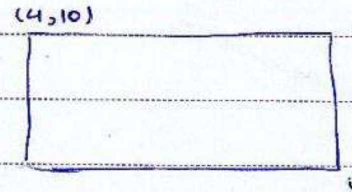
Subject:
Year. Month. Date. ( )

* برنامه‌ای بنویسید که یک مستطیل را ترسیم کند



کاربرد keyboard سریعی که به کلید فشار داده می‌شود ← scan code این انتقال serial است. و به صورت هم زمان

در صورتی که سیستم عامل داشته باشیم ⇒ INT 21H (DOS)

اگر با سخت افزار ⇒ INT 16H (BIOS)

scan code هم چنین برای اینکه کلید هنوز فشار داده شده یا رها شده است.

ممکن است که بیشتر مشغول باشد میتواند به keyboard پردازش یک پردازنده buffer استفاده می‌کند تا اگر کلید زده شد از بین نرود. یا این که چند کلید با هم زده شده باشد. از هیچ کدام این حالت‌ها برای انتقال صفی سیستم از بین نرود.

ممکن ابزاری هم کلید INT سخت hard ware هر وقت یک کلید از keyboard فشار داده شد

اولین چیزی که به پردازنده ارسال می‌شود ← press SCAN CODE key board ← INT 09

این (1 byte) make code و break را برای هر دو ascii تبدیل می‌کند

SCAN CODE → function key
ASCII code → حروف

بیش از 128 کلید هم که چه کلیدی فشار داده شده است

ENTER	0DH	1CH	فشار دادن ⇒ make code < 128 داشته باشیم
ESCAPE	1BH	01H	رها کردن ⇒ break code > 128
BACK SPACE	08H	0EH	برای break code ← 1
TAB	09H	0FH	

scan code برای همه کلیدهای ممکن فرق می‌کند

SCAN CODE برای این است که کدهای key board برای پردازنده با هم قاطی نشود

F1 → F10 ⇒ [0] [scan code]

مثل ASCII است فقط محتوای آن فرق می‌کند

AH → SCAN Code

AL → ASCII

PAPCO

Subject:

Year. Month. Date. ( )

Function 01H

AH ← 01

INT 21H

AL ← حرف

06H ⇒ خالی است یا نه keyboard buffer

AH ← 06H

DL ← FFH

- بافر خالی AL → 0 , ZF = 1 no character ready
- بافر خالی نباشد AL ← حرف , ZF = 0

07H

AH ← 07H

INT 21H

AL ← حرف

INT 21H

01: خواندن یک حرف همراه با نمایش آن

06: تشخیص خالی بودن بافر

07: خواندن یک حرف از صفحه کلید بدون نمایش

08: خواندن یک حرف از صفحه کلید بدون نمایش

(ctrl + Break)

PqPCO

Subject:

Year. Month. Date. ( )

0AH: خواندن یک رشته اطلاعات از صفحه کلید و قرار دادن آن در بافر نمایش

0BH: تست بافر صفحه کلید

0CH: می‌تواند service دیگری را call کند که باید buffer را خالی کنیم

08:

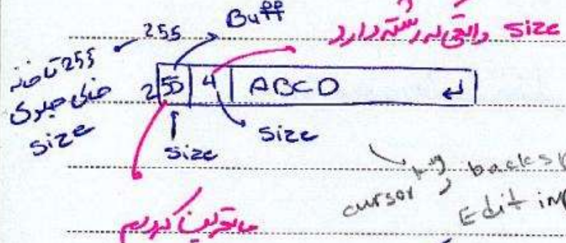
AH ← 08H

AL ← حرف

اگر extended باشه دفعه اول 0

باید برای کد کلید دوباره صدا زده بشه

0AH: یک بافر در حافظه باید مشخص شود تا داده‌ها ذخیره شود و با خواندن Enter تمام شود.



بافر باید دو بایت اول size را در خود داشته باشد

Ctrl+C می‌ره call INT 23

cursor و backspace

Edit input

خود INT ادیت می‌کند

```
Buff    DB    255, ?, 254 DUP(' ')
```

```
AH ← 0AH
DX ← offset buffer  →   mov AH, 0AH
INT 21H                 LEA DX, Buff
                        INT 21H
```

DS می‌ره buffer seg address

Subject:

Year. Month. Date. ( )

0AH ..... INDI از ...

0BH

AH ← 0BH

INT 21H

{ حرف در بافر نباشد AL → FFH
{ حرف در بافر نباشد AL → 0

0CH

AH ← 0CH

AL ← [?] → تابع سرویسی است که قبل از خالی شدن بافر اجرا شود

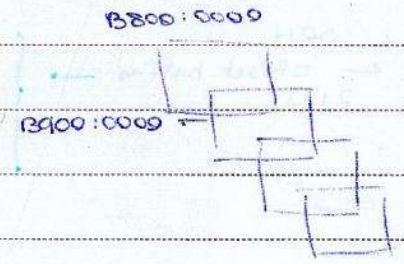
DX ← آدرس بافر DS:reg buffer

INT 21H

بافر خالی شود و منتظر صفحه کلید می‌شود

```
MOV ES, B800 H
MOV DI, 0000 H
MOV ES:[DI], 'A'
INC DI
MOV ES:[DI], 0EH
```

حرف A را در اولین خانه page چاپ کنند



Subject: فاطمه خدامرادی

Year. Month. Date. ( )

INT 16H

00H

AH ← 00H

INT 16H

AL → Ascii Code

AH → scan Code

AL=0 → F12 یا F1

AH → مقدار تعیین کننده → scan code

01H

AH ← 01

INT 16H

ZF → 0

ZF → 1

02H

AH ← 02H

INT 16H

AL →

AL 0 = 1 → Right shift

AL 1 = 1 → Left "

" 2 = 1 → ctrl

" 3 = 1 → Alt

" 4 = 1 → Scroll lock on

" 5 = 1 → Numlock on

" 6 = 1 → Caps on

" 7 = 1 → Insert on

Subject:

Year. Month. Date. ( )

11 H

9H ← 11H

INT 16H

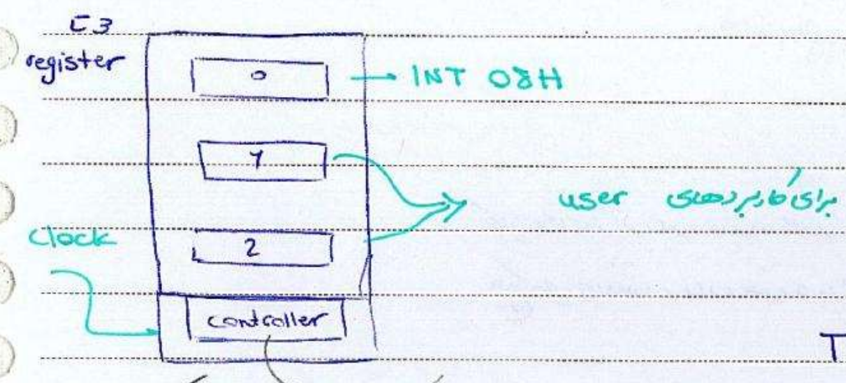
ZF=0 → حرف دریافت است

ZF=1 → ... نیست

AL → ASCII code

AH → Scan Code

8254 Timer (شمارنده



از تا کی چیپ‌ها ما چه بری استفاده کنیم مربوط

به اینجاست

T = 55ms

Clock 18.2 = ثانیه

PAPCO

Subject:
Year. Month. Date. ( )

1A

Date & Time

INT 1AH — BIOS

INT 21H — DOS

00H خواندن شمارنده

AH ← 00

INT 1AH

CX:DX ← شماره

یک counter 32 بیتی روی خواندن ... set کالند

در هر ثانیه ... بار ای ... تاند بیتی زند

شمارنده

32 بیتی

AL { 0 زمان بین خواندن آخرین ها < 24 hour ; 1 > 24 hour }

01H

AH ← 01

CX:DX ← ?

INT 1AH

INT 1A BIOS

CH — ساعت

CL — دقیقه

DH — ثانیه

02H خواندن ساعت

AH ← 02H

INT 1AH

CH → ساعت (BCD)

CL → دقیقه (BCD)

DH → ثانیه

ساعت به صورت BCD ذخیره می‌شود

CF → { 0 ساعت کار می‌کند ; 1 ،، ،، نمی‌کند }

PAPCO

Subject:

Year. Month. Date. ( )

03 H تنظیم ساعت

AH ← 03H

CH ← ساعت

CL ← دقیقه

DH ← ثانیه

INT 1AH

04 H خواندن تاریخ

AH ← 04H

INT 1AH

CH → قرن (19,20)

CL → سال

DH → ماه

DL → روز

CF → {0 / 1

05 H تنظیم تاریخ

AH ← 05

CH ← قرن

CL ← سال

DH ← ماه

DL ← روز

INT 1AH

برنامه‌ای بنویسید که یک عدد تصادفی در بازه [0,100] تولید کند و در صفحه نمایش نشان دهد

(09H, 01H → 1AH) با دستور زیر می‌توان عدد 32 بیتی تصادفی با اندازه میلی ثانیه تولید کند

Subject:

Year. Month. Date. ( )

INT 21H (DOS)

2CH خواندن ساعت

AH ← 2CH

INT 21H

CH → ساعت
CL → دقیقه
DH → ثانیه
DL → صدم ثانیه

(Binary)

2DH تغییر ساعت دستگاه

AH ← 2DH

CH ← h

CL ← m

DH ← s

DL ← صدم ثانیه

INT 21H

2AH خواندن تاریخ Read date

AH ← 2AH

INT 21H

AL → روز هفته (۰ - یکشنبه ، ۱ - دوشنبه ، ...)

CX → 1980 ≤ باینری ≤ 2099

DH → شماره ماه { 1 - Jan, 2 - Feb, ... }

DL → 1 ≤ باینری ≤ 31

Subject:

Year. Month. Date. ( )

2BH

AH ← 2BH

CX

DH

DL

INT 21H

AL | 0 → تغییر اندازه

FFH → نمایش

Subject:
Year. Month. Date. ( )

```
Back:   mov   AH,02H
        INT   16H
        TEST  AL,00001000B
        JNZ   OVER
        { not pressed }
OVER:   { Alt is pressed }
```

مثل and بیتی است ولی result در Accumulator اثر نمی‌گذارد

INT 21H ← DOS

1CH

تغییر روتین interrupt تایمر

یا

منظر برای routin Interrupt تایمر

یک ثانیه = 18.2 Hz بار این خط پخش می‌شود

تغییر روتین INT یک

```
mov   ES,0
mov   DI,0
mov   AX,mmmm H    offset Daly
MOV   ES:[DI],AX
ADD   DI,2
mov   AX,nnnn H    SEGMENT Delay
mov   ES:[DI],AX
```

نوشتن روتین offset

نوشتن seg روتین

	16	
	mmmm H	0
	nnnn H	2

25H

تغییر روتین اینترابت برای هر دفعه

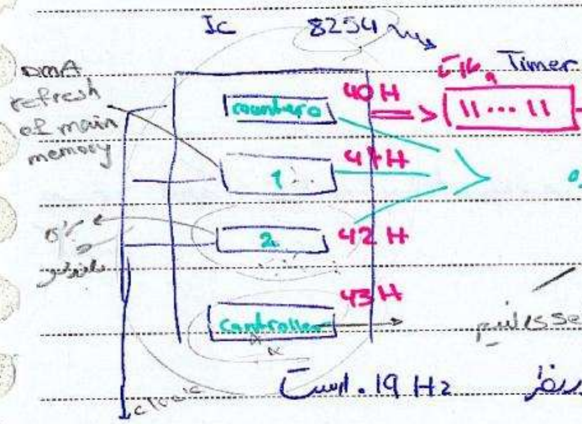
```
push DS
AH ← 25H
AL ← شماره وقفه‌ای که روتین را می‌خواهیم تغییر دهیم
DX ← offset روتین که می‌خواهیم
DS ← segment روتین جدید
INT 21H
pop DS
```

delay proc یک جایی در حافظه را اجرا کنیم

(35)

Subject:
Year. Month. Date. ( )

تولید صوت: توسط بلندگو که روی motherboard



در IBM PC به عنوان تراشه fix و کلید ... هم دارد

از هر کدام از سه (و) کلید استفاده کنیم یعنی set کنیم

هر اینها باید یک clock هر لحظه در هر کامپیوتری به فرکانس حدوداً 1.19 Hz است

1.19 Hz

سه رجیستر برای سرعت رجیستر map شده برای خواندن نوشتن

مثل memory که با memory نسبت I/O Device است برای خواندن نوشتن

out in

OUT 40H, AL — مقدار AL روی counter 0 قرار می‌دهد

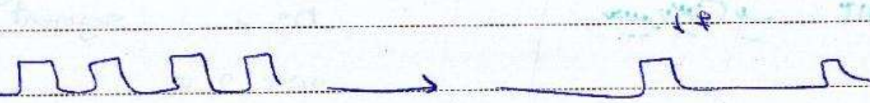
IN AL, 40H

counter 0، reserve شده است که مربوط به INT 8 است

" 1 " RAM را refresh کردن data

" 2 " سه برای ترتیب با این یک clock، generate کنیم

یک تعدادی از clock ها را در نظر می‌گیرد و اینها را یکی می‌کند

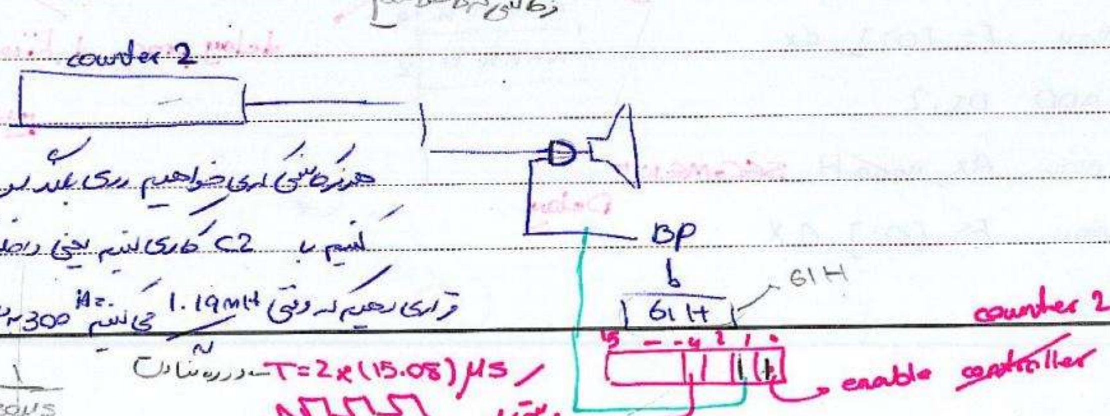


$$\frac{1.19\,H}{N} = 300$$

N ← عددی که باید قرار دهیم

فرکانسی که می‌خواهیم

$\frac{1.19}{\text{فرکانس}} = N$



هر فرکانسی که می‌خواهیم روی بلندگو generate کنیم با C2 کاری کنیم یعنی دامنش را طوری قرار دهیم که وقتی 1.19 MHz می‌کنیم 300 Hz بدهد

$T = 2\times(15.08)\,\mu s$ — سه دوره تناوب

$f = \frac{1}{30\mu s}$

یعنی که

P4PCO

Subject:
Year. Month. Date. ( )

$$counter = \frac{1.19 * 10^6}{4 \times 10^2} = \frac{1.19}{4} * 10^4 = 2975$$

```
MOV  BX, 2975      -> N

MOV  AL, BL
OUT  42H, AL

MOV  AL, BH
OUT  42H, AL
```

برای اینکه به controller بگیم که می‌خواهیم با این port کار کنیم چون عمل است فورش در حال کار باشد قبل از این که دیتا رو بدیم دستور را بنویسیم

اجباری
```
MOV  AL, 0B6H
OUT  43H, AL
```

برای اینکه بگوییم که صدااش در بیاد بیت بیت مربوط به BP که آدرس آن 61H است ۱ بشود

```
IN  AL, 61H
```

مقدار داخل آدرس 61H

برای اینکه هم enable رو فعال کنیم که هم بگوییم صداش در بیاد

```
OR   AL, 0000 0011B
OUT  61H, AL
; delay
IN   AL, 61H
AND  AL, 111111 00b
OUT  61H, AL
```

برای اینکه صدا قطع بشه

MOV AL, 0B6H
OUT 43H, AL

Subject:

Year. Month. Date. ( )

Delay

```
Back:   IN   AL, 61H
        AND  AL, 10H
        CMP  AL, 10H
        JZ   Back
```

15 μs → تأخیر طول کشیده

```
Back2:  IN   AL, 61H
        AND  AL, 10H
        CMP  AL, 10H
        JNZ  Back2
        LOOP Back
```

درلوپ پس گرفتن

Subject:
Year. Month. Date. (.)

File System

sector های در دسترس است ← برای اینکه بدانیم

برای FAT x حجم فرق دارند ← FAT → File Allocation Table

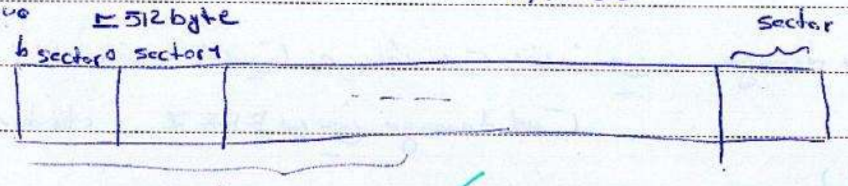
- FAT 12
- FAT 16
- FAT 32

cluster مجموعه‌ای از sector

FAT 12
FAT 16
FAT 32

برای اطلاعات بصورت byte، byte نگه‌ داشته بود پس تصمیم گرفتند به صورت sector ذخیره کنند (اندازه‌های متفاوت)

معمولاً 512 byte ← کم بود واحد بزرگتر cluster



- byte per section
- sector per cluster
- hardisk contains ? sector

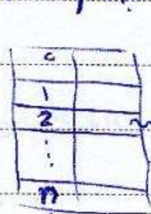
← با اینها می‌توان از fat استفاده کرد

FAT اطلاعات یک فایل را در خودش ذخیره نمی‌کند، کوچکترین واحدی که به یک file assign می‌شود cluster

هر چه cluster بزرگتر باشد ← هدر رفت بیشتر می‌شود.

به خاطر اینکه می‌خواهیم add، del، remove بشوند، cluster الزاماً پشت هم نیستند مثل linkedlist عمل می‌کنند ← link را در خود cluster نگه نمی‌دارند ⓪

در واقع یک data structure به نام FAT نگه می‌دارند



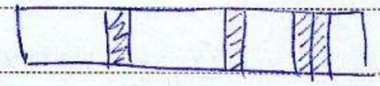
entry 2 را هم که بیاییم cluster بعدی را نشان می‌دهد

اگر جایی هر entry null باشد، آخرین cluster است

cluster FFF8 ← آخرین cluster اگر جایی هر cluster این بود یعنی دیگر آخرین cluster
FFFF

Subject:
Year. Month. Date. ( )

communication پس درجایی که می‌خواهیم اطلاعات رد و بدل کنیم کند باشد ← سرعت بیشتر



← file متشکل از 4 cluster

تبدیل logical به physical توسط OS

برحسب sector یا cluster ← برای انجام کار به physical تبدیل می‌شود ← logical

برای آدرس دهی hard کدام هد / track / کدام سیلندر ← physical

- hard disk driver
- Board Circuit

هر cluster یک جا را بصورت فیزیکی ذخیره می‌کند

بعضی وقت‌ها cluster قابلیت ذخیره اطلاعات را ندارند یعنی damage می‌شوند ← bad cluster

اگر محتوای cluster برابر FFF7 بود یعنی damage است.

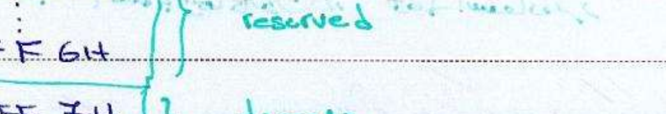
FFF 0H ... FFF 6H	reserved
FFF 7H	damage
FFF 8H ... FFFFH	پایانی

Drive ← A, B → Floppy disk, C, D... → Hard disk

volume ← 0, 1 ... 2, 3 ...

partition

hard قبل از استفاده به Drive یا volume یا partition تقسیم می‌شود



این partition ها file system خودشان را دارند

تعداد cluster
$\log_2$

FAT ← الحاق‌های داخلی table (data structure)

← ۱۶ بیت آدرس دهی از cluster ۰ تا $2^{16}-1$ ← cluster

Subject:

Year. Month. Date. ( )

چون تعداد cluster ها کمتر است → size harddisk کمتر → size volume کمتر

در هر partition، 2 cluster اول از قبل داده دارد و reserved است

(بین ۴، ۸، ... 32 ، حداکثر 64 متغیر)

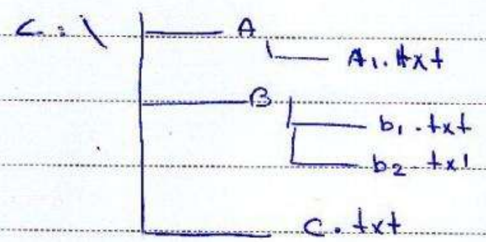
$$\text{size volume} = 2^{32} * \left(\frac{\text{sector per}}{\text{cluster}}\right) * \left(\frac{\text{bytes per}}{\text{sector}}\right)$$

2^{16} x

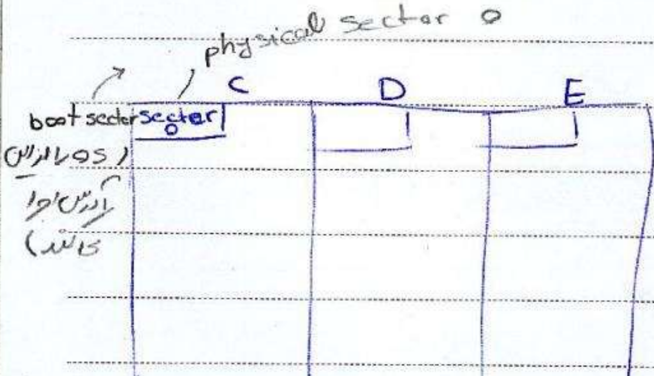
2^{12}

2^{32}

file ها و directory ها داخل volume چگونه ذخیره می‌شوند؟



در هر volume، 2 cluster اول ذخیره است یعنی به entry ... دسترسی پیدا می‌کنیم



فرض → OS را با چه پرتاب کند D:

PAPCO