

YILDIZ TEKNİK ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

Op/m İşletim Sisteminin İncelenmesi

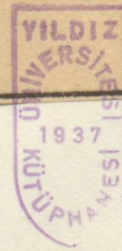
Hulusi K    r

Y  ksek Lisans Tezi

Ref
EHM
226
1985

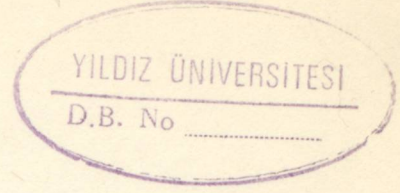
YILDIZ ÜNİVERSİTESİ
GENEL KİTAPLIĞI

Kot : R 152
Alındığı Yer : Fen Bil. Ens. 38
Tarih : 13.4.987
Fatura :
Fiatı : 2000 TL
Ayniyat No : 1/1
Kayıt No : 44774
UDC :
Ek :



Comp

YILDIZ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ



CP/M İŞLETİM SİSTEMİNİN İNCELENMESİ

YÜKSEK LİSANS TEZİ
MÜH. HULUSİ KÜLTÜR

İSTANBUL 1985

Mühendislik eğitimim ve tez çalışmalarım süresince değerli destek ve katkılarını esirgemeyen hocam ve yürütücüm Sayın Doç.Dr. Sezgin Alsan'a teşekkürü bir borç bilirim.

İ Ç İ N D E K İ L E R

ÖZET	V
SUMMARY	VI
GİRİŞ	1

BİRİNCİ BÖLÜM

CP/M İŞLETİM SİSTEMİNİN ÖZELLİKLERİ VE OLANAKLARININ TANITIMI

1. GİRİŞ	4
2. CP/M'İN İŞLEVLERİ	5
2.1. GENEL KOMUT YAPISI	6
2.2. KÜTÜK REFERANSLARI (FILE REFERENCES)	6
3. DİSKLERİN DEĞİŞTİRİLMESİ	8
4. KALICI (BUILT-IN) KOMUTLARI YAPISI.	9
4.1. ERA	9
4.2. DIR	9
4.3. REN	10
4.4. SAVE	11
4.5. TYPE	11
5. SATIR DÜZELTİMİ VE ÇIKIŞ KONTROLLARI.	12
6. GEÇİCİ KOMUTLAR	12
6.1. STAT	13
6.2. ASM	20
6.3. LOAD	21
6.4. PIP	22
6.5. ED	29
6.6. SYSGEN	30
6.7. SUBMIT	31
6.8. XSUB	33
6.9. DUMP	34
6.10. MOVCPM	34

İKİNCİ BÖLÜM

CP/M BİÇİMLEYİCİ "EDITOR"

1. GİRİŞ	46
1.1. ED İşlemi	46
1.2. Metin Aktarma İşlevleri	48
1.3. Tampon Belleğin Organizasyonu	49
1.4. Satır Numaraları ve ED Programının Başlatılması	50
1.5. Tampon Bellek İşlemi	51
1.6. Yazısal Komutlar (Command Strings)	53
1.7. Metin İçindeki Yazısalların Araştırılması ve Değiştirilmesi (Text Search and Alteration)	54
1.8. Kaynak Kütüphane Kütükleri (Source Libraries)	57
1.9. Komutun Ardışıl Olarak Yürütülmesi (Repetitive Command Execution)	58
2. ED PROGRAMI HATA KOŞULLARI (ED ERROR CONDITIONS)	58
3. ED KOMUTLARININ ÖZETİ	60

ÜÇÜNCÜ BÖLÜM

CP/M ÇEVİRİCİ "ASSEMBLER"

1. GİRİŞ	62
2. PROGRAMIN GÖRÜNÜMÜ	64
3. "OPERAND" ALANININ BİÇİMLENDİRİLMESİ	65
3.1. Etiketler (Labels)	66
3.2. Nümerik Sabitler (Numeric Constants)	66
3.3. Ayrılmış Kelimeler (Reserved Words)	67
3.4. Yazısal Sabitler (String Constants)	67
3.5. Aritmetik ve Mantıksal İşlemler	68
3.6. Öncelikli İşlemler	69
4. ÇEVİRİCİ DİREKTİFLERİ (ASSEMBLER DIRECTIVES)	70
4.1. ORG Direktifi	71
4.2. END Direktifi	71
4.3. EQU Direktifi	72
4.4. SET Direktifi	73

4.5. IF ve ENDIF Direktifleri	73
4.6. DB Direktifi	74
4.7. DW Direktifi	75
4.8. DS Direktifi	75
5. İŞLETİM KODLARI	76
5.1. Dallarına, Çağırma ve Geri Dönme Komutları (Jumps, Calls, Returns)	76
5.2. Hemen İşlenen Komutlar (Immediate operand instructions)	77
5.3. Artırma ve Azaltma Komutları (Increment and Decrement Instructions)	78
5.4. Veri Aktarma Komutları (Data Movement Instructions)	79
5.5. Aritmetik ve Mantıksal İşlem Komutları	80
5.6. Kontrol Komutları (Control Instructions)	81
6. HATA MESAJLARI (ERROR MESSAGES)	81

DÖRDÜNCÜ BÖLÜM

CP/M DİNAMİK DÜZELTME ARACI

"DYNAMIC DEBUGGING TOOL"

1. GİRİŞ	84
2. DDT KOMUTLARI	86
2.1. A (Assembly) Komutu	86
2.2. D (Display) Komutu	87
2.3. F (Fill) Komutu	87
2.4. G (Go) Komutu	88
2.5. I (Input) Komutu	89
2.6. L (List) Komutu	90
2.7. M (Move) Komutu	90
2.8. R (Read) Komutu	91
2.9. S (Set) Komutu	92
2.10. T (Trace) Komutu	92
2.11. U (Untrace) Komutu	93
2.12. X (Examine) Komutu	93
3. YÜRÜTME NOTLARI (IMPLEMENTATION NOTES)	95

BEŞİNCİ BÖLÜM

CP/M İŞLETİM SİSTEMİ GİRİŞ VE ÇIKIŞ İŞLEVLERİ

1. GİRİŞ	97
2. İŞLETİM SİSTEMİ ÇAĞRI DÜZENLERİ	98

ALTINCI BÖLÜM

PROGRAM ÖRNEKLERİ

1. RASTGELE ERİŞİM PROGRAMI	121
2. "DUMP" PROGRAMI	125
3. "TIME" PROGRAMI	129
KAYNAKLAR	137
ÖZGEÇMİŞ	138

ÖZET

CP/M (Mikrobilgisayarlar için kontrol programı), Zilog Z-80, Intel 8080 ve 8085 temelli makinalar için yazılmış bir monitor programıdır. İlk 8080 mikroişlemcisi için daha sonra Z-80 ve 8085 temelli makinalar için pazarlanmıştır.

CP/M işletim sistemi MONROE EC8800 modeli bir bilgisayar üzerinde incelenmiştir. Bu bilgisayar A ve B olarak isimlendirilen iki disket sürücüsüne, üç tane seri haberleşme kapısına sahiptir. MOS denilen kendi işletim sisteminde veya CP/M işletim sisteminde çalıştırılabilmektedir. Kendi belleği, 128K Bayt olduğu halde CP/M 64K Bayt'dan daha fazla bellek kullanılmasına izin vermemektedir.

CP/M dört bölümden meydana gelmiştir. Bunlardan birincisi, yaklaşık 56K Bayt'lık bir bellek bölümünü kullanıcıya sunan geçici program alanı, diğerleri sırasıyla 2K Bayt'lık konsol komut işleyicisi, 3K Bayt'lık temel disk işletim sistemi ve donanıma bağımlı olan 0.5 ilâ 1K Bayt arasında değişen temel giriş çıkış sistemidir. Ayrıca sıfırıncı sayfaya 256 Bayt'lık, sistem parametreleri yerleştirilmiştir.

CP/M kalıcı ve geçici olmak üzere iki komut yapısına sahiptir. Kalıcı komutlar CP/M'in konsol komut işleyici programı içersinde bulunan ve temel işlevleri yerine getiren komutlardır. Geçici komutlar ise disket üzerinde bulunurlar ve konsoldan komut olarak girildiklerinde geçici program alanına yüklenerek burada yürütülürler.

Bu çalışmada CP/M işletim sisteminin kullanılması, kütük oluşturulması, geçici komutları ve giriş çıkış işlevleri incelenmiştir.

SUMMARY

CP/M (A control program for microcomputers) is a monitor program for Zilog Z-80, Intel 8080 and 8085 based machines. It has been marketed first for 8080 microprocessors and then for Z-80 and 8085 based machines.

The CP/M operating system has been analyzed on a MONROE EC 8800 computer. This computer has 2 floppy disk drives named A and B and 3 serial communication ports. It may work in its own operating system called MOS or CP/M. Although its own memory is 128K Bytes, CP/M does not allow the usage of more than 64 Bytes.

CP/M consists of four parts. The first of these is a transient program area presenting the user a memory of 56K Bytes, the second is a console command processor of 2K Bytes, the third is a basic disk operating system of 3K Bytes and the fourth is a basic input output system changing between 0.5-1K Byte depending on equipment.

CP/M has 2 command structures, one built-in, the other transient. Built-in commands are those that are found in the console command processor and those that fulfil basic functions. Transient commands are found on the disk and as they are entered as commands, they are loaded on the transient program areas.

This study analyzes the usage of the CP/M operating system, file creation, transient commands and basic functions.

GİRİŞ

CP/M (Control Program for Microcomputers: Mikrobilgisayarlar için kontrol programı) işletim sistemi; Zilog Z-80, Intel 8080 ve 8085 temelli makinalar için yazılmış, kullanıcı ile bilgisayar arasında arabirim olarak kullanılan bir monitor programıdır. Performansın artması için makina dilinde yazılmıştır. Makina ve kullanıcı arasındaki bütünlüğü sağlar. İşletim sistemi aracılığıyla olur. Kullanıcının çevre birimlerini ve disket sürücüsünü yönetmesi çok zor olduğundan dolayı işletim sisteminin bu işlevleri yerine getirmesi gerekir. Kısaca işletim sisteminin amacı kullanıcının işini minimuma indirmektir. Bunu da kullanıcının bilmesi gerekmeyen, birtakım işleri üzerine alarak yapar.

CP/M sisteminin üç önemli işlevi şunlardır:

- . Tuş takımını çabuk okur,
- . Okuduğunu saklar,
- . Çevre birimleri ile iletişimi sağlar.

İşletim sisteminin giriş ve çıkış işlevleri kullanıcıya büyük kolaylık sağlar. Geçici programların birçoğu bu işlevlerin kullanılmasıyla oluşturulmuştur. Burada önemli olan bu işlevleri gerektiğinde ve yerinde kullanabilmektir. Bu sanki kitapçıya gittiğiniz zaman çok iyi bilinen bir eseri istemeniz gibidir. Sizin onu araştırmanız gerekseydi, bir zaman kaybı olacaktı. Fakat kitapçı bunu size hemen sağlayabilir.

CP/M işletim sistemi, yarı iletken firması Intel'de danışman olarak çalışan Garry Kildall tarafından oluşturulmuştur. 1970 yıllarında Kildall bilgisayar dersleri vermekteydi, aynı zamanda dünyanın ilk gerçek mikrobilgisayarlarından biri olan, Intel'in 8088-bazlı "Intellec" gelişim sis-

temine ilk disket sürücülerinden birini eklemeye çalışıyordu. Daha sonraları Digital Micro System'i kuran arkadaşı John Torode'un da yardımıyla ev-yapımı bir disk kontrolörünü üretti ve basit bir kontrol programı yazdı. Bu CP/M in ilk uyarlamasıydı. Temel olarak PL/M ile yazılmıştı, kağıt bant aracılığıyla yüklenmekteydi ve işe yarıyordu.

Kildall'un CP/M in üzerinde yeniden düşünmeye başlaması, bilgisayar üreticilerinin kendisine lisans hakları için müracaat etmeye başlamasına dek gerçekleşmedi. Parçaları yeniden kodlamaya karar vererek bütün donanım bağımlı özellikleri "Basic Input/Output System" adını verdiği tek bir modülde topladı. Bu CP/M in ortaya çıkışıydı. Artık yalnızca sistemin kendisini yeni bilgisayarlara taşımak, kolaylaşmakla kalmıyor, uygulama programlarının da değişik bilgisayarlar üzerinde çalıştırılabilmesi mümkün kılınıyordu. Bilgisayar üreticilerinin bunu satın almalarının nedeni satışa hazır durumda bulunmasıydı.

CP/M ve yazılımının kolay bulunabilir olması Intel 8080 ve Zilog Z-80 mikroişlemcilerinin yaygın olarak kullanımını sağladı. Z-80 in üstün hızı (4 MHz'e kadar çalışabilmektedir) geniş bilgi kümesiyle birleşince yalnızca 2 MHz'e kadar çıkabilen 8080'den daha çok kullanılır oldu. 8080'in geliştirilmiş bir uyarlaması olan Intel 8085, 5 MHz'den biraz daha hızlı çalışabilmektedir. CP/M'e yazılımından bugüne kadar birçok ekler yapılmıştır. Bu ekler çeşitli uyarlama numaraları ile anılmaktadır. Bu çalışmada CP/M'in 8080 uyarlaması, yani CP/M 2.2 yada CP/M-80 kastedilmektedir.

1. Giriş

CP/M işletim programı geniş bir yönetim paketi. Çoğu işletim programlarına benzer bir şekilde ulayım alanları ve programlar CP/M mantıklı olarak dört ayrı bağımsız parçaya bölünmüştür. Bunlar:

BIOS	Temel giriş/çıkış sistemi (Donanımına bağlıdır.)
BDOS	Temel disk işletim sistemi
CCP	Konsol komut işleyicisi
TPA	Seçici program alanıdır.

BIOS disket sürücülerine ve standart giriş/çıkış arabirimlere (Teletype, CRT, Paper Tape Reader/Punch, ...) erişilebilir. İçerdiği temel işlemleri gerçekleştirir. Kullanıcı firma kendi donanımına uygun için CP/M'ın bu kısmını değiştirir. BDOS birbirinden bağımsız küçük rutinlere sahip birden fazla disk sürücüsünü yönetir. BDOS sürücüler disk üzerinden en uygun şekilde alan tahsis eder. Bu sürücüler belirtilirken, bir taraftan da disk kafesinin başlarına yerleştirilen birer hareket ataması yapılır. Herhangi bir disk sürücüsüne erişim için gerekli olan her şey ile istenilen sonuçlara kavuşmak için yapılır. Bu standart CP/M sisteminde her disk bir ayrı küçük işlevdir. BDOS aşağıdaki temel işlemleri gerçekleştiren bölümlere erişilebilir. Giriş noktalarına sahiptir.

BİRİNCİ BÖLÜM

CP/M İŞLETİM SİSTEMİNİN

ÖZELLİKLERİ VE OLANAKLARININ TANITIMI

SEARCH	İstirahat belirli bir disk kütüğünde arama
OPEN	Diğer işlemler için kütük açılması
CLOSE	İşlemlerden sonra kütüğün kapatılması
RENAME	Belirli bir kütük isminin değiştirilmesi
READ	Belirli bir kütükten bir kayıta okunması
WRITE	Diska bir kayıt yazılması
SELECT	Diğer işlemler için belirli bir disk sürücüsünün seçilmesi.

1. GİRİŞ

CP/M monitor programı geniş bir yönetim paketi kütüğü üzerinden programlara hızlı bir şekilde ulaşım olanağı sağlar. CP/M mantıksal olarak dört ayrı bağımsız parçaya bölünmüştür. Bunlar:

BIOS	Temel giriş/çıkış sistemi (Donanıma bağlı bölümdür.)
BDOS	Temel disk işletim sistemi
CCP	Konsol komut işleyicisi
TPA	Geçici program alanıdır.

BIOS disket sürücülerine ve standart giriş/çıkış arabirimlere (Teletype, CRT, Paper Tape Reader/Punch.) erişilebilmesi için gerekli temel işlemleri gerçekleştirir. Kullanıcı firma kendi donanımına uyması için CP/M'in bu kısmını değiştirir. BDOS birbirinden bağımsız kütük rehberine sahip birden fazla disk sürücüsünü yönetir. BDOS kütüklere disk üzerinden en uygun şekilde alan tahsis etmek için stratejiler belirlerken, bir taraftan da disk kafasının bunlara erişirken minimum hareket etmesini planlar. Herhangi bir kütük bir diskin tüm alanını geçmemek şartıyla istenilen uzunlukta kayıta sahip olabilir. Bu standart CP/M sisteminde her disk 64 ayrı kütük içerebilir. BDOS aşağıdaki şu temel işlemleri gerçekleyen bölümlere erişebilen giriş noktalarına sahiptir.

SEARCH	İsmiyle belirli bir disk kütüğünün aranması
OPEN	Diğer işlemler için kütük açılması
CLOSE	İşlemlerden sonra kütüğün kapanması
RENAME	Belirli bir kütük isminin değiştirilmesi
READ	Belirli bir kütükten bir kayıttın okunması
WRITE	Diske bir kayıt yazılması
SELECT	Diğer işlemler için belirli bir disk sürücüsünün seçilmesi.

CCP kullanıcı konsolu ile CP/M'in kalan sistemleri arasında sembolik bir arabirim olanağı sağlar. CCP konsol cihazını okuyarak kütük rehberinin listelenmesi, kütüklerin içeriklerinin yazılması, çevirici, biçimleyici veya düzeltici gibi geçici programların çalışmasını kontrol etmek için verilen emirleri yürütür.

CP/M'in son elemanı "Transient Program Area" geçici program alanıdır. TPA, CCP ile yüklenmiş programları bulundurur. Örneğin, biçimleyici (Editör) ile program düzenlenirken TPA metin biçimleyici (text Editör) programının makina dilindeki kodunu ve veri alanlarını içerir. Benzer şekilde CP/M altındaki tüm programlar TPA'ya yüklenerek yürütülürler.

CP/M'de bellek 0000H adresi ile başlar. Ancak 8080 mikroişlemcisi ve eşdeğer sistemlerde kontrol 0000H adresindeki komutdadır. Bu bakımdan ilk yükleme sırasında 0000H adresi EPROM (Erasable Programmable Read Only Memory) üzerinde olmak zorundadır. Yükleme işlemi gerçekleştirildiği zaman donanım ve yazılım özellikleri kullanılarak EPROM yerine RAM (Random Access Memory) geçirilir. Yükleme işlemi özyükleme (bootstrap) olarak isimlendirilir. Yükleme işlemi, EPROM'da bulunan "boot" programının 0. iz ve 1. dilimdeki 128 baytlık yükleyiciyi belleğin en alt dilimine yerleştirilmesi ile başlar. Daha sonra bu yükleyici (bootstrap loader) CP/M işletim sistemine ait verileri 1. ve 2. izlerden belirtilen adreslere yerleştirir. Daha sonra sıfırıncı sayfaya sistem parametreleri yerleştirilir.

2. CP/M'İN İŞLEVLERİ

Kullanıcı CP/M işletim sistemini konsoldan komutları okuyan ve anlayan CCP vasıtasıyla yönetir. Genel olarak CCP çalışan disklerden birini adres-

ler. Bu diskler A,B,C,D diye isimlendirilir. Hangi disk üzerinde çalışıldığını belirlemek için CCP disk ismini ve bunu takip eden ">" işaretini monitörde gösterir. Bu gösterimden sonra CCP diğer bir komuta hazırdır. Başlangıç yüklemesinde CP/M A diskinden yüklenerek ekrana

xx K CP/M VER m.m

bildirisini yazar. xx KBayt olarak CP/M'in kullandığı bellek boyutu ve m.m CP/M'in uyarlama numarasıdır. Tüm CP/M sistemleri 16 K bellek boyutunda çalışmak üzere tasarlanmıştır; fakat istenilen bellek boyutuna göre sistem değiştirilebilir. Sistem koşullandırılmasından sonra CP/M A diskini yürürlüğe koyarak kullanıcıya "A>" sembolünü gösterir. Daha sonra CP/M komut bekler. Komutlar "Built-in" kalıcı komutlar ve "Transient" geçici komutlar şeklinde ikiye ayrılır.

2.1. GENEL KOMUT YAPISI

Kalıcı komutlar CCP programının bir bölümü iken geçici komutlar TPA'ya diskten yüklenerek yürütülürler. Kalıcı komutlar,

ERA	Belirlenen kütüğün silinmesini,
DIR	Rehberdeki kütük isimlerinin listelenmesini,
REN	Belirlenen kütük ismininde değiştirilmesini,
SAVE	Bellek içeriğinin kütükte saklanmasını,
TYPE	Yürürlükteki diskte bir kütük içeriğinin izlenmesini sağlar.

Genel olarak bu komutlar bir kütük veya kütükler grubunu belirler. Kütüklerin belirlenmesi aşağıdaki gibi olur.

2.2. KÜTÜK REFERANSLARI (FILE REFERENCES)

Kütük referansı sistemine eklenen belirli bir diskteki kütükleri tanımlar.

Kütük referansları "Unambiguous" (UFN) tamamen tanımlı veya "Ambiguous"

(AFN) tam tanımlı değil şeklinde olabilir. Bir UFN kütüğü referansı tek bir kütük tanımlarken, AFN kütük referansı değişik kütükleri belirler. Kütük referansı iki bölümü içerir. İlk isim ve ikinci isim. İkinci isim istek üzerine verilebildiği halde aslında faydalıdır. Örneğin "Asm" ikinci ismi bu kütüğün çevirici (Assembler) dilinde kaynak kütük olduğunu belirler. İlk isim ise kütüğün asıl ismidir. İki isim birbirlerinden "." ile ayrılır. Örneğin,

ppppppppp.sss.

gibi. Bu örnekte pppppppp sekiz veya daha az karakterden oluşan ilk isim ve sss üç karakterden fazla olmamak üzere ikinci isimdir. İkinci isim konmadığında üç adet boşluk karakterli ikinci isim atanır. Bir ufn kütük referansı

<> . , ; : = ? *] [

karakterlerini içeremez. Diğer alfanümerik karakterler ve özel karakterler kullanılabilir.

Bir afn kütük referansı rehberdeki kütük isimlerini tararken ve isim uyumu aranırken kullanılır. Afn kütük referansı Ufn kütük referansının formuna benzer; fakat Afn kütük referansında ilk veya ikinci isim içinde "?" karakterine izin vardır. "?" karakteri bulunduğu yerdeki karakterin herhangi bir alfanümerik karakter olabileceğini belirler. Böylece,

X?Z.C?M

Afn kütük referansını

XYZ.COM

şeklindeki örnekler Ufn kütük referanslarını gösterir.

,

şeklindeki Afn kütük referansı

?????????.???

Şeklindeki Ufn kütük referansına eşdeğerdir.

pppppppp . * = pppppppp . ??? ve

* . sss = ???????? . sss

özdeşirler. Örneğin,

ERA X.Y

tüm diskteki X.Y kütüğünü siler. Aşağıdaki kütük isimleri geçerli Ufn referanslarıdır.

X XYZ GAMMA

X.Y XYZ.COM GAMMA.1

Ek bir özellik olarak kullanıcı kütük isimlerini disk ismi ile beraber bildirebilir. Bu yöntemde disk ismini ":" karakteri takip etmeli, daha sonra kütük ismi gelmelidir. Aşağıdaki bildirimler geçerlidir.

A:X Y B:XYZ C:GAMMA

A:XYZ.COM B:X.A?M C:*.ASM

Bütün küçük yazı alfabetik harfler CCP tarafından işlenirken büyük harflere dönüştürülür.

3. DİSKLERİN DEĞİŞTİRİLMESİ

CCP konsoldan komut beklerken kullanıcı disk sürücü isimleri A,B,C veya D ve bunları takip eden ":" karakterini yazarak yürürlükte olan diski değiştirebilir. Böylece aşağıdaki örnekte olduğu gibi CP/M, A diskinden yüklendikten sonra disk değişmesi sağlanabilir.

16 K CP/M VER 1.4

A>DIR A disketindeki tüm kütükleri listele.

A: SAMPLE ASM A: NEL COM

A: SAMPLE PRN A: NEL BAS

A>B: B disketi ile çalışmaya başla

B>DIR * . ASM

B>A: Tekrar A disketine geç.

4. KALICI (BUILT-IN) KOMUTLARIN YAPISI

Kalıcı komutlarının yapısında yukarda anlatılan kütük ve cihaz referansları önem kazanır.

Ufn-"Unambiguous" tanımlı kütük referansı

Afn-"Ambiguous" tamamen tanımlı olmayan kütük referansı

<Cr>-"Carriage return" geri dön tuşudur.

4.1. ERA AFN <cr>

ERA komutu o an kullanılan diskteki afn kütüğünü siler. ERA komutu ile kütük referansına uyan birden fazla kütükde silinebilir. Aşağıda ERA komutunun geçerli şekilleri görülmektedir.

ERA x.y Yürürlükteki diskten X.Y isimli kütük rehberden kaldırılmak üzere silinir.

ERA X.* İlk isim X olan tüm kütükler kullanılan diskten silinir.

ERA *.ASM İkinci ismi ASM olan tüm kütükler kullanılan diskten silinir.

ERA X?X.C?M X?Y.C?M Afn kütük referansına uyan tüm kütükler kullanılan diskten silinir.

ERA *. * Diskteki tüm kütükler silinir. Bu komut CCP, "ALL FILES(Y/N) ." (Tüm kütükler ?) sorusuyla cevap verir. Kütüklerin silinmesi için Y tuşuna basılmalıdır.

ERA B:*.PRN B diskindeki, ikinci ismi PRN olan kütükler silinir.

4.2. DIR AFN <cr>

DIR komutu Afn kütük ismine uyan tüm kütüklerin konsol düzeninde listelenmesini sağlar. Özel bir hal olarak DIR kullanılan diskteki tüm kütükleri listeler. Bu komut DIR * . * ile aynı anlamı taşır. Geçerli DIR komutları aşağıdaki gibidir.

DIR X.Y

DIR X?Z. C?M

DIR ??Y

CCP'nin diğer komutlarına benzer şekilde DIR komutunda Afn kütük isminin önünde, disk sürücü ismi de bulunabilir.

DIR B:

DIR B:X.Y

DIR B:*. A?M

Rehberde istenilen kütük bulunmuyorsa, CRT ekranında "NOT FOUND" bulunmadı bildirisi görülür.

4.3. REN UFN1=UFN2 <cr>

REN komutu diskteki kütük isimlerinin değiştirilmesi içindir. Yürürlükteki diskin daha önceden ufn 2 isminde bir kütüğe sahip olduğu kabul edilir. CCP eşitlik işareti yerine kullanıcı konsolunda mevcut ise sola yönelmiş ok işaretini de kabul eder. REN komutuna örnek olarak

REN X.Y=Q.R QR kütüğünün ismi X.Y olarak değiştirilir.

REN XYZ.COM=XYZ.XXX XYZ.XXX kütüğü XYZ.COM ismini alır.

Operatör Ufn 1 veya Ufn 2 veyahutta her ikisini de disk sürücüsü ismini de belirterek tanımlayabilir. Ufn 1 önünde disk sürücüsü ismi varsa ve Ufn 2 tanımlanırken, disk sürücüsü yazılmazsa iki kütüğün de o disk sürücüsünde olduğu kabul edilir. Benzer şekilde sadece Ufn 2 tanımlanırken de disk sürücü ismi belirtilebilir. Geçerli REN komutları aşağıdadır.

REN A:X.ASM=Y.ASM A diskindeki Y.ASM kütüğü X.ASM şeklinde isimlendirilmiştir.

REN B:ZAP.BAS=ZOT.BAS B diskinde ZOT.BAS kütüğü ZAP.BAS ismini alır.

REN B:A.ASM=B:A.BAK B diskindeki A.BAK kütüğü A.ASM olarak isimlendirilir.

Ufn 1 kütüğü daha önceden var ise CCP "FILE EXISTS" kütük mevcut şeklinde bir bildiri ile bunu belirtecek ve değişikliği yapmayacaktır. UFN 2 kütüğü belirtilen diskte yok ise "NOT FOUND" (bulunamadı) mesajı CRT ekranına yazılacaktır.

4.4. SAVE n UFN <cr>

SAVE komutu TPA'dan 256 baytlık n adet sayfayı diske yerleştirerek bu kütüğe Ufn ismi verir. CP/M sisteminde TPA 100H adresinden başlar. Böylece kullanıcı programı 100H ile 2 FF H arasında ise 2 sayfa bellek diske saklanmalıdır. Makina dilindeki kütük hiçbir işlem yapılmadan tekrar TPA'ya yüklenip işleme sokulabilir. Örneğin,

SAVE 3 X.COM 100 H ile 3 FF adresleri arasını diske kopya eder. Kütük ismi X.COM olur.

SAVE 40 Q 100 H ile 28 FF H Q adresleri arasını Q kütüğüne yazar.

SAVE 4 X.Y 100 H ile 4 FF H adresleri arasındaki bölge X.Y kütüğüne yazılır.

SAVE komutu Afn alanında disk sürücüsü tanımlanarak da kullanılabilir.

Örneğin,

SAVE 10 B:ZOT.COM B diskinde ZOT.COM kütüğüne 10 sayfa kaydet.

4.5. TYPE UFN <cr>

TYPE komutu yürürlükte olan diskteki ASCII kaynak kütük içeriğini ekranda görüntüler. Geçerli TYPE komutları

TYPE X.Y

TYPE X.PLM

TYPE XXX

TYPE komutu Ufn'den önce disk sürücüsünü de belirliyebilir.

TYPE B:X.PRN B diskindeki X.PRN kütüğünün içeriği görüntülenir.

5. SATIR DÜZELTİMİ VE ÇIKIŞ KONTROLLARI

CCP komutları yazarken belirli satır düzeltme işlevlerine izin verir. Bu işlevler aşağıda belirtilmiştir.

RUBOUT	Konsoldan yazılmış en son karakteri veya devamını siler (Backspace)
Ctl-U	Konsoldan yazılmış en son satırı siler.
Ctl-X	Ctl-U ile aynı işlemi yapıyor.
Ctl-R	O anki komut satırını tekrar yazar.
Ctl-E	Satırın fiziksel sonunu bildirir.
Ctl-C	CP/M işletim sistemini tekrar başlatır.
Ctl-Z	PIP ve ED programlarında konsoldan işlem sonunu belirler.
Ctl-P	Konsoldaki tüm çıkışları o an tanımlı listeleme cihazına kopya eder. Tekrar basılan CTL-P komutuna kadar çıkış listelemeye devam eder.
Ctl-S	Konsol çıkışını geçici olarak keser. Tekrar başladığında program icrası ve çıkış devam eder. Bu özellik CRT gibi hızlı çıkış alınabilen birimler içindir. Böylece çıkışın devamı gelmeden arkadaki bölümler izlenebilir.

CCP komut satırları 255 karakter uzunluğunda olabilir ve geri dön tuşuna basılana kadar işleme tabi tutulmazlar.

6. GEÇİCİ KOMUTLAR

Geçici komutlar o an kullanılan diskten TPA'ya yüklenerek icra edilirler. CCP'nin altında yürütülen geçici komutlar aşağıda verilmiştir. Benzer şekilde ek fonksiyonlar kullanıcı tarafından tanımlanabilir.

STAT	Kullanılan diskteki belleğin boyutu listelenir. Belirli kütükler hakkında istatistiksel bilgi sağlar, ve cihaz tanımlamasını gösterir veya değiştirir.
------	--

ASM	CP/M Assembler'i yükler ve diskteki belirtilmiş kütükleri çevirir.
LOAD	Intel'in "hexadecimal" makina kodu formatında bir kütük yükler ve TPA alanında yüklenerek yürütülebilen bir kütük üretir.
DDT	CP/M DDT programını yükler ve yürütmeye başlar.
PIP	"Peripheral Interchange Program"ı yükler ve arabirim cihazları arasındaki iletişimi sağlar.
ED	CP/M metin biçimleyici programını yükler ve yürütür.
SYSGEN	Yeni bir CP/M sistem disketi yaratır.
SUBMIT	Dizi işlemi için komutlardan oluşan kütüğü yürürlüğe koyar.
DUMP	Kütüğün "Hexadecimal" içeriğini gösterir.
MOVCPM	CP/M sistemini belirli bellek boyutuna göre yeniden oluşturur.

6.1. STAT <cr>

STAT komutu kütük saklama alanları ve atanan cihazlar hakkında istatistiksel bilgiler sağlar. Aşağıda görülen iki biçimde yazılarak kullanılır.

STAT <cr>

STAT "Komut Satırı" <cr>

STAT "komut satırı" <cr> o an atanan cihazların incelenmesini ve değiştirilmesini sağlar. Çeşitli komut satırları ve gördüğü işlem aşağıda gösterilmiştir.

STAT <cr> Kullanıcı boş bir komut satırı verirse STAT aktif tüm sürücülerin üzerindeki disketlerde boş olan alanları hesaplar ve

X:R/W SPACE:nnnK veya

X:R/O SPACE:nnnK bildirisini yazar. X aktif disk sürücüsünü, R/W diske-
tin erişim modunun okunabilir/yazılabilir olduğunu,
R/O ise yalnız okunabilir olduğunu gösterir. X sürücüsünde bulunan diskette kalan alan nnnK Bayt olarak bildirilir.

- STAT X:<cr> Sürücü ismi belirtildiğinde istatistiksel hesaplama başlamadan önce o sürücü seçilir. Böylece "STAT B:" komutu A sürücüsünde iken yürütülebilir ve BYTES REMAINING ON B:nnnK bildirisi ekrana yazılır.
- STAT AFN <cr> Komut satırı STAT tarafından taranacak kütük ismini de belirtebilir. Afn'ye uyan kütükler alfabetik sıraya göre listelenir ve her kütüğün kapladığı alan RECS BYTS EX D:PILENAME.TYP rrrr bbbK ee d:pppppppp.sss şeklinde bildirilir. rrrr 128 baytlık kayıt sayısı, bbb kütüğün Kbayt olarak kapladığı alan ($bbb=rrrr \cdot 128/1024$). ee 16 K'nın kaç katı uzun olduğu ($ee=bbb/16$), d kütüğü içeren disk ismi, pppppppp 8 karaktere kadar uzunluktaki kütüğün birinci ismidir. Liste sonunda kalan bellek alanı özetlenmiştir.
- STAT X:AFN <cr> Değişik bir şekil olarak komut satırında önce disk sürücüsü ismi yazılabilir. Bu durumda önce belirlenen disk seçilir. Daha sonra STAT komutu yürütülür.
- STAT X:R/O <cr> Bu durumda X sürücüsü üzerindeki disket yalnız okunabilir/olarak tanımlanır. Bu şekil bir sonraki sıcak veya soğuk başlama komutu gelinceye kadar etkili olur. Disk R/O olarak tanımlı ise bu diske yazma isteği olduğunda
- BDOS ERR ON X : READ ONLY mesajı bildirilir.
- STAT komutu diğer fiziksel düzenlerin kontrol edilmesine izin verir. Genelde herbiri birçok fiziksel giriş/çıkış düzenine bağlanmış dört lojik giriş/çıkış elemanı vardır. Bunlar,

CON: Sistem konsol düzeni (Operatör ile CCP arasında iletişim amacıyla kullanılır)

RDR: Kağıt okuyucu düzeni

PUN: Kağıt delici düzeni

LST: Çıkış listelerini alma düzeni (yazıcı)

Belirli bir bilgisayar sistemine bağlanmış düzenler CP/M'in BIOS bölümündeki alt programlar ile yönlendirilir. Böylece örneğin lojik RDR düzeni gerçekte hızlı okuyucu, teletype okuyucu veya kasetli teyp olabilir.

TTY: Teletype düzeni (Düşük hızlı konsol)

CRT: Katod ışıklı tüp düzeni (Yüksek hızlı konsol)

BAT: Birlikte işleme düzeni olarak tanımlanmış (Konsol o anki RDR ve çıkış o anki LST cihazıdır)

UC1: Kullanıcı tarafından tanımlanan konsol

PTR: Kağıt bant okuyucu

UR1: Kullanıcı tarafından tanımlanan okuyucu #1

UR2: Kullanıcı tarafından tanımlanan okuyucu #2

PTP: Kağıt bant delici (Yüksek hızlı delici)

UP1: Kullanıcı tarafından tanımlanan delici #1

UP2: Kullanıcı tarafından tanımlanan delici #2

LPT: Satır yazıcı

UL1: Kullanıcı tarafından tanımlanan listeleme düzeni #1

Fiziksel düzenlerin gerçekte ismini aldığı lojik giriş/çıkış elemanı niteliği ile aynı olması gerekmez. Örneğin PTP düzeni bir kaset kayıt işlemi yapan düzen olarak tasarlanabilir. Düzenin tam niteliği düzeni süren alt program CP/M'in BIOS kısmında tanımlanır.

Sistemin esnek bir yaklaşımı olarak, kullanıcı fiziksel ve mantıksal düzenlerin yeniden atanmasına izin veren "IOBYTE" işlevini kullanabilir.

"IOBYTE" işlevi, CP/M işlemi esnasında değiştirilebilen fiziksel ve mantıksal düzenlerin bir haritasını üretir. IOBYTE işlevi INTEL standartlarına uygun olarak aşağıdaki gibi tanımlanmıştır. Bir tek bellek bölgesi (00003H adresi) IOBYTE olarak isimlendirilmiştir. 00003H deki değer her iki bit bir değer göstermek üzere 4 çevre düzenine ayrılmıştır.

IOBYTE=00003H	LIST	PUNCH	READER	CONSOLE
	(6.,7.bit)	(4.,5.bit)	(2.,3.bit)	(0.,1.bit)

Her alandaki değer 0-3 arasında değişir. Bu değerler kaynak veya aktarılacak çevre düzenini tanımlar.

"CONSOLE" alanı (0. ve 1. bitler)

- 0 "CONSOLE" bir seri yazıcı düzenine ayrılır (TTY:)
- 1 "CONSOLE" CRT düzenine ayrılır. (CRT:)
- 2 "CONSOLE" " " " "
- 3 "CONSOLE" bir seri yazıcı düzenine ayrılır (TTY:)

"READER" alanı (2. ve 3. bitler)

- 0 "READER" bir seri yazıcı düzenine ayrılır (TTY:)
- 1 "READER" CRT düzenine ayrılır (CRT:)
- 2 "READER" " " " "
- 3 "READER" bir seri yazıcı düzenine ayrılır (TTY:)

"PUNCH" alanı (4. ve 5. bitler)

- 0 "PUNCH" bir seri yazıcı düzenine ayrılır (TTY:)
- 1 "PUNCH" CRT düzenine ayrılır (CRT:)
- 2 "PUNCH" bir paralel yazıcı düzenine ayrılır (LPT:)
- 3 "PUNCH" CRT düzenine ayrılır (CRT:)

"LIST" alanı (6. ve 7. bitler)

- 0 "LIST" bir seri yazıcı düzenine ayrılır (LST:)

- 1 "LIST" CRT düzenine ayrılır.(CRT:)
- 2 "LIST" bir paralel yazıcı düzenine ayrılır. (LPT:)
- 3 "LIST" bir seri yazıcı düzenine ayrılır.(LST:)

Düzenlerin hangi fiziksel düzenlere karşılık geldiği,

STAT VAL: <cr>

yazılarak görülebilir. STAT her mantıksal düzene karşı gelen o an mümkün olan düzenlerin listesini verir.

CON:=TTY: CRT: BAT: UC1:

RDR:=TTY: PTR: UR1: UR2:

PUN:=TTY: PTP: UP1: UP2:

LST:=TTY: CRT: LPT: UL1:

Her satırda solda görülen mantıksal düzen sağdaki dört fiziksel düzenden birine atanabilir. O anki lojik-fiziksel atama,

STAT DEV:<cr>

komutuyla görülebilir. Bu komut solda mantıksal düzenleri ve sağda ise o anki fiziksel karşılıklarını listeler. Örneğin,

CON:=CRT:

RDR:=UR1:

PUN:=PTP:

LST:=TTY:

O anki mantıksal düzenlerin fiziksel düzenlere atanması,

STAT ld1=pd1, ld2=pd2,<cr>

şeklinde bir komut ile gerçekleşir. Burada ld1, ld2 mantıksal düzen isimleri, pd1 pd2 ise atanan fiziksel düzenlerin isimleridir. Aşağıda o anki atamaları değiştiren geçerli STAT komutları görülmektedir.

STAT CON:=CRT:<cr>

STAT CON:=TTY:, LST:=LPT<cr>

STAT VAL:

komutu mümkün olan durum komutlarını

Temp R/O Disk: d:=R/O

Set Indicator: d:file name.type \$ R/O \$ R/W \$ SYS \$ DIR

Disk Status : DSK:d:DSK:

User Status : USR:

Iobyte Assign:

çıkışını vererek bir liste şeklinde özetler. Örnek olarak,

STAT d : File name.typ \$S.

komutunda "d:" yazılması şart olmayan disk sürücüsü ismi, "filename.type"

Afn veya Ufn olan kütük ismidir.

Ekranda çıkış şekli aşağıdaki gibi olan

Size	Recs	Bytes	Ex	Acc
48	48	6K	1	R/O A:ED.COM
55	55	12K	1	R/O (A:PIP.COM)
65536	128	2K	2	R/W A:X.DAT

bildirisi yazılır. "\$S" parametresi kütük için ayrılan alanın boyutunu verir. Doldurulacak maksimum alan boyutunu gösterir. \$S yoksa boyut alanı görmeden geçilir. Fakat kalıcı alanlar gösterilir.

"Recs" her genişletmedeki gerçek kayıt sayısını (128 baytlık) gösterir. Ardışıl erişimli olarak oluşturulmuş kütükler için boyut ve kayıt alanları aynı değeri taşır.

"Bytes" alanı KBayt olarak kütüğe ayrılmış alanın gerçek değerini verir. Ardışıl erişimli kütükler için boş bölgelerde bu bölümde gösterilir. Yani disket üzerinde bir kütüğe minimum 2K'lık bir ayrılmış olur. (Amstrad CPC464'de 1K'lık bir yer ayrılmaktadır) "Ext" alanı kütüğe ayrılmış 16 KBaytlık blokları gösterir.

"Acc" alanı R/O veya R/W olarak kütüğün erişim modunu gösterir. Aşağıdaki komutlar ile kütük erişim durumları değiştirilebilir. Parantez içinde gösterilen PIP.COM kütüğü, sistem kütüğü haline getirilmiştir. Bu tekrar DIR komutu ile rehberde gösterilen kütük haline getirilebilir. Dört komut şekli aşağıdadır.

STAT d: filename.typ \$R/O

STAT d: filename.typ \$R/W

STAT d: filename.typ \$DIR

STAT d: filename.typ \$SYS

R/O göstergesi: Kütük bir sonraki STAT komutu ile düzeltilmedikçe yalnız okunabilir hiçbirşey yazılamaz.

R/W göstergesi: Kütük hem okunur hem de yazılabilir.

SYS göstergesi: Kütüğün . sistem kütüğü haline getirilmesini sağlar.

DIR göstergesi: Kütüğü tekrar rehber içine alır.

STAT d : DSK

komutu "d " diskinin sürücü karakter isteklerinin bildirilmesine neden olur. Ekranda,

```
d      : Drive Characteristics
65535  : 128 Byte Record Capacity
8192   : Kilobyte Drive Capacity
128    : 32 Byte Directory Entries
Ø      : Checked Directory Entries
1Ø24   : Records/Extent
128    : Records/Block
58     : Sectors/Track
2      : Reserved/Track
```

şeklinde görülen mesajda d: disk sürücüsü ismi bunun altında toplam kayıt kapasitesi ve bunun altında toplam bellek Kbyte olarak bildirilir. Rehber büyüklüğü ve test edilen giriş noktaları bunun altında yazılmıştır. Değiştirilebilen bellek ortamları için rehber büyüklüğü ile test edilen girişlerin sayısı aynıdır.

Sabit ortamlar için bu genellikle sıfırdır. Extent başına kayıt sayısı rehber girişlerinin adresleme kapasitesini belirler. Blok başına kayıt sayısı temel yerleştirme alanını belirler. Listede daha sonra her izdeki dilim sayısı ve yedek izlerin sayısı bulunur.

STAT DSK:

komutu ise o an erişilebilecek tüm disk sürücülerin karakteristiklerini listeler. En son

STAT USR:

o an kullanılan diskte kütüğü olan kullanıcıların numaralarını

Active User : Ø

Active Files : Ø 1 3

şeklinde verir. Operatör USER 1, USER 2, USER 3 şeklindeki komutlar ve DIR komutu ile diğer kullanıcıların rehberlerini izleyebilir.

6.2. ASM UFN <cr>

ASM komutu CP/M 8080 "Assembler" programını yükler ve yürütür. Ufn "Assembler" dilinde komutları içeren ve ikinci ismi ASM olmalıdır. Aşağıdaki ASM komutları geçerlidir.

ASM X

ASM GAMMA

iki geçişli "Assembler" otomatik olarak yürütülür. İkinci geçişte derleme hataları olduğunda bunlar konsola yazılır."Assembler"

X.PRN

şeklinde bir kütük yaratır. Burada X ASM komutunda belirtilen ilk ismidir. PRN kütüğü kaynak programının listesini bununla beraber her komutun makina dilindeki kodunu ve varsa hata mesajlarını içerir.

PRN kütüğü TYPE komutu kullanılarak konsolda listelenebilir veya herhangi

bir giriş/çıkış düzenine PIP komutuyla gönderilebilir. PRN kütüğünün sol 16 sütununda önemli "Assembler" bilgileri ile donatılmış asıl kaynak programı içerdiğine dikkat etmek gerekir. Böylece PRN kütüğü asıl kaynak programına yedek kütük (back-up) olarak kullanılabilir. Kaynak programı bir kaza ile bozulursa PRN kütüğü ED programı kullanılarak soldaki 16 karakter silinerek bunun yerine kullanılabilir. Bunun için PRN ismi REN komutu ile ASM'ye dönüştürülmelidir.

X.HEX

kütüğü 8080 makina dilinde yazılmış yükleme ve yürütme işlemleri için hazır bir kütük olarak üretilir. Diğer geçici komutlara benzer olarak derlenecek kaynak kütük herhangi bir diskten o diskin ismi yazılarak yüklenebilir. Böylece,

ASM B: ALPHA <cr>

komutu ASM programının o an kullanılan diskten yükler ve "B" diskindeki ALPHA programını derler. Bu durumda HEX ve PRN kütüklerinin her ikisi de B diskinde oluşur.

6.3. LOAD UFN <cr>

LOAD komutu makina dilinde olduğu kabul edilen ufn kütüğünü okur ve hemen yürütülebilecek bir bellek görüntüsü oluşturur. ufn kütük isminin

X.HEX

biçiminde olduğu kabul edilir. Böylece komutta sadece X isminin olması yeterlidir. LOAD komutu

X.COM

şeklinde isimlendirilen bir kütük meydana getirir. Bu isim o kütüğün makina tarafından yürütülebilen kodları içerdiğini belirtir. Kullanıcı CCP gösterge karakteri ">" den sonra sadece kütük ismi X yazdığında kütük belge yüklenir ve yürütülür.

Genel olarak CCP X ismini okuduğunda kalıcı komut isimlerine bakar. Böyle bir işlev bulunmadığında CCP sistem disk rehberine bakarak

X.COM

isminde kütük arar. Böyle bir isim bulunursa makina dilindeki program TPA'ya yüklenir ve yürütülür. Böylece kullanıcı CCP'de yeni geçici komutlar tanımlamış olur. LOAD komutu disk ismi belirleyerek de kullanılabilir.

Örneğin,

LOAD B: BETA

LOAD yükleme programını o an kullanılan diskten yükler ve B diskinden BETA programı üzerinde işlem yapmaya başlar. BETA.HEX kütüğü geçerli Intel makina dili komutlarına sahip olmalıdır. Bundan da öte makina dilinde kayıtlar birbiri ardında olmalıdır. Boşluklar LOAD komutuyla sıfırlarla doldurulur. Bunun için LOAD komutu TPA'da çalışan standart COM kütükleri meydana getirmek için kullanılmalıdır. TPA'da çalışmıyacak programlar DDT ile yüklenebilir.

6.4. PIP <cr>

PIP programı disk kütükleri için yükleme, yazma, kağıt delme, kopya çıkartma ve birleştirme gibi temel ortam dönüştürme işlemlerini yapan bir programdır. Geçerli PIP komutları aşağıdaki gibidir.

PIP <cr>

PIP "komut satırı" <cr>

Her iki durumda da PIP TPA'ya yüklenir ve yürütülür. Birinci durumda PIP komut satırlarını konsoldan direkt okur. Bu durumda ekranda "*" karakteri işaretleyici olarak kullanılır ve boş satır girilene kadar bu devam eder. İkinci şekilde komut satırı diğerinden farklı olarak PIP program çağrısı ile beraber yazıldığından işaretleyici ekranda görülmeden hemen ortam değişikliği gerçekleşir. Her komut satırı

Varış = Kaynak #1, Kaynak #2, Kaynak #n <cr>

şeklinde olup varış veriyi alacak kütük veya çevre düzeni, kaynak varışa aktarılacak kütük veya düzen isimleridir.

Komut satırında birden fazla kütük belirlendiğinde bunların herbirinin sonundaki Ctl-Z kütüğün ASCII karakterler içerdiğini gösterir. Yine kullanıcı konsolu sola dönük oku içeriyorsa eşitlik için bu karakter kullanılarak okunabilirlik kolaştırılabilir. Küçük harf ASCII karakterler, içerde büyük harflere dönüştürülerek CP/M kütük isimleri veya çevre düzen isimleri ile uyum sağlanır. Son olarak komut satırları 255 karakteri geçmemelidir.

Varış veya kaynak elemanları disk ismi belirlenerek veya belirlenmeden tam belirlenmemiş CP/M kaynak kütük referansları olabilir. Herhangi bir kütük önüne (A:, B:, C: veya D:) disk isimleri konularak belirtildiğinde bu kütük tanımlanan diskte kaynak veya varış yeri olarak anlaşılır. Sürücü ismi yazılmadığında o an yürürlükte olan disk kastedilmiştir. Aşağıdaki komut satırları ve açıklamaları PIP için geçerli girişlerdir.

X=Y <cr> Y isimli kütüğü X isimli kütüğe aktar. X ve Y tam belirlenmiş kütüklerdir ve Y işleminden etkilenmez.

X=Y,Z <cr> Y ve Z kütükleri değişmeden birleştirilerek X kütüğüne aktarılır.

X.ASM=Y.ASM,Z.ASM, FIN.ASM <cr> ASM tipinde Y,Z ve FIN kütüklerini birleştirerek X.ASM kütüğünü oluşturur.

NEW.ZOT=B:OLD.ZAP <cr> B diskindeki OLD.ZAP kütüğünü o an kullanılan diske NEW.ZOT olarak aktar

B:A.U=B:B.V,A=C.W,D.X <cr> B diskinde B.V kütüğü ile A daki C.W ve D.X kütüklerini birleştirerek B de A.U yaratılır.

Kolay kullanım için PIP disk sürücüleri arasında kütük transferi gerçekleştirilmek üzere diğer komut şekillerini de içerir. Bunlar,

PIP x: = afn <cr>

PIP x: = y:afn <cr>

PIP ufn= y:<cr>

PIP x: ufn = y:<cr>

İlk şekilde kütük ismine uyan yürürlükteki tüm kütükler, x: diskinde aynı isimlerle kopya edilir. İkinci durumda y diskinden ilk durumdaki işlem yapılır. Üçüncü komut "PIP ufn=y: ufn <cr> komutuna özdeştir y diskinden ufn kütüğü yürürlükteki diske kopya edilir. Dördüncü komut üçüncü komutla aynıdır.

Tüm durumlarda kaynak ve varış yerinin ayrı ayrı diskler olduğuna dikkat edilmelidir. Afn belirlenmiş ise PIP kopya işlemini yaparken afn'ye uyan ufn'lerin isimlerini yazar. Varış kütüğü ile aynı isimli bir kütük eskiden var ise kopya işlemi başarı ile gerçekleştiğinde bu kütük kaldırılır. Aşağıda PIP komutları ile diskten diske kopya işlemleri görülmektedir.

B:= *.COM <cr> Yürürlükteki tüm COM kütükleri B diskine kopya edilir.

A:=B: ZAP.* <cr> B diskinden ilk ismi ZAP olan tüm kütükler A diskine kopya edilir.

ZAP.ASM=B: <cr> ZAP.ASM=B:ZAP.ASM komutuna özdeştir.

B:ZOT.COM=A: <cr> B:ZOT.COM=A:ZOT.COM komutuna özdeştir.

B:=GAMMA.BAS <cr> B:GAMMA.BAS=GAMMA.BAS komutuna özdeştir.

B:=A:GAMMA.BAS <cr> B:GAMMA.BAS=A:GAMMA.BAS komutuna eşittir.

PIP komutları içinde kullanılan ilâve düzen isimleri aşağıda verilmiştir.

NUL: Cihaza 40 tane boşluk (ASCII Ø karakteri) yollanır. Delici çıkışlarının sonunda basılabilir.

EOF: CP/M kütük sonu bildirisi (Ctl-Z) varış cihazına yollanır. Otomatik olarak PIP içinde bütün ASCII veri transferlerinin sonunda gönderilir.

INP: Özel PIP giriş kaynağıdır.

OUT: Özel PIP çıkış varış yeridir.

PRN: LST ile aynıdır. Tab işlemi her sekiz karakterde bire yükseltilmiştir. Satırlar numaralanır ve her 60 satırda sayfa boşlukları eklenir. (T8NP) komutu ile aynıdır.

Kütük ve düzen isimleri PIP komutunda birlikte kullanılabilir. ASCII kütükler için ctl-z ve ASCII olmayan disk kütükleri için kütüğün gerçek sonu gelene kadar belirlenen düzen okunur. Düzendeki veri soldan sağa son karakter okunana kadar taşınır. Varış bir disk kütüğü ise kopya işleminin başlamasından sonra gerçek isme dönüştürülen geçici (\$\$\$ ikinci isminde) bir tampon kütüğü üretildiğine dikkat edilmelidir. COM ikinci ismiyle tanımlı kütükler ASCII olmayan veri içerirler.

Kopya işlemi herhangi bir tuşa basılarak kesilebilir. PIP işleminin bittiğini "ABORTED" mesajı bildirir. İşlem sırasında bir hata olduğunda veya işlem kesildiğinde SUBMIT komutuyla hazırlanan mesajlar verilir. Geçerli PIP komutları aşağıda gösterilmiştir.

PIP LST:=X.PRN <cr> : PIP programını çağırarak X.PRN kütüğü LST cihazına aktarılır.

PIP <cr> : PIP programı daha sonra verilecek bir dizi komut için çağırılır.

*CON:=X.ASM,Y.ASM,Z.ASM <cr> : Üç ASM kütüğünü birleştirerek CON cihazına kopya et.

*X.HEX=CON: ,Y,HEX,PTR: <cr> : CON cihazı ctl-z basılana kadar okunarak Y.HEX kütüğü ile PTR'den alınan veriler ile bir HEX kütüğü hazırlanır.

PIP PUN:=NUL:;X.ASM,EOF:;NUL: <cr>: Delgi cihazına 40 adet boşluk karakteri yollanarak daha sonra X.ASM kütüğü ve yine 40 boşluk karakteri kopya edilir.

* <cr> : Tek <cr> e basılınca PIP programından çıkılır.

Kullanıcı PIP komutlarında soldan ve sağdan köşeli parantez içine alınmış PIP parametrelerini kullanır. Bu parametreler kopya işlemine etki ederler. Sıfır veya boşluk ile birbirinden ayrılmış parantezler komuttaki kütük isimlerini hemen takip etmelidir. Genel olarak her parametreyi yazılması isteğe bağlı olan bir onlu tamsayı izler. Geçerli PIP parametreleri aşağıdadır.

B Blok transfer modunu belirler. Kaynak cihazdan ctl-s karakteri alınana dek PIP veriyi tampon alanda tutar. Bu özellik örneğin kaset okuyucu gibi sürekli okuyan cihazdan bir disk kütüğüne veri transferinin gerçekleşmesini sağlar. Tamponlanacak alan ana sistemin belleği kadardır.

Dn n sütündan sonraki karakterler veri transferi sırasında ortadan kaldırılır. Dar bir yazıcıya veya konsol düzenine uzun satırları içeren veriler yollandığında kullanılır.

E Transfer işlemleri yapılırken bunların ekrana yansması sağlanır.

F Kütükten "form feed" alanları filtre edilir. P parametresi ile yeni "form feed" alanları tanımlanabilir.

H Onaltılı veri transferi; Tüm verilerin Intel makina diline uyup uymadığı kontrol edilir. Onaltılı kayıtlar arasındaki geçersiz karakterler kaldırılır. Düzeltme işlemi çabuk olmalıdır.

- I Intel makina dilinde bir kütük transferinde 00 kayıtları ihmal edilir. I parametresi otomatik olarak H parametresini de yürürlüğe koyar.
- L Büyük harf yazılmış karakterler küçük harfe dönüştürülür.
- N Varış yerine her satır transfer edildiğinde birden başlayarak ve birer artan satır numaraları eklenir. Öndeki sıfırlar bastırılmıştır. N2 olarak belirlenmişse öndeki sıfırlar dahil edilir. Ve bir "tab" yerleştirilir.
- O Amaç kütük (ASCII olmayan) transferini belirler. CP/M'in kütük sonu belirlemesi tanımsız kılınır.
- Pn n satırda bir sayfa aralığı bırakılır. n=1 ise 60 satırda bir bu işlem görülür.
- Qs +Z Yazısal s (ctl-z) ile karşılığında kaynak düzenden veya kütükten kopya işlemi kesilir.
- Ss +Z Yazısal s (ctl-z) ile karşılaşıldığında kopya işlemi başlar. Q ve S parametreleri bir kütük içindeki alt program gibi bölümlerin transfer edilmesinde kullanılır.
- Tn n inci sütündan sonra tab işlemi araya sokulur.
- U Transfer işlemi sırasında küçük harfler büyük harflere dönüştürülür.
- V Varış yeri bir disk kütüğü olduğunda yazma işleminden sonra okuma yapılarak veri transferinin doğru yapılıp yapılmadığı test edilir.
- Z Her ASCII karakterin eşlik (parity) biti sıfırlanır.

Aşağıda kütük transfer işlemlerinde kullanılan geçerli PIP komutları ve PIP sistem parametreleri verilmiştir.

PIP X.ASM=B: [v] <cr> B sürücüsünden X.ASM kütüğü yürürlükteki diske kopya edilir ve doğruluğuna bakılır.

PIP LPT:=X.ASM [nt8u] <cr> X.ASM kütüğü LPT cihazına kopya edilirken her satır numaralanır her 8 satırda boşluk bırakılır ve küçük harfler büyük harflere dönüştürülür.

PIP PUN:=X.HEX [i], Y.ZOT [h] <cr> X.HEX kütüğü delgi düzenine, X.HEX içindeki 00 kayıtları ihmal edilerek, daha sonra Y.ZOT kütüğündeki "HEX" veriler 00 kayıtları dahil olmak üzere aktarılır.

PIP X.LIB=Y.ASM [SSUBR1:†ZqJmp L3†Z] Y.ASM kütüğü SUBR1 yazısına rastlanıldığında X.LIB kütüğüne kopya edilir ve bu işlem JMP L3 yazısına rastlanınca kesilir.

PIP PRN:=X.ASM [p50] X.ASM LST cihazına kopya edilirken her satıra numara verilir. Her sekiz satırda boşluk ve 50 satırda sayfa boşluğu verilir.

PIP programında üç yeni parametre daha tanımlanmıştır. Bu parametreler uygun kütük isimlerini izleyen köşeli parantezler içinde gösterilerek yerine getirilir. Bu parametreler,

Gn n sayılı kullanıcı (user) bölgesinden kütüğü alır.

W R/O kütükleri konsoldan herhangi bir giriş olmaksızın yazılabilir hale sokar. Kütüğü araştırmadan R/W şekline getirir.

R Sistem kütüklerini okur.

G komutu bir kullanıcı bölgesinden diğerine kütükleri aktarmak için kullanılır. Örneğinkullanıcı USER 4'de çalışsın.

PIP X.Y=X.Y [G2]

[G2] komutu ile USER 2 deki X.Y kütüğünü alır, yine aynı isimli olarak USER 4 e yerleştirir.

PIP A:=A:*. *[G2] A disketi USER 2 üzerindeki bütün kütükleri yürürlükteki kullanıcı bölgesine aktar.

PIP programı başka türlü olarak da diğer kullanıcı bölgelerine kaydedilir.

USER Ø	Yürürlükteki kullanıcı bölgesi USER Ø dır. PIP.COM
DDT PIP.COM	
GØ	programı belleğe yükleniyor. CCP ye dönülüyor ve
USER 3	
SAVE S PIP.COM	USER 3 e geçilip PIP.COM kaydediliyor.

S burada programın boyutunu sayfa sayısı olarak göstermektedir. Program DDT tarafından yüklendikten sonra "NEXT" göstergesi programın bittiği adresi göstermektedir. Yani boş bölgenin başladığı adresi. Bu adres 1DØØH ise PIP.COM programı Ø1ØØH den 1CFF e kadar olan bölgeyi kapsıyor demektir. Dolayısıyla kaydedilecek sayfa $16+12 = 28$ sayfadır.

W komutu aşağıdaki gibi kullanılır.

PIP A:=B:*.COM [W] B disket sürücüsündeki COM kütüklerini A ya aktar ve hepsini R/W haline getir.

PIP A.DAT=B.DAT,F:NEW.DAT,G:OLD.DAT [W] B.DAT yı A.DAT ile değiştir. F ve G disketlerindeki kütükleri R/W şekline getir.

Sistem kütükleri R parametresi ile okunarak transfer edilebilirler. Aksi taktirde bu kütükler tanınmazlar.

PIP ED.COM=B:ED.COM [R]

komutu B diskinden ED.COM'u R/O sistem kütüğü olmasına rağmen okuyarak transfer eder.

6.5. ED UFN <cr>

ED programı ASCII kütüklerin CP/M içinde üretilmesini ve değiştirilmesini sağlar. Genel olarak ED programı kullanıcının ASCII kaynak kütükleri üretmesini, bunlar üzerinde işlem yapmasını sağlar. Karakterler numaralanmış satırları oluştururlar. Satır uzunluğunda bir sınırlama yoktur ve satır sonu başa dön tuşu ile belirlenir. ED programı metin bi-

çimleme sırasında karakter katarı arama, değiştirme veya araya girme gibi komutlara sahiptir. Bu özellikler CP/M altında program veya metin hazırlanmasında büyük kolaylık sağlar. CP/M sınırlı bir çalışma alanına sahipse de veriler bu alana kolaylıkla sayfa olarak alınabildiğinden çalışma alanı sınırsızdır.

Başlangıçta ED, belirlenen kaynak kütük daha önceden tanımlanmamışsa erişim için ufn isminde kütük açar. Ufn isminde kütük önceden var ise bu kütükten çalışma alanına veriler atanır ve üzerinde düzeltme işlemleri yapılır. A komutuyla atanan veriler üzerinde listeleme, değiştirme işlemleri yapıldıktan sonra çalışma alanı tekrar diskte W komutuyla saklanabilir.

6.6. SYSGEN <cr>

SYSGEN geçici komutu formatlanmış bir disketin ilk koşullandırılması sırasında CP/M işletim sisteminin meydana getirilmesini sağlar. SYSGEN programı kullanıcı ile iletişime geçerek verilmesi gereken komutları belirtir.

SYSGEN <cr>	SYSGEN programının başlatılması
SYSGEN VERSION m.m	SYSGEN uyarlama mesajı (m.m:2.2 olabilir)
SOURCE DRIVE NAME (OR RETURN TO SKIP)	Yeni diskette oluşturulacak CP/M işletim sisteminin A ve B gibi hangi disk sürücüsünden okunacağı sorulmaktadır. CP/M'in bellekte bir kopyası varsa bir <cr> tuşuna basılarak yüklemenin bellekten yapılması temin edilebilir.
	Yanıt verilmişse ekranda kaynak disket sürücüsünün ismini belirten bir yazı görülür.
SOURCE ON x THEN TYPE RETURN	Yine bir <cr> tuşuna basarak CP/M'in belleğe aktarılması temin edilir.

FUNCTION COMPLETE	CP/M işletim sisteminin belleğe aktarılması işleminin bittiğini belirtir.
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)	CP/M'in ilk olarak yazılacağı disket sürücü ismi belirtilir. Örneğin B yazılmışsa
DESTINATION ON B TYPE RETURN	Bildirisi ile <cr> tuşuna basılması istenir. Böylece işlem başlamıştır ekrana,
FUNCTION COMPLETE	İşlem bitti yazıldığında yeni disketin üzerinde CP/M işletim sistemi yüklenmiştir.

Birden fazla disketin koşullandırılacağı varsayılarak tekrar tekrar varış disket ismi sorulacaktır. Bu bildiri yazıldığında <cr> tuşu ile SYSGEN programından çıkılır.

CP/M sisteminin meydana getirilmesi başarılı olmuşsa yeni koşullandırılan disket CP/M'in sadece kalıcı komutlarını içerir. SYSGEN programı ile CP/M yüklenildiğinde "directory" boştur. Kullanıcı diğer geçici komutları bir diğer disketten kopya etmelidir. Hazır olan bir disketten tüm kütüklerin yeni diskete PIP komutu ile çekilmesi,

PIP B:= A:*. * [v] <cr>

komutunun B disketi yürürlükte iken yazılması ile olur. B diskinden PIP programı yüklenecek ve daha sonra ".*" belirttiği gibi tüm kütükler A' dan B' ye kopya edilecek ve doğruluğu test edilecektir. Kopya işlemi sırasında her kütüğün adı ekranda görülecektir. SYSGEN programı ile sistem izlerini "track" bir başka diskete aktarmadan önce CP/M işletim sistemine ek yapılabilir.

6.7. SUBMIT UFN parm #1parm #n <cr>

SUBMIT komutu CP/M komutlarının ardarda yürütülmesini sağlar. Komutta belirlenen ufn kütüğü o an yürürlükteki diskte ikinci ismi "SUB" olan

bir kütüktür. SUB kütüğü parametreleri değiştirilebilen CP/M komutlarından oluşur. CP/M komutlarında değişebilir parametreler yerine parm #1....parm #n gerçek parametreler yerleştirilir. Süreç sırasında hata olmasa bu komutlar ardarda yürütülür.

SUB kütüğü ED programı kullanılarak hazırlanır. Daha sonra gerçek parametrelerle değiştirilecek değişkenler,

\$1 \$2 \$n

şeklinde komutlara yerleştirilir. SUB kütüğü SUBMIT komutuyla işlendiğinde \$1 yerine parm #1, \$2 yerine parm #2 gerçek parametreleri konulur. Parametre sayısı veya şekli tutmazsa SUBMIT işlemi hata mesajı ile kesilecektir. SUBMIT işlemi belirlenen komutlardan oluşan

\$\$\$SUB

isminde o an yürürlükteki diskte bir kütük açar. Bilgisayar sistemi "REBOOT" edildiğinde CCP konsoldan çok bu komut kütüğünden giriş alır. SUBMIT işlevi A sürücüsündeki bir başka disket üzerine uygulanacak ise komutlar, A sürücüsü üzerine bir disket yerleştirilinceye kadar veya sistem yeniden başlatılincaya kadar işlenmez. Bundan başka kullanıcı yürütme işlemini, komutlar sırayla ekrana yansıtıldığında "rubout" tuşuna basarak bitirmeden durdurabilir. Bu durumda \$\$\$SUB kütüğü silinir. SUB kütüğündeki en son komut başka bir SUB kütüğünü çağırabilir, böylece içiçine girmiş SUB kütükleri koşturulabilir. ASMBL.SUB kütüğünün diskte bulunduğunu ve

ASM \$1

DIR \$1.*

ERA *.BAK

PIP\$2:=\$1.PRN

ERA\$1.PRN

komutlarını içerdiğini kabul edelim

```
SUBMIT ASMBL X PRN <cr>
```

komutu SUBMIT programının ASMBL.SUB kütüğünü okumasını sağlar. §1 ile karşılaşıldığında X, §2 ile karşılaşıldığında PRN gerçek parametreleri ile değiştirilirler. §§\$.SUB isminde

```
ASMX
```

```
DIRX.*
```

```
ERA*.BAK
```

```
PIP PRN:=X.PRN
```

```
ERA X.PRN
```

CCP tarafından ardarda yürütülecek komutlardan oluşan bir kütük meydana gelir. SUBMIT işlemi sadece A diskinden yürüyebildiği halde SUB kütüğü herhangi bir diske erişebilir.

6.8. XSUB İşlevi

CP/M uyarılama 2.0'da SUBMIT işlevine XSUB özelliği eklenmiştir. XSUB özelliği ile CCP'nin komutları yanında erişilen programlara satır girişleri sağlanarak SUBMIT yeteneği geliştirilmiştir. XSUB komutu SUBMIT kütüğünün en başına konularak CCP'nin altında işlem yapılmasını sağlar. Daha sonra gelen tüm SUBMIT komutları XSUB tarafından işlenerek CCP'den alınacak direktifler SUB kütüğünden alınır. SAVER SUB kütüğü aşağıdaki,

```
XSUB
```

```
DDT
```

```
I§1.HEX
```

```
R
```

```
GØ
```

```
SAVE 1.§2.COM
```

komutlarını içerdiğinde ve

```
SUBMIT SAVER X Y
```

komutu ile yürürlüğe konulduğunda §1 yerine X ve §2 yerine Y kütük isim-

leri konularak işlem yapılmaya başlanır. Önce XSUB programı arkasından DDT programı okunur. "IX.HEX" "R" ve "GØ" DDT Programının satır komutlarıdır. "GØ" komutu ile CCP seviyesine dönülmüş olur. Son komut "SAVE 1 Y.COM" CCP tarafından işlenir. XSUB programı bellekde kaldığı sürece (XSUB ACTIVE) mesajı ile bu belirtilir. Her yazılım ile tekrar başlama işleminde bu bildiri yenilenir.

6.9. DUMP UFN <cr>

DUMP programı ufn disk kütüğünün "hexadecimal" içeriğinin CRT ekranına dökülmesini sağlar. Kütük içeriği en solda mutlak adres olmak üzere her satırda onaltı bayt olmak üzere yazılır. Döküm esnasında "rubout" tuşuna basılarak bu programın yürütülmesi durdurulabilir. EKLER bölümünde DUMP programının bir örneği verilmiştir.

6.10. MOVCPM <cr>

MOVCPM programı, kullanıcıya CP/M işletim sistemini herhangi bir uzunktaki bellek yapısı için yeniden şekillendirme olanağını verir. İki zorunlu olmayan parametre program sonunda, yeni sistemin istenilen boyutta ve şekilde olmasını sağlar. Birinci parametre CP/M sisteminin bellek boyutunu, ikinci parametre ise yeni oluşturulan sistemin işlem sonundaki durumunu belirler. Eğer ilk parametre konulmazsa veya yerine bir "*" konulursa MOVCPM programı bellekde CP/M'in görüntüsünü yaratarak sistem meydana getirme işlemini sağlar. Geçerli komut şekilleri aşağıda verilmiştir.

MOVCPM <cr> 1ØØH adresinden başlayan RAM bellek alanı için CP/M sisteminin yeri değiştirilir ve program yürütülür. Sistemin değiştirilmesi bittiğinde yeni sistem yürütülecek fakat belleğe yazılmayacaktır. Meydana getirilen sistem Intel MDS 8ØØ için BIOS programını içerir.

MOVCPM n <cr> CP/M sisteminin yeri değiştirilerek n K bayt bellek için için yeniden oluşturulur.

MOVCPM ** <cr> 0 anki bellek yapısı için CP/M tekrar meydana getirilir ve SYSGEN işlemi için sistem bellekde bırakılır.

MOVCPM n * <cr> n K bayt bellek için CP/M işletim sistemi yeniden oluşturulur. SYSGEN işlemi için bellek de bırakılır.

MOVCPM** komutu bellekde CP/M'in yeni bir şeklini meydana getirir ve ekrana

READY FOR "SYSGEN" OR

"SAVE 32 CPMxx.COM"

bildirisi yazılarak SYSGEN işlemi için hazır olunduğu belirtilir.

Kullanıcı

SYSGEN <cr> SYSGEN komutu ile sistem oluşturma işlemine başlar.

SOURCE DRIVE NAME (OR RETURN TO SKIP) Bildirisi yazıldığında <cr> tuşuna basılarak daha önce MOVCPM komutu ile CP/M işletim sistemi yüklendiğinden bu basamak atlanmalıdır.

DISTINATION DRIVE NAME (OR RETURN TO REBOOT) Bildirisine B tuşuna basılarak cevap verilmezlidir.

DESTINATION ON B, THEN TYPE RETURN <cr> tuşuna basılarak işlem başlatılır.

SYSGEN işlemi daha önce anlatıldığı gibi sürer. SYSGEN işlem yerine

kullanıcı

SAVE 32 CPM xx.COM

komutunu verebilir. Bu MOVCPM işlemi sonunda standart olmayan çevre birimleri için CP/M'in BIOS ve BOOT bölümünün değiştirilmesi için CPM xx.COM isimli yeni bir kütük oluşturur.

MOVCPM programı 8800 serisi bilgisayarlarda bulunan seri haberleşme kapılarının "BAUDRATES" değerlerini tayin etmek için de kullanılır. Burada bu kapılar (ports) program kontrolü altında erişilebilen CP/M düzenleri gibi tasarlanmıştır. Bunun en önemli avantajlarından biri ADM-31 gibi harici terminalin bağlanmasına olanak vermesidir.

CP/M sistemi kullanıcı tarafından belirli bir seçim yapılmadığı durumlarda yazıcı kapısına 1200 baud, iletişim kapısına 300 baud, harici terminal kapısına 9600 baud değerlerini yerleştirir. Aşağıdaki şekilde kullanılabilir.

MOVCPM n.CAP*

Burada;

- n Desimal olarak CP/M işletim sistemi tarafından kullanılan bellek boyutu (Kbayt)
- .CAP 0-7 arasında "BAUDRATES" değerlerini belirtir.
- C "Communication" haberleşme kapısını,
- A "Auxiliary" harici terminal kapısını,
- P "Printer" yazıcı kapısını gösterir.
- * İşaretinin yeri satırın sonunda ise CP/M SYSGEN işlemi için hazırlanır. İhmal edilmişse CP/M hemen yürütülecek şekilde düzenlenir.

"CAP" DEĞERLERİ

"BAUDRATES"

0	ihmal edilmiş kapıya bağlıdır.
1	110 Baud
2	300 Baud
3	600 Baud
4	1200 Baud
5	2400 Baud
6	4800 Baud
7	9600 Baud

A>MOVCPM *?*<cr> : CP/M kullanılacak maksimum bellek için şekillendirilmiştir. "Baudrates" değerlerinin seçimi CP/M'e bırakılmıştır.

A>MOVCPM *.002 c <cr> : CP/M kullanılacak maksimum bellek için şekillendirilir. Yazıcı kapısı 300 Baud'a ayarlanır ve SYSGEN işlemi için düzenlenir.

A>MOVCPM *.402 : CP/M maksimum kullanılabilecek bellek için şekillendirilir. İletişim kapısı (port) 1200 Baud'a ve yazıcı kapısı 300 Baud'a ayarlanır.


```

A>type hul.sub
pip x.hex=uc1:
pip crt:=x.hex
pip uc1:=x.hex
submit hul

```

```

A>STAT VAL:

```

```

Temp R/O Disk: d:=R/O
Set Indicator: d:filename.typ $R/O $R/W $SYS $DIR
Disk Status   : DSK: d:DSK:
User Status   : USR:
Iobyte Assign:
CON: = OCC: ECC: BAT: AUX:
RDR: = COM: CRT: AUX: PRI:
PUN: = COM: CRT: AUX: PRO:
LST: = LPT: CRT: COM: AUX:
A>STAT DSK:

```

```

A: Drive Characteristics
2464: 128 Byte Record Capacity
308: Kilobyte Drive Capacity
64: 32 Byte Directory Entries
64: Checked Directory Entries
256: Records/ Extent
16: Records/ Block
32: Sectors/ Track
3: Reserved Tracks

```

```

B: Drive Characteristics
2464: 128 Byte Record Capacity
308: Kilobyte Drive Capacity
64: 32 Byte Directory Entries
64: Checked Directory Entries
256: Records/ Extent
16: Records/ Block
32: Sectors/ Track
3: Reserved Tracks

```

```

A>STAT DEV:
CON: is ECC:
RDR: is COM:
PUN: is COM:
LST: is LPT:

```

```

A>STAT LST:=CRT:

```

```

A>STAT USR:

```

```

Active User : 0
Active Files: 0

```



```

ASM RND.AAA
CP/M ASSEMBLER - VER 2.0
0323
002H USE FACTOR
END OF ASSEMBLY

```

A>LOAD RND

```

FIRST ADDRESS 0100
LAST ADDRESS 02E1
BYTES READ 01E2
RECORDS WRITTEN 04

```

A>DIR

```

A: MOVCPM COM : DISKDEF LIB
A: MONTYID LIB : Z80 LIB
A: DUMP COM : DUMP ASM
A: SUBMIT COM : PIP COM
A: XSUB COM : ED COM
A: ASM COM : LOAD COM
A: DDT COM : STAT COM
A: FORMAT COM : HDINIT COM
A: HDBACKUP COM : MACROASM BAS
A: COPY COM : SYSGEN COM
A: IPL COM : HELP COM
A: INSTALLK COM : TIME COM
A: USER COM : BASIC COM
A: BOOT COM : TIME ASM
A: RND PRN : RND ASM
A: HDCOPYI COM : HDSELECT COM
A: SILME COM : RND HEX
A: X HEX : RND COM
A: SETKEY COM : SIL COM

```

A>DIR RND.*

```

A: RND PRN : RND ASM
A: RND HEX : RND COM

```

A>STAT RND.* \$5

Size	Recs	Bytes	Ext	Acc
18	18	4k	1	R/W A:RND.ASM
4	4	2k	1	R/W A:RND.COM
11	11	2k	1	R/W A:RND.HEX
43	43	6k	1	R/W A:RND.PRN

Bytes Remaining On A: 52k

A>RND

```

KOMUTU BELIRTINIZ YAZ=Y OKU=0 CP/M E DONUS=C Y
VERIYI KAYDEDIN : CPM PROGRAMI
KOMUTU BELIRTINIZ YAZ=Y OKU=0 CP/M E DONUS=C 0
CPM PROGRAMI
KOMUTU BELIRTINIZ YAZ=Y OKU=0 CP/M E DONUS=C 1Y
VERIYI KAYDEDIN : HK
KOMUTU BELIRTINIZ YAZ=Y OKU=0 CP/M E DONUS=C 10
HK
KOMUTU BELIRTINIZ YAZ=Y OKU=0 CP/M E DONUS=C C

```


dir

```

A: MOVCPM      COM : DISKDEF  LIB
A: MONTYIO     LIB : Z80      LIB
A: DUMP        COM : DUMP     ASM
A: SUBMIT      COM : PIP      COM
A: XSUB        COM : ED       COM
A: ASM         COM : LOAD     COM
A: DDT         COM : STAT     COM
A: FORMAT      COM : HDINIT   COM
A: HDBACKUP    COM : MACROASM  BAS
A: COPY        COM : SYSGEN   COM
A: IPL         COM : HELP     COM
A: INSTALLK    COM : TIME     COM
A: USER       COM : BASIC    COM
A: BOOT        COM : TIME     ASM
A: RND         ASM : HDCOPYI   COM
A: HDSELECT    COM : SILME    COM
A: X           HEX : SETKEY   COM
A: SIL         COM
A>stat a:*. * $s

```

Size	Recs	Bytes	Ext	Acc
64	64	8k	1	R/W A:ASM.COM
190	190	24k	1	R/W A:BASIC.COM
1	1	2k	1	R/W A:BOOT.COM
90	90	12k	1	R/W A:COPY.COM
38	38	6k	1	R/W A:DDT.COM
49	49	8k	1	R/W A:DISKDEF.LIB
33	33	6k	1	R/W A:DUMP.ASM
4	4	2k	1	R/W A:DUMP.COM
52	52	8k	1	R/W A:ED.COM
18	18	4k	1	R/W A:FORMAT.COM
115	115	16k	1	R/W A:HDBACKUP.COM
64	64	8k	1	R/W A:HDCOPYI.COM
8	8	2k	1	R/W A:HDINIT.COM
232	232	30k	1	R/W A:HDSELECT.COM
4	4	2k	1	R/W A:HELP.COM
110	110	14k	1	R/W A:INSTALLK.COM
5	5	2k	1	R/W A:IPL.COM
14	14	2k	1	R/W A:LOAD.COM
26	26	4k	1	R/W A:MACROASM.BAS
17	17	4k	1	R/W A:MONTYIO.LIB
116	116	16k	1	R/W A:MOVCPM.COM
58	58	8k	1	R/W A:PIP.COM
18	18	4k	1	R/W A:RND.ASM
6	6	2k	1	R/W A:SETKEY.COM
2	2	2k	1	R/W A:SIL.COM
2	2	2k	1	R/W A:SILME.COM
42	42	6k	1	R/W A:STAT.COM
10	10	2k	1	R/W A:SUBMIT.COM
8	8	2k	1	R/W A:SYSGEN.COM
78	78	10k	1	R/W A:TIME.ASM
9	9	2k	1	R/W A:TIME.COM
7	7	2k	1	R/W A:USER.COM
1	1	2k	1	R/W A:X.HEX
6	6	2k	1	R/W A:XSUB.COM
48	48	6k	1	R/W A:Z80.LIB

Bytes Remaining On A: 62k

A>b:

B>dir

```

B: STAT      COM : PIP      COM
B: BASIC     COM : BRUN     COM
B: BCLOAD    : BASCOM     COM
B: BASLIB    REL : OBSLIB   REL
B: PDT       COM : PDT      BAS
B: FDT       COM : L80      COM
B: GRAFIK    BAS : GRAFIK   PRN
B: GRAFIK    REL : cpm
B: deneme    BAS : MACROASM BAS

```

B>a:

STAT *.* \$S

Size	Recs	Bytes	Ext	Acc
253	253	32k	1	R/W B: BASCOM.COM
190	190	24k	1	R/W B: BASIC.COM
195	195	26k	1	R/W B: BASLIB.REL
1	1	2k	1	R/W B: BCLOAD
121	121	16k	1	R/W B: BRUN.COM
303	303	38k	2	R/W B: FDT.COM
5	5	2k	1	R/W B: GRAFIK.BAS
34	34	6k	1	R/W B: GRAFIK.PRN
4	4	2k	1	R/W B: GRAFIK.REL
84	84	12k	1	R/W B: L80.COM
26	26	4k	1	R/W B: MACROASM.BAS
378	378	48k	2	R/W B: OBSLIB.REL
98	98	14k	1	R/W B: PDT.BAS
191	191	24k	1	R/W B: PDT.COM
58	58	8k	1	R/W B: PIP.COM
42	42	6k	1	R/W B: STAT.COM
15	15	2k	1	R/W B: cpm
1	1	2k	1	R/W B: deneme.BAS

Bytes Remaining On B: 90k

B>A:

A>STAT *.HEX

Recs	Bytes	Ext	Acc
1	2k	1	R/W A: CONSOL.HEX
1	2k	1	R/W A: MAKINA.HEX
2	2k	1	R/W A: MESAJ.HEX
10	2k	1	R/W A: REMOTE.HEX

Bytes Remaining On A: 48k

A>STAT *.COM

Recs	Bytes	Ext	Acc
2	2k	1	R/W A: ARTIR.COM
64	8k	1	R/W A: ASM.COM
2	2k	1	R/W A: BEYAZ.COM
1	2k	1	R/W A: BOOT.COM
1	2k	1	R/W A: CONSOL.COM
38	6k	1	R/W A: DDT.COM


```

A>DIR
A: XSUB      COM : MAC      COM
A: DEN       COM : DUMP     COM
A: SUBMIT    COM : PIP      COM
A: ED        COM : LOAD     COM
A: DDT       COM : STAT     COM
A: Z80       LIB : HELP     COM
A: INSTALLK COM : USER     COM
A: ASM       COM : CONSOL   ASM
A: SETKEY    COM : MESAJ    PRN
A: SIL       COM : MOVCPM   COM
A: CONSOL    PRN : CONSOL   HEX
A: CONSOL    COM : RND      BAK
A: MESAJ     ASM : REMOTE   ASM
A: RENK      COM : MESAJ    HEX
A: MESAJ     COM : HUL      ASM
A: HK        BAK : BEYAZ    COM
A: SARI      COM : MAKINA   ASM
A: HAT       COM : MAKINA   COM
A: ARTIR     COM : FDT      COM
A: KIR       COM : MAVI     COM
A: HUL       SUB : YESIL    COM
A: MAKINA    BAK : HK       SUB
A: REMOTE    PRN : RND      COM
A: consol    : MAKINA     PRN
A: MAKINA    HEX : REMOTE   HEX
A: CPM2CPM   ASM : BOOT     COM
A: PRO       COM : CPM2CPM  BAK
A: EK        COM : RND      ASM
A: NORMAL    COM : HARF     COM
A:           : REMOTE     COM

```

A>INSTALLK

Monroe Function Key Installation Program
V1.05 7/9/82 WGW

Initial key setup from file

Ready to program the function keys
Enter ? for help

Function key: F1
The current value is: dir^M
Enter the new value : dir^M

*** No change made to the length ***

Function key: F2
The current value is: NORMAL^M
Enter the new value : NORMAL^M

*** No change made to the length ***

Function key: F3

The current value is: stat a:

Enter the new value : stat a:

*** No change made to the length ***

Function key: F4

The current value is: mavi

Enter the new value : mavi

*** No change made to the length ***

Function key: home

The current value is: sil^M

Enter the new value : sil^M

*** No change made to the length ***

Function key:

Saving program in file

Programming the function keys.

--- Returning to CP/M ---

A>TYPE CONSOL.PRN

0100		ORG	100H
	LABEL:		
0100 0E01		MVI	C,1
0102 CD0500		CALL	0005H
0105 FE30		CPI	'0'
0107 C20001		JNZ	LABEL
010A C9		RET	

DUMP CONSOL. ASM

```

0000 09 4F 52 47 09 31 30 30 48 0D 0A 4C 41 42 45 4C
0010 3A 0D 0A 09 4D 56 49 09 43 2C 31 0D 0A 09 43 41
0020 4C 4C 09 30 30 30 35 48 0D 0A 09 43 50 49 09 27
0030 30 27 0D 0A 09 4A 4E 5A 09 4C 41 42 45 4C 0D 0A
0040 09 52 45 54 0D 0A 1A 1A 1A 1A 1A 1A 1A 1A 1A
0050 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A
0060 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A
0070 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A

```

A>DUMP CONSOL. HEX

```

0000 3A 30 42 30 31 30 30 30 30 45 30 31 43 44 30
0010 35 30 30 46 45 33 30 43 32 30 30 30 31 43 39 35
0020 39 0D 0A 3A 30 30 30 30 30 30 30 30 30 30 0D 0A
0030 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A
0040 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A
0050 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A
0060 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A
0070 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A

```

A>DUMP CONSOL. PRN

```

0000 0D 0A 0D 0A 20 30 31 30 30 20 20 20 20 20 20 20
0010 20 20 20 20 09 4F 52 47 09 31 30 30 48 0D 0A 20
0020 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 4C
0030 41 42 45 4C 3A 0D 0A 20 30 31 30 30 20 30 45 30
0040 31 20 20 20 20 20 20 09 4D 56 49 09 43 2C 31 0D
0050 0A 20 30 31 30 32 20 43 44 30 35 30 30 20 20 20
0060 20 09 43 41 4C 4C 09 30 30 30 35 48 0D 0A 20 30
0070 31 30 35 20 46 45 33 30 20 20 20 20 20 20 09 43
0080 50 49 09 27 30 27 0D 0A 20 30 31 30 37 20 43 32
0090 30 30 30 31 20 20 20 20 09 4A 4E 5A 09 4C 41 42
00A0 45 4C 0D 0A 20 30 31 30 41 20 43 39 20 20 20 20
00B0 20 20 20 20 09 52 45 54 0D 0A 1A 1A 1A 1A 1A 1A
00C0 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A
00D0 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A
00E0 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A
00F0 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A 1A

```

A>DUMP CONSOL. COM

```

0000 0E 01 CD 05 00 FE 30 C2 00 01 C9 00 00 00 00 00
0010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```

A>STAT CONSOL.*S\$

Recs	Bytes	Ext	Acc
1	2k	1	R/W A:CONSOL. ASM
1	2k	1	R/W A:CONSOL. COM
1	2k	1	R/W A:CONSOL. HEX
2	2k	1	R/W A:CONSOL. PRN

Bytes Remaining On A: 94k

1. GİRİŞ

ED programı CP/M için biçimleyici bir programdır ve CP/M kaynak kütüklerinin üretimi ve değiştirilmesi için kullanılır. ED aşağıdaki komutlarla başlatılır.

ED kütükismi

ED kütükismi . tipi

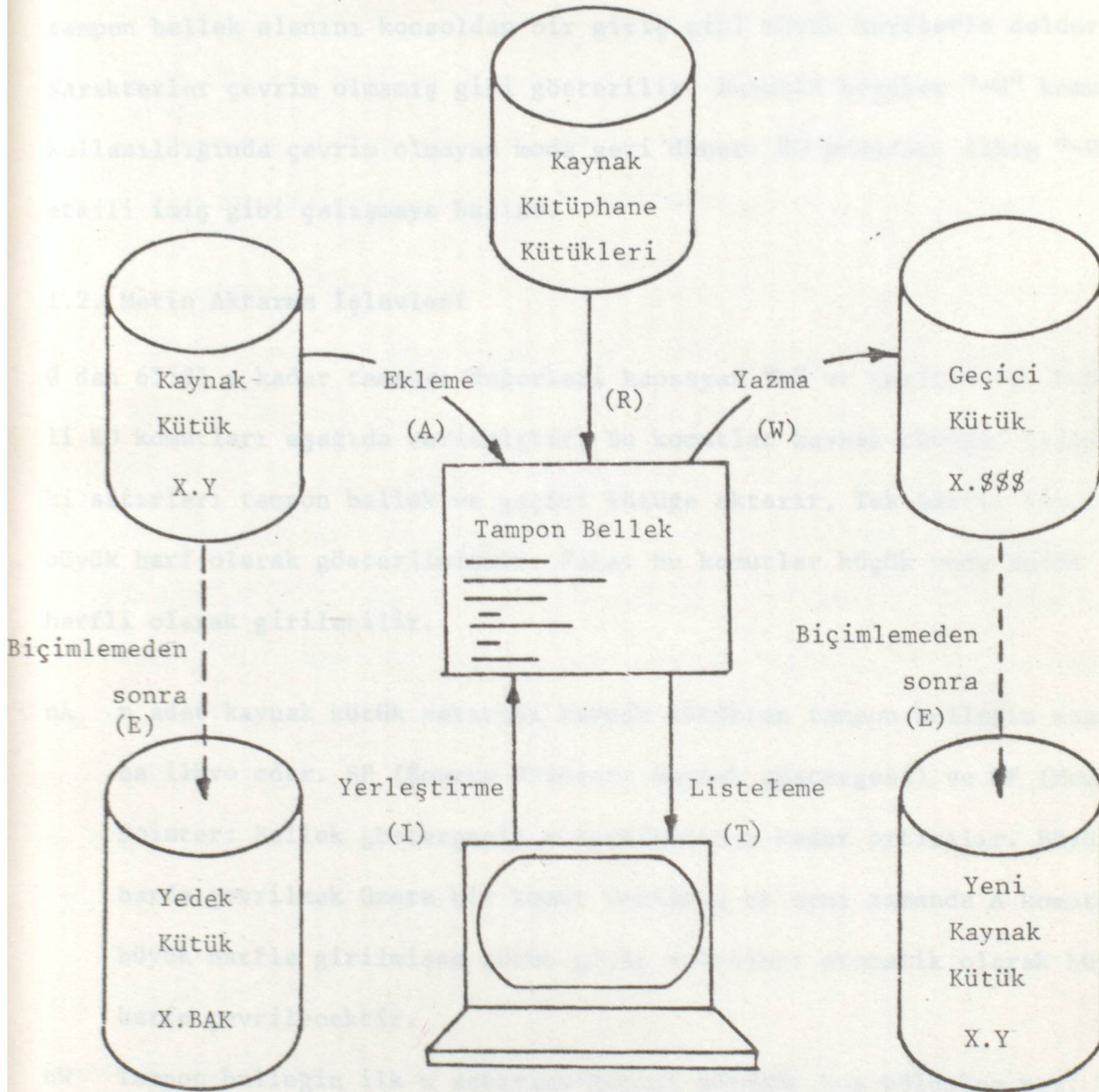
Genel olarak ED programı kütük ismi veya kütük ismi . tipi verilen kaynak kütüğün dilimlerini ana belleğin içine okur. Burada kullanıcı tarafından bir takım komutlar kullanılarak gerekli değişiklikler yapıldıktan sonra disket üzerine yeniden yazılır. Kaynak kütük biçimlemeden önce mevcut değilse yeniden oluşturulur. ED de ilgili bütün işlemler aşağıda verilmiştir.

1.1. ED İşlemi

ED programı yukarda belirtilen kaynak kütük üzerinde çalıştırılır. Ve bütün metinler bir tampon bellek içine yerleştirilir. Burada metinler görüntülenir veya değiştirilebilir. Tampon bellekde muhafaza edilen satırların sayısı ve satır uzunlukları çeşitlidir. Biçimlenecek metinler işleticinin komutları altında geçici çalışma kütüğüne yazılırlar. Biçimlenmenin sonunda tampon bellek geçici kütüğü yazılır, asıl kütüğün ismi (x.y) x.BAK olarak değiştirilir. Biçimlenmiş kütüğün sonucunda geçici kütük (x.\$\$\$) x.y olarak değiştirilir. Asıl kütüğün ismi x.BAK olarak değiştirildiğinden biçimlendirilmemiş kaynak kütük zorunlu olduğunda geri çağrılabilir.

Tampon bellek mantıksal olarak kaynak kütük ile çalışma kütüğü arasındadır.

ED İŞLEMİ



ED programı küçük ve büyük harfli ASCII karakterlerin ikisini birden kabul eder. Tek harfli komutlar durumlardan birini göstermek için kullanılır. U komutu kullanılarak küçük harfler büyük harflere çevrilir. Bu tampon bellek alanını konsoldan bir giriş gibi büyük harflerle doldurur. Karakterler çevrim olmamış gibi gösterilir. Bununla beraber "-U" komutu kullanıldığında çevrim olmayan moda geri döner. ED programı ilkin "-U" etkili imiş gibi çalışmaya başlar.

1.2. Metin Aktarma İşlevleri

Ø dan 65535 e kadar tamsayı değerleri kapsayan "n" ve çeşitli tek harfli ED komutları aşağıda verilmiştir. Bu komutlar kaynak kütüğün içindeki satırları tampon bellek ve geçici kütüğe aktarır. Tek harfli komutlar büyük harf olarak gösterilmiştir. Fakat bu komutlar küçük yada büyük harfli olarak girilebilir.

nA n adet kaynak kütük satırını kaynak kütükten tampon belleğin sonuna ilâve eder. SP (Source Pointer: Kaynak göstergesi) ve MP (Memory Pointer: Bellek göstergesi) n tarafından n kadar artırılır. Büyük harfe çevrilmek üzere bir komut verilmiş ve aynı zamanda A komutu büyük harfle girilmişse bütün giriş satırları otomatik olarak büyük harfe çevrilecektir.

nW Tampon belleğin ilk n satırını geçici kütüğün boş bölümüne yaz. Ve kalan satırları MP nin (n+1) konumundan itibaren tampon belleğin üstüne aktar (kaydır). TP (Temporary Pointer) geçici kütük göstergesi ni n kadar artır.

E Biçimleme işleminin sonu. Bütün tamponlanmış metini (text) ve işlenmemiş bütün kaynak satırlarını geçici kütüğe kopya et. Kütüklerin isimlerini önceden belirtildiği gibi değiştir. $x.y = x.BAK$ ve $x. $$$ = x.y$ olur.

- H E komutunun otomatik olarak gerçekleştirilmesiyle yeni kütüğün başına hareket edilir. Geçici kütük yeni kaynak kütük olur. Tampon bellek boşaltılır ve yeni bir geçici kütük üretilir.
- O Asıl kütüğe dönülür. Tampon bellek boşaltılır. Geçici kütük silinir ve SP (Source Pointer) kaynak kütüğün 1 nolu pozisyonuna döner. Biçimleme komutlarından önceki kütük değişmemiştir.
- a Biçimlemeden hiçbir değişiklik yapılmadan CP/M e geri dönülür.

Belirli durumların göz önünde tutulması gereken bir numarası vardır. Herhangi bir ED komutunda eğer tamsayı "n" unutulmuşsa burada n=1 olarak kabul edilir. Böylece A ve W komutları bir satır ilâve ederler veya bir satırıyazarlar. Bir "#" işareti bu komutların yanında verilmişse "n" in en büyük değeri olan 65535 değeri kabul edilmiş olur. #A komutu biçimleme için kaynak kütüğünün belleğe okunması için kullanılır. Benzer olarak #W komutu giriş tampon belleğini geçici kütüğe yazar. A ve W komutlarının iki özel şekli bir kolaylık sağlar. ØA komutu geçerli tempon belleğin en azından yarısını doldurur. ØW komutu tamponun yarısı boş olunca-ya kadar satırları yazar. Tampon belleğin boyutu aşıldığında bir hata oluştuğu ekranda belirtilir.

1.3. Tampon Belleğin Organizasyonu

Tampon bellek A komutu ile kaynak kütükten kaynak satırları getirmek üzere dikkate alınır. Tampon bellek kullanıcının komutları altında tampon belleği baştan başa hareket eden bir yardımcı CP (Character Pointer) karakter göstergesine sahiptir. Tampon bellek, mantıksal olarak aşağıdaki gibi oluşmuştur.

Tampon Bellek

ilk satır ----- <cr><lf>

----- <cr><lf>

Ve kütük ile birlikte olarak kaydedilmez. Kullanıcı bir mutlak satır numarasını geçerli satırdan ileriye veya geriye alarak mutlak satır numarası önüne ":" karakteri koyarak referans alabilir. Şöyleki,

:400T

komutu "geçerli satırdan mutlak satır numarası 400 olan satıra kadar listele" şeklinde yorumlanır. İki satır arasındaki kayıtlar aşağıdaki şekilde kullanılır. Bu komut,

345::400T

ise "mutlak satır 345 den mutlak satır 400 a doğru bütün satırları listele" şeklindedir. Mutlak satır numaraları bu kısalıkta herhangi bir ED komutunun önüne gelebilir. Satır numaralama, V "Verify" komutu ile kontrol edilir. Satır numaralama işlemi "-V" komutu kullanılarak geçersiz hale gelir. Eğer kütük biçimleme yeni oluşuyorsa ED programı,

NEW FILE

mesajını verir. Kullanıcı "i" komutunu kullanarak metni tampon bellek içine <cr> ile sonlandırarak yerleştirir. Bir tek <ctl-z> işleminin uygulanması ile ED komut moduna dönlür.

1.5. Tampon Bellek İşlemi

ED programı başladığında tampon bellek boştur. Kullanıcı ilâve edilecek satırları, kaynak kütükten veya direkt olarak konsoldan "I:Insert" komutuyla girer.

ED programı herhangi bir sayıda giriş satırını ctl-z belirtilinceye kadar kabul eder. Her satır <cr> ile sonlandırılmıştır. (<lf=line feed> otomatik olarak uygulanır).

Genel olarak ED programı komutları büyük veya küçük harf olarak kabul eder. Eğer komut büyük harfli ise bütün giriş değerleri büyük harfe çev-

rilir. "I" komutu kullanılıyorsa bütün girişler otomatik olarak büyük harfe çevrilir. "i" nin küçük harfle kullanılması durumunda girilen değerler aynen geçerlidir. Büyük girilirse büyük, küçük harf girilirse küçük harfli olarak kabul edilir.

Kaynak metnin içeriği çeşitli komutlar ile basılabilir. Aşağıda bu komutlar gösterilmiştir. Burada "n" pozitif veya negatif olan bir tam sayıdır. "n" negatif bir sayı ise "-" işareti basılarak belirtilir. "#" karakteri 65535 satırı temsil eder. n=1 ise gösterilmesi zorunlu değildir.

$\pm B$ (Buffer) B CP yi tampon belleğin başına al.

-B CP yi tampon belleğin sonuna götür.

```

+nC(Character) CP (character pointer: karakter göstergesi) yi +n karakter
geçerli satırda ileriye veya geriye hareket ettir.
(-n CP yi n karakter geriye götürür.)

```

±nD (Delete) CP nin önünde veya arkasındaki n karakteri sil. (-nD
CP nin arkasındaki n karakteri sil.)

±nK (Kill) CP yi referans alarak kaynak metinde ±n satır ortadan kaldırdır. Bu komut uygulandığında CP geçerli satırın başında değilse, komut nK şeklinde belirtilmişse CP den önceki -nK şeklinde belirtilmişse CP den sonraki karakterler silinerek ortadan kalkar.

$\pm nL$ (Line) $n=\emptyset$ ise CP yi geçerli satırın başına getir.

$n\neq\emptyset$ ise CP yi geçerli satırın başına getir ve nL şeklin-
de verilmişse CP yi n satır aşağıda, $-nL$ şeklinde ise
CP yi n satır yukarıya götür. CP tampon belleğin üstün-
de veya altında duracak olursa bu n için büyük bir de-
ğer belirtildiğini gösterir.

$\pm nT$ (Type)	$n=0$ ise geçerli satırın içeriğini CP ye kadar göster. $n=1$ ise geçerli satırın içeriğini CP den satırın sonuna kadar ver. $n>1$ ise (nT şeklinde belirtilmişse) $(n-1)$. satıra kadar olan satırları göster. $n>1$ ise ($-nT$ şeklinde belirtilmişse) n satır yukarıya CP ye kadar olan satırları göster. Herhangi bir tuşa basıldığında bu işlem bitirilmeden durdurulur.
$\pm n$	$\pm nLT$ komutunun aynısıdır. CP yi yukarıya veya aşağıya hareket ettirerek CP yi içeren satır gösterilir.

1.6. Yazısal Komutlar (Command Strings)

Komutlar konsol tampon belleğinin kapasitesi kadar girilebilirler ve bunlar $\langle cr \rangle$ den sonra yürütülürler. Kullanıcı aşağıdaki komutlarla CP/M satır düzeltme işlemlerini yapabilir.

Ctl-C	Satırın başlangıcında uygulandığında CP/M sistemi yeniden başlatılır.
Ctl-E	Satırın fiziksel sonunu belirler.
Ctl-H	Bir karakter geriye gelme ve karakterin silinmesi işlevlerini yerine getirir.
Ctl-J	Geçerli satırın sonlandırılması (line feed: satır atlatma) işlevini belirtir.
Ctl-M	Geçerli satırın sonlandırılması (cr: geri dön) işlevini belirtir.
Ctl-R	Geçerli komut satırının silinmesini sağlar.
Ctl-U	Konsoldan giriş satırının silinmesini sağlar.
Ctl-X	Ctl-U ile aynı işlevi sağlar.
Ctl-Z	Konsoldan girişin sonunu belirtir. (PIP ve ED programlarında kullanılır.)
rub/del	Konsoldan girilen son karakterin silinmesini sağlar.

Tampon belleğin içeriği bir önceki bölümde belirtilmişti. CP genel olarak son satırı izleyen son karakterin sonundadır. Yazısal komutlar ve bu komutların etkisi aşağıda açıklanmıştır. Küçük harfli komutların kullanımından yazısalların büyük harflere otomatik olarak çevrilmesi için kaçınılmıştır.

Örnekler

1. B2T CP yi tampon belleğin başına al ve baştan iki satırı göster.
2. 5CØT CP yi 5 karakter (satırın başından) ileriye al ve satırın başından CP ye kadar olan karakterleri göster.
3. 2L-T CP yi iki satır aşağıya götür ve bir önceki satırı göster.
4. -L#K CP yi bir satır yukarıya al ve 65535 satırı sil.
5. -2L#T CP yi iki satır yukarıya al ve bütün tampon bellek satırlarını göster.
6. <cr> CP yi bir satır aşağıya al ve geçerli satırı göster.

1.7. Metin İçindeki Yazısalların Araştırılması ve Değiştirilmesi (Text Search and Alteration)

ED programı, yazısalları (strings) tampon bellek içine yerleştiren bir komuta sahiptir. s karşılaştırılacak yazısalı gösterir. Bunu <cr> veya ctl-Z (↑Z) komutu izler.

nFs <cr>

nFs ↑Z

ED CP nin geçerli pozisyonundan başlar ve yazısalı karşılaştırmaya başlar. Yazısal (string) n kere karşılaştırılır. (Aynı yazısalın n.sinin bulunması istenmektedir.) CP direkt olarak yazısaldan sonra yerleşir.

Eğer aranan yazısal bulunamıyorsa "CP" ilk pozisyonuna geri döner. Aranan yazısala <cr>ve <If> sembol çiftini sapsayan ctl-L ilave edilebilir. Aşağıdaki komutlar "F" komutunun kullanımını göstermektedir.

B#T <cr> CP yi tampon belleğin başına al ve 65535 satırı göster.
 FST <cr> "ST" yazısalını araştır.
 FI+ZØTT CP den sonraki "I" yazısalını araştır ve "I" dan sonra satırın sonuna kadar olan karakterleri göster.

"Insert" komutunun kısaltılmış şekli, "F" komutu ile birlikte metine ait basit değişiklikler yapılmasına izin verir. Bu komutun şekli aşağıda verilmiştir.

Is +Z

Is <cr>

Burada s yerleştirilecek yazısalı göstermektedir. Yerleştirilecek yazısal bir "+Z" ile veya "<cr>" ile sonlandırılır. Bu "<cr>" <cr><If> gibi etkilidir.

F ve I komutlarının kullanım şekline ait örnekler aşağıda verilmiştir.

1. BITHIS IS +Z <cr> THIS IS yazısalını metnin başına yerleştir.
2. F TIME +Z -4DI PLACE +Z <cr> TIME yazısalını bulun, bunu silin ve PLACE yazısalını yerleştirin.
3. 3FO +Z -3D5DI CHANGES +Z<cr> 3. kere oluşan "O" harfini bul ve CP nin 3 karakter gerisini s karakter ilerisini sil ve "CHANGES" yazısalını yerleştir.
4. -8CI SOURCE <cr> CP yi 8 karakter geriye al ve "SOURCE" yazısalını yerleştir. Burada <cr>, <cr><If> yerine geçer. Yani <cr> ile sonlandırıldığında "SOURCE" yazısalından sonra satır atlatılmaktadır.

ED programında F ve I komutlarının yerine geçen ve bunların yaptığı işleri gören bir tek komut vardır. Bu komutun kullanımı aşağıda görülmektedir.

$nS \ S_1 \ \uparrow Z \ S_2 \ <cr>$ veya $nS \ S_1 \ \uparrow Z \ S_2 \ \uparrow Z$

Bu komutla aşağıdaki komutlar aynı etkiyi gösterirler.

$F \ S_1 \ \uparrow Z - KDI \ S_2 \ <cr>$ veya

$F \ S_1 \ \uparrow Z - KDI \ S_2 \ \uparrow Z$

Burada K yazısının uzunluğunu göstermektedir. Ve S_1 yazısı yerine S_2 yazısı yerleştirilmektedir.

F komutuna benzeyen, otomatik olarak satırların ilave edildiği bir araştırma işlevi gibi görev yapan bir komut da vardır. Bu komutun kullanılış şekli aşağıdadır.

$nNS \ <cr>$

$nNS \ \uparrow Z$

Bu komut yazısallarının n. oluşumu için giriş kaynak kütüğünü araştırır. N komutu yürütülürken tampon bellekde yazısal bulunamıyorsa giriş belleğinin içeriği yazılabilir. (Örneğin #W basılır) Giriş satırları tampon belleğin en azından yarısı dolana kadar veya giriş kaynak kütüğü bitene kadar okunur. Bu durumda araştırma işlemi yazısal n kere bulunana kadar veya kaynak kütük, geçici kütüğe tamamen aktarılanaya kadar devam eder.

Satır biçimleme işlevinin bir başkası, yanyana koyma (juxtaposition) diye isimlendirilen komuttur. Bu komut aşağıdaki şekillerde uygulanır.

$nJS_1 \ \uparrow ZS_2 \ \uparrow ZS_3 \ <cr>$ veya

$nJS_1 \ \uparrow ZS_2 \ \uparrow ZS_3 \ \uparrow Z$

S_1 yazısının, karakter göstergesinin "CP" geçerli pozisyonundan itibaren n. oluşumu araştırılır, bulunursa S_2 yazısı yerleştirilir. Bu durumda CP S_2 yazısından sonra yerini alır. Daha sonra S_3 kadar olan bütün karakterler silinir. (CP den S_3 kadar olan) CP (Karakter göstergesi) direkt olarak S_2 den sonra bırakılır. S_3 yazısı bulunmuyorsa silme işlemi yapılmaz. Geçerli satır

NOW IS THE TIME <cr><If>

şeklinde ve komut

JW+Z WHAT +Z+L <cr>

şeklinde ise bu işlemlerin sonucu

NOW WHAT <cr><If> olur.

ED programı tarafından F,S,N,J komutlarında kullanılan karakterler, 100 sayısı ile sınırlandırılmıştır.

1.8. Kaynak Kütüphane Kütükleri (Source Libraries)

ED programı R komutuyla kaynak kütüklerinin biçimleme işlemi esnasında dahil edilmesine izin verir. Bu komut

R kütükismi +Z veya

R kütükismi <cr> şeklinde kullanılır.

"Kütük ismi" disket üzerinde kütük tipi "LIB" olarak belirtilmiş kütüklerin ilk ismini göstermektedir. ED programı belirtilmiş kütükleri okur ve CP den sonra aynı I komutu gibi tampon bellek içine karakterleri yerleştirir. Komut

RMACRO <cr>

şeklinde ise ED programı MACRO.LIB kütüğünden kütüğün sonuna kadar karakterleri okur ve karakterleri tampon bellek içine yerleştirir. ED programı X (Xfer) komutuyla "blockmove" olanağı verir. Yalnız biçimleme işlemi esnasında aktif olan X\$\$\$\$\$.LIB olarak isimlendirilen geçici kü-

tüğe n den sonra gelen geçerli satırların aktarılmasına sağlayan komutun şekli nX şeklindedir.

Genel olarak kullanıcı geçerli satırları kaynak kütüğün herhangi bir parçasına veya aktarılacak satırları geçici kütüğe depo eder. Aktarılan satırlar bu kütük içinde birinden diğerine R komutu (bu kütüphane kütüğün okunmasına neden olan komuttur) kullanılarak depo edilebilir. R komutu aktarılmış "LIB" kütüğünü boşaltamaz. X komutu ile aktarılmış satırların bir bölümü kaynak kütüğe tekrar okunabilir. Bu komut "ØX" şeklindedir. ED programı Q ve E komutlarını kullanarak aktarma işlemini bitirir. Ve geçici "LIB" kütüğünü ortadan kaldırır.

1.9. Komutun Ardışıl Olarak Yürütülmesi (Repetitive Command Execution)

M "macro" komutu kullanıcının ED komutlarını ardışıl olarak yürütmesine izin verir. Bu komutun kullanımı

nM CS <cr> veya

nM CS ↑Z

şeklindedir. CS ED komutlarını göstermektedir. n>1 ise ED programı, yazısal komutu n kere yürütür. n=Ø veya 1 ise yazısal komut ardışıl olarak bir hata durumu ile karşılaşılana kadar yürütülür. Örnek olarak "GAMMA" ve "DELTA" yazısalları aşağıdaki şekillerde gösterildiği gibi ardışıl olarak değiştirilir.

MF GAMMA ↑Z-5DI DELTA ↑Z ØTT <cr> veya

MS GAMMA ↑Z DELTA ↑Z ØTT <cr>

2. ED PROGRAMI HATA KOŞULLARI (ED ERROR CONDITIONS)

ED programı içersinde bir hata oluştuğunda ekranda "BREAK X AT C" mesajı

görülür. Burada X aşağıda verilen hata göstergelerini belirtmektedir.

- ? Tanımlanmamış Komut
- > Tampon bellek dolu (D,K,N,S veya W komutlarından biri kullanılmış); F,N veya S yazısalları oldukça uzun.
- # Belirtilen komutun n. oluşumu bulunamadı (Örneğin F komutu içindeki n. yazısal)
- 0 R komutunda belirtilen "LIB" kütüğü açılmıyor.

Bir disk hatası oluştuğunda "d" disket sürücü ismi olmak üzere CP/M, CRT ekranından

BDOS ERR on d:BAD SECTOR

mesajını iletir. Kullanıcının bu mesajdan sonra iki seçeneği vardır. İlki <cr> tuşuna basarak konsola (bu durumda tampon bellek içindeki veri düzeltilmek üzere yeniden gözden geçirilir), diğeri ise ctl-C tuşuna basarak sisteme dönülmesidir. Disketdeki daha önceki X.BAK kütüğü gözden geçirilir. Bu kütük sağlam ise biçimleme işleminde üretilmiş olan X.Y kütüğü silinir. Ve X.BAK kütüğü ile X.Y kütüklerinin isimleri değiştirilir. Daha sonra bu kütüğe yeniden biçimleme işlemi uygulanır.

FILE IS READ/ONLY

Mesajı görüldüğünde kütüğün yalnız okunabileceği akla gelmelidir. Bu durumda kütük yüklenir ve gözden geçirilir. STAT programı kullanılarak kütüğün R/W olması sağlanır.

'SYSTEM' FILE NOT ACCESSIBLE

Bu mesaj ele alınan kütüğün sistem kütüğü olduğu içine girilemeyeceğini belirtir. Bu kütüğün de STAT programı kullanılarak disket üzerinde normal bir şekilde yer alması sağlanır.

3. ED KOMUTLARININ ÖZETİ

ED programında E (End), H (Head), O (Original), Q (Quit) komutları tek bir harfle kullanılırlar. I, J, M, N, R ve S komutları i, j, m, n, r ve s şeklinde büyük yada küçük harflerle girilebilirler. Bir komut büyük harfle kullanıldığı zaman ED programı otomatik olarak bütün girilen yazısalları büyük harfe dönüştürür. Aşağıda ED programı komutlarının bir özeti verilmiştir.

nj	Yazısalları yanyana koyma (yerleştirme)
±nK	Satırların silinmesi
±nL	CP nin aşağı veya yukarı hareket ettirilmesi
nM	"Macro" tanımlama ardışıl yürütme işlevi
nN	Otomatik tarama ile gelecek oluşumun bulunması
O	Asıl (Original) kütüğe dönülmesi
Q	Kütükden hiçbir değişiklik yapılmadan çıkılması
R	Kütüphane kütüğün okunması
nS	Yazısalın yerine başka bir yazısal yerleştirilmesi
±nT	Satırların gösterilmesi
±U	"U" ise küçük harfleri büyük harflerle değiştirir. "-U" ise değiştirmez.
±V	Satırları numarala veyahut kalan karakter boşluğunu göster.
ØV	Bu V komutunun özel bir durumudur. ØV komutu tampon belleğin istatiksel durumunu "FREE/TOTAL" şeklinde gösterir. Burada "FREE" desimal olarak tampon bellekte kalan bayt sayısını, "TOTAL" ise tampon belleğin boyutunu vermektedir.
nW	Satırların tampon bellekten geçici kütüğe aktarılması.
nZ	Yaklaşık n saniye bekle.
±n	CP yi aşağıya veya yukarıya hareket ettir. Geçerli satırı göster.

1. GİRİŞ

CP/M çevirici disketteki çevirilmiş kaynak kodları okur ve tutulan hexformat'ında ROM makine diline çevirir. CP/M çevirici aygıtındaki komutlar ile hareket ettirilebilir.

ASM kodlarını veya

ASM kodlarını . "param"

Bu komutların her ikisinde de ASM programı, disk diskinde ve ismi "ASM" kodlarını ASM" olan ROM çevirici dili kaynak kodlarını içeren bir dosya iletilir.

Birinci ve ikinci şekilde farklılık (farklılık) parametre değeri ile verilmektedir. Parametre çevirilecek ayar (ayarlar) normal kodların belirlenmesinde kullanılır. Bu iki şekilde de CP/M çevirici programı

ve sonunda şu satırlar:

ÜÇÜNCÜ BÖLÜM

CP/M ASSEMBLER VER 2.0

CP/M ÇEVİRİCİ "ASSEMBLER"

Bu dosya 2.0.0 versiyonunda yazılmıştır. Bu dosya 2.0.0 versiyonunda kullanılmak üzere yazılmıştır. Bu dosya 2.0.0 versiyonunda kullanılmak üzere yazılmıştır.

Kodları ASM

Kodları ASM

Bu dosya 2.0.0 versiyonunda yazılmıştır. Bu dosya 2.0.0 versiyonunda kullanılmak üzere yazılmıştır. Bu dosya 2.0.0 versiyonunda kullanılmak üzere yazılmıştır.

Bu dosya 2.0.0 versiyonunda yazılmıştır. Bu dosya 2.0.0 versiyonunda kullanılmak üzere yazılmıştır. Bu dosya 2.0.0 versiyonunda kullanılmak üzere yazılmıştır.

1. GİRİŞ

CP/M çevirici disketteki çevrilecek kaynak kütükleri okur ve Intelin hexformatı'nda 8080 makina diline çevirir. CP/M çevirici aşağıdaki komutlar ile harekete geçirilebilir.

ASM kütükismi veya

ASM kütükismi . "parms"

Bu komutların her ikisinde de ASM programı, disk üzerinde ve ismi "kütükismi.ASM" olan 8080 çevirici dili kaynak kütüğünü içeren bir kütükle ilgilenir.

Birinci ve ikinci şekildeki farklılık ikincisinde parametre değerlerinin verilmesidir. Parametre çevrilecek aynı isimdeki kaynak kütüğünün belirlenmesinde kullanılır. Bu iki şekilde de CP/M çevirici yüklenir ve ekranda şu satırlar görülür.

CP/M ASSEMBLER VER n.n

Burada n.n geçerli uyarlama numarasını göstermektedir. Birinci komutun kullanılmasıyla çevirici kaynak kütüğü okur ve iki çıkış kütüğü üretir. Bunlar

kütükismi . HEX

Kütükismi . PRN

kütükleridir. HEX kütüğü Intel'in hex formatında esas programın makina kodlarını içerir. PRN kütüğü çeşitli notlarla açıklanmış listesi verilmiş makina kodlarını hata mesajlarını ve kaynak satırları içerir. Çevrim esnasında hata oluştuğunda bu hatalar PRN kütüğü içinde CRT ekranında görülür.

ASM parametreleri giriş ve çıkış kütüklerinin düzenlenmesi amacıyla kullanılır. Bu sebeple "parms" kısmı kaynak kütüğü ve çıkış kütüklerini

belirleyen üç harfli bir komuttur. Bu

kütükismi.P1P2P3

şeklindedir.

P1: A,B,....P Kaynak kütüğü içeren disk ismini belirtir.

P2: A,B,....P Elde edilecek HEX kütüğünün disk ismini belirtir.

Z HEX kütüğü üretimini durdurur.

P3: A,B,....P Elde edilecek PRN kütüğünün disk ismini gösterir.

X PRN kütüğünü CRT ekranına aktarmak için kullanılır.

Z PRN kütüğünün üretimini durdurur.

Bu parametrelerin kullanımıyla ilgili olarak çeşitli örnekler aşağıda verilmiştir.

1. A>ASM X.AAA komutu disk üzerindeki kaynak kütüğü X.ASM olan ve yine A isimli disket üzerinde X.HEX ve X.PRN şeklinde çıkış kütüklerinin üretileceğini göstermektedir. Bu komut ASM X komutu ile aynıdır. ASM X.AAA komutu A disketini ilgilendiren bir komuttur. P1 parametresi kaynak kütüğü, P2 parametresi HEX kütüğü, P3 parametresi PRN kütüğünü göstermektedir.
2. A>ASM X.ABX komutu çevrilecek kaynak kütüğünün A disketinde olduğunu, HEX kütüğünün B disketinde oluşturulacağını, PRN kütüğünün CRT ekranına aktarılacağını gösterir.
3. A>ASM X.BZZ komutu X isimli kaynak kütüğünü B disket sürücüsünden alır. HEX ve PRN kütüklerini üretmez. Bu komut genellikle programın yazılımındaki hataların kontrolü ve dolayısıyla programın hızla yürütülmesi için kullanılır. Kaynak programın formatı Intel 8080 çevirici formatına uygundur. CP/M çevirici "assembler" programı MACRO komutları yürütmez. Bu çeviricinin kullanıcıya büyük kolaylık sağlayan komutları vardır.

2. PROGRAMIN GÖRÜNÜMÜ

"Assembler" programının formatı aşağıdaki gibidir.

```
satır no  etiket  işlem           işlenen  ;  açıklama
"line #"  "label" "operation"  "operand"; "comment"
```

Her "assembly" satırı bir <cr> ve <lf> (geri dön ve satır atlatma) işlevleri ile sonlandırılır. (Satır atlatma otomatik olarak ED programı içine yerleştirilmiştir. Veya ":" karakteri satırda bulunuyorsa bu "assembler" tarafından bir "satırın-sonu" komutu olarak kabul edilir.) Böylece bir satır ":" sembolü ile ayrılmışsa aynı fiziksel satırın sonuna kadar bir takım komutlar yazılabilir.

"Line #" isteğe bağlı, kaynak program satırnumarası olarak basılan bir desimal tamsayı değeridir. Genelde bu satır numaraları asıl programa bağlantı için kullanılıyorsa bunlar yerleştirilecektir.

"label" etiket alanı aşağıdaki şekilleri alır.

Tanımlayıcı (identifier) veya

Tanımlayıcı:(identifier:)

Etiket alanını tanımlayıcı olarak alfa nümerik karakterler kullanılır. Yalnız ilk karakterin alfabetik olması gerekir. Tanımlayıcılar etiket elemanları olarak kullanıcı tarafından program adımlarında özgürce kullanılırlar. Ancak bu etiketlerin uzunluğu 16 karakteri aşmamalıdır. Bütün karakterler tanımlayıcı içersinde kullanılabilirler. İçinde dolar "\$" sembolü olan isimler daha okunaklı bir duruma getirilerek kullanılırlar. Bundan başka bütün küçük harfler büyük harf gibi işlem görürler. Tanımlayıcı etiket işlevini izleyen ":" karakteri zorunlu değildir.

Aşağıda geçerli etiketler verilmiştir.

X XY long\$name

X: YX1: longer\$named\$data:

X1Y2 X1X2 X234\$5678\$9012\$3456:

Etiketden sonra gelen "operation" alanı bir çevirici direktifini, sözde işlemini (pseudo-operation = örneğin bir DS komutudur), yahut 8080 makina kodu işlemlerini içerir. Sözde ve makina kodu işlemleri aşağıda belirtilmiştir.

"Operand" alanı genellikle sabitlerin dışında etiketlerin belirttiği deyimlerin formatında oluşturulur. Bu deyim formatları aşağıda detaylı olarak verilmiştir.

"Comment" açıklama alanı ";" sembolünü izleyen keyfi karakterleri içerir. Mantıki "end-of-line" komutu gelene kadar açıklama yazılabilir.

Bu karakterler çevirici tarafından okunur, listelenir, ancak çeviri işlemine katılmaz. CPM çevirici "*" ile başlayan bir satıra da bir açıklama deyimini gibi davranır. Bu listelenir, ancak çevirme işleminden geçirilmez.

"Assembly" dili programı zorunlu olmayan bir END deyimini ile sonlandırılır. END'i izleyen bütün deyimler çevirici (ASM programı) tarafından işleme katılmazlar.

3. "OPERAND" ALANININ BİÇİMLENDİRİLMESİ

"Operand" alanındaki işlenenler (etiketler, sabitler ve ayrılmış kelimeler), aritmetik ve lojik işlemler alt deyimler içinde birleştirilir. Her deyim çevrim esnasında 16 bitlik bir değer oluşturmalıdır. Bundan başka kullanılan deyim belirtilen "digit" sayısını aşmamalıdır. Yani bir deyim bir baytlık derhal yükleme komutu kullanıyorsa (Immediate Move) deyim yüksek değerli 8 biti sıfır olmalıdır.

3.1. Etiketler (Labels)

Yukarıda belirtildiği gibi bir etiket ifadenin bir parçasını oluşturan bir tanıtıcıdır. Genelde etiket bir değerin belirtilmesi için komutlardan önce olarak verilir.

Makina kodu üretilen veya bellekde bölge ayrılan bir komutun üzerinde bir etiket varsa, etiketde bellek adresinin değeri verilir. Etiketden önce bir EQU veya SET oluşmuşsa "operand" alanında bu etiketin değeri belirtilir. SET deyimi hariç bir tanıtıcıya yalnız bir ifade iliştilerilebilir. "Operand" alanında bir etiket görüldüğünde bu değere çevirici "assembler" tarafından esas değeri yerleştirilir.

3.2. Nümerik Sabitler (Numeric Constants)

Nümerik sabit, çeşitli sayı sistemlerindeki 16 bitlik bir değerdir. Bu sistemler çeşitli tabanlarla ifade edilirler ve bu tabanlar sabitin yanında belirli harflerle belirtilir. Bunlar aşağıda gösterilmiştir.

- B "Binary" ikili sabit, "taban 2"
- O "Octal" sekizli sabit, "taban 8"
- Q "Octal" sekizli sabit, "taban 8"
- D "Decimal" onlu sabit, "taban 10"
- H "Hexdecimal" onaltılı sabit "taban 16"

Burada kullanılan "Q" sekiz tabanlı sabit sayı göstergesidir. Bu ilk hanedeki kullanılabilecek "0" sıfır sayısı ile karıştırılmaması için kullanılabilir. Herhangi bir taban göstergesi ile sonlandırılmayan nümerik sabitler desimal sabitler olarak kabul edilir. Bir sabit en sonundaki hanenin yanında tabanı gösteren harf ile düzenlenebilir. İkili sabitlerin haneleri 0 ve 1 lerden sekizli sabitlerin haneleri 0-7 arasındaki rakamlardan, onlu sabitler 0-9 arasındaki rakamlardan ve onaltılı

sabitler onlu rakamlardan ve ilâve olarak A,B,C,D,E ve F harflerinden oluşmuştur.

Tanıtıcı etiket içine yerleştirildiği gibi sabit içine de "\$" işareti yerleştirilerek sabitin daha okunaklı olması sağlanabilir. Nümerik sabitlerin geçerli şekilleri aşağıda verilmiştir.

1234	1234D	1100B	1111\$0000\$1111\$0000B
1234H	0FFEh	33770	33\$77\$22Q
33770	0fe3H	1234d	0ffffh

3.3. Ayrılmış Kelimeler (Reserved Words)

"Operand" alanında kullanılan, önceden tanımlanmış çeşitli karakterler vardır. 8080 mikroişlemcisinin yazmaçlarının isimleri olan bu karakterler aşağıda verilmiştir. Bu karakterler ile karşılaşıldığında, bu karakterlerin sağlarında yer alan değerler komut içinde işlenir.

A	7
B	0
C	1
D	2
E	3
H	4
L	5
M	6
SP	6
PSW	6

Bu karakterlerin küçük harfleri de aynı şekilde değerlendirilir.

3.4. Yazısal Sabitler (String Constants)

Yazısal sabitler ASCII karakterler şeklinde belirtilir. Ve bu yazısal- lar (') sembolü ile kapatılırlar. Bütün yazısalllar geçerli fiziksel satırı kapsamalı ve 64 karakteri aşmamalıdır. Çoğu durumlarda yazısal

uzunlukları, değerleri 8 bit veya 16 bit olan bir veya iki karakterle sınırlanır. İki karakterli bir yazısal 16 bitlik bir sabit olur. Birinci karakter yüksek ağırlıklı bayt, ikinci karakter düşük ağırlıklı bayttır. Bir karakterin değeri ASCII kodlar ile belirtilmiştir. Burada küçük harflerin, büyük harflere çevrimi sözkonusu değildir. Hangi karakterler basılmış ise o karakterlerin basımı sözkonusudur. Kullanıcı yalnız ASCII olan grafik karakterleri kullanabilir.

Geçerli yazısallar ve basılı değerleri aşağıda verilmiştir.

'A' 'B' 'ab' 'c'	A B ab c
'Walla Walla Wash.'	Walla Walla Wash.
'She said "Hello" to me.'	She said 'Hello' to me.
'I said "Hello" to her.'	I said 'Hello' to her.

3.5. Aritmetik ve Mantıksal İşlemler

İşlenenler (operands), yukarıda belirtilmiş normal cebirsel notasyonları kullanarak herhangi bir kombinezonu işlenenler, işlemler ve parantez deyimleri şeklinde birleştirebilirler. İşlemler "operand" alanında düzenlenirler. Bunlar, aşağıda verilmiştir.

$a+b$	a ve b nin işaretlenmemiş aritmetik toplamı
$a-b$	a ve b arasındaki işaretlenmemiş aritmetik fark
$+b$	b üzerinde toplama işlemi
$-b$	$(0-b)$ işlemi
$a*b$	a ve b nin işaretlenmemiş çarpımı
a/b	a nın b ile işaretlenmemiş bölümü
$a \text{ MOD } b$	a/b işleminden sonra kalıcı değer
$\text{NOT } b$	b nin mantıksal tersi ($\bar{b}$)
$a \text{ AND } b$	Bit, bit a ve b nin VE (AND) işlemine tutulması

- a OR b Mantıksal VEYA (OR) işlemi
- a XOR b Mantıksal "exclusive" işlemi
- a SHL b a daki değeri b nin miktarı kadar sola kaydır ve sıfır doldur.
- a SHR b a daki değeri b nin miktarı kadar sağa kaydır ve boşalan yerleri sıfırla doldur.

Her durumda a ve b olarak basılı olan değerler (etiketler, nümerik sabitler, bir veya iki karakter yazısalları) yahut parantez içindekiler aşağıdaki gibi işlenir.

10+20 10h+27Q LI/3 (L2+4)SHR3
 ('a' and 5fh)+'Ø' ('B'+B) OR (PSW+M)
 (1+(2+C)) shr (A-(B+1))

3.6. Öncelikli İşlemler

Programcıya bir kolaylık olarak ASM uygulamada bir bağıl önceliği olan işlemlerin parantez içine alınmadan uygulanmasına izin verir. Bağıl öncelik sırası ifadenin parantezlerini üstüne alır. Parantez içinde olmayan uygulanacak işlemlerin sırası aşağıda verilmiştir. İşlemler en yüksekte en alçak öncelik sırasına göre dizilmişlerdir. (Bunlar parantez içinde olmayan terimlerden önce uygulanırlar) Öncelik sırası eşit olan işlemler aynı satıra yazılmışlardır. Ve soldan sağa doğru uygulanırlar.

* / MOD SHL SHR

- +

NOT

AND

OR XOR

Aşağıda gösterildiği gibi ifadeler, sağda gösterilen tam olarak paran-

tez içine alınan ifadelere benzer olarak çevirici tarafından yorumlanır.

$a * b + c$	$(a*b) + c$
$a + b * c$	$a + (b*c)$
$a \text{ MOD } b * c \text{ SHL } d$	$((a \text{ MOD } b)*c) \text{ SHL } d$
$a \text{ OR } b \text{ AND NOT } c+d \text{ SHL } e$	$a \text{ OR } (b \text{ AND } (\text{NOT } C+d \text{ SHL } e)))$

Parantez içine alınmış alt deyimlerdeki parantezler pek önemsenmeden kullanılır. Şöyleki,

$(a \text{ OR } b) \text{ AND } (\text{NOT } c)+d \text{ SHL } e$

bu ifadenin çeviricide yorumlanması aşağıdaki gibi olacaktır.

$(a \text{ OR } b) \text{ AND } ((\text{NOT } c)+(d \text{ SHL } e))$

4. ÇEVİRİCİ DİREKTİFLERİ (ASSEMBLER DIRECTIVES)

Çevirici direktifleri, çeviri esnasında, koşullu çeviricinin gelişmesinde, depo alanının tanımlanmasında veprogramda başlangıç adreslerinin belirtilmesinde, etiketleri belirli değerlere yerleştirmek için kullanılırlar. Her çevirici direktifi, satırın işlem (operation) alanında gözüken bir sözde işlemi vasıtasıyla gösterilir. Kabul edilen sözde işlemler (pseudo-operations) şunlardır.

ORG	Program yada veri başlangıcının yerleştirilmesi.
END	Programın sonu, zorunlu olmayan başlangıç adresi.
EQU	Nümerik "equate"
SET	Nümerik "set"
IF	Koşullu çeviricinin başlangıcı
ENDIF	Koşullu çeviricinin sonu
DB	Tek baytlık veri alanının tanımlanması

DW Çift baytlık veri alanının tanımlanması

DS Veri depo alanının tanımlanması.

4.1. ORG Direktifi

ORG direktifi aşağıdaki şekilde gösterilir.

"label" ORG "expression"

"label" alanı zorunlu olmayan, ancak programı tanıtıcı bir etikettir.

"expression" 16 bitlik bir ifadedir. ORG deyiminden sonra tanımlanmış işlenenin değerini içerir.

Çevirici "expression" içinde belirtilen bölgede makina kodları yerleştirmeye başlar. Programcının bellek alanında iki programın üst üste gelmesini kontrol etme olanağı yoktur. Kullanıcı pekçok programı CP/M geçici program alanının başına yazar. Bunun sonucunda makina kodunu içeren programlar geçici program alanının başından itibaren yerleşir. ORG direktifinin içinde bir etiket belirtilmişse etiket ifadenin değerini verir.

4.2. END Direktifi

END direktifi bir çevirici dili programında zorunlu olmayan bir deyimdir. Fakat bu deyim sonuncu deyim olarak gerçekleştirilmelidir. Bundan sonra gelen bütün deyimler çevirici program tarafından hesaba katılmazlar. END direktifinin iki şekli vardır. Bunlar aşağıda verilmiştir.

label END

label END expression

Bu deyimlerde etiket "label" zorunlu değildir. Birinci şekil kullanılıyorsa çeviri işlemi durur. Ve programın olmayan başlangıç adresi 0000a alınır. "Expression" içinde programın başlangıç adresi belirtilir. (Bu başlangıç adresi çevirici programının yürütülmesinden sonra "hex" kütük

bu satır değiştirilerek programa tekrar çeviri işlemi uygulanmıştı.
içinde Intel formatlı makina kodları olarak sonuncu kayıtın içine eklenir. Böylece CP/M çevirici dili programlarının çoğunun sonu

4.3. SET END 100H.

deyimi ile bitirilir. (Bu geçici program alanının başlangıcının belirtilmesidir.)

4.3. EQU Direktifi

EQU (equate) direktifi, bir araya getirilmiş belirli nümerik değerlerin başlangıcı için kullanılır. Bu deyim kullanılışı aşağıdaki gibidir.

"label" EQU "expression"

Burada etiket "label" gerçek olmalı ve diğer ifadelerin etiketleri olmamalıdır. "Expression" alanında belirtilen değer etiket alanında belirtilen ifadenin değerini verir. Bu etiket program satırları için bir parametre olarak kullanılır. Örnek olarak bir "teletype" cihazından alınan ve gönderilen veriyle ilgili program bölümü aşağıdaki şekilde tanımlanabilir.

TTYBASE EQU 10H ; TTY için "PORT" numarası

TTYIN EQU TTYBASE ; TTY ye veri girişi

TTYOUT EQU TTYBASE+1; TTY ye veri çıkışı

Bir sonraki noktada bu deyimlerin program içinde kullanılışları görülmektedir.

IN TTYIN ; TTY deki veriyi A yazmacına oku

OUT TTYOUT ; A yazmacındaki veriyi TTY ye yaz

Program içersinde kullanılan TTYTYPE iletim kapısı (port) adresi 10H yerine 7FH olarak tanımlanabilir. Bu amaca uygun olarak yalnız ilk deyim değiştirilir.

TTYBASE EQU 7FH ; TTY için kapı numarası

Bu satır değiştirilerek programa tekrar çeviri işlemi uygulandığında diğer satırlarda hiçbir değişiklik yapılmadan düzeltme yapılmış olur.

4.4. SET Direktifi

SET direktifi EQU direktifinin benzeridir.

"label" SET "expression"

biçiminde kullanılır. Etiket (label) program içinde diğer SET deyimleri üzerinde meydana gelmedikçe yukardaki ifadelerde verilen değerler geçerli olur. Böylece EQU deyimi bir etiketi bir tek değerle tanımlarken SET deyimi, bir değeri geçerli SET deyiminden bir sonraki etiketde belirtilen SET deyimi gelene kadar tanımlar. SET deyimi EQU deyimi gibi, koşullu çevirici programı içinde sık olarak kullanılır.

4.5. IF ve ENDIF Direktifleri

IF ve ENDIF direktifleri, programın bir bölümünün çeviri işlemi esnasında ilâve edilmesini veya hariç tutulmasını sağlar.

IF "expression"

statement #1

statement #2

statement #n

ENDIF

IF deyiminin içinde bulunan satırların yürütülmesini, bu deyimi izleyen ifade (expression) belirler. Eğer ifade (expression) sıfır olmayan bir değer ile belirtilmişse program "statement #1" den "statement #n" e kadar işlenir. İfade sıfır ise, program satırları listelenir fakat çevrilmezler. Koşullu çeviri işlemi geniş kapsamlı programlarda kullanılır. Programın bir kısmı TTY veya CRT düzenlerinden biri ile yürütülecekse aşağıdaki şekildeki gibi program satırları oluşturulur.


```

TRUE      EQU      0FFFFH      ; TRUE nin deęeri
FALSE     EQU      NOT TRUE     ; FALSE ın deęeri
;
TTY       EQU      TRUE         ; TRUE = TTY   FALSE = CRT
TTYBASE   EQU      10H          ; TTY nin I/O kapısı
CRTBASE   EQU      20H          ; CRT nin I/O kapısı
          IF      TTY           ; TTY bazının baęıl evrimi
CONIN     EQU      TTYBASE       ; Konsol giriři
CONOUT    EQU      TTYBASE+1     ; Konsol ıkıřı
          ENDIF
          IF      NOTTTY        ; CRT bazının baęıl evrimi
CONIN     EQU      CRTBASE       ; Konsol giriři
CONOUT    EQU      CRTBASE+1     ; Konsol ıkıřı
          ENDIF
          IN      CONIN          ; Konsol veriřini oku
          OUT     CONOUT         ; Konsol veriřini aktar.

```

Program TTY iin konsol giriř ve ıkıřı olarak 10H ve 11H kapılarını seer. TTY nin tanımlanmıř olduęu komut satırı ařaęıdaki gibi deęiřtirilecek olursa

```
TTY EQU FALSE
```

program giriř ve ıkıř kapısı olarak 20H ve 21H kapılarını kabul eder.

4.6. DB Direktifi

DB komutu programcının bir baytlık alanları tanımlamasına izin verir.

Bu deyimın kullanılıř řekli ařaęıda verilmiřtir.

```
label DB e#1, e#2, e#2 ..... e#n
```

Burada e#1 den e#n e kadar olan ifadeler 8 bitlik deęerler (en yksek deęerli bit 7. bit sıfır olmalıdır.) veya 64 karakter uzunluęunu gemeyen

ASCII yazısallar olmalıdır. Tek bir kaynak satırda ifadelerin sayısının kısıtlanması pek pratik bir yol değildir. Yazısal karakterler bellek içine başlangıcı ilk karakter, sonu sonuncu karakter olmak üzere yerleştirilir. ASCII karakterlerin eşlik (parity) bitleri sıfıra çekilmiş olarak yerleştirilir. Küçük harften büyük harfe bir çeviri söz konusu değildir. Zorunlu olmayan etiket veri alanlarının tayini için kullanılır. Geçerli DB deyimlerinin örnekleri şunlardır.

data: DB 0,1,2,3,4,5

DB data and 0ffh,5,377Q,1+2+3+4

sign-on: DB 'please type your name',cr,lf,0

DB 'AB' SHR 8, 'C','DE' AND 7FH

'AB' SHR 8, 'C','DE' AND 7FH

4.7. DW Direktifi

DW direktifi, DB direktifinin benzeridir. Ancak burada her ifade için iki bayt (word) yer ayrılmıştır. Bu komutun şekli aşağıdaki gibidir.

"label" DW e#1, e#2, e#n

Burada e#1 den e#n e kadar olan ifadeler 16 bitlik değerler olarak yerleştirilir. Kullanıcı bir veya iki karakter olarak ASCII yazısallar belirtilebilir, ancak yazısal uzunluğu iki karakteri geçmemelidir. Belirtilen ifadenin düşük değerli baytı ilk olarak, yüksek değerli baytı son olarak belleğe yerleşir. Bu deyimle ilgili geçerli komutlar aşağıda verilmiştir.

doub: DW 0ffefh, doub+4, signon=0,255+255

DW 'a',5,'a','CD',6shl 8 or llb

4.8. DS Direktifi

DS direktifi bellekte özel bölgeler ayırmak için kullanılır. Bunun kullanılışı aşağıdaki gibidir,

"label" DS "expression"

Etiket (label) zorunlu değildir. Çevirici DS deyiminden sonra gelen ar-
dışıl kod ile bellekte yer ayırır. Aşağıdaki deyimlerde DS komutu ile
aynı işi görürler.

label: EQU \$; Etiketde belirtilen değerin geçerli
yer kodu

ORG \$ expression ; Ayrılacak olan alan

5. İŞLETİM KODLARI

Çevirici dili işletim kodları, çevirici dili program şekilleri ve işle-
tim komutları aşağıda verilmiştir. Genel olarak ASM programı 8080 mikro-
işlemcisinin komutlarını kabul eder. Her giriş satırı üzerindeki etiket-
ler daha önce açıklandığı gibi zorunlu değildir.

- e3 0-7 arasında A,B,C,D,E,H,L,M,SP,PSW yazmaçlarından biri be-
lirtilir. Üç bitlik değer basılır.
- e8 0-255 değerleri arasındaki 8 bitlik bir değer basılır.
- e16 0-65535 değerleri arasındaki 16 bitlik bir değer basılır.

5.1. Dallanma, Çağırma ve Geri Dönme Komutları (Jumps, Calls, Returns)

JMP	e16	JMP	L1	L1 etiketine koşulsuz olarak atla.
JNZ	e16	JNZ	L2	Z=0 ise L2 de belirtilen adrese git.
JZ	e16	JZ	100H	Z=1 ise 100H adresine git.
JNC	e16	JNC	L1+4	C=0 ise (L1+4) etiketine git.
JC	e16	JC	L3	C=1 ise L3 etiketine git.
JPO	e16	JPO	\$+8	Eşlik (Parity) biti çift ise (\$+8) etiketine git.
JPE	e16	JPE	L4	Eşlik (Parity) biti tek ise L4 etiketine atla.
JP	e16	JP	GAMMA	Sonuç pozitif ise "GAMMA" etiketine git.
JM	e16	JM	a1	Sonuç negatif ise "a1" etiketine git.

CALL	e16	CALL	S1	Koşulsuz olarak S1 alt yordamını çağır.
CNZ	e16	CNZ	S2	Z=0 ise S2 altyordamını çağır.
CZ	e16	CZ	100H	Z=1 ise "100 H" deki altyordamı çağır.
CNC	e16	CNC	S1+4	C=0 ise "S1+4" altyordamını çağır.
CC	e16	CC	S3	C=1 ise S3 altyordamını çağır.
CPO	e16	CPO	\$+8	Eşlik (Parity) biti çift ise "\$+8" adresini çağır.
CPE	e16	CPE	S4	Eşlik (Parity) biti tek ise S4 etiketinin gösterdiği adresi çağır.
CP	e16	CP	GAMMA	Sonuç pozitif ise "GAMMA" altyordamını çağır.
CM	e16	CM	b1\$C2	M=1 ise (negatif flag) "b1\$C2" altyordamını çağır.
RST	e3	RST	0	Bir baytlık çağırı komutudur. Programlanabilir. "restart" komutudur.
RET				Altyordamdan koşulsuz olarak dön.
RNZ				Z=0 ise altyordamdan dön.
RZ				Z=1 ise altyordamdan dön
RNC				C=0 ise (Elde biti) altyordamdan dön.
RC				C=1 ise (Elde biti) altyordamdan dön.
RPO				Eşlik biti (Parity) çift ise altyordamdan dön.
RPE				Eşlik biti (Parity) tek ise altyordamdan dön.
RP				M=0 ise (Pozitif) altyordamdan dön.
RM				M=1 ise (Minus) altyordamdan dön.

5.2. Hemen İşlenen Komutlar (Immediate operand instructions)

MVI	e3,e8	MVI B,255	Veriyi hemen (B yazmacına burada örnek olarak) A,B,C,D,E,H,L yazmaçlarına bellek gözüne (M) yerleştir.
ADI	e8	ADI L	"Operand"ı (burada L) A akumlatöründeki değerle "carry" olmaksızın topla.

ACI	e8	ACI 0FFH	"Carry" elde biti ile birlikte akümlatörü işle- nen "operand" ile topla.
SUI	e8	SUI L+3	A akümlatöründen "operand"ı "borrow (carry)" olmaksızın çıkar.
SBI	e8	SBI LAND 11B A	akümlatöründen işlenen "Operand"ı barrow(carry)" ile birlikte çıkar.
ANI	e8	ANI 8 AND 7FH	İşlenen alanında verilen veriyi akümlatördeki değer ile lojik AND işlemine tut.
XRI	e8	XRI 111180000B	Akümlatördeki değer ile işlenene "Exclusive or" işlemi uygula.
ORI	e8	ORI LAND 1+1	A akümlatörü ile işlenene "OR" işlemini uygula.
CPI	e8	CPI 'a'	A akümlatöründeki değer ile tırnak içindeki de- ğerin ASCII karşılığını karşılaştır.
LXI	e3,e16	LXI B,100H	Belirtilen yazmaç çiftine 16 bitlik değeri der- hal yükle (e3 B,D,H yahut SP olmalıdır).

5.3. Artırma ve Azaltma Komutları (Increment and Decrement Instructions)

8080 mikroişlemcisi tek yada çift yazmaçlar için artırma ve azaltma iş-
levlerine sahiptir. Bunlar şunlardır.

INR	e3	INR E	Belirtilen yazmaçdaki değeri bir artır. e3=A,B, C,D,H,L,M (bellek)
DCR	e3	DCR A	Belirtilen yazmaçdaki değeri veya bellek gözün- deki değeri bir azalt.
INX	e3	INX SP	e3 de belirtilen (B,D,H veya SP) yazmaçlardaki değeri bir artır.
DCX	e3	DCX B	e3 de belirtilen yazmacın içeriğini bir artır. (e3 B,D,H veyahut SP olmalıdır.)

5.4. Veri Aktarma Komutları (Data Movement Instructions)

Bellekden, CPU ya ve CPU dan, belleğe verinin taşınmasıyla ilgili komutlar aşağıda verilmiştir.

MOV	e3,e3	MOV	A,B	B yazmacındaki veriyi A yazmacına aktar. (Burada e3 A,B,C,D,E,H,L veya M dir.) MOV M,M komutuna izin verilmez.
LDAX	e3	LDAX	B	BC yazmacının gösterdiği adresdeki veriyi A akümülatörüne yükle (e3 B veya D yazmaçlarını göstermelidir.)
STAX	e3	STAX	D	A akümülatörünü DE veya BC yazmacının gösterdiği adrese aktar. (e3 D veya B olmalıdır.)
LHLD	e16	LHLD	L1	e16 belirtilen adresteki veriyi HL yazmacına yükle
SHLD	e16	SHLD	L5+X	e16 adresine HL yazmacının içeriğini depola
LDA	e16	LDA	GAMMA	e16 adresindeki veriyi A akümülatörüne yükle
STA	e16	STA	X3-5	e16 adresine A akümülatöründeki veriyi aktar.
POP	e3	POP	PSW	SP (Stack Pointer)nin gösterdiği bellekdeki değeri e3 yazmaç çiftine aktar. (Burada e3 B,D,H veya PSW yazmaçları olmalıdır.)
PUSH	e3	PUSH	B	e3 de belirtilen yazmaç çiftini yığıtın içine aktar. Yığıtın yerini yığıt göstergesi (SP) belirtmektedir. (e3 B,D,H ve PSW olmalıdır.)
IN	e8	IN	0	e8 kapısında (port) bulunan değeri A akümülatörüne aktar.
OUT	e8	OUT	255	A akümülatöründe bulunan değeri e8 kapısına (port) aktar.

XTHL				Yığıtın üstündeki değer ile HL içeriğini değiştir.
PCHL				Program sayacını HL in içeriği ile doldur.
SPHL				Yığıt göstergesini HL in içeriği ile doldur.
XCHG				DE yazmaç çiftini HL yazmaç çifti ile değiştir.

5.5. Aritmetik ve Mantıksal İşlem Komutları

ADD	e3	ADD	B	e3 de belirtilen yazmaç ile akümülatörü elde (carry) biti olmaksızın topla (e3 A,B, C,D,E,H veya L yazmaçlarından biri olmalıdır)
ADC	e3	ADC	L	e3 de belirtilen yazmaç ile akümülatörü elde (carry) bitiyle birlikte topla.
SUB	e3	SUB	H	e3 de belirtilen yazmacı elde biti olmaksızın A akümülatöründen çıkar. e3 yukardaki gibi tanımlanabilir.
SBB	e3	SBB	2	e3 de belirtilen yazmacı "carry" ile birlikte A akümülatöründen çıkar.
ANA	e3	ANA	1+1	e3 de belirtilen yazmaç ile A akümülatörüne "and" işlemi uygula.
XRA	e3	XRA		e3 de belirtilen yazmaç ile A akümülatörüne "exclusive or" işlemi uygula.
ORA	e3	ORA	B	e3 de belirtilen yazmaç ile A akümülatörüne "OR" işlemi uygula.
CMP	e3	CMP	H	e3 de belirtilen yazmaç ile A akümülatörünün içeriğini karşılaştır.
DAA				Son ALU işlemine göre A akümülatörünü desimal değere ayarla.

CMA	Komut (Command) hatası:	A akümülatöründeki değerin bütün bitlerinin evriğini al.
STC	Veri (Data) hatası:	Elde "Carry" bayrağını bir yap.
CMC	Değil.	Elde bayrağının tersini al. (1 ise 0, 0 ise 1 yap.)
RLC	İndirilmiyor.	A akümülatöründeki bütün bitleri sola ötele.
	Etiket (Label) hatası:	A nın en yüksek değerli biti "carry flag" in değerini alır.
RRC	İnliyor.	A akümülatöründeki bütün bitleri sağa ötele
	Taşı (Overflow): ifade:	Akümlatörün en küçük değerli biti "Carry flag" in değerini alır.
RAL	Yazmaç (Register) hatası:	A akümülatörünü sola döndür.
RAR	Değil.	A akümülatörünü sağa döndür.
DAD	e3 DAD B	e3 de belirtilen yazmaç çiftini HL yazma- cına ilâve et. (e3 B,D,H veya SP yazmaçla- rı olmalıdır.)

5.6. Kontrol Komutları (Control Instructions)

8080 mikroişlemcisinin 4 tane kontrol komutu vardır. Bunlar,

HLT	8080 mikroişlemcisinin durdurulması,
DI	Kesinti sisteminin olanaksız kılınması,
EI	Kesinti sisteminin olanaklı kılınması,
NOP	İşlem yok.

6. HATA MESAJLARI (ERROR MESSAGES)

Çevirici dili programının içinde hata oluştuğunda bu hatalar kaynak lis-
tenin en solunda tek bir karakterle belirtilir. Bu hata kodları şunlardır.

- C : Komut (Command) hatası: Komut doğru olarak tanımlanmamış; sözdizim (syntax) hatasının benzeridir.
- D : Veri (Data) hatası: Veri deyim elementleri belirtilmiş veri alanında değil.
- E : İfade (Expression) hatası: İfade çevirim (assembly) sırasında değerlendirilmeyecek.
- L : Etiket (Label) hatası: Metin içinde etiket bulunamıyor.
- N : Program yürütülemiyor (Not Implemented): Yürütülen programlar engelleniyor.
- O : Taşma (Overflow): İfadenin kıymetini belirtmek oldukça güç.
- P : Faz (Phase) hatası: İkinci geçişte etiket aynı değeri koruyamıyor.
- R : Yazmaç (Register) hatası: Belirtilen yazmaçdaki değer kod ile uygun değil.
- S : Sözdizim (Syntax) hatası: Uygun olmayan şekilde bir ifade kullanılmış.
- U : Tanımlanmamış (Undefined) etiket: Kullanılan etiketin değeri yok.
- V : Değer (Value) hatası: Bir ifadede uygun olmayan biçimde operand (işlenen) var.
- Çeşitli hata koşullarında CRT ekranında görülen mesajlar da şunlardır:
- NO SOURCE FILE PRESENT : ASM komutu ile belirtilmiş kaynak kütük disket üzerinde yok.
- NO DIRECTORY SPACE : Disk rehberi dolu.
- SOURCE FILE NAME ERROR : Uygun olmayan ASM kütük ismi.
- SOURCE FILE NAME ERROR : Kaynak kütüğü ASM tarafından uygun bir şekilde alınamıyor; TYPE komutu ile icra ederek hata noktasını bulabilirsiniz.
- OUTPUT FILE WRITE ERROR : Çıkış kütükleri uygun bir şekilde yazılmamış. Bunun sebebi diskin tamamen dolu olmasıdır.
- CANNOT CLOSE FILE : Çıkış kütüğü kapanmıyor. Disketin yazıma karşı korunmuş olup olmadığına bakın.

1. GİRİŞ

DDT (Dynamic Debugging Tool - Dinamik Düzeltme Aracı) programı CP/M çerçevesinde geliştirilmiştir; programların hata ayıklanması ve geliştirilmesi için dinamik testlerin yapılmasını sağlar.

CP/M konsol komutları yardımıyla çalıştırılan (debugger) işlemi yerinde gerçekleştirir. DDT programı aşağıdaki şekilde çalıştırılarak konsoldan çalıştırılabilir:

```
A > DDT ---
```

```
A > DDT kütükisi. EXE
```

```
A > DDT kütükisi. COM
```

Yukarıdaki yöntemlerle ve test edilerek programlar çalıştırılır. Bu durumda programlar DDT programı ile çalışır. DDT programı konsoldan çalıştırılarak çalıştırılır. DDT programı konsoldan çalıştırılarak çalıştırılır.

DÖRDÜNCÜ BÖLÜM

CP/M DİNAMİK DÜZELTME ARACI

"DYNAMIC DEBUGGING TOOL"

DDT programı çalıştırılarak çalıştırılır. DDT programı konsoldan çalıştırılarak çalıştırılır. DDT programı konsoldan çalıştırılarak çalıştırılır. DDT programı konsoldan çalıştırılarak çalıştırılır.

DDT programı konsoldan çalıştırılarak çalıştırılır. DDT programı konsoldan çalıştırılarak çalıştırılır. DDT programı konsoldan çalıştırılarak çalıştırılır. DDT programı konsoldan çalıştırılarak çalıştırılır.

"open" ve "close" ile çalıştırılır. DDT programı konsoldan çalıştırılarak çalıştırılır.

1. GİRİŞ

DDT (Dynamic Debugging Tool - Dinamik düzeltme aracı) programı CP/M çerçevesinde üretilmiş programların kusurlarının giderilmesi ve birbirini etkileyen dinamik testlerinin yapılmasını sağlar.

CP/M konsol komutları yardımıyla düzeltme (debugger) işlemi yerine getirilir. DDT programı aşağıdaki üç şekilden birisinin konsoldan girilmesi suretiyle yürütülür.

A > DDT

A > DDT kütükismi. HEX

A > DDT kütükismi. COM

Kütükismi yüklenecek ve test edilecek programın ismidir. Bu durumlardan hepsinde de DDT programı ana bellek içindeki CCP (Konsol komut işleyicisi) programının yerine getirilir. Böylece direkt olarak BDOS kısmının aşağısına oturur. Bu durumda BDOS başlangıç adresi (05H adresine yerleştirilen) geçici program alanı boyutuna (TPA) yansıtılıp, küçültülerek değiştirilir.

DDT programını yürüten yukardaki komutların ikincisi ve üçüncüsü, birinci şeklin aynısıdır, fakat yanındaki kütük isimlerini ardışıl olarak yükler. Yani belirtilen COM yada HEX kütüklerini otomatik olarak yükleyen bir işlevi vardır. Birinci şekilde gösterilen komuttan sonra HEX yada COM kütüklerin girilmesi aşağıdaki gibi olacaktır.

Komutlar 32. karaktere kadar girilebilir. 33. karakterde otomatik <cr> işlevi yerine getirilir. "-" karakterinden sonra girilebilen tek karakterli komut tipleri şunlardır.

A "operand" ve "mnemonics" ile birlikte çevirici dil girişidir.

- D Bellek içindeki "Hexdecimal" ve "ASCII" değerleri gösterir.
- F Başlangıç ve bitiş adresleri verilen belleğin içeriğini basit veri ile doldur.
- G İsteğe bağlı zorunlu olmayan kesme noktaları ile programı yürütür.
- I Bir standart giriş kütüğü kontrol bloğu ileri sürer.
- L Bellekdeki "Assembler Mnemonics"leri listelemek için kullanılır.
- M Bir bellek dilimini belirtilen bellek dilimine aktarır.
- R Giriş kütüğünü belleğe okur.
- S Bellek değerlerinin değişimi için kullanılır.
- T Belirtilen adımda programın yürütülmesini ve CRT ekranından izlenmesini sağlar.
- U Program yürütülür ancak ekrandan izlenmez.

Herhangi bir noktanın düzeltimi esnasında, işletici DDT nin icrasını `ctl-c` tuşunu kullanarak durdurabilir. `GØ` komutunu (`ØØØØH` e atla) kullanarak DDT Programından çıkabilir. `SAVE` komutunu kullanarak geçerli bellek görüntüsünü saklayabilir. `SAVE` komutunun şekli

`SAVE n kütükismi . COM` şeklindedir.

`n` Disket üzerine kaydedilecek sayfaların (256 baytlık bloklar) sayısıdır. TPA içindeki saklanacak alanın içeriği kaç sayfa tutuyorsa desimale çevrilir. Ve `n` olarak belirtilir. Örneğin geçici program alanındaki en yüksek adres `1234H` olsun. Sayfaların sayısı `12H` veya 18 desimaldır. İşletici DDT nin çalışmasını durdurup `CCP` ye dönüp aşağıdaki işlemi yapabilir.

`SAVE 18 X . COM`

Belleğin bir kopyası `X.COM` isimli kütüğe aktarılmıştır. Bu kütük direkt olarak icra edilebilir. Eğer tekrar bir düzeltme işlemi uygulanacak olursa bu kütük aşağıdaki komutla çağrılabilir.

DDT X.COM

Bu komutla X.COM programı 1000H adresinden 12FFH adresine (18 sayfa) kadar belleğin içine yüklenir.

2. DDT KOMUTLARI

DDT içinde kullanılan komutlar detaylı olarak aşağıda verilmiştir. Her durumda işletici (kullanıcı) komutları girmek için (-) karakterinin çıkmasını beklemelidir.

Kontrol test edilecek programdan başka bir programa geçmişse ve programda bir kesme noktası yoksa ön panelden bir RST 7 icrasıyla kontrol DDT ye döndürülebilir.

Her komutun açıklamasında komut harfi, bazı durumlarda yanına ilave edilen küçük harflerle kullanılmıştır. Bu harfler bazı sayıların yerine geçmektedir. Bu sayılar "hexadecimal" tabanlı sayı sisteminde uzunluğu 1 den 4 e kadar olan sayıları gösterir. (Dörtten uzun sayılar otomatik olarak sağdan kırpılırlar)

2.1. A (Assembly) Komutu

DDT izlencesi bir As komutu kullanarak "in-line assembly" dili komutlarını geçerli bellek bölgesine yerleştirir. "in-line assembly" için s bir heksadesimal başlangıç adresidir. DDT konsoldan harekete geçirilir. Nemonik ve işlenen yazıldıktan sonra gelecek komutun yazılacağı adres okunur. Yazmaçlar ve işlenen (operand)ler gerçek heksadesimal şekilde kaydedilir. Her ardışıl yükleme adresi konsoldan okumadan önce basılır. A komutu bir boş giriş satırı girildiğinde sonlandırılmış olur.

Kullanıcı A komutunun testini L komutu ile yapabilir.

2.2. D (Display) Komutu

D komutu kullanıcının bellek içindeki heksadesimal ve ASCII karakterleri şeklindeki değerleri görmesini sağlar. Komut şekilleri aşağıdaki gibidir.

D

Ds

Ds,f.

Birinci durumda geçerli gösterge adresinden itibaren 12 devamlı gösterge satırı verilir. Her gösterge satırının aldığı şekil aşağıda verilmiştir.

```
aaaa bb bb bb bb bb bb bb bb bb bb bb bb bb bb ccccccccccccc
```

aaaa Heksadesimal olarak gösterge adresini vermektedir.

bb ise aaaa adresinden itibaren bellekteki dataları vermektedir.

cccc ise aaaa adresinden başlayan ASCII karakterleri vermektedir. Grafik olmayan karakterlerde bir (.) nokta basılmaktadır. Her gösterge satırı 16 baytlık verinin değerini verir. Gelecek satır 16 sayısının ilavesiyle oluşan bir adresten başlar.

D komutunun ikinci şekli ilkinin benzeridir. Burada farklı olarak başlangıç adresi ilâve edilmelidir. Üçüncü şekilde başlangıç ve bitiş adresleri verilen verilerin gösterilmesi olanaklıdır. Uzun işlemlerden <cr> tuşuna basarak çıkılabilir.

2.3. F (Fill) Komutu

F komutu aşağıdaki şekilde kullanılır.

Fs,f,c

s başlangıç adresini

f bitiş adresini ise "hexdecimal" sabiti göstermektedir.

Bu komut başlangıç ve bitiş adresleri verilen alana sabit dataları doldurmak için kullanılır.

2.4. G (GO) Komutu

G komutu bir programın yürütülmesinde ikiye kadar zorunlu olmayan kesme noktalarını adresleyerek kullanılır.

G komutu aşağıdaki şekillerdeki gibi kullanılabilir.

1. G

2. Gs

3. Gs,b

4. Gs,b,c

5. G,b

6. G,b,c

Birinci şekilde program, program sayacının gösterdiği değerden itibaren çalışmaya başlar. Burada kesme noktaları yerleştirilmemiştir. (İşlemden sonra RST 7 gerçekleşmektedir.) Program sayacının değeri X veya XP komutları ile görülebilir. İkinci şekil birincinin benzeridir ancak burada icraya başlamadan önce program sayacı s başlangıç adresine kurulmuştur. Üçüncü şekil ikincinin aynısıdır. Programın icrası b adresinde durdurulmuştur. (b test edilecek program alanında olmalıdır) b adresindeki komut bir kesme noktasını gösterdiği için icra edilmez. Dördüncü şekil üçüncü şeklin benzeridir. Ancak burada b ve c adreslerini içeren iki kesme noktası bulunmaktadır. Bu kesme noktalarından biri ile karşı karşıya gelmesi sonucu icra durdurulur ve iki kesme noktası temizlenir. Son ikinci şekil makinadaki program sayacının değerine set edilmişlerdir, bir veya

iki tane kesme noktası ilâve edilmiştir. Yürütme başlangıç adresinden gerçek zamanda içinde gelecek kesme noktasına kadar devam eder. Yani DDT tarafından başlangıç ve kesme adresleri arasındaki bölüme karışılmaz. Program test edilecekse bir kesme noktası aranmaz, bir RST 7 komutunun icrası olmaksızın kontrol DDT ye dönmez. Bir kesme noktası ile karşılaşıldığında DDT nin yürütülmesi durdurulur ve

*d

şeklinde d bitiş (stop) adresi olmak üzere bir mesaj ekrana basılır.

Makinanın durumu (mikroişlemcinin yazmaçları) bu noktada X(Examine) komutu ile gözden geçirilir. Kullanıcı G komutunun başlangıcında program sayacı adresinden farklı olan kesme noktaları belirtebilir. Bu komutlar, geçerli program sayacı adresi 1234 H ise

G, 1234 ve

G400,400

dır. Herhangi bir komut icrası olmaksızın bunların ikisi de hemen işlenir.

2.5. I (Input) Komutu

I komutu işleticinin 5CH deki kütük kontrol bloğunun içine bir kütük isminin yerleştirmesine izin verir. (CP/M tarafından geçici programlar için üretilen kütük kontrol bloğu bu adrese yerleştirilir.) Konsol komut işleyicisi tarafından ihmal edilmiş kütük isimleri bu komut kullanılarak HEX ve COM isim tipleri ilâve edilerek kütük kontrol bloğuna yerleştirilir.

I komutunun şekli

I kütükismi veya

I kütükismi . kütük tipi

dır. İkinci şekil kullanılmışsa ve kütük tipi HEX veya COM olarak belir-

tilmişse, bir sonraki ardışıl R komutu ile binary veya "hex" formatındaki makina kodları bellek içine konur. (Daha detaylı bilgiler için R komutuna bakın)

2.6. L (List) Komutu

L komutu belirli program bölgesindeki nemoniklerin listelenmesi için kullanılır. Komut şekilleri aşağıda verilmiştir.

L

Ls

Ls,f

Birinci komut geçerli listeleme adresinden itibaren "disassembler" makina kodlarını 12 satır olarak listeler. İkinci şekilde s de belirtilen listeleme adresinden itibaren 12 satırın nemonikleri listelenir. Sonuncu şekilde nemonikler s başlangıç adresinden f bitiş adresine kadar listelenir. Bu üç şekilde de listeleme adresi, gelecek bir L komutuna hazırlık amacıyla listelenmemiş bölgenin adresine kurulur. Programın yürütülmesi sırasında bir kesme noktası ile karşılaşıldığında listeleme adresi program sayacının geçerli adresine "set" edilir. (G ve T komutlarına bakınız). Uzun listelemeler bir "rubout" tuşu kullanılarak durdurulur.

2.7. M (Move) Komutu

M komutu bellek içindeki programın veya veri alanlarının bir adresden diğer adrese taşınmasına izin verir. Bu komut

Ms,f,d

şeklinde kullanılır. s ve f taşınacak alanın başlangıç ve bitiş adresidir. d ise gidilecek yerin adresini vermektedir. Veri ilk olarak s den d ye taşınır ve bu adreslerin ikisi de birer birer artırılır. s ve f karşılaştırılır bunlar birbirine eşitse taşıma işlemi durdurulur.

2.8. R (Read) Komutu

R komutu I komutu ile, düzeltim sistemlerine hazırlık amacıyla, disketteki HEX veya COM kütüklerinin geçici program alanına okunması için aşağıdaki şekillerdeki gibi kullanılır.

R

Rb

b zorunlu olmayan yüklenecek bellek bölgesini belirleyen bir adrestir. Yükleme işlemleri 00H den FFH e kadar olan bölgede bulunan sistem parametresinin üzerine yapılmamalıdır. b altlanmışsa b=0000 olarak kabul edilir. I komutunda HEX veya COM olarak kütük ismi belirtildikten sonra R komutu istenir. Her kayıt için yükleme adresi her özel "hex" kayıtda bulunabilirken COM kütükleri için yükleme adresi 100H olarak kabul edilir. R komutuna herhangi bir sayı ilâvesiyle I komutundan sonra test edilecek program bellek içine yerleştirilebilir. Bu test edilecek program 5CH deki alanı bozmaz. Kütük tipi "COM" olarak belirtilmiş kütük saf binary şeklinde makina kodlarını içerir (LOAD yada SAVE komutu ile üretilmiş). Diğer kütükler intelin hex formatındaki makina kodlarını içerirler. (Örneğin ASM komutu için üretilmiş olanlar.) Komutu tekrar çağırılım

DDT kütükismi . kütüktipi

bu yazma şekline aşağıdaki yazılım şekli eşdeğerdir.

DDT

-I kütükismi . kütüktipi

-R

R komutu verildiğinde DDT "?" şeklinde bir hata göstergesi (ki bunun açıklaması kütük açık değildir veya HEX kütüğünde bir kontrol toplamı "checksum" hatası okumuştur) veya yükleme mesajı ekrana çıkar.

Yükleme mesajı

NEXT	DC
nnnn	pppp

şeklindedir. Burada nnnn yüklenmiş programdan sonraki boş olan bölgenin adresini göstermektedir. pppp ise program sayacındaki değeri (ki bu COM kütükler için 1000H dir veya sonuncu kayıttan alınan "hex" kütüğünde belirtilen değerdir).

2.9. S (Set) Komutu

S komutu bellek bölgelerinin kontrol edilmesine ve değiştirilmesine olanak verir. S komutunun kullanma biçimi Ss şeklindedir. s değiştirilecek veya gözden geçirilecek belleğin başlangıç adresini vermektedir. DDT içersinde "s" in yerinden başlangıç adresi verildiğinde ekranda bu adres ve yanında bir baytlık adresin içeriği görülür. Kullanıcı bir <cr> tuşuna basarak bir sonra gelen adresdeki veriyi görebilir. Veya bu adreslerdeki veriyi değiştirebilir. Bunun için bir baytlık geçerli veriyi girer ve <cr> tuşuna basar. Değiştirme işleminden sonra geri dönüş bir geçersiz giriş komutunun verilmesi veya bir "(.)" karakterine basılmasıyla oluşur.

2.10. T (Trace) Komutu

T komutu 1 den 65535 program adımı için program icrasının izlenmesine olanak verir. T komutu

T

Tn

şeklinde kullanılır.

Birinci durumda CPU yazmaçlarının değeri ve icra edilecek gelecek program adımı görülebilir. Program hemen durdurulur. Sonlandırma adresi

*hhhh

şeklindedir. Bu adres gelecek icranın başlangıç adresidir. Gösterge adresi (D komutunda kullanılan) HL yazmacına set edilir. Ve listeleme adresi (L komutunda kullanılan) hhhh değeridir.

CPU durumu program sonunda X komutunu kullanarak gözden geçirilir.

T komutunun ikinci şekli birincisinin benzeridir. İlâve olarak programın yürütülmesi n adımda izlenir. (n burada heksadesimal değerdir.) Bir kesme noktası izleme modunun içine "rubout" karakteri yerleştirilerek oluşturulabilir. Her program adımı izleme moduna alınmadan önce CPU nun durumu gösterilebilir. Göstergenin formatı X komutunda verilenin aynısıdır. Program izlemesi yarıda kesilebilir ve CP/M'e dönülebilir. (CP/M'e gitikten sonra test edilecek programa dönülebilir.) Bu yüzden I/O düzenlerine erişimli CP/M işlevleri ve disket sürücüyü ulaşma işlevleri kullanılabilir.

İzleme modunda çalıştırılan programlar (DDT nin kontrolü alıp yürüttüğü programlar), gerçek zamanda yürütülen programlardan 500 kere daha yavaş yürütülür. İzleme modunda test edilecek program mümkün olan kesintiler ile çalışır. Kullanıcı "rubout" tuşunu kullanarak izleme esnasında kontrolü DDT geri alabilir.

2.11. U (Untrace) Komutu

U komutu T komutunun benzeridir. Fakat program adımları ekranda gösterilmez. Bu komut 1 den 65535 e kadar yürütülecek adıma olanak verir. T komutunun sunduğu bütün olanaklar U komutu için de geçerlidir.

2.12. X (Examine) Komutu

CPU yazmaçlarının gösterilmesine ve değiştirilmesine izin verir. Bu ko-

mut iki şekilde kullanılır.

X

Xr

Burada r 8080 mikroişlemcisinin yazmaçlarını göstermektedir. Bu yazmaçlar aşağıda verilmiştir.

C	"Carry Flag"	(0/1)
Z	"Zero Flag"	(0/1)
M	"Minus Flag"	(0/1)
E	"Even Parity Flag"	(0/1)
I	"Interdigit Cary"	(0/1)
A	A Akümülatörü	(0-FFFFH)
B	BC yazmacı	(0-FFFFH)
D	DE yazmacı	(0-FFFFH)
H	HL yazmacı	(0-FFFFH)
S	Yığıt göstergesi	(0-FFFFH)
P	Program sayacı	(0-FFFFH)

Birinci durumda CPU yazmaçlarının içerikleri aşağıdaki şekilde görülür,

CfZfMfEfIf A=bb B=dddd D=dddd H=dddd S=dddd P=dddd "Komut"

f 0 veya 1 olan bayrak değerleridir.

bb Akümülatördeki değer

dddd Yazmaç çiftindeki değer

"Komut" Program sayacı adresinde bulunan "disassembler" edilmiş komutu gösterir.

Xr komutu yazmaçlardaki veya bayraklardaki (flag) değerlerin gösterilmesi ve değiştirilmesi için kullanılır. r olarak C,Z,M,E,I,A,B,D,H,S,P harflerinden biri kullanılır. DDT programına giriş tuş takımından olur.

Bir <cr> tuşuna basıldığında yazmaçlardaki değerler değişmeden kalır.

3. YÜRÜTME NOTLARI (IMPLEMENTATION NOTES)

DDT programının organizasyonu, büyük programlarının düzeltimi için geçici program alanının büyütülmesine izin verir. DDT programı iki bölümü içerir. DDT nin esas ve assembler / disassembler modülüdür. DDT nin esas konsol komut izleyicisinin üstüne yüklenir.

Özel olarak 6H deki BDOS adresi DDT programının temel giriş adresi olmak üzere geliştirilmiş ve dönüşte BDOS'a atlaması temin edilmiştir. (05H de JUMP komutunun "hex" karşılığı vardır) DDT nin başlangıcında belleğin sonu bulunur (TPA nın) ve DDT nin özünden BDOS'a aktarılır.

Assembler/disassembler modülü özünün aşağısından geçici program alanının içinde direkt olarak oturur. A,L,T veya X komutları düzeltim işlemleri esnasında kullanıldığında bu modüle ilâve olan 6H adresi alanındaki değer değiştirilir.

1. GİRİŞ

Bu bölümde sistem organizasyonu, geliştirilme süreci ve diğer konular hakkında belirtilmiştir. CP/M işletim sistemi, kullanıcı programları, donanım, temel I/O sistemi (BIOS), dosya sistemi (FAT), kullanıcı programları (UCP) ve diğer konular hakkında bilgi vermektedir. Bu bölümde I/O birimleri, dosya sistemi, kullanıcı programları ve diğer konular hakkında bilgi verilmektedir. Bu bölümde bir örnek programın geliştirilmesi ve çalıştırılması hakkında bilgi verilmektedir. Bu bölümde bir örnek programın geliştirilmesi ve çalıştırılması hakkında bilgi verilmektedir.

BEŞİNCİ BÖLÜM

CP/M İŞLETİM SİSTEMİ

GİRİŞ VE ÇIKIŞ İŞLEVLERİ

1. GİRİŞ

Bu bölümde sistem organizasyonu, belleğin yapısı ve sistem giriş noktaları belirtilmiştir. CP/M mantıksal olarak dört parçadan oluşmuştur. Bunlar, temel I/O sistemi (BIOS) Temel disk işletim sistemi, (BDOS) Konsol komut işleyici (CCP) ve geçici program alanıdır. (TPA). BIOS donanıma bağlı modüldür. Bu bölümde I/O noktaları belirtilmiştir. BIOS ve BDOS mantıksal olarak tek bir modül içinde ortak bir giriş noktası ile birleştirilmiş bu FDOS olarak isimlendirilmiştir. TPA, FDOS ve CCP alanlarından ar-ta kalan boş bellek bölgesidir. İşletim sistemine ait çeşitli komutlar ve kullanıcı programları burada yürütülür. Belleğin en alt bölgesi sistem parametreleri ve bölgesi için ayrılmıştır. CP/M in bellek organizasyonu aşağıda gösterildiği gibidir.

FFFFH	FDOS (BDOS+BIOS)
FBASE	
CBASE	
TBASE	
0000H	CCP
	TPA
	Sistem Parametreleri

Bütün standart CP/M uyarlamalarında BOOT programı rastgele erişimli belleğin tabanına 0000H adresine yerleştirilmiştir. FDOS'un giriş noktası 0005H de verilmiştir. 0006H ve 0007H adresleri sıçrama adresini içerirler. Geçici programlar TPA'nın içine yüklenirler ve yürütürler. Her komut satırı aşağıdaki komut satırlarından biriyle gösterilir.

Komut

Komut Kütük 1

Komut Kütük 1 Kütük 2

"Komut" olarak DIR veya TYPE gibi kalıcı komutlar veya geçici komut ismi veya program isimlerinden biri kullanılır. Komut kalıcı bir komut ise

CP/M bu komutu hemen yürütür. Aksi halde CP/M geçerli disk üzerinde komut . COM şeklinde olan bir kütük ismi araştırır. Kütük bulunursa program TPA'nın bazından itibaren (100H) yüklenir ve yürütülür. Komut bir veya iki kütük ile meydana getirilmişse CCP sistem parameteleri alanında bir veya iki tane kütük kontrol bloğu (FCB) düzenler.

Geçici program alanı kontrolü CCP'den alarak FDOS'un I/O olanaklarını kullanarak programı yürütmeye başlar. Geçici program CCP'de isimlendirilir. Geçici program CP/M'in I/O olanaklarını kullanarak işletim konsolu ve çevre düzenleri ve disk sürücü birimi ile iletişim sağlayabilir. I/O sistemine bir işlev numarası ve bilgi adresi kullanılarak erişilebilir.

2. İŞLETİM SİSTEMİ ÇAĞRI DÜZENLERİ

Aşağıda kullanıcı programlarında direkt olarak çağrılan işletim sistemine ait işlevler verilmiştir. CP/M olanakları genel olarak iki katedride incelenmiştir. Bunlardan birincisi basit I/O düzeni, diğeri disk kütükleri I/O düzenleridir. Genel olarak işlev numarası C yazmacına ve bilgi adresi çift bayt olarak DE yazmaç çiftine yüklenir. Tek baytlı geri dönüş değerleri A yazmacında, çift baytlı geri dönüş değerleri HL yazmaç çiftinde saklanır.

CP/M disket üzerinde bulunan isimlendirilmiş her kütüğü yürütür. Her disket sürücü mantıksal olarak disk rehberi ve kütük veri alanı olmak üzere bağımsız bölgelere ayrılmıştır. Disk kütük isimleri üç kısımdan meydana gelmiştir. Bunlar disk sürücüsünün kodu, kütük ismini belirten sekiz karakterli ve kütük tipini belirten üç karakterli bölümlerdir. Kütük tipi isimleri genel olarak kütüklerin belirli tiplerini gösterir.

ASM	Çevrilecek kaynak kütük
PRN	Adres veri ve kaynak programı içeren kütük
HEX	Makina kodlarında "hex" kayıtları içeren kütük
BAS	"BASIC" kaynak kütük
COM	Hemen yürütülebilen komutları içeren kütük
REL	Yeri değiştirilebilen kütük
BAK	ED "Backup" yedek kaynak kütük
\$\$\$	Geçici kütük

Kaynak kütükler ASCII karakterlerin birbiri arkasına eklenmesi ile ve <cr><lf> ile sonlandırılmış satırlardan oluşmuştur. 128 baytlık CP/M kayıtları kaynak metnin çeşitli satırlarını içerir. Bir ASCII kütüğünün sonu bir kontrol-Z karakteri (1AH) ile belirtilir. Kontrol-Z karakteri makina kodu kütüklerinin (COM) içine yerleştirilmiştir.

CP/M kütükleri herbiri 128 baytlık 65536 kayıda yerleştirilebilir. Bu kayıtlar 0 dan 65535 sayısına kadar numaralandırılmışlardır. Yani bir kütük maksimum olarak 8 Megabaytlık bir bölümü işgal edebilir.

Kütük işlemleri, işlev numarası 15 ile başlar. DE yazmaç çifti genellikle bir kütük kontrol bloğunu adreslemek için kullanılır. Geçici programlar ve basit kütük işlemleri için kütük kontrol bloğu 005CH adresinden başlamaktadır. Bütün kütük işlemleri için kullanılan 128 baytlık kayıt temel bir birimdir.

FCB (kütük kontrol bloğu) alanı ardışıl erişim için, ardışıl 33 bayt ve rastgele erişim içinse 36 baytı içerir. FCB normal olarak 005CH adresine rastgele erişimli kütüklerin kullanımı için yerleştirilmiştir. Son üç bayt 6DH adresinden başlamaktadır. FCB (kütük kontrol bloğu)nun şekli aşağıdaki gibidir.

dr	f1	f2	./	f8	t1	t2	t3	ex	S1	S2	rc	d0	./	dn	cr	r0	r1	r2
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

00 01 02 .. 08 09 10 11 12 13 14 15 16 .. 31 32 33 34 35

dr (drive) Disket sürücüsünün kodu

0 : kütük için sürücü kodu ihmal edilmiş

1 : A disket sürücüsü

2 : B disket sürücüsü

16: P disket sürücüsü

f1...f8 ASCII büyük harflerden oluşan kütük ismini içerir. En yüksek değerli bitleri 0 dır.

t1,t2,t3 ASCII büyük harflerden oluşan kütük tipini gösterirler. ASCII karakterlerin en yüksek değerli bitleri 0 dır. Bu t1',t2' ve t3' bitleri kütüğün durumunu belirtirler.

t1'=1 yalnız okunabilir kütük

t2'=1 (SYS) sistem kütüğü

ex Geçerli genişletme numarasını içerir. Her madde kullanıcı tarafından 0 a set edilmiştir. Fakat kütüğün (I/O) giriş/çıkış işlemleri esnasında değeri 0 ilâ 31 arasında değişir.

S1 Sistem kullanımı için ayrılmıştır.

S2 Dahili sistem kullanımı için ayrılmıştır. OPEN, MAKE, SEARCH işlemlerinde sıfıra "set" edilmiştir.

rc (record count) Genişletme için kayıt sayısını içerir. 0-127 arasında değer alır.

d0...dn CP/M tarafından doldurulur. Sistem kullanımı için ayrılmıştır.

cr Ardışıl kütük işlemlerinde okunan veya yazılan geçerli kayıtlı gösterir. Kullanıcı tarafından normalde 0 a "set" edilmiştir.

r0,r1,r2 0-65535 değerleri arasında isteğe bağlı kayıt numarasını verirler. 16 bitlik değer yüksek değerli baytı r1, düşük değerli baytı r0 ve taşma (overflow) r2 de temsil edilmektedir.

CP/M içinde her üretilen kütük bir kütük kontrol bloğuna (FCB) sahiptir. Bu blok bütün kütük işlemleri için kütük hakkında bütün bilgileri taşır. Kütüklere erişildiğinde FCB nin ilk 16 baytını doldurmak ve "cr" alanını başlatmak programcının sorumluluğundadır. Normal olarak 1 den 11 e kadar olan bölge kütük ismini ve tipini belirten ASCII değerler ile doldurulur.

FCB, kütük kontrol blokları disk üzerinde rehber (directory) alanında depo edilirler. Ve programcı kütük işlemlerine geçmeden önce bellek içine getirilirler.

FCB nin belleğe aktarılması ve kütük üzerindeki işlemler ve düzeltmelerden sonra FCB yeni değerler alarak diskete kaydedilir.

CCP komut satırında verilmiş olan iki kütüğün, kütük kontrol bloklarını 005CH adresinden itibaren yerleştirir. İkinci FCB ilk FCB'nin d0-dn kısmında oluşturulur.

CP/M'le ilgili işlevler aşağıda geniş bir şekilde verilmiştir.

İşlev 0 : sistemin "reset" edilmesi

Giriş parametreleri:

C Yazmacı : 00H

Sistem "reset" işlevinden sonra kontrol CP/M işletim sistemine (CCP ye) döner. CCP A disketini seçerek CP/M'i tekrar yükler. Bu işlev tam tamına BOOT işleminin etkisini gösterir.

İşlev 1: Konsol girişi işlevi

Giriş Parametreleri:

C Yazmacı : 01H

Çıkış Parametreleri:

A Yazmacı : ASCII karakter

Konsol giriş işlevi, konsoldan girilen karakteri A yazmacına yerleştirir. Grafik karakterler, <cr><lf> ve "backspace" (Ctl-H) kontrol karakterleri kullanılarak klavyeden girilir. Tab karakteri (Ctl-I) gelecek tab işlevinde durur. Ctl-S ve Ctl-P komutları, ekranın ve yazıcının kontrol edilmesini sağlar.

İşlev 2: Konsol çıkış işlevi

Giriş Parametreleri:

C Yazmacı : 02H

E Yazmacı : ASCII karakter

E yazmacındaki ASCII karakter konsol düzenine gönderilir. 1. işlev gibi tab, yazıcı ve ekran durdurma ve başlatma işlevleri kontrol edilir.

İşlev 3: Okuyucu (Reader) girişi

Giriş Parametreleri:

C Yazmacı : 03H

Çıkış parametreleri:

A Yazmacı : ASCII karakter

Okuyucu giriş işlevi A yazmacı içine okuyucu düzeninden gelecek karakterleri yerleştirir. Karakter okunana kadar bu işlev devam eder.

İşlev 4: Delici (Punch) çıkış işlevi

Giriş Parametreleri:

C Yazmacı : 04H

E Yazmacı : ASCII karakter

Delici çıkışı işlevi, delici düzenine E yazmacının içindeki karakterleri gönderir.

İşlev 5: Yazıcı çıkışı (List output)

Giriş parametreleri:

C Yazmacı : 05H

E Yazmacı : ASCII karakter

Yazıcı çıkışı işlevi E yazmacı içindeki ASCII karakteri listeleme düzeyine aktarır.

İşlev 6: Direkt konsol giriş/çıkışı

Giriş parametreleri:

C Yazmacı : 06H

E Yazmacı : 0FFH giriş için veya çıkış için karakter yerleştirilecek.

Çıkış parametreleri:

Yazmaç A : Karakter veya durum (status)

Direkt konsol giriş/çıkışı işlevi CP/M altında çeşitli konsol giriş/çıkış uygulamaları isteği üzerine desteklenmiştir. Genel olarak bu işlevin kullanımında, CP/M in normal kontrol karakterlerinin kullanımından kaçınılmıştır. (Ctl-S ve Ctl-P gibi)

İşlev 6 ya girildiğinde, E yazmacı bir konsol girişi isteğinde heksadesimal FF veya bir ASCII karakteri içerir. Giriş değeri FF ise ve karakter hazır değilse A yazmacında 00 değeri belirir. Diğer yandan A yazmacı konsoldan gelecek giriş değerini içerir. E yazmacında FF karakteri yoksa, E yazmacına geçerli ASCII karakter yerleştirilerek konsol düzenine gönderilir. İşlev 6 diğer giriş/çıkış işlevleriyle bağlantıda kullanılmamalıdır.

İşlev 7: Giriş/çıkış baytının alınması

Giriş parametresi :

C Yazmacı : 07H

Çıkış parametresi :

A Yazmacı : Giriş/çıkış baytının değeri

Bu işlevde giriş/çıkış baytının değeri A yazmacında bulunur.

İşlev 8: Giriş/çıkış baytının yerleştirilmesi

Giriş parametreleri:

C Yazmacı : 08H

E Yazmacı : Giriş/çıkış baytının değeri

Bu işlev E yazmacında verilen değeri giriş/çıkış baytına yerleştirir.

İşlev 9: Yazısalın ekrana bastırılması (Print string)

Giriş parametreleri:

C Yazmacı : 09H

DE Yazmacı : Yazısalın adresi

Bu işlev bellekte bulunan ve adresi DE yazmacında bulunan karakter dizisini bir "\$" işareti ile karşılaşıncaya kadar ekrana çıkartır. Tab işlevi, ekranı ve yazıcıyı durdurma ve başlatma işlevlerinin kontrolü yapılır.

İşlev 10: Konsol tampon belleğinin okunması

Giriş parametreleri: Tampon bellek içindeki konsol karakterleri

Bu işlev adresi DE yazmacında verilen biçimlendirilmiş konsol girişinin bir satırını belleğe aktarır. Konsol girişi, giriş tamponunun taşması, bir <cr> veya bir <lf> işleminin uygulanması ile sonlandırılır. Okunan tampon bellek aşağıdaki şekli alır.

DE: +0 +1 +2 +3 +4 +5 +6 +7 +8 .. +n

mx	nc	c1	c2	c3	c4	c5	c6	c7	/	??
----	----	----	----	----	----	----	----	----	---	----

mx Tampon bellekte tutulacak karakterlerin maksimum sayısını (1 den 255 e kadar)

nc Okunan karakterlerin sayısı

ve bunların izleyenlerde konsoldan okunan karakterlerdir.

nc<mx ise üstteki şekilde görüldüğü gibi son karakter "??" işaretleri ile belirtilir. Kontrol işlevlerinin bir çoğu satır biçimleme esnasında kabul edilir.

İşlev 11: Konsol durumunun (Status) belirlenmesi

Giriş parametresi :

C Yazmacı : ØBH

Çıkış parametresi :

A Yazmacı : Konsolun durumu

Konsol durumunu görmek için kullanılan bu işlev konsoldan girilmiş bir karakteri kontrol eder. Bir değer sıfırdan büyükse A yazmacı içine döner. Aksi halde A yazmacı içinde sıfır değeri bulunur.

İşlev 12: Uyarlama numarasının belirlenmesi

Giriş parametreleri:

C Yazmacı : ØCH

Çıkış parametreleri:

HL Yazmacı : Uyarlama numarası

12 numaralı işlev CP/M programlarının uyarlama numarasını verir. Uyarlama numarası HL yazmacı içinde gözüktür. H yazmacı ØØ ise bu uyarlama CP/M'e göre, Ø1 ise MPM'e göre gerçekleştirilmiştir. (MP/M programı CP/M işletim sisteminin çok terminalli türüdür.) L yazmacı içindeki değer CP/M için uyarlama numarasını verecektir.

L=ØØ ise 2.0 uyarlamasından önceki uyarlamaları kapsamaktadır. 2.0 uyarlaması L yazmacı içine heksadesimal 20 olarak yazılmaktadır, bu işlem

21,22 ve 2F e kadar gidebilmektedir. Kullanıcı 12. işlevi kullanarak sıralı veya rastgele erişimli işlevleri sağlayan uygulama programlarını yazabilir.

İşlev 13: Sistem diskinin "reset" edilmesi

Giriş parametreleri:

C Yazmacı : ØDH

Disk reset işlevi programlı olarak disk sistemini reset etmek için kullanılır. Bu işlev bir uygulama programı vasıtasıyla sistem "reboot" edilmeden diskler değiştirilir:

İşlev 14: Disklerin seçimi

Giriş parametreleri:

C Yazmacı : ØEH

E Yazmacı : Seçilecek diskin kodu

Disklerin seçilmesi işlevinde, seçilecek diskin E yazmaçına yerleştirilmesi gerekir. E=0 ise A E=1 ise B disketi sürücüsü ve 15 kadar çeşitli disket sürücüsü söz konusudur. 16 sürücülü sistem olarak Ø dan 15 e kadar sürücü kodları E yazmacının içine yerleştirilir.

İşlev 15: Kütük açma işlevi (openfile)

Giriş parametreleri:

C Yazmacı : ØFH

DE Yazmacı : FCB (Kütük kontrol bloğu) adresi

Çıkış parametreleri:

A Yazmacı : Rehber (Directory) kodu

Kütük açma işlevi, geçerli aktif kullanıcı numarası için disk rehberinde geçerli olarak oluşturulan bir kütüğü aktif hale getirmek için kulla-

nılır. FDOS inceden inceye DE yazmaç çiftinde adresi verilen kütük kontrol bloğunun 1 den 14 e kadarki bölümünün disk rehberindeki FCB'lere uyup uymadığını inceler. (s1 baytı otomatik olarak sıfırlanır.) Normal olarak FCB'nin(kütük kontrol bloğunun) "ex" ve "S2" baytları sıfırlanır ve ASCII soru işareti (3FH) eklenmez. Bir rehber elemanı benzer olarak bulunmuşsa ilgili rehber bilgisi FCB nin d0 dan dn e kadar olan baytlarının içine aktarılır. Böylece ardışıl okuma ve yazma işlevleri için kütüğe erişim sağlanmış olur.

Kullanıcı başarılı bir açma işlemi tamamlanıncaya kadar bir çıkış kütüğüne ilâveler yapabilir. Dönüşte kütük açma işlemi başarılı ise A yazmacında 0 ile 3 değerleri arasında rehber kodu oluşur. Eğer A yazmacında FF (255 desimal) değeri var ise kütük bulunamamıştır. FCB'nin içinde soru işaretleri oluşursa ilk karşılaşılan FCB aktif hale getirilir. Birinci kayıtda kütüğe ardışıl olarak erişilebiliyorsa program tarafından geçerli kayıt "cr" sıfırlanmalıdır.

İşlev 16: Kütük kapatma işlevi (Close file)

Giriş parametreleri:

C Yazmacı : 10H

DE Yazmacı : FCB (kütük kontrol bloğu) adresi

Çıkış parametreleri:

A Yazmacı : Rehber kodu

Kütük kapatma işlemi, kütük açma işlevinin tersi olarak geliştirilmiştir. Bir önceki açma (open) veya gerçekleştirme (make) işlevlerinde, FCB adresi, DE yazmaç çifti ile verilen kütük; yeni FCB ile birlikte disk rehberine kaydedilir. Başarılı bir kapatma işleminde A yazmacında rehber kodu 0,1,2,3 olarak belirir. A yazmacında 0FFH (255 desimal) varsa reh-

berde kütük ismi bulunamamıştır. Yalnız okuma işlemleri olduğunda bir kütük kapatılmaya gerek duymaz. Yazma işlemi oluştuğunda, kapatma işlevi yeni rehber bilgisinin kaydı için zorunludur.

İşlev 17: Verilen FCB için rehberi araştırma işlevi

Giriş parametreleri:

C Yazmacı : 11H

DE Yazmacı : FCB (Kütük kontrol bloğu) adresi

Çıkış parametreleri:

A Yazmacı : Rehber Kodu

Bu işlev FCB adresi DE yazma çifti ile verilen bir kütüğün bir benzeri için rehberi inceden inceye araştırır. A yazmacındaki değer kütüğün bulunup bulunmadığını gösterir. Şöyleki işlevin yürütülmesinden sonra A yazmacı 0FFH değerine sahip ise kütük bulunamamıştır. Aksi halde 0,1, 2,3 değerlerine sahipse kütük rehberde bulunmaktadır.

İşlev 18: Sonuncu FCB'yi arama işlevi

Giriş parametreleri:

C Yazmacı : 12H

Çıkış parametreleri:

A Yazmacı : Rehber kodu.

Bu işlev sonuncu rehber girişinin rehberde incelenmesinden başka 17. işlevin benzeridir. 17. işleve benzer olarak, 18. işlev A yazmacından 255 desimal değeri ile döndüğünde ayrıntılar rehberde bulunmamaktadır demektir.

İşlev 19: Kütük silme işlevi

Giriş parametreleri:

C Yazmacı : 13H

DE Yazmacı : FCB adresi

Çıkış parametreleri:

A Yazmacı : Rehber kodu.

Kütük silme işlevi adresi DE yazmaç çifti ile verilen kütükleri ortadan kaldırır. Kütük ismi ve tipi belirsiz (ambiguous) kayıtları içerebilir. (Örneğin soru işareti çeşitli pozisyonlarda olabilir.) Fakat sürücü seçici kodu arama ve sonrakini arama işlevleri gibi belirsiz değildir.

19. işlevde A yazmacında, belirtilen kütük bulunmuyorsa 255 desimal değeri, bulunuyorsa 0 ilâ 3 arasında bir değer oluşur.

İşlev 20: Ardışıl okuma işlevi

Giriş parametreleri:

C Yazmacı : 14H

DE Yazmacı : FCB adresi

Çıkış parametreleri:

A Yazmacı : Rehber kodu

DE yazmaç çifti ile FCB adresi verilen bir kütük açma veya oluşturma işlevlerinden (15. ve 22. işlevler) sonra ardışıl okuma işlevi sonraki 128 bayt kaydı kütükten okur ve geçerli DMA adresindeki bellek içine yerleştirir. Kayıt boyutun geçerli kayıt pozisyonunda ("cr" alanında yazan) okunur ve "cr" alanı otomatik olarak sonraki kayıt pozisyonuna atılır. "cr" alanında taşma olursa bir sonraki mantıksal boyut (extent) otomatik olarak açılır ve "cr" alanı sıfıra "reset" edilir. (Bir sonraki okuma işlemi için.) A yazmacının içinde 00H değeri oluşmuşsa okuma işlemi başarılı olmuştur. A yazmacında sıfırdan farklı bir değer varsa bir sonraki kayıt pozisyonunda veri oluşmamıştır. (Kütüğün sonuna gelinmiştir.)

İşlev 21: Ardışıl yazma işlevi

Giriş parametreleri:

C Yazmacı : 15H

DE Yazmacı : FCB (kütük kontrol bloğu) adresi

Çıkış parametreleri:

A Yazmacı : Rehber kodu.

DE yazma çifti tarafından adresi verilen ve FCB tarafından isimlendirilmiş kütüğü diskete kaydeder. Kayıt kütüğün geçerli kayıt numarası "cr" na göre yerleştirilir ve otomatik olarak bir sonraki kayıt pozisyonuna arttırılır. "cr" alanı taşarsa bir sonraki mantıksal boyut göstergesi otomatik olarak açılır ve "cr" alanı bir sonraki yazma işlemi için sıfıra çekilir. A yazmacında 00H değeri yazma işleminin başarılı olduğunu belirtir. A yazmacında sıfırdan farklı bir değer oluşması yazma işleminin başarılı olmadığını ve disketin tamamen dolu olduğunu belirtir.

İşlev 22: Kütük oluşturma işlevi (Make file)

Giriş parametreleri:

C Yazmacı : 16H

DE Yazmacı : FCB adresi

Çıkış parametreleri:

A Yazmacı : Rehber kodu.

Kütük oluşturma işlemi kütük açma işleminin bir benzeridir. Yalnız burada farklı olarak FCB de verilen kütük ismi rehberde oluşmamıştır. FDOS (işlevsel disk işletim sistemi) bir kütük oluşturur. Ancak programcı çift isim oluşmadığından emin olmalı ve öncelikli olarak bir silme işlemi uygulamalıdır. A yazmacında 0,1,2 veya 3 değerleri varsa işlem başarılıdır. Aksi halde 0FFH (255 desimal) değeri rehberin dolu olduğunu göstermektedir.

İşlev 23: İsim değiştirme işlevi

Giriş parametreleri:

C Yazmacı : 17H

DE Yazmacı : FCB adresi

Çıkış parametreleri:

A Yazmacı : Rehber kodu

Bu işlev FCB adresi DE yazmacı tarafından verilen kütük isimlerinin bütün oluşumlarının ilk 16 baytını ikinci 16 bayt ile değiştirir. Sürücü kodu "dr" Ø pozisyonundadır. Ve sürücü seçmek için kullanılır. Yeni 16 lık kısımda dr nin yerine Ø yerleştirilir. İşlemden sonra A yazmacında Ø ile 3 arasında bir değer bulunuyorsa işlem başarılı olmuştur. Aksi halde 255 desimal değeri rehberin taranması sonucunda ilk kütük isminin bulunmadığını gösterir.

İşlev 24: Sisteme bağlı olan disket sürücülerin belirlenmesi

Giriş parametreleri:

C Yazmacı : 18H

Çıkış parametreleri:

HL Yazmacı : "Log-in vector" değeri

"Log-in vector" değeri CP/M tarafından HL yazmaç çifti içine yerleştirilir. L yazmacının en düşük ağırlıklı biti ilk sürücü A nın yerini tutar ve H yazmacının en yüksek biti P olarak etiketlenen 16.sürücünün yerini tutar. Bitin Ø olması sürücünün hatta bağlı olmadığını göstermektedir. 1 ise kesin bir disk sürücü seçiminde aktif olunacağını göstermektedir. Bu işlemde dönüşte A yazmacı da L yazmacının değerini göstermektedir.

İşlev 25: Geçerli disk sürücünün belirlenmesi

Giriş parametresi:

C Yazmacı : 19H

Çıkış parametresi :

A Yazmacı : Geçerli disk sürücüsünün kodu.

Bu işlev A yazmacı üzerine geçerli disk numarasını yerleştirir. Disk numaraları A dap P ye kadarki isimlerin yerlerini tutan 0-15 arasındaki sayılarla belirtilmiştir.

İşlev 26: DMA adresinin yerleştirilmesi

Giriş parametreleri:

C Yazmacı : 1AH

DE Yazmacı : DMA adresi

DMA "Direct Memory Access" kelimelerinin kısaltılmışıdır. Bellek içinde direkt olarak disk kontrol birimine bağlanmıştır. Gerçi birçok bilgisayar sistemlerinde DMA erişim sistemi yoktur. (Veri aktarımı programlı I/O işlemleriyle almaktadır.) CPM'de DMA adresi bir diske yazma işleminin önce ve bir diski okuma işleminden sonra 128 bayt veri kayıtları olarak öngörülen adrese gelir. "Cold start", "warm start" ve disk sisteminin "reset" edilme komutları geldiğinde DMA adresi otomatik olarak 0080H adresine yerleştirilir. Bu adres yokluk halinde seçilen DMA adresidir. DME adresini yerleştirme işlevi, belleğin diğer kısımlarında bulunan veri kayıtlarının adreslerini belirtmek için kullanılır. Böylece "cold start", "warm start" veya disk sisteminin "reset" edilmesinden sonra DE yazma-
cında belirtilen değere DMA adresi yerleştirilir.

İşlev 27: "Alloc" adresinin elde edilmesi

Giriş parametreleri:

C Yazmacı : 1BH

Çıkış parametreleri:

HL Yazmacı : "Alloc" adresi

Bir "allocation vector" tahsis vektörü, her hatta bağlı disk sistemi için ana bellek içinde muhafaza edilir. Çeşitli sistem programları hakkında bilgi ve kalan boş yerlerin miktarı tahsis vektörünün (allocation vector) gözden geçirilmesi ile sağlanır. Tahsis bilgisi diskin yalnız okunabilir olması şeklinde geçerlidir. Ancak bu işlev uygulama programlarında pek kullanılmaz.

İşlev 28: Disketin yazmaya karşın korunması

Giriş parametreleri:

C Yazmacı : 1CH

Bu işlev geçerli olarak seçilmesi disket sürücüsü üzerindeki disketi geçici olarak yazmaya karşı korur. Gelecek "cold" veya "warm" start işlemlerine kadar herhangi bir yazma komutu geldiğinde ekranda

Bdos Err on d: R/0

mesajı belirir. "d" burada kullanılan disket sürücü ismidir.

İşlev 29: "READ/ONLY" vektörünün elde edilmesi

Giriş parametresi:

C Yazmacı : 1DH

Çıkış parametresi :

HL Yazmacı : R/0 vektörünün değeri

29. işlev geçici olarak R/0 işlevine sahip olan disket sürücülerini gösterir. 24. işlev gibi HL yazma çiftinin en az ağırlıklı biti A disket sürücüsünü en çok ağırlıklı biti P disket sürücüsünün yerini tutar. R/0 biti açık olarak 28. işlevi çağırmakla veya CP/M içine otomatik yazılım mekanizmasıyla yerleştirilir.

İşlev 30: Kütük niteliklerinin yerleştirilmesi

Giriş parametreleri:

C Yazmacı : 1EH

DE Yazmacı : FCB (Kütük kontrol bloğu) adresi

Çıkış parametreleri:

A Yazmacı : Rehber kodu

Bu işlev kütüğe iliştirilen sürekli göstergelerin programlı olarak idaresini sağlar. Bilhassa, R/O ve sistem nitelikleri (t1' ve t2') "set" yada "reset" edilebilirler. DE yazmacı çifti nitelikleri uygun olarak "set" yada "reset" edilmiş belirli kütük ismini adresler. 30. işlev bir benzer rehber girişi araştırır ve seçilmiş göstergelerine benzeyen rehber girişlerini değiştirir. f1'den f4'e kadar olan göstergeler geçerli olarak kullanılmaz, fakat genellikle uygulama programları içinde kullanılırlar. f5'den f8' ve t3' ne kadar olan göstergeler gelecek sistem genişletmeleri için ayrılmıştır.

İşlev 31: Disk parametreleri adresinin elde edilmesi

Giriş parametreleri:

C Yazmacı : 1FH

Çıkış parametreleri:

HL Yazmacı : DPB (Disk parametreleri bloğu) adresi

BIOS'a yerleşmiş disk parametreleri bloğu adresi bu işlevin çağrılmasıyla HL yazmacı çifti içine yerleşir. Bu adres iki amaçtan birisi için kullanılır. Bunlar, disk parametre değerlerinin gösterilmesi ve boş bölgele-
rin hesabı amacıyla veya geçici programların disk çevre şartları değiştiğinde geçerli disk parametre değerlerinin dinamik olarak değiştirilmesidir. Normal olarak uygulama programları buniteliklere ihtiyaç göstermezler.

İşlev 32: Kullanıcı (User Code) kodunun yerleştirilmesi/elde edilmesi

Giriş parametreleri:

C Yazmacı : 20H

E Yazmacı : 0FFH kullanıcı kodunun (get) elde edilmesi veya yerleştirilmesi (set)

Çıkış parametreleri:

A Yazmacı : Geçerli kod veya değer yok.

Bir uygulama programı geçerli aktif kullanıcı (user) numarasını 32. işlevi çağırmasıyla değiştirebilir veya öğrenebilir. E yazmacı 0FFH değerine sahipse A yazmacında aktif geçerli kullanıcı (user) numarası görülür. Bu 0 ilâ 15 değerleri arasındadır. E yazmacında 0FFH değeri yoksa E yazmacına geçerli kullanıcı numarası yerleştirilmiş demektir.

İşlev 33: Rastgele okuma işlevi (Read Random)

Giriş parametreleri:

C Yazmacı : 2IH

DE Yazmacı : FCB adresi

Çıkış parametreleri:

A Yazmacı : Dönüş kodu (Returned Code)

Rastgele okuma işlemi, FCB (kütük kontrol bloğu) den sonraki 3 baytta beliren kayıt numarası hariç ardışıl okuma işlevinin bir benzeridir. Rastgele okuma ile ilgili bilgileri kullanıcı 24 bitte bildirir. Burada 33. bayt r0 en az ağırlıklı bayt, 35. bayt r2 en yüksek ağırlıklı bayttır. CP/M r2 baytını bir kütüğün baytının hesabında geçerli kayıt olarak görmez. r2 baytı sıfır olmalıdır. Bununla beraber sıfırdan farklı bir değer, taşma olduğunu gösterir ve kütüğün sonu olduğunu bildirir.

Böylece rØ ve rl bayt çifti okunacak kayıtları içeren bir çift bayt veya "word" değeri gibi davranır. Bu değer Ø ilâ 65535 değerleri arasında 8 Megabaytlık bir kütüğün herhangi bir bölgesine erişimi sağlar. Bir kütük rastgele erişim işlemi kullanıldığında temel genişletme Ø ile açılmalıdır. Seçilen kayıt numarası rastgelekayıt alanına (rØ,rl) yerleştirildikten sonra BDOS bu kayıtları okumak için çağrılır. BDOS'a gidilir. BDOS dan döndüğünde A yazmacı içinde aşağıdaki listedeki hata kodlarından birini veya işlem başarılı ise ØØ kodunu içerir. Sonuncu durumda geçerli DMA adresi rastgele olarak erişilebilen kayıtları içerir. Her rastgele okuma işleminde mantıksal genişletme ve kayıt değerleri otomatik olarak "set" edilir. Böylece kütük ardışıl olarak okunabilir veya yazılabilir. Kütük bu durumda rastgele olarak erişilebilen geçerli pozisyonlardan başlar. Kullanıcı bir şeyler ilâve etmek istediğinde, sonuncu rastgele olarak okunan kayıt, rastgele moddan bir ardışıl okumaya değiştirilmiş gibi davranacaktır. Bir rastgele okuma işleminde A yazmacı içinde beliren hata kodları şunlardır.

- Ø1 yazılmamış veri okunması
- Ø2 rastgele moda dönülmemiş
- Ø3 geçerli boyut kapalı değil
- Ø4 yazılmamış boyuta bakılıyor
- Ø5 okuma moduna dönülmemiş
- Ø6 diskin fiziksel sonuna gelinmiş.

Ø1 ve Ø4 hata kodları, bir rastgele okuma işleminde bir veri bloğuna erişilmek istendiğinde bu blok öncelikli olarak yazılmamış ise veya bir boyut üretilmemiş ise veya benzeri koşullarda oluşurlar.

Ø3 hata kodu birincil sistem işlemlerinde normal olarak pek oluşmaz. Bu hata kodu oluşmuşsa bu hata kodu basit olarak tekrar okuma işlemi ile

veya tekrar açma işlemi ile boyutu sıfıra getirip silinebilir. 06 hata kodu oluşmuşsa, 2.0 ve bunun üstündeki uyarlamalar için r2 baytında sıfırdan farklı bir değer var demektir. Normal olarak sıfırdan farklı bir değer kayıp veri gibi davranabilir. A yazmacındaki sıfır kodu işlemin komple olarak bitirildiğini gösterir.

İşlev 34: Rastgele yazma işlevi

Giriş parametreleri:

C Yazmacı : 22H

DE Yazmacı : FCB adresi

Çıkış parametreleri:

A Yazmacı : Dönüş kodu.

Rastgele yazma işlemi, geçerli DMA adresindeki verinin diske yazılması hariç rastgele okuma işleminin aynısıdır. Rastgele okuma işlemi gibi, rastgele kayıt numarası yazmanın bir sonucu olarak değiştirilmez. FCB nin mantıksal genişletme (extent) numarası ve geçerli kayıt pozisyonları yazımın başında rastgele kaydın yerini tutmak üzere "set" edilirler.

Tekrar, ardışıl okuma ve yazma işlemleri geçerli olarak adreslenen kaydı izleyen bir rotasyonla başlayabilir. Hata kodları rastgele yazma işlemine benzer olarak A yazmacında görülebilirler. Farklı olarak 05 hata kodu rehberin dolmasıyla yeni bir genişletmenin üretilmediğini gösterir.

İşlev 35: Kütük boyutunun hesaplanması

Giriş parametreleri:

C Yazmacı : 23H

DE Yazmaç çifti : FCB adresi

Çıkış parametreleri: Rastgele kayıt alanının yerleştirilmesi

Bir kütüğün boyutunun hesaplanmasında DE yazmaç çifti bir FCB yi "random

mode" şeklinde (r0,r1 ve r2 baytlarını da içeren) adresler FCB rehberin taramasında kullanılan belirli bir kütük ismini içerir.

Dönüşte, rastgele erişim baytları "gerçek" kütük boyutunu içerirler.

35. işlevin yerine getirilmesinden sonra yüksek kayıt baytı r2 de 01 değeri varsa kütük maksimum kayıt sayısı olan 65535 değerini içerir. r1 ve r0 baytları kütük boyutunu içeren 16 bitlik bir değer gösterirler. (r0 önceki gibi en az ağırlıklı bayttır.)

Veri basit olarak 35. işlevin çağrılmasıyla bir çıkış kütüğünün sonuna rastgele kayıt pozisyonunun yerleştirilmesiyle eklenebilir. Kütük ardışıl olarak yazıldığında bir kütüğün gerçek boyutu fiziksel boyutunun yerini tutar.

İşlev 36: Rastgele kaydın yerleştirilmesi

Giriş parametreleri:

C Yazmaçısı : 24H

DE Yazmacısı : FCB (Kütük kontrol bloğu) adresi

Çıkış parametreleri: Rastgele kaydın yerleştirildiği alan

Rastgele kaydın yerleştirilmesi işlevi, otomatik ve ardışıl olarak okunabilen veya yazılabilen bir kütükten rastgele kayıt pozisyonunun meydana getirilmesine neden olur.

İşlev 37: Disk sürücünün "reset" edilmesi

Giriş parametreleri:

C Yazmacısı : 25H

DE Yazmacısı : Sürücü vektörü

Çıkış parametreleri:

A Yazmacısı : 00H

Disk sürücüsünün "reset" edilmesi işlevi belirtilmiş sürücülerin "reset" edilmesini sağlar. En düşük ağırlıklı bit A sürücüsünü göstermek üzere 16 bitlik "reset" vektörü DE yazmacına yüklenir. Bu işlevden sonra A yazmacındaki 00 değeri işlemin başarılı olduğunu gösterir.

İşlev 40: Belirtilmiş alanı sıfır ile doldurarak rastgele yazma işlevi

Giriş parametreleri:

C Yazmacı : 28H

DE Yazmacı : FCB adresi

Çıkış parametreleri:

A Yazmacı : Dönüş kodu

Bu işlev, 34. işlevin bir benzeridir. Ancak burada belirtilmiş alan sıfırlar ile veri yazılmadan önce doldurulur.

ALTINCI BÖLÜM

PROGRAM ÖRNEKLERİ

GIP LISTİNDEN ÖRNEKLER

11				
21				
31	0104		END	1204
41	0105	REBOOT	END	1205
51	0106	ADDS	END	1206
61	0107	CONINS	END	1207
71	0108	CONOUT	END	1208
81	0109	POSTING	END	1209
91	0110	POSTING	END	1210
101	0111	VERSION	END	1211
111	0112	OPEN	END	1212
121	0113	CLOSE	END	1213
131	0114	NAME	END	1214
141	0115	READ	END	1215
151	0116	WRITE	END	1216
161	0117	FILE	END	1217
171	0118	NAME	END	1218
181	0119	RANGE	END	1219
191	0120	NAME	END	1220
201	0121	CR	END	1221
211	0122	LF	END	1222
221				
231	0123	112003	END	1223
241	0124	END	END	1224
251	0125	CD400	END	1225
261	0126	FS00	END	1226
271	0127	021001	END	1227
281	0128	111002	END	1228
291	0129	CD4001	END	1229
301	0130	110000	END	1230

ALTINCI BÖLÜM

PROGRAM ÖRNEKLERİ

PIP LST:=RND.PRINTBUJ

```

1:
2:
3: 0100 ORG 100H
4: 0000 = REBOOT EQU 0000H
5: 0005 = BDOS EQU 0005H
6: 0001 = CONINP EQU 1
7: 0002 = CONOUT EQU 2
8: 0009 = PSTRING EQU 9
9: 000A = RSTRING EQU 10
10: 000C = VERSION EQU 12
11: 000F = OPENF EQU 15
12: 0010 = CLOSEF EQU 16
13: 0016 = MAKEF EQU 22
14: 0021 = READR EQU 33
15: 0022 = WRITER EQU 34
16: 005C = FCB EQU 005CH
17: 007D = RANREC EQU FCB+33
18: 007F = RANOVF EQU FCB+35
19: 0080 = BUFF EQU 0080H
20: 000D = CR EQU 0DH
21: 000A = LF EQU 0AH
22: ;
23: 0100 312303 LXI SP,STACK
24: 0103 0E0C MVI C,VERSION
25: 0105 CD0500 CALL BDOS
26: 0108 FE20 CPI 20H
27: 010A D21601 JNC VERSOK
28: 010D 111B02 LXI D,BADVER
29: 0110 CDDA01 CALL PRINT
30: 0113 C30000 JMP REBOOT
31:
32: 0116 0E0F MVI C,OPENF
33: 0118 115C00 LXI D,FCB
34: 011B CD0500 CALL BDOS
35: 011E 3C INR A
36: 011F C23701 JNZ READY
37: ;
38: ;
39: ;
40: 0122 0E16 MVI C,MAKEF
41: 0124 115C00 LXI D,FCB
42: 0127 CD0500 CALL BDOS
43: 012A 3C INR A
44: 012B C23701 JNZ READY
45: ;
46: 012E 114102 LXI D,NOSPACE
47: 0131 CDDA01 CALL PRINT
48: 0134 C30000 JMP REBOOT
49:
50: 0137 CDE501 CALL READCOM
51: 013A 227D00 SHLD RANREC
52: 013D 217F00 LXI H,RANOVF
53: 0140 3600 MVI M,0
54: 0142 FE43 CPI 'C'
55: 0144 C25601 JNZ NOTQ
56: ;

```


57:	0147 0E10	MVI	C, CLOSEF
58:	0149 115C00	LXI	D, FCB
59:	014C CD0500	CALL	BDOS
60:	014F 3C	INR	A
61:	0150 CA8901	JZ	ERROR
62:	0153 C30000	JMP	REBOOT
63:		NOTQ:	
64:	0156 FE59	CPI	'Y'
65:	0158 C28901	JNZ	NOTW
66:		:	
67:	015B 115C02	LXI	D, DATMSG
68:	015E CDDA01	CALL	PRINT
69:	0161 0E7F	MVI	C, 127
70:	0163 218000	LXI	H, BUFF
71:		RLOOP:	
72:	0166 C5	PUSH	B
73:	0167 E5	PUSH	H
74:	0168 CDC201	CALL	GETCHR
75:	016B E1	POP	H
76:	016C C1	POP	B
77:	016D FE0D	CPI	CR
78:	016F CA7801	JZ	ERLOOP
79:		:	
80:	0172 77	MOV	M, A
81:	0173 23	INX	H
82:	0174 0D	DCR	C
83:	0175 C26601	JNZ	RLOOP
84:		ERLOOP:	
85:	0178 3600	MVI	M, 0
86:		:	
87:	017A 0E22	MVI	C, WRITER
88:	017C 115C00	LXI	D, FCB
89:	017F CD0500	CALL	BDOS
90:	0182 B7	ORA	A
91:	0183 C2B901	JNZ	ERROR
92:	0186 C33701	JMP	READY
93:		NOTW:	
94:	0189 FE4F	CPI	'O'
95:	018B C2B901	JNZ	ERROR
96:	018E 0E21	MVI	C, READR
97:	0190 115C00	LXI	D, FCB
98:	0193 CD0500	CALL	BDOS
99:	0196 B7	ORA	A
100:	0197 C2B901	JNZ	ERROR
101:		:	
102:	019A CDCF01	CALL	CRLF
103:	019D 0E80	MVI	C, 128
104:	019F 218000	LXI	H, BUFF
105:		WLOOP:	
106:	01A2 7E	MOV	A, M
107:	01A3 23	INX	H
108:	01A4 E67F	ANI	7FH
109:	01A6 CA3701	JZ	READY
110:	01A9 C5	PUSH	B
111:	01AA E5	PUSH	H
112:	01AB FE20	CPI	' '
113:	01AD D4C801	CNC	PUTCHR

114:	01B0 E1	POP	H
115:	01B1 C1	POP	B
116:	01B2 0D	DCR	C
117:	01B3 C2A201	JNZ	WLOOP
118:	01B6 C33701	JMP	READY
119:		ERROR:	
120:	01B9 116F02	LXI	D, ERRMSG
121:	01BC CDDA01	CALL	PRINT
122:	01BF C33701	JMP	READY
123:			
124:		GETCHR:	
125:	01C2 0E01	MVI	C, CONINP
126:	01C4 CD0500	CALL	BDOS
127:	01C7 C9	RET	
128:		PUTCHR:	
129:	01C8 0E02	MVI	C, CONOUT
130:	01CA 5F	MOV	E, A
131:	01CB CD0500	CALL	BDOS
132:	01CE C9	RET	
133:		CRLF:	
134:	01CF 3E0D	MVI	A, CR
135:	01D1 CDC801	CALL	PUTCHR
136:	01D4 3E0A	MVI	A, LF
137:	01D6 CDC801	CALL	PUTCHR
138:	01D9 C9	RET	
139:		PRINT:	
140:	01DA D5	PUSH	D
141:	01DB CDCF01	CALL	CRLF
142:	01DE D1	POP	D
143:	01DF 0E09	MVI	C, PSTRING
144:	01E1 CD0500	CALL	BDOS
145:	01E4 C9	RET	
146:		READCOM:	
147:	01E5 119802	LXI	D, PROMPT
148:	01E8 CDDA01	CALL	PRINT
149:	01EB 0E0A	MVI	C, RSTRING
150:	01ED 11E102	LXI	D, CONBUF
151:	01F0 CD0500	CALL	BDOS
152:	01F3 210200	LXI	H, 0
153:	01F6 11E302	LXI	D, CONLIN
154:		READC:	
155:	01F9 1A	LDAX	D
156:	01FA 13	INX	D
157:	01FB B7	ORA	A
158:	01FC C8	RZ	
159:	01FD D630	SUI	'0'
160:	01FF FE0A	CPI	10
161:	0201 D21302	JNC	ENDRD
162:	0204 29	DAD	H
163:	0205 4D	MOV	C, L
164:	0206 44	MOV	B, H
165:	0207 29	DAD	H
166:	0208 29	DAD	H
167:	0209 09	DAD	B
168:	020A 85	ADD	L
169:	020B 6F	MOV	L, A
170:	020C D2F901	JNC	READC

171::199t

```

171: 020F 24          INR      H
172: 0210 C3F901      JMP      READC
173:                  ENDRD:
174: 0213 C630          ADI      '0'
175: 0215 FE41          CPI      'A'
176: 0217 D8            RC
177: 0218 E65F          ANI      101$1111B
178: 021A C9            RET
179:                  BADVER:
180: 021B 504152444F      DB      'PARDON CP/M 2 YE GEREKSINIMINIZ VAR.'
181:                  NOSPACE:
182: 0241 4449534B45      DB      'DISKETTE BOS YER KALMADI. $'
183:                  DATMSG:
184: 025C 5645524959      DB      'VERIYI KAYDEDIN : $'
185:                  ERRMSG:
186: 026F 4841544120      DB      'HATA YAPTINIZ,LUTFEN YENIDEN DENEYIN
187:                  PROMPT:
188: 0298 4B4F4D5554      DB      'KOMUTU BELIRTINIZ YAZ=Y OKU=0 CP/M
189: 02C8 59454E4920      DB      'YENI KOMUTU BELIRTINIZ :$'
190:                  ;
191: 02E1 21          CONBUF: DB      CONLEN
192: 02E2          CONSIZ: DS      1
193: 02E3          CONLIN: DS      32
194: 0021 =          CONLEN EQU      $-CONSIZ
195:                  ;
196: 0303          DS      32
197:          STACK:
198: 0323          END
171: *

```


g0

```

1:  ; FILE DUMP PROGRAM, READS AN INPUT FILE AND PRINTS IN HEX
2:  ;
3:  ; COPYRIGHT (C) 1975, 1976, 1977, 1978
4:  ; DIGITAL RESEARCH
5:  ; BOX 579, PACIFIC GROVE
6:  ; CALIFORNIA, 93950
7:  ;
8:  ; ORG      100H
9:  BDOS      EQU      0005H      ;DOS ENTRY POINT
10: CONS      EQU      1          ;READ CONSOLE
11: TYPEF      EQU      2          ;TYPE FUNCTION
12: PRINTF      EQU      9          ;BUFFER PRINT ENTRY
13: BRKF      EQU      11          ;BREAK KEY FUNCTION (TRUE IF CHAR READY)
14: OPENF      EQU      15          ;FILE OPEN
15: READF      EQU      20          ;READ FUNCTION
16: ;
17: FCB      EQU      5CH          ;FILE CONTROL BLOCK ADDRESS
18: BUFF      EQU      80H          ;INPUT DISK BUFFER ADDRESS
19: ;
20: ; NON GRAPHIC CHARACTERS
21: CR      EQU      0DH          ;CARRIAGE RETURN
22: LF      EQU      0AH          ;LINE FEED
23: ;
24: ; FILE CONTROL BLOCK DEFINITIONS
25: FCBDN      EQU      FCB+0      ;DISK NAME
26: FCBFN      EQU      FCB+1      ;FILE NAME
27: FCBFT      EQU      FCB+9      ;DISK FILE TYPE (3 CHARACTERS)
28: FCBRL      EQU      FCB+12     ;FILE'S CURRENT REEL NUMBER
29: FCBRC      EQU      FCB+15     ;FILE'S RECORD COUNT (0 TO 128)
30: FCBRCR      EQU      FCB+32    ;CURRENT (NEXT) RECORD NUMBER (0 TO 127)
31: FCBLEN      EQU      FCB+33    ;FCB LENGTH
32: ;
33: ; SET UP STACK
34: LXI      H,0
35: DAD      SP
36: ; ENTRY STACK POINTER IN HL FROM THE CCP
37: SHLD     OLDSP
38: ; SET SP TO LOCAL STACK AREA (RESTORED AT FINIS)
39: LXI      SP,STKTOP
40: ; READ AND PRINT SUCCESSIVE BUFFERS
41: CALL     SETUP      ;SET UP INPUT FILE
42: CPI      255        ;255 IF FILE NOT PRESENT
43: JNZ      OPENOK     ;SKIP IF OPEN IS OK
44: ;
45: ; FILE NOT THERE, GIVE ERROR MESSAGE AND RETURN
46: LXI      D,OPNMSG
47: CALL     ERR
48: JMP      FINIS      ;TO RETURN
49: ;
50: OPENOK:   ;OPEN OPERATION OK, SET BUFFER INDEX TO END
51: MVI      A,80H
52: STA      IBP          ;SET BUFFER POINTER TO 80H
53: ; HL CONTAINS NEXT ADDRESS TO PRINT
54: LXI      H,0          ;START WITH 0000
55: ;
56: GLOOP:

```



```

57:      PUSH    H           :SAVE LINE POSITION
58:      CALL    GNB         :
59:      POP     H           :RECALL LINE POSITION
60:      JC      FINIS       :CARRY SET BY GNB IF END FILE
61:      MOV     B,A
62:      :      PRINT HEX VALUES
63:      :      CHECK FOR LINE FOLD
64:      MOV     A,L
65:      ANI     0FH         :CHECK LOW 4 BITS
66:      JNZ     NONUM
67:      :      PRINT LINE NUMBER
68:      CALL    CRLF
69:      :
70:      :      CHECK FOR BREAK KEY
71:      CALL    BREAK
72:      :      ACCUM LSB = 1 IF CHARACTER READY
73:      RRC
74:      JC      FINIS       :DON'T PRINT ANY MORE
75:      :
76:      MOV     A,H
77:      CALL    PHEX
78:      MOV     A,L
79:      CALL    PHEX
80:  NONUM:
81:      INX     H           :TO NEXT LINE NUMBER
82:      MVI     A,' '
83:      CALL    PCHAR
84:      MOV     A,B
85:      CALL    PHEX
86:      JMP     GLOOP
87:      :
88:  FINIS:
89:      :      END OF DUMP, RETURN TO CCP
90:      :      (NOTE THAT A JMP TO 0000H REBOOTS)
91:      CALL    CRLF
92:      LHLD    OLDSP
93:      SPHL
94:      :      STACK POINTER CONTAINS CCP'S STACK LOCATION
95:      RET
96:      :
97:      :
98:      :      SUBROUTINES
99:      :
100:  BREAK: :CHECK BREAK KEY (ACTUALLY ANY KEY WILL DO)
101:      PUSH H! PUSH D! PUSH B! ENVIRONMENT SAVED
102:      MVI     C,BRK
103:      CALL    BDOS
104:      POP B! POP D! POP H! ENVIRONMENT RESTORED
105:      RET
106:      :
107:  PCHAR: :PRINT A CHARACTER
108:      PUSH H! PUSH D! PUSH B! SAVED
109:      MVI     C,TYPEF
110:      MOV     E,A
111:      CALL    BDOS

```



```

112: POP B: POP D: POP H: RESTORED CARRY INDEX
113: RET
114: ;
115: CRLF: MVI A,CR
116: CALL PCHAR
117: MVI A,LF
118: CALL PCHAR
119: RET
120: ;
121: ;
122: ;
123: PNIB: :PRINT NIBBLE IN REG A
124: ANI 0FH :LOW 4 BITS
125: CPI 10
126: JNC P10
127: ; LESS THAN OR EQUAL TO 9
128: ADI '0'
129: JMP PRN
130: ;
131: ; GREATER OR EQUAL TO 10
132: P10: ADI 'A' - 10
133: PRN: CALL PCHAR
134: RET
135: ;
136: PHEX: :PRINT HEX CHAR IN REG A
137: PUSH PSW
138: RRC
139: RRC
140: RRC
141: RRC
142: CALL PNIB :PRINT NIBBLE
143: POP PSW
144: CALL PNIB
145: RET
146: ;
147: ERR: :PRINT ERROR MESSAGE
148: ; D,E ADDRESSES MESSAGE ENDING WITH "4"
149: MVI C,PRINTF :PRINT BUFFER FUNCTION
150: CALL BDOS
151: RET
152: ;
153: ;
154: GNB: :GET NEXT BYTE
155: LDA IBP
156: CPI 80H
157: JNZ GO
158: ; READ ANOTHER BUFFER
159: ;
160: ;
161: CALL DISKR
162: ORA A :ZERO VALUE IF READ OK
163: JZ GO :FOR ANOTHER BYTE
164: ; END OF DATA, RETURN WITH CARRY SET FOR EOF
165: STC
166: RET
167: ;

```



```

168: GO:      :READ THE BYTE AT BUFF+REG A
169:      MOV    E,A      :LS BYTE OF BUFFER INDEX
170:      MVI    D,0      :DOUBLE PRECISION INDEX TO DE
171:      INR    A        :INDEX=INDEX+1
172:      STA    IBP      :BACK TO MEMORY
173:      :      POINTER IS INCREMENTED
174:      :      SAVE THE CURRENT FILE ADDRESS
175:      LXI    H,BUFF
176:      DAD    D
177:      :      ABSOLUTE CHARACTER ADDRESS IS IN HL
178:      MOV    A,M
179:      :      BYTE IS IN THE ACCUMULATOR
180:      ORA    A        :RESET CARRY BIT
181:      RET
182:      :
183: SETUP:    :SET UP FILE
184:      :      OPEN THE FILE FOR INPUT
185:      XRA    A        :ZERO TO ACCUM
186:      STA    FCBCR    :CLEAR CURRENT RECORD
187:      :
188:      LXI    D,FCB
189:      MVI    C,OPENF
190:      CALL   BDOS
191:      :      255 IN ACCUM IF OPEN ERROR
192:      RET
193:      :
194: DISKR:    :READ DISK FILE RECORD
195:      PUSH H! PUSH D! PUSH B
196:      LXI    D,FCB
197:      MVI    C,READF
198:      CALL   BDOS
199:      POP B! POP D! POP H
200:      RET
201:      :
202:      :      FIXED MESSAGE AREA
203: SIGNON: DB    'FILE DUMP VERSION 1.4*'
204: OPNMSG: DB    CR,LF,'NO INPUT FILE PRESENT ON DISK*'
205:      :
206:      :      VARIABLE AREA
207: IBP:      DS    2      :INPUT BUFFER POINTER
208: OLDSP:    DS    2      :ENTRY SP VALUE FROM CCP
209:      :
210:      :      STACK AREA
211:      DS    64      :RESERVE 32 LEVEL STACK
212: STKTOP:
213:      :
214:      END

```


TYPE TIME.ASM

```

;
; Console CP/M Time-of-day Program
;
;
; v1.08 12/22/82 WGW Changed name of program to TIME.COM
; v1.07 11/12/82 WGW Correct values for get/settime
; v1.06 11/10/82 WGW Changed help message to fit on EC8800
; v1.05 9/20/82 WGW Get number of days via gettime
; v1.04 7/01/82 WGW Correct noon time (12 PM)
; v1.03 6/07/82 WGW Correct get character routine
; v1.02 5/27/82 WGW Check validity of time, regular/military time
; v1.01 5/21/82 WGW Set time of day
; v1.00 5/20/82 WGW Read time of day (original version)
;

```

```

version equ 1
revision equ 8

month equ 12
day equ 22
year equ 82

cpm equ 0
bdos equ 5
fcb1 equ 5ch ; location of first fcb parsed by cpm
fcb2 equ 6ch ; location of second fcb
filename equ 1 ; offset into fcb
fileextension equ filename+8 ; offset into fcb

parameter1a equ fcb1+filename
parameter1b equ fcb1+fileextension
parameter2a equ fcb2+filename
parameter2b equ fcb2+fileextension

tpa equ 100h

beep equ 7
cr equ 13
lf equ 10
endmsg equ '$'

regulartime equ 0
militarytime equ 0ffh

uppercase equ 5fh
displaychar equ ' ' ; character indicating display tod
repeatchar equ 'p' ; character indicating repeat mode
militarychar equ 'M' ; character indicating military tod

maxhours equ 24h ; 24 hours
morning equ maxhours/2
noon equ 12h
midnight equ 0
maxminutes equ 60h ; 60 minutes

readconsole equ 1 ; bdos function calls

```



```

getconsole      equ      6
printstring     equ      9
consolestatus   equ     11
settime        equ     104
readtime       equ     105
readclock      equ     155

        org      tpa
        jmp      start

signon: db      cr,lf
        db      'Time-of-day Program  WGW  V'
        db      (version/10)+'0'
        db      (version mod 10)+'0'
        db      '.'
        db      (revision/10)+'0'
        db      (revision mod 10)+'0'
        db      ' '
        db      (month/10)+'0'
        db      (month mod 10)+'0'
        db      '/'
        db      (day/10)+'0'
        db      (day mod 10)+'0'
        db      '/'
        db      (year/10)+'0'
        db      (year mod 10)+'0'
        db      endmsg

displaytype:
        db      regulartime          ; default display type

;
; --- Main part of program -----
;

start:  lxi      sp,stack              ; use internal stack
        call    main                  ; perform main program
        lxi     d,crlfmsg
        call    print1
        jmp     cpm                  ; exit

main:   lda      parameter1a          ; a := first char of first parameter
        call    isdigit              ; see if a digit is entered
        jz      settod               ; see if the time is to be set
        cpi     militarychar         ; check for military time
        jnz     checkdisplaytype
        mvi     a,militarytime       ; set display type to military time
        sta     displaytype
        lda     parameter2a          ; a := first char of second parameter

checkdisplaytype:
        cpi     displaychar          ; check for print one time display
        jz      printtime
        cpi     repeatchar           ; check for repeat time display
        jz      repeatprinttime
        lxi     d,signon              ; otherwise print signon
        call    print
        lxi     d,helpmsg             ; and print format message

```



```

print: call    print1          ; print string
printf:      ;
        lxi    d,crLfmsg      ; begin a new line
print1: mvi    c,printstring
        jmp    bdos

helpmsg:
db          'The following display the time-of-day:',cr,lf
db          cr,lf
db          'TIME          Prints normal time',cr,lf
db          'TIME M       Prints military time',cr,lf
db          cr,lf
db          'The following display the time-of-day',cr,lf
db          '      until a key is pressed:',cr,lf
db          cr,lf
db          'TIME P          Prints normal time',cr,lf
db          'TIME M P       Prints military time',cr,lf
db          cr,lf
db          'The following set the time-of-day:',cr,lf
db          cr,lf
db          'TIME hh.mm      Set military time',cr,lf
db          'TIME hh.mm AM   Set normal time (morning)',cr,lf
db          'TIME hh.mm PM   Set normal time (evening)',cr,lf
db          '
db          '      when a key is pressed.',cr,lf
db          endmsg

printtime:   ; print time-of-day
        lxi    d,timeismsg     ; print 'time is'
        call   print1
        lxi    d,tod           ; read the tod clock
        mvi    c,readclock
        call   bdos
        call   converttime
        lxi    d,timemsg       ; print the time
        call   print1
        call   ismilitary      ; see if military time display
        cnz    printampm
        lxi    d,crmsg         ; print trailing cr
        jmp    print1

printampm:
        call   getampm         ; get right letter
        mov    a,l             ; a := am/pm character
        sta    ampm           ; update message
        lxi    d,ampmmsg       ; print am/pm message
        jmp    print1

getampm:
        mvi    l,'P'          ; l := possible result
        lda    todhours       ; a := hours
        cpi    morning        ; see if it is the morning
        rnc
        mvi    l,'A'          ; exit if in the evening
        ret

```



```

converttime:
    lda    todhours          ; convert to ascii and display
    call   converthourcheck  ; check the hours
    call   makeascii
    shld   msghours
    lda    todminutes
    call   makeascii
    shld   msgminutes
    lda    todseconds
    call   makeascii
    shld   msgseconds
    ret

converthourcheck:
    mov     b,a              ; b := hours (BCD)
    call    ismilitary       ; see if it is military time
    mov     a,b              ; a := hours (BCD)
    rz      ; exit if military time
    ora     a                ; check for midnight
    mvi     a,12h            ; return 12 if midnight
    rz      ; exit if midnight
    mov     a,b              ; a := hours (BCD)
    cpi     morning+1        ; check if evening
    rc      ; exit if in morning
    adi     88h              ; subtract 12 hours
    daa     ; make BCD
    ret

ismilitary:
    lda     displaytype      ; a := displaytype
    cpi     militarytime     ; return zero if military time
    ret

repeatprinttime:
    call    printtime        ; print tod until character entered
    mvi     c,consolestatus  ; get console status
    call    bdos
    ora     a
    jz      repeatprinttime  ; loop until done if no character typ
getchr:    mvi     c,consolestatus  ; get console status
    call    bdos
    ora     a
    jz      getchr           ; wait until character entered
    mvi     e,0ffh          ; e := get character
    mvi     c,getconsole     ; get the character
    jmp     bdos             ; and exit

settod:
    ; set time-of-day
    ; get time-of-day to set days
    lxi     d,tod
    mvi     c,readtime
    call    bdos
    lhd     parameteria      ; get hours
    xchg    ; de := parameter
    call    makebcdblank     ; convert to bcd, possible leading '
    cpi     0ffh             ; check for entry errors

```



```

jz      seterror
call    checkhours          ; check for range error
jnc     seterror            ; jump if greater or equal to limit
sta     todhours
lhld    parameter1b        ; get minutes
xchg
call    makebcd             ; de := parameter
cpi     0ffh               ; convert to bcd
jz      seterror           ; check for entry errors
cpi     maxminutes         ; check for range error
jnc     seterror           ; jump if greater or equal to limit
sta     todminutes
lxi     d, startmsg         ; print waiting to set time message
call    print1
call    getchr              ; get a character
call    printerlf
lxi     d, tod              ; set time-of-day
mvi     c, settime
call    bdos
jmp     printtime           ; print the time just set

checkhours:
mov     b, a                ; b := hours (BCD)
lda     parameter2a         ; a := first char of second parameter
cpi     'p'                 ; see if it is PM
jz      checkhourspm
cpi     'A'                 ; see if it is AM
jz      checkhoursam
cpi     ' '                 ; see if it is military time (default)
jz      checkhoursmilitary
ora     a                   ; no carry indicates an error
ret

checkhoursmilitary:
mov     a, b                ; a := hours (BCD)
cpi     maxhours            ; check max hours
ret

checkhourspm:
mov     a, b                ; a := hours (BCD)
ora     a                   ; see if it is zero
rz      ; return with no carry if zero
adi     morning             ; convert to PM hours
daa     ; make BCD
cpi     maxhours            ; check max hours
rnz     ; return in not midnight
mvi     a, noon             ; return a := 12h if noon
stc     ; set carry to indicate it is ok
ret

checkhoursam:
mov     a, b                ; a := hours (BCD)
ora     a                   ; see if it is zero
rz      ; return with no carry if zero
cpi     morning             ; check max hours
rnc     ; exit if ok (1 to 11 am)
cpi     12h                 ; see if it is 12 am

```



```

rnc                                ; exit if not 12 am (error)
sub    a, 00h                      ; a := 0 (12 midnight)
stc
ret                                ; set carry to indicate it is ok

startmsg:
db    cr,lf
db    'Enter any key to set time of day',endmsg

seterrormsg:
db    'Invalid time-of-day format',endmsg

seterror:
lxi    d,seterrormsg                ; print error message
call   error
lxi    d,helpmsg                    ; print how-to-get-help message
jmp    print

helpmsg:
db    cr,lf
db    'Enter the following to get more information:',cr,lf
db    cr,lf
db    'TIME ?',endmsg

error:  push    d                    ; save error message index
        lxi    d,errorprefix        ; print error prefix
        call   print1
        pop    d
        call   print1                ; print error message
        lxi    d,errorsuffix        ; print error suffix
        jmp    print

errorprefix:
db    cr,lf
db    '*** ',endmsg

errorsuffix:
db    ' ***',beep,endmsg

isdigit:                                ; returns zero if A is an ascii digit
        mov    c,a                  ; c := character
        push   b                    ; save it
        call   isdigit1             ; check digit
        pop    b                    ; restore character, return zero flag
        mov    a,c
        ret

isdigit1:
        cpi    '0'                  ; is it less than a '0'
        jnc    returnff
        cpi    '9'+1                 ; is it greater than a '9'
        jnc    returnff

return0:
sub    a,0                          ; a := 0, zero = true
ret

```



```

returnff:
    mvi    a,0ffh      ; a := 0ffh, zero = false
    ora    a
    ret

makebcdblank:
    mov    a,d          ; a := bcd value of de (ls, ms), ls can =
    cpi    ' '          ; a := ls ascii digit
    jnz    makebcd      ; see if it is a blank
    mov    d,e          ; convert if already a digit
    mvi    e,'0'        ; shift right one digit
    ; set in a leading zero

makebcd:
    mov    a,e          ; a := bcd value of de (ls, ms)
    call   isdigit      ; a := ms ascii digit
    rnz    ; see if it is a digit
    mov    a,e          ; exit if it is not
    ral    ; restore ms ascii digit
    ral    ; shift into upper nibble
    ral
    ani    0f0h         ; a (ms nibble) := bcd part of ms ascii dig
    mov    e,a          ; save it
    mov    a,d          ; a := ls ascii digit
    call   isdigit      ; see if this is a digit too
    rnz
    mov    a,d          ; restore ls ascii digit
    ani    0fh          ; a (ls nibble) := bcd part of ls ascii dig
    ora    e            ; a := result
    ret

makeascii:
    push   psw
    call   makedigit
    mov    h,a          ; h := ls digit (remember, msb stored last)
    pop    psw
    rrc
    rrc
    rrc
    rrc
    call   makedigit
    mov    l,a          ; a (ls nibble) := ms nibble
    ; hl := two ascii digits
    ret

makedigit:
    ani    0fh          ; a := ls nibble
    adi    '0'          ; convert to a digit
    ret

tod:
    dw     0
todhours:
    db     0
todminutes:
    db     0
todseconds:
    db     0

timeismsg:
    db     'The time is ',endmsg

timemsg:
    ; displays time

```



```

msghours:      db      '00.'
msgminutes:    db      '00.'
msgseconds:    db      '00',endmsg

```

```

amprmsg:      db      ' '
ampm:         db      ' M',endmsg

```

```

crmsg:        db      cr,endmsg

```

```

crlfmsg:      db      cr,lf,endmsg

```

```

ds      256      ; internal stack
stack:

```

```

end

```

2. DEC Decision Note 7 CP/M User's Guide.

3. DYNABYTE Business Computer: CP/M 2.2 User's Guide.

4. MICRO SYMBIOSIS, September 82, p. 128-134.

5. WIRELESS WORLD, January 1983, Floppy User and Systems: David Smith.

6. AND "Advanced Micro Devices", 8088 & 8086 Microprocessors & Family.

MSX Microprocessors and Peripherals: Data Book.

7. Flight Technical SP4000: Aug 84, Apple II/III.

8. Billings, David: Microsoft MS-DOS: The Definitive Guide.

KAYNAKLAR:

1. MONROE CP/M İşletim Sistemi Kullanıcı Kitapları.
 - . Introduction to CP/M Features and Facilities
 - . CP/M User's Guide
 - . CP/M Interface Guide
 - . CP/M ED Context Editor
 - . CP/M Assembler (ASM) User's Guide
 - . CP/M Dynamic Debugging Tool User's Guide
 - . CP/M Alteration Guide
 - . CP/M-80 Supplement
2. NCR Decision Mate V CP/M User's Guide.
3. DYNABYTE Business Computers CP/M 2.2 User's Guide.
4. MICRO SYSTEMES, Septembre 82, s.124-134.
5. WIRELESS WORLD, January 1985, Floppy Discs and Systems; David March.
6. AMD "Advanced Micro Devices", 8080A 8 bit Microprocessor s.5.1-5.8,
MOS Microprocessors and Peripherals Data Book.
7. Bilişim Terimleri Sözlüğü; Doç.Dr. Aydın Köksal.
8. Bilişim, Genel Sistemler ve Siberetik Terimler Sözlüğü; Emin D. Aydın.

ÖZGEÇMİŞ

Hulusi Kltr, 15.12.1960 tarihinde Kayseri'de doęmuştur. İlk öğrenimini Kadıky Osmangazi İlkokulunda, orta öğrenimini Kadıky Kemal Atatrk Ortaokulunda ve lise öğrenimini 1975-1979 yılları arasında Haydarpaşa Endstri Meslek Lisesi (I.Sınıf), Maçka Teknik Lisesi (II.,III.,IV.Sınıf) Elektronik Blmnde yapmıştır. Teknik lise eęitimi boyunca Koç Vakfından karşılıksız öğrenci bursu almıştır. Öğrencilięi sırasında, PTT Araştırma Laboratuvarında (PTT-ARLA) ve çeşitli şirketlerde çalışmıştır. 1979 yılında girdięi Yıldız Üniversitesi'nden 1983 yılında Elektronik ve Haberleşme Mühendisi olarak mezun olmuştur. Halen Yıldız Üniversitesi, Fen Bilimleri Enstits Elektronik ve Haberleşme Blmnde öğrenimine devam etmektedir.