

جزوه ی

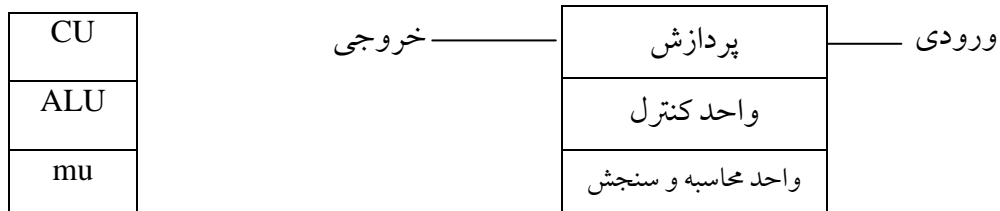
برنامه سازی سیستم

استاد خوشگفتار

۱۳۹۲

برنامه نویسی سیستم

برنامه سازی سیستم: نوشتن به زبان ماشین و سطح پایین مربوط به سخت افزار (نوشتن به زبان ++C).



S.P.K = Speaker

K.B = key board

H.D.D = Hard Disk Drive

M.B = Main board

V.G.A = Video Graphic Adaptor

C.P.U = Central Prosing Unit

F.D = Floppy Disk

F.D.D = Driver

C.D = حافظه

D.V.D = Digital Video Disk

D.V.D.D = Driver

سخت افزار با نقطه اصطلاح است و بی نقطه قطعه است.

انواع حافظه:

از لحاظ مدت نگهداری اطلاعات: (دائمی، موقتی)

➤ دائمی: D.V.D – CD - ROM – Flash – H.D.D

➤ موقتی: RAM

از لحاظ قابلیت دسترسی: (تصادفی، ترتیبی)

➤ تصادفی: هارد، DVD – CD – Flash - RAM

➤ ترتیبی: نوار مغناطیسی، ROM

از لحاظ نزدیکی به مادربرد: (اصلی، فرعی)

➤ اصلی: ثبات، RAM - ROM

➤ فرعی: DVD - CD - Flash

از لحاظ خواندن و نوشتن: (فقط خواندنی، خواندنی - نوشتنی)

➤ فقط خواندنی: DVD - CD - ROM

➤ خواندنی - نوشتنی: هارد، فلش، نوار، RAM

BIOS = Basic Input Output System

واسطی بین سخت افزار و سطوح نرم افزاری بالاتر

نخستین نرم افزاری است که پس از روشن شدن سیستم به اجرا در می آید تا به پردازنده بگوید چه کاری باید انجام دهد، و هدف از اجرا شدن آن این است که به سیستم اعلام کند که در خدمت است. این نرم افزار درون حافظه قرار دارد.

وظیفه اصلی بایاس بارگذاری سیستم عامل است.

Application	برنامه های کاربردی
Operating System	
BIOS	
ROM - RAM - CPU - ...	سخت افزار

بایاس چیست؟

الف) نرم افزاری که سیستم عامل را بارگذاری میکند. ب) يك نرم افزار سیستمی است. ✓

ج) يك حافظه است. د) برنامه ای که به پردازنده میگوید چه اعمالی باید انجام دهد.

تمام تنظیمات اولیه بایاس بر روی یک حافظه غیر فرار به نام cmos ذخیره میشود که با یک باتری پشتیبانی میشود. حافظه cmos بدون باتری کنارش فرار است.

مراحل بارگذاری سیستم عامل توسط بایاس:

- 1 - بررسی تنظیمات cmos (تنظیمات اولیه cmos توسط بایاس خوانده شده و به روز میشود).
- 2 - بررسی کارت گرافیک: در صورتی که خود کارت گرافیک بایاس نداشته باشد از درایو استاندارد ذخیره شده در ROM استفاده و آن را راه اندازی میکند.
- 3 - در صورتی که نوع راه اندازی کامپیوتر cold boot باشد صحت عملکرد حافظه RAM انجام میشود، و سپس پورت های سریال و usb برای اتصال صفحه کلید و ماوس بررسی میشود و اگر خطایی وجود داشته باشد با نواختن چند beep معنادار خطا را اعلام میکند سپس کارت های PCI نصب شده بر روی سیستم بررسی میشود.
- همچنین اطلاعاتی را درباره نوع پردازنده، فلاپی درایو، هارد دیسک، حافظه، تاریخ و ورژن خود بایاس، نوع صفحه نمایشگر را نشان میدهد.

تست) در کدام مرحله از مراحل عملیات آماده سازی بایاس حافظه بررسی میشود؟

الف) cold boot ✓ ب) Reboot ج) بررسی کارت گرافیک د) به روز کردن ادیتور بایاس

تست 2) بایاس در مورد خرابی کدام گزینه beep معنادار تولید میکند؟

الف) فلاپی ب) هارد دیسک ج) کارت گرافیک ✓ د) صفحه نمایش

4 - فراهم سازی یک سری وقفه ها و روتین های سطح پایین (زبان ماشین، یا زبان اسمبلی)

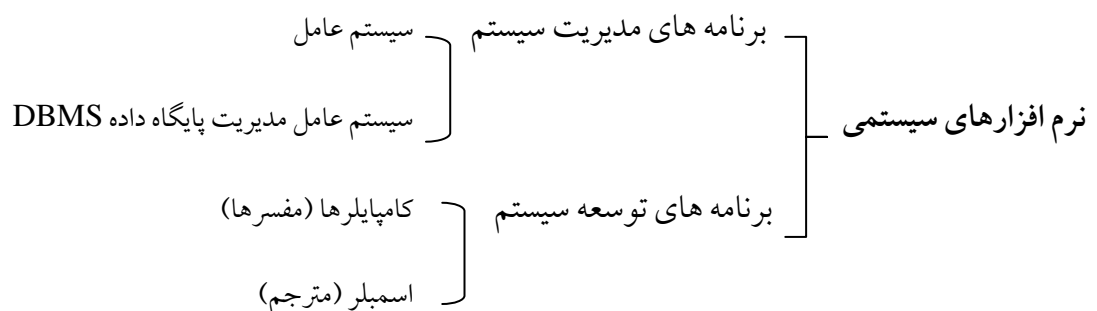
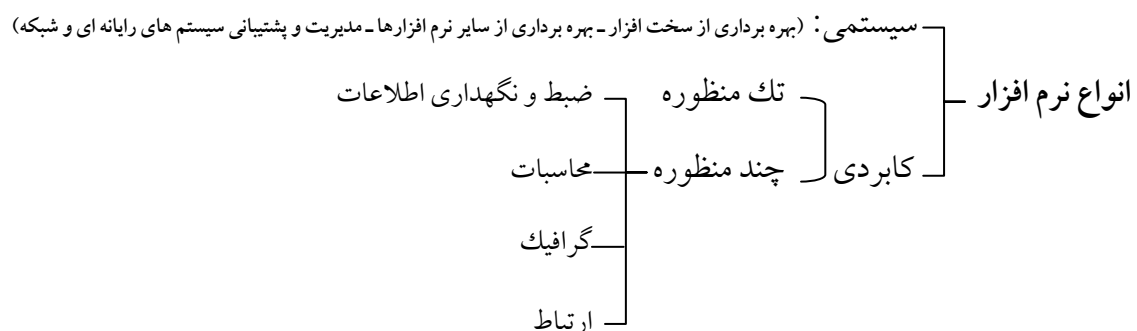
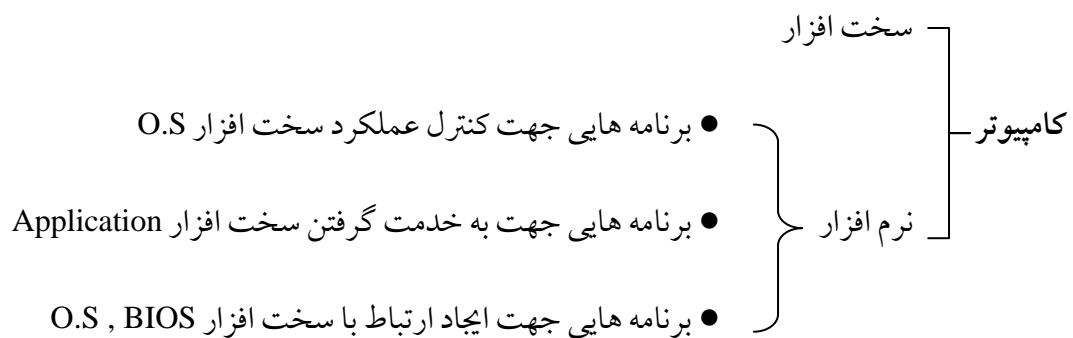
سیستم عامل توانایی برقراری ارتباط و تعامل با قطعات سخت افزاری مختلف را داشته باشد.

5 - نوع درایوی را که باید سیستم عامل از آن آغاز شود را تشخیص میدهد، در صورتی سیستم عامل پیدا نشود پیغام خطا میدهد (منظور بایاس است). تشخیص اینکه سیستم عامل باید از کدام درایو بارگذاری شود از پیکربندی تنظیمات تبعیت میکند.

✓ $10^9 * 10$ دستور العمل در يك ثانيه توسط پردازنده انجام میشود.

✓ **Chipset**: اطلاعات متنوعی را درباره ارتباط با توانایی ماذربورد مشخص میکند.

سرعت ها را بین اجزای ماذربورد هماهنگ میکند و يك سرعت یکسان برقرار میکند، سرعت آن از بایاس کمتر است.

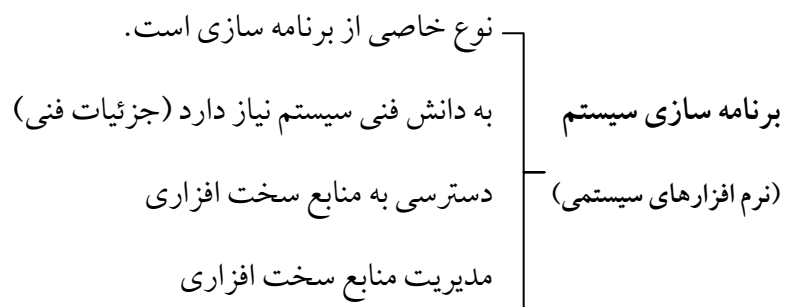


ایجاد و حذف پردازش های کاربر	پردازنده	سیستم عامل (مهمترین نرم افزار، مدیریت منابع سیستم را به عهده دارد)
زمانبندی پردازش		
مدیریت همزمانی پردازش ها و ارتباط بین آنها		
جلوگیری از بن بست		
هر بخش از حافظه توسط چه پردازشی مورد استفاده قرار میگیرد	حافظه اصلی و ثانویه	
تخصیص و بازپس گیری حافظه		
مدیریت هارد دیست		
مدیریت حافظه مجازی		
ایجاد و حذف فایل و پوشه	فایل	
انجام عملیات کپی - انتقال و تغییرات بر روی فایل ها		
ذخیره سازی و مدیریت قرار گرفتن فایل ها روی رسانه		
مدیریت دسترسی به فایل های مشترک		
مدیریت بافرها	I/O	
اجرای درایور I/O		
جلوگیری از تداخل وسایل I/O		
اداره بن بست ها		

✓ شکاف های روی مادربرد از لحاظ سرعت:

1 - ISA - 2 EISA - 3 مودم (شبکه) PCI - 4 - گرافیک 5 - PCIE

مبانی برنامه سازی سیستم System Programming Basics



ویژگی های برنامه نویسی (برنامه سازی کاربردی):

- روش ورودی خروجی اطلاعات وابسته به سیستم
- در قالب برنامه های کامپیوتری
- ارائه اطلاعات
- قابل تعریف برای اغلب کامپیوترها
- که در نهایت منجر به نرم افزارهای کاربردی میشود.

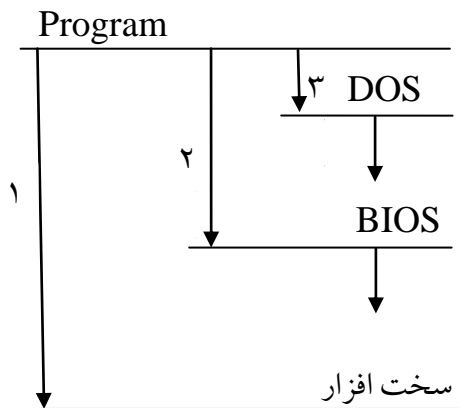
✓ برنامه سازی سیستم: کنترل سخت افزارهای ورودی - خروجی

✓ برنامه های کاربردی: برای پردازش اطلاعات ورودی - خروجی

سخت افزار به خدمت گرفته میشود توسط برنامه سازی کاربردی و سیستمی.

ارتباط 2 و 3 هزینه کمتری و زمان کمتری برای نوشتن برنامه صرف میکند.

برنامه هایی که از طریق 2 و 3 نوشته میشوند به سخت افزار وابسته نیستند.



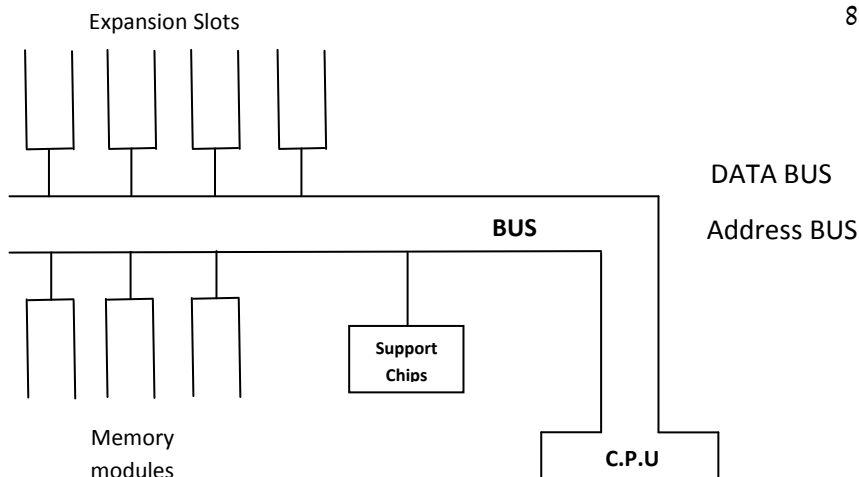
DOS سخت افزار را تب میبیند.

BIOS سخت افزار را تب میبیند.

ارتباط Program با سخت افزار از این سه راه امکان پذیر است:

✓ DOS، BIOS، مستقیم.

پردازنده از سری 80 * 86



پردازنده از طریق گذرگاه اطلاعات را از حافظه خوانده و در آن مینویسد.

تراشه های پشتیبانی: 1- کنترل بخشی از سخت افزار، 2- انجام برخی کارها (کم کردن بار cpu).

DMA controller: نوشتن مستقیم اطلاعات از دیسک سخت بر روی حافظه و ایجاد ارتباط با لوازم

جانبی (تسریع در انتقال اطلاعات، آزاد ماندن پردازنده).

Interrupt controller (مدیریت وقفه هاست): (وقفه) جهت آگاه کردن پردازنده از آمادگی device ها

برای نقل و انتقال.

Programmable Peripheral: جهت اتصال پردازنده به لوازم جانبی K.B، spk و ...

The Clock: قلب يك سیستم کامپیوتری با ضربان چند میلیون بار در ثانیه (233 MHz). هماهنگی بین

تمامی اجزای کامپیوتر.

The Timer: برای شمارش زمان با خروجی هایی با تناوب قابل برنامه ریزی.

شکاف های توسعه: محل قرار گرفتن برخی از تراشه های دیگر مورد نیاز سیستم از جمله:

CRT Controller: در کارت گرافیکی قرار دارد و نحوه نمایش اطلاعات و کدبندی آن ها را در صفحه

نمایش مدیریت میکند.

Disk Controller: مدیریت گردانه های دیسک (حرکت هد).

مدیریت حافظه توسط سیستم عامل DOS تا 64KB است.

قطعات حافظه 64KB است.

ثبات های پردازنده: پارامتر برنامه ها از طریق ثبات ها به توابع واسطه های DOS یا BIOS ارسال میشود و

نتایج وضعیت را دریافت میکند.

ثبات های عمومی: پرکارترین آنها

BP (Base Pointer), SP (Stack Pointer), SI (Source Index), DI (Destination Index), DX, CX, BX, AX

جهت فراخوانی توابع DOS و BIOS استفاده میشود و با اعمال ریاضی دستکاری میشوند.

ثبات های سگمنت:

Data Segment: DS, Code Segment: CS

Extra Segment: ES, Stack Segment: SS

IP (Program Counter) شمارنده برنامه:

ثبات فلگ (پرچم)

				O	D	I	T	S	Z		A		P		C
--	--	--	--	---	---	---	---	---	---	--	---	--	---	--	---

ثبات فلگ: برای برقراری ارتباط بین دستورات متوالی اسمبلی به کار میرود.

وضعیت عملیات ریاضی و منطقی را نگه داری میکند.

9 بیت از 16 بیت مورد استفاده قرار میگیرد.

درگاه (Port): واسطی بین پردازنده و سایر اجزاء سخت افزاری در واقع ثبات هایی هستند 8 بیتی که قابل آدرس دهی به 65536 حالت است.

پردازنده از گذرگاه برای دسترسی به درگاه ها استفاده میکند (در زبان اسمبلی برای دسترسی به درگاه ها از دستورات IN و OUT استفاده میشود).

وقفه ها: جهت کنترل سخت افزار (سخت افزاری) و ایجاد ارتباط برنامه با DOS یا BIOS (نرم افزاری) از NMI یا Trap وقفه هایی هستند که قابل disable شدن نیستند استفاده میشود.

محاورة سیستم (System Interaction):

برای فشار دادن یک کلید بر روی صفحه کلید، موارد زیر مطرح است:

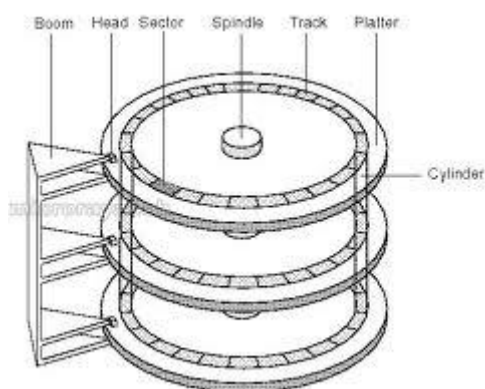
1 - سخت افزار صفحه کلید: وقفه 09H به پردازنده ارسال میشود و پردازنده با توجه به اولویت روال مربوطه را اجرا میکند.

2 - روال مدیریت صفحه کلید: BIOS کد مربوطه را به دست آورده و معتبر بودن آن را بررسی میکند (کد اسکی).

3 - بافر صفحه کلید: قرار دادن کد در بافر 16 بیتی در RAM.

4 - وقفه صفحه کلید: وظیفه ی خواندن از صفحه کلید و آماده سازی آن برای برنامه (16 H).

نکته: در سطح DOS وقفه 21 H همان کار وقفه 16 H را انجام میدهد با این تفاوت که کاراکتر زده شده نمایش میابد.



تراکم ناحیه ای: تعداد بیتی که در یک اینچ مربع میتوان نوشت (بر حسب گیگا بایت در ثانیه).

واحد آن گیکابایت در اینچ، هر چه بالاتر باشد هار سریعتر است.

Seek Time (زمان جستجو): زمان جلو و عقب رفتن بازو روی صفحه پلاتر.

سرعت چرخش محور (RPM دور در دقیقه).

نرخ انتقال مستمر اطلاعات: هارد درایو با چه سرعتی میتواند یک فایل پیوسته را سرویس دهی کند.

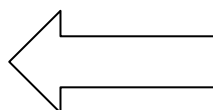
رابطه: جهت انتقال از هارد یا به هارد.

زبان اسمبلی اصلی ترین و بی محدودیت ترین راه برنامه سازی سیستم است.

✓ وقفه های BIOS در ROM و وقفه های DOS در RAM.

✓ فراخوانی وقفه ها:

آدرس روال	00



جدولی داریم به نام جدول بردار وقفه

شماره وقفه **int**

زمانی که وقفه فراخوانی میشود دو ثبات IF و TF تحت تأثیر قرار میگیرند و 0 میشوند.

روال های مدیریت وقفه: شبیه یک روال معمولی در زبان اسمبلی.

پس از رخ دادن وقفه: بعد از فراخوانی دستور INT:

1 - مقدار ثبات پرچم در پشته push میشود.

2 - سپس مقادیر IF و TF صفر میشود، IF از وقوع وقفه های دیگر جلوگیری میکند. $TF = 0$ غیرفعال کردن حالت single step.

3 - يك far call (صدا زدن از راه دور) به روال مدیریت وقفه انجام میشود.

دستور IF Cli را صفر میکند و IF Sti را يك میکند (ناتوان و تواناسازی وقفه ها).

دستورات دسترسی به درگاه ها: (شماره گروه های بزرگتر از FFH باید ابتدا شماره در MOV,DX شود)

in al port number

out port number al

in ax port number

out port number ax

in al dx

out dx ax

in ax dx

out dx al

$$\text{in ax, 07 cH} = \begin{cases} \text{mov dx, 07cH} \\ \text{out 0405H, al} \\ \text{in ax, dx} \end{cases}$$
 این دو با هم فرقی ندارند

تعیین نگارش BIOS: 256 وقفه در وجود دارد که به توابع متمم شده اند.

_d F000: FFF0

10 H video card Access

16 H keyboard functions

تعیین نوع PC:

نوع PC در آخرین بایت در نشانی F000:FFFE قرار دارد.

BIOS در 256 بایت حافظه ی RAM که به آن محدوده متغیر بایاس گفته میشود، متغیرهای داخلی خود را

ذخیره میکند.

CODE نشانی کامل از تلفیق نشانی مبدأ با نشانی قطعه 0040H به دست می آید.

FCH AT

FEH XT

FBH XT

FFH PC

کارت گرافیک: این کارت ها اطلاعات تولید شده توسط کامپیوتر را اخذ و آنها را به گونه ای تبدیل

میکند که برای انسان قابل مشاهد باشد. برخی از این کارت ها اطلاعات دیجیتال دریافتی را به آنالوگ و برخی

دیگر به دیجیتال تبدیل میکند.

✓ تصویر تشکیل شده از نقاطی به نام پیکسل است که هر پیکسل دارای يك رنگ است.

کارت های گرافیک اولیه فقط دو پیکسل سیاه و سفید داشتند، در نتیجه برای ذخیره هر پیکسل به يك بیت

نیاز داریم.

سؤال) اگر هر پیکسل در تصویر تولید شده توسط يك کارت گرافيك 256 رنگ باشد، حافظه مورد نیاز برای تولید يك تصویر 480 * 640 چقدر است؟

$$640 * 480 * 1 \text{ Byte} = 307200 \text{ Byte} \quad (\text{جواب})$$

$$640 * 480 * 1 \text{ Byte} = 307200 \text{ bit} = 38400 \text{ Byte}$$

کارت گرافيك چیست؟ يك مدار چاپی به همراه حافظه و پردازنده اختصاصی.

اسامی دیگر کارت گرافيك: کارت ويدئو - برد ويدئو - برد نمایش ويدئویی - برد گرافيك - آداپتور گرافيك - آداپتور ويدئو.

اجزای اصلی کارت گرافيك:

حافظه: اولین چیزی که هر کارت گرافيك به آن نیاز دارد و برای نگهداری رنگ پیکسل های تولید شده استفاده میشوند.

اینترفیس کامپیوتر: برای خواندن و نوشتن در حافظه کارت گرافيك با اتصال کارت به گذرگاه هر جمله به روی برد اصلی تعریف میشود.

اینترفیس ويدئو: روش به منظور تولید سیگنال برای مانیتور.

بدون پردازنده (Frame buffer): فقط میتواند يك فریم اطلاعات برای مانیتور ارسال کنند.

انواع کارت گرافيك { شتاب دهنده گرافیکی
دارای پردازنده (CPU): کمک پردازنده گرافیکی

شتاب دهنده: در سیستم های شتاب دهنده ی گرافیکی، درایور کارت گرافيك، هر چیز را در ابتدا برای پردازنده اصلی کامپیوتر ارسال میکند، در ادامه پردازنده اصلی شتاب دهنده کارت گرافيك را به منظور انجام عملیات خاص هدایت میکند. در این سیستم کمک پردازنده گرافیکی درایور و کارت گرافيك عملیات مربوطه به کارت های گرافيك را مستقیماً به پردازنده گرافیکی ارسال میکند.

عناصر دیگر بر روی کارت گرافیک:

Graphic BIOS: اطلاعات موجود در این تراشه به سایر عناصر کارت نحوه انجام عملیات را تبیین میکند. مسئولیت تست کارت گرافیک، حافظه مربوطه و عملیات ورودی - خروجی.

(DAC) Digital – to – Analog conventory: این تبدیل کننده داده های تبدیل شده در حافظه کارت گرافیک را مستقیماً از حافظه دریافت و به آنالوگ تبدیل میکند.

Display Connector: خروجی ویدئو توسط این قطعه به مانیتور منتقل میشود.

Computer (BUS) connectivy: یک گذرگاه است. این گذرگاه امکان دستیابی مستقیم به کارت گرافیک به حافظه اصلی را فراهم میکند.

مهمترین گونه های کارت گرافیک:

MDA: این کارت فقط مُد متن را پشتیبانی نمیکند، تک رنگ است، در قالب 80 * 25 کاراکتر میتواند نمایش دهد حافظه آن 4 کیلوبایت است و تنها یک صفحه را میتواند در RAM خود نگه داری کند.

CGA: اولین کارت رنگی که میتوانست از تلویزیون به جای صفحه نمایش استفاده کند، این کارت توانایی تولید خروجی RGB دارد و در قالب 80 * 25 در حالت متن و 200 * 320 در حالت گرافیک تولید تصویر میکند و حافظه آن 16 کیلوبایت است.

HGC: از هر جهت شبیه MDA است و توانایی مد گرافیک را دارد و میتواند تصویر با وضوح 348 * 720 را تولید کند.

EGA: وضوح 350 * 640 با 16 رنگ و 64 سری رنگ، حافظه 256 کیلو بایت. این کارت به وسیله ROM BIOS پشتیبانی نمیشود، به وسیله ی EGA – ROM - BIOS پشتیبانی میشود.

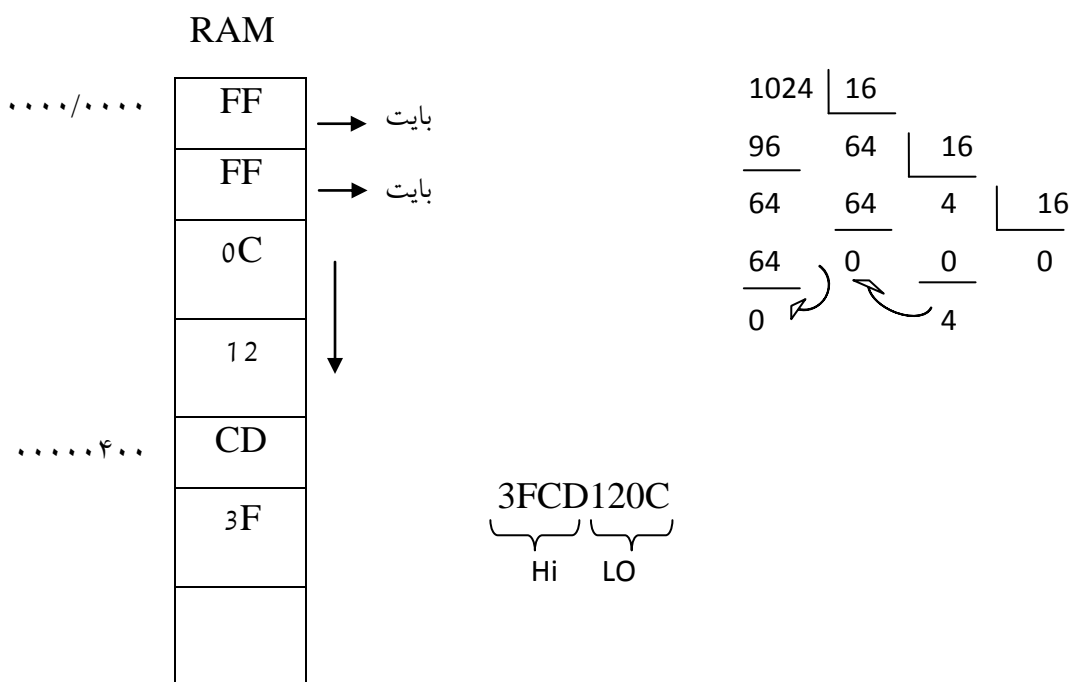
VGA: بر خلاف EGA سیگنال های رنگ را به صورت آنالوگ به صفحه نمایش ارسال میکند. بیش از 260/000 رنگ تولید میکند، حافظه 256 کیلو بایت است که تا 512 کیلو بایت قابل افزایش است.

وقفه: مکانیسمی برای مطلع کردن پردازنده جهت رسیدگی به دستگاه جانبی یا برنامه نرم افزاری است (و یا توقف کار خود پردازنده توسط خودش).

256 وقفه وجود دارد که از 0 تا 31 مخصوص خود پردازنده است (00H تا FH)

از وقفه 20H تا 3FH وقفه های مخصوص سیستم عامل DOS است.

جدول بردار وقفه حاوی آدرس (4 بیتی) روتین وقفه میباشد که این جدول در 1024 بایت اول حافظه یعنی 0000 تا 3FFF میباشد.



هر آدرس در جدول روتین 32 بیتی است (4 بیتی).

وقفه از طریق دستور العمل های وقفه در برنامه یا توسط تجهیزات خارجی در سیستم فعال میشود.

INT 01 H

هنگامی که وقفه برای پردازنده اتفاق می افتد شماره آن در 4 ضرب شده تا آدرس شروع روتین وقفه در جدول بردار وقفه مشخص شود. سپس آدرس را در CS و IP قرار میدهیم و شروع به اجرای دستور العمل ها میکنیم.

مثال) اگر شماره وقفه 4AH باشد آدرس روتین وقفه در جدول بردار وقفه برابر است با:

$$4AH * 4 = 128H$$

$$4H = 4 * 16^1 + A * 16^0 = 64 + 10 = 74$$

$$74 * 4 = 296$$

در واقع آدرس شروع 4H را پیدا کردیم

۲۸۶	16	
۱۶	۱۸	16
۱۳۶	۱۶	1
۱۲۸	۲	0
۰۰۸	۱	0

از تمام وقفه های DOS فقط وقفه 21H دارای روتین هایی برای برقراری ارتباط با کیبورد - کارت گرافیک - پرینتر و کار با هارد دیسک و وسایل ارتباطی غیر همزمان هست.

عملیاتی که هنگام فراخوانی دستور العمل INT انجام میشود به قرار زیر است:

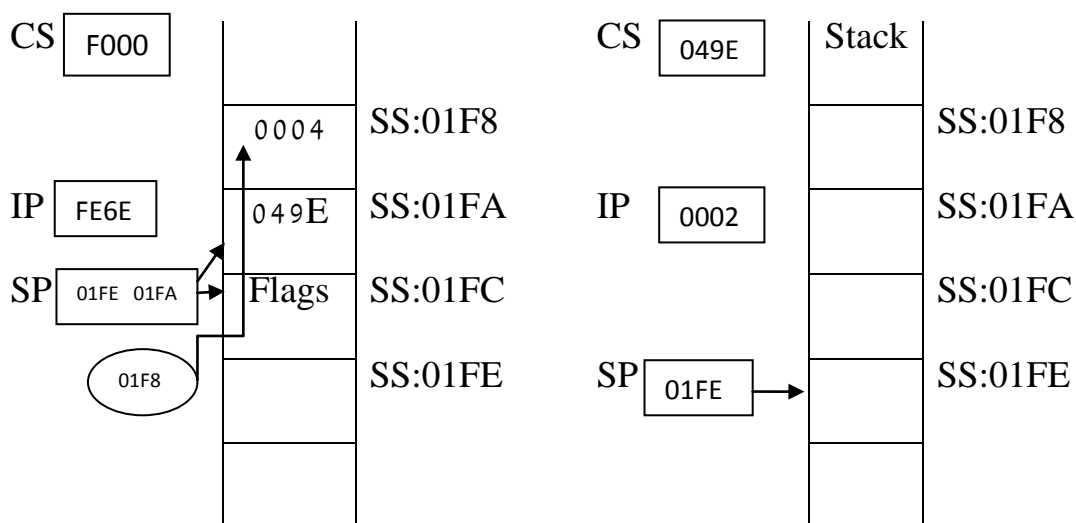
الف) IF و TF صفر شده و flag بر روی پشته push میشود.

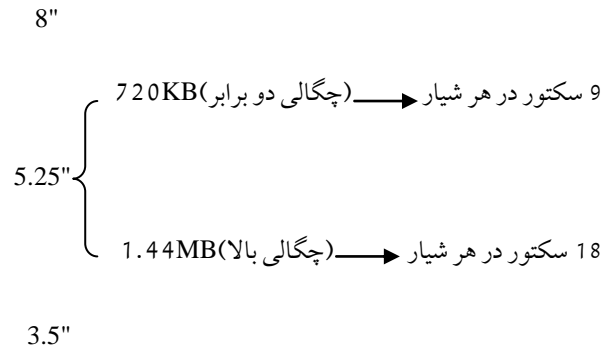
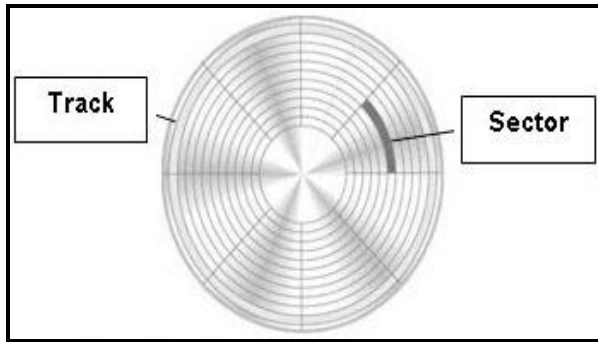
ب) محتویات CS وارد پشته میشود.

ج) آدرس برای وقفه محاسبه میشود (ضرب در 4) مثال: INT 1AH $1A * 4 = 68H$

د) محتویات ثبات IP را وارد پشته میکنیم.

هـ) محتویات word اول جدول بردار وقفه در ثبات IP و محتویات word دوم جدول بردار وقفه در ثبات CS قرار میگیرد.





دیسکت

تمام دیسکت ها 80 شیار دارند.

$$C_D = S * T * 512 * N_{Side}$$

T: تعداد شیار

S: تعداد سکتور در هر شیار.

N_{Side}: تعداد رویه

512: اندازه هر سکتور.

$$\text{Double - density: } C = 9 * 80 * 512 * 2 = 720 \text{ KB}$$

$$\text{High - density: } C = 18 * 80 * 512 * 2 = 1044 \text{ MB}$$

ظرفیت هارد:

$$C_{H.D.D} = C * T * S * 512 * N_{Side} * N_p$$

T: تعداد شیار در هر صفحه (رویه صفحه)

C: تعداد سکتور

N_p: تعداد صفحات پلاتر

S: تعداد سکتور در هر شیار

برای دسترسی به دیسکت از طریق BIOS از وقفه 13H استفاده میکنیم. این وقفه شامل توابع زیر است:

00: سیستم را Reset میکند.

01: وضعیت آخرین عملیات قبلی روی دیسک را مشخص میکند.

02: يك يا چند سكتور از روی دیسك میخواند.

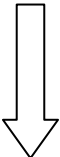
03: يك يا چند سكتور روی دیسك مینویسد.

04: صحت يك يا چند سكتور را مشخص میکند.

05: يك شیار را فرمت میکند.

08: نوع گرداننده و فرمت دیسك را مشخص میکند.

مثال) نوع گرداننده و فرمت دیسك را مشخص کنید:

Mov, نوع درایو, 	01H	5.25" (360 KB)
	02H	5.25" (1.2 MB)
	03H	3.5" (720 KB)
	04H	7.5" (1.44 MB)

MOV BL, 02

INT 13 H

خروجی:

DH → $\begin{cases} 01 \\ 02 \end{cases}$ تعداد رویه

CH → تعداد شیار

CL → تعداد سكتور

ES → DI اشاره گر به جدول پارامترهای درایو

برنامه نویسی :mouse

وقفه 33H مربوط به ماوس است. توابع وقفه 33H:

شماره تابع	هدف	ورودی	خروجی
00	تست وجود یا عدم وجود ماوس	AH	AX 0000H وجود ندارد FFFFH وجود دارد
01	نمایش اشاره گر ماوس	AH	ندارد
02	ناپدید شدن اشاره گر ماوس	AH	ندارد
03	تشخیص محل موشواره	AH	BX: وضعیت کلید CX: مختصات افقی DX: مختصات عمودی
04	حرکت مکان نما	AH ← DX, CX	<div> <div>↓</div> <div>↓</div> <div>ستون</div> <div>سطر</div> </div> (خروجی ندارد)

(مثال) برنامه راه اندازی موشواره بنویسید.

Mouse _INTIALIZATION PROC

MOV AH, 00

INT 33H

CMP AX, 00H

JMP FIN

FIN: RET

Mouse _ INTIALIZATION ENDP

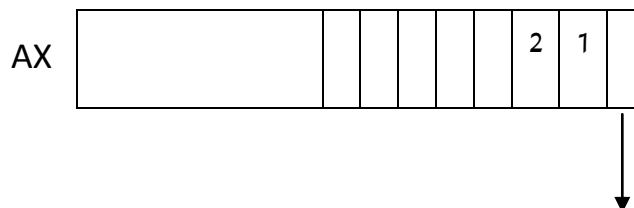
برنامه ای که مکان نه‌ای ماوس را مخفی کرده و نه‌ایش دهد سپس مجدد مخفی کرده و در مختصات دیگری نه‌ایش دهد.

MOV	AH, 02	} ماوس وجود دارد	MOV	DX, 80
INT	33H		INT	33H
INT	33H		MOV	AH, 01
MOV	AH, 01	} نه‌ایش ماوس	INT	33H
INT	33H			
MOV	AH, 04			
MOV	CX, 25			

خروجی	ورودی	هدف	وقفه
AX: وضعیت کلید فشرده شده	AH	بررسی فشردن کلید ماوس	۰۵
BX: تعداد دفعاتی که فشرده شده			
CX: سطر به پیکسل			
DX: ستون			

```
MOV    CX, 01101 B
MOV    CX, 7CH
MOV    CX, 25 D
```

✓ هر برنامه ای درباره ماوس مینویسیم اول باید هویت وجود ماوس را بررسی کنیم و بعد دستورات بعدی.



1 = کلید فشرده شده موشواره سمت چپ 0 = کلید چپ موشواره فشرده نشده

در بیت 1: (1): کلید سمت راست فشرده شده. (2): کلید سمت چپ فشرده شده.

در بیت 2: (1): کلید وسط فشرده شده. (0): کلید وسط فشرده نشده.

AX → 03H

AX → 07H

AX → 01H

AX → 02H

AX → 04H

برنامه ای بنویسید که مشخص کند کدام کلید ماوس فشرده شده و چند بار فشرده شده و مختصات آن نقطه

را در خروجی بنویسید.

MOV AH, 05

INT 33H

CMP AX, 01

AND AX, 07H

JNE 01 L1

L1: JNE 02 L2

L2: JNE 04 L4

L4;

وقفه صفحه نمایش 10H

وقفه صفحه کلید 09H و 16H

در حالت معمول مانیتور دارای 25 سطر (0 - 24) و 80 ستون میباشد (0 - 79).

ستون	سطر	ستون	سطر	
00H	00H	00	00	گوشه سمت چپ بالا
4FH	00H	79	00	گوشه سمت راست بالا
00H	18H	00	24	گوشه سمت چپ پایین
4FH	18H	79	24	گوشه سمت راست پایین
27H / 28H	0CH	39 / 40	12	وسط مانیتور

سرویس های وقفه 10H:

سرویس 00H: تعیین حالت مانیتور.

سرویس 01H: تعیین اندازه مکان نما.

سرویس 02H: تعیین محل مکان نما.

سرویس 06H: پاک کردن صفحه نمایش.

سرویس 09H: نوشتن يك يا چند کاراکتر با رنگ مشخص.

سرویس 0CH: نوشتن پیکسل در حالت گرافیک.

حداکثر صفحه	وفقی دهنده	تعداد رنگ	خطوط نمایش	حالت (در AL)
8	CGA, EGA, VGA	B&W	25 * 40	0H
				1H
				2H

05H	320 * 200	B&W	CGA,EGA,VGA	1
AH				
BH				
⋮				
FH				
10H				
11H				
12H				
13H	320 * 200	256	VGA	1

وقفه 21H سرویس 2BH:

AH ← 2BH

CX ← سال

DH ← ماه

DL ← روز

✓ وظیفه تغییر تاریخ را دارد.

در قطعه کد مقابل مقدار FF در ثبات AL به چه معناست؟

MOV AH, 2BH

INT 21H

الف) صحت ورودی

ب) عدم صحت ورودی

ج) تاریخ قبل از 2000 است

ج) تاریخ بعد از 2000 است

مثال برای وقفه 09H:

```
Data segment
My string db "This is my String"
Ends
Stack segment
dw 128 dup (0)
ends
Code segment
Start:
mov ax, data
mov ds, ax
move s, ax
Lea dax, my string
mov ah, 09h
Int 21H
mov ax, 4C00H
int 21H
ends
end start
```

MOV AH, 2CH

INT 21H

وقفه 21H سرویس 2CH:

✓ وظیفه ی آن خواندن ساعت است.

CL: دقیقه (0 تا 59)

CH: ساعت (0 تا 23)

DL: صدم ثانیه (0 تا 99)

DH: ثانیه (0 تا 59)

وقفه 21H سرویس 2DH:

✓ وظیفه آن تغییر ساعت است.

MOV AH,21DH

MOV CH,0EH

MOV CL,2EH

MOV DH,0FH

MOV DL,20H

INT 21H

قطعه کد مقابل بیانگر چیست؟

الف) 1981/4/5 ب) 14:46:15:22

ج) $2 * 16 = 32 + 14 = 46$

د) 14:15:32:46 ج) 1981/5/4

E = 14

سرویس 2CH: خواندن ساعت کامپیوتر.

سرویس 2DH: تغییر ساعت کامپیوتر.

MOV AH,01

INT 21H

سرویس 2AH: خواندن تاریخ.

سرویس 2BH: تغییر تاریخ.

برنامه ای بنویسید که از صفحه کلید رشته خوانده، در صورتی که بر کلید Enter فشار داده شود خواندن رشته متوقف گردد اندازه رشته حداکثر 20 کاراکتر میباشد پس از آن رشته مورد نظر را در خروجی چاپ کند.

User name db 20 dup (?)

mov cx, 20 شمارنده

Lea si, user name آدرس شروع را در اس آی قرار میدهد

For: اولین کاراکتر خوانده میشود و در AL میرود.

mov AH, 01H AL را با Enter مقایسه میکند

jz lbl 1 اگر برابر بود

mov [si], AL

inc si در غیر این صورت AL در حافظه میرود

cmp al, 0DH

```

Loop    NZ    for
Lbl 1:
For: mov  AH, 07H
INT 21H
cmp     AL,0DH
JNE for

```

برنامه سازی صفحه کلید:

7 6 5 4 3 2 1 0



يك بايت

Scan code

Make code: زمان فشردن شدن كد بـ 7 به 0 تبديل ميشود.

Bread code: زمان برداشتن كد بـ 7 يك ميشود.

Scan code: حاصل فشردن كليدها و تبديل شدن آنها به اعداد و ارسال سريال به PC.

هنگام فشردن يك كد بـ 7 هفتم scan code صفر است. زمان رها كردن اين بـ 7 يك ميشود. به دليل اينكه scan code يك بايت است نميتوان تمام عمليات ساده و تركيبى كد ها را در آن ايجاد نمود. بنابراين براى هر کدام از اجزاء كليدهاى تركيبى يك make code ارسال ميشود و سيستم با دريافت دو make code كه break code بعد از آن ارسال شده باشد آنها را با هم تركيب ميكند.

✓ وقفه 09H كار scan code و make code و break code را انجام ميدهد.

وقفه 16H: داراى دو تابع است:

00 : Read keyboard (پاك كردن كاراكتر از بافر)

01 : Read keyboard (كاراكتر را از بافر پاك نميكند)

02 : Read control keys

7	6	5	4	3	2	1	0
Insert	Caps lock	Num lock	Scroll lock	Alt	Ctrl	Left shift	Right shift

وقفه 21H: سرویس 01H: خواندن يك حرف از صفحه كلید و نمایش در مانیتور.

سرویس 07H: خواندن يك حرف از صفحه كلید بدون نمایش در مانیتور.

سرویس 09H: نمایش يك پیغام یا رشته روی مانیتور.

نوشتن يك یا چند کاراکتر با رنگ مشخص:

AH ← 09

MOV AH,09

(مثال)

AL ← کد اسکی

MOV AL,'T'

BL ← رنگ

MOV BL,011110101B

BH ← شماره صفحه

MOV BH,0

CX ← تعداد

MOV CX,4

INT 10H

نوشتن پیکسل در حالت گرافیک:

AH ← 0CH

MOV AH,0CH

AL ← color

MOV AL,30

(مثال)

CX ← x

MOV CX,100

DX ← y

MOV DX,100

BH ← page number

MOV BH,0

INT 10H

MOV AH,0

MOV AL,13H

INT 10H

→ خروجی

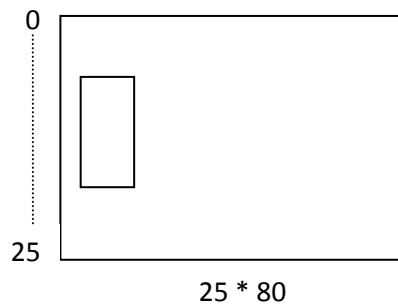
320 * 200 * 250

تعیین اندازه ی مکان نما (0 تا 15 سطر)

AH ← 01
CH ← START
CL ← END

(مثال)

```
MOV AH,1
MOV CH,5
MOV CL,10
INT 10H
```



تعیین محل مکان نما:

AH ← 2
DL ← شماره ستون
BH ← شماره صفحه فعال

```
MOV AH,2
MOV DH,10
MOV DL,15
MOV BH,0
INT 10H
```

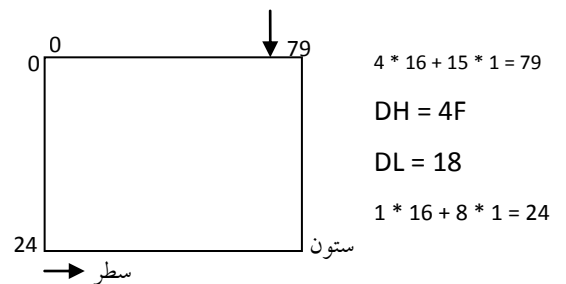
(مثال)

پاك کردن صفحه نمایش:

AH ← 6
AL ← 0
CL ← شماره ستون گوشه سمت چپ بالای پنجره
CH ← شماره سطر گوشه سمت چپ بالای پنجره
DL ← شماره سطر گوشه سمت راست پایین پنجره
DH ← شماره ستون گوشه سمت راست پایین پنجره
BH ← رنگ زمینه و حروف

```
MOV AX,0600H
MOV BH,61H
MOV CX,0000H
MOV DX,4F18H
INT 10H
```

(مثال)



INT 13.5 – Format Disk Track

AH = 05

AL = interleave value (XT only)

CX = track / cylinder number (see below for format)

DH = head number (0 – 15 dec.)

DL = drive number (0 = A:, 1 = 2nd floppy, 80h = drive 0, 81h = drive 1)

ES:BX = pointer to block of "track address fields" containing

Four byte fields for each sector to be formatted of the form:

1 byte track number

1 byte head number Size #

1 byte sector number Codes Bytes

1 byte sector size code 0 128

1 256

2 512

3 1024

on return:

AH = status (see ~ INT 13, STATUS ~)

CF = 0 if successful

= 1 if error

INT 13,10 – Test for Drive Ready (XT & newer)

AH = 10h

DL = drive number (0 = A:, 1 = 2nd floppy, 80h = drive 0, 81h = drive 1)

on return:

AH = status (see ~ INT 13, STATUS ~)

CF = 0 if successful

= 1 if error

ساختار دیسکت (soft disk): دارای تعدادی track است به صورت حلزونی که از 0 تا N شماره گذاری شده، هر track به چند sector تقسیم میشود و هر سکتور در فرمت DOS, 512 بایت است.

$$N_{\text{sectors}} \approx \text{Disket}_{\text{Type}} + \text{Format}_{\text{Type}}$$

تعداد سکتور 9	720 کیلوبایت (چگالی دو برابر)	5,25"	انواع دیسکت: } ۸"
تعداد سکتور 18	1044 کیلوبایت (چگالی بالا)	5,3"	

$$C_D = S * T * 512 * N_{\text{side}} \text{ ظرفیت دیسکت:}$$

$$\text{Double - density: } C = 9 * 80 * 512 * 2 = 720 \text{ KB}$$

$$\text{High - density} = C = 18 * 80 * 512 * 2 = 1044 \text{ MB}$$

ساختار دیسک سخت: چندین صفحه ی Platter بر روی هم (pack) با دو رویه

$$C_H = C * T * S * 512 * N_{\text{side}} * N_P$$

اطلاعات یا به صورت همزمان به صورت یک بایت در یک pack نوشته میشود و یا در شیارهای هم شعاع (سیلندر) از پایین ترین صفحه تا بالاترین صفحه نوشته میشود.

دسترسی به دسکت از طریق BIOS: وقفه ی 13H توابع دسترسی به دیسکت را در بایاس فراهم میکند.

ساختار ذخیره سازی دیسک نرم در نوع ۳,۵":

تعداد سکتور	سکتور پایان	سکتور شروع	
1	0	0	Boot Sector
9	9	1	FAT 1
9	18	10	FAT 2

DIRECTORY	19	32	14
DATA Area	33	2879	2847

توابع وقفه ی شماره ی H13:

شماره		شرح
00	Reset	سیستم دیسک به کنترلر درایو فرمان میدهد تا یک ریست سخت افزاری انجام دهد
01	Read Status	وضعیت آخرین عملیات دیسک را برمیگرداند
02	Rend	یک یا چند سکتور دیسک را میخواند
03	Write	یک یا چند سکتور را از حافظه، در دیسک مینویسد
04	Verify	صحت یک یا چند سکتور را مشخص میکند
05	Format	یک شیار را فرمت میکند

وضعیت گرداننده: تابع 01 پس از فراخوانی وضعیت گرداننده را در ثبات AH برمیگرداند، صفر نشان دهنده ی موفقیت آمیز بودن است و هر مقدار غیر صفر یا set شدن CF بیانگر خطاست.

نوع گرداننده و فرمت دیسک: تابع 08H اطلاعات زیر را در ثبات های مختلف قرار میدهد.

تعداد رویه 01 DH $01H = 5.25", 360KB$ نوع درایو BL

02 CH $02H = 5.25", 12MB$

تعداد شیار CL $03H = 3.5", 720KB$

تعداد سکتور $04H = 3.5", 1.44MB$

اشاره گر به جدول پارامترهای درایو ES:DI

DDFT: جدولی که پارامترهای مورد نیاز برای دسترسی به کنترلر دیسک در آن قرار دارد.

➤ برنامه ای بنویسید که وضعیت فعلی دیسکت را برگردانده و نوع گرداننده و فرمت دیسکت را مشخص کرده و سرانجام سیستم را reset کند.

➤ برنامه ای بنویسید که اطلاعات سکتور اول داده ی دیسک را خوانده و قبل از نوشتن مجدد این اطلاعات در همان محل از عدم تعویض دیسکت مطلع شود (تابع 16H). ثبات AH

خواندن بخش های دیسکت: تابع 02H این عمل را انجام میدهد و پارامترهای زیر نیاز میباشد.

پس از خواندن وضعیت خطا در AH و تعداد پخش های خوانده شدن در AL قرار میگیرد.

CF = 1 یعنی خطا رخ داده.

تعداد سکتور های خوانده شده: AL

مقدار نوع درایو (A یا B): DL

DH: شماره رویه

CL: تعداد سکتور

CH: تعداد شیار

آدرس بافر داده های خوانده شده: ES:BX

➤ برنامه ای بنویسید که چند سکتور از دیسکت را خوانده و مقادیر خوانده شده را در خروجی بنویسید.

نوشتن بخش های دیسکت: تابع 03H برای این عمل استفاده میشود (همانند قسمت قبل ثبات ها).

➤ برنامه ای بنویسید که شیارهای مشخص شده بر روی دیسکت را فرمت کند. (05H).

دسترسی به دیسک سخت از طریق BIOS: از همان وقفه ی 13H استفاده میشود، که برخی از این توابع برای کار با دیسک سخت قابل استفاده هستند.

تقسیمات دیسک سخت: آماده سازی دیسک سخت سه مرحله دارد:

1 - **Low – Level – Format** (قالب بندی سطح پایین): سازماندهی گرداننده به سیلندر، شیار و سکتور به نشانی هایی که بعداً قابل تشخیص و استفاده باشد. فقط برای هاردهای IDE است و در برنامه ی Bios setup وجود دارد.

2 - **FDisk**: برای استفاده از دیسک های بزرگتر از 2GB در DOS نیاز به این عمل یعنی بخش بندی کردن دیسک سخت میباشد.

3 - **Format**: دیسک به دو بخش primery و Extended فیزیکی تقسیم میشود. و قسمت Extended قابل تقسیم به گرداننده های منطقی است.

قسمت اول partition sector است و محل نگهداری اطلاعات نشانی شروع، خاتمه، نوع و ... در تقسیمات دیسک است که به این قسمت Partition Table گویند. در هنگام روشن شدن کامپیوتر در حافظه 0000:7C00 بارگزاری میشود. پس از بارگذاری در دو بایت آخر وجود مقادیر 55H و AAH به معنای بخش boot کننده است و در صورت نبود پیغام قرار دادن دیسک بوت کننده در درایو را میدهد.

➤ برنامه ای بنویسید که سکتور 333 دیسک سخت را خوانده و در خروجی بنویسد و بیان کند این سکتور در کدام صفحه و رویه و سیلندر و track قرار دارد؟