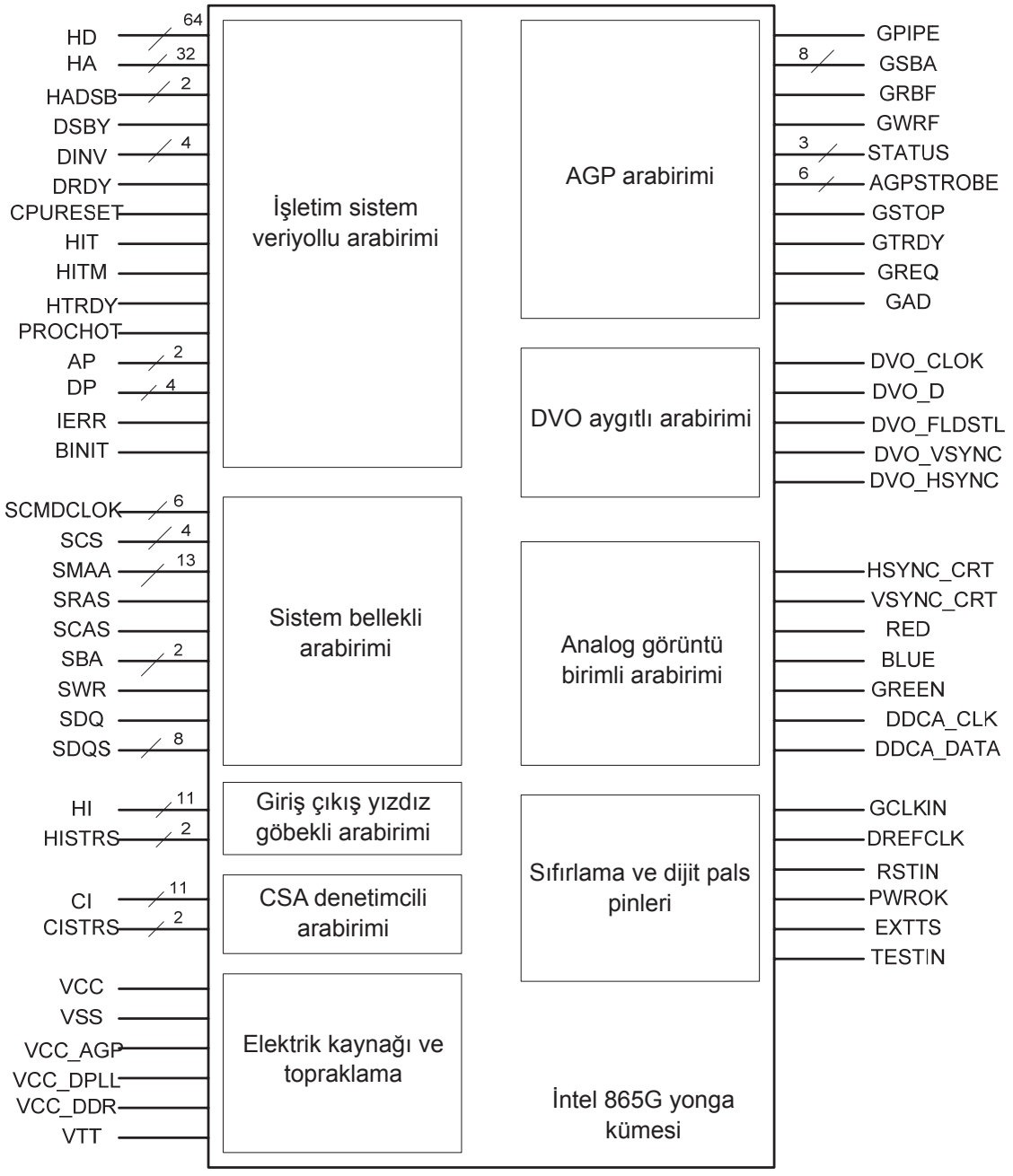
SBA → (Select Bank) - Bellek bankanın seçimi
SWE → (Write Enable) - Okuma için etkinleştirme
SDQ → (Data Lines) - Veri pinleri
SDQS → (Data line Strobe) - 64 - bitli veriden her bayt için anons

Yonga kümesinin, hızlı iletişim denetimcisi ve giriş çıkış yıldız göbeğiyle (habla) iletişim için anonslu düzen kullanılıyor ve bu arada sadece veri sinyalleri ve anons sinyalleri aktarılıyor. Giriş çıkış yıldız göbeğiyle iletişim sinyalleri şunlardır:

HI ⟶ Veri sinyalleri
HISTRS ⟶ Veri atarımı için anons (Strobe) sinyalleri

Hızlı iletişim denetimciyle (CSA) iletişim sinyalleri şunlardır:

CI ⟶ Veri sinyalleri
CISTRS ⟶ Veri aktarımı için anons (Strobe) sinyalleri



Resim 9.25. İntel 865G yonga kümesinin pin diyagramı

AGP denetimciyle iletişim sinyalleri birkaç gruba ayrılabilir: adres,kontrol, durum ve anons sinyalleri. Aşağıda onların işlevleri verilmiştir:

GPIPE → (Pipe read) - pipeline kavramında very aktarımı anons ediliyor
GSBA → (Sideband Address) - Yeni ek adres hatları ekleniyor

GAD → (Address Data) - Adres veri sinyalleri
GRBF → (Read Buffer Full) - Okuma başlatıyor
GWRB → (Write Buffer Full) - yazma başlatıyor
Status → Durum bitleri aktarma durumunu tanımlıyorlar
Strobe → Dört dijit palsı süren veri aktarımının zamanlamasını gerçekleştirmek için kullanılan sinyeller

GSTP → (AGP Stop) - AGP denetimcisi aktarımı durduruyor
GTRDY → (AGP Target Ready) - Aygıt verileri kabul etmek için hazırdır
GREQ → (AGP Request) - AGP veriyolları kullanmak için arama gönderiyor

Yonga kümesi ve DVO geşitlerin denetimcisi arasında iletişim sinyalleri şunlardır:

DVO_CLOK → DVO denetimcisi için dijit palsı
DVO_D → Her dijit palsında 12 - bitli pikselin aktarımı için veri veriyolu
DVO_VSYNC → Dikey senkronizasyon
DVO_HSYNC → Yatay senkronizasyon
DVO_FLDSTL → (TV Field and Flat Panel Stall Signal) - dijital video aygıtın seçimi için sinyal

Analog video aygıtların kullanımı sırasında şu kontrol sinyalleri kullanılıyor:

HSYNC, VSYNC → Dikey ve yatay senkronizasyon
RED, GREEN, BLUE → Renkler için analog sinyaller
DDCA_CLK → Analog video aygıtlar denetimcisi için dijit pals

DDCA_DATA → Veri sinyalleri

Diğer daha genel kontrol sinyalleri şunlardır:
GCLKIN → (AGP Clock) 68MHz frekanslı dijit palsı
DREFCLK → (Display Clock Input) 48MHz frekanslı dijit palsı

RSTIN → (Reset In) Yonga kümesinin sıfırlanması için giriş pini
TESTIN → Yonga kümesi test moduna giriyor
VCC_AGP → AGP geçidin elektrik kaynağı
VCC_DDR → Sistem belleğin elektrik kaynağı
VTT → Sistem veriyolun elektrik kaynağı

İşlevleri ve çalışma kompleksliğine göre, Pentium bilgisayar sistemlerin oluşturulması için kullanılan tümleşik devreler, mikroişlemcilere çok benzer olduğuna sonucuna varabiliriz.

Sonuçlar:

Pentium 1 mikroişlemcide birinci seviye önbelleği L1, 8KB'lık iki önbelleğe ayrılmıştır. Birinci önbellek yönerge önbelleğidir, ikinci önbellek ve veri önbelleğidir.

Pentium mikroişlemcilerin süper sayısal yapısı, onlarda üç işlem biriminin olmasından kaynaklanıyor. Bir işlem birimi kayan noktalı sayılar ve böylelikle iki işlem birimi tam sayılar için U pipe, V pipe). Buna gore parallel olarak üç farklı yönerge birden çalıştırılabilir.

Pentium mikroişlemcinin belleği 4GB büyüklüğündedir ve 512MB'lık 8 bellek bankasına ayrılıyor. Her bellek bankasına ayrıdan erişim, 8, 16, 32 ve 64 - bitli verilere ulaşılmasını sağlıyor. Bellek alanının adreslenmesi için 64 - bitli adres veriyolu kullanılıyor.

Pentium 1 mikroişlemcide, bellekten okuma akış (burst) döngüsü aracılığıyla gerçekleşiyor. Bir akış döngüsüyle, beş dijit paslı süresi içinde dört 64 - bitli veri aktarılabilir.

$\overline{BRDY}$ (Burst Ready) pini, bellek ya da dış aygıt veri veriyolundan veri aldığı ya da verdiği zaman aktifleştiriliyor.

$\overline{A20}$ pini, mikroişlemcinin gerçek çalışma modunda çalışması gerektiği zaman akifleştiriliyor.

Adreslerin çift değer kontrolu için iki pin kullanılıyor, AP ve $\overline{APCHK}$ pinleri. Çift değer kontrolu, bellekten ya da bazı dış aygıtından okunan veri bitlerine de yapılıyor. Her bellek bankası için birer çift değer kontrol pini vardır DP7 - DP0 (data parity).

Önbelleğin kontrolu için özel pinler vardır. $\overline{KEN}$ (cache enable) pini iç önbelleği etkinleştiriyor. WB/$\overline{WT}$ (write back/write - through) pini önbellek için işlem seçiyor. $\overline{FLUSH}$ pinin aktifleştirilmesiyle iç önbellek iptal oluyor.

Pentium mikroişlemcileri üç modda çalışabiliyor: gerçek çalışma modu, korumalı çalışma modu ve sanal çalışma modu. Korumalı çalışma modunda adres alanı 4GB'a (2^{32}) kadar artıyor, sanal çalışma modunda ise 64TB (2^{46}) değerinde çok büyük adres alanı elde ediyoruz.

Korumalı çalışma modunda bölüt yazmacı seçici (selektör) içeriyor. Seçici 8192 açıklayıcıdan birinin seçilmesini sağlıyor. Açıklayıcı aracılığıyla, bellekten istenen bölüt bulunuyor. İstenen bölüt bulunduktan sonra, istenen verinin bulunmasına geçiliyor. Adresin daha değerli onaltı bit seçicinin değerini veriyor, daha değersiz on altı bit kaydırmanın değerini belirliyor.

Korumalı sanal çalışma modunda sadece bir dosya var ve onun başlangıç adresi CR3 kontrol yazmacında bulunuyor. Dosya 1024 konumdan oluşuyor. Bu konumlardan hangisinin seçileceği, sanal adresten 22 ile 31 yerlerin arasında bulunan bitlerin değerine bağlıdır. Dosyada seçilen konum, 1024 sayfa tablosundan birine yönlendiriyor. Sayafalar tablosunda hangi yerin seçileceği, sanal adresten 12 ile 21 yerin arasında bulunan bitlerin değerine bağlıdır. Tablo sayfasında seçilen yer bir sanal sayfanın başlangıcına yönlendiriyor. Sonunda, sayfadan hangi yerin seçileceği, kaydırmaya bağlıdır, yani sanal adresin 0 ile 11 pozisyonu arasındaki bitlere bağlıdır.

Pentium Pro mikroişlemcisi yapısı açısından, öndan önce piyasaya sunulan tüm mikroişlemcilerden farklıdır. Pentium Pro mikroişlemcisinde yenilikler: farklı seviyede iki önbellek, kodları çözülmüş yönergeleri koruyan yönerge havuzu, çalıştırılması gereken yönergelerin sıralama birimi ve çalıştırılan yönergelerin çekme birimidir.

Pentium 2 mikroişlemci, ana kartında yerleşmemiş olmasından farklıdır. Pentium 2 mikroişlemci özel olarak tasarlanmış yuvada (slot - yuva) yerleşiyor. Birinci seviye önbelleğin 32 KB kapasitesi var, ikinci seviye önbelleği ise mikroişlemcinin aynı tümleşik devresinde bulunmuyor, ancak aynı pakette birinci seviye önbelleğine çok yakın yerde bulunuyor.

Pentium 3 ve Pentium 4 mikroişlemciler yapıları açısından benzerdir. Yerleşme birimi, mikro yönerge işletim sitemine gelince her şeyin hazıl olması için kaynakların hazırlığını gerçekleştiriyor. Yer değiştirme birimi programcının görebildiği 32 yazmacı (EAX, EBX, ECX, EDX, EDI, ESI, EBP ve ESP), 256 iç yazmaçatn birine göre yeniden adlandırıyor.Onlardan 128 yazmaç tam sayılar için, 128 yazmaç ise kayan noktalı sayılar için kullanılıyor. Sıralama birimi mikro yönergeleri, kodları çözümleme sırasına göre sıralıyor. Mikroişlemcinin her bölümünün devamlı çalışması için, izler önbelleğinde ve yerleştirme biriminde yönergeler karışıktır.

Pentium 4 mikroişlemcinin 6 işletim birimi var. Onlardan iki birim tam sayılarla çalışmak için, iki birim kayan noktalı sayılarla çalışmak için ve iki birim RAM belleğinden veri aktarımı için.

Pentim D mikroişlemcisi iki çekirdekli mikroişlemcidir, ancak iki merkezi mikro işlem birimi bir yongada bulunmuyor. Onlar birbirine çok yakın aynı pakette yerlemiş ayrı yongalarda yer bulunuyor. Pakette, iki ikiçekirdekli mikroişlemci dışında, ikinci seviye önbelleği ve veriyolu arabirim birimi yer alıyor. Bu mikroişlemcilerin yürütücü gücü bir (tek) çekirdekli mikroişlemcilerden belki daha büyüktür, ancak ısınma çok büyüktür. Bu eksikliğe Pentium Dual Core mikroişlemcide çözüm bulunmuştur. Pentium Dual Coe mikroişlemcide tüm birimler aynı bir tümleşik devrede yerleştirilmiştir.

Her yonga kümesi iki temel bileşenden oluşuyor: grafik ve bellek denetimciden (82865G CMCH - Graphics Memory Controller Hub) ve giriş çıkış denetimciden (82801EB ICH - Input/output Controller Hub).

82801EB giriş çıkış denetimci, şu elemanları içeriyor: sabit disk bellekle bağlanma denetimcisi, USB geçitlerin ATA denetimcileri, ISA ve PCI veriyolların denetimcisi, LAN ağlarla bağlanma denetimcisi, LED diyodlarla ya da anahtarlarla bağlanmak için GPIO denetimcisi.

Sorular ve Ödevler:

1. Dallanmaları öngörme mantığı, mikroişlemcinin çalışma hızını nasıl iyileştiriyor?
2. Süper sayısal yapı teriminin ne demek olduğunu açıkla.
3. Pentium mikroişleminde hangi pinler çift değer kontrolu için kullanlıyorlar?
4. Bellekten mikroişlemciye hızlı veri aktarımı sırasında hangi sinyaller aktifleştiriliyor?
5. Bellek ve dış aygıtların için kontrol sinyellerin üretimi için Pentium mikroişlemcilerde nasıl devre kullanılıyor?
6. Önbelleğin kontrol pinlerini say!
7. Sayısal yardımcı işlemcinin kontrolu için, Pentium 2 mikroişlemciden hangi pinler kullanılıyor?

8. Pentium 1 mikroişlemcinin, adres alanın büyüklüğü ne kadardır eğer şu modlarda çalışıyorsa:
   A) gerçek modu
   B) korumalı modu
   C) sanal modu

9. Mikroişlemci koruma çalışma modunda çalıştığı zaman bellekten bölüt yazmacının seçimi için uygulanan sürecini açıkla!

10. Açıklayıcılar ne için kullanılıyor?

11. Bir bölütün sınırı 01FFFH değerine eşitse ve G biti bire eşitse, bölütün büyüklüğü ne kadar olacaktır?

12. 80386 mikroişlecinin sanal adresi kaç bölümden oluşuyor? Bu bölümlerin her biri ne için kullanılıyor?

13. 80386 mikroişlemcide, 01405FFFH sanala adresi verilmişse, şunları belirle:
   • sayfalar tablosunun sıra numarasını
   • sayfanın sıra numarasını
   • Sayfanın başlangıcından istenen yere kadar mesafeyi

14. 80386 mikroişlemci sanal moduna çalıştığı zaman 64TB'lık adres alanının nasıl elde edildiğini açıkla?

15. Pentium 1 mikroişlemcide bellek kapasitesi ne kadardır?

16. Pentium Pro mikroişlemcide bellek kapasitesi ne kadardır?

17. Pentium 1 mikroişlemcide veri veriyolun genişliği ne kadardır?

18. Pentium mikroişlemcide AD pini ne için kullanılıyor?

19. Pentium 1 mikroişlemcide çalışma frekansının değeri nekadardır?

20. Pentium Pro mikroişlemcinin kaç tür önbelleği var?

21. Pentium Pro mikroişlemice, yönergenin yönerge havuzuna nasıl girildiğini ve çıkarıldığını açıkla!

22. Pentium 1 mikroişlemcide bellek nasıl örgütlenmiştir?

23. Pentium 2 mikroişlemci için adres ve veri veriyolunun genişliği ne kadardır?

24. Pentium 2 mikroişlemcide iki önemli yenilik hangileridir?

25. Pentium 2 mikroişlemcinin üç en popüler türevlerini açıkla!

26. İki çekirdekli mikroişlemcinin iç yapısını açıkla!

27. Pentium 4 mikroişlemcide birinci seviye yönerge önbelleğinin nerede bulunduğunu açıkla?

28. Yerleştirme ve yeniden adlandırma biriminin işlevini açıkla!

29. Pentium 3 ve Pentium 4 mikroişlemcilerin kaçar işletim birimi var?

30. Pentium D ve İntel Dual Core mikroişlemcileri arasında fark nedir?

31. Bilgisayar sisteminin çalışması için yonga kümenin nasıl işlevi olduğunu açıkla!

32. Pentium bilgisayar sistemlerinde İntel 865G yonga kümesinin hangi geçitleri kontrol ettiğini say!

Kaynakça

- Structured Computer Organization - Andrew S. Tannenbaum - 1998
- Fundamentals of Computer Organization and Design - Sivarama P. Dandamudi - 2002
- The Intel Microprocessors 8086/8088, 80186/80188, 80286, 80386, 80486,Pentium and Pentium Pro Processor Architecture, Programming and Inter - facing - 2003
- The Essentials of Computer Organization and Architecture - Linda Null and Julia Lobur - 2003
- PIC in Practice - D.W.Smith - 2002
- Intel 8085 processor - Data Sheet
- Intel 8086 processor - Data Sheet
- Intel 80386 processor - Data Sheet
- Intel 865G Chipset Family - Data Sheet
- Microchip PIC16f84 - Data Sheet