

YILDIZ TEKNİK ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

Mpf-1/88 İle Frekan. Uyg.

Yüksek Lisans Tezi

Faruk Özgüven

1988

Mpf-1/88 İle Frekan. Uyg.

Ref
EHM
237
1987

R152
85

YILDIZ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MPF-1/88 İLE FREKANSMETRE UYGULAMASI

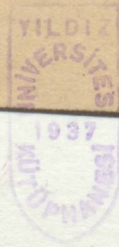
YÜKSEK LİSANS TEZİ

Ö. Faruk ÖZGÜVEN
ELEKTRONİK VE HABERLEŞME MÜH.

İSTANBUL 1987

YILDIZ ÜNİVERSİTESİ
GENEL KİTAPLIĞI

Kot : R 152
Alındığı Yer : Fen Bil. Enst. 85
Tarih : 9.5.1991
Fatura :
Fiatı : 8500 TL
Ayniyat No : 1/3
Kayıt No : 47627
UDC : 621.3 378.242
Ek :





MPF-I/88 İLE FREKANSMETRE UYGULAMASI

YÜKSEK LİSANS TEZİ

Ö. Faruk ÖZGÜVEN

ELEKTRONİK VE HABERLEŞME MÜH.



GİRİŞ

Günümüzde mikroişleyici kullanılan sistemler hızlı bir şekilde gelişmektedir. Dolayısıyla daha yetenekli mikroişleyicilere ihtiyaç duyulmaktadır. Tüm devre yapım teknolojisi de buna paralel olarak gelişmiştir. Böylece günümüzde yaygın olarak kullanılan güçlü yazılım ve donanım özelliklerine sahip gelişmiş mikroişleyiciler yapılmıştır. Bu tezde kullanılan MPF-I/88, 8088 yazılım ve donanım geliştirme setinin 16-bitlik mikroişleyicisi IBM ve IBM uyumlu kişisel bilgisayarlar da kullanılan gelişmiş bir mikroişleyicidir. Burada 8088 ile 8086 arasındaki farkı belirtmede fayda vardır. 8088 mikroişleyicisi komutlar açısından 8086'ya benzerdir. Fakat 8088'in adres yolunun en az ağırlıklı sekiz biti ile "Multiplex" çalışan 8 bit veri yolu, 8086'in adres yolunun en az ağırlıklı 16-biti ile "Multiplex" olarak çalışan 16-bit veri yolu vardır.

MPF-I/88'de yazılım ve donanım geliştirme setinde 8088-16 bit CPU, 16K Bayt ROM, 24K Bayta kadar genişleyebilir, 6KBayt RAM, 2x20 karakterlik alfanümerik LCD gösterge ve standart ASCII klavyesi vardır. 8088 mikroişleyicisinin yazılım özelliklerinin tam olarak öğrenilebilmesi için bir "line assembler" programı vardır. 8088 "assembler" dilini rahatlıkla yazabilmek için MPF-I/88 yeterli komutlara sahiptir.

8088 "assembler"dilini öğrenebilmek için IBM ve IBM uyumlu bilgisayarlarda yazılabilen birçok programlar yapılmıştır. Bu programların bazı yönleri MPF-I/88 monitör programından üstün olabilir. Fakat ekonomik yönden, bilgisayar boyutu açısından ve dışarıda kontrol edilecek bir sistem yönünden MPF-I/88'in üstünlükleri vardır.

Bu geliştirme setinin istenildiği zaman kapasitesini arttırma imkanı vardır. Gerekli malzemeleri seçerek RAM kapasitesi 6KBayttan

24KBayta kadar genişleyebilir. Ayrıca "EXT-BUS" kısmına kumanda edilecek herhangi bir sistem takılabilir.

Tez çalışmalarım sırasında program algoritmasındaki desteği nedeniyle, Ar.Gör.Yük.Müh.Tuncay UZUN'a, İngilizce çeviri kısmında büyük yardımı olan Müh.Ülker PEKGÜÇ'e ve Lisans, Lisans üstü öğrenimim boyunca öğrettikleri ve tez çalışmalarımıdaki yol göstericiliği nedeniyle Doç.Dr.Sezgin ALSAN'a teşekkür ederim.

Ö.Faruk ÖZGÜVEN

EYLÜL-1987

İÇİNDEKİLER :

- MPF-I/88 SİSTEM SIFIRLAMA.

- Sisteme ilk başlama.
- RAM Kontrol.
- ROM toplam kontrolu.
- Genişletme kartı kontrolu.
- Çalışırken yeniden başlama (Warm start)

- I/O PROGRAMLAMA

- I/O bellek düzenlemesi.
- I/O port adresleri.
- Klavye.
- Yazıcı sürücü.
- Programlama göstergisi.

- SES ARABİRİMİ (AUDIO İNTERFACE)

- TEYPE KAYDETME ŞEKLİ.

- MPF I/88 DEVRE TANIMLANMASI

. 1- CPU ve destekleyici devreleri.

- Sistem zamanlama devresi
- Bekle lojigi.
- Sistem sıfırlama devresi.
- Kesme isteği devresi.
- "Bus buffer" (Yol yastığı)
- Bellek ve I/O donanım çözücüleri.

- . 2- Sistem belleđi.
- . 3- Giriş ve çıkış arabirim devreleri.
- AUTO-ROM ve HARİCİ KOMUTLAR.
- MONİTOR KOMUTLARININ AÇIKLANMASI.
 - 1- "Line Assembler" ve Disassembler.
 - "Line Assembler"a başlamak.
 - "Disassembler"a başlamak.
 - 2- Programı yürütme komutları.
 - Programı yürütme.
 - Programı adım adım yürütme.
 - 3- "Debugging"
 - Programı kırma yerini yerleştirme ve gösterme.
 - Program kırma yerlerini iptal etme.
 - Yazmaçların içeriklerini deđiştirme/inceleme.
 - 4- Kaydedilen belleđi deđiştirme veya inceleme.
 - Belleđe veriyi yerleştirme (INSERT)
 - Bellekteki veriyi silme.
 - Bellekteki harfler dizisini bulma.
 - CS ve IP yazmaçlarına yeni deđer taşıma.
 - Bellek dizisinin içeriklerini bellekte belirtilen alana kopya etme.
 - 5- Çeşitli I/O komutları.
 - Ekranda gösterilenlerin hızını ayarlama.
 - Porttan verinin bir baytını gösterme.
 - Çıkış portuna veri baytını gönderme.
 - Sistem "konsulu" olarak dış terminali tespit etme.

- Teype bilgi yazma.

- Teypten bilgi okuma.

7- Kontrol harfleri.

- FREKANSMETRE

- Frekansmetrenin çalışma prensibi.

- Frekansmetrenin yazılımı.

- CPU BACAĞ İŞLEVLERİNİN AÇIKLANMASI

- SEGMENTLER

- YAZMAÇLAR

- Dört genel amaçlı yazmaç : AX, BX, CX ve DX

- İki gösterici yazmaç : SP ve BP

- İki gösterge yazmaç : SI ve DI

- Dört segment yazmaç : CS, DS, SS ve ES

- Komut gösterici yazmaç : IP

- Bayraklar yazmaç (Flag register)

ÖZET

Bu çalışmada amaç MPF-I/88 mikroişleyici sistemi ile yapılabilecek uygulamalara bir örnek vermektir. Cihaz ilk açıldığında bir ilk başlama işlemini (Cold Start) yerine getirir. Eğer cihaz çalışırken sıfırlama sinyali gelirse çalışırken yeniden başlama (Warm Start) işlemini yerine getirir.

Sisteme giriş ve çıkış yapabilmek için çeşitli partları vardır. Belirli adreslere yerleştirilen giriş ve çıkış birimleri LCD gösterge, yazıcı, klavye ve teyp girişi / çıkışıdır. MPF-I/88'in teyp kaydetme şekli IBM-PC'e benzerdir. Sistemde giriş/çıkış partlarını, bellek düzenini ve mikroişleyicinin çalışması için gerekli sinyali sağlayan CPU destekleyici lojikleri vardır. Sistem zamanlama devresi CPU için gerekli saat sinyalini üretir ve bu saat frekansını bölerek mikroişleyicinin çalışacağı frekansa getirir. Bekle lojigi sisteme giriş veya çıkış yapılacağı zaman CPU'yu bekletir. Bütün yazmaçları ve sistemdeki "latch"leri sıfırlamak için sistem sıfırlama devresi vardır. Kesme isteği lojigi CPU'nun yaptığı işlemi bırakıp istenilen işlemi yerine getirmesi için gerekli sinyali üretir. 8088'de adres ve veri yolunun "multiplex" olarak çalışması nedeniyle "Bus Buffer" yol yaptığı kullanılır. Bellek çözücüleri iki ayrı bölümde incelenir. Bunlar RAM ve ROM çözücüleridir. Giriş ve çıkış part adreslerininide giriş/çıkış arabirim lojigi belirler.

Sistemi genişletebilmek için genişletme kartı kullanılır. Harici komut oluşturmak ve sistemi istenilen adresten başlatmak için "AUTO-ROM" kullanılır.

MPF-I/88'de "line Assembler" yazabilmek için moitor programında oluşturulan birçok komutları vardır. Tezde bu komutlar teorik olarak anlatılmıştır. Eğer bu komutların örnekleri görülmek istenirse MPF-I/88'in kullanıcı kitabına bakılabilir.

8088'in komutlarını bir programda incelemek için frekansmetre programı esnek olarak verilmiştir. Bu frekansmetre birim zamanda gelen darbelerin sayısını sayar. Bu birim zaman 1 sn. olarak alınmıştır.

Bilindiği gibi bu sistemde 8088 mikroişleyicisi kullanılmıştır. 8088 mikroişleyicisinin 14 yazmaçı vardır. Aritmetik ve bazı diğer işlemlerde bir değeri kontrol edebilmek için FL yazmaçı kullanılır. CPU'nun 20 tane adres yolu vardır. Dolayısıyla 1MBaytlık adresleme yapılabilir. 16 bitlik yazmaç 1M Baytı adresleyemez. Bu yüzden segment yazmaçlarına ihtiyaç duyulmuştur. Bazı dizi işlemlerinde ise SI ve DI yazmaçları kullanılır.

SUMMARY SİSTEM SİFİRLAMA (RESET) :

The reason of this investigation is to give application as an example which can be done with MPF-I/88 microprocessor system. Device performs cold start when power is first applied. It executes warm start, if a reset signal comes while the device is on.

There are various ports at system to input to system and to output from it. Input and output units to be put into definite addresses are LCD display, printer, keyboard and tape input/output. Recording method of MPF-I/88 is similar to IBM-PC. There are logics supporting CPU that provide necessary signals in order to operate the micro processor, and memory map and I/O ports.

Memory decoders are investigated with two manner. They are RAM and ROM decoders. Input/Output interface logic determines the addresses of input/output ports. An expanding card is used to expand the system. AUTO-ROM is used to constitute external instruction and to start the system from specified address.

There are great number of instructions to be constituted at monitor program to write "line assembler" to MPF-I/88. These instructions are explained theoretically here. If you want to see other examples of these instructions, you can apply to user's manual book.

Frequency-meter program is given as an example to 8088's instructions. This frequency-meter counts per second.

As you know 8088 microprocessor is used at this system. 8088 CPU has 14 registers. There are 20 address buses at CPU. Therefore, addressing for 1M byte can be done, but this is impossible for the SI and DI registers are used at some string processes.

MPF-I/88 SİSTEM SIFIRLAMA (RESET) :

Aşağıda sistem resetlenmesi esnasında yapılan işlerin kısa bir açıklaması vardır. MPF-I/88 güç verildiğinde bir sisteme ilk başlama (cold start) işlemini üretir. Reset (sıfırlama) anahtarına basıldığında bir sistem çalışırken başlama işlemini (warm start) işlemini üretir.

Bir ilk başlama arasında, sistem aşağıdaki işleri yapar.

- 1- "MPF-I/88 üzerindeki mesaj işaretini gösterme ve monitor programının tercümesi.
- 2- Bir, RAM test üretme.
- 3- Bir ROM toplam kontrol üretme.

Sisteme İlk Başlama :

Güç ilk defa olarak sisteme uygulandığında, CPU monitor programının başlamasını fiziksel adres FFFF0H'den yürütmeye başlayacaktır. Bu 20-bit gerçek adres, aşağıdaki şekilde segment adres FFFFH'i ve efektif adres 0000H'i eklemekle hesaplatılır.

$$\begin{array}{rcl} \text{FFFFX} & \text{----} & \text{Segment adres kodu} \\ + & \text{---} & \text{0000} \text{ ---- Efektif adres} \\ \hline \text{FFFF0} & \text{----} & \text{Gerçek adres} \end{array}$$

Not edilmelidirki, segment adres gerçek adres hesaplama için ilk önce kaydırılmış sol 4 bitdir. Segment adres sola kaydırılırken, sıfırlar 20-bit bir segment adresi şekillendirmek için en az ağırlıklı 4 bitde kaydırılır.

Yalnızca 16 bayt'lık belleğin FFFF0H ve FFFFFH bellek bölgeleri arasında mevcut olmasından (bir büyük program için yeterli değildir). F00 3'a atlama komutu monitor programın ilk komutu olarak yürütülür. Sonra monitor programı, bir ilk başlama yada çalışırken başlamanın üretilip üretilmediğini temin edecektir.

RAM Test :

Bir ilk başlama durumunda, RAM test ilk önce yapılacaktır. RAM test programı 5555 ve AAAA iki sözcük örneğiyle her bir bellek sözcüğüne yazacak ve içerikleri geri okuyacaktır. Eğer bellek sözcüğünün içerikleri bu bellek sözcüğündeki yazılanlara uygunsa, RAM kontrol alt programı diğer sözcüğü kontrol etmek için devam edecektir. Eğer bir uygunsuzluk bulunursa bu durumda RAM kötü bellek gözlerini içerebilir ve program bir hata mesajı gösterecektir.

Not edilmelidir ki, RAM test programı bitişik bellek alanlarına kontrol eder. Eğer sistem RAM bitişik bellek alanında yerleşmek için gösterilmiyorsa, bu durumda yalnızca en küçük değerdeki diğer bellek sırası kontrol edilecektir.

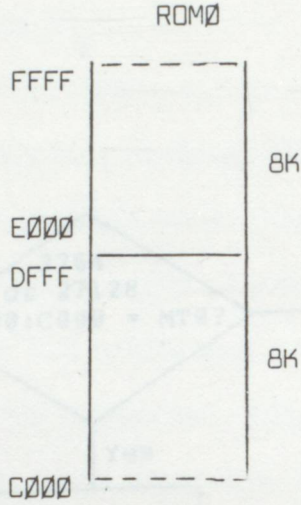
ROM Toplam Kontrol Testi :

ROM toplam kontrol testi için, bellek sözcüklerinin içerikleri bir bellek sözcüğündeki depolama bir toplam kontrolü şekillemek için birlikte eklenirler. Eğer bellek sözcüğünün en küçük değerli baytının değeri sıfırsa, bu durumda ROM'un tam hatasız test edildiğini kabul ederiz. Aksi takdirde, bir hata mesajı gözükecektir.

ROM toplam kontrol programı RAM test programından daha fazla karışık bir biçimde çalışır ROM toplam kontrol programının yazılım tasarımının karışıklığı ilerideki sistem genişlemeleri için sistem esnekliğini yapma amacına karşılık uygundur. ROM toplam kontrol programının sayısal programlamasını tartışmada önce, mümkün olan ROM bellek düzenini açıklayacağız.

Standart MPF-I/88 16 k bayt'lık bir bellek alanıyla bir 27128'le konulur. Böyle bir şekildeki bellek alanı aşağıdaki gibi gösterilmiştir.

ROM0 C000H'den FFFFH'e kadardır. ROM0'a ayrılan segment adresi F000H'dir.



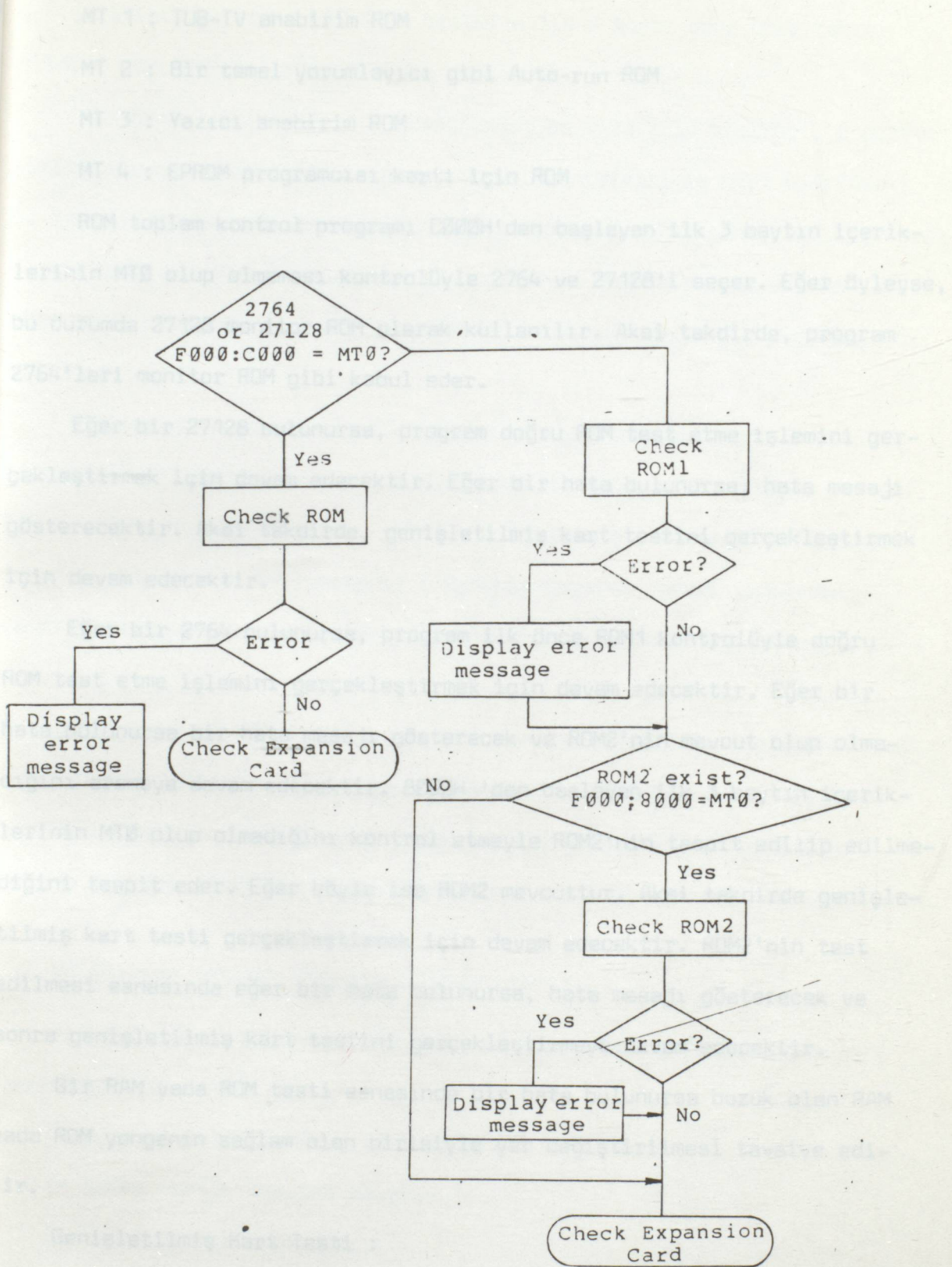
*** ROM bellek düzeni ***

Sistemde ROM yonga gibi kullanılan bir 27128'e rağmen ROM toplam kontrol programı sistem kartındaki ROM'lar gibi tespit edilebilen iki 2764'deki benzeri bir yolla sistemi işletir.

ROM toplam kontrol programı için bir akış diyagramı bir sayfa sonradaki şekil gibi gösterilebilir.

ROM toplam kontrol programı sistem kartındaki ROM yongalarının 2764 ROM yongalarını yoksa 27128 ROM yongalarını olup olmadığının kontrolüyle başlar. Bir 3.bayt ROM belirleyici her bir ROM'un ilk 3 bayt'ında depolanır. İlk iki bayt Multitech'i gösteren (MT) iki karakterle depolanır. ROM toplam kontrol programı iki karakteri bulduğu zaman, ROM'un Multitech tarafında tasarlanan kodlarla depolanmasını temin eder. Bir Multitech ROM'un 3.baytı 0'dan 6'ya kadar olan sırada daima ASCII karakterlerle doldurulur. ROM toplam kontrol programı 3 bayt ROM belirleyiciyi aşağıdaki şekilde çözer.

MT0 : ROM2 (MPF-I/88 hattında çevirici ve ters çevirici (disassemble-i) depo edilir.)



MT 1 : TUB-TV anabirim ROM

MT 2 : Bir temel yorumlayıcı gibi Auto-run ROM

MT 3 : Yazıcı anabirim ROM

MT 4 : EPROM programcısı kartı için ROM

ROM toplam kontrol programı C000H'den başlayan ilk 3 baytın içeriklerinin MT0 olup olmaması kontrolüyle 2764 ve 27128'i seçer. Eğer öyleyse, bu durumda 27128 monitor ROM olarak kullanılır. Aksi takdirde, program 2764'leri monitor ROM gibi kabul eder.

Eğer bir 27128 bulunursa, program doğru ROM test etme işlemini gerçekleştirmek için devam edecektir. Eğer bir hata bulunursa, hata mesajı gösterecektir. Aksi takdirde, genişletilmiş kart testini gerçekleştirmek için devam edecektir.

Eğer bir 2764 bulunursa, program ilk önce ROM1 kontrolüyle doğru ROM test etme işlemini gerçekleştirmek için devam edecektir. Eğer bir hata bulunursa bir hata mesajı gösterecek ve ROM2'nin mevcut olup olmadığını aramaya devam edecektir. 8000H 'den başlayan ilk 3 baytın içeriklerinin MT0 olup olmadığını kontrol etmeyle ROM2'nin tespit edilip edilmediğini tespit eder. Eğer böyle ise ROM2 mevcuttur. Aksi takdirde genişletilmiş kart testi gerçekleştirmek için devam edecektir. ROM2'nin test edilmesi esnasında eğer bir hata bulunursa, hata mesajı gösterecek ve sonra genişletilmiş kart testini gerçekleştirmeye devam edecektir.

Bir RAM yada ROM testi esnasında bir hata bulunursa bozuk olan RAM yada ROM yonganın sağlam olan birisiyle yer değiştirilmesi tavsiye edilir.

Genişletilmiş Kart Testi :

Genişletilmiş kart segment adresi E000H'i tahsis edilir. Genişletilmiş kart test programı 4 Kbayt'lık artmalarla belleği test eder.

Geniřletilmiş karttaki ROM ölçü birimine ilk 3 baytı daha önce bahsedildiđi gibi ROM belirleyicisiyle depolanır.

Bir geniřletilmiş kart bulunduđunda, bu kart için bařlangıç programı yürütülecektir. Bařlangıç program adresi, geniřletilmiş kart üzerindeki 4. baytdan bařlayarak depolanır. Bununla beraber, eđer geniřletilmiş kartla bir auto-run ROM bulunursa auto-run ROM'da depolanan kod derhal yürütülecektir. Böylece bir auto-run ROM bulunduđunda bařlangıç programı yürütülmez.

Bařlangıç programın bir CALL FAR komutuyla bařlamasından bařlangıç programını atlatmak için geniřletilmiş kartta 4.bayttan bařlayan bir RET FAR komutunu kullanmalısınız. (Eđer bir bařlangıç programla kendi uygunlanınızı tasarlamak istiyorsanız.) Bununla beraber, kendi uygulamalarınızın tasarlanmasında sistem yığınının içeriklerinin deđiřmemesine dikkat ediniz.

Eđer Auto-run ROM'ları sistem kartı ve geniřletilmiş kartın her ikisinde tespit edilirse, sistem kartındaki auto-run ROM'dan daha fazla üstünlükte tahsis edildiđinden geniřletilmiş karttaki biri yürütülür. Bir ilk bařlama dönüşü, geniřletilmiş kart testi bitirildiđinde tamamlanır.

Çalışırken Yeniden Bařlama : (Warm Start)

Bir çalışırken yeniden bařlama arasında sistem ařađıdaki görevleri gerçekleştirir.

- 1- Sistem vektör tablosu oluşturur.
- 2- Kesme vektör tablosu oluşturur.
- 3- Yazıcı portu bařlatır.
- 4- Klavyeyi bařlatır.
- 5- RS-232 post - 9600 Baud hızında, iki durdurma biti (stop-bit), çift parity kontrolü, 7-bit sözcük uzunluđu bařlatır.

6- Bir beep sesi üretir.

7- Bir AUTO-RUN ROM için bir genişletme kartı üzerindeki yedeklenmiş boş soketde bir AUTO-RUN ROM'un tespit edilip edilmediğinin kontrolunu yapar.

Eğer bir AUTO-RUN sunulursa, bu AUTO-RUN ROM'da içerilen program otomatik olarak yürür.

Monitor programını kontrole dönüş için, kesme komutu INT 7'yi kullanabilirsiniz.

I/O PROGRAMLAMA :

Bu bölüm MPF-I/88'deki giriş/çıkış programlanmasına kısa bir giriştir. Burada temin edilen bilgi I/O donanım programlamasında bazı fikirler kazanmanıza izin verir. Kullanışlı monitor alt yordamlarındaki konu çeşitli I/O görevlerini gerçekleştirmek için yapılan servis programların kullanımında size bazı bilgiler verir.

Eğer direkt olarak I/O donanımlarını programlamak amacınız yoksa bu bölüm atlatılabilir.

Fakat 8088'in donanım ve yazılımlarıyla ilgili kendinizi daha geliştirmek istiyorsanız bu bölüme bakılabilir.

Eğer I/O programıyla ilgileniyorsanız I/O programlanması fikirleriyle yöntem olmada bu kaynağa ek olarak, monitor programı kaynağı listesi ve yazılım referans rehberinde sunulan bilgilerinde bakılmalıdır.

MPF-I/88'de I/O donanımlarının bir geniş çeşitliliği vardır. Ekran ve yazıcı gibi bazı I/O donanımları yalnızca çıkış için kullanılır. Kalve gibi diğerleri veri giriş hassasiyetidir. Teyp arabirim yada seri arabirim gibi diğerleri, giriş yada çıkış verinin her ikisi olabilirler.

Bir I/O donanımın işlevsel olması için aşağıdaki şu durum olmalıdır.

1- Bir arabirim devresi mevcut olmalıdır. Bir arabirim devresi sistemle iletişime I/O donanımı yoluyla bir iletişim hattıdır.

2- Bir I/O donanımı sürücüsü mevcut olmalıdır. Bir donanım sürücüsü I/O donanımının sistemle etkileşebilmesi için I/O donanımını sunan bir programdır.

3- I/O donanımı sistem belleğine yüklenmiş yada yerleştirilmiş olmalıdır.

Eğer ilerideki kendi I/O donanımlarınızı tasarlamak amacınız varsa, kendi donanım sürücülerinizi yazmak zorundasınız. Hemde, yine kendine özgü bir uygulama için kendi uygulama programınıza bir donanım sürücüsü dahil edebilirsiniz. I/O programlanmasını öğrenmenin en iyi yolu, mevcut olan donanım sürücü programlarını komut komut izlemektir. MPF-I/88 sistemi için donanım sürücüleri monitor program kaynak listesinde bulunmaktadır.

I/O PORTLARI :

I/O Bellek Düzenlenmesi :

Mikrobilgisayarlarda I/O tasarımı için iki genel yol vardır. Biri I/O donanımların bellek düzenlenmesi olabilmesidir. I/O donanımları bellek bilgilerine erişebilir. Bilgisayar sistemlerinde I/O donanımı eğer bellek bölgelerine erişilmiş ise bellek düzenlenmesi olmayı gösterir.

I/O - I/O Düzenlenmesi :

Bilgisayar tasarımında bir genel pratik yolda sistem ve harici donanımlar arasında bilgi transferi için I/O partlarının kullanımıdır. Sistemde belirtilen her bir I/O donanımı bir kendine özgü I/O port adresiyle tahsis edilir. MPF-I/88 bu sonraki yöntemi kullanılır. Veri bir donanımda sisteme transfer edileceği zaman IN komutu kullanılır.

I/O Port Adresleri :

I/O donanımlarını birleştirmede tahsis edilen I/O part adresleri aşağıdaki gibidir.

<u>LCD :</u>	Oku	Yaz
Komut	1A2H	1A0H
Veri	1A3H	1A1H

Yazıcı :

Yazıcı çıkışı veri port : port 1E0H

Yazıcı seçici (strobe) STB : 180H portunun 7. biti

BUSY (yazıcı) : 1C0h portunun 6. biti

Klavye :

Klavye dizi çıkışı : 180h portunun 0.bitinden 3.bitine ve 160H portunun 7.bitinden 0. bitine

Klavye dizi girişi : 1C0H Portunun 4. bitinden 0. bitine

Kontrol tuşu : 160H portunun 6. biti

TEYP-ÇIKIŞI (BEEP) : 180H portunun 6. biti

TEYP-GİRİŞİ : 1C0H portunun 7. biti

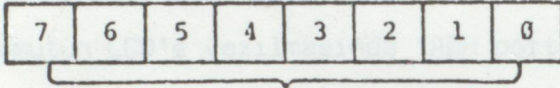
Yazıcı Sürücü :

Yazıcı sürücü sistemden yazıcıya bilgi göndermenizi tasarlayan bir programdır. Yazıcıya çıkış olmak için bilgi ilk önce AL yazmacından yazıcı portu 1E0H'e gönderilir. Sonra 1C0H portunun 6. bitinin kontroluyla yazıcının meşgul olup olmadığı test edilir. Eğer yazıcı meşgul değilse bu durumda veri yazıcı yastığına gönderilecektir. Bir sıfır seviye STROBE hattında (180H'in 7. biti) hissedildiğinde, veri yazıcı yastığından yazıcıya gönderilir.

Yazıcı program çağrıldığında sistem ilk önce yığına yazmaçların içeriklerini itmekle ve sonra sistem verisinin işaretlemek için veri segment yazmacının içeriklerini düzeltmekle sistem durumunu koruyacaktır. Sistem ve yazıcı arasındaki bir veri transferinin tamamlanmasından sonra, STROBE hattı tekrar 1" seviyeye çekilecek ve yazıcı sürücü alt yazmaç yolu ile bir veri transferi sonucuna dönecektir. Eğer bir veri transferi başarılı bir şekilde gerçekleştirildiyse, daha sonra bir sıfır alt yazmaça döndürülmüş olacaktır. Eğer bir bilgi transferi başarılı olarak

Port 160H:

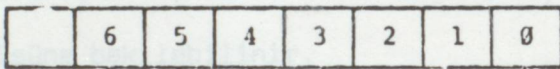
BITS



KEYBOARD ARRAY OUTPUT

Port 180H:

BITS



KEYBOARD ARRAY OUTPUT

NMI (Enabled=1)

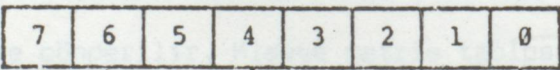
NC (No connection)

TAPE-OUT (BEEP) (Enabled=1)

PRINTER STROBE (Enabled=0)

Port 1C0H:

BITS



KEYBOARD ARRAY INPUT

CONTROL KEY

PRINTER BUSY

TAPE-IN

gerçekleştirilmediyse, bu durumda bir "1" AH yazmacına döndürülmüş olacaktır.

Programlama Göstericisi :

LCD dört I/O portunda - 1A0H, 1A1H, 1A2H ve 1A3H- programlanır. Bir komutun LCD'e yazılmasında 1A0H portu kullanılır. Bir komutun okunması isteniyorsa port 1A2H kullanılır. Bilgi 1A1H portu yoluyla LCD'e çıkış ve 1A3H portu yoluyla LCD'e giriştir.

LCD'i programlamak için, bir ilavede bulunan LCD göstergenin veri kitabına ve MPF-I/88 Monitor Program kaynak listesinde bulunan gösterge sürücüsüne bakılabilir.

Klavye :

Klavye matris 12 sütun hattı ve 5 satır hattı içerir. Sistem her 15 ms'de klavyeyi sarar. Birkalvyeye taranması sırasında diğer 11 kolon hattı 1'de iken 180H portunun 3.bitinden 0. bitine kadar 160H I/O portunun 8.bitine bağlanan 12 kolon hattının biri 0'a çekilir. Bir tuşa basıldığında, bu 0 darbesi satır hatlarının birinde hissedilir ve bir kod sisteme gönderilir. Klavye matris tablosuna bakılabilir.

Kontrol Tuşu :

1C0H çıkış portunun 5.biti kontrol tuşu tarafından kullanılır. Klavyedeki kontrol tuşuna basıldığında, bu bacak aktif durumdadır.

SES ARABİRİMİ (AUDIO INTERFACE) :

Sistemden teyp veri transfer edildiği zaman, OUT komutu kullanılmasıyla 180H çıkış portunun 6.bitinden bilgi gönderilir, Hemde yine buzzer için çıkış bu bit içinden gönderilir.

Veri teypden sisteme okunduğu zaman, IN komutunun kullanılmasıyla 1C0H giriş portunun 7.bitinden bilgi gönderilir.

BANDA KAYDETME BİÇİMİ :

Sistemin teyp'e bilgiyi yazdığı her zaman bir banda kaydetme biçiminde kaydedilir. Banda kaydetme biçimi aşağıdaki gibi tarif edilir.

- 1- Kılavuz ses : 00'dan 256'ya sıralı baytlar.
- 2- Zaman uyumlama biti : 1
- 3- Zaman uyumlama baytı : 016H
- 4- Bir öbek gibi (veri kaydı) depolanmış 256 bayt bilgi
- 5- CRC baytları : İki CRC' baytları tarafında takip edilen herbir data öbeği
- 6- Son kısım : Son kısım 4'baytı içerir.

Kılavuz ses veri transfer edildiği zaman, veri girişini bulmada sisteme yada teyp kaydına izin veren bir sinyal gibi davranması için tasarlanır.

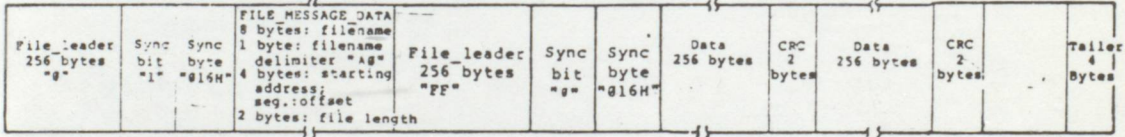
Zaman uyumlama bitine bazen bir biçim verme biti yada durdurma biti olarak baş vurulur. Sistem bir veri transferinin olduğu bir harici donanım (bu durumda teyp kaydedici) belirtmede bu biti çıkarır.

Zaman uyumlama baytı bir veri iletme tutanağı gibi kullanılır. Veri teypde sisteme okunduğunda eğer bir doğru zaman uyumlama baytı okunuyorsa bu durumda veri sisteme transfer edilebilir. Eğer doğru zaman uyumlama baytı bulunmadıysa arabirim sürücü baş tarafı yeniden araştıracaktır.

Bilgi 256 baytlık veri kayıtları serisi biçiminde teyp'e depolanır. Herbir 256 bayt veri kaydı iki CRC baytı tarafında izlenir. CRC baytları veri iletilmesi esnasındaki hata kontrolleri için kullanılır. CRC baytları bilgi teyp'e depolandığı zaman herbir veri kaydında sonra teype yazılır. Veri teypden sisteme okunduğunda sistem usüle göre iki CRC baytı üretecektir. Daha sonra üretilmiş olan iki CRC bayt'ı CRC bayt'ları ile mukayese edilecektir. Eğer CRC' baytlarının içerikleri uygunsa, veri kaydı sinyalleri doğru olarak geçirilir. Aksi takdirde veri transferi sırasında bir hata oluşur.

Son kısım bir dosyanın sonunu belirler. MPF-I/88 teypte kaydetme biçimi aşağıdaki gibidir.

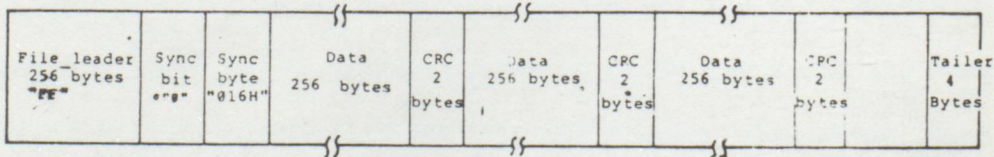
MRF-I/88 TEYPE KAYDETME BİÇİMİ.



Hemde kaset arabirim sürücü, sistem belleğine yüklenmiş olması için IBM PC kaset teypte kaydetme biçiminde bilgi depolanan kaset teypte izin verir.

IBM teyp kaydetme biçimi aşağıdaki gibi gösterilir.

IBM-PC TEYPE KAYDETME BİÇİMİ .



- 1- Kılavuz ses : FFH'in 256 sıralı baytları
 - 2- Zaman uyumlama biti : 0
 - 3- Zaman uyumlama baytı : 016H
 - 4- Dosya ismi : 8 bayta dosya ismi depolanır.
 - 5- 1 bayt'lık dosya ismi sınırlayıcı-00- derhal sonraki bir dosya ismine depolanır.
 - 6- Başlama adresi : 4 bayt dosya başlama adresini depolama için kullanılır. Başlama adresi segment adresleri için 2 bayt, ofset adresleri için 2 bayt içerir.
 - 7- Dosya uzunluğu : Bir dosya uzunluğu 2 bayt içinde depolanır.
- Hemde bir IBM'de biçimlenmiş teyp'de veri kaydı iki CRC' baytı tarafından izlenen 256 bayt tarafından şekillenir.

MPF-I/88 DEVRE TANIMLANMASI :

Bu bölüm size MPF-I/88'in kısa bir devre izahını verecektir. Bu bölümü okuduktan sonra, sistemdeki donanım parça işlevlerinin nasıl olduğuna dair biraz fikir sahibi olacaksınız.

Sistemin donanım görünüşüyle ilgilenmeyen okuyucular için bu bölüm atlanabilir. Bununla birlikte, donanımla ilgilenen ve kendi özel uygulamalarıyla sistemi genişletmeyi isteyen okuyucular için bu bölüm okunmalıdır.

Aşağıdaki işlevsel sistem parçalarını anlatacağız.

1- CPU ve destekliyiçi devreleri :

- 1) Sistem zamanlama devresi.
- 2) Sistem bekle lojiğı.
- 3) Sistem reset devresi.
- 4) Kesme öbeğı lojiğı.
- 5) Yol yastığı.
- 6) Bellek ve I/O donanım çözücüleri.

2- Sistem belleğı

3- Giriş/Çıkış arabirim lojiğı.

CPU VE DESTEKLEYİCİLERİ LOJİKLERİ :

1- Sistem zamanlama devresi :

Sistem zamanlama devresi 14.318 MHz'lik bir kristal osilatör ve U4 bölgesi kartında 74LS04'ü içerir. 14.318 MHz'lik bir frekans U4'ün 2.bacağında üretilir. Bu sinyal U₁₀'da uç devre tarafında bölünmeye gitmesinden sonra 3.bacakta 4.77 MHz'lik bir frekans elde etmek için üç ile bölünür. 4.77 MHz'lik bir saat frekansı sistem saati ve 62.bacak genişletme yerine (EXT-BUS) U₃'ün (CPU) 19.bacağına sistem saati olarak verilir.

2- Bekle Lojiği :

Bekle safhası lojiği, sistem bir I/O okuma yada yazmasını gerçekleştiretiyorken 8088 sıfır aktif olan READY girişini (22.bacak) sürmek için gereklidir. Bekle safhası lojiği U5, U6, U9, U13 ve U14 deki tümleşik devreleri içerir. Sistem bir I/O okuma yada yazmasını gerçekleştireceği zaman 8088'in RD (32.bacak) ve WR (29.bacak) çıkışları, daha sonra çıkışı U14'te 2 den 4'e hat çözücü çiftinin 1.bacağına gönderilecek olan U6'daki AND kapısında VE işlemi uygulanır. 8088'in IO/M çıkışının (28.bacak) U14'ün 2.bacağına gönderilmesiyle U14'ün 4., 5., 6. ve 7. bacakları, 1 ve 2.bacak girişlerinin durumuna bağlı olarak MEMR, IOR, MEMW, IOW gibi sinyallerin birini üretecektir. MEMR, IOR, MEMW, IOW sinyal hatları U13 de sekiz tane yol sürücü bulunan 74LS244 içinden EXT-BUS'ın B12,B14,B11 ve B13 bacaklarına bağlanır.

EXT-BUS'dan gitmeden sonra IOR ve IOW U6'da ve U5'in (4 tane 2 girişli OR kapısı) 13.bacağına U6'nın 3.bacağından çıkışda VE işlemi uygulanacaktır. OR kapısının çıkışı bir bekle dönüşü (Wait cycle) üretmek için U9'da bir çift 74LS74 D tipi Flip-flop içinden 8088'in READY girişini (22.bacak) besleyecektir. Not edilmelidirki U9'un 11.bacağına giriş, sistem saat dönüşünün (clock cycle) negatif kısmının sistem zaman uymunda kullanılabilmesi için U10'un 2.bacağıyla besletilir. Bekle safhasının uzun bir periyoduna bir çevresel donanım tarafında ihtiyaç duyulması durumunda, fazla bekle safhalarını ilave etmek için U9'un 13.bacağına EXT-BUS'ın A10 I/O kanal hazır (channel ready) hattı içinden bir 0 seviye darbesi gönderilebilir.

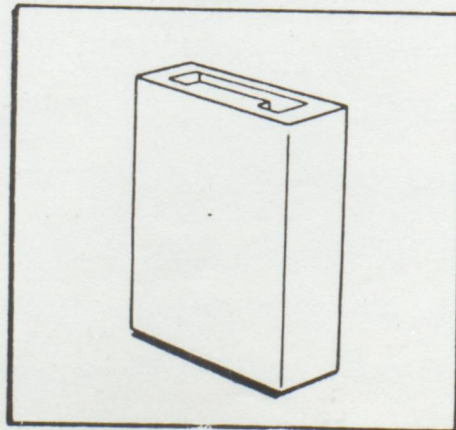
3-Sistem Sıfırlama (reset) Devresi :

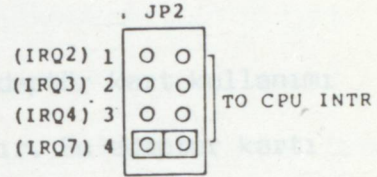
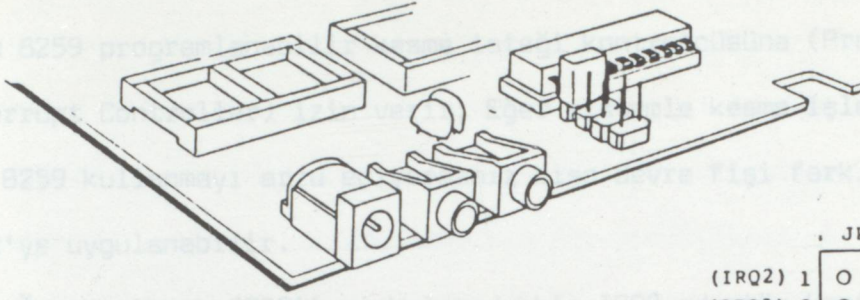
Sıfırlama sinyali ilk önce, klavyeden U_8 'in (74LS14 için 6 tane evirici bulunan "Schmitt" tetikleyicisi) 1.bacağındaki sistem sıfırlama devresine gönderilir. Sıfırlama sinyalinin Schmitt tetikleyici tarafında tamamlanmasından sonra bir sütun sıfırlama dönüşü başlatmak için CPU'nun 21.bacağı Schmitt tetikleyicisinin 3.bacağından besletilir. Hemde, yine sıfırlama sinyali EXT-BUS'ın B2 hattına sunulur. Schmitt tetikleyicinin 4.bacağı flip-flop'un içeriklerini temizlemek için (temizlenmeli sekiz taneli D tip flip-flop) U_{21} ve U_{22} 'nin 1.bacağına bağlanır.

4- Kesme isteği lojiği :

Bir çevresel donanım tarafından üretilen bir kesme isteği ilk önce U_8 'in 13.bacağına EXT-BUS'ın B8 hattından CPU'ya gönderilir. Daha sonra U_4 'ün (bir altılı çevirici) 11.bacağı içinden INTR'a (8088'in 18.bacağı) gönderilir. 8088 tarafında INTR sinyalinin kabul edilmesinden sonra bir kesme kabülünü (INTA), bir sıfır aktif darbesini üretecektir. Sıfır darbesi EXT-BUS'un B5 hattından kesme cihazına gönderilir.

Kullanıcılar 8088'e IRQ2, IRQ3 ve IRQ7'den kesme isteklerini takip etmek için JP2'deki bir kısa devre fişine (close Jumper) başvurabilirler. 8088'e gelen kesme isteklerinin nasıl takip edileceğini daha sonra açıklayacağız. NOT : Kısa devre fişleri bir standart MPF-I/88 paketinde temin edilir. Aşağıdaki gibi gösterilir.





Kısa Devre Fişinin Şekli.

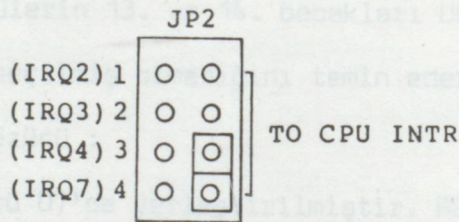
Eğer IBM paralel yazıcı adaptörü gibi bir adaptör kartına sahipseniz secmeli genişletme biriminde bu adaptör kartı tıkayabilirsiniz ve 8088 için IRQ7 yoluna takip etmede kısa devre fişine başvurabilirsiniz.

Not edilmelidir ki, kısa devre fişi aşağıdaki kartta gösterildiği gibi arzu edilen bacak çiftleri için uygulanır.

MPF-I/88'in donanım tasarımı kesme işlemini tutmada kullanılabilmek için 8259 programlanabilir kesme isteği kontrolcüsüne (Programmable Interrupt Controller) izin verir; Eğer sistemle kesme işlemini tutmada bir 8259 kullanmayı arzu ediyorsanız kısa devre fişi farklı bir yolda JP2'ye uygulanabilir.

Örnek olarak 8088'i maskelemeye bir IRQ2 adaptör kart kullanımı 8259 kesme isteği kontrolcüsüyle birlikte kullanılır. Bu adapter kartı seçmeli genişletme biriminde tıkayabilirsiniz ve sonra genişletme biriminde 8259 kartını tıkayın.

Sistemi bu biçimde şekillendirmenizden sonra, kısa devre fişini aşağıdaki gibi JP2'ye uygulanabilir. Not edilmelidirki, sistemde 8259 kullanıldığı zaman kısa devre fişleri yalnızca sağ kolondaki bacağının herhangi ikisine uygulanmalıdır. 8259'ın 17.bacağı (INT) genişletme biriminin 88.bacağına (INTR) bağlanır.



5- Yol Yastığı (Bus Buffer) :

Yol yastığı U1(74LS373), U12(74LS245 sekiz tane tri-state yol verici (bus transceiver) ve U13(74LS244)'ü içerir. 41'deki 74LS373 en yüksek değerli adres/durum hatları A16/S3, A17/S4, A18/S5, A19/S6, D+/R, S50 ve IO/M için bir "latch" gibi kullanılır. U2'deki 74LS244 A16'dan A8'e kadar Sekiz adres hattı için bir yol sürücü görevini yapar. U11'deki 74LS373 AD1'den AD0'a kadar sekiz çoğullamalı adres/veri hatları için bir "latch" gibi hareket eder. Sistemin çoğullamalı yol şeklini kullanmasından 8088'in 25.bacağı (ALT-Adress latch Enable) U13'deki yol

sürücünün 13.bacağından kart bölgesi U1 ve U11'deki iki "latch" in saat girişlerine (pin 11) bağlanır. U1 ve U11'de iki adres "latch"lerine ALE hattı bağlanmasıyla sabit adres, ALE sinyali 1'e çekilir çekilmez bir yol dönüşünün birinci T durumunda "latch" edilebilir. U1'in 3,8 ve 1b.bacakları, sistem yol dönüşlerinin durumunu yansıtmak için birleştirilmelerinden IO/M, DT/R ve SSØ'a bağlanırlar. Hemde yine sistemin bir HALT durumunda olduğu zaman aydınlanan kırmızı LED'in durumunu temin ederler.

6- Bellek ve I/O Donanım çözücüler :

a- ROM/RAM çözücüler :

U14'deki 74LS139 (çift 2'den 4'e hat çevirici), U7'deki 74LS138L3' den 8'e hat çevirici) ve U16 ve U15'deki 74LS138'ler bellek ve I/O donanın çözücülerini gibi kullanılırlar. U14'deki çözücü hatta RAM/ROM çözücüleridir. Hat çözücülerin 13. ve 14. bacakları U6'nın 6.bacağıyla beraber RAM yada ROM'un seçilmiş olmadığını temin eder.

b- ROM Çözücü :

ROM çözücü U7'de yerleştirilmiştir. ROM çözücünün 3.ve 6.bacakları (A16 ve A17 adresli hatları) 1.ve 2.bacakları (A14 ve A15) ve 5.bacağı ROM yonganın seçildiği sıralı ifadede, U7'nin Y5,Y6 ve Y7 (7.,9. ve 10.bacaklar) çıkışlarının durumunu temin eder. 8Kx8 yada 16Kx8 ROM yongalarının her ikisi sistem ROM'u olarak kullanılabilir. Standart MPF-I/88 bir 16Kx8 ROM yongasıyla yapılır. Bu yüzden, standart MPF-I/88 toplam 16K baytlık bir bellek kapasitesine sahiptir.

Eğer A16=1, A15=0 ve A14=1 ise bu durumda Y5 sıfıra çekilir. Y5'in sıfıra çekildiği anda U2Ø'de yerleştirilen ROM yonga seçilecektir. Bu ROM yonganın başlangıç adresi F4ØØØH'dır.

Eğer $A16=1$, $A15=1$ ve $A14=0$ ise bu durumda $Y6$ 0'a çekilir. $Y6$ 'nın 0'a çekildiği anda $U20$ 'de yerleştirilen ROM yonga seçilecektir. Bu ROM yonganın başlangıç adresi $F8000H$ 'dir.

Eğer $A16=1$, $A15=1$ ve $A14=1$ ve bu durumda $Y7$ 0'a çekilir. $Y7$ 'nin 0'a çekildiği anda $U18$ 'de yerleştirilen ROM yonga seçilir. Bu ROM yonganın başlangıç adresi $FC000H$ 'dir.

c- RAM Çözücüsü :

RAM Çözücü $U16$ 'da yerleştirilir. $U16$ 'nın 4.bacağının durumu $A16$ ve $A17$ tarafından belirlenir. $U16$ 'nın 5.bacağının durumu $A13$ ve $A14$ tarafından belirlenir. $U16$ 'nın $Y0, Y1$ ve $Y2$ çıkışları 4,5,2($A12$), 1($A11$), ve 3($A15$) bacakları tarafında belirlenir. $2K \times 8$ yada $8K \times 8$ RAM'ın her ikisi sistem RAM'ı olarak kullanılabilir. Eğer $2K$ RAM kullanırsa, RAM çözümü- lenmesi aşağıdaki gibi gösterilir.

Eğer $A15, A14, A13$, ve $A12$ sıfır fakat $A11=1$ ise $Y1$ seçilir. Bu durumda seçilen RAM'ın başlangıç adresi $00800H$ 'dir.

Eğer $A15=0$, $A14=0$, $A13=0$, $A12=1$ ve $A11=0$ ise $Y2$ seçilir. Bu durumda seçilen RAM'ın başlangıç adresi $01000H$ 'dir.

Eğer $8K$ 'lık RAM kullanılırsa RAM çözümlemesi aşağıdaki gibi olur.

Eğer $A15=0$, $A14=0$, ve $A13=0$ ise $Y0$ seçilir. Bu durumda seçilen RAM'ın başlangıç adresi $00000H$ 'dir.

Eğer $A15=0$, $A14=0$ ve $A13=1$ ise $Y1$ seçilir. Bu durumda seçilen RAM'ın başlangıç adresi $02000H$ 'dir.

Eğer $A15=0$, $A14=1$ ve $A13=0$ ise $Y2$ seçilir. Bu durumda seçilen RAM'ın başlangıç adresi $04000H$ 'dir.

Eğer 8K'lık RAM kullanılıyorsa JP3 ve JP4 aşağıdaki gibi olmalıdır.

	JP4	JP3
2K x 8	Closed	Open
8K x 8	Open	Closed

d- I/O Çözücü :

I/O çözücü U15'de yerleştirilmiştir. I/O çözücünün 1,2,3 (A7'den A5'e) 4,6 (A8,A9) ve 5 (IOR yada IOW) numaralı bacakları I/O çözümlemesini belirlemek için kullanılır. U15 içinden erişimli donanımlar;

1- Gösterge (I/O part adresi 1A0H)

2- Klavye (I/O adresleri 160H, 180H, ve 1C0H)

3- Yazıcı (I/O adres 1E0H)

4- Ses ara birimi (I/O adresleri 180H ve 1C0H)'dir.

Bir I/O part seçildiği zaman çözücülerin karşı gelen akışı (Y3, Y4,Y5,Y6,Y7) aktif hale getirilir.

SİSTEM BELLEĞİ :

Sistem ROM yongaları, RAM yongaları U23, U24 ve U25'de yerleştirilirken U18, U19, U20'de yerleştirilir. 8K yada 16K ROM'ların herbiri U18, U19 ve U20 bölgelerine yerleştirilebilirler. 2K yada 8K RAM'ların herbiri U23,U24 ve U25 bölgelerine yerleştirilebilirler. Atlatma yapan telleri RAM yonga kullanım birimi seçmek için U7 yada U16'ya uygulanmış olmalıdır.

GİRİŞ/ÇIKIŞ ARABİRİM LOJİĞİ :

(Y7'den 13'e) U15'in çıkışları kesin olarak I/O portu erişilmesini belirler. U21 ve U22, U26 klavye ve teyp veri girişi için sürücü gibi

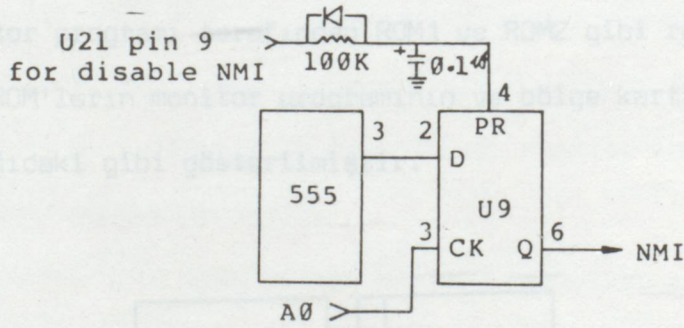
kullanılırken klavye çıkış verisi için "latch"ler olarak kullanılırlar.

U27 ve U8 teypden giriş sinyalleri için kullanılırlar. Teyp çıkış sinyalleri U21'in çıkışında (5.bacak) dışarı gönderilir. Hemde bu bacak yeşil LED'in ve "buzzer"ın durumunu belirler.

U28'in (555) 3.bacağı NMI için kaynak sinyali gibi bir 15 ms'lik saat sinyali üretir. U9'un 4.bacağı programlanabilir. Bu bacadan bir NMI üretilmesi istenmediği zaman U9'un 4.bacağından bir NMI isteğini geçerli kılmamak için 0'a çekilmiş olmalıdır.

U17 yazıcı çıkış bilgisi için bir "latch" gibi kullanılır.

7805 + 9 V'luk girişi + 5 V çıkışa çeviren bir gerilim regülatörüdür. +5V gerilimi ile uygun çalışma için sistem besletilmiş olmalıdır. Bu aratarlamalı güç kaynağı genişletme kartı sisteme yerleştirilmiş olduğunda gerek duyulan gücü depolamada kullanılmalıdır.



AUTO-ROM VE HARİCİ KOMUTLAR :

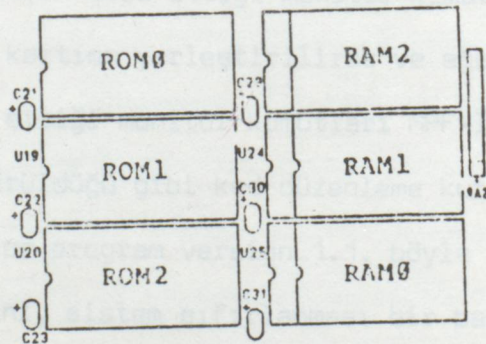
AUTO-ROM :

MPF-I/88'in monitor programı (Version-1.1) istediğiniz uygulamalar için kendi AUTO-ROM'lerinizi tasarlamaya izin verir. AUTO-ROM'da depolanma program makineye beslemeyi verir vermez yürütülmeye başlar. AUTO-ROM, ROM soketlerinin (yada bir genişletme kartındaki ROM soketi) herhangi birimde tespit edilebilir.

Monitor programı "version 1,1" in altında, bir AUTO-ROM ana MPF-I/88 kartındaki kurulan iki ROM soketlerin herhangi birimde yerleştirilebilir. U19 ve U20 kart durumları sistem ROM genişlemesi için saklanır.

Monitor programı 1.0. altında AUTO-ROM'u bu biçimde yerleştiremiyeceğinizi dikkate almalısınız.

Monitor programı sıralı olarak ROM1 ve ROM2 gibi U19 ve U20 bölge kartlarında yerleştirilen iki kartı görüntüler. Diğer bir deyişle U19 ve U20 monitor programı tarafından ROM1 ve ROM2 gibi referans alınır. Genişletme ROM'ların monitor programının ve bölge kartlarının arasındaki bağıntı aşağıdaki gibi gösterilmiştir.



Gücün verilmesinde, monitor programı, herhangi bir sistem genişletme ROM'unun sistem sıfırlanması sırasında yerleştirilip yerleştirilmediği kontrol edecektir. Eğer monitor programı bir AUTO-ROM'un hazır bulunduğunu tespit ederse, bu durumda AUTO-ROM'da depolanan programı yürütecektir.

Monitor program "versionu" 1.1 altında sistemi genişletmeyi sizin için uygun yapmak için, AUTO-ROM'da depolanmış programın önünde ORGØH çevirici yönlendiricisini yerleştirebilirsiniz. Bununla birlikte, Monitor Program "version'u 1.0 altında, AUTO-ROM, ROM1'in başlama adresi FØØØ ; 8ØØØ ve ROM2'nin başlama adresi FØØØ : 4ØØØ olacak biçimde tasarlanmalıdır.

HARİCİ KOMUTLAR :

Monitor program "version" 1.1 kolay bir şekilde yürütmek için kullanıcının tasarladığı monitor komutlarına yada programlara izin verir. Kullanıcının tasarladığı monitor komutları kullanıcı tarafından EPROM'da depolanmalıdır. Bu kullanıcının tasarladığı monitor komutları harici komutlar olarak gönderilmelidir.

Eğer kullanıcının arzu ettiği monitör komutlarını içeren bir EPROM sistem genişletme kartına yerleştirilirse ve eğer ROM belirleyici ve kullanıcının arzu ettiği monitör komutları MPF-I/88 sistem sıfırlaması konusunda ileri sürüldüğü gibi kod düzenleme kurallarına uygun EPROM'da depolanırsa, monitor program version 1.1, böyle bir EPROM'un mevcuyetinin temin edildiğini, sistem sıfırlanması bir başlangıç işlemini gerçekleştirmek için kullanıcının arzu ettiği EPROM kullanmasına şans tanır. Eğer yonga başlangıç istenmiyorsa, RET FAR komutu ROM belirleyicisinden sonra derhal depolanmalıdır. Bir başlangıç gerçekleşmeyi arzu ettiğinizde

başlangıç programın son komutu olarak RET FAR'ı kullanmalısınız. Kullanıcının istediği bir EPROM'la bir sistemde, sistem sıfırlanması gerçekleştirildikten sonra, monitör programı kullanıcının istediği EPROM'da harici komutların erişebilir olduğunu bilir.

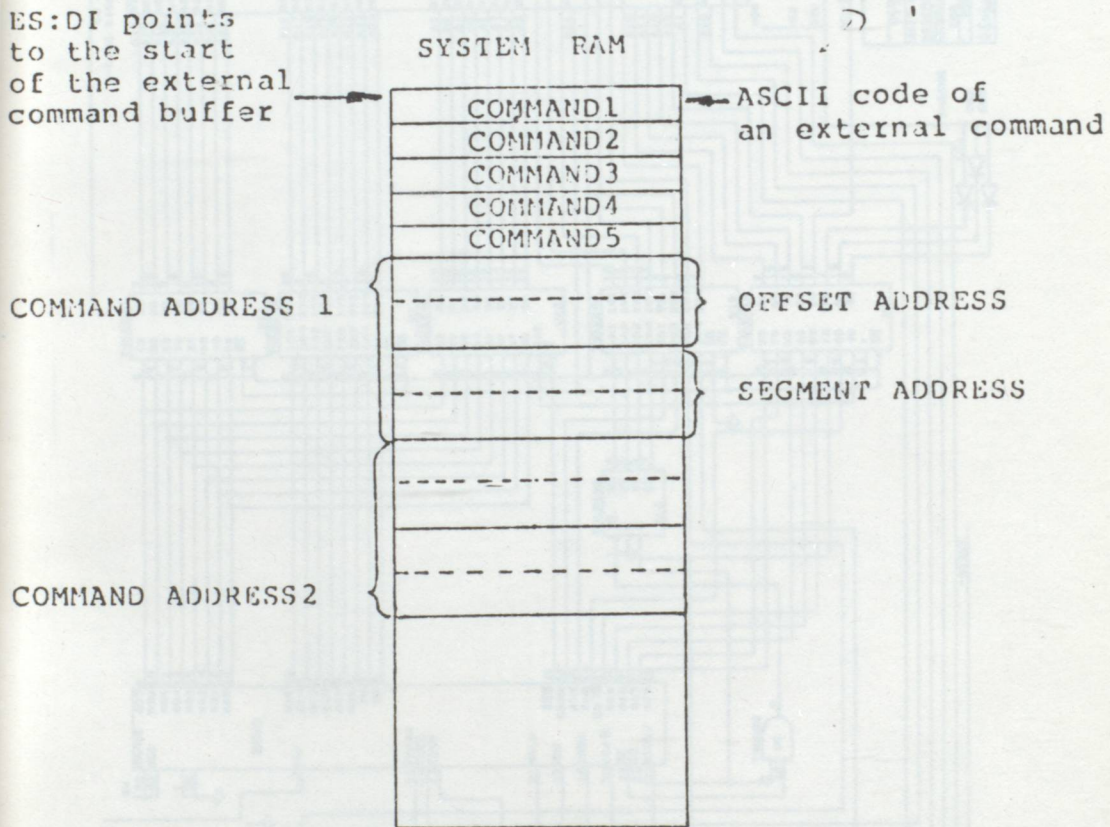
Yorumlayıcı komutu yalnızca dahili monitör komutlarının yöntemi değil harici komutlarında yöntemi olabilir. Herhangi bir an komut girildiğinde, yorumlayıcı komutu, dahili komut karakteri için bir yoklama tablosu kontrolüyle komutun bir dahili komut olup olmadığını kontrol edecektir. Eğer girişilen komut karakteri dahi komut tablosunda bulunmaz ise yorumlayıcı komutu girilir komutun bir harici komut olup olmadığını anlamak için harici komut yastığı (Sistem RAM'ındaki diğer bölge) araştırılacaktır.

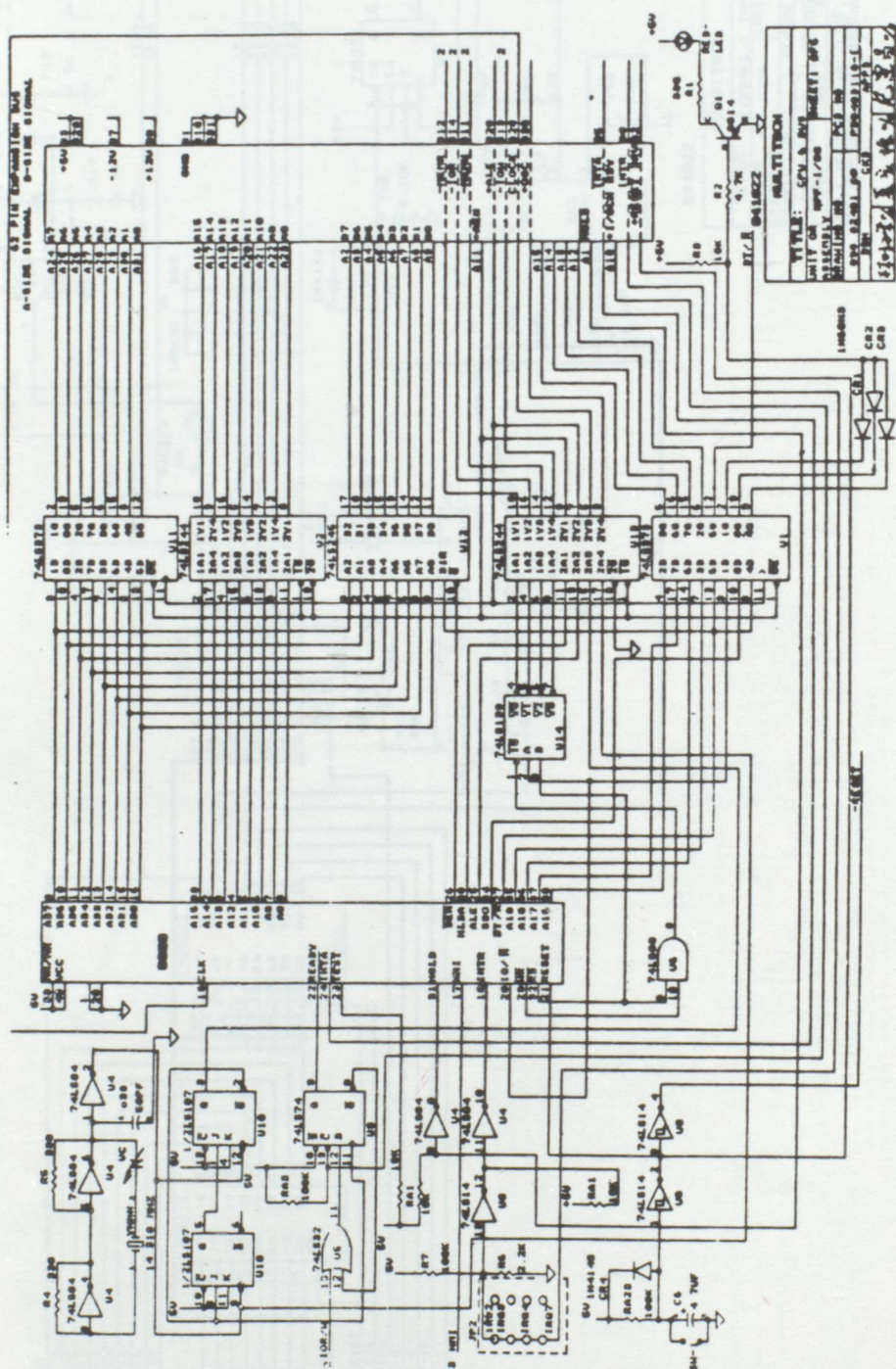
Harici yastık (buffer) komutu kullanıcılar tarafından tasarlanan beş harici komutu yığmak için monitor programı tarafından bir tarafa koyulan bir bellek öbeğidir. Harici komutlar aşağıdaki şekilde RAM'da depolanır.

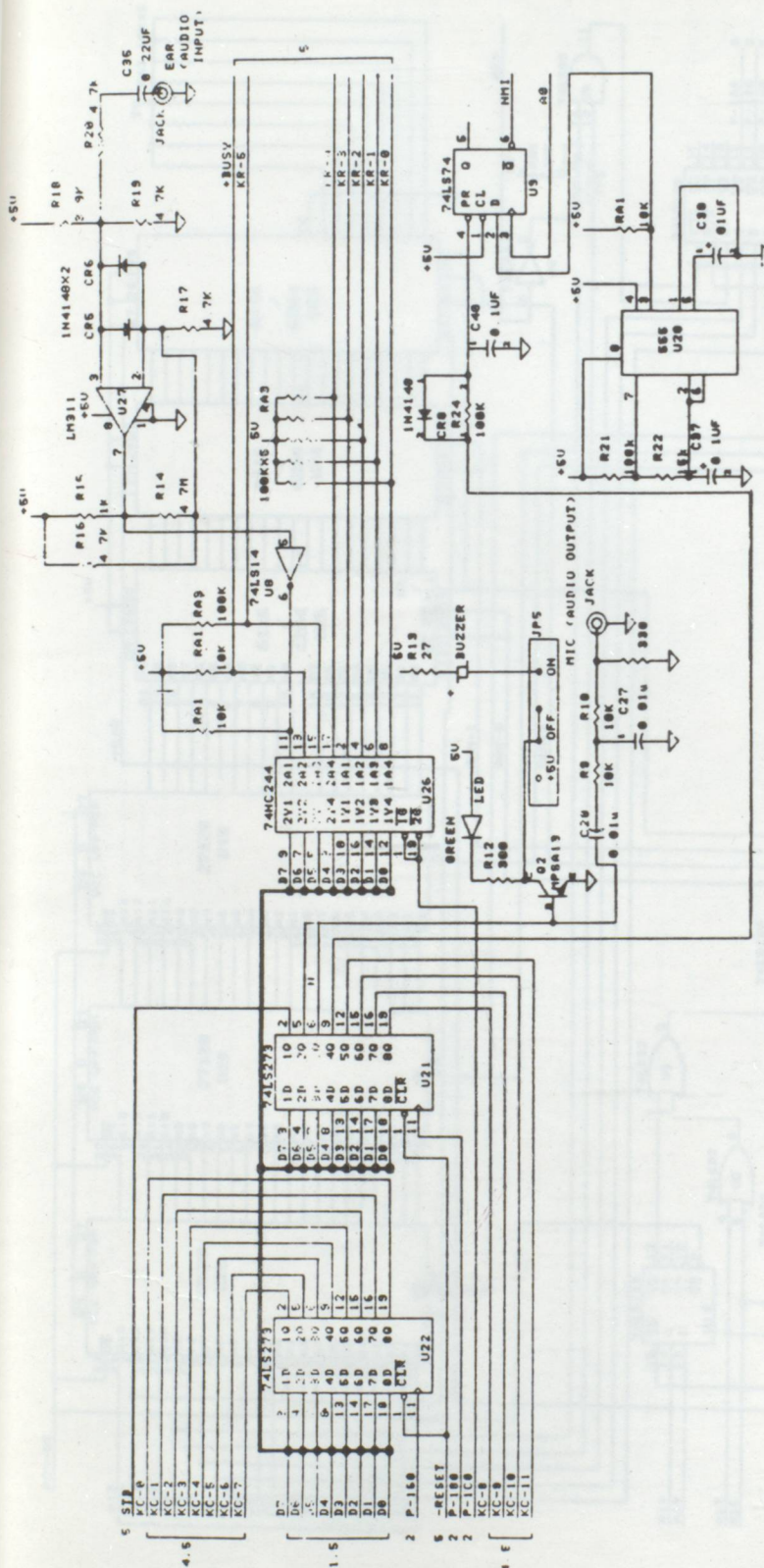
Harici komut yastığının (buffer'in) ilk 5 baytı komut karakterlerinin ASC11 kodlarını depolamada kullanılabilir. Herbir bayt bir harici komut karakterinin ASCII koduyla depolanabilir.

Haric komut yastığının sonra gelen 20 baytı harici komutların başlangıç adresleriyle depolanır. Bir komutun başlangıç adresi ilk iki baytta depolanmış segment adresine depolanabilmesi için 4 bayt alır. Komut yastığının başlangıç adresi monitör programı sizin başlangıç programınıza atlarken ES ve DI yazmaçları tarafından işaretlenir.

Eğer girilen komut bir harici komut ise bu durumda monitor hangi komutun girildiğini bulmak için harici komut yastığına erişecektir. Eğer girilmiş monitör komut karakteri harici komut yastığına henüz depolanmış ASCII kodların herhangi biri ile aynı bulunursa, monitör komutun başlangıç adresinin alıp getirecek ve daha sonra kullanıcının istediği harici komutu yürütecektir.

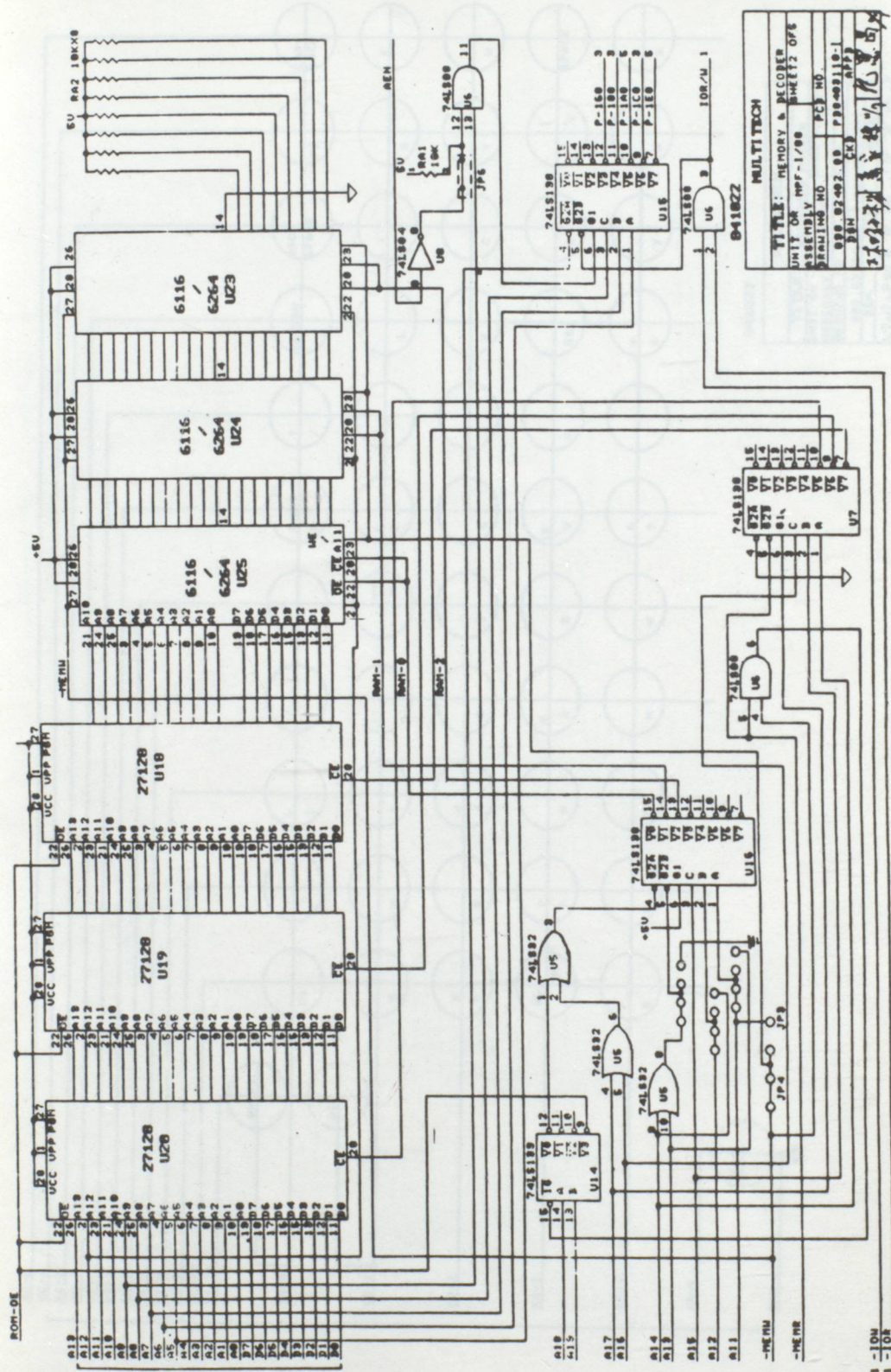


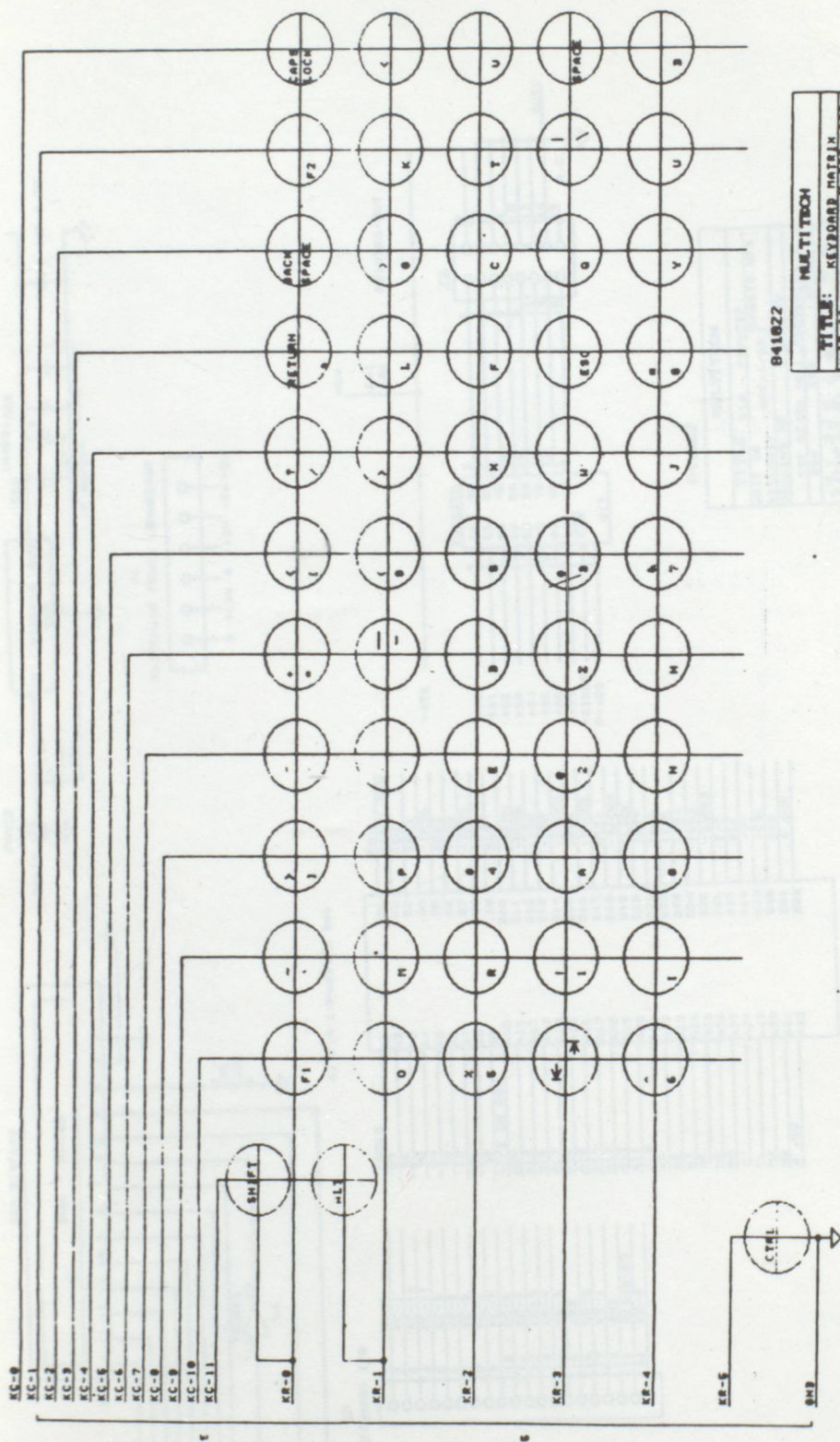




841022

MULTI TECH			
TITLE: KEYBOARD & TAPE			
UNIT OR	MPF-1/80	SHEET 3	OF 5
ASSEMBLY NO.	PCB NO.		
888 02493 90	P39493119-1		
REV	APP		
1	1		





941822

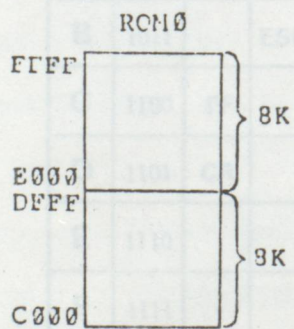
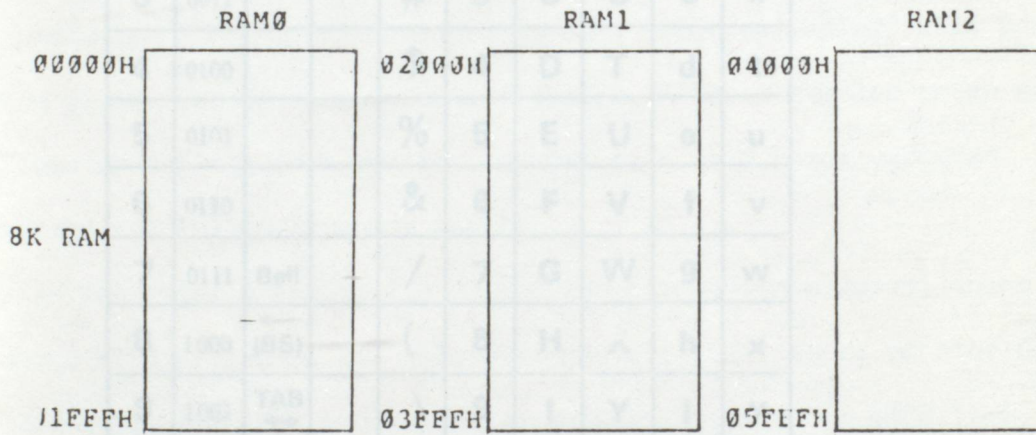
MULTITECH	
TITLE: KEYBOARD MATRIX	
UNIT OR	PROJECT NO.
ASSEMBLY	DATE
DRAWING NO.	PCB NO.
999-82484-00	P29-03110-1
REV	CHKD
APPROVED: [Signature]	



MULTITECH	
TITLE: LCP - PRINTED	
ASSEMBLY	SHEETS OF 6
NOF-1/08	
DRAWING NO	PCB NO
000-82405-00	P10-003110-1
BYN	CSB
APPD	

NOTE - IS 64 PIN CARD EDGE ONLY

BELLEK DÜZENİ



ASCII KOD TABLOSU

*** Table of ASCII Code

MSD LSD		0	1	2	3	4	5	6	7
		0000	0001	0010	0011	0100	0101	0110	0111
0	0000			space	0	@	P	'	p
1	0001			!	1	A	Q	a	q
2	0010			"	2	B	R	b	r
3	0011			#	3	C	S	c	s
4	0100			\$	4	D	T	d	t
5	0101			%	5	E	U	e	u
6	0110			&	6	F	V	f	v
7	0111	Bell		/	7	G	W	g	w
8	1000	← (BS)		(	8	H	X	h	x
9	1001	TAB ↔		)	9	I	Y	i	y
A	1010	LF		*	:	J	Z	j	z
B	1011		ESC	+	;	K	[	k	{
C	1100	FF		,	<	L		l	/
D	1101	CR		-	=	M	]	m	}
E	1110			•	>	N		n	~
F	1111	.		/	?	O		o	

MONİTOR KOMUTLARININ AÇIKLANMASI.

Bölüm-A."Line Assembler" ve Disassembler.

Komut A-"Line Assembler"a başlamak.

İsim - "ASSEMBLER"

Amaç - Bellekteki 8088 "assembler" dili programın kullanıcıya açıklamasına müsaade eder ve onları satır tarafından makina kodu dizisinde toplar.

Sözdizimi- (1) A

(2) A (adres)

Yorumlar : MPF-I/88'ın memory'sındaki 8088 "assembler" dili programına girmek için bu komut kullanılabilir. "line assembler"ından çıkmak için ve tam vaktinde monitöre dönmek için, bir komut yazmadan (carriage return) (CR) tuşuna basılır.

Adres : Eğer A komutu ile kullanılan parametrenin adresi yoksa yani A komutu sözdizimindeki (1) nolu gibi gösterilirse, sonra monitor CS yazmacının içeriklerinin dışından yararlanacaktır. (CS'i 0000 ihmal diğerine sahiptir.) ve programın yazılımını başlatmak için 0000 ihmal sonucunun ofset değeri olur. Sırası gelmişken; adres, ofset ilavesiyle bir segment adres veya bir ofset adreslerin her ikisinden biri olarak açıkça belirtilebilir.

Komut-L : "Disassembler"a başlamak.

İsim : "DISASSEMBLER"

Amaç : Bellekteki makina kodunun bir öbeğini 8088 "assembler" komutları şekline çevirir.

Sözdizimi : (1) L

(2) L (adres 1)

(3) L (adres 1) / (n)

(4) L (adres 1) (adres 2)

Yorumlar : Komutların belirli bir parçasının yazmayı veya A komutunu kullanarak programın tümünü yazmayı gerektirir. Daha önce komut-A bölümünde söylendiği gibi monitör program komutlarını otomatik olarak makina kodunda toplar. Bu bakımdan, monitor programınızı acaba tam olarak toplayıp toplamadımı diye merak edebilirsiniz. Monitöre makina koduna çevrilenlerin hepsini onların orijinal komutlar haline "disassembler" ettirmek için L komutundan yararlanmayı tavsiye etmelisiniz. Bu komut orijinal 8088 assembler komutlarına uygun biçimde istediğiniz gibi, bellekte depoladığınız makina kodlarının baytları kadar çevirmenize yardım eder. Uygunluk için, söz dizimindeki (1) nolu komut şeklinde makina bir komut verilirse 32 byte makina kodu kadar disassembler yapar. Ekranda gösterilen ve disassembler edilen makina kodlarının baytlarının miktarı istenilenden daha fazlası ihmal edilebilir. Bu komutlarının uzunluğunun değişken olduğu içindir. Bu nedenle, bu komut istenilen son komutun ilk byte ile sona erdiğinde, orta kalan komutlarında "disassembler" edecektir. Bundan başka, elbetteki adres parametresi 8088 makina kodlarını tam içeren bellek yerine göstermelidir. Eğer açıkça belirttiğiniz adres doğru komutun ilk baytını içermezse, "displayde" görülenler yanlış olacaktır.

Adres : Her adres bir ofset adres ilavesiyle bir segment adres ve bir ofset adresin herikisinden biri gibi açıkça belirtilebilir.

n : "Disassembler" edilen makina kodunun baytlarının sayısını belirtir.

Sözdizimi 1-

Bu komut monitor tarafından kullanılan iç gösterge vasıtasıyla belirli adresten makina kodunun 32 baytını "assembler" komut biçimine çevirecektir.

Sözdizimi-2.

Bu komut, komutta açıkça belirtilmiş adres parametresi (adres 1) vasıtasıyla gösterilen adresten makina kodunun 32 baytını assembler komut biçimine çevirir.

Bölüm B : Program yürütme komutları :

Komut G : Programı yürütme.

İsim : GO

Amaç : Bellekteki bir programı yürütmek

Sözdizimi:

(1) G

(2) G (adres)

Yorumlar : Adres bir ofset adres ilavesiyle bir segment adres veya bir ofset adresin herikisinden biri gibi açıkça belirtilebilir.

Eğer söz dizimi (1) de açıkça belirtildiği gibi adres yoksa, monitör yürütmeyi başlatacak adres değerini hesaplamak için komut göstericisinin (IP) içeriği yönünü ve segment değeri olarak CS registerının içerikleri yönünü kullanacaktır. Yürütme program bitinceye kadar veya "breakpoint" ile karşılaşınca kadar devam edecektir.

Komut S : Programı adım adım yürütme.

İsim : "STEP"

Amaç : Programda tek adım atmak veya komutların açıkça belirtilen sayıda yürütme ve sonra ekranda (displayde) yazmaçlar içeriklerini gösterilimi ile durmak. Yürütme komut göstericisi (IP) ve "code segment" CS yazmağı tarafından gösterilen adresten başlar.

Sözdizimi :

(1) S

(2) S n

Yorumlar :

Eğer sözdizimi (1)'de listelendiği gibi S komutunda açıkça belirtilen parametre yoksa, sonra monitor programın tek adım atmasını üzerine alır. Dikkat edilmelidir ki, yukarıda sözdizimi 2 deki komutta, n parametresi S komutuna kaydedildikten sonra yürütülecek komutun sayısını açıkça belirtir.

Bu komutu kullanmadan önce CS ve IP yazmaçlarının içeriklerinin istediğimiz adresi gösterdiğine emin olunmalıdır.

Bir komut veya açıkça belirtilen komutun sayısında yürütüldükten sonra, program durdurulacak ve yazmaç içerikleri gösterilecek ekranın boyutu yüzünden sadece iki yazmaç ve yürütülmekte olan sonraki komut bir defada gösterilecektir. Bu zaman, ekranı kaydırmanız için CTRL-A veya CTRL-Z tuşlarını kullanmalısınız. S komutu gösterilen yazmaç içeriklerini ihtiyaca göre uygun hale getirmek ve "lojik ekranın son hattında gösterilen komuttan başlamak için program yürütülmesine neden olur. "Return" tuşunu kullanma tam vaktinde monitor programına gire döndürecek.

Ekranda gösterilen 14 register vardır : AX, BX, CX, DX, SI, DI, IP, SP, FL (flag), BP, CS, DS, ES, ve SS.

S komutuna girdiğiniz her zaman, klayenin en üst soldaki yeşil LED monitör programının komutu yürüttüğünü ve kabul ettiğini göstermek için yalnız bir kez parıldayacaktır.

Bölüm : C : "Debugging."

Komut B- Programı kırma yerine gösterir ve yerleştirir.

Name: Programı kırma yeri.

Amaç : Üç tane program kırma yerine yerleştirir veya onların program akışındaki yerlerini gösterir. Program yürütüldüğü ve kırma yerine varıldığı zaman program yürütülmesi durdurulacaktır.

Sözdizimi :

(1) B

(2) B (n)

(3) B (n) (adres)

Yorumlar : Bu komut üç adresi programınıza kırma yeri olarak yerleştirmenize izin verir. O aynı zamanda genellikle programda yerleştirdiğiniz kırma yerlerini gösterebilir. Programı kırma yerleri programınızın kendine özgü yada durdurarak ve bellek içeriklerini, yazmaçların incelenmesine izin veren çok faydalı bir komuttur.

n : Program kırma yeri numarası. 1 ile 3 arasında değişir.

Adres : Program kırma yerinin adresini yerleştirmektedir. Program kırma yerine ROM bellekteki adrese yerleştirilmemesine dikkat edilmelidir. Sırası gelmişken, adres bir ofset ilavesiyle bir segment adres veya bir ofset adresin her ikisinden biri olarak açıkça belirtilebilir.

Komut- C- Program kırma yerlerini iptal eder.

İsim : Silme

Amaç : Daha önce yerleştirilen program kırma yerlerinin birini veya hepsini silmek.

Sözdizimi :

(1) C

(2) C (n)

Yorumlar : Daha önce sisteme yerleştirdiğiniz program kırma yerlerinin hepsini veya bir tanesini silmek için bu komut kullanılabilir. Eğer programınızı bitirdiğinizde ve aralıksız çalıştırmak için bu yöntemi kullanabilirsiniz.

n : 1 ile 3 arasında değişen bir sayıdır.

Sözdizimi- 1.

Bu komut genellikle programa yerleştirilen kırma yerlerinin hepsini silecektir.

Sözdizimi-2.

Bu komut C komutundan sonraki (n) tarafından açıkça belirtilen program kırma yerini siler.

Komüt.X : Yazmaçların içeriklerini değiştirir./inceler.

İsim : YAZMAÇLAR

Amaç : Yazmaçların herhangi birinin içeriklerini değiştirir veya gösterir.

Sözdizimi :

(1) X

(2) X (yazmaç ismi)

Yorumlar :

Sözdizimi-1.

X komutunun bu değişik biçimi bütün yazmaçlarının içeriklerinin gösterilmesine izin verir. X komutuna girildikten sonra dört yazmaçın içerikleri gösterilecektir. CTRL-Z tuşunu kullanırsanız sonraki dördünü görebilirsiniz. CTRL-A tuşunu kullanarak önceki dört yazmaçı geri göreceksiniz. "(Return)" tuşuna kullanarak tam vaktinde monitor programına geri dönebilirsiniz.

Yukarıda anlatılan tuşları kullanarak hep birlikte görülebilen 14 yazmaç vardır. AX, BX, CX, DX, SI, DI, IP, SP, FL (flag), BP, CS, DS, ES, ve SS

Sözdizimi-2.

X komutunun bu değişik biçimi bir tek yazmaçın değerini gösterir ve yazmaça yazdığınız değer tarafından yazmac içeriğinin yerine yenisini koyma işlemini verir.

Yazmaç isimleri aşağıdaki gibidir.

AP	BX	CX	DX
SI	DI	IP	SP
PL	BP	CS	DS
ES	SS		

Yukarıdaki 14 yazmaç arasında herhangi birinin içeriklerini değiştirmek için bu komutu kullanabilirsiniz.

Komut M : Belleğin içeriklerini değiştirme/inceleme.

İsim : "MEMORY"

Amaç : Bir bellek dizisinin veya bir bellek yerinin içeriklerini değiştirir veya gösterir.

Sözdizimi :

- (1) M
- (2) M (adres)
- (3) M (adres 1) (adres 2)
- (4) M (adres 1) (adres 2) |(veri 1) |
- (5) M (adres 1) (adres 2) (veri 2).....|
- (6) M (adres 1) (adres 2) | (veri 1) (veri 2)....|

Yorumlar :

Bu komutun işlevleri üç sınıfa ayrılabilir;

1. Bellek yerinin içeriklerini inceler veya gösterir.
2. Bellek dizisinin içeriklerini değiştirir veya gösterir.
3. Bellek yerlerini değerler ile doldurur.

Adres : Her adres bir ofset ilavesiyle bir segment adres veya bir ofset adresin her ikisinden biri olarak açıkça belirtilebilir.

Veri : Herhangi alfanumerik harfi veya her sayıyı gösterir. Eğer veriyi harf kullanıyorsanız, 'ABCD' gibi onların önüne bir virgöl konulmalıdır. Bununla birlikte eğer "hex" veri kullanıyorsanız virgüle ihtiyaç yoktur.

Bölüm-D- Kaydedilen belleği değiştirme veya inceleme.

Komut: I- Bellek yerinin içine veriye yerleştirir.

İsim : "INSERT"

Amaç : Bellek yerinin içine veriyi yerleştirir veya bellek yerinin bir kısmının içine yerleştirir.

Sözdizimi :

- (1) I
- (2) I /(veri 1) (veri 2)...../
- (3) I (adres 1)
- (4) I (adres 1) / (veri 1) (veri 2).../
- (5) I (adres 1) (adres 2)
- (6) I (adres) (adres 2) /(veri 1) (veri 2).../

Yorumlar : Belleğin içine veriyi yerleştirmek için bu komutu kullanmalısınız. Yerleştirme işleminizi yerine getirdikten sonra, aşağıda kalan bellek yerinin içerikleri verinin yerleştirilmesiyle otomatik

olarak yerleřtirilen verinin uzunluęunda geriye doęru kayacaklardır.

Adres : Her adres bir ofset ilavesiyle bir segment adres veya bir ofset adresin her ikisinden biri olarak açıkça belirtilebilir.

Veri : Herhangi bir alfanumerik harfi veya "hex" sayıyı gösterir. Eęer veriyi harf kullanırsanız. 'ABCD gibi, onların önüne bir virgöl konulmalıdır. Bununla birlikte, eęer her veri kullanırsanız virgüle ihtiyacı yoktur.

Sözdizimi-1.

Bu komut yön segmentinin iç göstericisi tarafından gösterilen adres olan bellek yerinin içine sıfır değeri ile bir bayt yerleştirecektir. Aşağıda kalan asıl verilerin hepsi verinin yerleřtirilmesiyle bir bayt geriye doęru kayacaktır.

Sözdizimi-2.

Bu komut yön segmentinin iç göstericisi tarafından gösterilen ad-
resden başlayarak belleğin bir kısmının içine bir veya daha fazla ve-
rinin baytlarını yerleřtirir. Aşağıda kalan asıl verilerin hepsi son
yerleřtirilen veriyle yerleřtirilen verinin uzunluęuyla geriye doęru
kayacaktır.

Sözdizimi-3.

Bu komut kendine özgü bir adresin dışında işlerde sözdizimi 1'e benzerdir. Komuttaki (adres 1) adres tarafından açıkça belirtilen bellek yerinin içine sıfır değerli verinin bir baytını yerleřtirmek için bu komutu kullanabilirsiniz. Aşağıda kalan asıl veri verinin yerleřmesiyle geriye doęru bir bayt kaydırılacaktır.

Sözdizimi-4.

Bu komut kendine özgü bir adresin dışında işlerde sözdizimi 2'ye çok benzerdir. Komuttaki (adres 1) tarafından açıkça belirtilen adres ile başlayan belleğin bir bölümünün içine bir veya daha fazla veri baytını yerleştirmek için bu komutu kullanabilirsiniz. Aşağıda kalan asıl verilerin hepsi son yerleştirilen veriyle yerleştirilen verinin uzunluğu tarafından geriye doğru kaydırılacaktır.

Sözdizimi-5.

İşlevde sözdizimi 1'e benzer. Bu komut komuttaki (adres 1) birinci adres tarafından açıkça belirlenmiş bellek yerinin içine sıfır değerli verinin bir baytını yerleştirir. İkinci adres dahil olmak üzere veri yerleştirilen önceki bellek yeni içeriğini ikinci adrese kadar bir bayt geriye doğru kaydırılacaktır.

Sözdizimi-6.

İşlevde sözdizimi 2'ye benzerdir. Bu komut, komuttaki (adres 1) birinci adres tarafından açıkça belirtilen adres ile başlayan belleğin bir kısmının içine bir veya daha fazla veri baytının yerleştirmenize müsaade eder. İkinci adres dahil olmak üzere veri yerleştirilen bellek yerininin önceki içeriklerini ikinci adrese kadar yerleştirilen veri uzunluğu kadar geriye doğru kaydıracaktır.

Komut D : Bellek'deki veriyi siler.

İsim : Silme

Amaç : Bellekteki verinin bölünmüş bir parçasını veya verinin bir baytını silmek.

Sözdizimi :

- (1) D
- (2) D / (n)
- (3) D (adres 1)
- (4) D (adres 1) / (n)
- (5) D (adres 1) (adres 2)
- (6) D (adres 1) (adres 2) / (n)

Yorumlar : Bellekteki verinin bir bölümünü veya verinin bir bay-
tını silmek için bu komutu kullanabilirsiniz. Silme işlemini yerine ge-
tirdikten sonra, aşağıda kalan bellek yerlerinin içerikleri verinin
silinmesiyle otomatik olarak silinen veri uzunluğunda ileri doğru kay-
dırılacaktır. Bununla birlikte, eğer bu komutta ikinci adresi açıkça be-
lirtirseniz, o zaman ikinci adres ve silinen veri dizisinin sonu arasın-
daki veri boş olan alanı doldurmak için ileri doğru taşınacaktır.
Eğer yalnız bir adres açıkça belirtilmişse, silinen dizinin sonundan yön
segmentinin sonuna kadar olan verinin hepsi ileri doğru taşınacaktır.

Adres : Her adres bir ofset ilavesiyle bir segment adres veya
bir ofset adresin her ikisinden biri olarak açıkça belirtilebilir.

n : Silmeye gittiğiniz veri baytlarının sayısını gösterir. "Hex"
sayıdır.

Sözdizimi-1.

Bu komut yön segmentinin iç göstericisi tarafından gösterilen adres-
deki verinin bir baytını silecektir. Verinin silinmesiyle aşağıda kalan
asıl verinin hepsi bir bayt ileri doğru kaydırılacaktır.

Sözdizimi-2.

Bu komut, komuttaki (n) tarafından açıkça belirtilen bir veya daha

fazla veri ^{baytını} baştını silecektir. Silinmekte olan veri baytlarının başlama adresi yön segmentinin iç göstericisi tarafından gösterilir. Silme işlemine uygulanmasından sonra silinen son veriyle aşağıda kalan asıl verinin hepsi silinen verinin uzunluğu kadar ileri doğru kaydırılacaktır.

Sözdizimi-3.

Komuttaki kendine özgü bir adresin dışında; bu komut işlevde sözdizimi 1'e çok benzerdir. Komuttaki (adres) tarafından açıkça belirtilen adresteki verinin bir baytını silmek bu komutu kullanmalısınız. Veri silinmesiyle aşağıda kalan asıl verinin hepsi bir bayt ileri doğru kaydırılacaktır.

Sözdizimi-4.

Komuttaki kendine özgü bir adresin dışında bu komut işlevde sözdizimi 2'ye çok benzer. Bir veya daha fazla veri baytını silmek için bu komutu kullanmalısınız. Komuttaki (n) silinmekte olan baytların sayısını açıkça belirtir ve silinmekte olan verinin başlama adresi komuttaki (adres 1) tarafından açıkça belirtilir. Silinen son veriyle aşağıda kalan asıl verinin hepsi silinen veri uzunluğu kadar ileri doğru kaydırılacaktır.

Sözdizimi-5.

İşlevde sözdizimi 1'e benzerdir. Bu komut, komuttaki birinci adres (adres 1) tarafından açıkça belirtilen adresde olan verinin bir baytını siler. Silinmesi istenen son bellek yerinden ikinci adres dahil olmak üzere ikinci adrese kadar olan veri silinmesi istenen yere bir bayt kaydırılacaktır.

Sözdizimi-6.

İşlev açısından sözdizimi 2'ye benzer. Bu komut bir veya daha fazla veri baytını silmenize izin verir. Komuttaki (n) silinmekte olan baytların sayısını açıkça belirler ve silinmekte olan verinin başlangıç adresi komuttaki birinci adres (adres 1) tarafından gösterilir. Silinmesi istenen son bellek yerinden ikinci adres dahil olmak üzere ikinci adrese dahil olmak üzere ikinci adrese kadar olan veri silinmesi istenen yere en fazla silinen veri uzunluğu kadar kaydırılacaktır.

Komut-F- Bellekdeki bir harler dizisini bulur.

İsim : Arayıp bulma.

Amaç : Açıkça belirtilen bir değer veya bellekteki diğer grubu için araştırma yapar.

Sözdizimi:

- (1) F / (veri dizisi)
- (2) F (adres 1) / (veri dizisi)
- (3) F (adres 1) (adres 2) / (veri dizisi)

Yorumlar :

Bu komut açıkça belirtilen bir değer veya bellekdeki değer grubu için araştırma yapan ve onların bulunduğu yer adresini bildirir. Eğer bellekte araştırılması olan değere veya değerlere bir çok defa rastlanıldığı süre içinde, sistem ekranda her olayın adresini gösterecektir.

İstediğiniz bellek başlangıcından belleğin sonuna kadar veya sadece komuta kaydettiğine bağlı olan açıkça belirtilmiş bellek dizisinin sınırları içinde diziye arayabilirsiniz.

Adres : Her adres ofset ilavesiyle bir segment adres veya bir ofset adresin her ikisinden biri olarak belirtilebilir.

Veri Dizisi :

Herhangi alfanümerik harfi veya "hex" sayıyı gösterir. Eğer veriyi harf kullanırsanız, ABCD gibi onların önüne bir virgül konulmalıdır. Eğer veriyi "hex" kullanırsanız, virgüle ihtiyaç olmaz.

Sözdizimi-1.

Bu komut (S ilavesiyle iç gösterici yönü ile gösterilen bellek yerinden yön segmentinin sonuna kadar olan yol boyunca komuttaki (veri dizisi) tarafından belirtilen değeri veya değerleri araştırır.

Sözdizimi-2.

Bu komut, komutta adres parametresi (adres 1) tarafından belirtilen adres olan bellek yerinden başlayarak yön segmentinin sonuna kadar olan yol boyunca komuttaki (veri dizisi) tarafından belirtilen diziyi araştırır.

Sözdizimi-3.

Bu komut, komuttaki (adres 1) tarafından belirtilen adres olan bellek yerinden başlayarak, komuttaki (adres 2) tarafından belirtilen bellek yerine kadar komuttaki (veri dizisi) tarafından belirtilen değer veya değerleri araştırır.

Komut-J : CS ve IP Yazmaçlarına yeni değeri taşır.

İsim : Sıçrama

Amaç : Gelecek defa programı yürütmeye başlatmak istediğiniz özel adrese doğrudan doğruya sıçramak.

Sözdizimi :

(1) J (adres)

Yorumlar :

Eğer adres sadece bir ofset olarak belirtilirse, ofset değeri IP yazmacının içeriklerinin yerini tutacaktır. Eğer segment değeri ve ofset değerinin her ikisi de açıkça belirtilirse, o zaman, onlar herbiri söylendiği sıraya göre CS' yazmacının ve IP yazmacının içeriklerinin yerini tutacaklardır.

Adres :

Adres bir ofset ilavesiyle bir segment adres veya bir ofset adresin her ikisinden biri olarak belirtilebiliriz.

Komut-T : Bellek dizisinin içeriklerini bellekte belirtilen alana kopya eder.

İsim : Bir yerden başka yere taşıma.

Amaç : Bellek içerikleri dizisini bellekteki başka bir alana kopya etmek.

Sözdizimi :

(1) T (adres 1) (adres 2) (adres 3)

(2) T (adres 4) (adres 5) / (n)

Yorumlar :

Sözdizimi-1.

Komutta (adres 1) ile başlayan ve (adres 2)'de sona eren bellek içerikler dizisini komutta (adres 3) ile başlayan bellek yerleri dizisine kopya etmek için bu komutu kullanabilirsiniz.

adres 1 : Kopya olan bellek dizisinin başlama adresi

adres 2 : Kopya olan bellek dizisinin bitiş adresi

adres 3 : Kopya olan bellek dizisinin gönderileceği yer adresi.

Sözdizimi-2.

Bu komut anlam olarak sözdizimi 1'e tam benzerliğe sahiptir. Onların arasındaki fark etkilenen bellek dizilerini tanımlamadaki yoldur. Bu komutta (adres 4) kopya edilen dizinin başlangıç adresi, (adres 5) kopyanın taşınacağı adresin başlangıç adresidir. Kopya edilen baytların sayısında belirtilir.

adres 4 : Kopya edilen veri baytlarının sayısının başlama adresidir.

adres 5 : Kopya edilen veri baytları sayısının gideceği yerin adresidir.

n : "Hex" değeri 1 ile FF arasında değişen kopya edilen data baytlarının sayısidir.

Dikkat edilmelidirki, adres bir ofset ilavesiyle bir segment adres veya bir ofset adresin her ikisinden biri olarak belirtilebilir.

Bölüm - E : Çeşitli I/O komutları :

Komut- P : Ekranda gösterilenlerin hızını ayarlama.

İsim : Duraklama

Amaç : Ekranda gösterilenlerin hızını ayarlamak.

Sözdizimi : P (n)

Yorumlar :

Ekranda gösterilenlerin hızını ayarlamak için bu komutu kullanmalısınız. Monitor her karakteri gösterme arasında belirtilen zaman süresinde duraklayacaktır. Verinin bir ekran dolusundan daha fazlasını göstermek için işlev olan M komutu veya diğer komutlar kullanıldığı zaman bu komutun faydalı olduğuna karar verebilirsiniz.

n : 0 ile FF arasında değişen "hex" sayıyı gösterir. Çok büyük sayılarda gösterme hızını çok yavaş yapacaktır. Eğer FF sayısını belirtirseniz, sistem her harfi gösterme arasında aşağı yukarı 5 saniye duraklayacaktır.

Komut- N : Porttan verinin bir baytının ekranda gösterir ve giriş yapar.

İsim : Giriş yapma

Amaç : Açıkça belirtilmiş porttan verinin her bir baytını göstermek ve giriş yapmak.

Sözdizimi : N (port adresi)

Yorumlar :

Bu komutu kullanmadan önce, cihaza gerçekten ayrılan giriş yapacağınız port adresi olduğuna emin olunmalıdır. Her cihazın adresleri ve I/O portlarının tanımlanması için aşağıdaki bölüme ilişkin olur.

port_adresi : Tanımlanan porta uygun - 0 ile FFFF arasında değişen "hex" sayıyı gösterir.

Komut- D : Çıkış portuna bir veya daha fazla veri baytının gönderir.

İsim : Çıkış yapma

Amaç : Tanımlanan çıkış portuna bir veya daha fazla veri baytlarını göndermek.

Sözdizimi : D (port_adresi) / (veri)

Yorumlar :

Bu komut, komutta açıkça belirtilen çıkış portuna bir veya daha fazla veri baytlarını bir defada gönderebilir. Onların port adreslerine uygun ve I/O portalırını tanımlama için aşağıdaki bölüm ilişkin olur.

port_adresi : Tanımlanan porta uygun D ile FFFF arasında değişen "hex" sayıyı gösterir.

Veri : Her sayıyı veya herhangi bir alfabetik harfi gösterir. Eğer harf veri kullanırsanız.'ABCD gibi onların önüne bir virgül konulmalıdır. Bununla birlikte her veri kullanırsanız virgüle ihtiyaç yoktur.

Komut-E : Sistem "konsolu" olarak dış terminali tespit eder.

İsim : "EXTEND"

Amaç : MPF-I/88 için "konsolu" olarak dış terminale izin vermek.

Sözdizimi : E (n)

Yorumlar :

Bu komut MPF-I/88'in "konsolu" olarak (gösterme ekranı, klavye) kullanılan dış terminale izin verir. Hemde MPF-I/88'in kendi klavye ve ekranını geri kontrole dönmek için işlemi vardır. Çünkü onların aralarında arabirim olarak kullanılan ACA (Asynchronous Communications Adapter) olan dış terminal MPF-I/88 ile haberleşirler. Bu komut kullanıldığı zaman MPF-I/88'in sistem kartının sol-enüst köşesindeki oluğa ACA kartı yerleştirmek asla unutulmamalıdır.

Bu komutu kullanmadan önce, MPF-I/88'de olan aşağıdaki parametreleri kullandığınız arabirimde oluşturulmalıdır.

"Baud Rate" 9600

"Parity"- even (çift)

Veri bitleri-7

Durma bitleri-2

n : Aletin sistem kontrolunu bu değer belirtir.

Ø : MPF-I/88 kendi sistemindedir.

1 : CRT terminalin seriport 1'e bağlandığını gösterir.

2 : CRT terminalin seriport 2'ye bağlandığını gösterir.

Komut-W- Teybe bilgi yazar.

İsim : Yazmak

Amaç : Teybe bellek dizisinin içeriklerini kaydeder.

Sözdizimi : W (adres 1) (adres 2)/(dosya--ismi)

Yorumlar :

Adres 1 : Teybe kaydolan bellekteki veri dizisinin başlangıç adresi.

Adres bir ofset ilavesiyle bir segment adres veya bir ofset adresin her ikisinden biri olarak belirtilebilir.

Adres 2 : Teybe kaydolan bellekteki veri dizisinin son adresi.

Adres bir ofset ilavesiyle bir segment adres veya bir ofset adresin her ikisinden biri olarak belirtilebilir.

Dosya_ismi : Dosya isminin harf uzunluğu 1 ile 8 arasında olabilir.

Herhangi bir dosya ismini büyük veya küçük harf yazabilirsiniz. Dosya isminin önüne virgöl konulmalıdır.

Note : (Adres 1) ve (adres 2) adreslerinin her ikisi benzer segment dizisinin içinde olmalıdır. W komutunun satırındaki segment adresleri farklı ise geçersizdir. Örneğin "W 0080 : 0 0090 : A0 /"TEST" komutu hata ile sonuçlanır.

Komut- R : Teyp'ten bilgi okur.

İsim : Okuma

Amaç : Teyp'ten bilgi okumak

Sözdizimi :

(1) R / (dosya ismi)

(2) R (adres) / (dosya ismi)

(3) R

(4) R (adres)

Yorumlar :

Dosya_ismi- Dosya isminin harf uzunluğu 1 ile 8 arasında olabilir. Herhangi bir dosya ismi büyük veya küçük harflerle yazılabilir. Dosya isminin önüne virgül konulmalıdır.

Adres : Belirttiğiniz dosyanın bellekteki bölgesinin başlangıç adresini gösterir. Adres bir ofset ilavesiyle bir segment adres veya bir ofset adresin her ikisinden biri olarak belirtilebilir.

Eğer dosya ismi okuma süresi esnasından sistem tarafından tamamiyle bulunursa, aşağıdaki gibi mesaj verilecektir : (İsim vermek istenilen dosya ismi "TEST" olarak kabul edilirse.)

READ FILE : TEST (Okunan dosya)

Teypinizde birçok dosya ismi bulunabilir ve ilk dosya isteğiniz dosya olmayabilir. Eğer sistem belirtiniz bir diğer dosya isimlerine karşılırsa, onlara önem vermeyecek ve aşağıdaki mesajları tamamiyle gönderir.

SKIP FILE : XXXXX (Okunmadan geilen dosya)

NOT : " XXXXX" Karşılaştığı herhangi bir diğ er dosyanın dosya ismini g sterir.

Sistem komutla belirtilen dosya ismini buluncaya kadar araştırmaya devam edecektir.

S zdizimi-1.

Eğ er sistem belirttiğ iniz dosyaya karşılaşırsa, dosyayı banda aldığ ınız zaman kullandığınız başlama adresiyle aynı tutulan bellek yerine onu y kleyecektir.

S zdizimi-2.

Komutun bu değ işik biğ imi işlevde s zdizimi 1'e ok benzerdir, sadece farkları komutta belirtilen y klemesi yapılan bellek dizisinin başlama adresidir.

S zdizimi-3.

Bu komut kullanılırsa, sistem herhangi bir dosya ismine  nem vermez. Teypte  nce karşılaşılan herhangi bir dosya belleğ e y klenilecektir. Y klemesi yapılan bellek yerleri dizisinin başlangıç adresi dosyayı banda aldığ ınız zaman kullandığınız adres tarafından belirtilir.

S zdizimi-4.

Bu komut kullanılırsa, sistem, tarafınızdan belirtilen y klemesi yapılan bellek yerleri dizisinin başlangıç adresi hariç s zdizime 3'e benzeyen işlevi yerme getirir.

Çok önemli :

Yukarıdaki dört işlemten herhangi birini yerine getirdikten sonra, programı yüklerken kullandığınız adresle göreceli olup olmadığını kontrol etmeyi hatırlayınız. JMPF ve CALLI gibi komutlar, belirttiğiniz segment adresin sınırları içinde ofset değerle karşılaşırlar.

KONTROL HARFLERİ :

Kontrol harfleri basılacak bir tuşu tanımlamadan önce CTRL veya ALT tuşları aşağıda tutulurken girilir. Her kontrol harfi kendine özgü bir işlevi yerine getirir. Kontrol harflerinin özeti ve onların işlevleri aşağıda tanımlanır.

*(CTRL-S) : Gösterici çıkışını geçici olarak durdurur. Herhangi bir tuşa basma göstergenin çıkışını yeniden başlatır.

*(CTRL-P) : Bu kontrol harfi iki durumlu bir anahtardır. Bu kontrol harfini yalnız bir kere yazdırma yazıcıyı açar. Yazıcıya ekran çıkışını göndermeye neden olur. Bu kontrol harfine ikinci defa basma yazıcıyı kapatır.

*(CTRL-X) : Yazılmakta olan satırı iptal eder.

*(ALT-Y) : Sistem kontrolundan monitor programına geri döner.

Komuttaki kursorü hareket ettiren kontrol harfleri :

*(ALT-E) : Kursorü bir satır yukarı hareket ettirir.

*(ALT-X) : Kursorü bir satır aşağı hareket ettirir.

*(ALT-S) : Kursorü bir pozisyon sola hareket ettirir.

*(ALT-D) : Kursorü bir pozisyon sağa hareket ettirir.

*(ALT-F) : Kursorü baştaki ilk durumuna geri döner.

Ekranı kaydırma için kontrol harfleri :

*(ALT-A) : Ekranı aşağı kaydırır.

*(ALT-Z) : Ekranı yukarı kaydırır.

*(ALT-Q) : Kürsör ekran kaydırılmadan önceki yerine geri döner.

FREKANSMETRE :

Frekansmetre sadece 8088'in komutlarının kullanışına örnek olsun diye yazılım olarak gerçekleştirilmiştir. Frekansı ölçülecek işaret MPF-I/88'e yazıcı portunun BUSY ucundan girilmiştir. Frekansmetre üç bölümde incelebilir.

a- Gelen darbeleri sayma

b- Sayılan darbeleri "binary" olan "BCD"ye çevirme

c- Bu verileri "displayde" gösterme.

Frekansmetrenin Çalışma Prensibi :

Frekansmetrenin prensibi birim zamanda gelen darbeleri sayar. Programın gelen darbeleri sayma bölümü düşen kenara göre çalışmaktadır.

Programın en başında program çalışırken "kürsörün" ortadan kalkması için 1A0H nolu porttan 0CH verisi atılır. Bu durumda "Cursor off" olur. Örneklemeye sayısını CX yazmaçı belirler. Bu yazmaçtaki sayı program akışında en fazla örnek alabileceği sayı kadar olmalıdır. BX yazmaçı sayılan darbeleri içerir. Program çalıştığında örneği alınan darbenin 1 veya 0 olduğuna bakılır. Bunun için 1C0H nolu porttan bir giriş yapılır. 1C0H nolu portun 7.biti BUSY'i gösterir. AH yazmaçı örneği alınan darbenin 1 veya 0 olduğunu gösterir. AH 0 ise örneği alınan darbe o anda 1, FFH ise o darbe 0 dır. Giriş yapıldıktan sonra XOR işlemi ile AH yazmaçı sıfırlanır. AL'nin 7.biti (BUSY) test edilir. Eğer gelen darbe "1" ise AH'nın evriği alınarak AH'ya FFH yerleştirilir. Sonra yine bir örnek daha alınır. Yine AL'nin 7.biti test edilir. Eğer bu bit "1" ise AH tekrar sıfırlanır. Program buraya sıçrarken AH'nın içeriği FFH'de

olabilir. AH sıfırlandıktan sonra örnek alan sayıcı CX bir azaltılır. Eğer örnek alma sayısı bitmemiş ise tekrar girişten bir örnek daha alınır. Bu alınan örnek 0 olmuş ise düşen darbe olabilir. Darbenin önceki durumunu değerlendirip BX bir arttırılır. Darbenin bir önceki durumu halen 0'da ise BX'i arttırmaz. Çünkü darbe 1'e çıkıp 0'a düşmemiştir. Önceki durumu halen 0'dadır. Ancak düşen darbe olursa BX arttırılır. Bu işlem CX sıfır olana kadar devam eder. BX'in son içeriğini "BCD" şekle getirmek için önce 200 H adresine yerleştirilir. BX'e "BCD" bilginin yerleşeceği yerin adresi yüklenir. DX yazmaçının DH yazmaçında "binary"bilginin kaç bayt olduğu ve DL yazmaçında bu binary bilginin BCD'ye dönüşmesiyle yerleşeceği bayt sayısını gösterir. BCD bilginin yerleşeceği 3 bayt sıfırlanır. 2 baytın bit sayısı CH yazmaçına yerleştirilir. 16-bitlik binary bilgi CF'in içine bir bit sağa kaydırma yapılır. 3 baytlik "carry" olarak toplama yapılacağı için DL'nin içeriği DL yerleştirilir. BX'in göstermiş olduğu BCD bilgi baytı AL'ye yerleştirilir. Daha sonra "carry" ile beraber AL ile AL toplanır. Böylece bit ağırlığı bu sayede sağlanmış olur. Eğer 3.bitte "1" var ise üst üste 2 defa toplama yapılarak bu bitin değeri olan 4 sayısı elde edilir. Toplamadan hemen sonra DAA komutu kullanılmıştır. Bu komut toplama sonucu AL'de oluşan sayıyı "binary" düzenden BCD düzene çevirir. Örneği $16+16 = 20H$ ise DAA komutundan sonra AL'de 32 sayısı oluşur. "Binary" sayı BCD düzene çevirildikten sonra ekrana bu sayıları çıkarmak için ASCII düzene geçmek gereklidir. Önce ilk sekiz bitin alt dört biti ASCII'ye dönüştürülür. Bunun için AL önce 0F ile AND işlemine tabi tutulur. Bu değere 30 eklenerek ilk dört bit ASCII'ye dönüştürülen AL'nin en ağırlıklı üst dört bitini ASCII'ye dönüştürmek için bitler 4 defa sola kaydırılır. Yine bu değere 30H eklenerek ASCII'ye

dönüştürme yapılır.

Ekрана bu bilgileri çıkarmak için INT 09 alt programı kullanılır. INT 09 alt programı SI'in gösterdiği yerden itibaren ekrana karakteri çıkartır. LODSB komutu kullanılarak SI yazmaçının gösterildiği yerden AL'ye yükleme yapılır. AL sıfır ise programın birinci çevrimi bitmiştir. Tekrar frekans ölçmek için en başa döner. AL sıfırdan farklı ise ekrana yazı çıkarmaya devam eder. Karakterin ekrana çıkmasıyla SI bir artar, SI'nın artması için DF' sıfırlanmıştır. F8H'den 100'e kadar "FREKANS=" verisi, 100, 104 arası ASCII BCD bilgi, 105'den 108'e kadar "Hz" verisi ve 109 nolu adreste 00 verisi vardır. Bu 00 verisi sayesinde yazma işlemi her defasında sona erer. Tekrar aynı işlemler tekrarlanır.

0018 IN AL, 0A:FC00

0019 TEST AL, 40

001A JNC 0027

001B JMP AL, FF

001C JZ 0025

001D INC BX

001E MOV AL

001F JMP 0028

0020 MOV BX, 00

0021 DEC BX

0022 JNZ 002E

0023 JMP 0024

0024 MOV I-0001, BX

0025 MOV BX, 0000

0026 MOV BX, 0000

0027 JZ AL, 00

0080 :
0000 MOV DX, 01A0
0003 MOV AL, 0C
0005 OUT DX, AL: F000
0006 MOV CX, DA00
0009 MOV BX, 0000
000C MOV DX, 01C0
000F IN AL, DX: F000
0010 XOR AH, AH
0012 TEST AL, 40
0014 JNE 0018
0016 NOT AH
0018 IN AL, DX : F000
0019 TEST AL, 40
001B JNE 0027
001D CMP AH, FF
0020 JE 0029
0022 INC BX
0023 NOT AH
0025 JMP 0029
0027 XOR AH, AH
0029 DEC CX
002A JCXZ 002E
002C JMP 0018
002E MOV (+0200),BX
0032 MOV BX, 0208
0035 MOV DX, 0203
0038 XOR AX, AX

003A MOV (0208), AX
003D MOV (020A), AL
0040 MOV AL, 08
0042 MUL DH
0044 MOV CH, AL
0046 RCL WORD (+0200), 1
004A MOV CL, DL
004C MOV AL, (BX)
004E ADC AL, AL
0050 DAA
0051 MOV (BX), AL
0053 INC BX
0054 DEC CL
0056 JNE 004C
0058 DEC CH
005A MOV BX, 0208
005D JNE 0046
005F MOV CH, 03
0061 MOV DI, 0104
0064 MOV AL, (BX)
0066 AND AL, 0F
0068 OR AL, 30
006A MOV (DI), AL
006C DEC CH
006E JE 007F
0070 MOV AL, (BX)
0072 MOV CL, 04

0074 SHR AL, CL

0076 DEC DI

0077 OR AL, 30

```
0079 MOV     (DI), AL
```

0078 DEC DI

```
007C INC BX
```

```
007D JMP 0064
```

007F CLD

```
0080 MOV     SI, 00F8
```

```
0083 MOV    (+0109), CH
```

0087 LODSB

```
0088 CMP     AL, 00
```

008A JE 0090

008C INT 09

```
008E JMP 0087
```

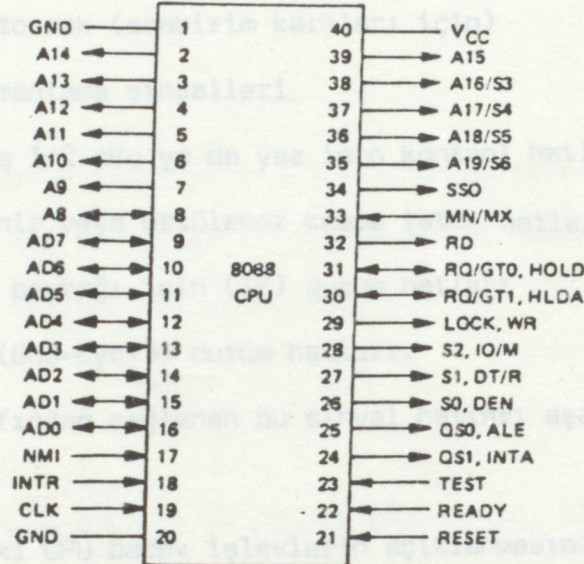
```
0090 JMP 0000
```

00F8 DW "FREKANSMETRE="

00105 DW "Hz"

00108 DB 00

CPU BACAK BAĞLANTILARI



Pin Name	Description	Type
AD0-AD7	Address/Data Bus	Bidirectional, tristate
A8-A15	Address Bus	Output, tristate
A16/S3, A17/S4	Address/Segment identifier	Output, tristate
A18/S5	Address/Interrupt enable status	Output, tristate
A19/S6	Address/status	Output, tristate
SSO	Status output	Output, tristate
RD	Read control	Output, tristate
READY	Wait state request	Input
TEST	Wait for test control	Input
INTR	Interrupt request	Input
NMI	Non-maskable interrupt request	Input
RESET	System Reset	Input
CLK	System Clock	Input
MN/MX	= GND for a maximum system	
S0, S1, S2	Machine cycle status	Output, tristate
RQ/GT0, RQ/GT1	Local bus priority control	Bidirectional
QSO, QSI	Instruction queue status	Output
LOCK	Bus hold control	Output, tristate
MN/MX	= VCC for a minimum system	
IO/M	Memory or I/O access	Output, tristate
WR	Write control	Output, tristate
ALE	Address Latch enable	Output
DT/R	Data transmit/receive	Output, tristate
DEN	Data enable	Output, tristate
INTA	Interrupt acknowledge	Output, tristate
HOLD	Hold request	Input
HLDA	Hold acknowledge	Output
VCC, GND	Power, ground	
Maximum System Signals		Minimum System Signals

8088 Pins and Signal Assignments

CPU BACAK İŞLEV AÇIKLAMASI :

8088 CPU aşağıdaki bacakları içerir.

- 1- 8- bit'lik sistem veri yolu
- 2- 20- bacaklı adres yolu
- 3- Kaynak ve toprak (arabirim kartları için)
- 4- Saat ve zamanlama sinyalleri
- 5- Bellek yada I/O oku ya da yaz için kontrol hatları
- 6- Örtülebilir veya örtülemez kesme istek hatları
- 7- Kesme izin bayrağı için (IF) durum hatları
- 8- Yol dönüş (Bus-cycle) durum hatları.

8088 CPU tarafından sağlanan bu sinyal hatları aşağıdaki gibi açıklanır.

NOT : Aşağıdaki CPU bacak işlevlerin açıklanmasında "I" sinyal hattının sistem yoluna bir giriş sinyali olduğunu göstermek için kullanılır. "O" sistem yoluna bir çıkışı gösterir. ("I/O" bir giriş yada bir çıkış hattının herhangi biri olabilen bir sinyali gösterirken)

$\overline{MN}/\overline{MX}$ (I) :

Bu bacak 8088'in minimum yada maksimum modda çalışıp çalışmadığını kontrol etmek için kullanılır. Eğer 8088 maksimum modda çalışıyorsa 8087 gibi diğer yardımcı işlemcilerle (coprocessors) kapalı etkileşimli çalışabilir. Eğer bu bacak 1'e çekilirse, bu durumda 8088 minimum modda çalışır. Eğer 0'a düşürülürse CPU maksimum modda çalışır. MPF-I/88-in minimum modda çalıştırılması için, bu bacak 1'e çekilir.

$\overline{D0} - D7$ (I/O) :

Bu hatlar sistem veri yoluna bağlanır. Bu bacaklar tri-state (üç durumlu) sinyal hatlarıdır. Veriyi CPU'ya veya CPU'dan taşımak için kullanılır.

$A0 - A19$ (O) :

Bu hatlar 1 Megabayt'lık belleğin üstünde adreslenebilen sistem adres yoluna bağlanır. Bu hatlar tri-state sinyal hatlarıdır. Ve işleyici (processor) tarafından oluşturulurlar.

READY (I) :

Bu hattı, CPU'nun bir yol dönüşünde (Bus-cycle) ilave edilen bekle durumlarını tespit edebilmesi için bir "not ready" durumu göstermede I/O donanımları yada yavaş bellek (slow memory) tarafından kullanılır.

$\overline{TEST}$ (I) :

Bu bacak, CPU'nun bekle (WAIT) komutunu yürütmesiyle esneklenir. Eğer test edildiğinde 0'daysa WAIT komutunu izleyen komut yürütülür. Eğer 1'de ise CPU duracaktır.

$\overline{RD}$ (O) :

Bu bacak aktif 0'dadır. CPU'nun bir bellekten yada I/O donanımından bilgi okuyor olduğunu göstermek için 0'a çekilir.

$\overline{WR}$ (O) :

Bu bacak aktif 0'dadır. CPU'nun bir belleğe yada I/O donanımının bilgi yazıyor olduğunu göstermek için 0'a çekilir.

$\overline{IO}/M$ (0) :

Bir bellek erişimi için 1'e çekilir. Bir I/O erişimi için 0'a çekilir.

HOLD (1) :

Bu bacak sistem yol servis isteği için bir harici donanım tarafından genellikle ana yol (bus master) tarafından kullanılır. Eğer veri transferi için bir harici donanımın sistem yoluna ihtiyaç varsa, bu bacak 1'e yükselecektir. Bu bacakta bir "1" meydana çıkmasına bağlı olarak CPU akım yol dönüşünü tamamlayacak ve HLDA bacağından bir hold talimatı dışarı çıkaracaktır. Bir harici donanımdan gelen (HOLD) tutma sinyali harici donanımın donanım ihtiyacı için geçici olarak sistem yolunu ayırmak için CPU'yu ister.

HLDA (0) :

Bu sinyal aktif 1'dedir. Bu sinyal CPU, bir harici donanımdan gelen bir tutma isteğini (HOLD) onayladığında 1'e yükseltilir. Bir kez tutma isteği onaylandığında, CPU I/O donanımların sistem yolunun kontrolünü harici donanımların üzerine olmasına izin vermesi için ve bellek ve kendisi arasındaki bağlı olmayan sistem tarafından harici donanım isteği için yol kontrolu bırakır.

ALE (0) :

Bu bacak, (Adress latch Enable) aktif 1'dedir. Ve sabit adres kitleme (latchleme) için 8088 tarafından üretilir.

SSØ (0) :

Bu tri-state çıkış çevresel cihazlar için yol dönüş durumunu temin etmekte diğer iki sinyal hattıyla kullanılır. Yol dönüş durumunu tespit eden diğer iki sinyaller IO/M ve DT/R'dir.

DT/R (0) :

Ek olarak, yol dönüş durumunu temin etmek için, 3-state çıkış hatlı CPU ve veri yastık arasındaki veri transferinin yönünü belirler.

Eğer bu bacak 1'e yükseltirirse, bilgi CPU'dan veri yastığına transfer edilir. Eğer bu bacak 0'a çekilirse bilgi veri yastıktan CPU'ya transfer edilir.

S₃, S₄ (0) :

Bu iki sinyal hatlı A₁₆ ve A₁₇ adres hatlarıyla çoğaltılır. Bu iki bacak bir gerçek adresin segment (kesim) kısmını hangi segment yazmacının temin edeceğini belirler.

Bütün T (clock periyodu) durumları esnasında ilk T durumu hariç, gerçek fiziksel adresin segment kısmındaki bilgi bu iki bacağın çıkışında sunulur. Birinci T durumunda, bu iki bacak tipik adres hatları (A₁₆-A₁₇) gibi kullanılırlar. Bir I/O yol dönüşünde, bu bacaklar daima 1.saat periyodunda 0'dadır.

S₅ (0) :

Bu hat A₁₈ adres hatlıyla çoğaltılır. 1.saat periyodu esnasında bu bacak A₁₈ gibi çalışır. Diğer bütün saat periyodlarında bu bacak CPU'nun kesme isteği izin bayrağının (Interrupt enable Flæg) durumunu yansıtmak için kullanılır. Bir I/O yol dönüşünde, bu bacak 1.saat periyodunda daima 0'dadır.

S6 (0) :

Bu hat A₁₉ adres hattıyla çoğaltılır. İlk saat periyodu sırasında bu bacak A₁₉ gibi işlem görür. Bir I/O dönüş yolundaki birinci saat periyodunda bu bacak daima 0'dadır. Diğer bütün saat periyotları esnasında eğer sistem yolu CPU'nun kontrolü altında ise bu bacak 0'a çekilir. Bununla birlikte, eğer sistem yolu bir DMA kontrolü gibi diğer ana yolların kontrolü altında ise bu bacak sistem yolunun kazanç kontrolü için diğer ana yollarına izin vermek için CPU tarafından ayırd edilir.

INTR (I) :

Bu hat CPU'ya kesme istemini üretmek için I/O donanımı tarafından kullanılır. Bir kesme isteği üretildiği zaman bu bacak işlemi aldığını bildirine kadar 1'de tutulur.

NMI (I) :

Bu hat CPU'ya kesme istemini üretmek için I/O donanımı tarafından kullanılır. Bir kesme isteği üretildiğinde bu bacak 0'dan 1'e (kenar tetiklemeli) gider.

SOR (0) :

Bu kontrol sinyali (aktif 0) veri yolundaki bilgiyi sunmak için bir adreslenmiş I/O donanıma sebep olur. İşlemci tarafından sürülür.

MEMR (0) :

Bu kontrol sinyali (aktif 0) veri yolundaki veriyi göstermek için adreslenmiş bellek donanımına sebep olur. İşlemci tarafından sürülür.

MEMW (0) :

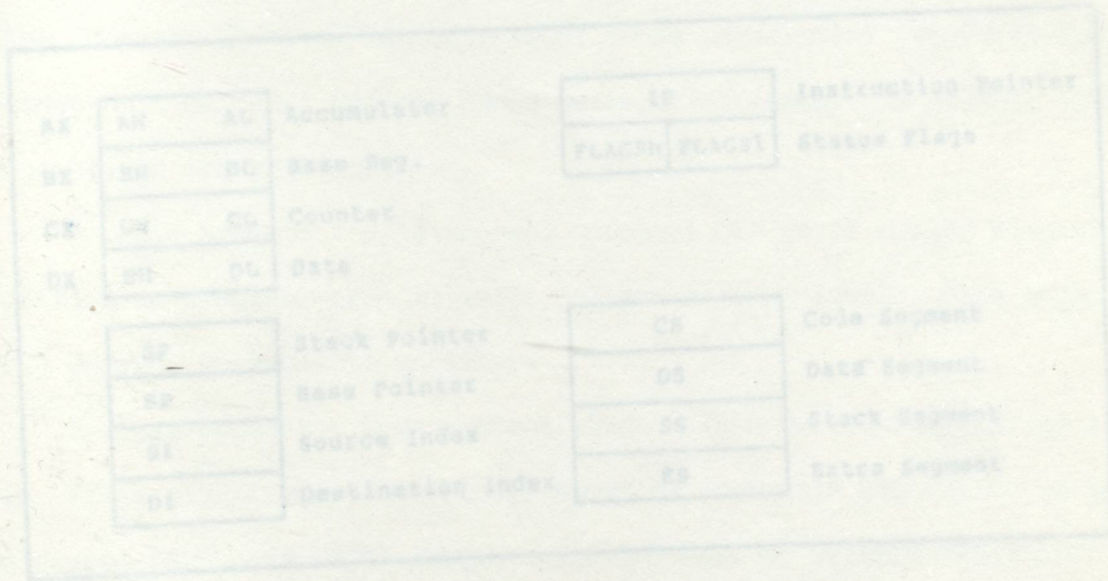
Bu kontrol sinyali (aktif 0), veri yolundaki sunulan bilgiyi alıp

8088'İN YAZMAÇLARI

getirmek için adreslenmiş bellek donanımına sebep olur. İşlemci tarafından sürülür.

CLK (0) :

Sistem saati. Osilatörün 3'le bölünmesinden türetilir. 210n S ve % 33 iş frekansı periyodunda bir saata sahiptir.



8088'İN YAZMAÇLARI

SEGMENTLER

Bir segment 64K Bayt uzunluğundan fazla bir bellek bölgesidir. Bellekte herhangi bir yere yerleştirilebilen segmentler belirli bir şekilde bölünebilir. Bir bölgedeki paragrafın sınırlarını belirler. Çoğu zaman program en azından üç segment içerir. Herhangi bir segmentin başlangıcı bir bölgeden yukarı olabileceğine rağmen segmentler işlem için gerekli olan miktardaki alan kadar adreslenirler.

Üç esas segment gruplarıdır.

1- Yığın (stack) segment: Basit terimlerde, yığın çağırma sistemi

AX	AH	AL	Accumulator	IP	Instruction Pointer
BX	BH	BL	Base Reg.	FLAGSh	FLAGSl
CX	CH	CL	Counter		Status Flags
DX	DH	DL	Data		
	SP		Stack Pointer	CS	Code Segment
	BP		Base Pointer	DS	Data Segment
	SI		Source Index	SS	Stack Segment
	DI		Destination Index	ES	Extra Segment

Aşağıdaki şekilde yazmaçların ve segmentlerin CS, DS ve SS yazmaçlarıyla

bir grafik gösterildiği bir örnek.

Segment Yazmaçları

ES	Extra Segment
DS	Data Segment
CS	Code Segment

Bellek

Operating System
Stack Segment
Data Segment
Code Segment

Segmentler ve Yazmaçlar

SEGMENTLER :

Bir segment 64K Bayt uzunluğundan fazla bir bellek bölgesidir. Bellekte herhangi bir yere yerleştirilebilen segmentler onaltıyla eşit olarak bölünebilir. Bir bölgedeki paragraph sınırında başlar. Çoğu zaman program en azından üç segment içerir. Herhangi bir segmentin 64Kbayt-lar ölçüsünden yukarı olabilmesine rağmen segmentler işlem için gerekli olan miktardaki alan kadar adreslenirler.

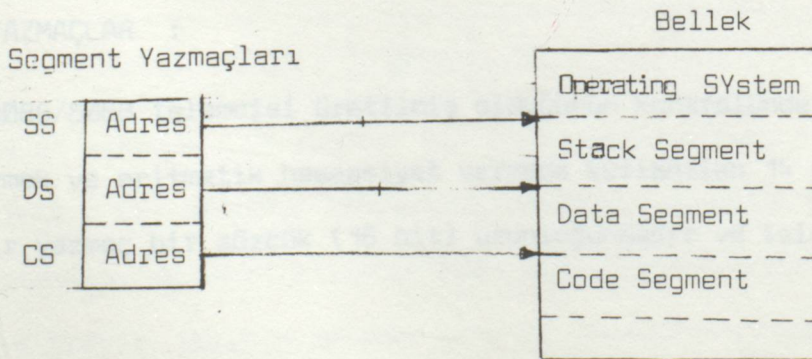
Üç esas segment şunlardır,

1- Yığın (stack) segment : Basit terimlerde, yığın çalışma sisteme (Dosyada CP/M) dönmek için kendi dönüş adreslerini çağırdığınız alt yordamların herhangi birinin dönüş adreslerini kapsar. SS yazmacı (Stack segment) bu segmente adresler.

2- Data Segment : Data segment herhangi bir tarif edilen bilgiyi, sabitlemiş ve programınızın istediği çalışma alanını içerir. Data segment (DS) yazmacı bu segmenti adresler.

3- Code Segment : Code segment yürütüm için makina komutlarını içerir. Tipik olarak yürütülebilir komut bu segmentin başındadır ve çalışma sistem (DOS veya CP/M) program yürütümü için bu bölgeye bağlanır.

Aşağıdaki şekilde yazmaçların ve segmentlerin CS,DS ve SS yazmaçlarının bir grafik görüntüsünü temin eder.



Segmentler ve Yazmaçlar.

Üç segment yazmaç her bir segmentin başlama adresini içerir ve her bir segment bir paragraph sınırında başlar.

Bir diğer segment yazmaç özel amaçlarla kullanılan Extra Segment ES yazmaçtır.

Bir program içinde, bütün bellek bölgeleri bir segmentin başlanıcısına bağlıdır. Bütün böyle bölgeler bir segment başlangıcından bir ofset (koyma) gibi gösterilir. (Bir koyma (ofset) keza bir yer değiştirme gibi bilinir.)

Bir iki bayt (16bit) ofset 0000H'dan FFFFH'e kadar sıralanabilir. Yada 0'dan 65535'e kadar. Sonuç olarak, bir programdaki herhangi bir bellek adresi segment yazmacındaki adres ilavesiyle bir ofset adres içindeki adres tarafından belirtilir.

Bir adresleme örneği olarak Data Segment yazmacının 045FH'i içerdiği ve bir 0032H ofsetli Data Segment içinde bir bölgede bir komutun belirtildiğini kabul edelim. Belirtilen gerçek bellek bölgesi bu yüzden aşağıdaki gibidir.

DS adresleri : 045F0H

Ofset : 0032H

Gerçek adres : 04622H

YAZMAÇLAR :

8086/8088 işlemcisi üretilmiş olduğunun kontrolünde bellek adresini göndermek ve aritmetik hassasiyet vermede kullanılan 14 yazmaca sahiptir. Her bir yazmaç bir sözcük (16 bit) uzunluğundadır ve isim ile adreslenebilir.

Dört Genel Amaçlı Yazmaç : AX, BX, CX ve DX

Çevirici (assembler) programlama için, genel amaçlı yazmaçlar en çok görev yapan yazmaçlardır. Onları bir sözcük yada bir bayt olarak tetikleyebildiğinizde tektirler. Örneğin AX yazmaç bir AH ve bir AL bölümle-
rinden oluşur ve üç yazmacın herhangi birini sıfırdan alabilirsiniz.

Aşağıdaki üç çevirici komut AX,AH ve AL yazmaçlarına sıralı olarak sıfırları yükler.

```
MOV     AX,00
MOV     AH,00
MOV     AL,00
```

1- AX yazmacı : 1.birikeç olarak bilinir. Bütün giriş çıkış işlemlerinde bazı dizi işlemlerinde ve bazı matematik işlemlerde kullanılır. Örneğin çarpmak bölmek ve AX yazmacının kullandığı kabul edilen komutları transfer eder. Bazı komutlar eğer AX'i referans aldılarsa daha elverişli kodu üretirler.

AX :	AH	AL	
------	----	----	--

2- BX yazmacı : BX yazmacı adres genişletmede bir gösterge gibi kullanılabilen yalnızca genel amaçlı yazmaç olduğundan taban yazmacı (Base register) olarak bilinir. Diğer kullanım amacı hesaplamalar içindir.

BX :	BH	BL	
------	----	----	--

3- CX yazmacı : CX yazmacı bir sayma yazmacı (count register) olarak bilinir. Bir döngünün tekrarlama zamanının sayısını kontrol etmek için istenilir ve hangi bitlerin sağa yada sola kaydırılan bit sayısı değerini içerir. Bununla beraber yine CX hesaplama içinde kullanılır.

CX :	CH	CL	
------	----	----	--

4- DX yazmacı : DX yazmacı herhalde "data" kelimenin "d" ile başlaması yüzünden "Data register" veri yazmacı olarak bilinir. Bazı giriş/çıkış işlemleri bu yazmaç kullanımı ister. Büyük değerleri içeren çarpma ve bölme işlemleri DX ve AX çiftini kabul eder.

DX :	DH	DL	
------	----	----	--

Genel amaçlı yazmaçların herhangi biri 8 bitlik yada 16 bitlik değerlerin ikisinden birinde toplama ve çıkarma işlemleri için kullanılabilir.

İKİ GÖSTERİCİ YAZMAÇ : SP ve BP

İki gösterici yazmaç, SP ve BP, yığın segmentindeki (SS) veri erişimi için sisteme izin verir. Öte yandan yine nadiren toplama ve çıkarma için kullanılır.

1- SP yazmacı : Yığın göstericisi bellekteki bir yığının tamamlanmasına izin verir. Bu yazmaç yığını adresleme için SS yazmacıyla birleştirilir.

2- BP yazmacı : Taban gösterici parametre referanslarını kolaylaştırır. (Bilgi ve adresler yığın yoluyla geçti)

İKİ GÖSTERGE YAZMAÇ : SI ve DI

Her iki gösterge yazmacının genişletilmiş adresleme, toplama ve çıkarma kullanımı mevcuttur.

1- SI yazmacı : Bu yazmaç kaynak gösterge olarak bilinir ve bazı dizi işlemleri için istenilir. İçinde bulunan şartlarda SI,DS yazmacıyla birleştirilir.

2- DI yazmacı : Bu yazmaç verinin gideceği yazmaç (destination register) olarak bilinir ve keza yine bazı dizi işlemler için istenilir. İçinde bulunan şartlarda, DI ES yazmaç 'la birleştirilebilir.

DÖRT SEGMENT (KESİM) YAZMAÇ : CS,DS,SS ve ES

Segment yazmaçlar bellek adreslemesinde önemlidir. Herbiri 64 Kbayt-
lık bir bellek alanının adreslemesini temin eder ve "yön segmenli (Current
Segmenti) olarak bilinirler. Başlarda tartışıldığı gibi, bir segment bir
pragraph sınırı üzerinde sıralanır ve bir segment yazmaç içindeki en
sağdaki dört biti ihmal eder.

1- CS yazmaç : Düşüm segment yazmaç (Code Segment registir) düşüm
segmentin başlangıç adresini içerir. Bu adres ilavesiyle ofset değeri
yürütümü başlatabilmek için bir komutun adresini gösterir. Normal prog-
ramlama kullanımları için CS referansına ihtiyaç yoktur.

2- DS yazmaç : Veri segment yazmaç veri segmentin başlangıç
adresini içerir. Basit terimlerde, bu adres veri segment içinde açıkça
belirtilen bir bölge için bir referansa sebep veren bir komutta ofset
ilavesiyle adresi gösterir.

3- SS yazmaç : Yığın segment yazmaç yığın segmentinin başlangıç
adresini içerir.

4- ES yazmaç : Bazı dizi işlemler bellek adreslemesini kullanmak
için Extra Segment yazmacı kullanılır. İçinde bulunan duruma göre ES
yazmaç DI yazmacıyla birleştirilir. Eğer ES isteniyorsa bir çevirici
(assembler) program onu başlatmalıdır.

Bir Komut Gösterici Yazmaç : IP

IP yazmaç yürütüm için komutun ofset adresini içerir. Normal olarak
bir programda bu yazmaç referans almayacaktır. IP yazmaç ayrıca yine
Program sayıcı (PC , Program Counter) olarakta bilinir.

Bir Bayrak (flag) Yazmaçı :

Bayrak yazmaçı hemde yine durum yazmaçı (status register) yada program durum sözcüğü olarakta bilinir. 16 bit'inin 9'u aktiftir. Makina akışı durumunu ve yürütüm sonuçlarını gösterir. Karşılaştırma ve aritmetik, olarak pek çok komut, bayraklarının durumunu değiştirir. Kısacası bayrak bitleri aşağıdadır.

<u>Bayrak</u>	<u>Amaç</u> :
O (over flow)	Sonraki işleme bir diğer en büyük bit koşmasını gösterir.
D (Direction (yön))	Veri dizi (bir sözcüğü bastırılmış bellekteki veri) karşılaştırılması yada hareket etmesi için sol sağ yönünü belirler.
I (Interrupt)	Bir kesmenin olup olmadığını gösterir.
T (İrap)	CPU'yu adım adım çalıştırılmasına izin verir. Yazmaçlardaki ve bellekteki tesiri kontrol etmek için bir anda yürütülmüş bir komut içinden izleyebilmeniz için Trap bayrağını 1'e çeker.
S (Sign)	Aritmetik işlemlerin işaret sonuçlarını içerir. (0 = + ve 1 = -)
Z (Zero)	Aritmetik yada karşılaştırma işlemlerin sonuçlarını gösterir. (0 = sonuç sıfır değil ve 1= sonuç sıfır)

LOOP komutunun iki değişkeni vardır. LOOPE ve LOOPNE. Her iki işlem CX'i birer azaltır. LOOPE (yada LOOPZ) eğer CX sıfır değilse ve sıfır durumu olmuşsa (ZF= 1) operant adresine transfer eder. LOOPNE (yada LOOPNZ) eğer CX sıfır değilse ve sıfır değil durumu olmuşsa (ZF = 0) transfer eder.

BAYRAKLAR YAZMACI : (Flag register)

Bu bölümde geri kalan konunun çoğu veri detaylı bir bayrak yazmacı bölümü ister. Bayrak yazmaçlar bir işlemin durumunu göstermek için çeşitli komutların kurduğu 16 bit'i içerir. Bazı sistemlerde, bayrak yazmaç, durum yazmaç yada Program durum sözcüğü olarak bilinir. Bayrak yazmaçlar aşağıdaki 9 kullanılan biti içerir. (* İşareti kullanılmayan bir biti gösterir.)

Bit no	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bayrak	:	*	*	*	*	0	D	I	T	S	Z	*	A	*	P	*	C

CF (Carry Flag) : Aritmetik işlemleri, bazı kaydırma ve döndürme işlemlerini takip eden en yüksek değerli bittten elde edilen (0 yada 1) eldelerini içerir.

PF (Parity Flag) : Veri işlemlerinin en küçük değerlikli 8.bitinin bir kontrolüdür. Veri bitlerinin bir tek numarası bayrağı 0'a getirir ve çift numarası bayrağı 1'e getirir.

AF (Yardımcı Carry flag) : Eğer aritmetik işlem bir yazmaç bir bayt işleminin 3.bitinden (sağdan 4.) bir eldeye sebep oluyorsa 1'e gelir. Bu bayrak desimal düzende paketlenmiş ve ASCII kodundaki aritmetik işlemlerle alakalıdır.

ZF (Sıfır bayrağı) : Aritmetik yada karşılaştırılma işlemlerin bir sonucu olarak elde edilir. Umulmadık bir şekilde sıfır olmayan bir sonuç bu bayrağı 0'a, 0 olan bir sonuç ise bu bayrağı 1'e getirir. Bununla birlikte, görünüşte bu biçimde bayrağın değer alması doğru olmasa bile sayısal olarak doğrudur: 0, sonuç 0'a eşit değil anlamında ve 1'de sonuç 0'a eşit anlamındadır.

JE ve JZ bu bayrağı test eder.

SF (İşaret bayrağı) : Bir aritmetik işlemden sonra (en yüksek değerli bit yada en soldaki biti) işarete göre değer alır. Pozitiflerde 0'a, negatiflerde 1'e gelir. JG ve JL komutları bu bayrağı test eder.

TF (Trap Bayrağı) : Eğer bu bayrak 1'e çekilirse işlemcinin adım adım çalışmasına neden olur. (Şöyleki, kullanıcı kontrolü altındaki bir anda komut)

IF (İnterrupt Bayrağı) : 0 olduğu zaman bütün kesme istekleri engellenir, 1 olduğu zaman bütün kesme isteklerine izin verilir.

DF (Direction Flag) : Veri transferinin yönünü temin etmede dizi işlemleri tarafından kullanılır. 0 olduğunda işlem, veri transferinin sağdan sola olmasına sebep olur. SI ve DI yazmaçlarını azaltır.

OF (Overflow Bayrağı) : İşaretlenmiş bir aritmetik işlemi takip eder. En yüksek değerli biti (en soldaki bit) işaret bitinin içinde ve dışında bir eldeyi gösterir.

Bir örnek olarak CMP komutu iki operantı karşılaştırır ve AF, CF, OF, PF, SF ve ZF bayraklarına etki eder. Bununla beraber, bu bayrakları başlı başına test etme zorunluğu yoktur.

Aşağıda eğer BX yazmaçları bir 0 değeri içerip içermediği test edilir.

CMP BX, 00H ; BX'i 0'la karşılaştır.

JZ B50H ; eğer sıfırsa B50'yı atla.

Eğer BX sıfır içeriyorsa CMP ZF'ı 1'e çeker ve diğer bayrakları değiştirebilir yada değiştirmez. JZ (eğer sıfırsa atla) komutu yalnızca ZF bayrağını test eder. ZF'nin 1'i içermesinden (bir sıfır durumu anlamında), JZ B50H operantı tarafından gösterilen adrese kontrolü transfer eder.

8088'İN ADRESLEME TURLERİ

TABLE 12.7 ADDRESSING MODES OF THE 8086 MICROPROCESSOR

Mode	Physical memory location	Examples (using MOV)	
		Mnemonic	Symbolic
Immediate	Within instruction + code segment	MOV AX,1000H	10 → AH 00 → AL
Register	In register + code segment	MOV BX,DX	DL → BL DH → BH
Direct	Address + segment	MOV 8000H,BX	BL → (8000) BH → (8001)
Register indirect	SI + segment DI + segment BX + segment BP + segment	MOV CX,(BX)	(BX) → CL
Indexed	SI + displacement + segment DI + displacement + segment	MOV AL,(SI)+6	(SI+6) → AL
Relative	Displacement + IP + code segment	JMP target	Displacement + IP → IP
Based	BX + displacement + segment BP + displacement + segment	MOV (BX)+6,AL	AL → (BX+6)
Based and indexed	BX + SI + segment BX + DI + segment BP + SI + segment BP + DI + segment	MOV DX,(BX+DI)	(BX+DI) → DL (BX+DI+1) → DH
Based and indexed with displacement	BX + SI + displ. + segment BX + DI + displ. + segment BP + SI + displ. + segment BP + DI + displ. + segment	MOV (BX+DI)+5,DX	DL → (BX+DI+5) DH → (BX+DI+6)

Notes:

1. All operands of the form destination, source
2. Parentheses indicate the contents of memory addressed by the operand
3. In all cases the operand specifies an offset that must be added to the proper segment register contents to form the physical address
4. Table 12-6 indicates the default segment register to be used depending on the type of memory reference

8088'İN KOMUT TABLOSARI

Instruction	Object Code	Bytes	Clock Periods
AAA	37	1	4
AAD	D5 0A	2	60
AAM	D4 0A	2	83
AAS	3F	1	4
ADC	ac,data 0001010w kk [jj]	2 or 3	4
ADC	mem/reg ₁ ,data 100000sw mod 010 r/m [DISP] [DISP] kk [jj]	3, 4, 5 or 6	reg: 4 mem: 17 + EA
ADC	mem/reg ₁ ,mem/reg ₂ 000100dw mod rrr r/m [DISP] [DISP]	2, 3 or 4 2, 3 or 4	reg to reg: 3 mem to reg: 9 + EA reg to mem: 16 + EA
ADD	ac,data 0000010w kk [jj]	2 or 4	4
ADD	mem/reg,data 100000sw mod 000 r/m [DISP] [DISP] kk [jj]	3, 4, 5 or 6	reg: 4 mem: 17 + EA
ADD	mem/reg ₁ ,mem/reg ₂ 000000dw mod rrr r/m [DISP] [DISP]	2, 3 or 4	reg to reg: 3 mem to reg: 9 + EA reg. to mem: 16 + EA
AND	ac,data 0010010w kk [jj]	2 or 4	4
AND	mem/reg,data 1000000w mod 100 r/m [DISP] [DISP] kk [jj]	3, 4, 5 or 6	reg: 4 mem: 17 + EA
AND	mem/reg ₁ ,mem/reg ₂ 001000dw mod rrr r/m [DISP] [DISP]	2, 3 or 4	reg to reg: 3 mem to reg: 9 + EA reg to mem: 16 + EA
CALL	addr 9A kk jj hh 99 E8 kk jj FF	5	28
CALL	disp 16 E8 kk jj	3	19
CALL	mem FF mod 011 r/m [DISP] [DISP]	2, 3 or 4	32-bit mem pointer: 37 + EA
CALL	mem/reg FF mod 010 r/m [DISP] [DISP]	2, 3 or 4	16-bit reg pointer: 16 16 bit mem pointer: 21 + EA

Instruction	Object Code	Byte	Clock Periods
CBW	98	1	2
CLC	F8	1	2
CLD	FC	1	2
CLI	FA	1	2
CMC	F5	1	2
CMP	ac,data	0011110w kk [ij]	2 or 3 4
CMP	mem/reg,data	100000sw mod 111 r/m [DISP] [DISP] kk [ij]	3, or 4 or 6 reg: 4 mem: 10 + EA
CMP	mem/reg ₁ ,mem/reg ₂	001110dw mod rrr r/m [DISP] [DISP]	2, 3 or 4 reg to reg: 3 mem to reg: 9 + EA reg to mem: 9 + EA
CMPS		1010011w	1 22 9 + 22/repetition*
CWD		99	1 5
DAA		27	1 4
DAS		2F	1 4
DEC	mem/reg	1111111w mod 001 r/m [DISP] [DISP]	2, 3 or 4 reg: 3 mem: 15 + EA
DEC	16-bit reg	01001 rrr	1 2
DIV	mem/reg	1111011w mod 110 r/m [DISP] [DISP]	2, 3 or 4 8-bit reg: 80 → 90 16-bit reg: 144 → 162 8-bit mem: (86 → 96) + EA 16-bit mem: (150 → 168) + EA mem: 8 + EA reg: 2
ESC	mem/reg	11011xxx mod xxx r/m [DISP] [DISP]	2, 3 or 4 2
HLT		F4	1 2
IDIV	mem/reg	1111011w mod 111 r/m [DISP] [DISP]	2, 3 or 4 8-bit reg: 101 → 112 16-bit reg: 165 → 184 8-bit mem: (107 → 118) + EA 16-bit mem: (171 → 190) + EA
IMUL	mem/reg	1111011w mod 101 r/m [DISP] [DISP]	2, 3 or 4 8-bit reg: 80 → 98 16-bit reg: 128 → 154 8-bit mem: (86 → 104) + EA 16-bit mem: (134 → 160) + EA
IN	ac, DX	1110110w	1 8
IN	ac, port	1110010w	2 10

Instruction		Object Code	Bytes	Clock Periods
INC	mem/reg	1111111w mod 000 r/m [DISP] [DISP]	2, 3 or 4	reg: 3 mem: 15 + EA
INC	16-bit reg	01000rrr	1	2
INT		11001100*	1	52
		11001101	2	51
INTO		type CE	1	interrupt: 53 no interrupt: 4
IRET		CF	1	24
JA	disp	77	2	4/No Branch
JNBE		disp		16/Branch
JAE	disp	73	2	4/No Branch
JNB		disp		16/Branch
JB	disp	72	2	4/No Branch
JNAE		disp		8/Branch
JBE	disp	76	2	4/No Branch
JNA		disp		16/Branch
JCXZ	disp	E3	2	6/No Branch
		disp		18/Branch
JE	disp	74	2	4/No Branch
JZ		disp		16/Branch
JG	disp	7F	2	4/No Branch
JNLE		disp		16/Branch
JGE	disp	7D	2	4/No Branch
JNL		disp		16/Branch
JL	disp	7C	2	4/No Branch
JNGE		disp		16/Branch
JLE	disp	7E	2	4/No Branch
JNG		disp		16/Branch
JMP	addr	EA kk jj hh 99	5	15
JMP	disp	EB	2	15
JMP	disp 16	disp E9 kk jj	3	15
JMP	mem	FF mod 101 r/m [DISP] [DISP]	2, 3 or 4	mem ptr 32: 24 + EA
JMP	mem/reg	FF mod 100 rr/m [DISP] [DISP]	2, 3 or 4	reg ptr 16: 11 mem ptr 16: 18 + EA
JNE	disp	75	2	4/No Branch
JNZ		disp		16/Branch
JNO	disp	71	2	4/No Branch
		disp		16/Branch
JNP	disp	78	2	4/No Branch
JPO		disp		16/Branch
JNS	disp	79	2	4/No Branch
		disp		16/Branch
JO	disp	70	2	4/No Branch
		disp		16/Branch

* Implied type = 3

Instruction		Object Code	Bytes	Clock Periods
JP	disp	7A	2	4/No Branch
JPE	disp	disp		16/Branch
JS	disp	78	2	4/No Branch
LAHF		disp		16/Branch
LDS	reg,mem	9F	1	4
		C5	2, 3 or 4	16 + EA
		mod rrr r/m		
		[DISP]		
		[DISP]		
LEA	reg,mem	8D	2, 3 or 4	2 + EA
		mod rrr r/m		
		[DISP]		
		[DISP]		
LES	reg,mem	C4	2, 3 or 4	16 + EA
		mod rrr r/m		
		[DISP]		
		[DISP]		
LOCK		F0	1	2
LODS		1010110w	1	12
LOOP	disp	E2	2	9 + 13/repetition*
		disp		5/No Branch
LOOPE	disp	E1	2	17/Branch
LOOPZ	disp	disp		6/No Branch
LOOPNE	disp	E0	2	18/Branch
LOOPNZ	disp	disp		5/No Branch
MOV	mem/reg ₁ , mem/reg ₂	100010dw	2, 3 or 4	19/Branch
		mod rrr r/m		reg to reg: 2
		[DISP]		reg to mem: 8 + EA
		[DISP]		mem to reg: 9 + EA
MÖV	reg,data	1011 wrrr	2 or 3	4
		kk		
		[jj]		
MOV	ac,mem	1010000w	3	10
		kk		
		jj		
MOV	mem,ac	1010001w	3	10
		kk		
		jj		
MOV	segreg,mem/reg	8E	2, 3 or 4	reg to reg: 2
		mod Orr r/m		mem to reg: 8 + EA
		[DISP]		
		[DISP]		
MOV	mem/reg,segreg	8C	2, 3 or 4	reg to reg: 2
		mod Orr r/m		reg to mem: 9 + EA
		[DISP]		
		[DISP]		
MOV	mem/reg,data	1100011w	3, 4, 5 or 6	reg/mem: 10 + EA
		mod 000 r/m		
		[DISP]		
		[DISP]		
		kk		
		[jj]		
MOVS		1010010w	1	18
				9 + 17/repetition*

* When preceded by REP prefix

Instruction		Object Code	Bytes	Clock Periods
MUL	mem/reg	1111011w mod 100 r/m [DISP] [DISP]	2, 3 or 4	8-bit reg: 70 → 77 16-bit reg: 118 → 133 8-bit mem: (76 → 83) + EA 16-bit mem: (124 → 139) + EA
NEG	mem/reg	1111011w mod 011 r/m [DISP] [DISP]	2, 3 or 4	reg: 3 mem: 16 + EA
NOP		90	1	3
NOT	mem/reg	1111011w mod 010 r/m [DISP] [DISP]	2, 3 or 4	reg: 3 mem: 16 + EA
OR	ac,data	0000110w kk [ij]	2 or 3	4
OR	mem/reg,data	1000000w mod 001 r/m [DISP] [DISP] kk [ij]	3, 4, 5 or 6	reg: 4 mem: 17 + EA
OR	mem/reg ₁ , mem/reg ₂	000010dw mod rrr r/m [DISP] [DISP] kk [ij]	3, 4, 5 or 6	reg to reg: 3 mem to reg: 9 + EA reg to mem: 16 + EA
OUT	DX,ac	1110111w	1	8
OUT	port,ac	1110011w YY	2	10
POP	mem/reg	8F mod 000 r/m [DISP] [DISP]	2, 3 or 4	reg: 8 mem: 17 + EA
POP	reg	01011rrr	1	8
POP	segreg	000ss111	1	8
POPF		9D	1	8
PUSH	mem/reg	FF mod 110 r/m [DISP] [DISP]	2, 3 or 4	reg: 11 mem: 16 + EA
PUSH	reg	01010rrr	1	10
PUSH	segreg	000ss110	1	10
PUSHF		9C	1	10
RCL	mem/reg,count	110100cw mod 010 r/m [DISP] [DISP]	2, 3 or 4	count = 1 reg: 2 mem: 15 + EA count = [CL] reg: 8 + (4 * N) mem: 20 + EA + (4 * N)

N = count value in CL

Instruction		Object Code	Bytes	Clock Periods
RCR	mem/reg,count	110100cw mod 011 r/m [DISP] [DISP]	2, 3 or 4	count = 1 reg: 2 mem: 15 + EA count = [CL] reg: 8 + (4 * N) mem: 20 + EA + (4 * N)
REP	/REPE/REPNE	1111001z	1	2
RET	(Inter-segment)	CB	1	18
RET	(Intra-segment)	C3	1	8
RET	disp 16 (Inter-segment)	CA kk jj	3	17
RET	disp 16 (Intra-segment)	C2 kk jj	3	12
ROL	mem/reg,count	110100cw mod 000 r/m [DISP] [DISP]	2, 3 or 4	count = 1 reg: 2 mem: 15 + EA count = [CL] reg: 8 + (4 * N) mem: 20 + EA + (4 * N)
ROR	mem/reg,count	110100cw mod 001 r/m [DISP] [DISP]	2, 3 or 4	count = 1 reg: 2 mem: 15 + EA count = [CL] reg: 8 + (4 * N) mem: 20 + EA + (4 * N)
SAHF		9E	1	4
SAR	mem/reg,count	110100cw mod 111 r/m [DISP] [DISP]	2, 3 or 4	count = 1 reg: 2 mem: 15 + EA count = [CL] reg: 8 + (4 * N) mem: 20 + EA + (4 * N)
SBB	ac,data	0001110w kk [jj]	2 or 3	4
SBB	mem/reg,data	100000sw mod 011 r/m [DISP] [DISP] kk [jj]	3, 4, 5 or 6	reg: 4 mem: 17 + EA
SBB	mem/reg ₁ ,mem/reg ₂	000110dw mod rrr r/m [DISP] [DISP]	2, 3 or 4	reg from reg: 3 mem from reg: 9 + EA reg from mem: 16 + EA
SCAS		1010111w	1	15 9 + 15/repetition*
SEG	segreg	001ss110	1	2
SHL	mem/reg,count	110100cw mod 100 r/m [DISP] [DISP]	2, 3 or 4	count = 1 reg: 2 mem: 15 + EA count = [CL] reg: 8 + (4 * N) mem: 20 + EA + (4 * N)
SAL				

* When preceded by REP prefix
N = count value in CL

Instruction		Object Code	Bytes	Clock Periods
SHR	mem/reg,count	110100cw mod 101 r/m [DISP] [DISP]	2, 3 or 4	count = 1 reg: 2 mem: 15 + EA count = [CL] reg: 8 + (4 * N) mem: 20 + EA + (4 * N)
STC		F9	1	2
STD		FD	1	2
STI		FB	1	2
STOS		1010101w	1	11
SUB	ac,data	0010110w kk [jj]	2 or 3	9 + 10/repetition* 4
SUB	mem/reg,data	100000sw mod 101 r/m [DISP] [DISP] kk [jj]	3, 4, 5 or 6	reg: 4 mem: 17 + EA
SUB	mem/reg ₁ ,mem/reg ₂	001010dw mod rrr r/m [DISP] [DISP] [DISP]	2, 3 or 4	reg from reg: 3 mem from reg: 9 + EA reg from mem: 16 + EA
TEST	ac,data	1010100w kk [jj]	2 or 3	4
TEST	mem/reg,data	1111011w mod 000 r/m [DISP] [DISP] kk [jj]	3, 4, 5 or 6	reg: 5 mem: 11 + EA
TEST	reg,mem/reg	1000010w mod rrr r/m [DISP] [DISP] [DISP]	2, 3 or 4	reg with reg: 3 reg with mem: 9 + EA
WAIT		9B	1	3(min.) + 5n
XCHG	reg,ac	10010rrr	1	3
XCHG	reg,mem/reg	100011w mod rrr r/m [DISP] [DISP] [DISP]	2, 3 or 4	reg with reg: 4 reg with mem: 17 + EA
XLAT		D7	1	11
XOR	ac,data	0011010w kk [jj]	2 or 3	4
XOR	mem/reg,data	100000w mod 110 r/m [DISP] [DISP] kk [jj]	3, 4, 5 or 6	reg: 4 mem: 17 + EA
XOR	mem/reg ₁ ,mem/reg ₂	001100dw mod rrr r/m [DISP] [DISP] [DISP]	2, 3 or 4	reg with reg: 3 mem with reg: 9 + EA reg with mem: 16 + EA

* When preceded by REP prefix
n = clocks per sample of the TEST input

ÖZGEÇMİŞ

1963 yılında Malatya'da doğdum. İlk ve ortaokulu Malatyada bitirdim. Daha sonra 1979 yılında Ş.K.Ö. Endüstri Meslek Lisesinin Elektronik bölümüne girdim. 1981 yılında buradan mezun oldum. O yıl Yıldız Üniversitesi Elektronik ve Haberleşme Mühendisliği bölümünü kazandım. 1985 yılında buradan Elektronik ve Haberleşme Mühendisliği diploması aldım. 1985 yılında aynı Üniversitenin Yüksek lisans bölümüne başvurdum. Halen Yıldız Üniversitesi Elektronik ve Haberleşme Mühendisliğinde Yüksek lisans yapmaktayım.