عنوان درس:

برنامه سازی سیستم

جلسه ۳: مروری بر زبان اسمبلی

مدرس:
اعظم کبوتری کهنه شهری

فهرست مطالب

توضیحات

- توضیحات در زبان اسمبلی بعد از علامت ; در خط جدا یا در همان خط دستور و بعد از دستور نوشته می شوند:

```
; this example prints out  "hello world!“
mov cx, 12  ; number of characters.
```

- اسمبلی نسبت به بزرگ و کوچک بودن حروف حساس نیست.

کلمات رزرو شده

کلمات رزرو شده در زبان اسمبلی به پنج دسته تقسیم می شوند:

- اسم ثبات ها: ax, bx, cx,...
- دستورات: mov, add, sub,...
- شبه دستورات: page, end, assume,...
- عملگرها: size, far, near,...
- سمبولهای از پیش تعریف شده: @Data, @Model,...

شناسه

- نام یک عنصر (داده/دستور) است و جهت ارجاع به آن مورد استفاده قرار می گیرد، انواع شناسه:
  - **نام (شناسه داده):** به آدرس یک داده اشاره می کند.

```
data1 db 0
```

  - **برچسب (شناسه دستور) :** به آدرس یک دستور اشاره می کند.

```
start: mov ax,0
```

- کاراکترهای مجاز برای شناسه:
  - حروف الفبای انگلیسی بزرگ و کوچک
  - ارقام ۰ تا ۹
  - کاراکترهای . @ $ _ ?
  - کاراکتر اول می تواند یک حرف یا یکی از علامت های $ _ ? باشد.

ساختار دستورات اسمبلی

- ساختار یک دستور اسمبلی:

```
[شناسه]      دستورالعمل      [عملوندها]
start:       mov             ax,0
```

- انواع دستورات:
  - دستوراتی که اسمبلر آنها را به کد ماشین قابل اجرا تبدیل می کند:

```
add, move, sub, ...
```

  - شبه دستورات (پیش پردازنده ها/راهنماها): دستوراتی که فقط در زمان اسمبل کردن برنامه عمل می کنند و برای آنها کد ماشین تولید نمی شود.
    - تعریف داده، تعریف پروسیجر و ...

شبه دستورات کنترل لیست برنامه اسمبلی

- **شبه دستور page**

```
page Length, width
page 50,80
```

  - length : تعداد خطوط در هر صفحه
  - width : تعداد کاراکتر در هر خط

- **شبه دستور title**

```
title text
title ‘example1.ASM’
```

شبه دستورات تعریف سگمنت

```
name segment [align] [combine] [‘class’]
…
…
…
name ends
```

- align : مرز شروع سگمنت که مقدار پیش فرض آن PARA است.
- combine: برای پیوند سگمنت با سگمنت های دیگر
- class: تعیین گروه سگمنت

```
codeSG segment ‘code’
…
…
…
codeSG ends
```

شبه دستورات تعریف روال

```
name proc [far/near]
…
…
…
name endp
```

- near: روال از داخل سگمنتی که در آن تعریف شده قابل فراخوانی است.
- far: روال از سگمنت های دیگر قابل فراخوانی است.

```
main proc far
…
…
…
main endp
```

شبه دستور خاتمه برنامه

- آخرین دستور برنامه است:

```
end progname
```

به جای progname می توان از برچسب اولین دستور برنامه یا نام اولین روال far برنامه استفاده کرد:

```
end main
```

شبه دستور برقراری ارتباط بین هر سگمنت با ثبات مربوطه

- این شبه دستور باید در ابتدای سگمنت کد و قبل از روال main باشد:

```
assume ss:stacksegname,ds:satasegname,cs:codesegname,…
```

```
assume ss:stackSG,ds:dataSG,cs:codeSG
```

- مقداردهی به ثبات های ds و es در برنامه های exe اولین دستورات روال main هستند:

```
mov ax,dataSG
mov ds,ax
```

شبه دستور ORG

- آدرس افست دستور یا داده بعدی را در حافظه مشخص می کند:

```
        org 100H
start:  jmp start
```

  - در برنامه های com اولین دستور باید در آدرس 100H قرار گیرد.

```
        org 30H
data1   DB 52
```

شبه دستور تعریف ثابت

```
name equ value
pi equ 3.14
```

- اعداد در دستورات می توانند در سه مبنا نوشته شوند:
  - دهدهی: 120d
  - دودویی: 100110110B
  - هگزادسیمال: 100H

شبه دستورات تعریف متغیر

این شبه دستورات در سگمنت داده برای تعریف متغیر (رزرو حافظه) و تخصیص مقدار اولیه به آن استفاده می شوند:

- **شبه دستور db:** تعریف متغیر یک بایتی

```
data1 db 2      ;char data1=2
data2 db ?      ;char data2
data3 db ‘A’
```

- **شبه دستور dw:** تعریف متغیر دو بایتی

```
data1 dw 14     ;int data1=14
```

شبه دستورات تعریف متغیر (ادامه)

- **شبه دستور dd:** تعریف متغیر چهار بایتی

```
data1 dd 14          ;long int data1=14
data2 dd 17.5        ;float data2=17.5
```

- **شبه دستور dq:** تعریف متغیر هشت بایتی
- **شبه دستور dt:** تعریف متغیر ده بایتی
- **شبه دستور dup :** تعریف متغیر ده بایتی

```
data1 db 20 dup(?)        ;char data1[20]
data2 db 20 dup(5)
```

مجموعه دستورات اسمبلی برای 80X86

محاسباتی	add, adc, sub, sbb, mul, imul, div, idiv, inc, dec, neg, …
تبدیل	aaa, aas, aad, aam, daa, das
شیفت بیت ها	shl, sal, shr, sar, ror, rcr, rol, rcl, …
مقایسه	cmp, cmpsn, test, …
انتقال داده	mov, lea, lds, les, lea, xchg, …
عملیات پرچم	clc, stc, cld, std, cli, sti,…
ورودی/خروجی	in, out,…
منطقی	and, or, not, xor

مجموعه دستورات اسمبلی برای 80X86 (ادامه)

loop, loope, loopne, loopz, loopnz,…	حلقه
nop, hlt, wait, esc, lock	کنترل پردازشگر
pop, push, …	عملیات پشته
cmps, movs, …	عملیات رشته ای
Je, jne, jz, jnz, jcxz, …	پرش شرطی
jmp, call, ret, int, iret,…	پرش بدون شرط
cbw, cwd, …	تبدیل نوع

عملوند دستورات

- دستورات دو عملوندی:

منبع ، مقصد **دستورالعمل**

  - عملیات با استفاده از مقادیر منبع و مقصد انجام شده و نتیجه در مقصد ذخیره می شود.

- منبع : مقدار ثابت (بلافصل) ؛ ثبات یا آدرس خانه حافظه
- مقصد: ثبات، آدرس خانه حافظه

روش های آدرس دهی

بلافصل:
• منبع مقدار ثابت
• مقصد: ثبات (غیر سگمنت) یا آدرس حافظه
	`mov ax, 123h`
	`data1 db 23h`
	`add data1, 37h`

ثبات ها: • منبع یا مقصد یا هر دو ثبات	`mov ax, 123h` `add ax, bx`
مستقیم : • منبع یا مقصد افست آدرس خانه حافظه • ثبات سگمنت پیش فرض: ds	`mov al, [4000]` `mov al, data1`

غیر مستقیم با ثبات:
• افست آدرس در یکی از ثبات های bx,bp,si,di
• ثبات سگمنت پیش فرض: ds یا ss
	`mov al, [bx]`
	`mov al, [si]`
	`mov al, [di]`
	`mov al, [bp]`

روش های آدرس دهی

غیرمستقیم نسبی با ثبات پایه: • افست آدرس موثر : محتوای bx یا bp بعلاوه یک مقدار جابه جایی • ثبات سگمنت پیش فرض: ds یا ss	`mov al,[bx+2]` `mov al,[bp+2]`
غیرمستقیم نسبی با ثبات ایندکس: •افست آدرس موثر : محتوای siیا di بعلاوه یک مقدار جابه جایی • ثبات سگمنت پیش فرض: ds	`mov al, [si+2000h]` `mov al, [di+2000h]`

غیرمستقیم نسبی با ثبات پایه و ایندکس:
•افست آدرس موثر : محتوای bx/bp بعلاوه si/di بعلاوه یک مقدار جابه جایی
•ثبات سگمنت پیش فرض: ds یا ss
	`mov al, [bx][si+2]`
	`mov al, [bx][di+2]`
	`mov al, [bp][si+2]`
	`mov al, [bp][di+2]`