

YILDIZ TEKNİK ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

Kişisel Bilgi Disk Paylaşımı

YÜKSEK LİSANS TEZİ

A. Sedat Alış

1990

YILDIZ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KİŞİSEL BİLGİSAYARLARDA
DİSK PAYLAŞIMI

YÖKSEK LİSANS TEZİ

BİLGİSAYAR MÜH. AHMET SEDAT ALIŞ

TEZ YÖNETİCİSİ

Prof. M. YAHYA KARSLIGİL

İSTANBUL 1990

YILDIZ TEKNİK ÜNİVERSİTESİ
KÜTÜPHANE DOKÜMANTASYON
DAİRE BAŞKANLIĞI

Kot : R 368
Alındığı Yer : 74
F.B. Enstitüsü
Tarih : 19.9.1996
Fatura :
Fiyatı : 40 Bin
Ayniyat No : 1/7
Kayıt No : 52474
UDC :
Ek :
Y.T.Ü.
KÜTÜPHANE DOK. Bn. BAŞKANLIĞI

İçindekiler

Özet

Abstract

Metin

1. Giriş

2. Materyal Ve Yöntem

2.1 RS232 ile Haberleşme

2.1.1 Programlanabilir Band-Hız Ölçetirici

2.1.2 Saklayıcılar

2.2

2.3

2.3.1

2.3.2

2.4 RS232 ile ilgili Keşme Ve Mesajlar Serisi

2.5 MS-DOS İşletim Sisteminde Cihaz Sürücüler

2.5.1 Karakter Cihazlar

2.5.2

2.5.3

2.5.4

2.6 MS-DOS İşletim Sisteminde Cihaz İş Yürütme

2.6.1

2.6.2

BİLGİSAYAR MÜH. AHMET SEDAT ALIŞ

YÜKSEK LİSANS TEZİ

TEZ YÖNETİCİSİ

Prof. M. YAHYA KARSLIGİL

4. Tartışma Ve Sonuç

4.1 Parametrelere Göre Programın Tasarımı

4.2 Keşme Hizmet Programı Ve Sürücünün Yazılması

4.2.1 INIT-Yordamı

İçindekiler

Özet

Abstract

Metin

1. Giriş	1
2. Materyal Ve Yöntem	2
2.1 RS232 İle Haberleşme	2
2.1.1 Programlanabilir Baud-Hız Oluşturucu	3
2.1.2 Saklayıcılar	3
2.2 RS232 Kablo Bağlantısı	14
2.3 80X86 Ailesi Mikroişlemcilerde Kesmeler	15
2.3.1 Kesmeler Hakkında Genel Bilgi	15
2.3.2 Kesme Vektörleri	16
2.4 RS232 İle İlgili Kesmeler Ve Bunları Sezme	17
2.5 MS-DOS İşletim Sisteminde Cihaz Sürücüler	19
2.5.1 Karakter Cihazlar	20
2.5.2 Blok Cihazlar	20
2.5.3 Cihaz Sürücüler Nasıl Çalışır?	21
2.5.4 Cihaz Sürücü Detayları	22
2.6 MS-DOS İşletim Sisteminde Çoklu İş Yürütme	48
2.6.1 TSR Programların Kuralları	48
2.6.2 InDOS Bayrağı, DOSOK Kesmesi ve Zamanlayıcı Kesmesi	50
2.6.3 DOS Servis Programlarına Tekrar Girme	52
2.7 Disklerin Paylaşımı Ve Genel Prensi	53
3. Deneysel Çalışma Ve Bulgular	56
3.1 Veri İletişim Yöntemi	56
3.2 RS232 Kesme Sezme Deneyi	60
3.3 Makina Tipinin Belirlenmesi	61
3.4 Disklere Ait Bilgilerin Elde Edilmesi	63
4. Tartışma Ve Sonuç	67
4.1 Parametrelere Göre Programın Tasarımı	67
4.2 Kesme Hizmet Programı Ve Sürücünün Yazılması	69
4.2.1 INIT Yordamı	71

4.2.2	MEDIA CHECK Yordamı	71
4.2.3	BUILD BPB Yordamı	72
4.2.4	READ Yordamı	72
4.2.5	WRITE Yordamı	73
4.2.6	WRITE WITH VERIFY Yordamı	73
4.3	Disk Paylaşımının Kurulması	82
4.4	Programlar	
4.4.1	INTOCPC.ASM	
4.4.2	INTOCPS2.ASM	
4.4.3	RSINIT.MAC	
4.4.4	RSINIT.INC	
4.4.5	RSTR.INC	
4.4.6	PTERM.C	
4.4.7	PMAIN.C	
4.4.8	PMAIN.H	
4.4.9	SVBS.ASM	
4.4.10	RDDRVR.ASM	
4.4.11	INTSRVR.ASM	
4.4.12	MP.BAT	

Kaynaklar

Özgeçmiş

ÖZET

Disk paylaşımı, temel olarak bir terminal durumundaki bilgisayarın, master durumundaki bilgisayarın diskini ortak olarak kullanmasıdır. Bu sistemin amacı, terminal makinanın, master makinanın diskinde bulunan programlardan (compiler, database dilleri v.b.) yararlanmasını sağlamaktır.

İki makina arasındaki veri iletişimi ise, RS232 kablo bağlantısı ile mümkün olmaktadır. Bağlantı, iki bilgisayar arasında yapıldığından, son derece yüksek baud oranları ile haberleşilebilir (19200, 38400, 115200 v.b.).

Terminal durumundaki makina için yazılan program, temel olarak bir cihaz sürücüdür ve master makinanın disklerine erişmek için bir arabirimdir. Master durumundaki makinanın, terminal durumundaki makinadan gelen istekleri karşılayabilmesi için, terminalden gelen verileri kesme ile sezmesi sağlanmıştır. Bunun için bir kesme kontrol programı ve master makinasının cihaz sürücüsüne erişebilen bir ara program yazılmıştır.

ABSTRACT

Basically, **Disk Sharing** is the common use of disk of a master computer (host) by another computer called terminal. The main concept of this system is the use of some executable files (compiler, database languages etc.) by terminal from the master computer's disks.

The data communication between computers is provided by a RS232 cable connection. As the connection is done between master computer and terminal, we could use higher baud rates as 19200, 38400, 115200 etc.

The program written for the terminal is simply a device driver and an interface to access the master computer's disk units. The master is equipped by an interrupt handler to satisfy the terminal requests. For this purpose, an interrupt handler and a supplementary program is written to access master computer's device driver.

İletişimi sağlayan programlar, 8086 Assembly dilinde yazılmıştır. Bu programlara bağlı olarak yazılan diğer programlar ise başlıca C, bazılarını Assembly dilindedir.

Bu tez, n. benden sonra bu konuda çalışmak isteyen meslektaşlarıma açık olacaktır. İnanıyorum ki, tezimin gerçekleştirmesindeki katkılarından dolayı ayrıca Sayın Prof. N. Yahya Karabıçak'a teşekkür ederim.

İstanbul, 1996

Ahmet Sedat ALIŞ

1. GİRİŞ AL VE YONTEN

Bilgisayar ağlarının son derece yaygın kullanıldığı şu günlerde, artık bilgisayarlar tek iş düzeninden uzaklaşıp, çok iş düzeninde ve çok kullanıcı olarak çalışmaya başlamışlardır. İşte bu ihtiyaçtan faydalanarak, MS-DOS işletim sisteminin tek kullanıcı durumu, disk paylaşımı ile iki veya daha fazla kullanıcıya çıkarmak fikri doğmuştur. Tezin içeriği, basit olarak bir terminal durumundaki bilgisayarın, bir master durumundaki bilgisayarın disklerini ortak olarak kullanması olarak özetlenebilir. Bu tezin amacı aslında çok yüzeysel görünmekle birlikte, birçok konuyu da içine almaktadır.

Tezin bağlantılı bulunduğu konular, RS232 haberleşme, Hafızada kalıcı program yazma, Kesmeler, Cihaz Sürücüler, DOS'da çoklu iş yürütme gibi başlıklar altında toplanabilir. Tabii, bu başlıkların dalları da oldukça detaylıdır.

İletişimi sağlayan programlar, 8086 Assembly dilinde yazılmıştır. Bu programlara bağlı olarak yazılan diğer programlar ise bazıları C, bazıları Assembly dilindedir.

Bu tezin, benden sonra bu konuda çalışmak isteyen meslektaşlarıma ışık tutacağına inanıyorum. Tezimin gerçekleşmesindeki katkılarından dolayı ayrıca Sayın Prof. M.Yahya Karslıgil'e teşekkür ederim.

Seri çıkış portu, seri çıkış portu 1 (Seri 1) veya seri
İstanbul, 1990 (Seri 2) olarak adreslenir. Ahmet Sedat ALIŞ

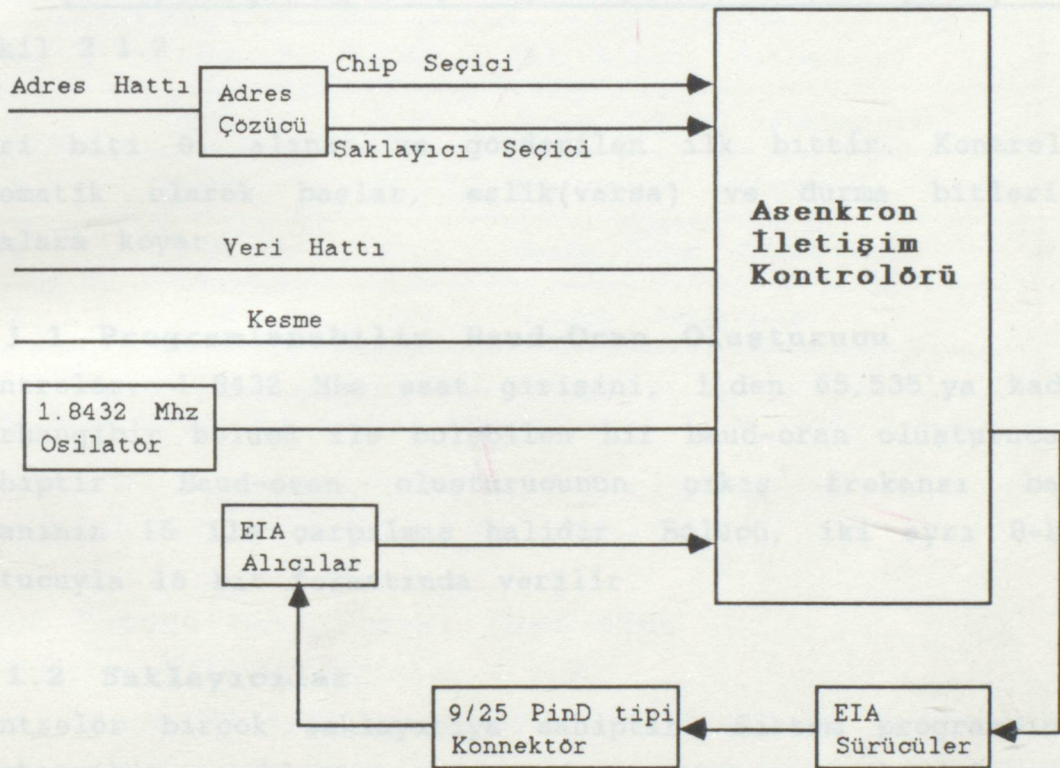
Kullanılan n. Seri 1 için 3, Seri 3 için 2 değeri alır.

Sisteme iki kesme hattı girmektedir. Kesme düzeyi 4
(Interrupt level 4 - IRQ4) Seri 1 için, kesme düzeyi 3
(IRQ3) Seri 2 içindir. Kontroloran, sistem kesme

2. MATERİYAL VE YÖNTEM

2.1 RS232 İle Haberleşme

Kişisel bilgisayarlarda seri haberleşme, Asenkron İletişim Kontrolörü (Asynchronous Communication Controller) yoluyla yapılır. Genel olarak bu tip adaptörler, içerik olarak aynı işlevlere sahiptirler. Bu kontrolörün de içinde bulunduğu genel şema Şekil 2.1.1'de gösterilmiştir.



Şekil 2.1.1

Seri çıkış portu, seri çıkış portu 1 (Seri 1) veya seri çıkış portu 2 (Seri 2) olarak adreslenebilir. Adreslerde kullanılan n , Seri 1 için 3, Seri 2 için 2 değerini alır.

Sisteme iki kesme hattı gitmektedir. Kesme düzeyi 4 (Interrupt level 4 - IRQ4) Seri 1 için, kesme düzeyi 3 (IRQ3) Seri 2 içindir. Kontrolörün, sistem kesme

kontrolörüne kesme gönderebilmesi için Modem Kontrol Saklayıcısının 3. bitinin 1 olması gerekir. Bu durumda, Kesme İzin Saklayıcısı tarafından izin verilen kesme tipleri bir kesme oluşmasına yol açacaklardır.

Veri formatı Şekil 2.1.2'deki gibidir.

İşaretçi	Başla Biti	D0	D1	D2	D3	D4	D5	D6	D7	Eşlik Biti	Durma Biti
----------	------------	----	----	----	----	----	----	----	----	------------	------------

Şekil 2.1.2

Veri biti 0, alınan ve gönderilen ilk bittir. Kontrolör otomatik olarak başlar, eşlik(varsa) ve durma bitlerini aralara koyar.

2.1.1 Programlanabilir Baud-Oran Oluşturucu

Kontrolör, 1.8432 Mhz saat girişini, 1'den 65,535'ya kadar herhangi bir bölücü ile bölebilen bir baud-oran oluşturucuya sahiptir. Baud-oran oluşturucunun çıkış frekansı baud oranının 16 ile çarpılmış halidir. Bölücü, iki ayrı 8-bit tutucuyla 16 bit formatında verilir.

2.1.2 Saklayıcılar

Kontrolör birçok saklayıcıya sahiptir. Sistem programcısı, herhangi bir saklayıcı üzerinde, sistem mikroişlemcisi yardımıyla işlem yapabilir. Bu saklayıcılar, hem işlemleri kontrol etmek hem de veri almak ve göndermek için kullanılır. Saklayıcılar Tablo 2.1.1'de gösterilmiştir.

BTEB ** Durumu	Port Adresi (hex)	R/W	Saklayıcı
0	0nF8 *	W	Verici Tutucu Saklayıcısı n
0	0nF8 *	R	Alıcı Tampon Saklayıcısı n
1	0nF8 *	R/W	Bölücü Tutucu, Alçak an. byte
1	0nF9 *	R/W	Bölücü Tutucu, Yüksek an. byte
0	0nF9 *	R/W	Kesme İzin Saklayıcısı
X	0nFA *	R	Kesme Tanım Saklayıcısı
X	0nFA *	W	FIFO Kontrol Saklayıcısı
X	0nFB *	R/W	Hat Kontrol Saklayıcısı
X	0nFC *	R/W	Modem Kontrol Saklayıcısı
X	0nFD *	R	Hat Durum Saklayıcısı
X	0nFE *	R	Modem Durum Saklayıcısı
X	0nFF *	R/W	Tırmık Saklayıcısı
* n seçilen portu gösterir. Seri 1 için 3, Seri 2 için 2'dir. ** BTEB durumu Hat Kontrol Saklayıcısının 7. biti ile kontrol edilir.			

Tablo 2.1.1 Seri Port Saklayıcı Adresleri.

Verici Tutucu Saklayıcısı (hex nF8)

(Transmitter Holding Register - THR)

Verici Tutucu Saklayıcısı, gönderilecek karakteri içerir. Bu saklayıcının herbiti gönderilecek veri bitine karşılık gelir.

Tablo 2.1.3 Bölücü Tutucu Saklayıcısı(LSB) / Bölücü Tutucu Saklayıcısı(HSB)

Bölücü Tutucu Saklayıcısı (hex nF9)

(Divisor Latch Register - DLR)

Bölücü Tutucu saklayıcısı, baud oran alıştırıcuyu programlamak için kullanılır. Bu iki saklayıcıdaki değerler,

Bit	Fonksiyon
7	Veri Biti 7
6	Veri Biti 6
5	Veri Biti 5
4	Veri Biti 4
3	Veri Biti 3
2	Veri Biti 2
1	Veri Biti 1
0	Veri Biti 0

Tablo 2.1.2 Verici Tutucu Saklayıcısı(THR) / Alıcı Tampon Saklayıcısı(RBR).

Alıcı Tampon Saklayıcısı (hex nF8)

(Receiver Buffer Register - RBR)

Alıcı Tampon Saklayıcısı, alınan karakteri içerir. Bu saklayıcının herbiti gönderilecek veri bitine karşılık gelir.

Bölücü Tutucu Saklayıcısı (hex nF8)

(Divisor Latch Register - LSB)

Bit	Fonksiyon
7	Bit 7
6	Bit 6
5	Bit 5
4	Bit 4
3	Bit 3
2	Bit 2
1	Bit 1
0	Bit 0

Tablo 2.1.3 Bölücü Tutucu Saklayıcısı(LSB) / Bölücü Tutucu Saklayıcısı(MSB).

Bölücü Tutucu Saklayıcısı (hex nF9)

(Divisor Latch Register - MSB)

Bölücü Tutucu saklayıcıları, baud oran oluşturucuyu programlamak için kullanılır. Bu iki saklayıcıdaki değerler,

saat girişinin (1.8432 Mhz) bölücüsünü oluşturur. Tablo 2.1.4'de bazı bölücü değerleri için baud oran değerleri verilmiştir.

Baud Oranı	16xsaat bölücü değeri (ondalık)	İstenen ve gerçek değer arasındaki hata yüzdesi
50	2304	-
75	1536	-
110	1047	0.026
134.5	857	0.058
150	768	-
300	384	-
600	192	-
1200	96	-
1800	64	-
2000	58	0.069
2400	48	-
3600	32	-
4800	24	-
7200	16	-
9600	12	-
19200	6	-

Tablo 2.1.4 1.8432 Mhz'de baud oranları.

Kesme İzin Saklayıcısı (hex nF9)

(Interrupt Enable Register - IER)

Bu 8-bit saklayıcı dört tip kontrolör kesmesine ayrı ayrı chip-kesme sinyalleri oluşturur. Kesme sistemi, 0-3. bitler 0 yapılarak tamamen iptal edilebilir. Benzer olarak, uygun bitleri 1 yaparak istenilen kesmelere izin verilebilir.

Bit	Fonksiyon
7-4	Saklıdır = 0
3	Modem-Durum Kesmesi
2	Alıcı-Hat-Durum Kesmesi
1	Verici-Tutucu-Sak.-Boş Kesmesi
0	Alınmış Veri Hazır Kesmesi

Tablo 2.1.5 Kesme İzin Saklayıcısı (IER).

Bit(7-4) : saklıdır ve daima 0 olmalıdır.

Bit(3) : 1 olduğunda modem durum kesmesine izin verir.

Bit(2) : 1 olduğunda alıcı hat durum kesmesine izin verir.

Bit(1) : 1 olduğunda alınmış veri hazır kesmesine izin verir.

Kesme Tanım Saklayıcısı (hex nFA)

(Interrupt Identification Register - IIR)

Bu saklayıcı veri transferi sırasında oluşacak kesmeler için tanım saklayıcısıdır. Dört öncelik düzeyi vardır.

- * Öncelik 1 - Alıcı Hat Durumu
- * Öncelik 2 - Alınmış Veri Hazır
- * Öncelik 3 - Alıcı Tutucu Saklayıcısı Boş
- * Öncelik 4 - Modem Durumu

Kesme tanım saklayıcısı adreslendiğinde, en yüksek öncelikli kesme tutulur, ve sistem mikroişlemcisi kesmeye hizmet verene kadar başka kesmelere cevap verilmez.

Bit	Fonksiyon
7-3	Saklıdır = 0
2	Kesme Tanımı, Bit 1
1	Kesme Tanımı, Bit 0
0	Kesme Sırada = 0

Tablo 2.1.6 Kesme Tanım Saklayıcısı (IIR).

Bit(7-3) : Saklıdır ve daima 0 olmalıdır.

Bit(2,1) : Bu iki bit, Tablo 2.1.7'de gösterilen en öncelikli oluşmuş kesmeyi gösterir.

Bit(0) : Bu bit 1 olduğunda, herhangi bir kesme oluşmamış demektir. Bit 0 olduğunda ise kesme askıda olup, 2,1 bitleri kesme yordamına bir gösterge olarak

kullanılabilir.

Bitler 2 1 0	Öncelik	Tip	Neden	Kesme Cevap Kontrolü
0 0 1	-	Yok	Yok	-
1 1 0	En yüksek	Alıcı Hat Durumu	Overrun, Parity veya Framing hatası veya Durma Kesmesi	LSR Oku
1 0 0	İkinci	Alınmış Veri Hazır	Alıcı Tamponunda Veri	RBR Oku
0 1 0	Üçüncü	Verici Tutucu Sak. Boş	Veri Tutucu Sak. Boş	IIR Oku veya THR'a Yaz
0 0 0	Dördüncü	Modem Durumu	Modem sinyal durumunda değişme	MSR Oku

Tablo 2.1.7 Kesme Kontrol Fonksiyonları.

Hat Kontrol Saklayıcısı (hex nFB)

(Line Control Register - LCR)

Asenkron iletişim formatı, Hat Kontrol Saklayıcısı yoluyla yapılır.

Bit	Fonksiyon
7	Bölücü Tutucu Erişim Biti
6	Durmayı Belirle
5	Çubuk Eşliği
4	Tek Eşlik Seçimi
3	Eşlik İzni
2	Durma Bit Sayısı
1	Sözcük Genişliği Seçimi, Bit 1
0	Sözcük Genişliği Seçimi, Bit 0

Tablo 2.1.8 Hat Kontrol Saklayıcısı (LCR).

Bit(7) : Bu bit baud oranı bölücü saklayıcısına erişilirken 1 durumunda olmalıdır. RBR ve THR veya IER ile işlem yapılırken ise, 0 olmalıdır.

Bit(6) : Bu bit 1 yapıldığında, durma-belirle (set-break) aktif yapılır. 0 ise durma-belirle pasif yapılır.

Bit(5) : 5,4 ve 3. bitler 1 olduğunda, eşlik gönderilir ve lojik sıfır olarak kontrol edilir. 5 ve 3. bitler 1, ve 4. bit 0 ise, eşlik gönderilir ve lojik 1 olarak kontrol edilir.

Bit(4) : Bu bit ve 3. bit 1 ise, çift sayıdaki lojik birler, veri sözcüğündeki bitler ve eşlik biti gönderilir ve kontrol edilir. Bu bit 0 ve 3. bit 1 ise, tek sayıdaki lojik birler, veri sözcüğündeki bitler ve eşlik biti gönderilir ve kontrol edilir.

Bit(3) : 1 ise, son veri biti ile durma biti arasında eşlik oluşturulur(gönderilen veri) veya kontrol edilir(alınan veri). (Eşlik biti, veri sözcük bitleri ve eşlik biti toplandığında, birlerin çift veya tek sayısını oluşturmak için kullanılır.)

Bit(2) : Bu bit 0 ve 1. bitlerle, her seri karakter gönderildiğinde veya alındığında oluşacak durma bitlerinin sayısını Tablo 2.1.9'da olduğu gibi belirtir.

Bit 2	Sözcük Genişliği *	Durma Bit Sayısı
0	N/A	1
1	5-Bit	1-1/2
1	6-Bit	2
1	7-Bit	2
1	8-Bit	2
* Sözcük genişliği bu saklayıcının 1 ve 0. bitleri ile belirtilir.		

Tablo 2.1.9 Durma Bitleri.

Bit(1,0) : Bu iki bit, gönderilen veya alınan her seri karakterin bit sayısını gösterir. Sözcük genişliği Tablo 2.1.10'daki gibi seçilir.

Bit(0) : Bu bit, modem kontrol çıkış sinyali olan veri

Bit 1 0	Sözcük Genişliği
0 0	5-Bit
0 1	6-Bit
1 0	7-Bit
1 1	8-Bit

Tablo 2.1.10 Sözcük Genişliği.

Modem Kontrol Saklayıcısı (hex nFC)

(Modem Control Register - MCR)

Bu 8 bitlik saklayıcı, modem, veri kümesi veya modemi emule eden bir çevre birimi ile veri değişimini sağlar.

Bit	Fonksiyon
7-5	Saklıdır = 0
4	Çevrim
3	Out2
2	Out1
1	RTS (Request to send)
0	DTR (Data terminal ready)

Tablo 2.1.11 Modem Kontrol Saklayıcısı(MCR).

Bit(7-5) : Saklıdır ve daima 0 olmalıdır.

Bit(4) : Bu bit, seri portun genel amaçlı teşhis testi için geri çevrimini içerir.

Bit(3) : Bu bit, kesme izin sinyali anlamına gelen, 'output 2' (OUT2) sinyalini kontrol eder. OUT2, kanalın kesme sinyalini kontrol etmek için kullanılır. Bu bitin 1 yapılması kesmeye izin verir, 0 yapılması kesmeyi maskeler.

Bit(2) : Bu bit, kullanıcıya tahsis edilmiş yedek çıkış sinyali olan 'output 1' (OUT1) sinyalini kontrol eder. 1 olursa, OUT1 aktif, 0 olursa OUT1 pasif olur.

Bit(1) : Bu bit, modem kontrol çıkış sinyali olan gönderme isteğini(RTS) kontrol eder. 1 yapılırsa RTS aktif, 0 yapılırsa RTS pasif olur.

Bit(0) : Bu bit, modem kontrol çıkış sinyali olan veri

alıcı hazırı(DTR) kontrol eder. 1 yapılırsa DTR aktif, 0 yapılırsa DTR pasif olur.

Hat Durum Saklayıcısı (hex nFD)

(Line Status Register - LSR)

Bu 8 bitlik saklayıcı, veri transferi hakkında sistem mikroişlemcisine bilgi verir.

Bit	Fonksiyon
7	Saklıdır = 0
6	Verici Kaydırma Sak. Boş
5	Verici Tutucu Sak. Boş
4	Durma Kesmesi
3	Framing Hatası
2	Eşlik Hatası
1	Overrun Hatası
0	Veri Hazır

Tablo 2.1.12 Hat Durum Saklayıcısı(LSR).

Bit(7) : Saklıdır ve daima 0 olmalıdır.

Bit(6) : Bu bit, verici tutucu saklayıcısı ve verici kaydırma saklayıcısının her ikisi de boşsa 1 olur. Bu saklayıcılardan herhangi biri bir veri karakteri içerirse 0 olur.

Bit(5) : Bu bit, seri port kontrolörünün, göndermek için yeni bir karakter alabileceğini gösterir.

Bit(4) : Bu bit, alınmakta olan verinin tam sözcük alınmadan yarıda kesildiğini gösterdiğinde 1 değerini alır.

Bit(3) : Bu bit, son veri biti transferinden sonra, kontrol bitleri alınmadan işlemin durdurulduğunu gösterir. Bu bit, alınan karakterin uygun bir durma biti olmadığını gösterir.

Bit(2) : Bu bit, bir eşlik hatası sezildiğinde 1 olur. Bu bit, sistem mikroişlemcisi hat durum saklayıcısını okuduktan sonra sıfırlanır.

Bit(1) : Bu bit, alınmış veri sistem mikroişlemcisi

tarafından okunmadan önce başka bir verinin alındığını göstermek için 1 yapılır.

Bit(0) : Bu bit, RBR'ye tam olarak bir karakter okunduktan sonra, 1 yapılır. RBR okunursa, bu bit sıfırlanır.

Modem Durum Saklayıcısı (hex nFE)

(Modem Status Register - MSR)

Bu saklayıcı, modemden sistem mikroişlemcisine kadar olan kontrol hatlarının halihazırdaki durumunu verir. İlave olarak, 0-3. bitler değişim bilgisini verirler. Bu dört bit, modem kontrol girişlerinden biri değiştiğinde 1 olurlar. Sistem mikroişlemcisi tarafından saklayıcı okunduğunda sıfırlanırlar.

Bit	Fonksiyon
7	Data-Carrier-Detect
6	Ring Indicator
5	Data-Set-Ready
4	Clear-to-Send
3	Delta-Data-Carrier-Detect
2	Trailing Edge Ring Indicator
1	Delta-Data-Set-Ready
0	Delta-Clear-to-Send

Tablo 2.1.13 Modem Durum Saklayıcısı(MSR).

Bit(7) : Bu bit 'data-carrier-detect' modem kontrol giriş sinyalidir. Eğer MCR'nin 4. biti 1 yapılırsa, bu bit MCR'nin 3. bitine eşit olur.

Bit(6) : Bu bit, 'ring-indicator' modem kontrol giriş sinyalidir. Eğer MCR'nin 4. biti 1 yapılırsa, bu bit MCR'nin 2. bitine eşittir.

Bit(5) : Bu bit, 'data-set-ready' modem kontrol giriş sinyalidir. Eğer MCR'nin 4. biti 1 yapılırsa, bu bit MCR'nin 0. bitine eşittir.

Bit(4) : Bu bit, 'clear-to-send' modem kontrol giriş

sinyalidir. Eger MCR'nin 4. bitini 1 yapırsa, bu bit MCR'nin 1. bitine eşittir. kablo bağlantısı 9-pin veya 25-pin D tipi

Bit(3-0) : Yukarıda anlatıldığı gibi olup, Bit(7-4)'e karşılık düşer. Mikroişlemci tarafından en son karşılık düşen bit okunduğundan beri konum değiştiğini göstermek için 1 yapılır.

Tırmık Saklayıcısı

(Scratch Register - SR)

Bu saklayıcı, seri portu kontrol etmez. Yalnızca, mikroişlemcinin geçici olarak veri tutmasını sağlar.

Şekil 2.2.1 9/25 Pin D-tipi konektörler

RS232C Üç Kontrol Sinyalleri (25 pin)

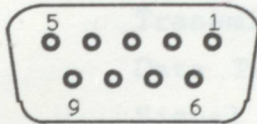
Pin no Anlamı

1	Protective Ground
2	Transmitted Data
3	Received Data
4	Request To Send
5	Clear To Send
6	Data Set Ready
7	Signal Ground
8	Received Line Signal Detector
9-10	
11	Select Standby
12-14	
15	Transmit Signal Element Timing
16	
17	Receive Signal Element Timing
18	Test
19	
20	Data Terminal Ready
21	

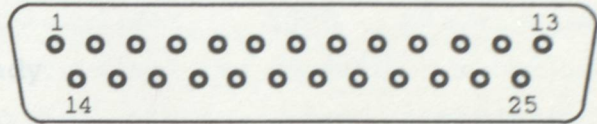
2.2 RS232 Kablo Bağlantısı

Genel olarak RS232 kablo bağlantısı 9-pin veya 25-pin D tipi konnektörlerle yapılır. Bu bölümde, her konnektör tipinin bağlantıları verilmiştir. Ayrıca, tezin çalışabilmesi için yapılması gereken minimum bağlantı şekli de verimştir.

9-Pin D tipi konnektör



25-Pin D tipi konnektör



Şekil 2.2.1 9/25 Pin D tipi konnektörler.

RS232 Uç Gösterilimleri (25 pin)

Uç no Anlamı

1	Protective Ground
2	Transmitted Data
3	Received Data
4	Request To Send
5	Clear To Send
6	Data Set Ready
7	Signal Ground
8	Received Line Signal Detector
9-10	-
11	Select Standby
12-14	-
15	Transmit Signal Element Timing
16	-
17	Receive Signal Element Timing
18	Test
19	-
20	Data Terminal Ready
21	-

- 22 ~~alıklar~~ Ring Indicator
 23 Speed Select
 24-25 -

RS232 Uç Gösterilimleri (9 pin)

Uç no	Anlamı
1	Carrier Detect
2	Receive Data
3	Transmit Data
4	Data Terminal Ready
5	Signal Ground
6	Data Set Ready
7	Request To Send
8	Clear To Send
9	Ring Indicator

2.3 80X86 Ailesi Mikroişlemcilerde Kesmeler

2.3.1 Kesmeler Hakkında Genel Bilgi

Bilindiği gibi 8086 ailesi (80X86) mikroişlemcilerinde üç ana kesme sınıfı mevcuttur:

- * Dahili Donanım Kesmeleri
- * Harici Donanım Kesmeleri
- * Yazılım Kesmeleri

Dahili donanım kesmeleri, mikroişlemcinin yürütülen komutlarında oluşabilecek bazı özel durumlardaki kesmelerdir. Bu tip kesmelere, sıfıra bölme hatası, tek adım (single step) komut yürütme, maskelenemez kesme, durma noktası (break point), taşma gibi durumlarda oluşan kesmeleri örnek olarak gösterebiliriz. Dahili donanım kesmeleri 8086'da 5, 80286'da 14 çeşittir. Bu kesmelerin herbiri farklı düzeylerde oluşur ve birbirlerine göre

öncelikleri sözkonusudur.

Harici donanım kesmeleri, mikroişlemciye bağlı cihazlardan (ana devre üzerinde veya kart olarak bağlı) bir kesme sinyali gelmesi ile oluşur. Bu kesmeler için iki bağlantı şekli mevcuttur. Birincisi maskelenemez kesme (NMI), ikincisi maskelenebilir kesmedir (INTR). Maskelenemez kesmelere (NMI) örnek olarak, bazı sistemlerdeki fiziksel reset anahtarlarını gösterebiliriz. INTR kesmelerini, kesmeleri yazılımla kontrol edebilmek ve öncelik verebilmek için, 8259A PIC entegre devresi kontrol eder. Mikroişlemci komutları, direk olarak kesmelere izin verebilir veya maskeleyebilir (STI, CLI), PIC'e gönderilecek komutlar bazı kesmelere izin verebilir veya maskeleyebilir. Kesmeler, bu nedenle donanım düzeyinde belirlenir. Bazı durumlarda chip-üreticileri kesme düzeylerini donanım olarak belirlerler ve bu düzeyleri hiçbirşey değiştiremez.

Yazılım kesmeleri, mikroişlemciye sanki bir donanım kesmesi oluşmuş gibi, yazılım kesme komutunu (INT nnn) yürütmesini sağlar. Bunlar herhangi bir programdan DOS ve BIOS servis programlarının INT komutu ile çağırılmasıyla oluşurlar.

2.3.2 Kesme Vektörleri

Kesme vektörleri, sistem hafızasının en dibindeki (0.segment, 0.offset) 1,024 byte'lık bölgede bulunurlar. Herbir kesme vektörü, 4 byte (2 byte segment, 2 byte offset) yer kaplar ve 256 ayrı kesme vektörü tanımlanabilir.

Kesme vektörlerinin elde edilmesi veya belirlenmesi direk olarak bellekten yapılabileceği gibi "AH=35h vektörü getir" (get vector) ve "AH=25h vektörü belirle" (set vector) DOS fonksiyon çağırımları kullanılarak da yapılabilir.

2.4 RS232 ile İlgili Kesmeler Ve Bunları Sezme

Kişisel bilgisayarlarda (PC ve PS/2) RS232 iletişimi sağlayan Asenkron İletişim Kontrolör entegre devreleri mevcuttur. Bu entegreler, 80x86 mikroişlemcilerine kesme üretebilmektedirler. Bu nedenle Harici Donanım Kesmeleri tipinde kesme oluştururlar.

Bilindiği gibi, harici donanım kesmeleri, kesme düzeylerine sahiptirler. Asenkron İletişim Kontrolörünün oluşturduğu kesmeler için genel olarak IRQ3 (Interrupt Request - Kesme Düzeyi) ve IRQ4 kesme düzeyleri ayrılmıştır. Bu kesme düzeyleri sabit olup herhangi bir şekilde değiştirilemez. IRQ4, (1) numaralı seri port için, IRQ3 iki (2) numaralı seri port için ayrılmıştır.

Kontrolörün kesme oluşturmaya mücade etmek için Modem Kontrol Saklayıcısının (Modem Control Register - MCR) 3. biti Out2=1 yapılmalıdır. Daha sonra, Kesme İzin Saklayıcısı (Interrupt Enable Register - IER) yardımıyla hangi tür kontrolör kesmelerine izin verileceği belirtilir. Bilgisayarın, kesme oluştuğunda hangi vektör numarasına ait adresi çağıracağı da, o kesme düzeyine ait kesme vektör adresinin belirtilmesiyle mümkün olabilir. Kesme vektör numaraları makina tiplerine göre (PC-XT, PC-AT, PS/2) farklılık göstermektedir ve Tablo 2.4.1'de gösterilmiştir.

Kesme Vektör No.	PC/PC-XT	PC-AT/PS/2
0Bh	COM1/COM3	COM2/COM4
0Ch	COM2/COM4	COM1/COM3

Tablo 2.4.1 Makina tipleri ve seri portlara göre kesme vektörleri.

Kesme izin saklayıcısı yardımıyla, modem durumu değiştiğinde, alıcı-hat durumu değiştiğinde, verici tutma saklayıcısı boş olduğunda ve alınmış veri hazır durumunda kesme oluşturabilmek mümkündür. Bu tezde ihtiyaç duyulan kesme, yalnızca veri hazır kesmesidir. Birbirine bağlı iki bilgisayar arasında bağlantı kurarken, birisi diğerinin işini, gerektiğinde başka bir istekle kesebilmelidir. Bu durumda, terminal durumunda olan bilgisayarın master durumunda olan bilgisayara isteğini belirten bir kod (1 byte) göndermesi gereklidir. Master durumundaki bilgisayar da bunu ancak kesme ile sezebilir. Eğer kesme kullanılmasaydı, bu takdirde isteği içeren verinin gelip gelmediği ancak asenkron iletişim kontrolörünün saklayıcılarını sürekli okumakla yapılabilirdi. Bu ise hiç de efektif olmayan bir yoldur.

Asenkron iletişim kontrolörünü kesme için programlamak, kesmelerin oluşması için yeterli değildir. Bunun için ayrıca, sistemin kesme maske saklayıcısının ilgili IRQ'ya (kesme isteği) izin vermesini sağlamak gereklidir. IMR (Interrupt Mask Register - Kesme Maske Saklayıcısı) 21h numaralı adreste olup, 1 byte'lık bir veriyi içerir. Bu 8-bit verinin her bir biti, bitlere karşılık düşen (4.bit = IRQ4) kesme maskelerini verir. Kesme maske biti 0 ise kesme düzeyine izin verilmiş demektir. (Örneğin IRQ4'e mücade etmek için IMR(21h) port adresindeki veri 11101111B sayısı ile veyalanmalıdır.)

PC, PC-XT ve PC-AT uyumlu sistemlerde kesmeye izin verme, yalnızca kesme maske saklayıcısının (IMR) belirlenmesi ile yapılabilir. Ancak PS/2 ailesi bilgisayarlarda, donanım farklılığından dolayı bazı ekstra işlemlerin yapılması gerekmektedir. Bu işlemler de, her kesme düzeyine ait özel kesme sonu sayısının (IRQx için 6xh) işlemsel kontrol

saklayıcısına (Operational Control Register - OCR) gönderilmesinden ibarettir. Bu durumda donanım, asenkron kontrolörden gelecek kesmeleri oluşturabilecektir.

PC ve PC-AT uyumlu bilgisayarlarda kesme yordamına gelindiğini ve kesme talebinin kabul edildiğini belirtmek için, sistem kesme kontrolörüne(System Interrupt Controller(20h) - SIC) 20h kodu yazılır. Diğer bir adım, kesme yordamına gelindiğinde, gerçekten kesmenin oluşup oluşmadığının kontrolüdür. Bu da asenkron iletişim kontrolörünün, Kesme Tanım Saklayıcısını(IIR) kullanarak yapılabilir. Eğer gerçekten kesme oluşmuşsa (IIR'ın 0. biti = 0) bu halde kesmenin hangi durumdan dolayı oluştuğuna karar vermek için tekrar IIR kullanılır. Daha sonra da kesme yordamı çağırılır. Kesme yordamından dönerken, yalnızca PS/2 ailesi bilgisayarlarda kesmenin bittiğine her IRQ için özel kesme sonu işareti gönderilir ve daha sonra gelecek kesmelere izin verilir.

2.5 MS-DOS İşletim Sisteminde Cihaz Sürücüler

Daha önceki işletim sistemlerinde, cihazlara ait bütün programlamalar donanım düzeyinde yapılıyordu. Bilgisayarlar geliştğinde görüldü ki, bu tür programlar bazı tekrarlı işleri yapmakta ve sonuç olarak aynı veriler ortaya çıkmaktaydı. Bunun üzerine cihazlara ait standart yordamlar (giriş/çıkış gibi) belirlendi ve her cihaz için aynı standart işleri yapan **cihaz sürücüler** ortaya çıktı.

Cihaz sürücüler, ilk günlerde sistem programlarının birer parçası olarak yazılmaktaydı. Daha sonra, cihaz sürücüler sistemden ayrı ve bağımsız olarak tasarlanıp kullanılmaya başlandı. Böylece kullanıcıların cihaz sürücülerini kurabilme esnekliği sağlanmış oldu.

veya disket sürücüsüne hizmet verebilirler. Örneğin, ilk blok MS-DOS işletim sisteminde ise cihaz sürücüler CONFIG.SYS dosyası içine DEVICE=isim olarak yazılabilir. Bilgisayar ilk açıldığında, her cihaza ait cihaz sürücüler işletim sistemi tarafından kurulur ve gerektiğinde bazı işletim sistemi tarafından erişilebilir.

Cihaz sürücülerin yazıldığı dil mutlaka Assembly dili olmalıdır. Çünkü, cihaz sürücüler çağırıldığında, bütün mikroişlemci saklayıcılarını korumak durumundadır. Bunu C dilinden yapmak oldukça zordur. Cihaz sürücülerini C dili ile yazmak UNIX işletim sisteminde yaygındır.

Sonuç olarak cihaz sürücüler iki gruba ayrılmışlardır. Bunlar karakter bazında işlem yapabilen karakter cihazlar ve blok bazında işlem yapabilen blok cihazlardır.

2.5.1 Karakter Cihazlar

Karakter cihazlar, CON, AUX ve PRN gibi seri halde karakter G/Ç yapmak için tasarlanmıştır. Bu cihazlar, CON, AUX, CLOCK\$ gibi isimlere sahiptirler ve bunlara giriş/çıkış yapmak için kanallar açabilirler. Karakter cihazların yalnızca bir ismi olması nedeniyle bir cihazı destekleyebilir.

2.5.2 Blok Cihazlar

Blok cihazlar, sistemdeki sabit disk veya disket sürücülerdir. Bunlar blok diye adlandırılan genellikle diskin fiziksel sektör uzunluğu olan parçalarda rasgele G/Ç yapabilirler. Bu cihazlar, karakter cihazlarda olduğu gibi isimlendirilemezler ve direk olarak açılmazlar. Bunun yerine, A,B,C ve bunun gibi cihaz harflerini kullanarak erişilirler. Blok cihazlar içlerinde birimler tutabilirler. Bu nedenle bir tek blok sürücü, bir veya daha fazla disk

veya disket sürücüye hizmet verebilirler. Örneğin, ilk blok cihaz sürücü A,B,C ve D ünitelerine cevap verebilir. Bu, tanımlı dört birime sahip olduğunu ve böylece dört cihaz harfi alacağını gösterir. Bütün sürücüler zincirinde, sürücünün pozisyonu, sürücü üniteleri ve sürücü harflerine karşılık gelen yola karar verir. Örneğin, eğer cihaz zincirindeki cihaz sürücü ilk blok sürücü ise ve dört ünite tanımlarsa, o zaman bu üniteler A,B,C ve D'dir. Eğer ikinci blok sürücü üç birim tanımlarsa, o zaman üniteler E,F ve G olur. Cihazlara atanan A'dan Z'ye harflerle 26 cihaz sürücü limiti vardır.

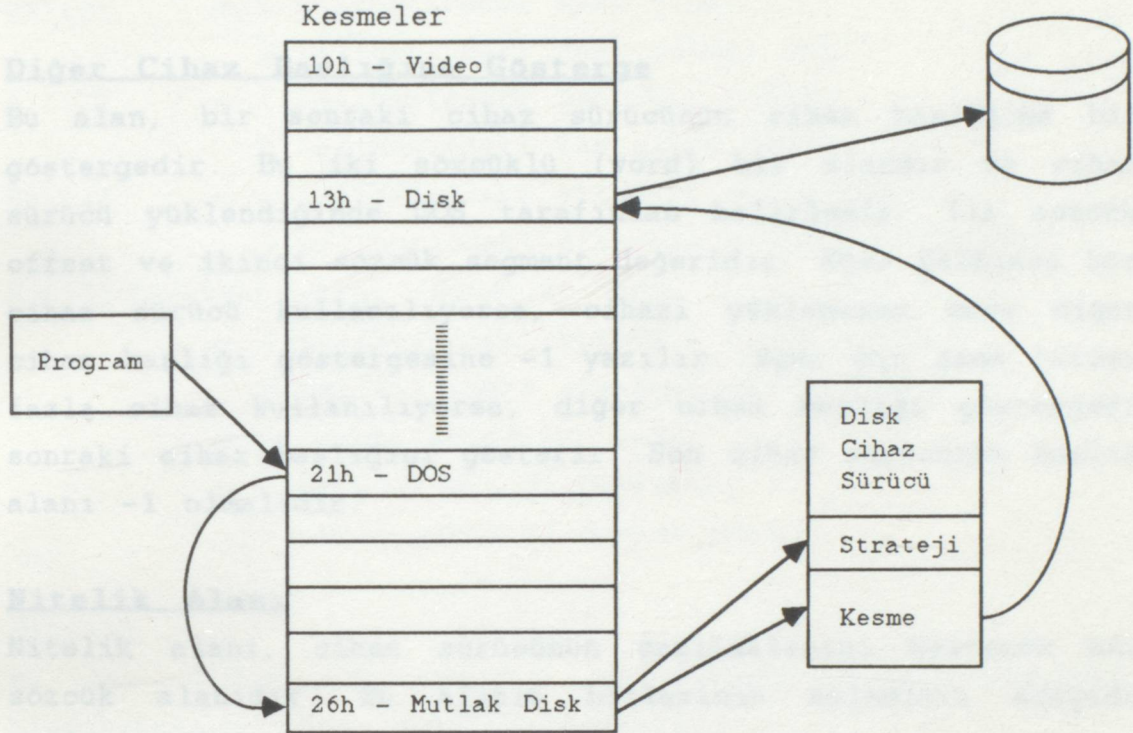
2.5.3 Cihaz Sürücüler Nasıl Çalışır?

Bir aplikasyon programı DOS Int 21h'i çağırdığında birkaç fonksiyon hariç cihaz sürücüler çağırılır ve istenen işlem cihaz sürücüler yardımıyla gerçekleşir. Bunu, disk üzerindeki bir dosyaya yazma örneği vererek açıklayalım.

Bir dosya ile ilgili giriş/çıkış yapmak istenirse, ilk olarak gerekli saklayıcı belirlemeleri yapılarak DOS Int 21h fonksiyonu çağırılır. DOS kontrolü aldığı anda gerekli saklayıcı belirlemelerini yaparak Int 26h'i (mutlak disk yazma) çağırır. Int 26h saklı bir bellek bölgesinde, bir istek başlığı oluşturarak diskin cihaz sürücüsünün strateji yordamını çağırır. Strateji yordamı, basit olarak istek başlığının adresini saklar.

Daha sonra, cihaz sürücünün kesme yordamı çağırılır. Kesme kısmı, istek başlığını okur ve ne istenildiğine karar verir. Kesme kısmı daha sonra, BIOS disk yazma fonksiyonu Int 13h'i çağırır. Disk yazma işlemi bittiğinde, kontrol anlatılan zincirden aplikasyon programına döner. Şekil 2.5.1 bu işlemi açıklamaktadır.

Tablo 2.5.1 Cihaz Başlığı



Şekil 2.5.1 Disk Yazma İçin Cihaz Sürücünün Çağırılması.

2.5.4 Cihaz Sürücü Detayları

Bu bölümde cihaz sürücülerle ilgili ayrıntılı bilgiler bulunmaktadır.

Cihaz Başlığı

Bir cihaz sürücüsü, dosyanın başında bir cihaz başlığı gerektirir. Tablo 2.5.1'de başlığın içeriği açıklanmıştır.

Alan	Uzunluk
Diğer cihaz başlığına gösterge	DWORD
Nitelik	WORD
Cihaz strateji yordamına gösterge	WORD
Cihaz kesme yordamına gösterge	WORD
İsim/ünite alanı	8 BYTE

Tablo 2.5.1 Cihaz Başlığı.

Diğer Cihaz Başlığına Gösterge

Bu alan, bir sonraki cihaz sürücünün cihaz başlığına bir göstergedir. Bu iki sözcüklü (word) bir alandır ve cihaz sürücü yüklendiğinde DOS tarafından belirlenir. İlk sözcük offset ve ikinci sözcük segment değeridir. Eğer yalnızca bir cihaz sürücü kullanılıyorsa, cihazı yüklemeyen önce diğer cihaz başlığı göstergesine -1 yazılır. Eğer bir anda birden fazla cihaz kullanılıyorsa, diğer cihaz başlığı göstergesi sonraki cihaz başlığını gösterir. Son cihaz sürücünün başlık alanı -1 olmalıdır.

Nitelik Alanı

Nitelik alanı, cihaz sürücünün özelliklerini gösteren bir sözcük alanıdır. Bu alanın bitlerinin anlamları aşağıda açıklanmıştır.

- Bit(15) = 1 karakter cihaz
0 blok cihaz
- Bit(14) = 1 IOCTL destekler
0 IOCTL desteklemez
- Bit(13) = Blok cihazlar için,
= 1 non-IBM formatı
0 IBM formatı
Karakter cihazlar için,
= 1 meşgul olana kadar çıkışı destekler
0 meşgul olana kadar çıkışı desteklemez
- Bit(12) = 0 DOS tarafından saklıdır
- Bit(11) = 1 taşınabilir cihazı destekler
0 taşınabilir cihazı desteklemez
- Bit(10-7) = 0 DOS tarafından saklıdır
- Bit(6) = 1 Lojik Cihazı Belirle/Getiri destekler
0 Lojik Cihazı Belirle/Getiri desteklemez
- Bit(5-4) = 0 DOS tarafından saklıdır

- Bit(3) = 1 halihazırdaki saat cihazı
 0 halihazırdaki saat cihazı değil
- Bit(2) = 1 halihazırdaki boş(NUL) cihazı
 0 halihazırdaki boş cihaz değil
- Bit(1) = 1 halihazırdaki standart çıkış cihazı
 0 halihazırdaki standart çıkış cihazı değil
- Bit(0) Karakter cihaz sürücüler için,
 = 1 halihazırdaki standart giriş cihazı
 0 halihazırdaki standart giriş cihazı
- Blok cihaz sürücüler için,
 = 1 Genel IOCTL'yi destekler
 0 Genel IOCTL'yi desteklemez

Bit(15)

Bu bit cihaz tip bitidir ve sisteme cihaz sürücünün blok/karakter cihaz olduğunu gösterir.

Bit(14)

Bu bit IOCTL (G/Ç kontrolü) bitidir. Hem karakter hem de blok cihazlar için kullanılır. Bit 14, cihaz sürücünün IOCTL fonksiyon çağırması(44h) yoluyla karakter katarlarını tutup tutamayacağını DOS'a belirtmek için kullanılır. Bit 14, yalnızca IOCTL'un AL=2 ile AL=5 arasındaki fonksiyonlarını etkiler.

IOCTL fonksiyon çağırımları, gerçekte normal bir okuma veya yazma yapmadan, cihaz sürücüye bilgi gönderme veya almaya imkan verir. Bu yolla, cihaz sürücü veriyi kendisi için kullanır (örneğin baud oranı, eşlik biti seçimi gibi). Gönderilen bilgi cihaz sürücüde kalır herhangi bir gönderme işlemi yapılmaz.

Bit(13)

Blok cihaz sürücüler için, bu bit sürücünün cihaz tipine karar vermek için kullandığı metodu gösterir. Karakter cihaz

sürücüler için meşgul oluncaya kadar çıkışı destekleyip desteklemeyeceğini gösterir. Eğer bir blok cihaz sürücü, cihaz tipine karar vermek için BPB(BIOS Parametre Bloğu) bilgisini kullanacaksa, bit 13 = 1 olacaktır. Eğer cihaz sürücü, cihaz tipine karar vermek için cihaz tanım byte'ını kullanacaksa bit 13 = 0 olacaktır.

Karakter cihaz sürücüler için meşgul oluncaya kadar çıkış, yazıcı cihaz sürücüleri tarafından kullanılır. Meşgul oluncaya kadar çıkış, cihaz sürücünün, karakteri eğer cihaz hazırsa cihaza göndermesi demektir.

Bit(11)

Bu bit, cihazın taşınabilir olup olmadığını gösterir. Cihaz taşınabilirse 1 değerini alır.

Bit(3)

Bu bit, saat cihaz bitidir. Yalnızca karakter cihazlar için kullanılır. 1 değerini alırsa cihaz sürücü yeni CLOCKŞ cihazı demektir.

Bit(2)

Bu bit NUL nitelik bitidir. Yalnızca karakter cihazlar için kullanılır. 1 değerini alırsa boş cihaz demektir.

Bit(1-0)

Karakter cihazlar için bit 1 ve bit 0 standart çıkış/giriş bitleridir. Cihazın yeni standart g/ç cihazı olduğunu göstermek için kullanılır. Blok cihazlar için, bit 0 = 1 (bit 15 = 0) ise, Genel IOCTL destekleniyor demektir.

Strateji ve Kesme Yordamlarına Gösterge

Bu iki alan strateji ve kesme yordamlarının giriş noktalarına göstergedir. Bunlar sözcük değerlerdir ve bu nedenle cihaz başlığı ile aynı dilimde (segment)

bulunmalıdırlar. Ancak bu yordamlar çağırılırken FAR çağırma kullanılır.

İsim/Ünite Alanı

Bu alan, karakter cihazın ismini veya blok cihazın ünitesini kapsayan 8-byte'lık bir alandır. Karakter cihazlar için, isim sola dayalıdır ve 8 byte'a boşluklarla tamamlanır. Blok cihazlar için, ilk byte, kapsanan ünite sayısını gösterir. Bu, isteğe bağlıdır, çünkü DOS, sürücünün INIT kodu ile dönen değeri bu alanın başına yazar.

Bir Cihaz Sürücü Yaratma

DOS'un kurabileceği bir sürücüyü yaratmak için, aşağıdakileri yerine getirmek gerekir:

- * Dosyanın başında bir cihaz başlığı ile bir bellek dosyası veya bir .EXE dosyası yaratın.
- * Cihaz başlığını kapsayan kodun başlangıç adresini, 100h değil 0 olarak belirleyin (ORG 0).
- * Diğer cihaz başlık alanını belirleyin. Detaylı bilgi için "Diğer Cihaz Başlığına Göstege" kısmına bakınız.
- * Cihaz başlığının nitelik alanını belirleyin. Detaylı bilgi için "Nitelik Alanı" kısmına bakınız.
- * Kesme ve strateji yordamları için giriş noktalarını belirleyin.
- * İsim/Ünite alanını karakter cihazın ismi veya blok cihazın ünite sayısı ile doldurun.

DOS, halihazırda tanımlanmış karakter cihazların tekrar tanımlanmasına imkan verir. Örneğin yeni bir CON cihazı yaratılmak istenirse, cihaz isminin yalnızca CON yapılması yeterli olacaktır. Ayrıca sürücü, hafızada herhangi bir yere yerleşebilecek şekilde yazılmalıdır.

Cihaz Sürücülerini Kurmak

DOS, yeni cihaz sürücülerini dinamik olarak CONFIG.SYS dosyası içerisindeki DEVICE komutunu okuyarak ve işleyerek sistemin açılması (boot) sırasında kurar. Örneğin, DRIVER1 isminde yeni bir cihaz sürücüsü yazılmışsa, onu kurmak için aşağıdaki komutu CONFIG.SYS dosyasının içine koymak gerekir :

```
DEVICE=DRIVER1
```

DOS, bir cihaz sürücüyü ilk olarak, DOS'un sürücülerden ne yapmasını istediğini tanımlayan bilgiyi başlık isteğinde vererek, strateji yordamını çağırır.

Strateji yordamı, isteği icra etmez fakat, bunun yerine isteği kuyruklar veya başlık isteğine bir göstergeyi saklar. İkinci giriş noktası, kesme yordamıdır ve strateji programından döndükten hemen sonra DOS tarafından çağırılır. Kesme yordamı parametresiz olarak çağırılır. Fonksiyonu, kuyruklanmış isteklere dayandırılan işlemi icra etmek ve herhangi bir dönüş bilgisini belirlemektir.

DOS, başlık isteğine göstergeyi ES:BX içerisinde gönderir. Bu gösterge, icra edilecek işlemle ilgili hemen ardından verinin geldiği sabit uzunluklu bir başlığı (başlık isteği) kapsar. Cihaz sürücüsü makina durumunu ve bütün saklayıcı içeriklerini korumalıdır.

DOS tarafından kullanılan yığıt (stack), bütün saklayıcıları korumak için yeterli yere sahiptir. Eğer daha fazla yığıta ihtiyaç varsa, başka bir yığıt organize etmek ve ayırmak cihaz sürücüsünün sorumluluğundadır.

Cihaz sürücülere olan bütün çağrılar FAR tipindedir. Bu yüzden DOS'a dönmek için FAR dönüş kullanılmalıdır.

Karakter Cihazları Kurmak

Her cihaz için tanımlanan fonksiyonlardan biri INIT'dir. Bu yordam, yalnızca bir defa çağırılır ve tekrar asla çağırılmaz. INIT yordamı aşağıdaki bilgileri verir:

* Bitiş adres alanında saklanan bir dur ve kalıcı ol gibi, cihaz sürücünden sonraki ilk boş bellek byte'ına bir gösterge verir. Bu yolla, ilk koşullandırma kodu bir kere kullanılabilir ve yerden kazanmak için serbest bırakılabilir.

* Bitiş adres alanını belirledikten sonra, bir karakter cihaz sürücünün durumunu belirleyip dönebilir.

Blok Cihazları Kurmak

Blok cihazlar, karakter cihazların kurulduğu yolla kurulur. Fark yalnızca, blok cihazların ilave bilgi dönmesidir. Blok cihazlar ek olarak aşağıdakileri dönmelidir:

* Blok sürücü için ünite sayısı. Bu sayı cihazların sahip olacağı lojik isimlere karar verir. Örneğin INIT çağırması anındaki maksimum lojik cihaz harfi F ise, ve blok cihaz sürücüsünü INIT yordamı üç lojik ünite dönerse, cihazların lojik isimleri G,H ve I olur. Lojik isimler, cihaz sürücünün cihaz zincirindeki pozisyonuna göre belirlenir. INIT tarafından dönen ünite sayısı, cihaz başlığının isim/ünite alanındaki değerin üzerine yazar.

* BPB (BIOS Parametre Bloğu) gösterge dizisine bir gösterge. n tanımlanan ünite sayısı olmak üzere, bu bir n sözcük göstergesine bir göstergedir. Bu sözcük göstergeler BPB'lere gösterir. Eğer, bütün üniteler aynı ise, bütün dizi, boşluktan kazanmak için aynı BPB'ye gösterebilir. BPB,

sektör büyüklüğü, yer ayırma birimi başına sektörü ve bunun gibi cihazla ilgili bilgiyi kapsar. BPB içindeki sektör büyüklüğü, DOS ilk koşullandırma zamanında belirlenen maksimum tayin edilen büyüklükten fazla olamaz. (Bu dizi, dönüş tarafından belirlenen boş göstergenin aşağısında gizlenmelidir.

* Cihaz ayırıcı byte'ı. Bu byte, cihazlara geçer ve böylece DOS'un halihazırdaki bir kısmı sürücü ünitesi için kullandığı parametreleri içerir.

Blok cihazlar, birçok defa başlatılabilir. Bunlar aptal veya akıllı olabilir. Bir aptal cihaz, her mümkün cihaz sürücü birleşimi için bir ünite (ve ayrıca BPB) tanımlayabilirler. Ünite 0=sürücü 0; tek taraflı, ünite 1=sürücü 0; çift taraflı v.b. Bu yaklaşım için, cihaz byte'ları, hiçbirşey ifade etmeyecektir. Bir akıllı cihaz, her ünite için birçok cihaza mücadele edecektir. Bu durumda, INIT'de dönen BPB tablosu, desteklenen mümkün en büyük cihazı telif etmek için yeteri kadar büyük boşluğu tanımlamalıdır. Akıllı cihazlar, bir ünite halihazırda kullanılan cihaz hakkında bilgi geçirmek için cihaz(media) byte'ını kullanırlar.

İstek Başlığı

İstek başlığı, DOS'un cihaz sürücünden ne yapmasını istediğini tanımlayan bilgiyi geçirir. İçeriği, Tablo 2.5.2'de gösterilmiştir.

Alan	Uzunluk
Istek başlığı + istek başlığının sonundaki verinin toplam uzunluğu	BYTE
Ünite kodu. İşlemin yapılacağı alt ünite (karakter cihazlar için anlamsızdır).	BYTE
Komut kodu	BYTE
Durum	WORD
DOS için saklı alan	8 BYTE
İşleme uygun veri	Değişken

Tablo 2.5.2 İstek Başlığı.

Ünite Kod Alanı

Ünite kod alanı, bir blok cihaz içindeki isteğin hangi ünite için olduğunu belirtir. Örneğin, eğer bir blok sürücü tanımlanmış üç üniteye sahipse, ünite kod alanının mümkün değerleri 0,1 ve 2 olacaktır.

Komut Kod Alanı

Başlık isteğindeki komut kod alanı aşağıdaki değerleri alabilir:

Kod	Fonksiyon
0	INIT (ilk koşullandırma)
1	MEDIA CHECK (sadece blok cihazlar)
2	BUILD BPB (sadece blok cihazlar)
3	IOCTL giriş kontrol katarı (yalnızca IOCTL biti 1 ise çağırılır)
4	INPUT (okuma)
5	NONDESTRUCTIVE INPUT NO WAIT (Yalnızca karakter cihazlar)
6	INPUT STATUS (Yalnızca karakter cihazlar)

7	INPUT FLUSH (Yalnızca karakter cihazlar)
8	OUTPUT (yazma)
9	OUTPUT WITH VERIFY (kontrollü yazma)
10	OUTPUT STATUS (Yalnızca karakter cihazlar)
11	OUTPUT FLUSH (Yalnızca karakter cihazlar)
12	IOCTL çıkış kontrol katarı (yalnızca IOCTL biti 1 ise çağırılır)
13	DEVICE OPEN (Yalnızca bit 11=1 ise çağırılır)
14	DEVICE CLOSE (Yalnızca bit 11=1 ise çağırılır)
15	REMOVABLE MEDIA (Yalnızca bit 11=1 ve blok cihaz ise çağırılır)
19	GENERIC IOCTL REQUEST
23	GET LOGICAL DEVICE
24	SET LOGICAL DEVICE

Not: 13,14 ve 15 numaralı komut kodları 3.00'dan sonraki DOS versiyonları için geçerlidir. 19,23 ve 24 numaralı komut kodları 3.20'den sonraki versiyonlar için geçerlidir.

Durum Alanı

İstek başlığındaki durum alanı, Tablo 2.5.3'de gösterilenleri içerir. Durum sözcük alanı girişte sıfırdır ve dönüşte sürücü kesme yordamı tarafından belirlebilir.

15	14-10	9	8	7-0
H A T A	SAKLIDIR	M E Ş G U L I	Y A P I D I	HATA KODU (bit 15=1)

Tablo 2.5.3 Durum Sözcüğü.

15. bit hata (ERROR) bitidir. Eğer bitin değeri 1 ise, durum sözcüğünün düşük anlamlı 8 biti (7-0) hata kodunu belirtir.

14-10. bitler saklıdır ve 0 olmalıdır. 9. bit meşgul (BUSY) bitidir. Bu bit, sadece durum çağırması ve taşınabilir cihaz çağırması tarafından belirlenir. Daha detaylı bilgi için "STATUS" ve "REMOVABLE MEDIA" kısımlarına bakınız. 8. bit yapıldı (DONE) bitidir. Değeri 1 ise, işlem tamamlandı demektir. Sürücü yapıldı bitini çıkarken 1 yapar. Durum sözcüğünün 8 biti (7-0), 15. bit 1 ise hata kodunu belirtir. Hata kodları ve anlamları Tablo 2.5.4'de verilmiştir.

Hata(hex) Kodları	Tanım	Açıklama
00	Write protect violation	Yazma koruma ihlali
01	Unknown unit	Bilinmeyen ünite
02	Device not ready	Cihaz hazır değil
03	Unknown command	Bilinmeyen komut
04	CRC error	CRC hatası
05	Bad drive request structure len.	Yanlış sürücü isteği uzun.
06	Seek error	Seek hatası
07	Unknown media	Bilinmeyen cihaz
08	Sector not found	Sektör bulunamadı
09	Printer out of paper	Yazıcıda kağıt yok
0A	Write fault	Yazma hatası
0B	Read fault	Okuma hatası
0C	General failure	Genel başarısızlık
0D	Reserved	Saklıdır
0E	Reserved	Saklıdır
0F	Invalid disk change	Yanlış disk değişimi

Tablo 2.5.4 Hata Kodları.

Cihaz Sürücü Fonksiyonları

Bütün strateji yordamları, istek başlığını işaret eden ES:BX ile çağırılır. Kesme yordamları, strateji yordamlarının istek başlığını depoladığı kuyruktan göstergeleri alır. Başlık isteğindeki komut kodu cihaz sürücüyeye hangi fonksiyonun icra edileceğini söyler. (Bütün DWORD göstergeler önce offset sonra segment olarak yerleştirilir).

Aşağıdaki fonksiyon çağırımları parametreleri

tanımlanmıştır:

- * INIT sayısını belirle (yalnız blok cihazlar)
- * MEDIA CHECK göstergesi için
- * BUILD BPB (BIOS Parametre Bloğu) (yalnız blok cihazlar)
- * MEDIA DESCRIPTOR BYTE bit adresini belirle
- * INPUT veya OUTPUT durum sözöğünü belirle
- * NONDESTRUCTIVE INPUT NO WAIT
- * STATUS anında bir cihaz sürücüsüne CONFIG.SYS'den geçen
- * FLUSH elde stack için, BPB gösterge alanı CONFIG.SYS'de
- * OPEN veya CLOSE takip eden bilgiyi kapsayan bir tampon
- * REMOVABLE MEDIA tarihi (0 ile biten) bir satır başı (0Dh,
- * GENERIC IOCTL REQUEST, LF) ile sonlandırılır. Bu bilgi
- * GET LOGICAL DEVICE sürücünün INIT kodu tarafından sadece
- * SET LOGICAL DEVICE çağrılmaları (Int 21h) çağırılabilir

INIT Yordamı

Komut kodu = 0

Amaç : İlk koşullandırma.

ES:BX içeriği Tablo 2.5.5'de gösterilmiştir. karar verirse ve herhangi bir bellek kullanmadan işlemi kesmek isterse.

Alan	Uzunluk
Istek başlığı	13-BYTE
Unit sayısı(karakter cihazlar tarafından belirlenmez)	BYTE
Kalıcı program kodunun bitiş adresi	DWORD
BPB dizisine gösterge (karakter cihaz için belirlenmez) / geri kalan argümanlara gösterge	DWORD
3.10 ve daha yüksek versiyonlar için ünite numarası	BYTE

Tablo 2.5.5 INIT yordamı istek başlığı.

Sürücü aşağıdakileri yapmalıdır:

- * Ünite sayısını belirle (yalnız blok cihazlar)
- * BPB dizisine göstergiyi düzenle
- * İlk koşullandırma kodunu icra et (yalnız blok cihazlar)
- * Kalıcı program kodunun bitiş adresini belirle
- * İstek başlığındaki durum sözcüğünü belirle

INIT zamanında bir cihaz sürücüyü CONFIG.SYS'den geçen bilgiyi elde etmek için, BPB gösterge alanı CONFIG.SYS'de "=" operatörünü takip eden bilgiyi kapsayan bir tampon gösterir. ASCII katarı (0 ile biten) bir satır başı (0Dh, CR) ve yeni satır (0Ah, LF) ile sonlandırılır. Bu bilgi salt-okunabilir ve sürücünün INIT kodu tarafından sadece 01h-0Ch ve 30h sistem çağırımları (Int 21h) çağırılabilir.

Son byte'ta bulunan parametre bir blok sürücünün ilk ünitesi için sürücü harfini kapsar. Örneğin 0=A, 1=B v.b.

Eğer bit INIT yordamı, cihazı kuramayacağına karar verirse ve herhangi bir bellek kullanmadan işlemi kesmek isterse, aşağıdaki prosedürü izlemelidir.

- * Ünite sayısına 0 koyun.
- * Bitiş adres offset'ine 0 koyun.
- * Bitiş adres segment'ine kod segment'i koyun.

MEDIA CHECK Yordamı

Komut kodu = 1

Amaç : Cihazın değişip değişmediğinin kontrolü.

ES:BX içeriği Tablo 2.5.6'de gösterilmiştir.

- * Eğer cihaz sabit disk (kapalı devre) ise, donatı byte'ını "cihaz değişmedi" olarak belirle.
- * Eğer son başarılı erişimden beri 2 saniye geçmemişse, donatı byte'ını "cihaz değişmedi" olarak belirle.

Alan	Uzunluk
İstek başlığı	13-BYTE
DOS'tan gelen cihaz tanımlayıcısı	BYTE
Dönüş	BYTE
3.00 ve daha yüksek versiyonlar için bir önceki etiket belirteğine gösterge verir (bit 11=1 ve disk değişti dönülmüşse) (volume ID)	DWORD

Tablo 2.5.6 MEDIA CHECK yordamı istek başlığı.

Komut kod alanı 1 ise, DOS bir sürücü ünitesi için MEDIA CHECK fonksiyonunu çağırır. ve halihazırdaki Cihaz Tanımlayıcı byte'ını bu yordama geçirir. "Cihaz Tanımlayıcı Byte'ı" hakkında detaylı bilgi daha sonraki sayfalarda yer almaktadır. MEDIA CHECK aşağıdakilerden birisini döner:

- * Cihaz değişmedi
- * Cihaz değişti
- * Emin değil
- * Hata kodu

Sürücü aşağıdakileri yapmalıdır:

- * İstek başlığındaki durum sözcüğünü belirle.
- * Dönüş byte'ını belirle.
 - 1 Cihaz değişti
 - 0 Cihazın değişip değişmediği bilinmiyor
 - 1 Cihaz değişmedi

Dönüş byte'ının nasıl belirleneceğine karar vermek için sürücü tarafından aşağıdaki metod kullanılır:

- * Eğer cihaz sabit disk (taşınamaz cihaz) ise, dönüş byte'ını "cihaz değişmedi" olarak belirle.
- * Eğer son başarılı erişimden beri 2 saniye geçmemişse, dönüş byte'ını "cihaz değişmedi" olarak belirle.

* Eğer yeni BPB'deki cihaz byte'ı eski cihaz byte'ı ile uyuşmuyorsa, dönüş byte'ını "cihaz değişti" olarak belirle.

* Eğer halihazırdaki etiket belirteçi bir önceki belirteç ile uyuşuyorsa, "dönüş byte'ını "cihazın değişip değişmediği bilinmiyor" olarak belirle.

DOS 3.00 veya daha yüksek versiyonlarda, sürücü nitelik sözcüğündeki taşınabilir cihaz biti (11) 1 ise ve sürücü -1 dönerse (cihaz değişti), sürücü DWORD göstergeyi bir önceki etiket belirteçi olarak belirlemelidir. Eğer DOS cihaz değişiminin hatalı olduğuna karar verirse, DOS cihaz adına 0Fh (Invalid disk change) hatasını oluşturur. Eğer, sürücü etiket belirteç desteğini gerçeklemezse ve 11.bit 1 ise, sürücü "NO NAME", 0 katarına bir gösterge dönecektir.

Cihaz Tanım Byte'ı (Media Descriptor Byte)

Cihaz tanım byte'ı birkaç cihaz tipi için tanımlanmıştır. Eğer cihaz non-IBM format biti 0 ise, bu byte cihaz byte'ına karşılık gelir. Daha önceden tanımlı değerler aşağıdadır:

Cihaz tanım byte'ı -> 1111xxxx (2 tabanı)

Bit	Anlamı
0	1 = 2 taraflı 0 = 2 taraflı değil
1	1 = 8 sektör 0 = 8 sektör değil
2	1 = taşınabilir 0 = taşınamaz
3-7	1 olarak belirlenmelidir.

DOS cihaz tanım byte'ı örnekleri Tablo 2.5.7'de gösterilmiştir.

DOS, BUILD-BPB yardımını aşağıdaki iki durumda çağırır:

- * Eğer "cihaz değişti" dönerse,
- * Eğer "sani değil" dönerse, kullanılan tampon yoktur. Kullanılan tamponlar diske henüz yazılmayan değiştirilmez.

Disk tipi	Taraf sayısı	Sektör başına iz(track) sayısı	Cihaz Tanım Byte'ı
Sabit Disk	--	--	F8h
5 1/4 inç	2	15	F9h
5 1/4 inç	1	9	FCh
5 1/4 inç	2	9	FDh
5 1/4 inç	1	8	FEh
5 1/4 inç	2	8	FFh
8 inç	1	26	FEh
8 inç	2	26	FDh
8 inç	2	8	FEh

Tablo 2.5.7 DOS Cihaz Tanım Byte'ları.

BUILD BPB Yordamı

Komut kodu = 2

Amaç : BIOS Parametre bloğunu oluşturma.

ES:BX içeriği Tablo 2.5.8'de gösterilmiştir.

Alan	Uzunluk
Istek başlığı	13-BYTE
DOS'tan gelen cihaz tanımlayıcısı	BYTE
Transfer adresi(tampon adresi)	DWORD
BPB tablosuna gösterge	DWORD

Tablo 2.5.8 BUILD BPB yordamı istek başlığı.

DOS, BUILD BPB yordamını aşağıdaki iki durumda çağırır:

- * Eğer "cihaz değişti" dönmüşse.
 - * Eğer "emin değil" dönmüşse, kullanılan tampon yoktur.
- Kullanılan tamponlar diske henüz yazılmayan değiştirilmiş

verileri kapsayan tamponlardır.

Sürücü aşağıdakileri yerine getirmelidir:

- * BPB'ye bir gösterge belirle.
- * Başlık isteğindeki durum sözcüğünü belirle.

Cihaz sürücü, BPB tablosuna gösterge dönmek için ünitede halihazırda bulunan cihaz tipine karar vermelidir. IBMBIO'nun daha önceki versiyonlarında, FAT ID byte'ı, cihazın yapısı ve düzenine karar verir. Zira, FAT ID byte'ı sadece sekiz mümkün değere (F8h'den FFh'e kadar) sahiptir. Açıktır ki, yeni cihaz tipleri icat edildiğinde mümkün değerler tüketilecektir. Değişen cihaz tertipleriyle, DOS, onları okuma isteğinden önce FAT ve directory'lerin yerlerinden haberdar olmaya ihtiyaç duyar.

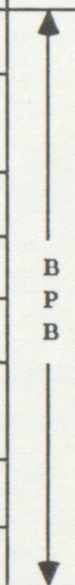
Aşağıdaki paragraflar, DOS'un cihaz tipine karar vermekte kullanacağı yeni metodu açıklamaktadır.

Cihaz sürücü için nitelik sözcüğü, cihaz DOS 1.00 veya 1.10 sistemiyle formatlanmadıkça FAT ID'lerin cihaz tipine karar vermeyi işaret etmesini işaret etmeyecektir. Bu durumda, nitelik sözcüğü FAT ID'nin kullanıldığını gösterir.

DOS 2.00'dan sonraki sistemler için tampon (DOS tarafından geçilen gösterge) boot sektöründeki bilgiye karar vermek için kullanılır.

Cihaz sürücü, boot sektörü, gösterilen tampona okur. Eğer boot sektörü, DOS 2.00'dan yüksek ise, boot sektöründen BPB dönülür. Eğer boot sektörü, DOS versiyonu 2.00'dan küçükse, cihaz sürücü FAT'ın ilk sektörünü belirtilen tampona okur. FAT ID gözden geçirilir ve karşılık gelen BPB dönülür.

Bir kısmi cihaz için BPB'ye bağlı bilgi, cihaz için boot sektörde saklıdır. Kısmi olarak, boot sektör formatı Tablo 2.5.9'da gösterilmiştir.

NOP(90h) ile takip edilen, DOS 2.10 için, 3 byte NEAR JUMP (E9h) veya DOS 3.00 veya yukarısı için 2 byte SHORT JUMP (EBh)	
8 BYTE ismi ve versiyonu	
WORD sektör başına byte (bytes/sector)	
BYTE yerleştirme birimi başına sektör (2'nin kuvveti olmalıdır, sectors/cluster)	
WORD gizli sektör sayısı (lojik sektör 0'dan itibaren)	
BYTE FAT sayısı	
WORD root dir. dosya sayısı (kabul edilebilecek maksimum)	
WORD lojik imajda sektör sayısı (boot sektörü, director'ler v.b. dahil cihazdaki toplam sektör sayısı)	
BYTE cihaz tanımlayıcı	
WORD tek bir FAT tarafından işgal edilen sektör sayısı	
WORD iz başına sektör (sectors/track)	
WORD kafa sayısı (head)	
WORD gizli sektör sayısı	

Tablo 2.5.9 Boot Sektörü.

Sondaki üç sözcük (WORD) cihaz sürücünün cihazı anlamasına yardımcı olmak için tasarlanmıştır. Kafa sayısı, depolama kapasitesi aynı, fakat farklı yüzey sayısına sahip, farklı çok kafalı sürücüleri desteklemek için kullanılışlıdır. Gizli sektör sayısı ünite bölme şemalarını desteklemek için kullanılır.

Etiket tanımı ve disk değiştirmeyi destekleyen sürücüler

için, bu çağırma diskten okunacak yeni bir etiket tanımına neden olacaktır. Bu çağırma, diskin kurallara uygun olarak değiştirildiğini gösterir.

INPUT ve OUTPUT Yordamları

Komut kodu = 3,4,8,9,12

Amaç : Giriş ve çıkış işlemi.

ES:BX içeriği Tablo 2.5.10'de gösterilmiştir.

Alan	Uzunluk
Istek başlığı	13-BYTE
Cihaz tanım byte'ı	BYTE
Transfer adresi(tampon adresi)	DWORD
Byte/sektör sayısı	WORD
Başlangıç sektör numarası (karakter cihazlarda anlamsızdır)	WORD
DOS 3.00 ve yukarı versiyonlar için hata kodu 0Fh dönülmüşse etiket belirteğine gösterge	DWORD

Tablo 2.5.10 INPUT ve OUTPUT yordamları istek başlığı.

Sürücü aşağıdakileri yapmalıdır:

- * İstek başlığındaki durum sözcüğünü belirle.
- * Talep edilen fonksiyonu icra et.
- * Transfer edilen gerçek sektör (veya byte) sayısını belirle.

Not: Bir IOCTL G/Ç çağırmasında hiçbir hata kontrolü yapılmaz. Bu nedenle, sürücü, dönüş sektör (byte) sayısını transfer edilen gerçek byte sayısı olarak belirlemelidir.

Aşağıdakiler, blok cihaz sürücülere uygulanır:

Belirli koşullarda, cihaz sürücünden, cihaz sürücü istek paketindeki transfer adresini kullanarak 64KB'lık bir yazma işlemini yapması istenebilir. Bu, DOS'taki yazma koduna uygun bir optimizasyon eklenmesini ortaya çıkarır. Bu, yalnızca halihazırdaki dosya sonundan itibaren eklenen dosyalarda 64KB'lık bir sektör büyüklüğü içerisinde olan yazmalarda vukubulacaktır. Eğer istenirse, cihaz sürücünün 64KB sınırına önem vermemesine mücadele edilebilir. Örneğin, XXXX:1 transfer adresi ile 10000h byte (64KB) değerindeki sektörü yazma son iki byte'ı göz önüne almaz.

Hatırlatma: DOS fonksiyonları kullanan bir program, FFFFh byte'dan daha fazla giriş veya çıkış işlemi asla talep edemez; bu nedenle, adres çakışması meydana gelemez.

Eğer sürücü 0Fh (Invalid disk change) hata kodu dönerse, kullanıcıya diski yeniden koymasını sormak için doğru etiket belirteçinin (volume ID) olduğu bir ASCIIZ katarına bir DWORD gösterge koymalıdır.

DOS 3.00 ve yukarı versiyonlarda, diskteki açık dosyalara referans sayısı, (OPEN veya CLOSE çağrımları tarafından bakılan) sürücünün ne zaman 0Fh hatasını döneceğine karar vermesine mücadele eder. Eğer açık dosya yoksa (referans sayısı = 0) ve disk değişmiş ise, G/Ç tamamen doğrudur ve 0Fh hatası dönülmez. Eğer açık dosyalar varsa ve disk değişmişse, bir 0Fh hata durumu oluşabilir.

NONDESTRUCTIVE INPUT NO WAIT Yordamı

Komut kodu = 5

Amaç : Beklemesiz, zararlı olmayan giriş.

ES:BX içeriği Tablo 2.5.11'de gösterilmiştir.

Alan	Uzunluk
Istek başlığı	13-BYTE
Cihazdan okunan byte	BYTE

Tablo 2.5.11 NONDESTRUCTIVE INPUT NO WAIT yordamı istek başlığı.

Sürücü aşağıdakileri icra etmelidir:

- * Cihazdan bir byte dön.
- * Başlık isteğindeki durum sözcüğünü belirle.

Eğer karakter cihaz, meşgul bitini 0 dönerse (tampondaki karakterler), o zaman bir sonraki okunacak karakter dönülecektir. Bu karakter, giriş tamponundan kaldırılmaz (zararlı olmayan giriş deyiminden dolayı). Bu çağırma DOS'un bir giriş karakterlerine göz atmasına mücade eder.

STATUS Yordamı

Komut kodu = 6,10

Amaç : Durumu dön.

ES:BX içeriği Tablo 2.5.12'de gösterilmiştir.

Alan	Uzunluk
Istek başlığı	13-BYTE

Tablo 2.5.12 STATUS yordamı istek başlığı.

Sürücü aşağıdakileri yapmalıdır:

- * Talep edilen fonksiyonu icra et.
- * Meşgul bitini belirle.
- * İstek başlığındaki durum kelimesini belirle.

Meşgul biti aşağıdaki gibi belirlenir:

Karakter cihazlarda çıkış için; eğer dönüşte meşgul biti 1

ise, bir yazma isteği halihazırdaki isteğin tamamlanmasını bekleyecektir. Eğer meşgul biti 0 ise, halihazırda istek yoktur. Böylece bir yazma isteği hemen başlayacaktır.

Bir tamponlu karakter cihazda giriş için; eğer dönüşte meşgul biti 1 ise, bir okuma isteği fiziksel cihaza gider. Eğer meşgul biti 0 ise, cihaz tamponunda karakterler vardır ve bir okuma çabukça döner. Bu kullanıcının birşeyler girdiğini gösterir. DOS, bütün karakter cihazların bir ileri-girilen giriş tamponuna sahip olduğunu farzeder. Bu tampona sahip olmayan cihazlar, daima meşgul=0 döner, böylece DOS olmayan bir tampona bilgi koymak için beklemeyi düşünmez.

FLUSH Yordamı

Komut kodu = 6,10

Amaç : İptal yordamı.

ES:BX içeriği Tablo 2.5.13'de gösterilmiştir.

Alan	Uzunluk
Istek başlığı	13-BYTE

Tablo 2.5.13 FLUSH yordamı istek başlığı.

Bu çağırma, sürücüye hakkında bilgi sahibi olduğu bütün kararlaştırılmış istekleri keser. Birinci derecedeki kullanım alanı, karakter cihazlardaki giriş kuyruğunu temizlemektir. Sürücü, istek başlığındaki durum sözcüğünü belirlemelidir.

OPEN veya CLOSE Yordamları (DOS 3.00 ve yukarısı)

Komut kodu = 13,14

Amaç : Açma veya kapama yordamları.

ES:BX içeriği Tablo 2.5.14'de gösterilmiştir.

Alan	Uzunluk
Kalıcı istek başlığı	13-BYTE

Tablo 2.5.14 OPEN veya CLOSE yordamları istek başlığı.

Bu çağrılar, eğer nitelik sözcüğünün 11. biti 1 yapılırsa, cihazdaki halihazırda bulunan dosya hareketi hakkında cihaz bilgisi vermek için tasarlanmıştır. Blok cihazlarda, bu çağrılar, kısmi tamponlamayı düzenlemek için kullanılır. Cihaz, bir referans sayacı saklayabilir. Her OPEN (açma), cihazın referans sayacını arttırmasına neden olur. Her CLOSE (kapama), cihazın referans sayacını eksiltmesine neden olur. Referans sayacının 0 olması, cihazda açık dosya yok demektir. Bundan dolayı cihaz, yazmakta olduğu cihazın içindeki tamponları temizleyebilir, çünkü kullanıcı taşınabilir bir araç sürücünde aracı(disket) taşıyabilir.

Eğer araç değişmişse, tamponları temizlemeden referans sayacını 0 yapmak tavsiye edilir. Bu "son kapama temizlemeye neden olur" olarak düşünülebilir. Bu çağrılar, karakter cihazlarda daha kullanışlıdır. OPEN çağırması, bir cihaza, bir ilk koşullandırma katarı gönderilmek için kullanılabilir. Bir yazıcıda font, sayfa büyüklüğü gibi değerleri belirleyecek bir katarın gönderilmesi için kullanılabilir, böylece her G/Ç akışının başlangıcında durum yazıcı tarafından bilinecektir. Benzer olarak, CLOSE çağırması, G/Ç akışının sonunda bir posta katarını (sayfa başı gibi) göndermek için kullanılabilir. IOCTL kullanılarak, bu katarlar seri G/Ç cihaz akış kontrolunda esnek bir mekanizma sağlarlar.

Not: Bütün işlemler, STDIN, STDOUT, STDERR, STDAUX ve STDPIN (0, 1, 2, 3, 4 kodları) gibi cihazlara erişebilir. CON, AUX ve PRN cihazları daima açıktır.

REMOVABLE MEDIA Yordamı (DOS 3.00 ve yukarısı)

Komut kodu = 15

Amaç : Taşınabilir cihaz yordamı.

ES:BX içeriği Tablo 2.5.15'de gösterilmiştir.

Alan	Uzunluk
Kalıcı istek başlığı	13-BYTE

Tablo 2.5.15 REMOVABLE MEDIA yordamı istek başlığı.

Bu çağırmaı kullanmak için, nitelik alanının 11. bitini 1 olarak belirleyin. Blok cihazlar, yalnızca IOCTL fonksiyon çağırmasının (Int 21h, fonk. 44h) bir alt fonksiyonundan geçerek bu çağırmaı kullabilirler. Bu çağırma, bir uygulama programının, bir taşınabilir araç sürücüyle mi yoksa taşınamaz araç sürücüyle mi iş yaptığını bilmesine müsaade etmesi bakımından yararlıdır. Örneğin, FORMAT programının, aracın bir taşınabilir araç olup olmadığını bilmesi gerekir, çünkü buna göre değişik mesajlar yazacaktır.

Bilgi, durum sözcüğünün meşgul (BUSY) bitinde dönmektedir. Meşgul biti 1 ise, araç taşınamaz, 0 ise araç taşınabilir. (burada hata bit kontrolü yoktur).

Generic IOCTL Request Yordamı (DOS 3.20 ve yukarısı)

Komut kodu = 19

Amaç : Genel IOCTL isteği yordamı.

ES:BX içeriği Tablo 2.5.16'de gösterilmiştir.

Alan	Uzunluk
Kalıcı istek başlığı	13-BYTE
Major fonksiyon	BYTE
Minor fonksiyon	BYTE
SI'nın içeriği	WORD
DI'nın içeriği	WORD
Genel IOCTL istek paketine gösterge	DWORD

Tablo 2.5.16 Generic IOCTL Request yordamı istek başlığı.

Sürücü aşağıdakileri yapmalıdır:

- * Genel IOCTL isteği altında tanımlanan fonksiyonları desteklemelidir.
- * Kendi iz tablosunu oluşturmmalıdır (iz planı).

DOS 3.20 ve daha yukarı versiyonlar için, AL=0Dh (Generic IOCTL Request) çağırması, cihaz parametrelerini getirmek veya belirlemek ve bir lojik cihazda bir iz (track) formatlamak ve kontrol etmek için kullanılır. Generic IOCTL tarafından desteklenen fonksiyonlar DOS kitaplarında açıklanmıştır. Bütün fonksiyonlar için major kod 08h'dir.

Get/Set Logical Device Yordamları (DOS 3.20 ve yukarısı)

Komut kodu = 23,24

Amaç : Lojik cihaz parametre belirleme/öğrenem yordamı.

ES:BX içeriği Tablo 2.5.17'de gösterilmiştir.

Alan	Uzunluk
Kalıcı istek başlığı	13-BYTE
Giriş (ünite kodu) Çıkış (son referans edilen cihaz)	BYTE
Komut kodu	BYTE
Durum	WORD
Saklıdır	DWORD

Tablo 2.5.17 Get/Set Logical Device yordamları istek başlığı.

Bu yordamlar, cihazlara ait bilgileri öğrenmek ve bazı bilgileri değiştirmek için kullanılır. Ayrıntılı bilgi için DOS teknik referans kitaplarına bakınız.

Diger bir TSR yordamı, istek başlığı ile çalışır. Bu yordam kullanılarak, kalıcı bellek bölgesinde belirli bir paragraf (byte/16) belirli bir cihazla bağlantı kurulur. Aynı zamanda, 64K limit de belirtilir.

Bu ter içerisinde, TSR programı çalışır. İst 64K kullanılmıtır. Çıkış, verilen TSR programı ile 64K sınırı geçilmez, hem de COM uzantılı dosyalar çalıştırılır.

2.6.1 TSR Programların Kurulumu

Hafızada kalıcı programların yerleştirilmesi, ist 64K limit mevcuttur. Bu kısıtlara rağmen, belirli bir sınıra kadar veya daha fazla program çalıştırılabilir. Bu kısıtlar, verilen

2.6 MS-DOS İşletim Sisteminde Çoklu İş Yürütme

MS-DOS İşletim Sisteminde çoklu iş yürütme, donanımın ve yazılımın birlikte sağladığı birtakım imkanlarla yapılabilir. Bunun için, öncelikle hafızada kalıcı olan ve kesme ile devreye giren "Hafızada Kalıcı" programların yazılması gereklidir. Bu programlara, genel olarak TSR (Terminate and Stay Resident - Dur ve Kalıcı Ol) diye isimlendirilir. Programın hafızada kapladığı alanın, program sona erdiğinde serbest bırakılmaması gerektiğinden, programın hafızada kalıcı olması birkaç DOS kesme yordamı kullanılarak yapılabilir. Programları TSR durumuna getirmek için en kolay DOS kesme yordamı, Int 27h'dir. Bu yordam, 64K ve daha küçük programların hafızada kalıcı olmasını sağlar. Bu yordam çağırılırken, DX saklayıcısının içeriği "programın kalıcı kodunun bitiş adresi + 1" olarak belirlenmesi gereklidir. Ayrıca, programın özellikle .COM uzantısına sahip olması zorunludur. Tabii olarak .COM programların 64K limiti de burada geçerlidir.

Diğer bir TSR yöntemi, Int 21h fonksiyon 31h'dir. Bu yordam kullanılarak, kalıcı kodun ihtiyaç duyduğu bellek alanı paragraf (byte/16) birimi olarak verilir. Böylece hem .EXE hem de .COM uzantılı programlar hafızada kalıcı olabilir. Aynı zamanda, 64K limiti de söz konusu değildir.

Bu tez içerisinde, TSR program yöntemi olarak, Int 27h kullanılmıştır. Çünkü, yazılan TSR programda hem 64K sınırı geçilmemiş, hem de .COM uzantılı dosyalar oluşturulma imkanı sağlanmıştır.

2.6.1 TSR Programların Kuralları

Hafızada kalıcı programların yazılabilmesi için birkaç kural mevcuttur. Bu kurallara uyulmadığı takdirde sistemin çökmesi veya hasar görmesi söz konusu olabilir. Bu kurallar aşağıda

sıklar halinde açıklanmıştır:

a) Kalıcı programda istediğinizi elde etmenin başka yolu kalmadıkça DOS fonksiyonlarını çağırmayın. (Dosya sistem erişimi, DOS'u çağırmanın başında gelir).

b) Eğer bir dosya giriş/çıkışı yapmak zorundaysanız, aşağıdakilerden birini yapabilirsiniz:

1) DOS konsol G/Ç fonksiyonlarından (Int 21h, fonksiyon 01h-0Ch) başka yöntemler kullanın. (Konsolu okumanın daha iyi yolları mevcuttur).

2) InDOS bayrağını izleyin. Eğer bu bayrak sıfırdan farklı ise, DOS bit Int 21h fonksiyonunu yürütüyor demektir. Bu bayrak sıfırdan farklı işleminizi yürütmeyin.

3) Int 28h'i izleyin. Bu kesme de DOS'un Int 21h fonksiyonlarını yürüttüğünü, konsol G/Ç için meşgul çevriminde olduğunu gösterir. Eğer kendi konsol G/Ç'ınızı DOS altında çalıştırıyorsanız başka birşeyi emin olarak yürütebilirsiniz.

c) TSR programınızın hafızaya yüklenip yüklenmediğini anlayan bir kontrol fonksiyonu sağlayın. Kullanılmış bir kesme vektörüne bağlantı, TSR'a bellekteki daha önceki kopyasını sezmek için kontrol sağlar.

d) TSR'nin olup olmadığını anlamak için yürütülebilir kodun içine bir işaret koyun.

e) Daima diğer TSR'lerin de bulunduğunu farzedin. TSR'niz tarafından kullanılan herhangi bir kesmeyi, TSR'nizin bulunduğu kesme vektörüne kontrolü geçirerek zinciri sağlayın. Böylece daha önceden kurulmuş TSR'leri safdışı etmemiş olursunuz.

f) Kesilen program tarafından kontrol edilen yığıttan ziyade kendi yığıtınızı kullanın. Zira o yığıtta ne olması gerektiğini, yığıtın büyüklüğünü veya bir şeyleri bozmadan önce ne kadar yer kaldığını bilmenize imkan yoktur.

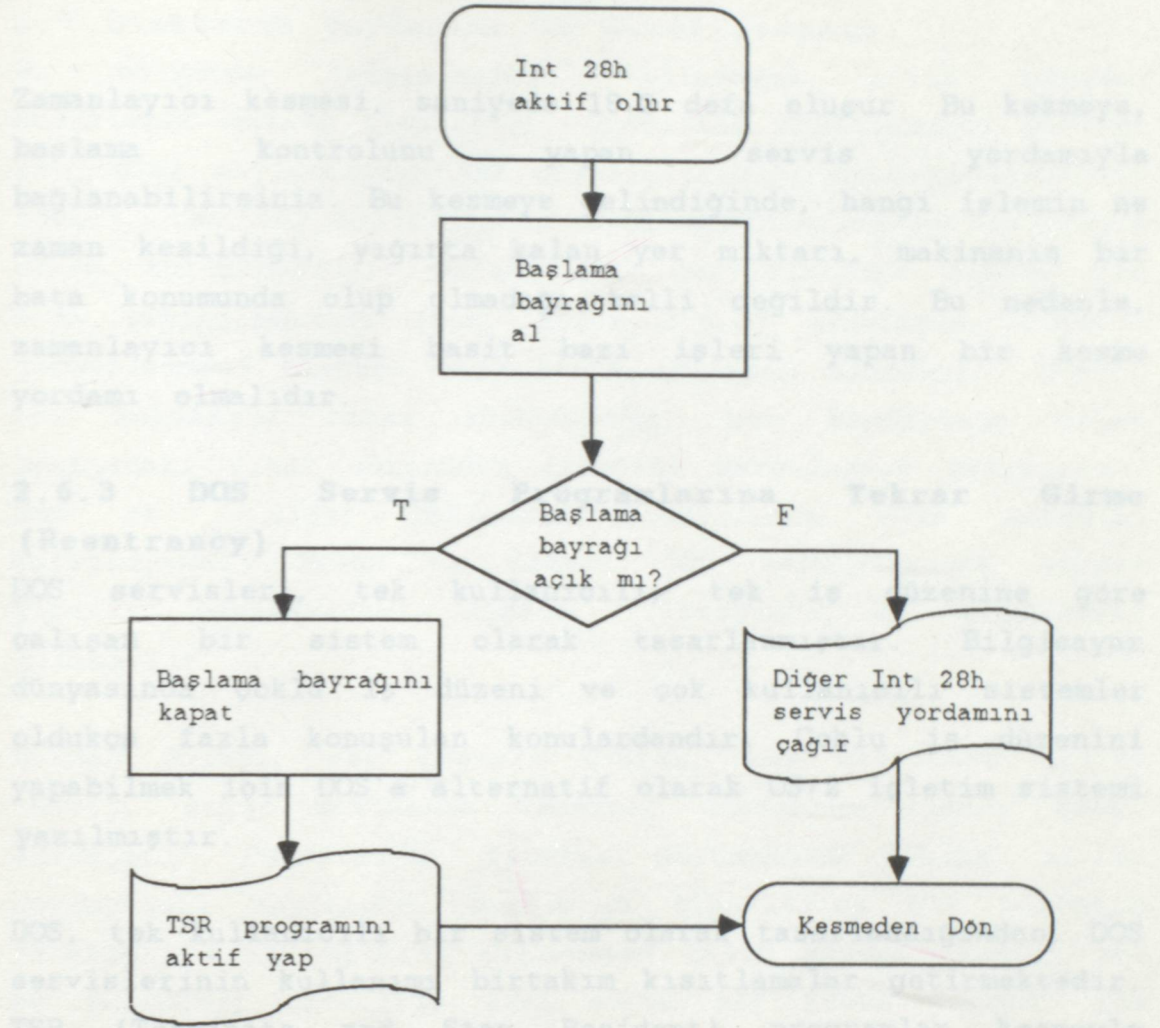
2.6.2 InDOS Bayrağı, DOSOK Kesmesi ve Zamanlayıcı Kesmesi

Int 21h, Fonksiyon 34h InDOS bayrağına (DOS meşgul bayrağı) bir gösterge döner. Eğer bu bayrak sıfırdan farklı ise, DOS, bir fonksiyonu yürütmek için kesilmiş demektir. Eğer TSR'niz DOS'u keserse, fonksiyon genellikle bir saniyenin bir parçasında dönecektir. InDOS bayrağının kontrol zamanı herhangi bir anda olabilir. Başlama bayrağını birledikten sonra, onu sıfırlamak için bir yol bulmak gerekir. Bu amaçla, DOSOK kesmesini ve zamanlayıcı kesmesini kullanmak gerekir. Başlangıçta, Int 28h (DOSOK) için Akış 2.6.1'deki gibi özel bir kesme tutucu yordamı kurmak gerekir.

Akış 2.6.1 Int 28h kesme yordamı

InDOS bayrağının adresi Int 21h, AH=34h olarak çağırıldığında elde edilebilir. Int 21h'den çağırıldığında ES:BX adresi InDOS bayrağının bulunduğu bellek adresini verir. TSR bu adresteki sayının sıfır olması halinde ancak DOS servis programlarını yürütebilir.

Int 28h (DOSOK kesmesi), DOS Int 21h, 01h-0Ch arasındaki fonksiyonlarda konsol girişi için beklerken aktif olur. Eğer bu kesmeyi sezerseniz, diğer DOS fonksiyonları da aynı olarak kullanabilirsiniz demektir. Eğer DOS girişi için beklemiyorsa zamanlayıcı kesmesini kullanabilirsiniz.



Akış 2.6.1 Int 28h kesme yordamı.

IndOS bayrağının adresi Int 21h, AH=34h olarak çağırıldığında elde edilebilir. Int 21h'den çıkıldığında ES:BX adresi IndOS bayrağının bulunduğu bellek adresini verir. TSR bu adresteki sayının sıfır olması halinde ancak DOS servis programlarını yürütebilir.

Int 28h (DOSOK kesmesi), DOS Int 21h, 01h-0Ch arasındaki fonksiyonlarda konsol girişi için beklerken aktif olur. Eğer bu kesmeyi sezerseniz, diğer DOS fonksiyonlarını emin olarak kullanabilirsiniz demektir. Eğer DOS giriş için beklemiyorsa zamanlayıcı kesmesini kullanabilirsiniz.

Zamanlayıcı kesmesi, saniyede 18.2 defa oluşur. Bu kesmeye, başlama kontrolunu yapan servis yordamıyla bağlanabilirsiniz. Bu kesmeye gelindiğinde, hangi işlemin ne zaman kesildiği, yığıtta kalan yer miktarı, makinanın bir hata konumunda olup olmadığı belli değildir. Bu nedenle, zamanlayıcı kesmesi basit bazı işleri yapan bir kesme yordamı olmalıdır.

2.6.3 DOS Servis Programlarına Tekrar Girme (Reentrancy)

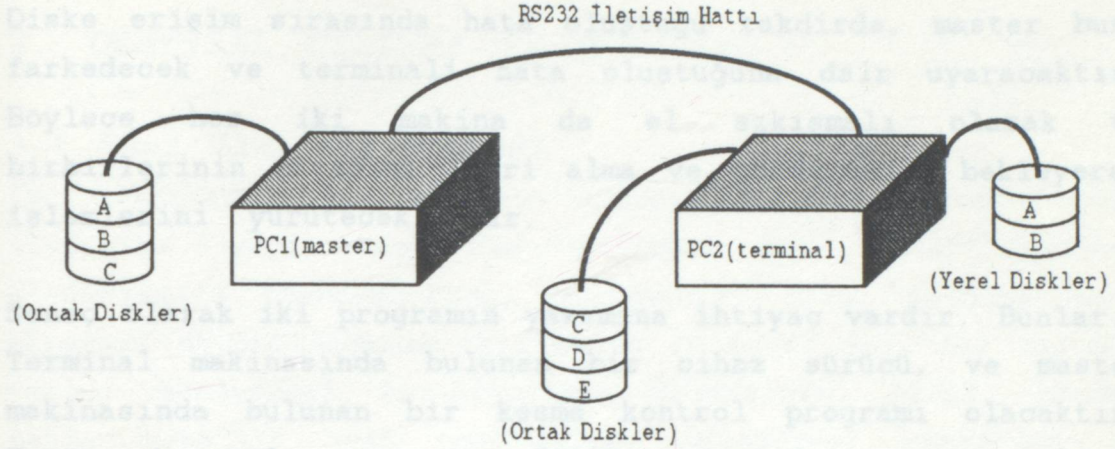
DOS servisleri, tek kullanıcı, tek iş düzenine göre çalışan bir sistem olarak tasarlanmıştır. Bilgisayar dünyasında çoklu iş düzeni ve çok kullanıcı sistemler oldukça fazla konuşulan konulardandır. Çoklu iş düzenini yapabilmek için DOS'a alternatif olarak OS/2 işletim sistemi yazılmıştır.

DOS, tek kullanıcı bir sistem olarak tasarlandığından, DOS servislerinin kullanımı birtakım kısıtlamalar getirmektedir. TSR (Terminate and Stay Resident) programlar kesmeyle devreye girdiklerinde herhangi bir DOS servis programı yürütülüyor olabilir. Aynı şekilde TSR programı da bazı DOS servislerine ihtiyaç duyabilir. Oysa, DOS servisleri global bazı veri alanları kullanır. Şayet kesme yordamı bir DOS servisini tekrar çağırarak olursa, bu takdirde kesme oluşmadan önce yürütmekte olan servise ait veriler bozulacaktır. Bu nedenle bir DOS servis programlarının içine tekrar girilemez. Eğer, bir TSR tarafından DOS servisleri çağırılmak istenirse, bu takdirde kesme geldiğinde DOS servisinin içinde bulunulmadığından kesinlikle emin olmak gerekir. Bunun için daha önce anlatılan TSR yöntem ve uyarıları kullanılabilir.

2.7 Disklerin Paylaşımı Ve Genel Prensip

Bu bölümüde anlatılanları kullanarak artık yöntemi belirlemek mümkün olmaktadır. Öncelikle belirlenmesi gereken şey, disk paylaşımının ana prensipleridir. Disk paylaşımının ideali, her iki makinanın da disk birimlerini birbirleriyle ortak kullanmasıdır. Ancak, DOS servislerinin tekrar ard arda çağırılmaması (bkz. Bölüm 2.6.3) sebebiyle, idealin gerçekleşmesi oldukça güç, hatta imkansız olacaktır. Çünkü, bir makinanın cihaz sürücüsünün, hem kendisinin diğer makinadaki cihaz sürücünden isteğini karşılamaya çalışırken, hem de diğer makinanın kendisinden olan isteğini karşılayamaz. Bunun da nedeni DOS servislerinin tekrar girişli olmamasıdır. Ayrıca, iletişimde de problemler çıkacaktır.

Bundan dolayı, ana prensip olarak, bir makinanın master (host), diğer makinanın terminal durumunda olması kabul edilmiştir. Dolayısıyla, terminal durumundaki makina, master durumundaki makinadan, master'ın cihazlarına ait birtakım istekleri RS232 hattını kullanarak talep edecek, master ise bu talepleri öncelikle veri sezen bir kesme yordamı ile kuyruğa atacak, ve DOS servislerinin dışında bulunduğu sırada (Int 28h), bu istekleri karşılayarak RS232 hattından terminale gönderecektir. Böylece master durumundaki makinaya ait diskler ortak olarak kullanılacaktır. Bunun için tasarlanan şema, Şekil 2.7.1'de gösterilmiştir.



Şekil 2.7.1 Disk Paylaşım Şeması.

Şekilden görüldüğü gibi, öncelikle iki bilgisayar arasında veri iletişimi kurulması gerekmektedir. Tabii bu iletişim kesme yoluyla olmak zorundadır. Yani, terminal, master'dan bir iş talep ettiğinde, terminal istek kodunu gönderen bir veri (kod) gönderecek ve master bulunan ve veri geldiğinde devreye giren bir kesme hizmet programı bu veriyi alarak kuyruğa atacaktır. Master'da bulunacak olan ayrı bir yordam da, kuyruğa atılmış bu kodu değerlendirerek kendi cihaz sürücülerini çağırarak, ve terminalin istediği verileri ona gönderecektir.

Terminal durumundaki, makinada bir cihaz sürücü bulunacaktır. Şekilden de görüldüğü gibi, bu cihaz sürücünün C,D ve E diskleri diğer makinanın A,B ve C disklerine karşılık gelecektir. Dolayısıyla, terminaldeki cihaz sürücü, RS232 hattını kullanarak, diğer makinaya cihaz sürücü isteğini gönderecektir. Bu defa, kontrol master'da olacak ve master gereken işlemi yapmak için İstek Başlığını terminalden alacaktır. Eğer bir sektör okuma/yazma işlemi gerçekleştirilecekse, master blok blok (BUFFER sayısı kadar - 512'nin katı) verileri gönderecek veya alacaktır. Blok uzunluğu 64K sınırına kadar verilebilecektir.

Diske erişim sırasında hata oluştuğu takdirde, master bunu farkedecek ve terminali hata oluştuğuna dair uyaracaktır. Böylece her iki makina da el sıkışmalı olarak ve birbirlerinin sırasını (veri alma ve göndermede) bekleyerek işlemlerini yürüteceklerdir.

Sonuç olarak iki programın yazımına ihtiyaç vardır. Bunlar: Terminal makinasında bulunan bir cihaz sürücü, ve master makinasında bulunan bir kesme kontrol programı olacaktır. Kesme kontrol programı RS232 portundan gelen isteği kuyruklayacak, Int 28h yordamı ise bu isteği yerine getirerek cevaplayacaktır.

Genel olarak, Asenkron İletişim Kontrolöründe el sıkışmalı çalışması için, Modem Kontrol Saklayıcısının (Modem Control Register - MCR) 0 ve 1 bitleri (0-Data Terminal Ready, 1-Request To Send) ve karşı tarafın MCR sinyal kontrol aracı amacıyla Modem Durum Saklayıcısı (Modem Status Register - MSR) kullanılır.

Asenkron veri iletişimi için birkaç yöntem mevcuttur. Bu yöntemler, el sıkışmayı sağlamak amacıyla MCR ve MSR saklayıcılarına kullanılır. Ben de bu taraftan geliştirilen veri transferi için tamamen kendime ait bir yöntem kullandım. Burada öncüm, minimum kablo bağlantısı ve minimum işlemlerle veri transferini gerçekleştirmekti.

Veri transferi için yalnızca 3 kablo bağlantısı yapılmaktadır. Normal seri veri transferi için RxD (Receive Data), TxD (Transmit Data) ve Signal Ground olarak bağlanmalıdır. RxD ve TxD karşılıklı olarak bağlanır. Buna ilaveten, el sıkışmayı sağlamak amacıyla, 1. Bilgisayarın Data Terminal Ready girisi ve 2. Bilgisayarın Data Set Ready girisi birbirine bağlanmıştır. Tekli ki bu bağlantının sonucu, 2. bilgisayarın Data Terminal Ready

3. DENEYSEL ÇALIŞMA VE BULGULAR

3.1 Veri İletişim Yöntemi

İki bilgisayar arasında veya modem-bilgisayar arasında veri iletişimi yapılırken hiç kuşkusuz en ideal yöntem, el-sıkışmalı (hand-shaking) çalışmadır. Oysa, basitçe veri aktarımı düşünüldüğünde akla gelen en basit yol, bir birimin veriyi göndermesi ve diğerinin okumasıdır. Ancak, birden fazla byte'lık veri gönderilmek istenirse, gönderici ve alıcının hızları farklı olduğundan el-sıkışmalı bir yöntemle çalışmaları gerekmektedir.

Genel olarak, Asenkron İletişim Kontrolörünün el sıkışmalı çalışması için, Modem Kontrol Saklayıcısının (Modem Control Register - MCR) 0. ve 1. bitleri (0-Data Terminal Ready, 1-Request To Send) ve karşı tarafın MCR'sini kontrol etmek amacıyla Modem Durum Saklayıcısı (Modem Status Register - MSR) kullanılır.

Asenkron veri iletişimi için birkaç yöntem mevcuttur. Bu yöntemler, el sıkışmayı sağlamak amacıyla MCR ve MSR saklayıcılarını kullanırlar. Ben de bu tezi geliştirirken veri transferi için tamamen kendime ait bir yöntem kullandım. Burada amacım, minimum kablo bağlantısı ve minimum işlemle veri transferini gerçekleştirmektir.

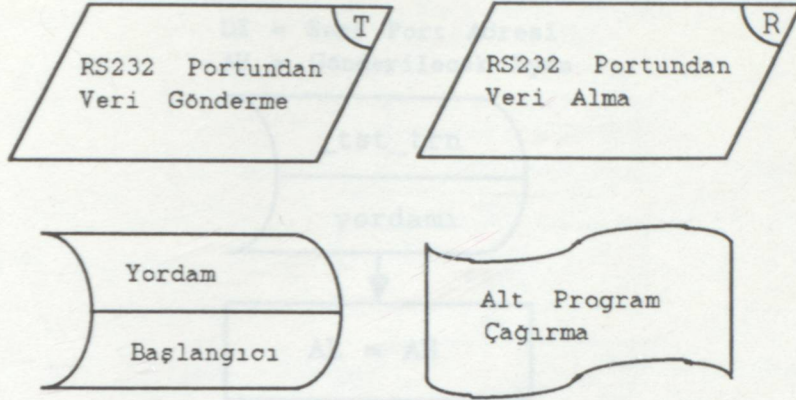
Veri transferi için yalnızca 5 kablo bağlantısı yapılmaktadır. Normal seri veri transferi için RxD (Receive Data), TxD (Transmit Data) ve Signal Ground uçları bağlanmalıdır(RxD ve TxD karşılıklı bağlanır). Buna ilaveten, el sıkışmayı sağlamak amacıyla, 1. bilgisayarın Data Terminal Ready çıkış ucu ile 2. bilgisayarın Data Set Ready giriş ucu birbirine bağlanmıştır. Tabii ki bu bağlantının benzeri, 2. bilgisayarın Data Terminal Ready

çıkışı ile 1. bilgisayarın Data Set Ready girişi arasında yapılmalıdır. Böylece 5 kablo bağlantısı yeterli olmaktadır.

Veri transfer yönteminde, Asenkron İletişim Kontrolörünün MSR saklayıcısının bir özelliği önem arz etmektedir. Bilindiği gibi, MCR'nin 0. biti (DTR) karşı tarafın MSR'sinin 5. bitine (DSR) bağlıdır. MSR'nin 1. biti de DDSR olup, Modem Durum Saklayıcısının mikroişlemci tarafından son okunduğu andan sonra, DSR girişinin konum değiştirdiğini gösterir. Ayrıca hat durum, saklayıcısının (Line Status Register - LSR) 0. biti verinin hazır (Data Ready - DR) olduğunu gösterir.

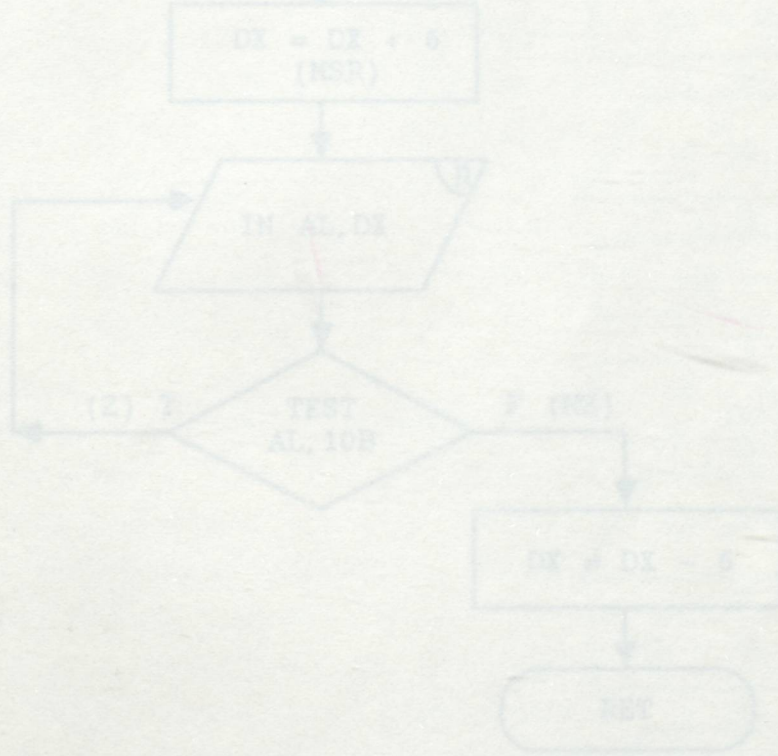
Bu bilgiler ışığında, alıcı veriyi almak için LSR saklayıcısının 0. bitinin (DR) aktif olmasını bekleyecek, veriyi alacak ve verinin alındığını belirtmek üzere, kendisinin DTR çıkışının konumunu değiştirerek vericinin DDSR bitini aktif yapacaktır. Verici ise, veriyi tampona yazıp gönderecek ve diğer veriyi göndermek için DDSR işaretinin aktif olmasını bekleyecektir. Böylece, veri transfer edilene kadar her iki makina birbirlerini bekleme durumunda kalacaktır. Birden fazla veri transferi ise, yukarıda anlatılan verici ve alıcı yordamlarının tekrar çağırılması ile yapılacaktır. Bunun için yazılan programlar, en sonda verilmiş olup, RSTR.INC, RSINIT.INC, RSINIT.MAC isimlerine sahiptir. Programlar, kaynak Assembly programlardan INCLUDE komutu ile çağırılırlar.

Burada akış diyagramı çizilirken bazı özel semboller kullanılmıştır. Bunlar Şekil 3.1.1'de gösterilmiştir.



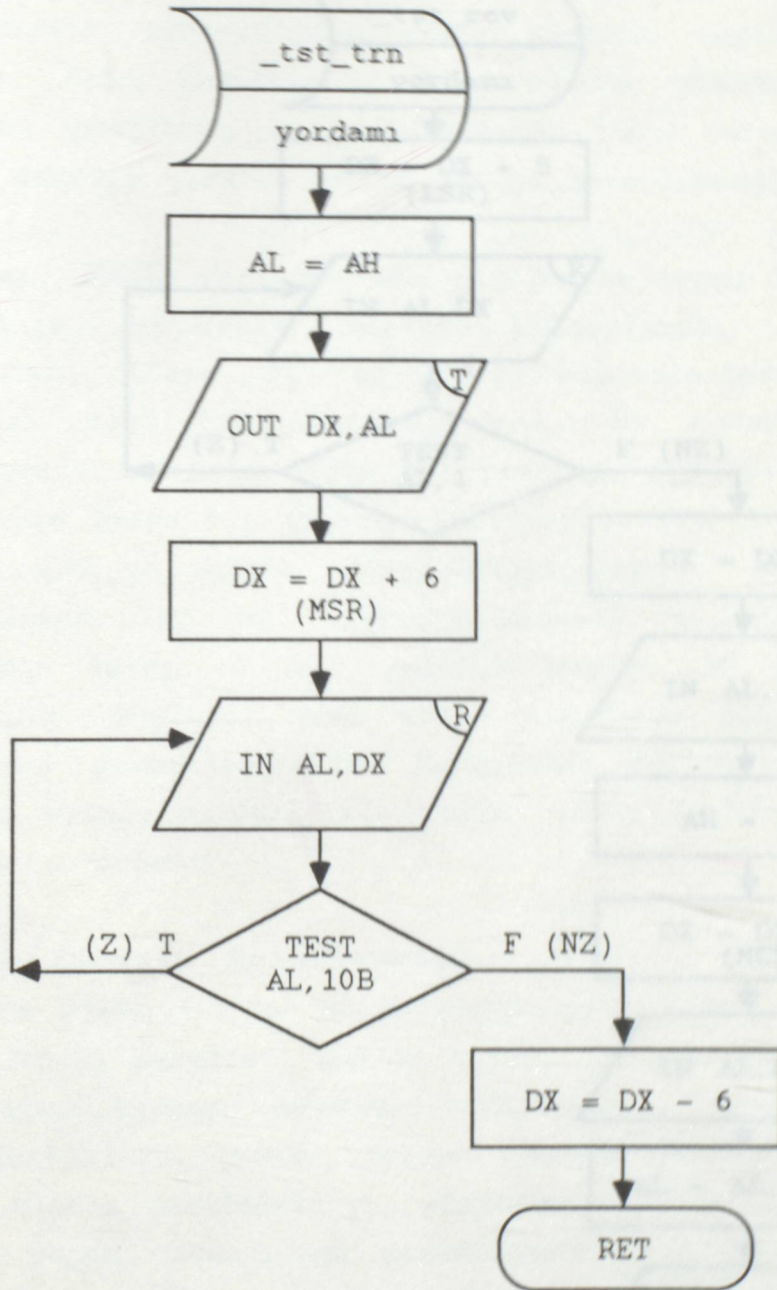
Şekil 3.1.1 Özel Akış Diyagram Sembolleri.

Yukarıda anlatılan yöntem, Akış 3.1.1 ve 3.1.2'de ayrıntılı olarak çizilmiştir.



Akış 3.1.1 RS232 Portundan Veri Gönderme

DX = Seri Port Adresi
AH = Gonderilecek byte

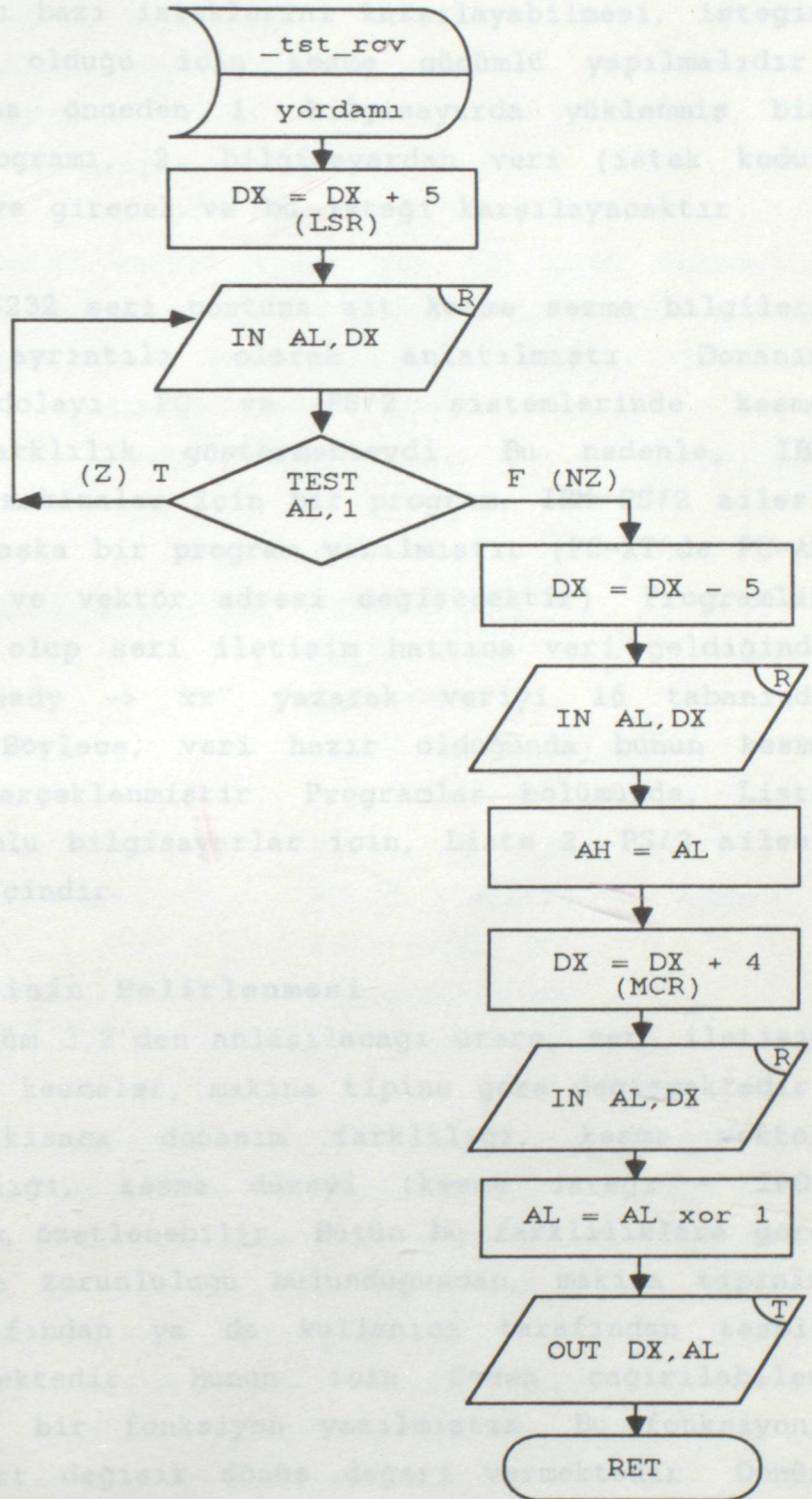


Akış 3.1.1 RS232 Portundan Veri Gönderme.

Akış 3.1.2 RS232 Portundan Veri Alma

3.2 RS232 Kesme Servis Deneyi

DX = Seri Port Adresi
AH = Alınacak byte



Akış 3.1.2 RS232 Portundan Veri Alma.

3.2 RS232 Kesme Sezme Deneyi

Hiç kuşkusuz, iki bilgisayar arasında iletişim kurarken, birinin diğerinin bazı isteklerini karşılayabilmesi, isteğin zamanı belirsiz olduğu için kesme güdümlü yapılmalıdır. Dolayısıyla, daha önceden 1. bilgisayarda yüklenmiş bir kesme hizmet programı, 2. bilgisayardan veri (istek kodu) geldiğinde devreye girecek ve bu isteği karşılayacaktır.

Daha önceden, RS232 seri portuna ait kesme sezme bilgileri Bölüm 2.4'te ayrıntılı olarak anlatılmıştı. Donanım farklılığından dolayı PC ve PS/2 sistemlerinde kesme tanımlamaları farklılık göstermekteydi. Bu nedenle, IBM PC-AT ve uyumlu makinalar için bir program, IBM PS/2 ailesi makinalar için başka bir program yazılmıştır (PC-XT'de PC-AT programında IRQ ve vektör adresi değişecektir). Programlar hafızada kalıcı olup seri iletişim hattına veri geldiğinde ekrana "Data Ready -> xx" yazarak veriyi 16 tabanında göstermektedir. Böylece, veri hazır olduğunda bunun kesme ile sezilmesi gerçekleşmiştir. Programlar bölümünde, Liste 1, PC-AT ve uyumlu bilgisayarlar için, Liste 2, PS/2 ailesi bilgisayarları içindir.

3.3 Makina Tipinin Belirlenmesi

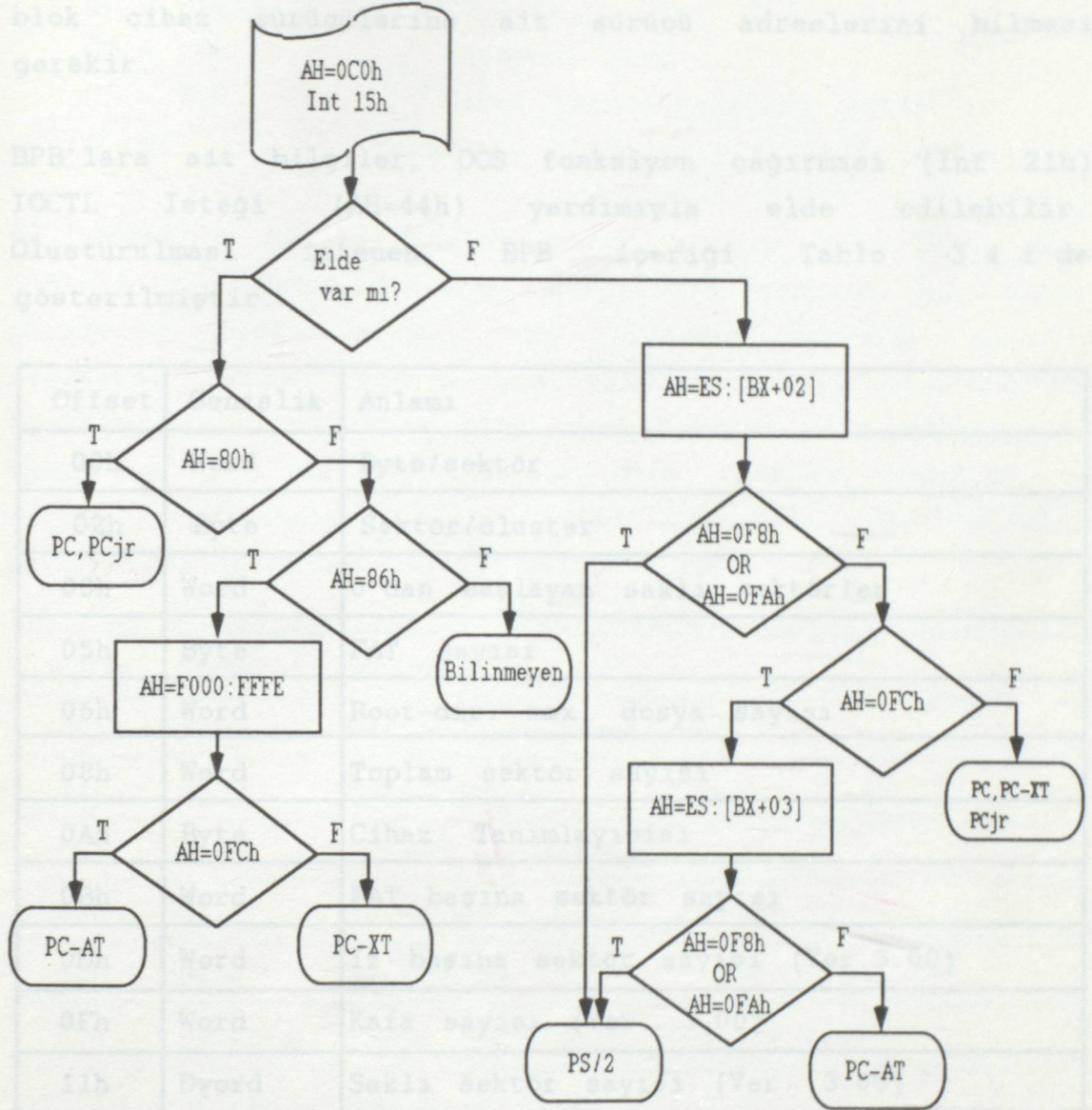
Bölüm 2.3 ve Bölüm 3.2'den anlaşılabileceği üzere, seri iletişim hattı ile ilgili kesmeler, makina tipine göre değişmektedir. Bunun nedeni, kısaca donanım farklılığı, kesme vektör numarası farklılığı, kesme düzeyi (kesme isteği - IRQ) farklılığı olarak özetlenebilir. Bütün bu farklılıklara göre ayrı işlem yapma zorunluluğu bulunduğundan, makina tipinin ya program tarafından ya da kullanıcı tarafından tesbit edilmesi gerekmektedir. Bunun için C'den çağırılabilen Assembly dilinde bir fonksiyon yazılmıştır. Bu fonksiyon, sonuç olarak dört değişik dönüş değeri vermektedir. Dönüş değerine göre makina tipi aşağıda gösterilmiştir:

<u>Dönüş değeri</u>	<u>Makina tipi</u>
0	PC-XT veya uyumlu
1	PC-AT veya uyumlu
2	PS/2 ailesi
-1	Bilinmeyen

Makina tipine karar vermek için, Int 15h BIOS kesmesinin 0C0h numaralı fonksiyonu ve F000h:FFFFh adresinde bulunan sistem model tanım byte'ı kullanılmıştır. _dmtpe (determine machine type) yordamına ait yöntem Akış 3.3.1'de verilmiştir. Sistem Model Tanım Byte'ına ait kodlar aşağıda verilmiştir.

<u>Model Tanım Byte (F000h:FFFFh)</u>	<u>Makina Tipi</u>
9Ah	COMPAQ Plus
FFh	IBM PC
FEh	PC-XT, Portable PC
FDh	PCjr
FC h	PC-AT, PS/2 Model 50/60
FBh	PC-XT (1/10/86'dan sonra)
FAh	PS/2 Model 30
F9h	Convertible PC
F8h	PS/2 Model 70/80

Int 15h kesmesi, fonksiyon 0C0h için giriş parametresi, AH=0C0h'dır. Dönüş değerinde elde varsa, AH=80h ise, makina tipi ya PC ya da PCjr'dır. Elde var ve AH=86h ise, ya PC-XT ya da PC-AT'dir. Bu durumda, sistem model tanım byte'ına bakılır. Diğer durumlarda (elde yok), ES:BX, ROM'da bulunan sistem tanım byte'ına göstergeyi verir. Burada sırasıyla, ES:[BX+2], model byte'ını, ES:[BX+3] ise alt model byte'ını gösterir. Bu kontroller de yapılarak makina tipi net olarak bulunur.



Akış 3.3.1 Makina Tipine Karar Verme Algoritması (_dmtyp).

3.4 Disklere Ait Bilgilerin Elde Edilmesi

Bölüm 2.5 (cihaz sürücüler) incelendiğinde, görüleceği gibi terminal makina için, INIT (ilk koşullandırma) cihaz sürücü fonksiyonu çağırıldığında, cihaz sürücüyeye bağlı olan blok cihazlara ait BPB (BIOS Parametre Bloğu) bilgilerinin dönüş değeri olarak verilmesi gerekmektedir. Ayrıca, terminal durumundaki makinadan gelecek cihaz sürücü isteklerine cevap verebilmek için, master durumundaki makinanın kendisinin

blok cihaz sürücülerine ait sürücü adreslerini bilmesi gerekir.

BPB'lara ait bilgiler, DOS fonksiyon çağırması (Int 21h) IOCTL isteği (AH=44h) yardımıyla elde edilebilir. Oluşturulması istenen, BPB içeriği Tablo 3.4.1'de gösterilmiştir.

Offset	Genişlik	Anlamı
00h	Word	Byte/sektör
02h	Byte	Sektör/cluster
03h	Word	0'dan başlayan saklı sektörler
05h	Byte	FAT sayısı
06h	Word	Root-dir. max. dosya sayısı
08h	Word	Toplam sektör sayısı
0Ah	Byte	Cihaz Tanımlayıcısı
0Bh	Word	FAT başına sektör sayısı
0Dh	Word	İz başına sektör sayısı (Ver.3.00)
0Fh	Word	Kafa sayısı (Ver. 3.00)
11h	Dword	Saklı sektör sayısı (Ver. 3.00)
15h	Dword	08h'deki sektör sayısı sıfırsa top. sek.
19h	7-byte	Saklıdır

Tablo 3.4.1 BIOS Parametre Bloğu.

IOCTL isteği için saklayıcıların içerikleri aşağıdaki gibi olmalıdır:

Saklayıcı	Anlamı
AH = 44h	IOCTL isteği
AL = 0Dh	Genel IOCTL isteği

BL = Ünite no (0=default, 1=A, 2=B v.b.)
 CH = 08h Kategori kodu
 CL = 60h Cihaz parametrelerini getir
 DS:DX Parametre bloğunun yazılacağı adres

Bu değerler ilgili saklayıcılara atanıp DOS (Int 21h) çağırılırsa, Tablo 3.4.2'de gösterilen şekilde bir cihaz parametre bloğu oluşur. Bu listeden gerekli alanlar seçilerek Tablo 3.4.1'de gösterilmiş alanlar oluşturulur. Böylece her sürücü için ayrı ayrı BPB'ler elde edilebilir. Eğer Int 21h'den dönüldüğünde hata oluşmuş demektir ve sürücü numarasının yanlış olduğu anlaşılır. Bu durumda, bilgi alma işleminin sonuna gelinmiştir.

Offset	Genişlik	Anlamı
00h	Byte	Özel fonksiyonlar
01h	Byte	Cihaz tipi
02h	Word	Cihaz nitelikleri
04h	Word	Silindir sayısı
06h	Byte	Alet tipi
07h	32-byte	Tablo 3.4.1'deki BPB alanları

Tablo 3.4.2 Cihaz Parametre Bloğu.

Cihaz sürücünün başlangıç adresini öğrenmek üzere (master için), IBM ve Microsoft firmalarının saklı (reserved) olarak açıkladığı ve ofis dışı (dokümanter olmayan) kaynaklardan bulunan bir DOS fonksiyon çağırması kullanılmıştır. Bu fonksiyon çağırması, AH=1Fh değeri ile Int 21h'in çağırılması ile gerçekleşir. Bu çağırmadan sonra, kullanımda olan disk için, Disk Parametre Bloğuna (DPB) ait adres DS:BX saklayıcılarında dönülür. AL=00h ise hata yok, AL=FFh ise

hata vardır. Genel olarak kullanımda olan diskin DPB değerleri elde edilebilir. DS:BX bellek adresindeki verilerin içeriği Tablo 3.4.3'de gösterilmiştir.

Offset	Genişlik	Anlamı
00h	Byte	Sürücü no. (0=A, 1=B v.b.)
01h	Byte	Cihaz sürücü ünite numarası
02h	Word	Byte/sektör
04h	Byte	Sektör/cluster
05h	Byte	Kaydırma sayısı (cluster için)
06h	Word	Saklı boot sektör sayısı
08h	Byte	FAT kopya sayısı
09h	Word	Root-dir dosya sayısı
0Bh	Word	İlk veri sektör numarası
0Dh	Word	En büyük cluster no + 1
0Fh	Byte	FAT başına sektör sayısı
10h	Word	Root-dir. başlangıç sektör no.
12h	Dword	Cihaz sürücünün başlangıç adresi
16h	Byte	Cihaz tanımlayıcı byte'ı
17h	Byte	DPB uygunluk byte'ı (0FFh ise yanlış)
18h	Dword	Bir sonraki cihaz sürücü adresi
1Ch	Word	Current dir. başlangıç sektör no (Ver.2)
1Eh	64-Byte	Current dir. ASCII izi (Ver.2)

Tablo 3.4.3 Disk Parametre Bloğu.

4. TARTIŞMA VE SONUÇ

4.1 Parametrelere Göre Programın Tasarımı

Daha önceki bölümlerde anlatılan master makina için cihaz sürücü, ve terminal makina için kesme yordamlarının yazılmasında, programın modülerliği açısından birçok değişken parametre ortaya çıkmaktadır. Buna örnek olarak, iletişimin baud oranı verilebilir. Şimdi bu parametreleri belirleyelim:

Master makinadaki kesme hizmet programı için:

Makinanın Tipi (RS232 kesme sezme için),

Seri Port Adresi ve Kart Numarası (iletişim için),

Cihaz Sürücü Adresi (bağlı bulunan disk birimleri için),

Buna bağlı olarak cihaz sürücü strateji ve kesme yordamlarının adresleri,

Kullanılacak Tampon Genişliği (okuma/yazma veri transferi için),

RS232 ilk koşullandırma bilgilerin (baud oranı gibi) belirlenmesi gerekir.

Terminal makinadaki cihaz sürücü programı için:

Seri Port Adresi ve Kart Numarası (iletişim için),

Kullanılacak Tampon Genişliği (okuma/yazma veri transferi için),

Master makinanın blok cihazlarına ait, BIOS Parametre Blokları (INIT yordamına göndermek için),

RS232 ilk koşullandırma bilgilerin (baud oranı gibi) belirlenmesi gerekir.

Bu parametrelerin belirlenmesi tamamlandıktan sonraki en önemli aşama, bu parametrelerin programa nasıl adapte edileceğidir. Bunun için iki yöntem mümkündür. Birincisi, direk olarak programın derlenmiş hali üzerinde değişiklik,

ikincisi programın kaynak dosyası üzerinde değişikliktir. Yalnız ikinci şık için programın tekrar derlenmesi gerekmektedir. İlk bakışta, birinci yöntem daha pratik gibi görünecektir, ancak bunu yapmak oldukça zararlı olabilir. Çünkü, programda yapılacak en ufak bir değişiklikte, düzeltilecek verilerin adresleri tamamen yerdeğiştirecektir. Bu nedenle, ikinci yöntem daha pratik görülmüş ve programın kaynağı üzerinde değişiklik yapılmıştır.

Bu amaçla, kaynak programda değişiklik yapılacak yerleri ve değişiklik kodunu gösteren iki karakter kullanılmıştır. Eğer kaynak .ASM programında ">" ile başlayan bir satır görülürse, bu değişiklik yapılacak satırı gösterecektir. Değişiklik kodu ise ">" karakterlerinden hemen sonra tek dijital sayı olarak verilmiştir. Yani, ">x" formatının görüldüğü yerler değişiklik yapılacak yerleri gösterecektir.

Kaynak programdaki ">x" kodlarının yerine yazılacak değişken veriler ise, C dilinde yazılmış bir yönetici program yardımıyla oluşturulacaktır.

Bu program, PMAIN.C dosyası olup Liste 7'de verilmiştir. PMAIN.C'nin bazı sabit tanımlamalarını, veri yapı tanımlamalarını (DP, BPB gibi) içeren PMAIN.H dosyası da Liste 8 olarak verilmiştir. Ayrıca, PMAIN.C programının kullandığı Assembly bazı yordamları içeren SUBS.ASM dosyası da Liste 9 olarak verilmiştir.

Kaynak programlar içerisinde görülüp, değiştirilmesi gereken kısımlar aşağıda gösterilmiştir:

INTSRVR.ASM için;

Kod Programa yazılacak satır

>1 MACTYPE Equ PCXT | PCAT | PS2 ; Makina tipi


```

PORTADR Equ      xxxxh   ; Seri Port Adresi
CARDNO  Equ      0 | 1 | 2 | 3   ; Seri Kart no.
BUFFER  Equ      xx      ; Tampon sayısı (sektör bazında)
>2      Startegy   Label   Dword
                Dw       strategy_offset,  strategy_segment
                Interrupt Label   Dword
                Dw       interrupt_offset,  interrupt_segment
>3      rs232init   card, baud, parity, stop, wordlen

```

RDDRVR.ASM için;

Kod	Programa yazılacak satır
>1	PORTADR Equ xxxxh ; Seri Port Adresi BUFFER Equ xx ; Tampon sayısı (sektör bazında)
>2	Dw xxxxh ; Cihazın niteliği
>3	BIOS Parametre Blokları ve ünite sayısı
>4	rs232init card, baud, parity, stop, wordlen

PMMAIN.C, INTSRVR.ASM ve RDDRVR.ASM dosyalarını kaynak olarak alıp, gerekli yerlere yukarıdaki parametreleri yazarak, INTSRVR.TMP ve RDDRVR.TMP dosyalarını yaratmaktadır.

PMMAIN.C, master makinada çalıştırılacağından, RDDRVR.TMP'ye yazılacak 1 numaralı parametrelerden PORTADR (seri port adresi), PTERM.C (Liste 6) isimli program yardımıyla oluşturulacaktır. PTERM.C, terminalde çalıştırıldığında, PTERM.OUT isimli ve port adresini içeren bir dosya oluşturacaktır. Bu dosya, PMMAIN.C'nin çalıştırıldığı yere kopyalanarak, programın RDDRVR.TMP dosyası için giriş parametresi temin edilmiş olacaktır.

4.2 Kesme Hizmet Programı Ve Cihaz Sürücünün Yazılması

Parametre tanımından sonra, artık sıra hafızada kalıcı kesme

hizmet programının ve cihaz sürücünün tasarlanmasına gelmiştir. Bu programlar prensip olarak aynı frekansta çalışmalıdır. Yani, birbirlerini gerektiğinde beklemeli ve birbirlerinin isteğine en kısa zamanda cevap verebilmelidirler. INTSRVR.ASM, INTSRVR.TMP dosyası oluşturulup derlendiğinde INTSRVR.COM dosyası oluşturmalıdır. Bu sebeple, programın orijini 100h seçilmiştir. Programın kullanacağı tampon, kalıcı kodun bitiş adresine eklenecektir. Burada bahsedilen tampon, INTSRVR'nin getirdiği bir kısıtlamadır. Çünkü, terminal makinada işlem için ayrılan tampona, master makinanın erişmesi imkansızdır. Bu sebeple, INTSRVR kendi tamponunu kullanacak ve diske okuma/yazma işlemlerini bu tamponun sınırlarında gerçekleyecektir. Terminal makina da bu tamponun sınırlarını bildiğinden, cevapları ona göre kabul edecektir.

RDDRVR.ASM, RDDRVR.TMP dosyası oluşturulup derlendiğinde, cihaz sürücü olması dolayısıyla RDDRVR.SYS dosyasını oluşturacak, ancak bu dosyanın orijini 0h olarak seçilecektir. Bu cihaz sürücünün kesmeye ihtiyacı olmadığından, istekleri diğer makinadan gelecek cevaplara göre yönlendirecektir. Böylece, RDDRVR.SYS'ye gelen cihaz sürücü isteği, sanki direk sisteme bağlı bir cihazmış gibi işlemleri yürütecek ve böylece **disk paylaşımı** sağlanmış olacaktır.

Her iki makinada da, disk birimleri için fonksiyonları gerçekleştirilecek sürücü yordamları aşağıda belirtilmiştir:

Komut Kodu Sürücü Yordamı

0	INIT (yalnız terminalde)
1	MEDIA CHECK
2	BUILD BPB


```

4      READ (INPUT)
8      WRITE (OUTPUT)
9      WRITE WITH VERIFY

```

Öncelikle terminaldeki cihaz sürücüyü bir istek geldiğinde, transfer edilecek ilk bilgi, İstek Başlığıdır. Tabii ki, istek başlığı alanının, ilgili fonksiyon için gerekli kısmı master'a gönderilecektir. Bu bölümde, her fonksiyona ve her makina a ait akış diyagramı çizilmiştir.

4.2.1 INIT Yordamı

Terminal makinasındaki, RDDVR.SYS'nin içerisindeki INIT yordamı Bölüm 2.5'de(cihaz sürücüler) anlatılan istekleri gerçekleyecektir. Ancak bu istekler (BPB göstergeleri gibi), istenen veriler daha önceden kaynak programın içine dahil edildiğinden, master makina a ihtiyaç duymayacaktır. Bunun için INIT yordamının akışını çizmeye gerek yoktur ve yalnızca RDDVR.ASM'nin içinde yer alacaktır. INIT yordamı, ayrıca terminal makinanın sistemi açıldığında, kendisine (versiyon) ve master'daki disklere (ünite kodları) ait ekrana birtakım mesajlar yazacaktır.

4.2.2 MEDIA CHECK Yordamı

Bu yordam, anlatıldığı gibi, cihazın değişip değişmediğini kontrol eder. Bu fonksiyonu gerçeklemek için az sayıda veri gönderilmesi yeterlidir. Öncelikle, terminal programı, istek başlığını master'a aktaracak, daha sonra, dönüş byte'ını bekleyecektir. Eğer yanlış disk değişimi hatası oluşmuşsa, ek olarak bir önceki etiket belirteğini de isteyecektir.

MEDIA CHECK yordamının akış diyagramı, RDDVR.ASM için Akış 4.2.1'de, INTSRVR.ASM için, Akış 4.2.2'de, gösterilmiştir.

4.2.3 BUILD BPB Yordamı

Bu yordamda da, MEDIA CHECK yordamına benzer işlemler yapılmaktadır. Öncelikle, terminal programı, istek başlığını master'a göndermektedir. Daha sonra, durum byte'ı okunur. Hata yoksa, BPB'a ait 32 byte'lık veri daha okunur ve işlem tamamlanmış olur. Master da bu işlemlerin karşılığı kendi tarafında yapacaktır.

BUILD BPB yordamının akış diyagramı, RDDRVR.ASM için Akış 4.2.3'de, INTSRVR.ASM için, Akış 4.2.4'de, gösterilmiştir.

4.2.4 READ (INPUT) Yordamı

Bu ve bundan sonraki yordamlar, temel olarak blok işlem yaparlar. Ancak, blok(tampon) için ayrılan alanlar, isteğin geldiği terminal makinasındadır. Bu nedenle, Master makinası kendisinin sahip olduğu tamponları kullanmak zorundadır. Terminal makinanın, master makinaya işlem sırasını vermesinden sonra, master'ın bu işlemi yapabilmesi için tampona ihtiyacı vardır. Bu tampon, ancak kesme yordamı kurulduğunda ayrılabilir. Tampon için ayrılan alan değişkendir ve birimi sektördür. Örneğin 10 adet tampon ayrılmışsa, bellekte harcanan bölge, $10 \times 512 = 5120$ byte (5K) uzunluğundadır. Ancak, terminal, master'dan 18 sektörlük işlem isterse, bu defa master işlemi bir 10'luk ve bir de 8'lik blok halinde gerçekleyecektir. Böylece hafızadan tasarruf edilmiş olacaktır. Kullanıcı, tampon sayısını da istediği gibi ayarlayabilmektedir. Veri akışı, master'dan terminale doğrudur.

Eğer işlem tamamlanmadan herhangi bir hata oluşursa, bu takdirde, master makinası, terminal makinasını uyararak işlemini yarıda kesecektir. İstenirse, yapılan transferlerin raporunu ekranda göstermek mümkündür. Ancak bu

gerçeklenmemiştir. Nedeni de ekranda bulunan verileri bozmamaktır.

READ yordamının akış diyagramı, RDDRVR.ASM için Akış 4.2.5'de, INTSRVR.ASM için, Akış 4.2.6'de, gösterilmiştir.

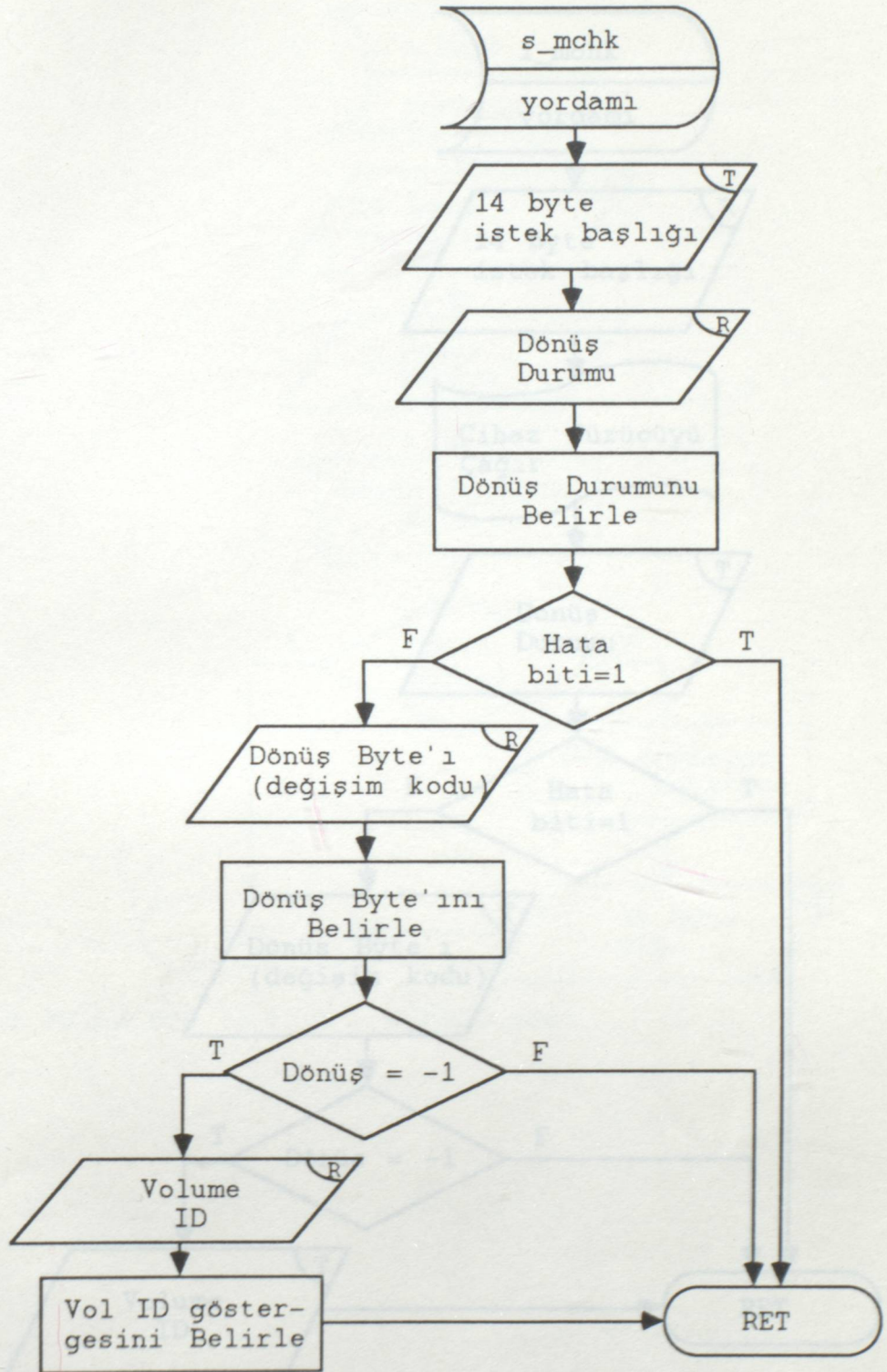
4.2.5 WRITE (OUTPUT) Yordamı

Bu yordam da, READ yordamına benzer bir mantıklar çalışır. Yalnız, veri akışı tersine işlemektedir. Yani, terminalden master'a veri akışı sözkonusudur. Prensipte, gene tampon kullanma esasına dayanır. Aynı şekilde hata oluştuğu takdirde, master terminal'e veri akışını kesmesi için bir uyarı (durum sözcüğü) göndermektedir.

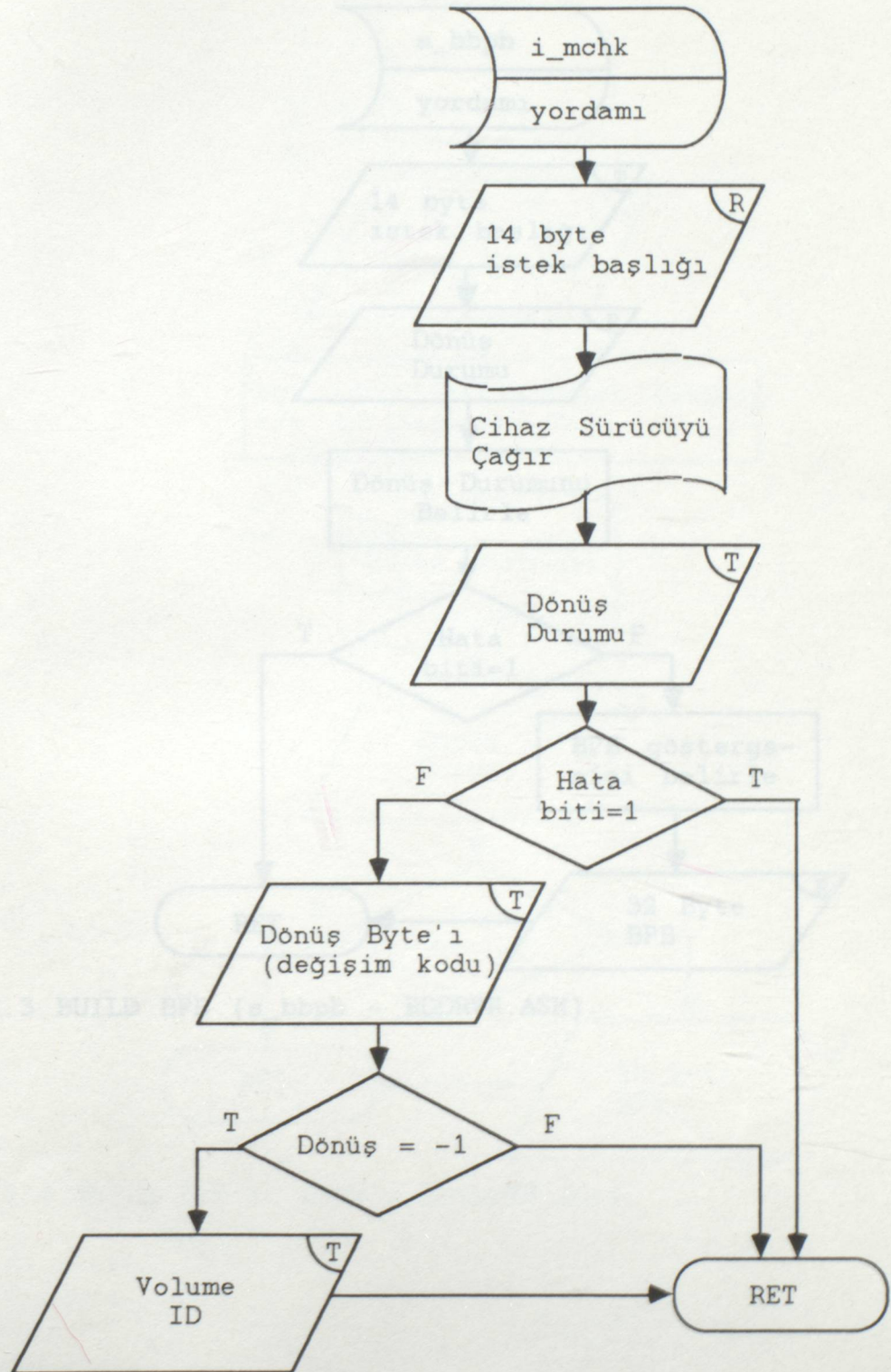
WRITE yordamının akış diyagramı, RDDRVR.ASM için Akış 4.2.7'de, INTSRVR.ASM için, Akış 4.2.8'de, gösterilmiştir.

4.2.6 WRITE WITH VERIFY (OUTPUT WITH VERIFY) Yordamı

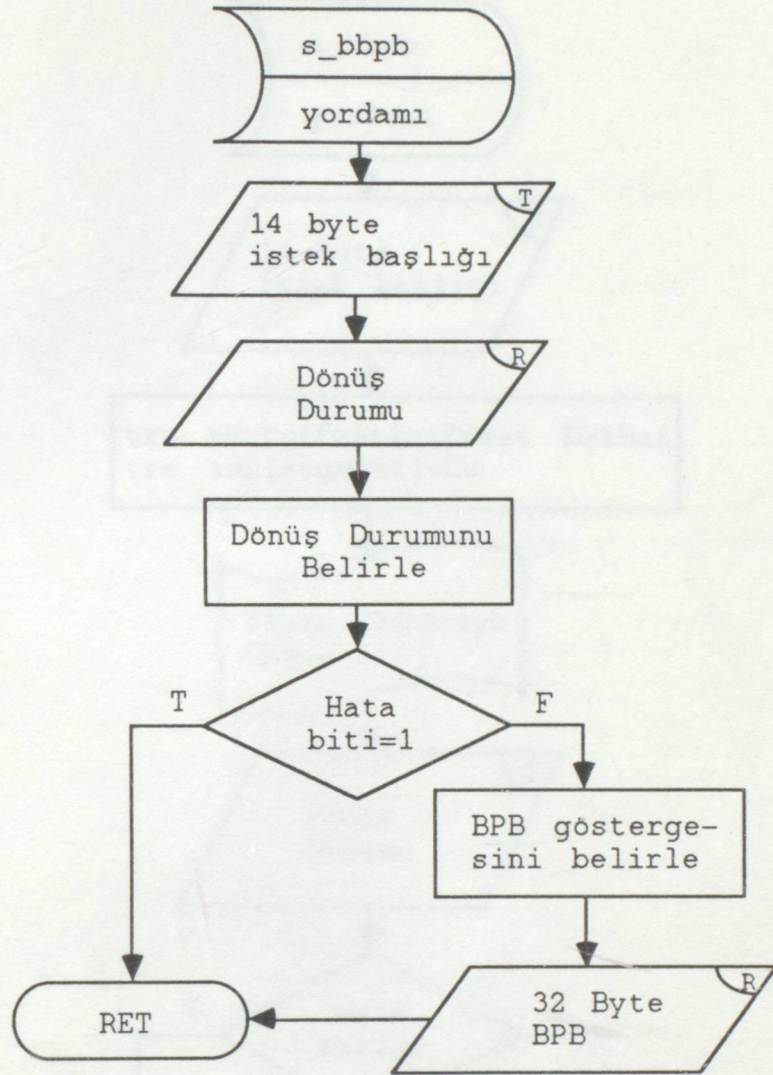
Bu yordam da tamamen WRITE yordamıyla aynıdır. Tek farklılık, yazılan verinin okunup yazmanın başarısının kontrol edilmesidir. Akış Diyagramları WRITE ile aynıdır. Burada, ilaveten master okuma yapmaktadır. Bu kontrol için yapılan okuma, hattan herhangi bir veri transferine yol açmadığından fazla bir ağırlaşma görülmez.



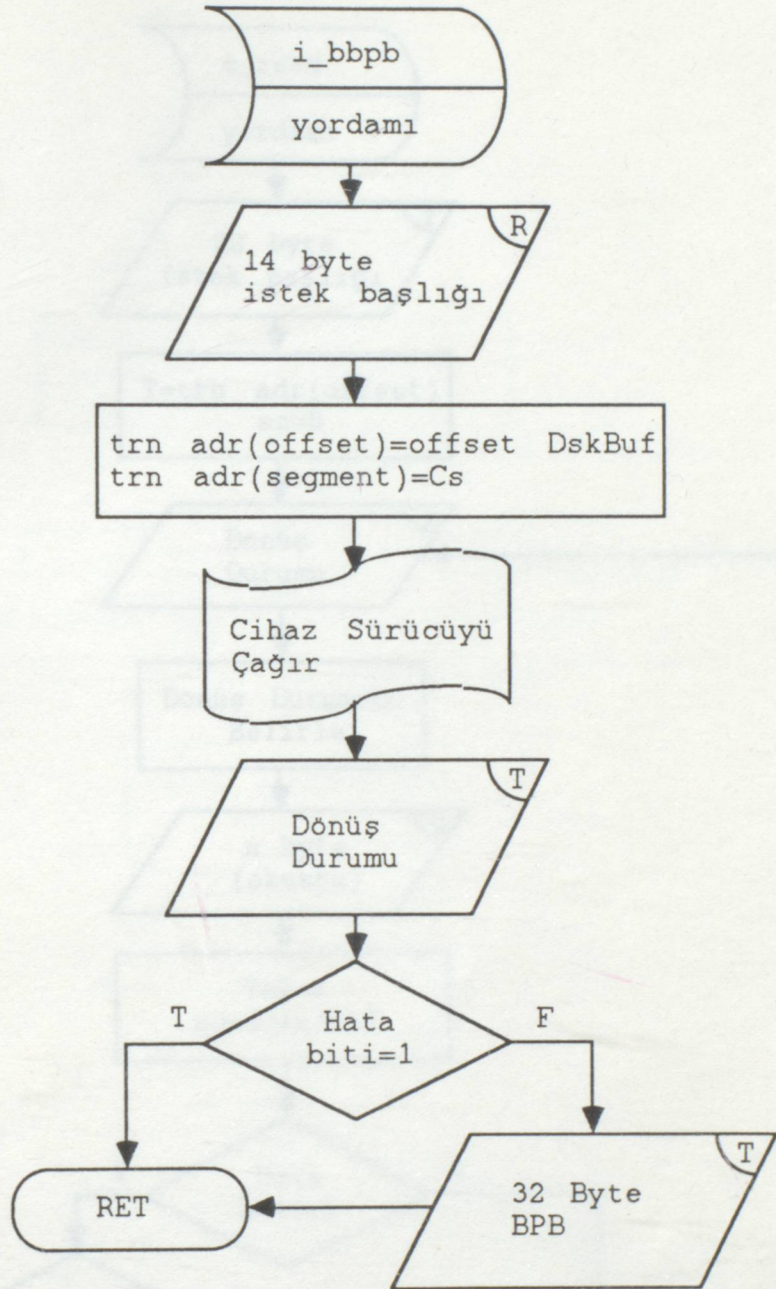
Akış 4.2.1 MEDIA CHECK (s_mchk - RDDRVR.ASM).



Akış 4.2.2 MEDIA CHECK (i_mchk - INTSRVR.ASM).

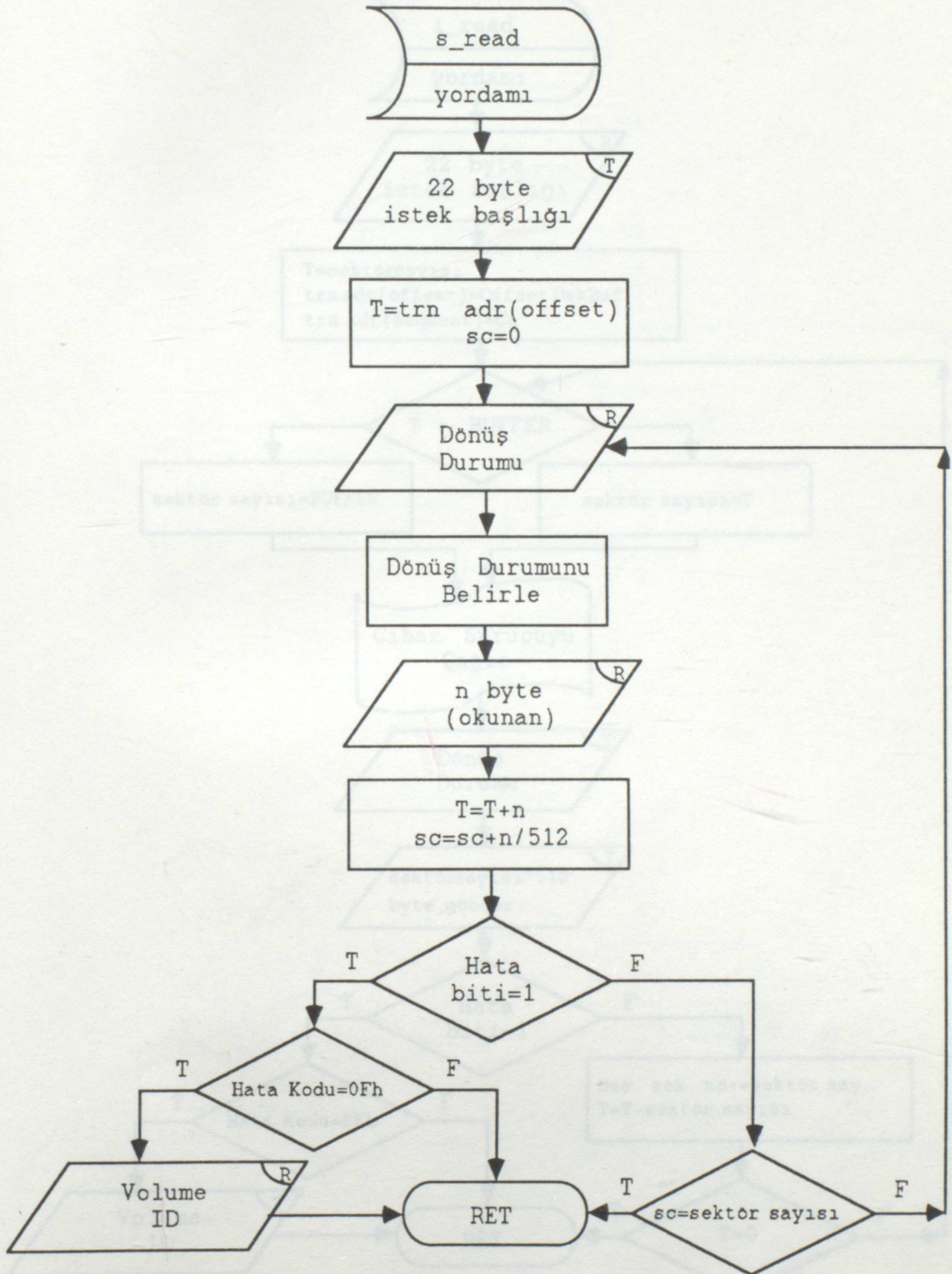


Akış 4.2.3 BUILD BPB (s_bbp - RDDVR.ASM).

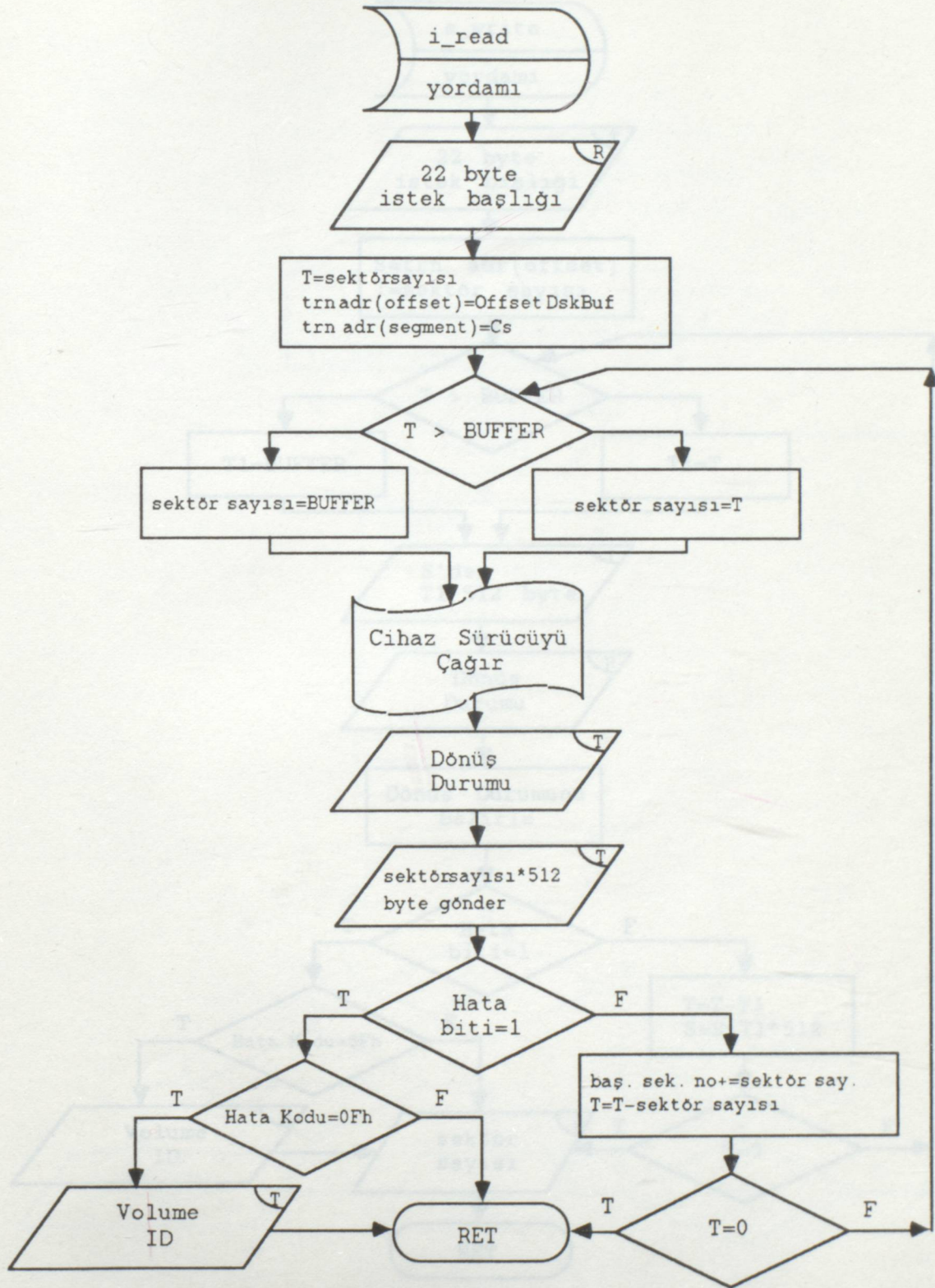


Akış 4.2.4 BUILD BPB (i_bbpb - INTSRVR.ASM).

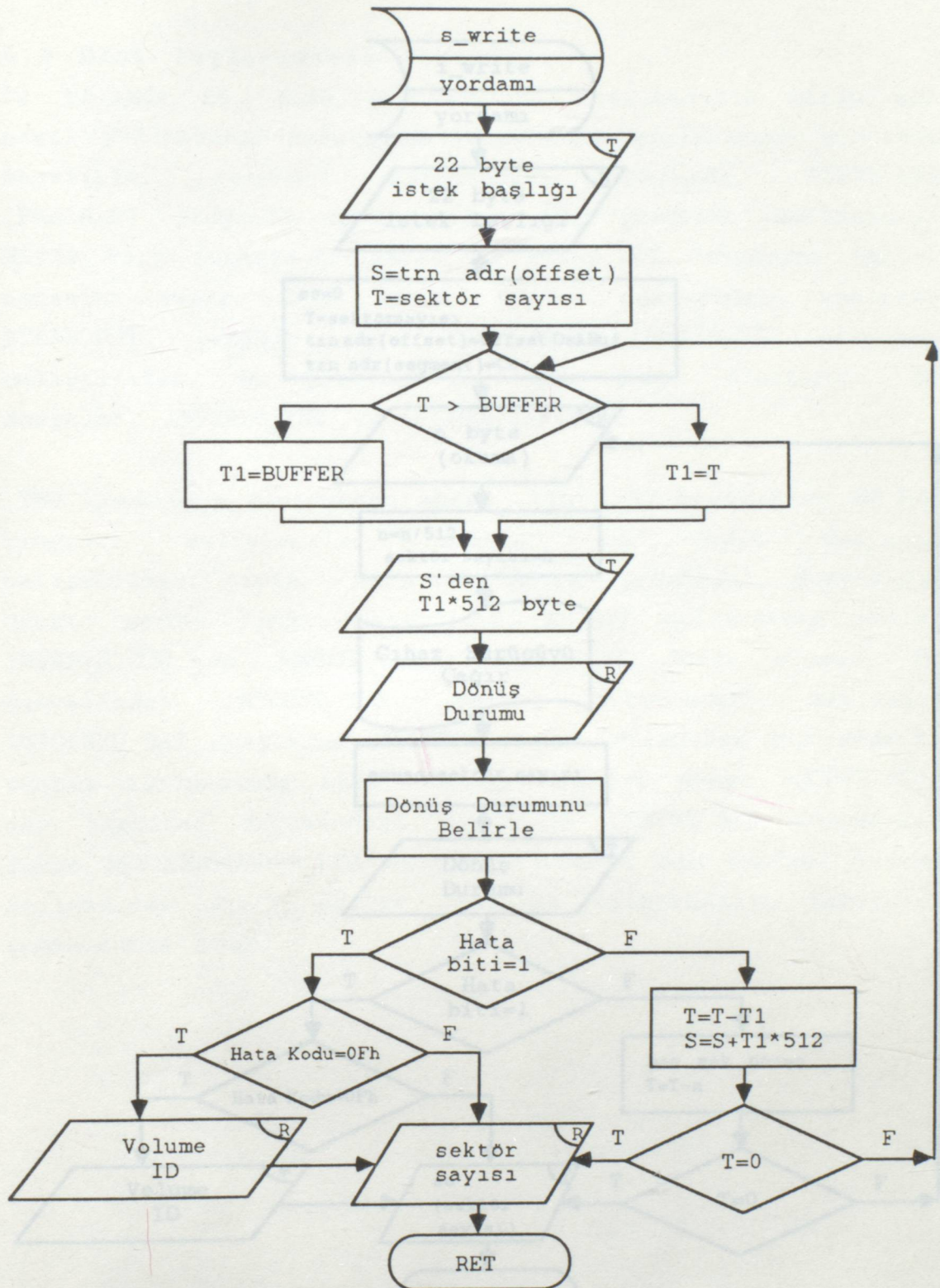
Akış 4.2.5 READ (s_read - INTSRVR.ASM).



Akış 4.2.5 READ (s_read - RDDRVR.ASM).



Akış 4.2.6 READ (i_read - INTSRVR.ASM).



Akış 4.2.7 WRITE (s_write - RDDRVR.ASM).

4.3 Disk Paylaşımı

Bu bölümde de, disk

nasıl yüklenmesi

öncelikle, terminal

(PMAIN.C) programı

RS232 kartı

adresini yazar

PTERM.OUT

valiştirilir.

dosyalar, INTSRVR

THP dosyası

programı

çalıştırılır.

diskte mevcut

INTSRVR.COM

dosyalardan

AUTOEXEC.BAT

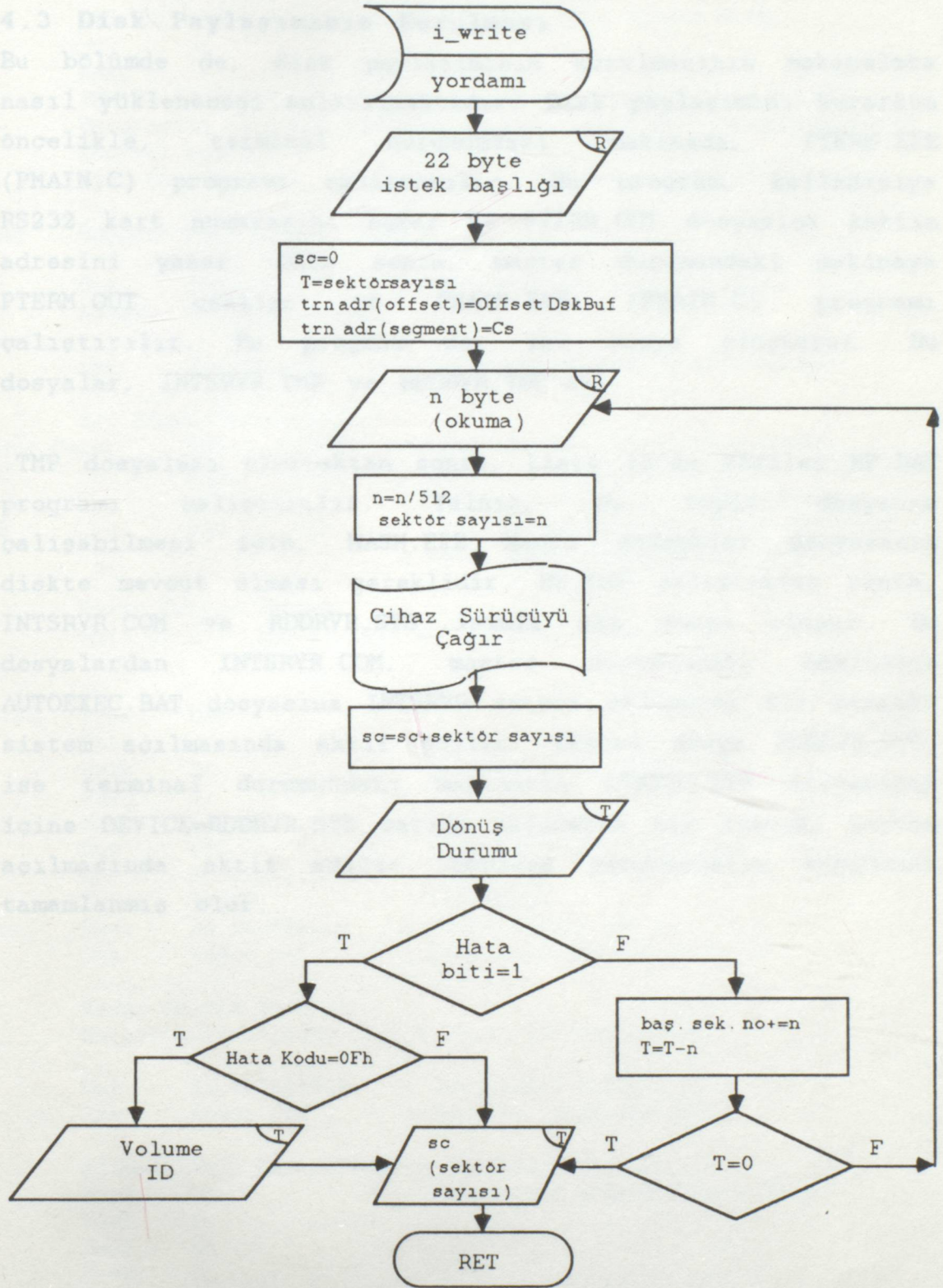
sistem açılmasında

ise terminal

ine SERVICE-USER

açılmasında

tamamlanmış



Akış 4.2.8 WRITE (i_write - INTSRVR.ASM).

4.3 Disk Paylaşımının Kurulması

Bu bölümde de, disk paylaşımının kurulmasının makinalara nasıl yükleneceği anlatılmaktadır. Disk paylaşımını kurarken öncelikle, terminal durumundaki makinada, PTERM.EXE (PMAIN.C) programı çalıştırılır. Bu program, kullanıcıya RS232 kart numarasını sorar ve PTERM.OUT dosyasına kartın adresini yazar. Daha sonra, master durumundaki makineye PTERM.OUT çekilir ve PMAIN.EXE (PMAIN.C) programı çalıştırılır. Bu program da, iki dosya oluşturur. Bu dosyalar, INTSRVR.TMP ve RDDRVR.TMP'dir.

.TMP dosyaları oluştuktan sonra, Liste 12'de verilen MP.BAT programı çalıştırılır. Yalnız, bu toplu dosyanın çalışabilmesi için, MASM.EXE Macro Assembler dosyasının diskte mevcut olması gereklidir. MP.BAT çalıştıktan sonra, INTSRVR.COM ve RDDRVR.SYS isimli iki dosya oluşur. Bu dosyalardan INTSRVR.COM, master durumundaki makinenin AUTOEXEC.BAT dosyasına INTSRVR satırı eklenerek bir sonraki sistem açılmasında aktif edilir. İkinci dosya RDDRVR.SYS, ise terminal durumundaki makinenin CONFIG.SYS dosyasının içine DEVICE=RDDRVR.SYS satırı eklenerek bir sonraki sistem açılmasında aktif edilir. Böylece programların kurulması tamamlanmış olur.

INTOCPC.ASM

```

; Yazan: A. Sedat ALIŞ
; (c) Kopya Hakkı Ve Bütün Hakları Yalnızca A. Sedat ALIŞ'a aittir.
; (c) Copyright & All Rights Reserved by A. Sedat ALIŞ
;
; Amaç :
; IBM PC-AT/PC-XT için Asenkron İletişim Kesme Tutucu Yordamı.
; Bu program Asenkron iletişim kesmelerinden veri hazır kesmesini
; kontrol eder.
; Eğer veri hazırsa, "Data Ready -> XX" mesajı ekranda görülür.

Cseg Segment Para Public 'code'
Org 100h ; .COM dosyalar için orijin
Assume Cs:Cseg,Ds:Cseg

Start:
Jmp Initialize ; ilk koşullandırma yordamına git

; G/Ç Adres Tanımlamaları
Async_Card Equ 03F8h ; Asenkron Seri port 1 G/Ç adresi
SIC Equ 20h ; Location of System Interrupt Controller
; Sistem Kesme Kontrolörü Adresi
IMR Equ 21h ; Location of Interrupt Mask Register
; Kesme maske saklayıcısı Adresi

; Kesme Tutucu Yordamı
; Seri port 1 (IRQ 4) için 0Ch yazılım kesmesi

Int0C Proc Far
Sti ; Kesmelere izin ver
Push Dx ; Saklayıcıları koru
Push Ax
Mov Al,20h ; Kesmeyi cevapla
Out SIC,Al ; Sistem kontrolörüne yaz
Mov Dx,Async_Card+2 ; IIR
In Al,Dx ; IIR okuma
Test Al,00000001B ; Kesme isteği var mı?
Jnz Async_ret ; Eğer kesme isteği yoksa dön

; Kesme Servis Yordamı :
; Buraya gelindiğinde, artık kesme gerçekleşmiştir.

Cmp Al,00000100B ; Veri hazır kesmesi mi?
Jne Async_ret ; Hayırsa, kesmeden dön

; Alınmış Veri Hazır Kesmesi Sezildi.
Push Ds ; Kullanılan saklayıcıları koru
Push Bx
Push Cx
Push Si
Xor Bh,Bh ; Bh <- 0
Mov Dx,Async_Card ; RBR
In Al,Dx ; Seri porttan veriyi oku
Push Ax ; Ax'i sakla (Al-Alınan veri)

```



```

Mov     Cl,4                ; Kaydırma sayısı
Shr     Al,Cl               ; Yüksek anlamlı dört biti getir
Mov     Bl,Al
Mov     Al,Cs:HexTbl[Bx]; 0-15 arası sayılar için hex. kodu bul
Mov     Cs:Hex[0],Al        ; Karakteri ekran geçici tamponuna koy
Pop     Ax                  ; Ax'i eski haline getir
And     Al,0Fh              ; Alçak anlamlı dört biti getir
Mov     Bl,Al
Mov     Al,Cs:HexTbl[Bx]; 0-15 arası sayılar için hex. kodu bul
Mov     Cs:Hex[1],Al        ; Karakteri ekran geçici tamponuna koy
Mov     Ax,0B800h           ; Renkli cart için ekran segment başlangıcı
Mov     Ds,Ax               ; Ekran taban segment adresini belirle
Mov     Bx,Offset Msg       ; Mesajın offset'i
Xor     Si,Si               ; Ekran tamponuna koy

; Karakterleri (mesajı) görüntülemek için çevrim
Lp:
Mov     Al,Cs:[Bx]          ; Yazılacak byte'ı al
Or      Al,Al               ; Sıfır mı? (katar sonu)
Jz      EndLp               ; Sıfırsa, çevrimden çık
Mov     Ds:[Si],Al          ; Byte'ı ekrana direk koy
Inc     Bx                  ; Katar göstergesine 1 ekle
Add     Si,2                ; Ekran göstergesine 2 ekle
Jmp     Lp                  ; Diğer adıma git

EndLp:
Pop     Si                  ; Saklayıcıları eski haline getir
Pop     Cx
Pop     Bx
Pop     Ds

Async_ret:
Pop     Ax
Pop     Dx
Iret

Int0C   Endp

; Code Segment içerisindeki veriler:
Msg      Db      'Data Ready -> '
Hex      Db      '00',0
HexTbl   Db      '0123456789ABCDEF' ; Hex. karakterler için tablo

Initialize:

; Asenkron iletişim kesmesini (0Ch) ilk koşullandır

; 0Ch kesmesi için kesme vektör adresini belirle.
Mov     Ah,25h              ; Vektör belirleme isteği
Mov     Al,0Ch              ; İletişim kesme numarası
Mov     Dx,Offset Int0C     ; Kesmenin başlangıç adresi
Int     21h                 ; DOS'u çağır

In      Al,IMR              ; Kesme maske saklayıcısını oku
And     Al,11101111B        ; Async kesmesinin (IRQ 4) maskesini kaldır

```



```

Out    IMR,A1          ; Yeni maskeyi yaz
Mov    Dx,3FCh         ; MCR
Mov    A1,00001011B    ; Out2=1 (kesme izni) , RTS, DTR
Out    Dx,A1           ; MCR'yi belirle
Sub    Dx,3            ; IER
In     A1,Dx           ; Halihazırdaki durumu al
Or     A1,00000001B     ; Alınmış Veri Hazır Kesmesine izin ver.
Out    Dx,A1           ; IER'i belirle
Nop                      ; Boş makina komutu

```

```

; Dur ve Kalıcı Ol yordamını çağır
Mov    Dx,Offset Initialize
Int    27h

```

```

Cseg   Ends
Start: End    Start
      Jmp    Initialize

```

1. G/C Adres Tanımlamaları

```

Async_Card Equ 03F5h ; Asenkron seri port 1 adresi
REARM      Equ 02F4h ; IRQ 4 için Global Rearm adresi
SPC_EOI    Equ 44h   ; IRQ 4 için özel kesme sonu (Specific EOI)
EOI        Equ 20h   ; Özel olmayan kesme sonu (Nonspecific EOI)
; EOI - End of Interrupt
OCR        Equ 09Ch  ; Kesme kontrolörünün ayımsal kontrol
; saklayıcısı
IMR        Equ 21h   ; Kesme maske saklayıcısının yordamı

```

2. Kesme Tutucu Yordamı

Seri port 1 (IRQ 4) için 0Ch yazılan kesme

```

Int0C Proc Far
      Sti                      ; Kasmalara izin ver
      Push Dx                 ; Saklayıcıları koru
      Push Ax
      Mov  Dx,Async_Card+1    ; IIR kesme tanımı saklayıcısı
      In  A1,Dx               ; Saklayıcıya oku
      Test A1,1               ; Kesme var mı?
      Jnz  Async_ret          ; Hayır, son

```

3. Kesme Servis Yordamı

Bu yaya geldiğinde, artık kesme gerçekleştirilmiştir.

```

Cseg   A1,00001005B      ; Veri hazır durumu mı?
      Jnz  Async_ret      ; Hayır, kesmeden dön

```

4. Asenkron Veri Hazır Kesmesi Servisi

```

      Push Dx             ; Kullanılan saklayıcıları koru
      Push Bx
      Push Dx
      Push SI
      Mov  Bx,Dx           ; Bx 0-3
      Mov  Dx,3F5h        ; IIR
      In  A1,Dx            ; Seri porttan veriyi oku

```



```

; Yazan: A. Sedat ALIŞ
; (c) Kopya Hakkı Ve Bütün Hakları Yalnızca A. Sedat ALIŞ'a aittir.
; (c) Copyright & All Rights Reserved by A. Sedat ALIŞ
;
; Amaç :
; IBM PS/2 (herhangibir model) için Asenkron İletişim Kesme Tutucu
; Yordamı.
; Bu program Asenkron iletişim kesmelerinden veri hazır kesmesini
; kontrol eder.
; Eğer veri hazırsa, "Data Ready -> XX" mesajı ekranda görülür.

Cseg      Segment Para Public 'Code'

Org       100h                ; .COM dosyalar için orijin
Assume    Cs:Cseg,Ds:Cseg

Start:
    Jmp     Initialize        ; ilk koşullandırma yordamına git

; G/Ç Adres Tanımlamaları
Async_Card Equ     03F8h      ; Asenkron seri port 1 adresi
REARM      Equ     02F4h      ; IRQ 4 için Global Rearm adresi
SPC_EOI    Equ     64h        ; IRQ 4 için özel kesme sonu (Specific EOI)
EOI        Equ     20h        ; Özel olmayan kesme sonu (Nonspecific EOI)
; EOI - End of interrupt
OCR        Equ     03Ch        ; Kesme kontrolörünün işlemsel kontrol
; saklayıcısı
IMR        Equ     21h        ; Kesme maske saklayıcısının adresi

; Kesme Tutucu Yordamı
; Seri port 1 (IRQ 4) için 0Ch yazılım kesmesi

Int0C      Proc     Far
    Sti                     ; Kesmelere izin ver
    Push     Dx              ; Saklayıcıları koru
    Push     Ax
    Mov      Dx,Async_Card+2 ; IIR kesme tanım saklayıcısı
    In       Al,Dx           ; Saklayıcıyı oku
    Test     Al,1            ; Kesme var mı?
    Jnz      Async_ret       ; Hayırsa, dön

; Kesme Servis Yordamı :
; Buraya gelindiğinde, artık kesme gerçekleşmiştir.

    Cmp      Al,00000100B     ; Veri hazır kesmesi mi?
    Jne      Async_ret       ; Hayırsa, kesmeden dön

; Alınmış Veri Hazır Kesmesi Sezildi.
    Push     Ds              ; Kullanılan saklayıcıları koru
    Push     Bx
    Push     Cx
    Push     Si
    Xor      Bh,Bh           ; Bh <- 0
    Mov      Dx,3F8h         ; RBR
    In       Al,Dx           ; Seri porttan veriyi oku

```



```

Push    Ax                ; Ax'i sakla (Al-Alınan veri)
Mov     Cl,4              ; Kaydırma sayısı
Shr     Al,Cl             ; Yüksek anlamlı dört biti getir
Mov     Bl,Al
Mov     Al,Cs:HexTbl[Bx]; 0-15 arası sayılar için hex. kodu bul
Mov     Cs:Hex[0],Al      ; Karakteri ekran geçici tamponuna koy
Pop     Ax                ; Ax'i eski haline getir
And     Al,0Fh            ; Alçak anlamlı dört biti getir
Mov     Bl,Al
Mov     Al,Cs:HexTbl[Bx]; 0-15 arası sayılar için hex. kodu bul
Mov     Cs:Hex[1],Al      ; Karakteri ekran geçici tamponuna koy
Mov     Ax,0B800h         ; Renkli cart için ekran segment başlangıcı
Mov     Ds,Ax             ; Ekran taban segment adresini belirle
Mov     Bx,Offset Msg     ; Mesajın offset'i
Xor     Si,Si             ; Ekran tamponuna koy

```

; Karakterleri (mesajı) görüntülemek için çevrim

```

Lp:
Mov     Al,Cs:[Bx]        ; Yazılacak byte'ı al
Or      Al,Al             ; Sıfır mı? (katar sonu)
Jz      EndLp             ; Sıfırsa, çevrimden çık
Mov     Ds:[Si],Al        ; Byte'ı ekrana direk koy
Inc     Bx                ; Katar göstergesine 1 ekle
Add     Si,2              ; Ekran göstergesine 2 ekle
Jump    Lp                ; Diğer adıma git

```

```

EndLp:
Pop     Si                ; Saklayıcıları eski haline getir
Pop     Cx
Pop     Bx
Pop     Ds

```

```

Async_ret:
Cli                      ; Kesmeleri maskele
Mov     Al,EOI            ; Özel olmayan kesme sonunu (EOI) gönder
Out     OCR,Al            ; Asenkron kesmesine tekrar izin ver(Rearm)
Mov     Dx,REARM
Out     Dx,Al
Pop     Ax                ; Saklayıcıları eski haline getir
Pop     Dx
Iret
Int0C   Endp

```

```

; Code Segment içerisindeki veriler:
Msg     Db    'Data Ready -> '
Hex     Db    '00',0
HexTbl  Db    '0123456789ABCDEF' ; Hex. karakterler için tablo

```

Initialize:

```

; Asenkron iletişim kesmesini (0Ch) ilk koşullandır
Cli                      ; Disable interrupts

```



```

; 0Ch kesmesi için kesme vektör adresini belirle.
Mov     Ah,25h      ; Vektör belirleme isteği
Mov     Al,0Ch      ; İletişim kesme numarası
Mov     Dx,Offset Int0C ; Kesmenin başlangıç adresi
Int     21h         ; DOS'u çağır

; 4. düzey kesmelerin (IRQ 4) maskesini kaldır
In      Al,IMR      ; Kesme maske saklayıcısını oku
And     Al,11101111B ; Async kesmesinin (IRQ 4) maskesini kaldır
Out     IMR,Al       ; Yeni maskeyi yaz
Mov     Al,SPC_EOI   ; Kesme isteklerine cevap verebilmek için,
Out     OCR,Al       ; IRQ 4 için özel olmayan EOI

; Durma (Stop) modu tanımlama
Stop1   Mov     Dx,3FCh ; MCR
Stop2   Mov     Al,00001011B ; Out2=1 (kesme izni) , RTS, DTR
Out     Dx,Al        ; MCR'yi belirle
Sub     Dx,3         ; IER
Word5   In      Al,Dx ; Halihazırdaki durumu al
Word6   Or      Al,00000001B ; Alınmış Veri Hazır Kesmesine izin ver.
Word7   Out     Dx,Al ; IER'i belirle
Word8   Nop       ; Boş makina komutu

Sti     ; Kesmelere izin ver

; 332ini Dur ve Kalıcı Ol yordamını çağır
Mov     Dx,Offset Initialize
Int     27h

Cseg    Ends
End     Start

```



```

; Yazan: A. Sedat ALIŞ
; (c) Kopya Hakkı Ve Bütün Hakları Yalnızca A. Sedat ALIŞ'a aittir.
; (c) Copyright & All Rights Reserved by A. Sedat ALIŞ
;
; Amaç : RS232 ilk koşullandırma makrosu.
;
; Eşlik (Parity) biti tanımlamaları
PNone Equ 00B ; Eşlik yok
POdd Equ 01B ; Eşlik çift
PEven Equ 11B ; Eşlik tek
;
; Durma (Stop) biti tanımlamaları
Stop1 Equ 0 ; (1) Durma biti
Stop2 Equ 1 ; (2) Durma biti
;
; Bit genişliği (word length) tanımlamaları
Word5 Equ 00B ; Sözcük genişliği 5 bit
Word6 Equ 01B ; Sözcük genişliği 6 bit
Word7 Equ 10B ; Sözcük genişliği 7 bit
Word8 Equ 11B ; Sözcük genişliği 8 bit
;
; Makro tanımlaması
rs232init Macro PortAdr, Baud, Parity, Stop, Wordlen
Mov Dx, PortAdr
Mov Ch, Baud
Mov Al, Parity
Shl Al, 1
Or Al, Stop
Shl Al, 1
Shl Al, 1
Or Al, Wordlen
Call Init_RS232
Endm

```



```

; Yazan: A. Sedat ALIS
; (c) Kopya Hakkı Ve Bütün Hakları Yalnızca A. Sedat ALIŞ'a aittir.
; (c) Copyright & All Rights Reserved by A. Sedat ALIŞ
;
; Amaç : RS232 portunu ilk koşullandırma programı.

```

```

Baud_Rate_Table Label Word
Dw 1,2,3,4,5,6,8,12,16,24,32,48,58,64,96,192

```

```

; Amaç :
; RS232 portunu ilk koşullandırma
Init_rs232 Proc Near

```

```

; (AL) ilk koşullandırma paramterlerini içerir.
;

```

```

; (bits 4,3) (bit 2) (bits 1,0)
; --Eşlik--- -Durma biti- -Sözcük genişliği-
; X0 - Yok 0 - 1 00 - 5 bit
; 01 - Tek 1 - 2 01 - 6 bit
; 11 - Çift 10 - 7 bit
; 11 - 8 bit
;
; (CH) baud oranı.
; 0 - 115200 1 - 57600 2 - 38400 3 - 28800
; 4 - 23040 5 - 19200 6 - 14400 7 - 9600
; 8 - 7200 9 - 4800 10 - 3600 11 - 2400
; 12 - 2000 13 - 1800 14 - 1200 15 - 600
;

```

```

DX = RS232 port adresi

```

```

Push Si ; Saklayıcıları koru
Push Bx
Mov Ah,Al ; Parametreleri sakla
Add Dx,3 ; LCR'yi işaret et
In Al,Dx ; LCR'yi oku
Or Al,80h ; Bölücü Tamponu Erişim bitini 1 yap
Out Dx,Al ; LCR'ye yaz
Mov Bl,Ch ; Baud oranını getir
Xor Bh,Bh ; Bh = 0
Shl Bx,1 ; Sözcük değerler için *2
Mov Si,Bx
Add Si,Offset Baud_Rate_Table
Sub Dx,3 ; Bölücü tutucu, düşük anlamlı byte'ı
Lodsb ; Baud oranının düşük anlamlı byte'ını getir
Out Dx,Al ; Saklayıcıyı belirle
Lodsb ; Baud oranının yüksek anlamlı byte'ını getir
Inc Dx ; Bölücü tutucu, yüksek anlamlı byte'ı
Out Dx,Al ; Saklayıcıyı belirle
Add Dx,2 ; LCR'yi göster
Mov Al,Ah ; Port ilk koşullandırma parametreleri
And Al,00011111b ; Düşük anlamlı 5 bitini maskeleye
Out Dx,Al ; LCR'yi belirle
Pop Bx
Pop Si
Ret

```

```

Init_rs232 Endp

```



```
; Yazan: A. Sedat ALIŞ
; (c) Kopya Hakkı Ve Bütün Hakları Yalnızca A. Sedat ALIŞ'a aittir.
; (c) Copyright & All Rights Reserved by A. Sedat ALIŞ
;
; Amaç : RS232 veri transferi ile ilgili altprogramlar.
```

```
; RS232 portundan bir byte gönderme
```

```
; AH = Gönderilecek byte
```

```
_tst_trn Proc Near
    Mov     Al,Ah          ; Gönderilecek byte
    Out     Dx,Al
    Add     Dx,6           ; Modem Status Register
T1:
    In      Al,Dx          ; MSR'i oku
    Test    Al,00000010B   ; Delta-Data-Set-Ready ?
    Jz      T1
    Sub     Dx,6           ; Transmitter Holding Register
    Ret
_tst_trn Endp
```

```
; RS232 portundan bir byte okuma
```

```
; AH = Okunacak byte
```

```
_tst_rcv Proc Near
    Add     Dx,5           ; Line Status Register
T2:
    In      Al,Dx          ; LSR'i oku
    Test    Al,1           ; Data Ready ?
    Jz      T2
    Sub     Dx,5           ; Receiver Buffer Register
    In      Al,Dx          ; Byte'ı oku
    Mov     Ah,Al          ; Byte'ı Ah'a ata
```

```
; Karşı taraftaki bilgisayarın DDSR durumunu aktif yapma.
```

```
Add     Dx,4             ; Modem Control Register
In       Al,Dx            ; MCR oku
Xor      Al,00000001B     ; Data-Terminal-Ready'i deęille(not)
```

```
; Karşı tarafın DDSR'sini aktif yap.
```

```
Out      Dx,Al            ; MCR'yi tekrar belirle
Sub      Dx,4
Ret
```

```
_tst_rcv Endp
```

```
; RS232 portundan n byte gönderme
```

```
; Giriş saklayıcıları :
```

```
; Dx = Port adresi,
; Ds:Si = Veri alanına gösterge,
; Cx = Gönderilecek byte sayısı.
```

```
_rstrnn Proc Near
    Push    Si
; Delta-Data-Set-Ready bitini sıfırlamak için MSR oku.
    Add     Dx,6           ; MSR
```



```

;      Byte alma
;      AH = Alınan byte
_rsrcvb Proc Near
    Call _tst_rcv
    Ret
_rsrcvb Endp

L1:
;      Word alma
;      AX = Alınan word
_rsrcvw Proc Near
    Push Cx
    Call _tst_rcv
    Mov C1,Ah
    Call _tst_rcv
    Mov A1,C1
    Pop Cx
    Ret
_rsrcvw Endp

;      Word gönderme
;      AX = Gönderilecek word
_rstrnw Proc Near
    Push Ax
    Mov Ah,A1
;      Delta-Data-Set-Ready bitini sıfırlamak için MSR oku.
    Add Dx,6 ; MSR
    In A1,Dx
    Sub Dx,6
    Call _tst_trn
    Pop Ax
    Call _tst_trn
    Ret
_rstrnw Endp

L3:
    Mov [SI],Ah
    Inc SI
    Loop L3

L4:
    Pop Ax
    Pop SI
    Ret
_rsrcvb Endp

;      Byte gönderme
;      AH = Gönderilecek byte
_rstrnb Proc Near
;      Delta-Data-Set-Ready bitini sıfırlamak için MSR oku.
    Add Dx,6 ; MSR
    In A1,Dx
    Sub Dx,6
    Call _tst_trn
    Ret
_rstrnb Endp

```



```

In      Al,Dx
Sub     Dx,6
Mov     Ah,Cl
Call    _tst_trn      ; Alçak anlamlı byte'ı gönder
Mov     Ah,Ch
Call    _tst_trn      ; Yüksek anlamlı byte'ı gönder
Jcxz    L2

L1:     Mov     Ah,[Si]      ; Byte'ı getir
        Call    _tst_trn    ; Byte'ı gönder
        Inc     Si
        Loop    L1

L2:     Pop     Si
        Ret

_rstrnn Endp

; RS232 portundan n byte alma
; Giriş saklayıcıları :
; Dx = Port adresi,
; Ds:Si = Veri alanına gösterge.
; Dönüş saklayıcıları:
; Ax = Alınan byte sayısı.

_rsrcvn Proc Near
        Push    Si
        Call    _tst_rcv
        Mov     Cl,Ah      ; Alçak anlamlı byte
        Call    _tst_rcv
        Mov     Ch,Ah      ; Yüksek anlamlı byte
        Push    Cx          ; Transfer edilen byte sayısını sakla
        Jcxz    L4          ; Byte sayısı sıfırsa dön

L3:     Call    _tst_rcv
        Mov     [Si],Ah     ; Alınan byte'ı [Si]'ye at
        Inc     Si
        Loop    L3

L4:     Pop     Ax          ; Alınan byte sayısı
        Pop     Si
        Ret

_rsrcvn Endp

; Byte gönderme
; AH = Gönderilecek byte
_rstrnb Proc Near
; Delta-Data-Set-Ready bitini sıfırlamak için MSR oku.
        Add     Dx,6        ; MSR
        In      Al,Dx
        Sub     Dx,6
        Call    _tst_trn
        Ret

_rstrnb Endp

```



```

/*      Yazan: A. Sedat ALIŞ
      (c) Kopya Hakkı Ve Bütün Hakları Yalnızca A. Sedat ALIŞ'a aittir.
      (c) Copyright & All Rights Reserved by A. Sedat ALIŞ

```

Amaç :

Disk Sürücü Yordamlarının, master ile iletişim kurması için kullanacağı seri portun adresini PTERM.OUT dosyasına yazar. */

```
#include <stdio.h>
```

```
main()
```

```
{
```

```
    FILE *fp;
```

```
    int card,port;
```

```
    #include <dos.h>
```

```
    fp=fopen("PTERM.OUT","wt");
```

```
    /* Kart numarasının sorulması */
```

```
L1:
```

```
    printf("RS232 Cart Numarası (0-3): ");
```

```
    scanf("%d",&card);
```

```
    port=testport(card);
```

```
    if(port==0 || card<0 || card >3) {
```

```
        printf("Bu cart mevcut değil\n");
```

```
        goto L1;
```

```
    }
```

```
    fprintf(fp,"%04X",port);
```

```
    fclose(fp);
```

```
}
```

```
    char far *a;
```

```
    char buffer[131];
```

```
    /* Diğer makinenin RS232 port adresini oku */
```

```
    fi=fopen("PTERM.OUT","r");
```

```
    if(fi == NULL) {
```

```
        printf("PTERM.OUT dosyası bulunamadı\n");
```

```
        exit(0);
```

```
    }
```

```
    fscanf(fi,"%X",&port);
```

```
    fclose(fi);
```

```
    w=ctype();
```

```
    if(w == -1) {
```

```
        printf("Makine tipi bulunamadı\n");
```

```
L0:
```

```
        printf("Makine tipini veriniz (0:PO-XT 1:PC-XT 2:PS/2) : ");
```

```
        scanf("%d",&w);
```

```
        if(w < 0 || w > 2) {
```

```
            printf("Yanlış makine tipi\n");
```

```
            goto L0;
```

```
        }
```

```
        printf("Makine tipi : %d\n",w);
```



```

/*      Yazan: A. Sedat ALIŞ
      (c) Kopya Hakkı Ve Bütün Hakları Yalnızca A. Sedat ALIŞ'a aittir.
      (c) Copyright & All Rights Reserved by A. Sedat ALIŞ

```

```

      Amaç :

```

```

      RDDVR.ASM ve INTSRVR.ASM dosyalarında, yeniden düzenlenmesi
      gereken (>x) yerleri düzenlemek ve değişken verileri ilgili
      yerlere yazmak. Sonuç olarak RDDVR.TMP ve INTSRVR.ASM
      programları, derlemek üzere oluşturulur.

```

```

      Derleme seçenekleri :

```

```

      CL /Zp /lpmain,,NUL,subs; pmain.c      */

```

```

#include <stdio.h>

```

```

#include <dos.h>

```

```

#include "pmain.h"

```

```

struct DP far p[8];

```

```

char *mt[]={"PCXT","PCAT","PS2"};

```

```

char *mtp[]={"PC-XT veya Uyumlu","PC-AT veya Uyumlu","PS/2 Ailesi"};

```

```

char *wparity[]={"PNone","POdd","PEven"};

```

```

char *wstop[]={"Stop1","Stop2"};

```

```

main()

```

```

{

```

```

    int i,in,m=1,port,baud=-1,card=0;

```

```

    int portterm,bufnum=0,parity=-1,stop=-1;

```

```

    FILE *f1,*f2;

```

```

    char far *a;

```

```

    char buffer[132];

```

```

    /* Diğer makinanın RS232 port adresini oku */

```

```

    f1=fopen("PTERM.OUT","r");

```

```

    if(f1 == NULL) {

```

```

        printf("PTERM.OUT dosyası bulunamadı\n");

```

```

        exit(0);

```

```

    }

```

```

    fscanf(f1,"%X",&portterm);

```

```

    fclose(f1);

```

```

    m=dmttype();

```

```

    if(m == -1) {

```

```

        printf("Makina tipi bulunamadı\n");

```

```

L0:

```

```

    printf("Makina tipini veriniz (0:PC-XT 1:PC-AT 2:PS/2) : ");

```

```

    scanf("%d",&m);

```

```

    if(m < 0 || m > 2) {

```

```

        printf("Yanlış makina tipi\n");

```

```

        goto L0;

```

```

    }

```

```

}

```

```

printf("Makina tipi : %s\n",mtp[m]);

```



```

/* Kart numarasının sorulması */
L1:
printf("\nRS232 Cart Numarası (0-3): ");
scanf("%d",&card);
port=testport(card);
if(port==0 || card<0 || card >3) {
    printf("Bu cart mevcut değil\n");
    goto L1;
}

/* Baud oranının sorulması */
L2:
printf("\n----- Baud Oran Kodları ----- \n");
printf(" 0 - 115200      1 - 57600      2 - 38400      3 - 28800\n");
printf(" 4 - 23040      5 - 19200      6 - 14400      7 - 9600\n");
printf(" 8 - 7200       9 - 4800       10 - 3600      11 - 2400\n");
printf("12 - 2000      13 - 1800      14 - 1200      15 - 600\n");
printf("Baud oran kodu : ");
scanf("%d",&baud);
if(baud < 0 || baud > 15) {
    printf("Yanlış baud oranı\n");
    goto L2;
}

/* Tampon sayısının sorulması */
L3:
printf("\nTampon sayısı (1-120) : ");
scanf("%d",&bufnum);
if(bufnum<1 || bufnum>120) {
    printf("Yanlış tampon sayısı\n");
    goto L3;
}

/* Eşlik biti belirlemesinin sorulması */
L4:
printf("\nEşlik (parity) modu (0-None, 1-Odd, 2-Even) : ");
scanf("%d",&parity);
if(parity<0 || parity>2) {
    printf("Yanlış eşlik belirlemesi\n");
    goto L4;
}

/* Durma biti belirlemesinin sorulması */
L5:
printf("\nDurma (Stop) biti sayısı (0-[1 bit], 1-[2 bit]) : ");
scanf("%d",&stop);
if(stop<0 || stop>1) {
    printf("Yanlış durma belirlemesi\n");
    goto L5;
}

/* Cihaz parametrelerinin alınması */
for(in=1 ; ; in++ ) if(getdevparm(in,&p[in-1])) break;
a=getdrvradr(); /* cihaz sürücü adresini al */

```



```

/* INTSRVR.TMP dosyasının oluřturulması */
f1=fopen("INTSRVR.ASM","rt");
f2=fopen("INTSRVR.TMP","wt");
while(1) {
    fgets(buffer,132,f1);
    if(feof(f1)) break;
    if(*buffer == '>') {
        switch(buffer[1]) {
            case '1':
                fprintf(f2,"MACTYPE\\tEqu\\t%s\\n",mt[m]);
                fprintf(f2,"PORTADR\\tEqu\\t%04Xh\\n",port);
                fprintf(f2,"CARDNO\\tEqu\\t%d\\n",card);
                fprintf(f2,"BUFFER\\tEqu\\t%d\\n",bufnum);
                break;
            case '2':
                i=FP_SEG(a);
                fprintf(f2,"Strategy\\tLabel\\tDword\\n");
                fprintf(f2,"\\tDw\\t%04Xh,%04Xh\\t; Strategy yordami\\n",
                    *(int far *)&a[6],i);
                fprintf(f2,"Interrupt\\tLabel\\tDword\\n");
                fprintf(f2,"\\tDw\\t%04Xh,%04Xh\\t; Interrupt yordami\\n",
                    *(int far *)&a[8],i);
                break;
            case '3':
                fprintf(f2,"\\trs232init\\tPORTADR,%d,%s,%s,Word8\\n",
                    baud,wparity[parity],wstop[stop]);
                break;
        }
    }
    else fprintf(f2,buffer);
}
fclose(f1);
fclose(f2);

/* RDDRV.R.TMP dosyasının oluřturulması */
f1=fopen("RDDRV.R.ASM","rt");
f2=fopen("RDDRV.R.TMP","wt");
while(1) {
    fgets(buffer,132,f1);
    if(feof(f1)) break;
    if(*buffer == '>') {
        switch(buffer[1]) {
            case '1':
                /* Dięer makinanın RS232 port adresi */
                fprintf(f2,"PORTADR\\tEqu\\t%04Xh\\n",portterm);
                fprintf(f2,"BUFFER\\tEqu\\t%d\\n",bufnum);
                break;
            case '2':
                fprintf(f2,"\\tDw\\t%04Xh\\t\\t\\t; Cihazın nitelięi(attribute)
                    *(int far *)&a[4]);
                break;
            case '3':
                writebpb(f2,p,in-1);
                break;
        }
    }
}

```



```
#define BYTE    unsigned char
#define WORD    unsigned int
#define DWORD   unsigned long
```

```
/* BIOS Parametre Bloğu */
```

```
struct BPB {
    WORD bpsector;      /* Bytes per sector */
    BYTE spcluster;     /* Sectors per cluster(zero based) */
    WORD rbsector;      /* Number of reserved boot sectors */
    BYTE fcopy;         /* Number of FAT copies */
    WORD reentry;       /* Number of root-directory entries */
    WORD tsector;       /* Number of total sectors */
    BYTE mdbyte;        /* Media descriptor byte */
    BYTE spfat;         /* Sectors per FAT */
    WORD sptrack;       /* Sectors per Track */
    WORD heads;         /* Number of heads */
    DWORD hsector;      /* Hidden sectors */
    DWORD tsector2;     /* Total sectors if tsector=0 */
    BYTE Reserved_2[6]; /* Reserved for DOS */
};
```

```
/* Device Parameter Block (cihaz parametre bloğu) : Int 21h fonksiyon 44h */
```

```
struct DP {
    BYTE SpecialFunctions;
    BYTE DeviceType;
    WORD DeviceAttributes;
    WORD NumberOfCylinders;
    BYTE MediaType;
    struct BPB b;      /* Bios Parametre Bloğu */
};
```

```
int getdevparm(char, struct DP far *);
void writebpb(FILE *, struct DP far [], int);
char far *getdrvradr();
```

```
void writebpb(fp, p, dc)
FILE *fp;      /* BPB'ların yazılacağı dosya */
struct DP far p[]; /* BPB'lara gösterge */
int dc;        /* BPB sayısı */
{
    int i,j;
    for(i=0; i<dc; i++) {
        fprintf(fp,"DRV_%c\t\tLabel\ttBYTE\n",'A'+i);
        fprintf(fp,"\tDW\t\t\t; Bytes per sector\n",p[i].b.bpsector);
        fprintf(fp,"\tDB\t\t\t; Sectors per cluster\n",p[i].b.spcluster);
        fprintf(fp,"\tDW\t\t\t; Reserved sectors\n",p[i].b.rbsector);
        fprintf(fp,"\tDB\t\t\t; Number of FATs\n",p[i].b.fcopy);
        fprintf(fp,"\tDW\t\t\t; Number of root-dir entries\n",p[i].b.reentry);
        fprintf(fp,"\tDW\t\t\t; Total sectors\n",p[i].b.tsector);
        fprintf(fp,"\tDB\t\t\t03Xh\t; Media descriptor\n",p[i].b.mdbyte);
        fprintf(fp,"\tDW\t\t\t; Sectors per FAT\n",p[i].b.spfat);
        fprintf(fp,"\tDW\t\t\t; Sectors per track\n",p[i].b.sptrack);
```



```

fprintf(fp, "\tDW\t%u\t; Heads\n", p[il.b.heads]);
fprintf(fp, "\tDD\t%lu\t; Number of hidden sectors\n", p[il.b.hsector]);
fprintf(fp, "\tDD\t%lu\t; Total sectors, previous is zero\n", p[il.b.tsec]);
fprintf(fp, "\tDB\t");
for(j=0; j<6; j++) fprintf(fp, "%d,", p[il.b.Reserved_2[j]]);
fprintf(fp, "0\t; Reserved\n\n");

```

```

}
fprintf(fp, "DRV_NUM\tEqu\t%d\t; Number of units\n", dc);
fprintf(fp, "BPB_PTR\tLabel\tDWORD\n");
for(i=0; i<dc; i++)
    fprintf(fp, "\tDW\tOffset DRV_%c\n", 'A'+i, 0);

```

```

Public _getdrvredr
Public _getdevparm
Public _testpart

```

```

;
; Makina tipine karar verme
;
; Qayima Dili : C
;
; Donus Degeri
;
; 0 PC XT veya uyumlu
; 1 PC AT veya uyumlu
; 2 PS/2
; -1 Bilinmeyen
;
; Nullish
;
; int st;
;
; st = datatype();

```

```

_datatype Proc      Near
;
; Push      Es
;
; Mov       Ah, 000h      ; Sistem konfigürasyon parametrelerini oku.
; Int       15h          ; BIOS'u çağır
;
; Jc        _ds1
;
; Mov       Ah, Es: (BX+02) ; Model byte
;
; Cmp       Ah, 0F8h
; Jnc       _ds2
;
; _ds2:
;
; Mov       Ax, 2         ; Makina tipi PS/2
;
; Pop       Es
; Ret
;
; _ds1:
;
; Cmp       Ah, 0F0h
; Jc        _ds2
;
; Cmp       Ah, 0F8h
; Jc        _ds3
;
; _ds3:
;
; Ret       Ax, Ax        ; Makina tipi PC/PC-XT
;
; Pop       Es
; Ret
;
; _ds1:
;
; Mov       Ah, Es: (BX+03) ; Submodel byte
;
; Cmp       Ah, 30h
; Jc        _ds2
;
; Cmp       Ah, 35h
; Jc        _ds3

```



```
;
; PCAT: Yazan: A. Sedat ALIŞ
;
; (c) Kopya Hakkı Ve Bütün Hakları Yalnızca A. Sedat ALIŞ'a aittir.
;
; (c) Copyright & All Rights Reserved by A. Sedat ALIŞ
;
;
;
; Amaç : Proje Yardımcı Altprogramları (PMAIN.C için)
```

```
_Text Segment Word Public 'Code'
```

```
Assume Cs:_Text
```

```
Public _dmttype
```

```
Public _getdrvradr
```

```
Public _getdevparm
```

```
Public _testport
```

```
; Amaç : Makina tipine karar verme.
```

```
; Çağırma Dili : C
```

```
; Dönüş değeri :
```

```
; 0 PC XT veya uyumlu
```

```
; 1 PC AT veya uyumlu
```

```
; 2 PS/2
```

```
; -1 Bilinmeyen
```

```
; Kullanım :
```

```
; int mt;
```

```
; mt = dmttype();
```

```
_dmttype Proc Near
```

```
Push Es
```

```
Mov Ah,0C0h
```

```
; Sistem konfigürasyon parametrelerini dön.
```

```
Int 15h
```

```
; BIOS'u çağır
```

```
Jc _dm1
```

```
Mov Ah,Es:[Bx+02]
```

```
; Model byte
```

```
Cmp Ah,0F8h
```

```
Jne _dm2
```

```
_PS2:
```

```
Mov Ax,2
```

```
; Makina tipi PS/2
```

```
Pop Es
```

```
Ret
```

```
_dm2:
```

```
Cmp Ah,0FAh
```

```
Je _PS2
```

```
Cmp Ah,0FCh
```

```
Je _dm3
```

```
_PCXT:
```

```
Xor Ax,Ax
```

```
; Makina tipi PC/PC-XT.
```

```
Pop Es
```

```
Ret
```

```
_dm3:
```

```
Mov Ah,Es:[Bx+03]
```

```
; Submodel byte.
```

```
Cmp Ah,04h
```

```
Je _PS2
```

```
Cmp Ah,05h
```

```
Je _PS2
```



```

_PCAT:
    Mov     Ax,1           ; Makina tipi PC-AT.
    Pop     Es
    Ret

_dm1:
    Cmp     Ah,80h
    Je      _PCXT
    Cmp     Ah,86h
    Je      _dm4
    Mov     Ax,-1          ; Makina tipi bilinmiyor.
    Pop     Es
    Ret

_dm4:
    Mov     Ax,0F000h      ; Model id byte segment.
    Mov     Es,Ax
    Mov     Ah,Es:[0FFFEh] ; Model id byte offset.
    Cmp     Ah,0FCh
    Je      _PCAT
    Jmp     _PCXT

_dmtyp e Endp

```

```

; Amaç      : Cihaz sürücünün adresini öğrenme.
; Çağırma Dili : C
; Kullanım   :
; char far *p;
; p = getdrvradr();

```

```

_getdrvradr Proc Near
    Push    Bp
    Mov     Bp,Sp
    Push    Ds
    Mov     Ah,30h          ; DOS versiyon numarasını getir
    Int     21h
    Cmp     Al,04h          ; DOS major versiyonu 4 mu?
    Stc     ; Elde = 1
    Je      D1
    Clc     ; Elde = 0

D1:
    Pushf
    Mov     Ah,1Fh
    Int     21h             ; DOS'u çağır.
; Ds:Bx = DPB(DOS Parametre Bloğu) göstergesi
    Popf
    Adc     Bx,12h          ; Cihaz sürücü adresi (elde = 1 - DOS 4.00)
    Mov     Ax,Ds:[Bx]      ; Cihaz sürücü offset adresi
    Mov     Dx,Ds:[Bx+2]    ; Cihaz sürücü segment adresi
    Pop     Ds
    Pop     Bp
    Ret     ; Dönüş değeri : Dx:Ax

_getdrvradr Endp

```



```
; Amaç      : Cihaz sürücüyü ait parametreleri öğrenme.
; Çağırma Dili : C
; Kullanım   :
; struct DP far *p;
; p = getdevparm();
```

```
_getdevparm    Proc      Near
    Push        Bp
    Mov         Bp,Sp
    Push        Ds
    Mov         Ah,44h      ; IOCTL isteği
    Mov         Al,0dh      ; Genel IOCTL isteği
    Mov         Bl,[Bp+4]   ; Ünite no. 0=default,1=A,2=B,...
    Mov         Ch,08h      ; Kategori kodu (major kod)
    Mov         Cl,60h      ; Cihaz parametrelerini çağır
    Lds         Dx,[Bp+6]   ; Parametre bloğunun adresi
    Int         21h        ; DOS kesmesini çağır
    Jc          _err1
    Xor         Ax,Ax      ; Hata yok.
    Jmp         _ret1
_err1:
    Mov         Ax,-1      ; Hata oluştu.
_ret1:
    Pop         Ds
    Pop         Bp
    Ret
_getdevparm     Endp
```

```
; Amaç      : Verilen RS232 kart numarasının port adresini getirme.
; Çağırma Dili : C
; Kullanım   :
; int r,cardno; /* 0-3 */
; r=testport(cardno);
; r=0(FALSE) kart yok, aksi halde port adresi
```

```
_testport      Proc      Near
    Push        Bp
    Mov         Bp,Sp
    Mov         Dx,[Bp+04]  ; RS232 kart no.
    Call        _getportadr
    Mov         Ax,Dx      ; Dönüş değeri
    Pop         Bp
    Ret
_testport       Endp
```

```
RBios_Dseg      Equ      0040h      ; Rom Bios Veri Alanının Segment Adresi
```



```

; Amaç : Card numarasına göre port adresini bulma.
; Çağırma Dili : Assembly.
; Giriş : DX=card no.
; Çıkış : DX=port adr.

```

```

_getportadr Proc Near
    Push Ds
    Push Bx
    Mov Ax,RBios_Dseg ; BIOS veri alanı segmenti.
    Mov Ds,Ax
    Mov Bx,Dx
    Shl Bx,1 ; *2 word değerler için
    Mov Dx,[Bx] ; port adr.
    Pop Bx
    Pop Ds
    Ret

```

```

_getportadr Endp

```

```

_Text Ends
End

```

```

>1 ; Port adresi

```

```

Cseg Segment Public 'Code'
Org 0 ; Orijinal başlangıç
Assume Cs:Cseg,Ds:Cseg

```

```

; Cihazın Başlığı (Device Header)

```

```

RD_Drvr Proc Far
    Cdl -1 ; Diğer sürüm göstergesi
    >2 ; Cihazın niteliği (attribute)
    Dv Strategy ; Strategy yordak göstergesi
    Dv Interrupt ; Keskme yordak göstergesi
    Dv 8 dup(32) ; Cihaz ismi

```

```

RH Label Word
rh_off Dv ? ; RH/offseti
rh_seg Dv ? ; RH/segmenti
vol_id Dv 11 dup(2) ; Volume ID (etiket belirtir)
_bc Dv ?

```

```

; Strategy Yordakı :

```

```

; Tutarak RH/offseti göstergesini saklar (REQUEST HEADER - RH)

```

```

; RH-RK saklayıcıları rh_seg ve rh_off değişkenlerine saklanır.

```

```

Strategy Proc Far
    Mov Cx,rh_seg.Dv
    Mov Cx,rh_off.Dv
    Ret

```

```

Strategy Endp

```



```
;
;   Yazan: A. Sedat ALIS
;   (c) Kopya Hakkı Ve Bütün Hakları Yalnızca A. Sedat ALIS'a aittir.
;   (c) Copyright & All Rights Reserved by A. Sedat ALIS
```

```
Title   Remote Device Driver
Name     RDDRvr
```

```
Include RSINIT.MAC      ; RS232 ilk koşullandırma makrosu
```

```
;   Kontrol karakterleri ile ilgili tanımlamalar
CR     Equ     0Dh      ; Carriage return
LF     Equ     0Ah      ; Line feed
ENDSTR Equ     36       ; DOS için string sonu ($)
```

```
;   Cihaz sürücü tanımlamaları
MAXCMD Equ     24       ; DOS Ver 3.20 veya yukarısı için max. komut
ERROR  Equ     8000h    ; Hata bitini birle
BUSY   Equ     0200h    ; Meşgul bitini birle
DONE   Equ     0100h    ; Yapıldı bitini birle
UNKNOWN Equ     8003h   ; Bilinmeyen komut durumu
Illegal Equ     -1      ; Yanlış komut için işaretçi
```

```
>1      ; Port adresi
```

```
Cseg    Segment Public 'Code'
Org      0              ; Orijini sıfırla
Assume   Cs:Cseg,Ds:Cseg
```

```
;   Cihaz Sürücü Başlığı (Device Header)
```

```
RD_Drvr Proc      Far
Dd        -1        ; Diğer sürücü göstergesi
>2        ; Cihazın niteliği (attribute)
Dw        Strategy   ; Strategy yordam göstergesi
Dw        Interrupt  ; Kesme yordam göstergesi
Db        8 dup(32)   ; Cihaz ismi

RH        Label     Dword
rh_off    Dw        ?      ; RH(offset)
rh_seg    Dw        ?      ; RH(segment)
vol_id    Db        12 dup(?) ; Volume ID (etiket belirteçi)
_sc       Dw        ?
```

```
;   Strateji Yordamı :
;   istek Başlığı göstergesini saklar (REQUEST HEADER - RH)
;   ES:BX saklayıcıları rh_seg ve rh_off değişkenlerine saklanır.
```

```
Strategy Proc      Far
Mov       Cs:rh_seg,Es
Mov       Cs:rh_off,Bx
Ret
Strategy  Endp
```



```

; Komut Kodu Ayrıştırma Tablosu
d_tbl Label Word
Dw s_init ; (0) Initialization
Dw s_mchk ; (1) Media check
Dw s_bbpb ; (2) Build BIOS Parameter Block
Dw Illegal ; (3) Tanımsız komut
Dw s_read ; (4) Read
Dw 3 dup(Illegal) ; (5-7) Tanımsız komutlar
Dw s_write ; (8) Write
Dw s_write ; (9) Write With Verify
Dw 15 dup(Illegal) ; (10-24) Tanımsız komut

>3 ; BIOS Parametre Bloğu

; Mesajlar :
Msg1 Db 'Remote Device Driver Version 1.0',CR,LF
Db '(c) Copyright and All Rights Reserved By Sedat ALIS',CR,LF
Db 'Remote Drives ; '
Dno1 Db 'A: - '
Dno2 Db 'A:',CR,LF,ENDSTR

; Kesme Yordamı
; Bütün cihaz sürücü işlemlerini yürütür.
; -----
Interrupt Proc Far
Push Es ; Bütün saklayıcıları koru
Push Ds
Push Ax
Push Bx
Push Cx
Push Dx
Push Si
Push Di

; RH göstergesini getir
s_init Les Bx,Cs:RH
Mov Al,Es:[Bx+2] ; Komut kodu

Cmp Al,MAXCMD ; Yasal komut mu?
Jle Ok ; Yasalsa OK'e git

s_unknown:
Mov Es:[Bx+3],UNKNOWN ; Bilinmeyen komut
Jmp Short Finish

; Cihaz Sürücü Fonksiyonunu Yürüt
Ok:
Xor Ah,Ah
Shl Ax,1 ; Word değerler için *2
Mov Si,Ax

```



```

Mov     Ax,Cs:d_tbl[Si]
Cmp     Ax,Illegal      ; Komut yanlış mı?
Je      s_unknown
Push    Cs
Pop     Ds              ; Ds <- Cs
Mov     Dx,PORTADR      ; RS232 port adresi
Push    Ax
Mov     Ah,Es:[Bx+2]    ; Komut kodu
Or      Ah,Ah           ; Ah = 0 ? (INIT)
Jz      Ok1
Call    _rstrnb         ; Komut kodunu gönder
Ok1:
Pop     Ax
Call    Ax              ; ilgili yordamı çağır
Mov     Ah,Es:[Bx+3]
; Cihaz sürücünün bitimi
Finish:
Pop     Di              ; Saklayıcıları eski haline getir
Pop     Si
Pop     Dx
Pop     Cx
Pop     Bx
Pop     Ax
Pop     Ds
Pop     Es
Ret                    ; DOS'a geri dön
Interrupt Endp

; *** Cihaz Sürücü Fonksiyonları ***
; Yordam : INIT
; Komut Kodu = 0
; Amaç : Cihaz ilk koşullandırma yordamı
; Es:Bx = istek Başlığı Başlangıç Adresi
; Ds = Cs

s_init Proc Near
Mov     Byte Ptr Es:[Bx+13],DRV_NUM ; Ünite sayısını DRV_NUM yap
; Kalıcı kodun bitiş adresini belirleme.
Mov     Word Ptr Es:[Bx+14],Offset End_adr ; Kalıcı kodun bitiş offset
Mov     Word Ptr Es:[Bx+16],Cs ; Kalıcı kodun bitiş segment
; Kabul edilen(default) BPB adresi
Mov     Word Ptr Es:[Bx+18],Offset BPB_PTR
Mov     Word Ptr Es:[Bx+20],Cs
Mov     Word Ptr Es:[Bx+3],DONE ; Durum sözcüğünü DONE yap
; rs232init makrosunu çağırıp ilk koşullandırma yapar
; Başlangıç ve bitiş ünite kodlarını hesaplayıp mesaj verme.
Mov     Al,Es:[Bx+22] ; ilk ünite numarası

```



```

Add      Cs:Dno1,A1                ; Bitiş ünite kodu
Add      A1,DRV_NUM
Dec      A1
Add      Cs:Dno2,A1                ; Başlangıç ünite kodu
Mov      Ah,9                      ; Karakter katarı yazma isteği
Mov      Dx,Offset Msg1            ; Ünitelere ait mesaj
Push     Cs
Pop      Ds                        ; Ds <- Cs
Int      21h                      ; DOS'u çağır
Ret
s_init   Endp

; Volume ID'nin yanlış disk değiştirme hatası oluştuğunda alınması
s_chkvol Proc Near
Mov      Ah,Es:[Bx+3]              ; Alçak anlamlı byte
And      Ah,7Fh                    ; Hata kodu
Cmp      Ah,0Fh                    ; Disk değişti hatası mı?
Jne      s_chkvol_ret              ; Değilse dön
; Volume ID'nin alınması
Push     Cs
Pop      Ds                        ; vol_id(segment)
Mov      Si,Offset vol_id          ; vol_id(offset)
Mov      Es:[Bx+22],Si
Mov      Es:[Bx+24],Cs
Call     _rsrcvn
s_chkvol_ret:
Ret
s_chkvol Endp

; Yordam : MEDIA CHECK (Cihaz Surucu)
; Komut kodu = 1
; Amaç : Cihazın değiştirilip değiştirilmediğinin kontrolu
;
; Giriş Saklayıcıları :
; Es:Bx = istek başlığı başlangıç adresi
; Ds = Cs
; Dx = RS232 port adresi
;
; istek başlığı değişen adresler
; Word Ptr Es:[Bx+3] = Dönüş durumu
; Byte Ptr Es:[Bx+14] = Dönüş (cihaz değişim kodu)
; Dönüş = -1 (cihaz değişti) ise bir önceki volume ID'ye bir gösterge

s_mchk Proc Near
Mov      Si,Bx                    ; istek başlığı(offset)
Mov      Cx,14                    ; istek başlığından 14 byte
Push     Es
Pop      Ds                        ; Ds <- Es istek başlığı(segment)
Call     _rstrnn                  ; istek başlığını gönder

Call     _rsrcvw                  ; Dönüş durumunu al
Mov      Es:[Bx+3],Ax             ; Dönüş durumunu belirle
Test     Ax,ERROR                 ; Hata var mı?

```



```

Jnz      s_mchk_ret          ; Hata varsa dön
Call     _rsrcvb             ; Dönüş byte'ı
Mov      Es:[Bx+14],Ah       ; Dönüş byte'ını belirle
Cmp      Ah,-1               ; Cihaz değişti mi?
Jne      s_mchk_ret          ; Değişmediyse dön
Mov      Si,Offset vol_id    ; Sektör sayısı = 9
Mov      Es:[Bx+15],Si       ; Volume ID(offset)
Mov      Es:[Bx+17],Cs       ; Volume ID(segment)
Push     Cs
Pop      Ds                  ; Ds <- Cs
Call     _rsrcvn             ; Dönüş durumunu belirle
Call     _rsrcvn             ; Okunan verileri al

s_mchk_ret:
Ret
s_mchk   Endp

; Yordam : BUILD BIOS PARAMETER BLOCK (Cihaz Sürücüsü)
; Komut kodu = 2
; Amaç : Cihazın değiştirilip değiştirilmediğinin kontrolu
;
; Giriş Saklayıcıları :
; Es:Bx = istek başlığı başlangıç adresi
; Ds = Cs
; Dx = RS232 port adresi

s_bbpb   Proc      Near
Mov      Si,Bx               ; istek başlığı(offset)
Mov      Cx,14               ; istek başlığından 14 byte
Push     Es
Pop      Ds                  ; Ds <- Es istek başlığı(segment)
Call     _rstrnn             ; istek başlığını gönder
Call     _rsrcvw             ; Dönüş durumunu al
Mov      Es:[Bx+3],Ax        ; Dönüş durumunu belirle
Test     Ax,ERROR            ; Hata var mı?
Jnz      s_bbpb_ret          ; Hata varsa dön
Lds      Si,Es:[Bx+14]       ; Tampon adresi
Mov      Es:[Bx+18],Si       ; BPB(offset)
Mov      Es:[Bx+20],Ds       ; BPB(segment)
Call     _rsrcvn             ; BPB al

s_bbpb_ret:
Ret
s_bbpb   Endp

; Yordam : READ (INPUT) (Cihaz Sürücüsü)
; Komut kodu = 4
; Amaç : Okuma
;
; Giriş Saklayıcıları :
; Es:Bx = istek başlığı başlangıç adresi
; Ds = Cs
; Dx = RS232 port adresi

s_read   Proc      Near
Mov      Si,Bx               ; istek başlığı(offset)

```



```

Mov     Cx,22                ; İstek başlığından 22 byte
Push    Es
Pop      Ds                  ; Ds <- Es istek başlığı(segment)
Call    _rstrnn

Mov     Cs:_sc,0             ; Sektör sayısı = 0
Lds     Si,Es:[Bx+14]        ; Transfer adresi(offset)

s_read_lp:
Call    _rsrcvw              ; Dönüş durumunu al
Mov     Es:[Bx+3],Ax         ; Dönüş durumunu belirle
Call    _rsrcvn              ; Okunan verileri al

Add     Si,Ax                ; Alınan veri sayısını ekle
Mov     Cl,9                 ; 512 byte
Shr     Ax,Cl                ; Sektör sayısını bul
Add     Cs:_sc,Ax            ; Sektör sayısını ekle
Test    Word Ptr Es:[Bx+3],ERROR ; Hata var mı?
Jnz     s_read_err          ; Sıfırdan farklı ise hata
Mov     Ax,Cs:_sc

Cmp     Ax,Es:[Bx+18]        ; Sektör sayısı doldu mu?
Je      s_read_ret          ; Esitse don
Jmp     Short s_read_lp      ; Çevrime git

s_read_err:
Call    s_chkvol

s_read_ret:
Ret

s_read Endp

; Yordam : WRITE & WRITE WITH VERIFY (Cihaz Sürücü)
; OUTPUT & OUTPUT WITH VERIFY
; Komut kodu = 8,9
; Amaç : Yazma
;
; Giriş Saklayıcıları :
; Es:Bx = istek başlığı başlangıç adresi
; Ds = Cs
; Dx = RS232 port adresi
s_write Proc Near
Mov     Si,Bx                ; İstek başlığı(offset)
Mov     Cx,22                ; İstek başlığından 22 byte
Push    Es
Pop      Ds                  ; Ds <- Es istek başlığı(segment)
Call    _rstrnn

Lds     Si,Es:[Bx+14]        ; Transfer adresi(offset)
Mov     Ax,Es:[Bx+18]        ; Sektör sayısı
Mov     Cs:_sc,Ax            ; _sc = sektör sayısı

s_write_lp:
Cmp     Cs:_sc,BUFFER

```



```

Ja      s_write_1
Mov     Di,Cs:_sc
Jmp     Short s_write_2
s_write_1:
Mov     Di,BUFFER
s_write_2:
Mov     Ax,Di
Mov     Cl,9
Shl     Ax,Cl
Push    Ax                      ; Transfer edilen byte sayısını sakla
Mov     Cx,Ax
Call    _rstrnn
Call    _rsrcvw
Mov     Es:[Bx+3],Ax
Pop     Cx                      ; Transfer edilen byte sayısını al
Test    Ax,ERROR                ; Hata var mı?
Jnz     s_write_err             ; Sıfırdan farklı ise hata
Add     Si,Cx                    ; Yazılan byte sayısını offset'e ekle
Sub     Cs:_sc,Di                ; Sektör sayısını azalt
Jz      s_write_ret
Jmp     Short s_write_lp
s_write_err:
Call    s_chkvol
s_write_ret:
Call    _rsrcvw
Mov     Es:[Bx+18],Ax
Ret
s_write Endp

Include RSTR.INC                ; RS232 veri transfer altprogramları

RD_Drvr Endp

End_adr:

Include RSINIT.INC              ; RS232 ilk koşullandırma altprogramları

Cseg  Ends
End    RD_Drvr

```



```
;
; Yazan: A. Sedat ALIŞ
; (c) Kopya Hakkı Ve Bütün Hakları Yalnızca A. Sedat ALIŞ'a aittir.
; (c) Copyright & All Rights Reserved by A. Sedat ALIŞ
```

```
Title   Interrupt Server for RS232 port
Name     IntSrvr
```

```
Include RSINIT.MAC      ; RS232 ilk koşullandırma makrosu
```

```
; Makina tipleri ile ilgili tanımlamalar :
```

```
PCXT    Equ    0      ; PC-XT veya uyumlu
PCAT    Equ    1      ; PC-AT veya uyumlu
PS2     Equ    2      ; PS/2 ailesi
```

```
; Kontrol karakterleri ile ilgili tanımlamalar
```

```
CR      Equ    0Dh    ; Carriage return
LF      Equ    0Ah    ; Line feed
ENDSTR  Equ    36     ; DOS için string sonu ($)
```

```
>1      ; Makina tipi ve port adresi
```

```
; Kesmelerle ilgili adres tanımlamaları :
```

```
REARM   Equ    02F4h  ; IRQ 4 için global Rearm adresi (PS/2)
SPC_EOI Equ    64h    ; IRQ 4 için özel EOI
                     ; (EOI - End of Interrupt - kesme sonu) (PS/2)
EOI     Equ    20h    ; Özel olmayan EOI (PS/2)
OCR     Equ    03Ch   ; Kesme kontrolörünün işlem saklayıcısının adresi (PS/2)
IMR     Equ    21h    ; Kesme maske saklayıcısının adresi (PS/2, PC)
SIC     Equ    20h    ; Sistem kesme kontrolörünün adresi (PC)
```

```
BLOCKSIZE Equ    512      ; Blok uzunluğu (byte)
BUFFERSIZE Equ    BUFFER*BLOCKSIZE ; Tampon uzunluğu (byte)
ERROR     Equ    8000h    ; Hata bitini birle
```

```
Cseg     Segment Para Public 'Code'
```

```
Org      100h          ; .COM dosyalar için
Assume   Cs:Cseg, Ds:Cseg
```

```
Start:
```

```
Jmp      Initialize    ; Başlangıç programına git
```

```
i_tbl    Label    Word
Dw        i_mchk      ; (1) Media check
Dw        i_bbpb      ; (2) Build BPB
Dw        ?           ; (3) Tanımsız Komut
Dw        i_read      ; (4) Read
Dw        3 Dup(?)    ; (5-7) Tanımsız Komut
Dw        i_write     ; (8) Write
Dw        i_write     ; (9) Write with Verify
```

```
_sc      Dw        ?           ; Değişken
DskBuf   Db        BUFFERSIZE Dup(?) ; Tampon
Req_hdr   Db        32 Dup(?) ; İstek başlığı
Req_queue Db        -1        ; İstek kodu
```


_mactype Db ? ; Makina tipi

>2 ; Strategy, Interrupt yordam göstergeleri

int_server Proc Far

Sti ; Kesmelere izin ver
Push Dx ; Kullanılan saklayıcıları koru
Push Ax

Cmp _mactype,PS2 ; Makina tipi PS/2 mi?
Je int_1 ; PS/2 ise int_1'e git
Mov Al,20h ; Kesme askısını kaldır
Out SIC,Al ; Sistem kontrolörüne koy

int_1:

Mov Dx,PORTADR ; Seri kart adresi
Add Dx,2 ; IIR (Interrupt Identification Register)
In Al,Dx ; Saklayıcıyı oku
Test Al,1 ; Kesme var mı?
Jnz int_ret ; Sıfırdan farklı ise dön.

; Kesme Hizmet Programı

Cmp Al,00000100B ; Veri hazır kesmesi mi?
Jne int_ret ; Değilse dön.
Sub Dx,2 ; RBR (Receiver Buffer Register)
Call _rsrcvb ; Komut kodunu oku
Mov Cs:Req_queue,Ah ; isteği kuyruğa at

; Kesme iznini kaldır

Inc Dx ; IER
Xor Al,Al ; Veri hazır kesme iznini kaldır
Out Dx,Al ; Kesme izin saklayıcısını belirle

int_ret:

Cmp Cs:_mactype,PS2 ; Makina tipi PS/2 mi?
Jne i_1
Cli
Mov Al,EOI ; Kesme sonu - End of Interrupt
Out OCR,Al
Mov Dx,REARM ; Rearm async interrupt
Out Dx,Al
Sti

i_1:

Pop Ax
Pop Dx
Iret

int_server Endp

int28_server Proc Far

Sti ; Kesmelere izin ver
Push Ax
Mov Al,Cs:Req_queue
Cmp Al,-1 ; Kuyruk boş mu?


```

i_chkvol Je      int28_ret      ; Eşitse dön
          Push    Bx            ; Saklayıcıları koru
          Push    Cx
          Push    Dx
          Push    Si
          Push    Di
          Push    Ds
          Push    Es
          Xor     Ah,Ah
          Dec     Al            ; init olmadığı için Al'yi 1 eksilt
          Shl     Ax,1          ; Word değerler için *2
          Mov     Bx,Ax
          Mov     Ax,Cs
          Mov     Ds,Ax
          Mov     Es,Ax

; RS232 hattından istek başlığını (Req_hdr) al
          Mov     Dx,PORTADR
          Mov     Si,Offset Req_hdr
          Call    _rsrcvn

          Call    Cs:i_tbl[Bx]   ; ilgili yordamı çağır

          Mov     Cs:Req_queue,-1 ; Komut işlendi, kuyruğu boşalt
          Cli
          Inc     Dx            ; IER
          Mov     Al,1          ; Veri hazır kesme izin ver
          Out     Dx,Al        ; Kesme izin saklayıcısını belirle
          Pop     Es
          Pop     Ds
          Pop     Di
          Pop     Si
          Pop     Dx
          Pop     Cx
          Pop     Bx

int28_ret:
          Pop     Ax
          Iret

int28_server Endp

; Eğer yanlış disk değişimi hatası oluştu ise,
; Bir önceki Volume ID'yi gönder.
i_chkvol Proc Near
          Mov     Ah,[Bx+3]      ; Alçak anlamlı byte (hata kodu)
          And     Ah,7Fh        ; Hata kodu
          Cmp     Ah,0Fh        ; Disk değişti hatası mı?
          Jne     i_chkvol_ret   ; Değilse dön
; Volume ID'nin gönderilmesi
          Lds     Si,Es:[Bx+22]  ; Volume ID göstergesi
          Mov     Cx,12          ; Volume ID uzunluğu
          Call    _rstrnn

i_chkvol_ret:
          Ret

```


i_chkvol Endp

```
; Yordam : MEDIA CHECK (Kesme Yordamı)
; Komut kodu = 1
; Amaç : Cihazın değiştirilip değiştirilmediğinin kontrolu
; Giriş Saklayıcıları :
; Ds:Si = istek başlığı
; Ds, Es = Cs
i_mchk Proc Near
    Mov     Bx, Si                      ; istek başlığının(offset)
    Call    Strategy                    ; Cihaz strategy yordamını çağır
    Call    Interrupt                   ; Cihaz interrupt yordamını çağır

    Mov     Ax, Es:[Bx+3]               ; Dönüş durumu
    Call    _rstrnw                     ; Dönüş durumunu gönder
    Test    Word Ptr Es:[Bx+3], ERROR ; Hata var mı?
    Jnz     i_mchk_ret                 ; Hata varsa dön
    Mov     Ah, Es:[Bx+14]              ; Dönüş byte'ı
    Call    _rstrnb                     ; Dönüş byte'ını gönder

    Cmp     Ah, -1                      ; Cihaz değişti mi?
    Jne     i_mchk_ret                 ; Değişmediyse dön
    Lds     Si, Es:[Bx+15]              ; Volume ID göstergesi
    Mov     Cx, 12                      ; Vol_ID (12 byte)
    Call    _rstrnn
```

```
i_mchk_ret:
    Ret
i_mchk Endp
```

```
; Yordam : BUILD BIOS PARAMETER BLOCK (Kesme Yordamı)
; Komut kodu = 2
; Amaç : BIOS Parametre Bloğunu Oluşturma
; Giriş Saklayıcıları :
; Ds:Si = istek başlığı
; Ds, Es = Cs
i_bbpb Proc Near
    Mov     [Bx+14], Offset DskBuf      ; Transfer adresi(offset)
    Mov     [Bx+16], Cs                 ; Transfer adresi(segment)
    Mov     Bx, Si                      ; istek başlığı(offset) Es(segment)
    Call    Strategy                    ; Cihaz strategy yordamını çağır
    Call    Interrupt                   ; Cihaz interrupt yordamını çağır

    Mov     Ax, [Bx+3]                  ; Dönüş durumu
    Call    _rstrnw                     ; Dönüş durumunu gönder
    Test    Word Ptr [Bx+3], ERROR      ; Hata var mı?
    Jnz     i_bbpb_ret                 ; Hata varsa dön
    Lds     Si, Es:[Bx+18]              ; BIOS Parametre Bloğu adresi
    Mov     Cx, 32                      ; BPB uzunluğu 32 byte
    Call    _rstrnn                     ; BPB gönder
```

```
i_bbpb_ret:
```



```

Ret
i_bbpb Endp

; Yordam : READ (INPUT) (Kesme Yordamı)
; Komut kodu = 4
; Amaç : Okuma
; Giriş Saklayıcıları :
; Ds:Si = istek başlığı
; Ds, Es = Cs
i_read Proc Near
    Mov     Bx, Si                ; istek başlığı
    Mov     Ax, [Bx+18]          ; Sektör sayısı
    Mov     _sc, Ax              ; Sakla
    Mov     Si, Offset DskBuf    ; Disk tampon adresi
    Mov     [Bx+14], Si          ; transfer adresi(offset)
    Mov     [Bx+16], Cs          ; transfer adresi(segment)

i_read_lp:
    Cmp     _sc, BUFFER          ; Sektör sayısının BUFFER'a göre durumu?
    Ja      i_read_1             ; (sektör sayısı > buffer) ise -> i_read
    Mov     Di, _sc              ; sektör sayısı <- BUFFER
    Jmp     Short i_read_2

i_read_1:
    Mov     Di, BUFFER           ; sektör sayısı > BUFFER

i_read_2:
    Mov     [Bx+18], Di          ; sektör sayısı
    Call    Strategy
    Call    Interrupt

    Mov     Ax, [Bx+3]           ; Durum sözcüğü
    Call    _rstrnw              ; Durum sözcüğünü gönder
    Mov     Ax, [Bx+18]          ; Okunan sektör sayısı
    Mov     Cl, 9                ; 512 için shift sayısı
    Shl     Ax, Cl               ; toplam byte=sektör sayısı*512
    Mov     Cx, Ax               ; Gonderilecek byte sayısı
    Call    _rstrnn              ; okunan verileri transfer et
    Test    Word Ptr [Bx+3], ERROR ; Hata var mı?
    Jnz     i_read_err           ; Sıfırdan farklı ise hata

    Add     [Bx+20], Di          ; Başlangıç sektör sayısını
                                   ; okunan sektör sayısına ekle
    Sub     _sc, Di              ; sektör sayısını azalt
    Jz      i_read_ret           ; Sıfır ise dön
    Jmp     Short i_read_lp

i_read_err:
    Call    i_chkvol

i_read_ret:
    Ret

i_read Endp

; Yordam : WRITE & WRITE WITH VERIFY (Kesme Yordamı)
; OUTPUT & OUTPUT WITH VERIFY

```



```

;      Komut kodu = 8,9
;      Amaç : Yazma
;      Giriş Saklayıcıları :
;      Ds:Si = istek başlığı
;      Ds, Es = Cs
i_write Proc      Near
    Mov     Bx, Si                ; istek başlığı
    Mov     _sc, 0                ; Sektör sayısı(yazılmış) = 0
    Mov     Di, [Bx+18]          ; Sektör sayısı
    Mov     Si, Offset DskBuf    ; Disk tamponu
    Mov     [Bx+14], Si          ; Transfer adresi(offset)
    Mov     [Bx+16], Cs          ; Transfer adresi(segment)

i_write_lp:
    Call     _rsrcvn
    Mov     Cl, 9                ; 512 için shift sayısı
    Shr     Ax, Cl               ; Sektör sayısını bul
    Mov     [Bx+18], Ax          ; Yazılacak sektör sayısı
    Mov     Cx, Ax
    Call     strategy
    Call     interrupt
    Add     _sc, Cx
    Mov     Ax, [Bx+3]           ; Dönüş durumu
    Call     _rstrnw             ; Dönüş durumunu gönder
    Test    Word Ptr[Bx+3], ERROR ; Hata var mı?
    Jnz     i_write_err          ; Sıfırdan farklı ise hata
    Add     [Bx+20], Cx          ; Başlangıç sektör no. arttır
    Sub     Di, Cx
    Jz      i_write_ret
    Jmp     Short i_write_lp

i_write_err:
    Call     i_chkvol
i_write_ret:
    Mov     Ax, _sc              ; Sektör sayısı
    Call     _rstrnw             ; Sektör sayısını gönder
    Ret
i_write Endp

    Include RSTR.INC             ; RS232 veri transferi altprogramları

Initialize Proc      Near

>3      ; rs232init makrosunu çağırıp ilk koşullandırma yapar

;      Veri hazır kesmesini kur.
    Mov     Ah, MACTYPE          ; Makina tipi
    Mov     Al, CARDNO          ; Seri kart no
    Mov     Dx, PORTADR          ; Port adresi
    Mov     Bx, Offset int_server ; Kesme alt programı (Ds = Cs)
    Call     init_irq
    Jc      i_err                ; Elde varsa hata oluştu

;      Kesme no=28h'in kesme adresini belirleme

```



```

Mov     Ah,25h                ; Kesme vektörü belirleme isteği
Mov     Al,28h                ; Kesme vektör no
Mov     Dx,Offset int28_server ; Kesme adresi
Int     21h                   ; DOS'u çağır

Mov     Ah,09h                ; Karakter katarı yazma isteği
Mov     Dx,Offset _msg2       ; Mesajın adresi
Int     21h                   ; DOS'u çağır

; Dur ve kalıcı ol.
Mov     Dx,Offset Initialize   ; Kodun son adresi
Int     27h

i_err:
Mov     Ah,09h                ; Karakter katarı yazma isteği
Mov     Dx,Offset _msg1       ; Mesajın adresi
Int     21h                   ; DOS'u çağır
Ret

Initialize Endp

_msg1 Db 'Error occured! ',CR,LF,ENDSTR ; Hata mesajı
_msg2 Db 'Device Driver Service Routines Are ' ; Duruş mesajı
      Db 'Succesfully Installed. ',CR,LF,ENDSTR

; Kesme kontrol eden altprogramın giriş parametreleri:
; Giriş saklayıcıları :
; AH = Makina tipi
; AL = Kart no. (0-3) 0=COM1, 1=COM2, 2=COM3, 3=COM4
; DX = Port adresi
; DS:BX = Kesme oluştuğunda çağırılacak kesme yordamı
init_irq Proc Near
Mov     Cs:_mactype,Ah ; Makina tipini sakla
Mov     Cl,11101111B   ; Kesme düzeyi 4 (IRQ 4)
Mov     Ch,0Bh         ; Kesme no.
Cmp     Ah,PCXT        ; PC-XT veya uyumlu mu?
Jne     I1
; PC-XT veya uyumlu
Test    Al,1           ; Kart no çift mi?
Jz      Ipc
Inc     Ch             ; Kesme no = 0Ch, COM2 ve COM4
Ror     Cl,1           ; Kesme düzeyi 3 (IRQ 3)
Jmp     Short Ipc

I1:
Cmp     Ah,PCAT        ; PC-AT veya uyumlu mu?
Jne     I2
; PC-AT veya uyumlu
Ror     Cl,1           ; Kesme düzeyi 3 (IRQ 3)
Test    Al,1           ; Kart no tek mi?
Jnz     Ipc
Inc     Ch             ; Kesme no = 0Ch, COM1 ve COM3
Rol     Cl,1           ; Kesme düzeyi 4 (IRQ 4)
Jmp     Short Ipc

```



```

I2:      Cmp      Ah,PS2      ; PS/2 ailesi mi?
        Jne      init_err    ; Eşit değilse hata!
;        PS/2 ailesi
        Test     Al,1        ; Kart no tek mi?
        Jnz      Ips2
        Inc      Ch          ; Kesme no = 0Ch, COM1 ve COM3
Ips2:    Cli
        Mov      Al,SPC_EOI   ; 4. düzey kesmelerin yürütülmesi için
        Out      OCR,Al       ; Özel kesme sonu (EOI) gönder.
Ipc:     Cli
        In       Al,IMR       ; Kesme maske saklayıcısını oku
        And      Al,C1        ; Kesme düzey 3:4 (IRQ 3:4) maskesini kaldır
        Out      IMR,Al       ; Yeni maskeyi yaz.

        Add      Dx,4         ; MCR (Modem Control Register)
        Mov      Al,00001000B ; Out2=1 (kesme izni)
        Out      Dx,Al        ; MCR belirle
        Sub      Dx,3         ; IER (Interrupt Enable Register)
        Mov      Al,00000001B ; Alınmış veri hazır kesmesine izin ver
        Out      Dx,Al        ; IER belirle
        Sti

;        Kesmeyi belirleme
        Mov      Ah,25h       ; Kesme vektörü belirleme isteği
        Mov      Al,Ch        ; Kesme vektör no
        Mov      Dx,Bx        ; Kesme adresi
        Int      21h          ; DOS'u çağır
        Clc                 ; Eldeyi sıfırla (hata yok)
        Ret

init_err:
        Stc                 ; Eldeyi birle (hata var!)
        Ret
init_irq      Endp

        Include  RSINIT.INC   ; RS232 ilk koşullandırma altprogramları

Cseg      Ends
        End      Start

```



```
MASM INTSRVR.TMP;  
LINK INTSRVR;  
EXE2BIN INTSRVR.EXE INTSRVR.COM  
ERASE INTSRVR.EXE  
ERASE INTSRVR.OBJ  
MASM RDDRVR.TMP;  
LINK RDDRVR;  
EXE2BIN RDDRVR.EXE RDDRVR.SYS  
ERASE RDDRVR.EXE  
ERASE RDDRVR.OBJ
```

3) IBM PC-XT Model 30 Technical Reference

4) IBM Programmer's Reference

Paul A. Bettmann, Inc. Corporation

ISBN 0-1999-337-2

5) Memory Resident Programming On The IBM PC

Harper & Row, Addison-Wesley Publishing Company, Inc.

ISBN 0-201-10091-3

6) EARN (European Academic Research Network)

PC File Servers (LISTSERVONDSUVM, UNICOMPCOM)

KAYNAKLAR

- 1) Disk Operating System Technical Reference
1983 yılında
- 2) IBM PC-AT Technical Reference
Bilgisayar Bilimleri
1987 yılında yüksek öğrenimini
- 3) IBM PS/2 Model 30 Technical Reference
Mühendisliği
Yıldırım Üniversitesi Bilgisayar Bilimleri Mühendisliği
- 4) DOS Programmer's Reference
Terry R. Dettmann, Que Corporation
ISBN 0-88022-327-8
- 5) Memory Resident Programming On The IBM PC
Thomas A. Wadlow, Addison-Wesley Publishing Company, Inc.
ISBN 0-201-18595-2
- 6) EARN (European Academic Research Network)
PC File Servers. (LISTSERV@NDSUVM1, TRICKLE@TREARN)

ÖZGEÇMİŞ

A. Sedat Alış, 1965 yılında Çanakkale'de doğdu. Orta öğrenimini Kuleli Askeri Lisesinde tamamladı. 1983 yılında ÖSYS ile Yıldız Üniversitesi, Bilgisayar Bilimleri Mühendisliği bölümüne girdi. 1987 yılında yüksek öğrenimini Mühendislik Fakültesi birincisi olarak bitirdi. Halen, Yıldız Üniversitesi Bilgisayar Bilimleri Mühendisliği Bölümünde yüksek lisans eğitimini sürdürmektedir ve aynı bölümde Araştırma Görevlisi olarak çalışmaktadır.

