Notas II

Organización de las Computadoras

Luis Eduardo Galindo Amaya (1274895)

Asignatura	Organización de Computadoras (331)
Docente	Arturo Arreola Alvarez
Fecha	09-11-2022

Organización de las Computadoras

Luis Eduardo Galindo Amaya (1274895)

09-11-2022

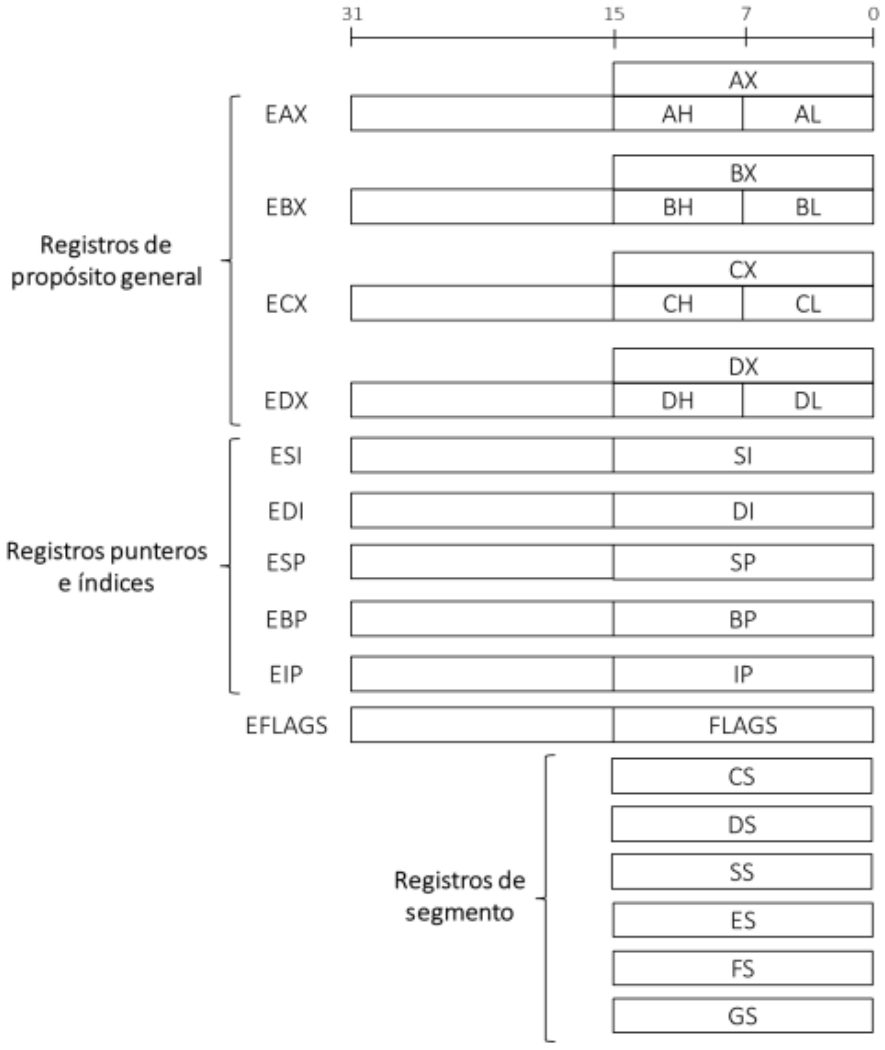

Figura 1: Registros del procesador

Contador De Programa (PC)

El registro más importante es el Contador de programa (PC), que apunta a la siguiente instrucción que se buscará para su ejecución.

Registro De Propósito General

- son 4 registros de 32 bits, EAX, EBX, ECX y EDX.
- Son considerados de propósito general, pero tienen algunas funciones específicas.

  EAX El registro de aritmética principal.

  EBX Para almacenar punteros a memoria.

  ECX Bucles

  EDX Se utiliza para multiplicacion y division

Partes Del Registro

cada registro de proposito general esta dividido en cuatro secciones, en el caso del registro EAX, EAX, AX, AH y AL.

EAX es el registro completo de 32 bits

AX es la mitad del registro EAX (32 bits)

AH es la mitad superior del registro AX (8 bits)

AL es la mitad inferior del registro AX (8 bits)

Podemos interpretar esto como una jerarquía de acceso al registro, no podemos pasar datos entre los registros si no son del mismo tamaño. Por ejemplo no podemos copiar el valor de `BX` en `EAX` porque `BX` es de 16 bits y `EAX` es de 32, pero si podemos pasar los valores de `EBX` a `EAX`, por lo tanto si queremos pasar un valor de un registro mas pequeño a uno mas grande tendremos que mover copiar el valor a un registro del mismo tamaño:

```
1 ;; esto no funciona
2     mov eax, bx
3
4 ;; esto si funciona
5 ;; aseguramos que el registro este vació
6     mov eax, 0
7     mov ax, bx
```

es importante entender que podemos ver el registro como un solo valor unico o como un valor con multiples partes para esto imaginemos la siguiente situacion, queremos representar un punto con sus coordenadas (cada una con una variable de 8 bits) :

```
1 ;; VALOR_X:   36  0010'0100
2 ;; VALOR_Y:  127  0111'1111
3
4     mov al, VALOR_X
5     mov ah, VALOR_Y
```

como cada variables es de 8 bits en total esta estructura ocupa 16 bits de memoria, por lo que podemos ponerla en el registro `AX` sin problemas y así unimos dos intrucciones en una:

```
1 ;;  VALOR_X:     36              0010'0100
2 ;;  VALOR_Y:    127              0111'1111
3 ;; VALOR_YX:  32548  0111'1111'0010'0100
4
5     mov aex, VALOR_YX
```

Registros De Punteros

- Contienen un registro de 16 bits en su parte menos significativa.
- son 4 registros de 32 bits:

  ESI y EDI se utilizan para almacenar punteros a memoria, especialmente para operaciones con cadenas

  ESP Manipulación de la pila

  EBP Apuntador a la pila

  EIP Apuntador de instrucción

Registros De Segmentos

- No se pueden separar.
- Todos son de 16 bits.

  CS Segmento de código.

  DS Segmento de datos.

  SS Segmento de pila

  ES, FS y GS Segmentos extra.

Registro De Banderas

Registros de banderas

- Registro de 32 bits.
- Bits 18 al 31 están reservados.
- Bits 0 al 11 bits de banderas.
- Indica la condición actual del procesador.
- Contiene información de la última operación aritmética

Banderas

Signo de la operacion (S) 0 positivo y 1 negativo

Parieradad (P) 0 impar, 1 par

Interrupciones (I) 0 activadas, 1 desactivadas

sobreflujo (O) bit de sobreflujo

Significado De Cada Registro

POS	Abreviacions	Significado
0	C	Acarreo
1		
2	P	Paridad
3		
4	A	Acarreo auxiliar
5		
6	Z	Cero
7	S	Signo
8	T	Bandera trampa
9	I	Habilitar interrupciones
10	D	Bandera de dirección
11	O	Sobreflujo
12	IO PL	-
13	IO PL	-
14	NT	-
15		
16	RF	
17	VM	
18-31	RESERVADOS	

Modos De Redireccionamiento

Caracteristicas

Deplazamiento que se se suma un valor fijo a DS[1]

Base que tiene un registro para la posicion

Indice que tiene un registro que puede cambiar su valor

Índice escalado es un registro que se multiplica por el tamaño del tipo de dato que se desea conocer

Tabla De Redireccionamiento Completa

Inmediato	`MOV EAX, 0x12345678`
Registro	`MOV EBP, EAX`
Desplazamiento	`MOV EAX, [12]`
Base	`MOV EAX, [EBX]`
Base con desplazamiento	`MOV EAX, [EBX+E7027193]` `MOV [ESP-5],AH`
Base con índice	`MOV EAX, [ECX+EBP]` `MOV word [EDI+ESI], F5`

Base con índice y desplazamiento	`MOV EAX, [ECX+EBP+2F19]`
	`MOV [ESP][ECX][4B024], AH`
	`MOV EDX, [EBP+EBX-E02719]`

Índice escalado	`MOV EAX, [ECX*4]` `MOV [EBP*2], AH`
Índice escalado y desplazamiento	`MOV EAX, [5][ECX*4]` `MOV word[EDI*8+1F3], 1B`
Base con índice escalado	`MOV EAX, [EBP+ECX*4]` `MOV [EDX*2+EBP], CH`
Base con índice escalado y Con desplazamiento	`MOV [38][ECX+EBP*4], AX` `MOV [2A][EDX*2+EBP], CH`

[1] Segmento de datos.

Conjunto De Instrucciones

Instrucción	Descripción
XCHG	Intercambia valores entre una dos registros del mismo tamaño o una posición de memoria y un registro.
IN	Lee un dato de un dispositivo de E/S y lo almacena en el acumulador.
OUT	Transfiere un dato del acumulador a un puerto de E/S.
LAHF	Carga el byte menos significativo del registro de banderas en AH
SAHF	Almacena el valor de AH en el byte menos significativo en EFLAGS.
LEA	Guarda en el primer operando la dirección efectiva del segundo operando.
	INSTRUCCIONES DE LA PILA
Instrucciones	Descripción
PUSH	Inserta un dato de 16/32 bits a la pila
POP	Remueve un dato de 16/32 bits de la pila
PUSHF	mete los bits 0-15 del registro de banderas a la pila
POPF	remueve 2 bytes de la pila y los almacena en EFLAGS.
PUSHFD	PUSHFD mete los bits 0-31 del registro de banderas a la pila
POPFD	POPF remueve 4 bytes de la pila y los almacena en EFLAGS

Pila

Generalidades de la pila

La pila guarda los valores de manera decreciente por lo tanto los valores agregados a la pila al final obtendran posiciones mas significativas:

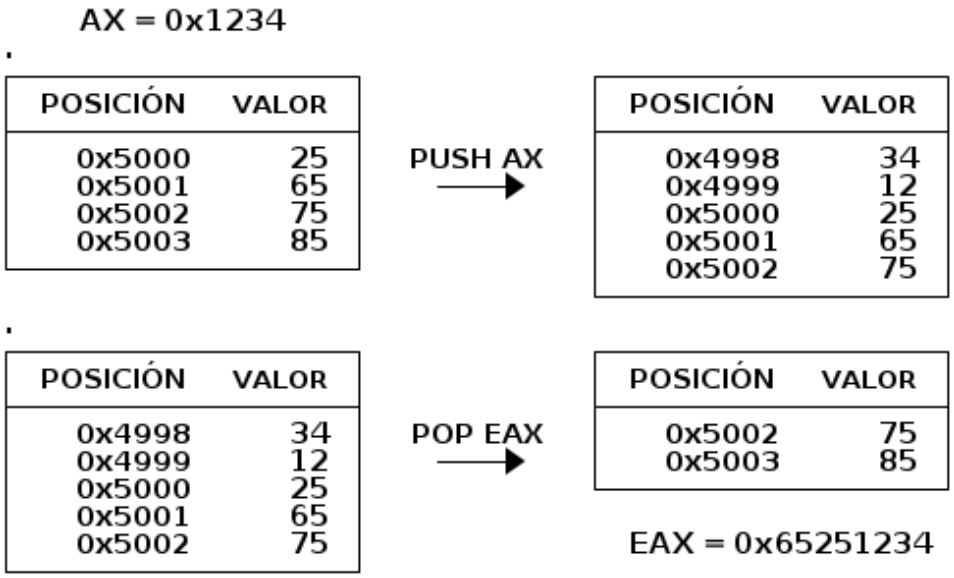

Figura 2: Comportamiento de la pila

PUSH

16 bits push con un registro de 16 bits (`AX`, `BX`, `CX`, `DX`)

32 bits push con un registro de 32 bits (`EAX`, `EBX`, `ECX`, `EDX`)

Otras formas de PUSH

`PUSH WORD 4567h` Inserta a la pila el valor inmediato `4567h`.

`PUSH DWORD 154567h` Inserta a la pila el valor inmediato `154567h`.

`PUSH WORD[ECX]` Inserta a la pila el valor almacenado en `Mem[ECX]`.

`PUSH DWORD[ECX+ESI*4]` Inserta a la pila el valor almacenado en `Mem[ECX+ESI*4]`.

POP

16 bits pop con un registro de 16 bits (`AX`, `BX`, `CX`, `DX`)

32 bits pop con un registro de 32 bits (`EAX`, `EBX`, `ECX`, `EDX`)

Otras formas de POP

`POP WORD[ECX]` Remueve de la pila 16 bits y los almacena en `Mem[ECX]`.

`PUSH DWORD[ECX+ESI*4]` Remueve de la pila 32 bits y los almacena en `Mem[ECX + ESI*4]`.

Operaciones Aritméticas

Instrucción	Descripción
`ADD`	Suma los operandos y almacena el resultado en el primer operando.
`ADC`	Realiza la operación de suma entre los operandos, sumándole el bit de acarreo del registro de banderas (suma de 64 bits)
`INC`	Incrementa en 1 el operando
`SUB`	Resta los operandos y almacena el resultado en el primer operando
`SBB`	Realiza la operación de resta entre los operandos, restando el bit de acarreo del registro de banderas (resta de 64 bits).
`DEC`	Decrementa en 1 el operando.
`TEST`	Realiza una operación AND entre dos operandos y actualiza el estado del registro de banderas

Multiplicación (`MUL`)

Multiplica el operando por AL, AX o EAX.

- `MUL BL` se puede traducir como `AX = AL*BL`
- `MUL DI` se puede traducir como `DX:AX = AX*DI`
- `MUL ECX` se puede traducir como `EDX:EAX = EAX*ECX`
- `MUL Word[EBP]` se puede traducir como `DX:AX = AX*Mem[EBP]`

Multiplicación (IMUL)

IMUL reg / IMUL mem

- IMUL BL puede traducirse como AX = AL*BL
- IMUL DI puede traducirse como DX:AX = AX*DI
- IMUL ECX puede traducirse como EDX:EAX = EAX*ECX
- IMUL Word[EBP] puede traducirse como DX:AX = AX*Mem[EBP]

División de 8 bits (DIV e IDIV)

- Coloca el resulado de la division de AL y el modulo de la operacion en AH.
- DIV BL se puede traducir como AL = AX/BL y AH = AX%BL
- IDIV CL se puede traducir como AL = AX/CL y AH = AX%CL
- DIV byte[ESI] se puede traducir como AL = AX/Mem[ESI] y AH = AX%Mem[ESI]

División de 16 bits (DIV e IDIV)

- Coloca el resulado de la division de EAX y el modulo de la operacion en EDX.
- DIV EBX se puede traducir como EAX = EDX:EAX/EBX
- DIV ECX se puede traducir como EAX = EDX:EAX/ECX
- DIV dword[ESI] se puede traducir como EAX=EDX:EAX/Mem[ESI]

Instrucciones Lógicas

OR Se activa cuando al menos una variable está activa.

```
;; Ejemplo de OR

    MOV AL, 10001010b
    OR AL, 01001010b            ;11001010b

;; 10001010b
;; 01001010b
;; ----------
;; 11001010b
```

AND Solo se activa cuando ambas variables estan activas

```
;; Ejemplo de AND

    MOV AL, 10001010b
    AND AL, 01001010b            ;00001010b

;; 10001010b
;; 01001010b
;; ----------
;; 00001010b
```

XOR Se activa cuando solo una variables está activa

```
;; Ejemplo de AND

    MOV AL, 10001010b
    XOR AL, 01001010b            ;11000000b

;; 10001010b
;; 01001010b
;; ----------
;; 11000000b
```

Tablas de verdad

VAR X	VAR Y	OR	AND	XOR
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	0

Corrimientos

Las instrucciones de corrimiento posicionan o mueven números a la izquierda o a la derecha dentro de un registro o localidad de memoria, excepto los registros de segmento.

Instrucción	Descripción	Valor En El Carry
`SHL`	shift a la izquierda	El valor mas a la izquierda
`SHR`	shift a la derecha	El valor mas a la Derecha

Corrimientos

Las instrucciones de corrimiento posicionan o mueven números a la izquierda o a la derecha dentro de un registro o localidad de memoria, excepto los registros de segmento.

Instrucción	Descripción	Valor En El Carry
`SHL`	shift a la izquierda	El valor mas a la izquierda
`SHR`	shift a la derecha	El valor mas a la Derecha

Control de Programa

`JMP` Salto incondicional.

`LOOP` decrementa el valor de `CX` hasta llegar a 0.

`LOOPE` Salta si `CX` es diferente de cero mientras una condición de igual existe (Z=1).

`LOOPNE` Salta si CX es diferente de cero mientras una condición de no igual exista (Z=0).

`CALL` Realiza un salto hacia un procedimiento. Guarda en la pila la dirección de retorno.

`RET` Realiza un salto hacia la dirección de retorno, sacándola de la pila.

`CMP` Realiza una resta entre los operandos sin modificarlos.

Instrucción	Banderas que comprueba
JC	C = 1
JNC	C = 0
JZ, JE	Z = 1
JNZ, JNE	Z = 0
JS	S = 1
JNS	S = 0
JO	O = 1
JNO	O = 0
JP, JPE	P = 1
JNP, JPO	P = 0

Comparaciones

Números sin signo

Instrucción	Banderas que comprueban	Descripción
JA, JNBE	CF = 0 y ZF = 0	Salta si es mayor
JAE, JNB,JNC	CF = 0	Salta si es mayor o igual
JB, JNAE, JC	CF = 1	Salta si es menor
JBE, JNA	CF = 1 ó ZF = 1	Salta si es menor o igual

Números con signo

Instrucción	Banderas que comprueba	Descripción
JG, JNLE	SF = OF y ZF = 0	Salta si es mayor
JGE	SF = OF	Salta si es mayor o igual
JL, JNGE	SF OF	Salta si es menor
JLE, JNG	SF OF ó ZF = 1	Salta si es menor o igual