

OC: grupo 1/16/16

decimal: 0-16 = todos son sist. posiciones

Computadora ejecutar programas Almacenados en memoria en binario → Es lo + fácil de hacer. N° enteros sin/sin signo - Reales con signo - Decimales identificar

3/7 → Tarea prioridad 5 codificadas en binario (BCD) - caracteres

4/7 → Evolox fp. Sin signo, módulo y signo, (x_3) complemento a la base restringida)

5/8 → Evolox teoría (por proxim.) (x_2) complemento a la base) - exceso

4/8 → Recuperatorio fp.

18/8 → Recuadro binario → N° enteros sin signo de 0 a + Rango de 0 a $2^n - 1$

Ejemplo n=8 bits 0 00000000

128 10000000

255 11111111

$$\text{Teorema Fundamental de la Numeración} \quad N = \sum_{i=0}^n (d_i)_{10} \times (base)^i$$

$$x_4 \times B^4 + x_3 \times B^3 + x_2 \times B^2 + x_1 \times B^1 + x_0 \times B^0 + x_{-1} B^{-1} + \dots$$

Ejemplo: base 10

$$3574: 3000 + 500 + 70 + 4$$

$$3 \times 10^3 + 5 \times 10^2 + 7 \times 10^1 + 4 \times 10^0$$

3 unidades de mil + 5 centenas + 7 decenas + 4 u.

$$3.1416: 3 \times 10^0 + 1 \times 10^{-1} + 4 \times 10^{-2} + 1 \times 10^{-3} + 6 \times 10^{-4}$$

3 u. + 1 décima + 4 centésimas + 1 milésima + 6 diezmilésimas

Números en punto fijo: se considera q todos los números a representar

tienen exacta la misma cant de dígitos y la cosa
esta ubicada en el mismo lugar

Ej. bolígrafo

Rango: de donde hasta donde puede ubicar 1/los dígitos q tienen
Resol: df q 2 no consecutivos.

$$\text{Ej.: } 0.00 \text{ a } 9.99 \text{ Resolv: } 0.01 \quad 9.99 - 9.98 = 0.01 \quad \{$$

NOTA

Operaciones de las aritméticas \rightarrow Rec. 200 x la ALU

- Suma en binario: hoy acordemos las posiciones Ej: 111:10 (1 y me llevó 1)
 - Resta: 0-0:0 1-0:1 1-1:0 0-1:el de la izquierda me presto 2.

Bits de condic (Banderas) Bits q el procesador establece acorde al rito cada q realizada.

- Permite tomar decisiones como: hacer o no una transferencia de control, determinar sobre el no. Salir adicionales, nos lleva a una parte del código situada Z: vale si el estado + la operación son falsos P.

C(carry): en la suma vote 1 si hay acarreo en el bit + significativo en la resta vote 1 si hay "borrar" hacia el bit, significativo mirando < al sustituendo

$$\begin{array}{r} \cancel{F_1} \\ + \frac{1001}{1010} \\ \hline \underline{\cancel{0011}} \\ \text{Flag crossed} \end{array} \quad \begin{array}{r} 1001 \\ - \frac{1100}{\cancel{1010}} \\ \hline \cancel{1010} \\ \text{Flag carry} \end{array}$$

Sistema hexadecimal, ₁₆: Base 16

Dig, los: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}

$$\begin{aligned}Ej: \quad 2CA_8_{16} &= 2 \times 16^2 + C \times 16^1 + A \times 16^0 + B \times 16^{-1} \\&= 512 + 192 + 10 + 0,6 \\&= 714,6_{10}\end{aligned}$$

$$(2 \leftarrow 16 = 2^4 - 16 \rightarrow 4 \text{ b.t.s.} \quad \underline{10} \quad \underline{2})$$

0

si me faltó, lo pongo en los grupos de 4	1	0001	1
	2	0010	2
	3	1100	3
	4	0100	4
	5	1010	5
	6	0110	6
	7	1110	7
	8	0001	8
	9	1001	9
	A	1010	10
	B	1101	11
	C	1100	12
	D	1101	13
	E	1100	14
	F	1111	15

卷之三

P.45.
431,754.00
451
-296
155

BCD F/S preferencia: los nº se codifican usando un byte por dígito. Se dice q' el nº está desempaqueado. Decimal En cálculo, se reservan 4 bits x dígito. Se dice q' el nº está empaqueado codificado en binario Sin embargo: el empaq. y separar el nº en col.

Sin signo: p/ emisor y receptor c/ <> reflejo

as 4 $^{\circ}$ la diferencia al digito

Ejemplo: 934: 11110000 11110011 11110100

- F8 F3 F4

Con signo: +834 = 1111000 11110011 110001000
FB F3 C4 T

$$C = \text{Sign} + 110.1$$

$$D = \text{sign} -$$

Los 4 bits q' acompañan al último dígito son reemplazados x el signo

$$-894 = 399,900$$

Empaque todo el sabor.

$$+834 = \underline{10000011} \quad 01001100 \quad 834$$

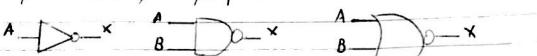
-31: 00000011 00000010 11110000

$$\begin{array}{r}
 \text{Suma en BCD} \quad 41 \quad 0100\ 0001 \\
 + 22 \quad \underline{\quad} \quad \underline{0010\ 0010} \\
 \hline
 63 \quad 0110\ 0011
 \end{array}$$

Nivel de lógica digital

- Un circuito digital es en el q' están presentes dos valores lógicos.
- Componentes son dispositivos electrónicos q' pueden realizar funciones q' estos 2 valores lógicos.
- Componentes básicos: AND, OR, NOT, NAND, NOR, XOR

Componentes simbolo y descripción funcional.



NOT	NAND	NOR
A X	A B X	A B X
0 1	0 0 1	0 0 1
1 0	0 1 0	0 1 0
	1 0 1	1 0 0
	1 1 0	1 1 0



AND	OR
A B X	A B X
0 0 0	0 0 0
0 1 0	0 1 1
1 0 0	1 0 1
1 1 1	1 1 1

XOR XNOR

- 2 entradas = salida 0 & salida 1
- " <> salida 1 <> salida 0



Algebra de Boole: leyes matemáticas q' gobiernan el comportamiento de las operaciones el. bits y describen circuitos

De Morgan: $\bar{A} \cdot \bar{B} = \bar{A} + \bar{B}$ / $\bar{A} + \bar{B} = \bar{A} \cdot \bar{B}$

$$\bar{\bar{A}} = A$$

Identidad: $1 \cdot A = A$ / $0 + A = A$

Nulla: $0 \cdot 0 = 0$ / $0 + 0 = 0$

Idempotencia: $A \cdot A = A$ / $A + A = A$

Inversa: $A \cdot \bar{A} = 0$ / $A + \bar{A} = 1$

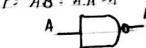
(Comutativa) $A \cdot B = B \cdot A$ / $A + B = B + A$

Asociativa: $(A \cdot B) \cdot C = A \cdot (B \cdot C)$ / $(A + B) + C = A + (B + C)$

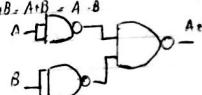
Distributiva: $A + B \cdot C = (A + B) \cdot (A + C)$ / $A \cdot (B + C) = AB + AC$

Absorción: $A \cdot (A + B) = A$ / $A + A \cdot B = A$.

Ejemplo: • construir un NOT q' NAND $F = \bar{A} \cdot \bar{\bar{A}} = \bar{A}$



• Construir OR con NAND $F = A + B = \bar{\bar{A}} \cdot \bar{\bar{B}} + \bar{A} \cdot \bar{B}$



• Escribir la tabla de verdad p/ la fr

• Dibujar una AND p/ cada término q' tiene 1 en la columna de salida.

• Invertir E necesaria

• Unir todas las AND a una OR

Recoratorios

- En una AND, basta q: una de sus variables de E sea 0 p/q: lo fijo.
- En una OR,
-

Punto ①

1- Representar en el sistema BCS o B bits

1- 1000 0001	35.000 0011	No pude representar -253, 256, -1, +8+
2- 1000 0001	35.000 0011	-127, +128, -199, -256, -100, 0.5, +1.25
3- 1000 1000	x9 excede los 8 bits o es negativo	0.00000000

2- Interpretar cadenas de 8 bits en BCS

0000 0000	1000 0000	1111 0000	1111 1111	1111 1110	0111 1110	1111 1111	0000 0000
0000 0000	1000 0001	1111 0001	1111 1110	1111 1111	0111 1111	1111 1111	0000 0001

3- 11111111 1111 00000000 Range: 0.00000000

4- Interpretar con el punto 3

10.000.00000	10.000.00000	10.000.00000	X	10.000.00000	10.000.00000	10.000.00000	10.000.00000
11.111111	11.111111	11.111111	0	11.111111	11.111111	11.111111	0

5- 0

0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000

BCS Col. Col2: n+1

- Técnica de complemento: Complemento de N dígs: N-A
- Complemento a N de (N-A): N-(N-A)=A

Caso ② → Número d signo →

- Representar en BCS. Con n bits, 1 bit representa al signo y n-1 bits.
- Modulo y signo.

$$\begin{array}{c} \text{Ej: } 10010 = +2^4 \\ 1010 = -2 \\ \hline \end{array}$$

- El bit extremo izquierdo representa solo el signo.
- Los bits 0 a n-2 lo magnitud
- 0: positivo 1: negativo
- Range $-(2^{n-1}-1) \rightarrow +(2^{n-1}-1)$
- Doble representación del 0. $\rightarrow 1000$ y 0000

• Complemento a 1 de m: $2^{n-1} - m$

• Complemento a 2 de m: $2^n - m$

• Representar en Col2: 0+1-

$$\begin{array}{c} \text{Ej: } +32: 0010 0000 \\ -32: 1101 1111 \\ \hline \end{array}$$

$$\begin{array}{c} +7: 0000 0111 \\ -7: 1111 1000 \\ \hline \end{array}$$

$$\begin{array}{c} n: 8 \text{ bits} \quad N^+ = \left\{ \begin{array}{l} 0000 0000 \\ 0000 0001 \\ \vdots \\ 1111 1111 \end{array} \right\} \\ N^- = \left\{ \begin{array}{l} 1111 1111 \\ 1111 1110 \\ \vdots \\ 0000 0000 \end{array} \right\} \end{array}$$

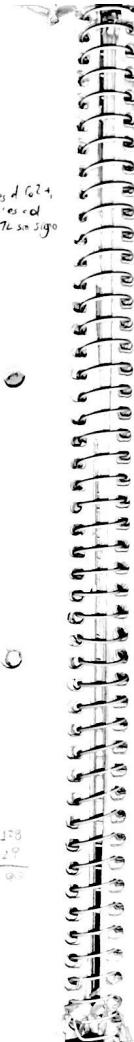
• Dada una cadena de bits, ¿que valor representaría si lo interpreto en Col2?

$$+0110 0000 = 1 \times 2^6 + 1 \times 2^5 = 64 + 32 = 96$$

→ Ambas formas

$$\begin{array}{c} \text{Col2 del } n^{\text{a}} \text{ y obtengo el } + = 1110 0000 = -32 \\ 0001 1111 + 32 \end{array}$$

NOTA



• antes de un dígito no signo, signo decimal

• Copiar de representación: n° q se puede representar

Banderas aritméticas: N: si el bit + signo/fractionario 0+1.D.N.

V: Overflow - condición fuera de rango. En 102 representando el dato es

Close 3 → Representación de n° en Pto Flotante = Punto fijo + el signo

Todos los m° se representan tienen lo = cant. de dígitos y la coma flotante está siempre en el mismo lugar

Surgir de la necesidad de representar la dif. e/ la representación en el papel y en la PC, es q no se guarda cosa alguna, se supone q esto en un lugar fijo.

M° fuera de rango = 31.975 -> 31.975.7100f
representación Punto dif. e/ n° > y <

A) q. n° Absolut. q 2 no consecutivos → Libre dato por el no - significativo

Frac de pto fijo: El m° entre comillas en una representación puede considerarse como la mitad de la diferencia (real) q 2 ° cercanas

z.S. B = E → Exponente

Base implícita

Mínus

5.01 5.05 5.02 5.03 5.04 5.05 5.06 5.07 5.08 5.09 5.10

5.015 < n° < 5.015 se representa x 5.015
5.015 < n° < 5.02 x representado < 5.02

En cada uno de los 2 nos el frac Absoluto m° < resultado se

FA m° = 5.015 - 5.01 = 0.005 o (5.02 - 5.01)/2 = 0.005

Nº en plt. flotante. En plt. fijo un rango de 10^{-1} y resultados centrados en 0.

Siguiendo en nro el componente decimalario, en este formato de plt. fijo trab. puede representarse:

Límites de nro muy grandes y muy pequeños

permite tener un rango de nro muy pequeños o grandes y ser representados por pocos dígitos.

↓

Sólo dígitos No es necesario almacenar.

↓ ↓ ↓

Exponente

↓

-976 000 000 000 = 9,76 × 10⁹

↓

↓ ↓ ↓

9,76

↓

Número

± M × 10^E

M y F representan a BSS, BCS, CO₂, CO₁, E_{CO} y
apartir, mantener,

Sympatros of sg formato o pto flotante

$$\begin{array}{ll}
 \text{Mantisa} & \left\{ \begin{array}{l} \text{BSS} \\ 4 \text{ bits} \\ \text{entero} \end{array} \right. \\
 & \left. \begin{array}{l} \text{Exponente} \\ \left\{ \begin{array}{l} \text{BSS} \\ 4 \text{ bits} \\ \text{entero} \end{array} \right. \end{array} \right. \\
 \text{Rango: } & 15 \times 2^{15} \xrightarrow{\text{gran rango}} \\
 \text{Resolviendo: } & \text{extremo sup: } [0, \dots, 15 \times 2^{15}] \text{ [0, ..., 49152]} \\
 & \text{inf: } R = (1-0) \cdot 2^{15} \\
 & R = (15-14) \cdot 2^{15} \xrightarrow{\text{1}} 2^{15}
 \end{array}$$

• Montiso y exponente en C2

Mantisa (c)	Exponente (c)
4.616	4.616
Entero	Entero
4.616×10^4	
Mínimo: $0.111 \times 2^{011} = +3 \cdot 2^7$	
Máximo: $1.000 \times 2^{011} = -8 \cdot 2^{27}$	
Respu. $= [-8 \cdot 2^7, +7 \cdot 2^7]$	
Resuelve el extremo superior: $R: 7.6 \times 2^7 = 1 \times 2^{28}$	
" en el origen: $R: (1 \times 2^8) \cdot 0 = 1 \times 2^8$	

• *Mentha franciscana* todos los bits son fraccionarios

Marken $\left\{ \begin{array}{l} BCS \\ 23 \text{ bits} \\ \text{fraktionär} \\ 1 \text{ bit signo} \end{array} \right.$ Exponent $\left\{ \begin{array}{l} \pm 2 \\ 8 \text{ bits} \\ \pm 1 \end{array} \right.$

* Resdux semper *

$$\text{Máximo positivo } 0,111\ldots111 \times 2^{0,111\ldots111} = + (1 - 2^{-2^3}) \cdot 2^{12}$$

$$f^* \text{Min}_D + (f_0) = 0 \cdot 0.000\ldots 001 \cdot 2^{1000000} = + (2^{-23}) \cdot 2^{-12}$$

$$\text{Maximum negative } (-1) + 0.000\ldots001 \times 2^{1000000} = (-1)^{2^{-13}} \cdot 2^{-28}$$

Formato final Sigue, responde, B 9 Motiva 31

$$\text{Normalzationsfaktor: } 40 \cdot 10^3 = 4 \cdot 10^4 : 0,4 \cdot 10^2 = 400 \cdot 10^{-1}$$

↓ Existe \Leftrightarrow valores de matriza y exponente p representan un mismo n°
 $\pm 0,1666 \cdot 6 \cdot 2^k$ / el obj. de tener un的印象 de valores de matriza y exponente p
 un número se reduce la **exactitud**. → solo p/facilidades.

• Y el objeto menor, los montajes formarán se definir como
binarios de 8 bits. Es decir, 0,1,0,1,0,1,0,1 es un dígito binario que vale 0,0,1

Ejemplo

Monteira	BCS	$\text{Exponente } f_{\text{Kresso}}$	$\text{Max. } + 0,111 \dots .111 \times 2^{-1} = + (1,2) \cdot 2^{-1}$
	23 bits	f_{Bobs}	$\text{Max. } + 0,100 \dots .000 \times 2^{0000000} = (0,5) \cdot 2^{228}$
	Françomoro	f_{Fabro}	$\text{Max. negativo } - 0,10000 \times 2^{000000} = -(0,5) \cdot 2^{-100}$
	16 bit signo		$\text{Min. } - 0,11111 \dots .111 \times 2^{-1} = -(1,2) \cdot 2^{-23} = 2^{-23}$
	Numeros reais		$\text{Range } F(12 \cdot 2^{-23}) \cdot 2^{23} = -(0,5) \cdot 2^{-128} \rightarrow F(-0,5) \cdot 2^{-128} + (2^{23} \cdot 1)$

El resultado: Si no lo deseas, puedes adicionar un bit en la notación.

El bit no almacenado se lo añade como bit implícito

$$\frac{1.10000...10}{(1 \cdot 2^{-2})} 2^{23} - 0.52 \cdot 0.5 \cdot 2^{23} (1 \cdot 2^{-2}) 2^{23}$$

Ejemplo:

$$\frac{1.0000...00}{(1 \cdot 2^{-2})} 2^{23} - 0.52 \cdot 0.5 \cdot 2^{23} (1 \cdot 2^{-2}) 2^{23}$$

Resumen:

$$1.0000...00 \times 2^{23} - 0.52 \cdot 0.5 \cdot 2^{23} = 1.0000...00 \times 2^{23}$$

$$= 1.0000...00 \times 2^{23} + 2^{23} \times 0.52 = 1.0000...00 \times 2^{23} + 0.52 \times 2^{23}$$

* El efecto matemático es el bit significativo bit significativo $\cdot 2^{\text{exponente menor}}$

* Resuelve el efecto inferior bit $-0.5 \cdot 2^{\text{exponente menor}}$

¿Cómo se escribe un N° en punto flotante normalizado?

1. Se escribe el N° en el sistema de la notación

2. Se desplaza la coma y se cambia el exponente

3. Se convierte al exponente del sistema de la notación

-13,5

$$1.1101100...0 \cdot 1.101100...0 \times 2^0$$

$$2. 1.0.110110...0 \times 2^4$$

$$3. 4 en C02: 00000000$$

en exceso: 10000000 sin

$$1.100000100...1001100...00 \text{ Impleto}$$

$$1.100000100...1001100...00 \text{ Bit implícito}$$

Error absoluto: FA muestreos resuelve f2. Muestra lo más cerca.

Error relativo: FA (número o representación) - f2 cerca de f2

f2 balance paráctico / biseguiente

> valor sistemático

> valor > / error ej. 0.5 of $1/2^2$

$1/2^2 = 0.5 = FA$

Frac infat: 1

FA / 2.5 = 1

convierte ambos sistemas de acuerdo p/ uno representar

Flage uno sistema

Estandar IEEE 754: Notación científica normalizada, ej. como desp del primer bit

Expresión representada en exceso $2^{23}-1$

Sistema 32 bits: Doble precisión 64 bits

Bits en signo	1	1
en exponente	8	11
En fracción	23	52
Total	32	64
Exponente en exceso	127	1023
Rango en exponente	$-126 \dots +127$	$-1023 \dots +1023$
Rango de N°	$2^{-126} \dots 2^{127}$	$2^{-1023} \dots 2^{1023}$

Combinación de bits pt cosa exponente

Simple

doble

$$38600000 = 0.00111111111111111111111111111111$$

en exceso $127 = 0$

$$1.0 \times 2^0 = 1$$

$$00066666 = 0.00000000000000000000000000000000$$

00000000 = 128 en exceso $127 = 1$

$$-1.05 \times 2^1 = -2.4$$

NOTA

Cases especiales

1. Exponente: 255 o 2047, M<0 \Rightarrow NoN "Not a number"
2. E.p.: 255/2047, M>0 \Rightarrow zero
3. E.p.: 0 M<0 \Rightarrow zero
4. E.p.: 0 M<0 \Rightarrow Desnormalizado: $n = \text{signo de } p \text{ y } q = \text{exponente}$
 $\pm 0, m = s \cdot p^{2^{-16}} / \text{mantisa } a$

Operaciones aritméticas en pto flotante

- 1º solo igualar los exponentes \rightarrow Hacer coincidir el p/exponente en ambos.
- Suma y resta Cuando los exponentes son \neq , se resta el menor y se da el resultado. M es $M_1 \cdot 10^e + M_2 \cdot 10^{e-m}$.

Multiplicar y dividir: Suma y resta exponentes:

Multiplicar y dividir mantisas y exponetes

Normalizar

Redondear

Todos los dígitos intermedios se borran. Se guarda el resultado p/ almacenar.

$$x = x_s \cdot B^{x_e} \quad y = y_s \cdot B^{y_e}$$

$$x \cdot y = (x_s \cdot y_s) \cdot B^{x_e + y_e}$$

$$x - y = (x_s - y_s) \cdot B^{x_e}$$

$$x \cdot y = (x_s \cdot y_s) \cdot B^{x_e + y_e}$$

$$\frac{x}{y} = \left(\frac{x_s}{y_s} \right) \cdot B^{x_e - y_e}$$

Cojunto de portas para conectar una sola salida f a la salida de las portas de ese instante.

La salida de las portas viene seguida de una etapa de inversedor.

Clase 4: circuitos lógicos combinacionales y secuenciales:

- Repetir o voltear logros en E.
- Si cambia la E, cambia S \Rightarrow f se invierte.
- Los vol. ordenados de las E, cambian la f, pero no su signo.
- No influyen los v. de S.
- Se representa mediante tabla de verdad o diagrama de flujo de puertas.

Tiempo de respuesta de la primera f.

Circuito multiplicador de 8E/4d: tiene muchas f y una sola S.

P/selección E, tiene 3 líneas de selección o selector de datos.

Ej.: televíscos, selecciona el v. de E p/

ALU, hace operaciones

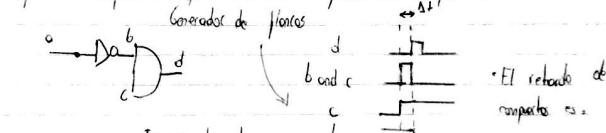
S se hace + y - pedo
hace el resto de op.

Internet: computadoras d/ acceso de banda.

Simultáneamente al mismo IR y se distribuye p/ los dispositivos.

• Existe tab. multiplexor en tiempo.

Repetir temporalmente g. tardan en responder los competidores.



Hace en da el resultado de competidor.

P/ cada una señal q' sea un pulso de para diox q'.

Si asciendan o suben un reloj p/ otros circuitos.

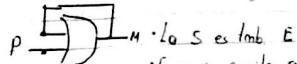
NOTA: El resultado de competidor es una señal que asciende o sube cuando se activa un reloj.

•PL: atmósfera y procesos
Se acceden circuitos separados de registros se
establecen anterior.

As. surgen los circuitos secundarios

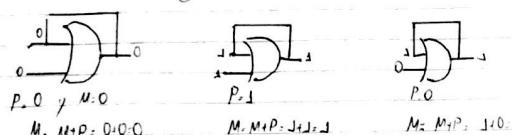
- Dependen de su E exterior y E interno del recto
 - Se separan de E (Son las q. forman)

Are we almost in the logic?

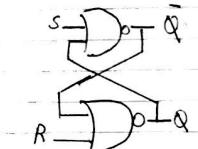


En ningún circuito combinacional una S transmite info. a una E
Lo decimos porque $M = M + P$.

Emplois



FLIP-FLOP $S\bar{A}$ \rightarrow S non- c and S c \rightarrow S complement c



<u>SR</u>	<u>Q_{n+1}</u>	} n: instante actual, n+1: próximo s/actual
0 0	Q _n	
0 1	0	S. Set = poner 0 →
1 1	1	R = Reset = "0"
1 0	Prohibido.	Q y Q complementarias
↓		
PQ Q / Q̄ = 1		

Los secuenciales se pueden clasificar en:

Asternutritas: el cambio de la S, cruce al cambio de la E + S anterior
Sincretorios: la S no cambia en el momento, sino cambian a partir de una señal de reloj.

Reloj: "serial especial"
-Temporiza eventos del sist.

-emporiza eventos del SIST

T largo: frecuencia baja } Ciclo de tipo: q' /
 T corto frecuencia alta. } del periodo lo ciclo
 ciclo de tipo esto alto o bajo.

FLIP FLOP $D =$
 $\downarrow s \quad R = \bar{D}$

FLIP FLOP JK.

FLIP FLOP T \oplus combina de 0 a 1
o 0 0 on cada pulso
de la E de T

Recordando un bit: plasmando un bit en el registro temporal que usa un Flip-flop D

Selección y operaciones: identificación y celdas

Contador módulo 8 3 FLIP FLOP 3. contador módulo 8

Cuadro combinacional

Entrada \rightarrow Cuadro combinacional \rightarrow salida

Cuadro lógico

Entrada \rightarrow Cuadro combinacional \rightarrow Salida

AND

Memoria

P Busco q sepa 2 salidas

	S	R	Q	Q'
Ajunto	1	0	1	0
Memoria	1	1	Memorizar, vale el E y sale	
Limpio	0	1	0	1
Memoria	1	1	1	0
	0	0	Indefinible	

OR

	S	R	Q	Q'
Set	0	1	1	0
Memoria	0	0	Vale anterior	
Resetar J	0			
Memoria 0	0			
Alarma 1	0		Indefinible	

Flip flop con reloj \rightarrow Pulso del reloj:
Pulso bajo=0, alto=1

(1) forcede el flip-flop
depende del reloj.

Pulso alto=1

OR

	Q	S	R	Q'
	0	0	0	0
	0	0	1	0
	0	1	0	1
	0	1	1	Indefinible
	1	0	0	1
	1	0	1	0
	1	1	0	
	1	1	1	0

FLIP FLOP D \rightarrow modo de entrada



D	Q(t+1)
0	0
0	1
1	1
1	0
1	1

FLIP FLOP JK \rightarrow reset



J, K	Q(t+1)
0, 0	0
0, 1	1
1, 0	1
1, 1	1
1, 0	0
1, 1	0
1, 1	1
1, 1	0

FLIP FLOP T \rightarrow toggle lo cambia



T	Q(t+1)
0, 0	0
0, 1	1
1, 0	0
1, 1	1
1, 1	0

0 \rightarrow memoriza

1 \rightarrow complementa

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

1 \rightarrow 1

1 \rightarrow 0

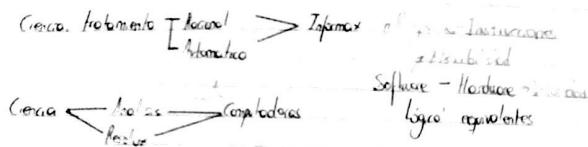
0 \rightarrow 1

1 \rightarrow 1

0 \rightarrow 0

Clase 5

Información automática



Computadora es una máquina (con inteligencia propia), digital (que opera en el mundo digital).
Sincronizada (que controlada + señales), controla número y lógica (puede modificar D de E),
controlada por un programa (permite q la propia máquina opere de manera autónoma) q/
se almacena en una memoria. Y tiene comunicaciones del mundo exterior.

Arquitectura y organizaciones

Arquitectura → atributos visibles al programador. Ej: instrucciones, no bits codados pl/datos, mecanismos I/S, técnicas de discriminamiento.

Organizaciones → como son implementados esos atributos. Ej: señales de control, interfaz, tipo de memoria.

Ej: procesador 2937 (el que controla la computadora)

Toda la familia Intel x86 comparte la misma arquitectura básica

• Busco compatibilidad de diseño
• Soportar nuevas plataformas / func.
• El procesador anterior
• Lo q giga difiere en las versiones.

• Procesador grande para aplicaciones más complejas

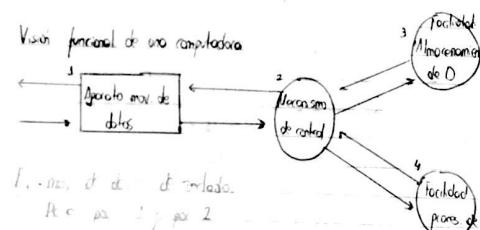
• Procesador pequeño para aplicaciones más simples.

• 30% de los nuevos procs.

• 10% de los nuevos procs.

Estudio y fx: Estudio → modo en el cual se estudian los componentes q se incluyen → fx → grupo de los componentes individuales como parte de la estructura.

- Procesamiento de D. Muy poca - bajo precio.
- Almacenamiento de D e información en una memoria.
- Movimiento de D de una sra cap. S de D al espacio.
- Control y coordinación.



1. Procesamiento de D.

2. Movimiento de D.

3. Control de D.

4. Entrada y salida de datos.

Algo pa 1 y 2

Algo pa 3 y 4

Algo pa 5

Algo pa 6

Algo pa 7

Algo pa 8

Algo pa 9

Algo pa 10

Algo pa 11

Algo pa 12

Algo pa 13

Algo pa 14

Algo pa 15

Algo pa 16

Algo pa 17

Algo pa 18

Algo pa 19

Algo pa 20

Algo pa 21

Algo pa 22

Algo pa 23

Algo pa 24

Algo pa 25

Algo pa 26

Algo pa 27

Algo pa 28

Algo pa 29

Algo pa 30

Algo pa 31

Algo pa 32

Algo pa 33

Algo pa 34

Algo pa 35

Algo pa 36

Algo pa 37

Algo pa 38

Algo pa 39

Algo pa 40

Algo pa 41

Algo pa 42

Algo pa 43

Algo pa 44

Algo pa 45

Algo pa 46

Algo pa 47

Algo pa 48

Algo pa 49

Algo pa 50

Algo pa 51

Algo pa 52

Algo pa 53

Algo pa 54

Algo pa 55

Algo pa 56

Algo pa 57

Algo pa 58

Algo pa 59

Algo pa 60

Algo pa 61

Algo pa 62

Algo pa 63

Algo pa 64

Algo pa 65

Algo pa 66

Algo pa 67

Algo pa 68

Algo pa 69

Algo pa 70

Algo pa 71

Algo pa 72

Algo pa 73

Algo pa 74

Algo pa 75

Algo pa 76

Algo pa 77

Algo pa 78

Algo pa 79

Algo pa 80

Algo pa 81

Algo pa 82

Algo pa 83

Algo pa 84

Algo pa 85

Algo pa 86

Algo pa 87

Algo pa 88

Algo pa 89

Algo pa 90

Algo pa 91

Algo pa 92

Algo pa 93

Algo pa 94

Algo pa 95

Algo pa 96

Algo pa 97

Algo pa 98

Algo pa 99

Algo pa 100

Algo pa 101

Algo pa 102

Algo pa 103

Algo pa 104

Algo pa 105

Algo pa 106

Algo pa 107

Algo pa 108

Algo pa 109

Algo pa 110

Algo pa 111

Algo pa 112

Algo pa 113

Algo pa 114

Algo pa 115

Algo pa 116

Algo pa 117

Algo pa 118

Algo pa 119

Algo pa 120

Algo pa 121

Algo pa 122

Algo pa 123

Algo pa 124

Algo pa 125

Algo pa 126

Algo pa 127

Algo pa 128

Algo pa 129

Algo pa 130

Algo pa 131

Algo pa 132

Algo pa 133

Algo pa 134

Algo pa 135

Algo pa 136

Algo pa 137

Algo pa 138

Algo pa 139

Algo pa 140

Algo pa 141

Algo pa 142

Algo pa 143

Algo pa 144

Algo pa 145

Algo pa 146

Algo pa 147

Algo pa 148

Algo pa 149

Algo pa 150

Algo pa 151

Algo pa 152

Algo pa 153

Algo pa 154

Algo pa 155

Algo pa 156

Algo pa 157

Algo pa 158

Algo pa 159

Algo pa 160

Algo pa 161

Algo pa 162

Algo pa 163

Algo pa 164

Algo pa 165

Algo pa 166

Algo pa 167

Algo pa 168

Algo pa 169

Algo pa 170

Algo pa 171

Algo pa 172

Algo pa 173

Algo pa 174

Algo pa 175

Algo pa 176

Algo pa 177

Algo pa 178

Algo pa 179

Algo pa 180

Algo pa 181

Algo pa 182

Algo pa 183

Algo pa 184

Algo pa 185

Algo pa 186

Algo pa 187

Algo pa 188

Algo pa 189

Algo pa 190

Algo pa 191

Algo pa 192

Algo pa 193

Algo pa 194

Algo pa 195

Algo pa 196

Algo pa 197

Algo pa 198

Algo pa 199

Algo pa 200

Algo pa 201

Algo pa 202

Algo pa 203

Algo pa 204

Algo pa 205

Algo pa 206

Algo pa 207

Algo pa 208

Algo pa 209

Algo pa 210

Algo pa 211

Algo pa 212

Algo pa 213

Algo pa 214

Algo pa 215

Algo pa 216

Algo pa 217

Algo pa 218

Algo pa 219

Algo pa 220

Algo pa 221

Algo pa 222

Algo pa 223

Algo pa 224

Algo pa 225

Algo pa 226

Algo pa 227

Algo pa 228

Algo pa 229

Algo pa 230

Algo pa 231

Algo pa 232

Algo pa 233

Algo pa 234

Algo pa 235

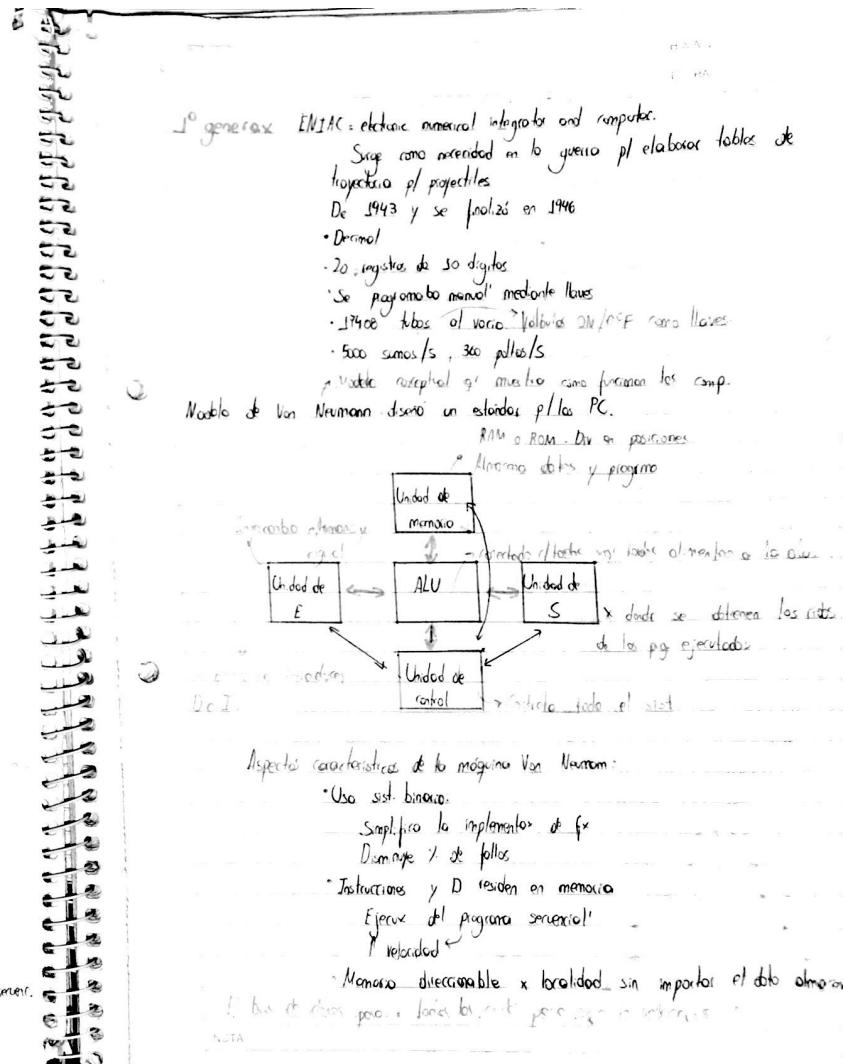
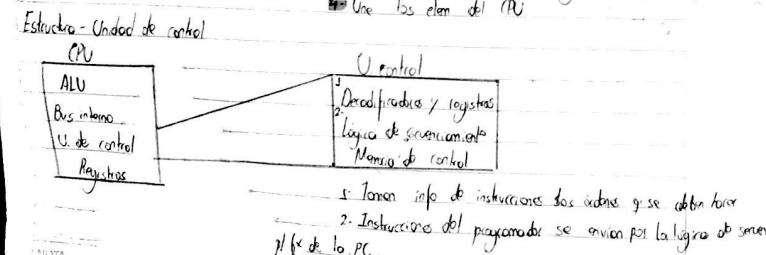
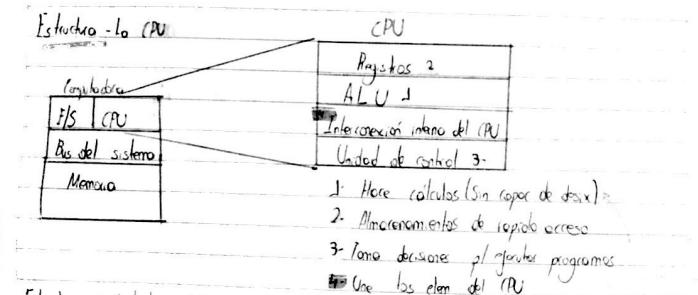
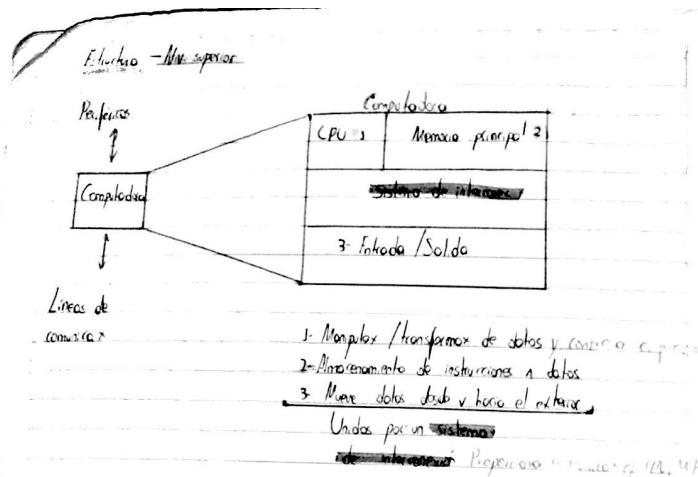
Algo pa 236

Algo pa 237

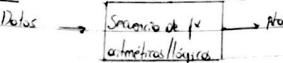
Algo pa 238

Algo pa 239

Algo pa 240

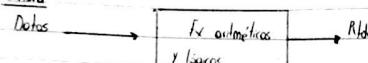


Concepto de programa en los

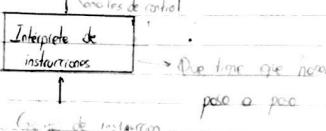


- Programar en hardware: cuando tienes que cambiar las formas, debes cambiar el hardware

Ahora



- Programas de software: en software se efectúa alguna operación sobre los datos



Caja de instrucciones

- Si el paso se necesita un nuevo conjunto de señales de control
- Las instrucciones proporcionan esas señales de control
- Aplicar el nuevo concepto de programación → cambia la forma pero no te das cuenta
- No hay q. cambiar hardware

(Qué es un programa): Serie de pasos.

- Se hace operaciones aritméticas/lógicas y cada paso
- → señales de control se generan pt. de operaciones.
- La UC sera info de instrucciones.

DATA

IAS: Institute of Advanced Study • Memoria / 4096 palabras de 40 bits

• N° C2

• 2 inst de 20 bits

• Set de registros

• Buffer de memoria (MBR)

• direcciones de memoria (MAR) Dirección memoria

Controles 8 bits de control de operaciones

• Instrucción y buffer de instrucciones

• Dirección de la memoria para ejecutar instrucciones + controlador de programa (program counter) PC

• Memoria operativa u RAM temporaria de 16Kb dividida en bloques de 1024x1024 bits temporales de lectura y escritura AC HQ

UNIVAC I: Universal Automatic Computer

• 1º PC comercial

• Puede usar un compilador pt. traducir idioma de programación a máquina

• Directorio de 12 dígitos x palabra

• Principal特色: set de bobinas magnéticas q se giran hacia otras o adelante + procedimiento de comprobación de errores

• Memoria de tareas de trabajo de memoria y tareas o volúmenes de trabajo

LBM • Lenguaje de programación de bajo nivel:

• 1953: 701 → 1º PC q programó almacenados de IBM

Aplicaciones científicas

• 1955: 702 → aplicaciones de gestión

• 1º serie de compiladores: 700/1000

2º generación: transistores

- Cambian los tubos al vero

- más pequeñas y bocas

- menos calor

- Dispositivos de 5 soldadura

- Tamaño log. mejorado de 1 mil.

3º generación: circuitos integrados

- Tamaño o pequeño escala

- + de los componentes en un chip

- a media escala

- 100-3000 componentes

- gran escala

- 3000-10000 componentes

- very gran escala

- hasta 1000 millones de componentes

Serie IBM 600

- Instrucciones similares

- E/S similares

- > velocidad

- NO 1º de E/S paralelos

- > memoria

- > precio

DEC PDP-8

- 1º minicomputador

- No necesitaba oscilador coordinado

- App monostable y ROM

- Estructura de bus + bocinas

Algo por los computadores; se fue desarrollando la fabricación de memorias

Memoria Semiconductora

- 1970

- 1º memoria 1/256 bits

- Capacidad duplicada x año.

- Lectura no destructiva

Microprocesadores: Intel

- 1971 - 4004 - 1º microprocesador de 4 bits

- Todo lo del CPU en un solo chip

- 1974 - 8080 - 1º microprocesador de uso general

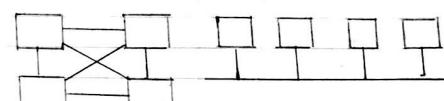
Intercanvier de en sist de computo separare

constituido por 3 subsistemas

• CPU
• Memoria
• E/S

Los componentes deben comunicarse el/ si

El concepto de buses tiene canales independientes y de rápido comunicar



• Canales independientes el/ <>
dispositivos

• Comunicación mediante un medio compartido.

Bus: • Camino de comunicar q' conecta 1 o + dispositivos (local, "broadcast"). Negra
posicionamiento.

Bus serie: 1 bit por vez

+ arriba el bus = + info q' viaja

Bus paralelo: conjunto en serie

• Transporta datos

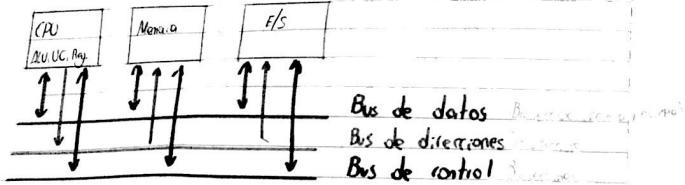
• No hay diferencia q' dato o instrucción

• El ancho es un valor determinado de las prestaciones

NOTA

• 8, 16, 32, 64 bits

Interconexión de los sistemas:



- Direcciones:
- Si el bus es compartido por > elementos, estos deben tener identidad > direcciones
 - La dirección de memoria identifica una celula de memoria en lo q almacene info
 - La lectura y escritura se plantean respecto de la PU

Bus de direcciones: Identifica el origen/destino de los datos

- La PU necesita tener una instrucción de dato abierto
- El ancho de bus determina la máxima capacidad de memoria del sistema

Ej: 8 bits tiene bus de 8 bits dando un espacio de dirección de $2^8 = 256$ posiciones de memoria

Ej: 16 bits tiene bus de 16 bits dando un espacio de dirección de $2^{16} = 65536$ posiciones de memoria

El tamaño de datos refleja veces abierto x el bus de datos

Bus de control: Info de control y temporizado

- Sentidos de escritura/lectura de memoria o E/S
- Sentidos de selección o habilitar
- Sentidos de reloj (clock)
- Sentidos de parada o interrupc.

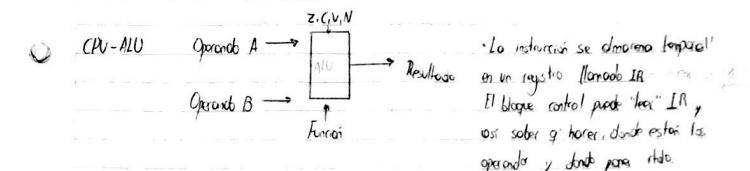
Componentes del hardware dedicado a fx

- E/ - Teclado
- Monitor
- Mouse
- Impresora
- Joystick

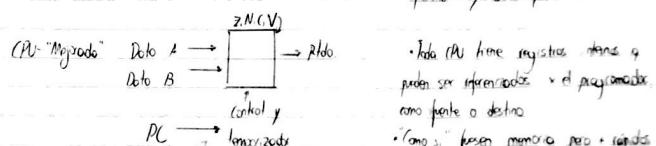
Almacenamiento - Memoria

Disco (reg. de discos)

Cintas (CD, DVD)

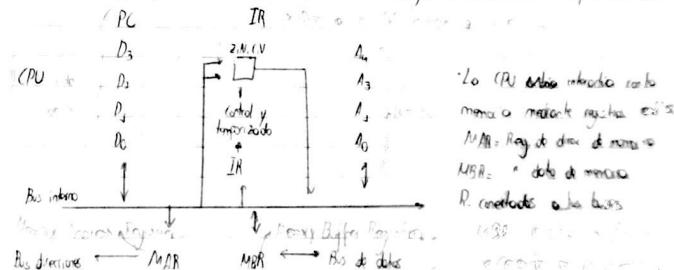


• La instrucción se almacenó temporalmente en un registro llamado IR - $IR \rightarrow R$
El bloque control puede leer IR y los sobre q hacer, donde estén los operando y dest para rta.



Toda PU tiene registros internos q pueden ser referenciados x el programador como fuente o destino

Como si fueran memoria pero + rápida
Llamado al almacenamiento temporal



• La PU tiene memoria central memoria mediante registros externos MAR: Registro direc de memoria MBR: dato de memoria P: controlador sobre buses

• MBR → D₀, D₁, D₂, D₃ → Bus de datos

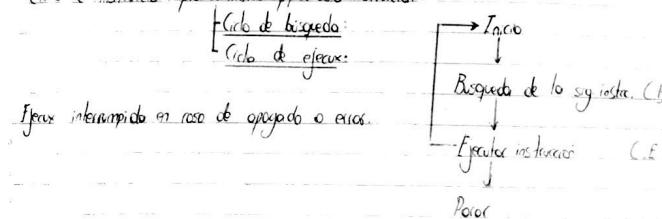
• Bus de control → MDR → Bus de control

• Tanto todo esto x pl/ almacenar direcc. pl/ > flexibilidad

Clase 6. Ciclo de instrucción → surge de la ejecución de programas

- El procesamiento de instrucciones se divide en 2 etapas q' produce D de S
- Buscado: lee dato memoria
 - Cada q' las instrucciones
 - Ejecución: depende de la instrucción
- Ejecución: depende de la instrucción
puede implicar varias instrucciones

Ciclo de instrucción: procesamiento p/ 1 sola instrucción



CICLO DE BÚSQUEDA y EJECUCIÓN

1. Al principio de todo ciclo, la CPU busca una instrucción en memoria
2. En la CPU hay un registro, llamado PC, q' tiene la dirección de la próxima instrucción a buscar
3. La CPU: dep. de buscar la instrucción, incrementa el valor del PC p/ buscar la sig. instrucción
4. La instrucción buscada se carga dentro de un registro de la CPU, llamado registro de instrucción (RI)
5. La instrucción está en la forma de un código binario q' especifica las órdenes q' tomará la CPU
6. La CPU interpreta cada instrucción y lleva a cabo los órdenes requeridos

DIFERENTES CLASES DE INSTRUCCIONES

- CPU-MEMORIA: Datos q' pueden transferirse el memoria y CPU
lectura
- CPU-E/S: datos q' pueden transferirse el CPU y E/S
- Procesamiento de datos: CPU efectúa operaciones aritméticas/lógicas en datos

Control: alterar la secuencia de ejecución de instrucciones.

• Se usa el control de memoria o bus.

- EJEMPLO
- Cargar en el registro D el contenido de la posición de memoria 940.10
 - Sumar el contenido de la posición de memoria 941.10 al registro D y guardar el resultado en D
 - Almacenar el valor del registro D en la posición 942.10

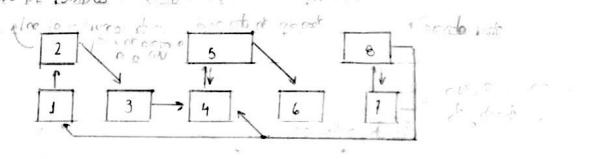
Consideremos q' el posix de memoria almacena 32 bits. Los 10 primeros bits indican lo q' se hace o realizar, los siguientes 22 bits es la dirección.

0001.1 → cargar D desde la memoria

0010.2 → almacenar D en memoria

0101.5 → sumar D / un dato de memoria

DIAGRAMA DE ESTADOS → "Ciclo de ejecución"



- Los E en la parte superior implican un intercambio c/ CPU y la memoria
- Los E de la parte inferior implican solo q' interc. en la CPU

LÉNGUAJE ASSEMBLY

No resulta confortante pensar y escribir programas como una secuencia de n° que es natural verlo como una secuencia de instrucciones. P/ ello se define un lenguaje q' es un anglo de esos códigos comprensibles x un humano. A esos lenguajes se lo denomina assembly o lenguaje de ensamblaje. El programa q' entienda esos códigos se llamará ensamblador.

Cierre 7 → formato de instrucción y modo de direccionamiento.

Elementos de una instrucción de máquina

- opcod /-Código de operación (especifica lo q' hace o realizar, es un código binario)
- Referencia del operando fuente, establece donde se encuentra el operando.
- Lo operando puede incluirse j/o + operando fuente (o de t.)
- Referencia al destino para establecer dónde almacenar el resultado.
- Referencia de la sig. instrucción: le dice al CPU dónde buscar la sig. instrucción desp. de la ejecución de la instrucción anterior. En la mayoría de los casos se ubica a continuación de la instrucción actual.

• Los operandos fuente y resto pueden estar en memoria, registro CPU o disp. de I/S.

• Representación de instrucciones:

- Dentro de la computadora cada instrucción está representada mediante una secuencia de bits, que se divide en campos en correspondencia a los elementos q' la componen.
- Este esquema se conoce como formato de la instrucción.
- Es difícil p/ el programador tratar el representación binaria, por lo tanto se usa una representación simbólica.
- Los códigos de operación se representan x medio de abreviaturas llamadas mnemónicas q' indican lo q' hacen.

FORMATO DE INSTRUCIONES ADD 12 M1

• Los operandos también pueden representarse simbólicamente

Ej: mem, reg, mem/... → variable q' indica q' opera. Reg: Operando fijo /... → Instrucción q' apunta al valor contenido en la posición de memoria llamado memoria, o en un registro denominado reg!

modo de operar, Referencia al operando, Ref. al operando

modo de acceso memoria

Ej: $x = x + y \rightarrow$ otro nivel lenguaje → Una instrucción q' nos regresa variables instrucciones de máquina. Expresa operaciones en forma concisa usando variables

• El lenguaje de máquina expresa las q' básicas usando mem. de datos y registros.

• Un lenguaje de alto nivel debe convertirse a un lenguaje de máquina. Un conjunto de instrucciones de máquina debe ser capaz de expresar cualquier instrucción de T. nivel

• Clasificación de las instrucciones

- Almacenamiento de datos → operaciones aritméticas y lógicas
- Almacenamiento de D → transferencia dentro del sistema
- Instrucciones E/S → transf. de D qf PC y mecanismos externos
- Control

Número de direcciones fija s/ la máquina

{ 2 direcciones p/ los operandos, uno q' almacena el valor y el q' la próxima instrucción
Add, Dic1, Dic2, DirOp, DirFct, DirReg, DirIns}

Máximo de 4 direcciones

• Muchos campos q' espacio y + largo

NOTA

MDD Inmediato Dirección Operando El operando se obtiene directamente de la memoria o mismo tiempo q la instrucción.

- No requiere una referencia extra o memoria de doble.
- Se usa para definir constantes y personalizar variables.
- Índice de memoria limitado x el tamaño del campo de desplazamiento
- Apuntar de r^o mareas (no q no se tiene q q' abra).

MDD Directo Dirección Operando Preguntar direc de memoria

Memoria • Espacio limitado de direc x razón de tamaños del campo.

• Uso: variable global; dirección constante al compilador.

MDD por registro Dirección Operando • = el directorio

Registro (RN) • Uso - bits.

• Punto registros y su resultado.

MDD por memoria Dirección Dirección Operando • 1/ una direc de bit en la instrucción.

• se apunta a una direc de bits

Memoria • = espacio de almacenamiento

• Multiples accesos a memoria

MDD indirecto por registro

Registro	Dirección	Operando
Registro RRU	Memoria	• bits p/ un registro q pase de memoria

MDD x desplazamiento R, A Dirección Operando

R	A	Dirección	Operando
Reg RRU	Reg RA	Memoria	• Cambio directo e indirecto
			3 modos de desplazamiento.

- Nota: registro referenciado = PC, la dirección q lo ind. se suma al campo de direc p/ producir dirección efectiva. El campo de direc se hace largo = 60 bits.
- Registro base: el registro referenciado contiene una dirección de memoria y el campo de direc tiene un desplazamiento.

Inverso: se direcciona la memoria q un registro + un desplazamiento.

- = q el autor para se intercambian partes del registro y desplazamiento.
- Proporciona un mecanismo eficiente p/ realizar q's iterativas.
- Se usa un registro llamado índice (se incrementa o decrementa s/ index).

MDD del stack → Apunta hacia el trío de trazabilidad de memoria. Lista abriendo el último en entrar es el último en salir. Zona de memoria reservada.

Asociado q la pila o stack, hoy un registro apuntador.

• Pueden gestionar las llamadas y roturas de procedimientos, y pueden contenerse entre una forma alternativa de direccional memoria. C/LD direcciones conjunto de posiciones last-in-first-out.

No de elem varia. Siempre se apunta q la cabecera de la pila.

El procedimiento se determina x las instrucciones q ejecutan llamadas instrucciones de máquina.

Al conjunto de instrucciones q: puede ejecutar el procedimiento se llama repertorio de instrucciones del procedimiento.

• Punto: direc base de la pila. El punto o dirección o incremento q si se crece o se resta un elemento.

Bucle: contiene la direc base del bloque reservado p/ la pila. Si se intenta un pop q la pila vacía, crece un error.

Límite de pila: direc del otro extremo del bloque reservado. Si se intenta un push q el bloque lleno, da error.

Calse B: Organización de registros e instrucciones

Organización de registros - Visibles al usuario vs. el programador

Primitiva memoria referencias a memoria principal gastos o
El registro viene de: R de control y estado usado p/ lo UC p/ controlar lo que entra en memoria y op de la CPU (no son visibles p/ el programador) y registro de control y tiempo. No hay separación clara.

R visibles al usuario \leftrightarrow tipos - Propósito: que el usuario para lo que quiere - puede contener operandos
• Son referencias x lenguaje de datos: solo se pueden saber datos
• Tamaño: direcciones fijas - dinámicas p/ q
• Puede haber restricciones: • Edges de control: c.d.c. (búferes)
• En los registros p/ el usuario.
Ej: p/ flotante/fijo etc. • Bits fijados p/ el resultado del procedimiento q. v. id. de una instrucción
• Se pueden usar p/ directamente o indirectamente
• Los registros de dirección pueden ser asignados p/ un mdd
• Si todos los registros se usan p/ propósito gen., pude afectar al funcionamiento de las instrucciones

Nº de registros: > 100 reg., + bits p/ especificar en la instrucción > para cada
- Afondo del funcionamiento de la instrucción
- poco reg = + referencias a memoria
- 16 y 32 registros (nuevo), + no hay gran mejora, > tamaño de instr.

Largo de los registros: - De dirección: ser capaz de almacenar Adr. direc + grande
- De datos: habilidad p/ almacenar la mayoría de tipos de D
- Algunos magníficos permiten 2 reg. contiguos utilizados como un solo reg. p/ almacenar vol. de doble longitud

Bits de radix: - Bits establecidos como club de garantías
Gral: no alterados x el programador, no de forma directa.

3) para hacer R controlar el R/S de la ALU e interconexión
dato y MBR \rightarrow y R.v. bloq

R de control y establecidos visibles) - Empleados p/ controlar lo op de la PU
Hoy: 4 escalones PC controlado de programador. Datos deben estar en control
Reg. visibles p/ {
• 1R: registro de instrucciones (entre las instrucciones y memoria)
función de datos q/ interfaz q/ MAR y D de memoria tiene q/ no p/ el controlador y memoria
• 2R: MBR: reg. de buffer de memoria q/ bus de datos
• 3R: controlador p/ controlar el D a escribir la memoria o readear
• 4R: memoria

Elos 4 reg. se emplean p/ manejar datos q/ el PU y memoria. Sub registrador

Organización de registros CPU P/T Intel

	CS	S	B	F	Control
A	AH ₁	AL	EAX		
B	BH ₁	BL	EBX		
C	CH ₁	CL	ECX		
D	DS ₁	DL	EDX		
E	ES ₁				
F	FS ₁				
G	GS ₁				

2) M: se regula el tamaño, cada uno 32 bits EIP PC y banderas

el código q/ q/ y genera la interconexión EFLAGS

datos q/ de memoria mediante MAR y MBR. MBR controla q/ bus de direcciones. MBR tiene q/ de datos

• Ax: acumuladores, es el principal q/ op aritméticas.
• Dx: punto base (de base).
• Cx: controlador, interviene en instrucciones de control.
• Dx: datos, participa en multiplicación y división.

• SI y PI: controladores q/ manejar los instrucciones arraigos o tablas. Registros de datos q/ resp con
• BP y SP: por los cuales p/ manejar el piso. Apuntan a zonas determinadas.
• F: extensión de A de 32 bits.

NOTA

Organización registros CPU Motorola 68000

	32	80
A0		
A1		
D2	Página de inicio	
D3	o cualquier dato	
A3		
D4		
A4		D5 8 bits de 32 bits
A5		
D6		
A6		D7
A7	Apalabra de start word	
PC	superior A7' 32 bits	8 bits inferior

De 256 bits

Por el bus Z

8 reg de 32 bits de datos
9 reg de direcciones
2 slots, 1 p/usuario, 1 p/uso

función de control de pines en 6 bits de uso interno

Instrucciones - Tercero: destinatario destino, fuente. Destino y fuente son 2 operando.

Intel - donde 1 de ellos está especificado x alguno de los molt usos, el otro operando es un reg de la CPU

Llamada: men: especificar de una dirección de memoria

reg: reg de la CPU

mem: dato inmediato.

1 operando se referencia usando un modo de direccionamiento

• Instrucción mem, reg : reg, reg : mem, imm : como parte

• Inst. reg, mem : reg, imm

El nombre destino y fuente proviene del hecho q si hay un mol. de datos, es decir la otra(fuente) hora lo ingresa (destino)

En una suma hoy 2 operandos y el res. se almacena en el lugar del operando requerido (destino)

DATA

No hace direccionamiento x memoria

Instrucciones - Intel 8086 - ADD AX,BX → AX = AX+BX • MOV AL, CH → AL=CH

• ADD AL, AH → AL = AL+AH • SUB AX,BX → AX = AX-BX

Direccionamiento por registro

• ADD AX,35AFh → AX = AX+35AFh • MOV AL,3EH → AL = 3EH 8 bits

modo: directo

• ADD AL,15+AL = AL+15 • SUB AX,1234h → AX = AX-1234h

Direccionamiento inmediato

grado de la salida

• ADD AX,[35AFh] → AX = AX+ contenido directo 35AFh , 3500h

• ADD AL,DATA → AL = AL + contenido variable DATA (8 bits)

• MOV CH,NUM1 → CH = contenido variable NUM1 (8 bits)

Direccionamiento directo

• Direc + bajo : significativa

8 bits

• ADD [BX],BX → AX = AX+ dato almacenado en dirección contenida en BX y lo

q sigue → p/ completar 16 bits.

• MOV [BX],AL → dato almacenado en BX:AL

Direccionamiento por registro(indirecto)

• MOV CX,[BX+SI] → CX = dato almacenado en la direc BX+SI y lo sig.

Estructura • MOV EBX+DI],AL → Dato almacenado en BX+DI = AL

Direccionamiento base + indice

• MOV AL,[BX+2] → AL = dato almacenado en dir BX+2

• MOV [BX+2Ah],AX → dato almacenado en dir BX+2Ah y lo q sigue = AX (16 bits)

Direccionamiento relativo por registro

• MOV AL,[BX+SI+2] → AL = dato almacenado en la dir BX+SI+2

• MOV [BX+SI+2Ah],AX → dato al " y lo q sigue = AX (16 bits)

Direccionamiento relativo base + indice

Formato de instrucción - número de dato: Instrucción consta o logica depende de si

16 bits/pal. = constante de fondo de memoria de datos

Velocidad procesador/velocidad memoria

Instrucción + constante + rapidez del procesador

Suficiente bits p/ expresar todos los q/ deseas:

La exp. demuestra dejar bits libres p/ el futuro.

Const de bits de datos

la memoria puede devolver en el caso de la tecla si

Ejemplo p/ MSX88 el procesador p/ leer datos res + rapidez q/ esq.

lectura en apilado. P/ solucionar se usa las memoria cache

Editor prueba com-Usar ed. de fotos.

Asamblea prueba.com-Usar Amiga(prueba o p/ prueba.st)

ORG 2000H

MOV BX,3000H

MOV AX,BX

Enlazar prueba o Usar LK88 + prueba.exe

Usa MSX88 - carga prueba.exe y ejecutar

ADD BX,[BX]

MOV CX,[BX]

ADD AX,CX

Dic. Carga magnética linea const. log. ensamblaje

2000 89 00 30 1 oxy 2000H

2001 89 01 2 MOV BX,2000H

2005 81 F3 02 00 3 MOV AX,[BX]

2009 8B OF 4 ADD AX,CX

200B 03 01 5 PUSH AX

200D 50 6 POP DX

200E 5A 7 HLT

200F F4 8

9

10

11 oxy 3000H

3000 55 33 44 22 12 db 55h,33h,44h,22h

13 end

Simbolos

nombre:

tipo: valor.

Caso 9 → subsistemas de memoria / sig mem. p/pal.

Memoria vel. procesador: duplica cada mes los inst. ejecutados / seg.

memoria: reduplica tiempo 1/36 meses.

Este genera desequilibrio entre procesador y memoria. P/ equilibrar esto biehacese con los tipos de memoria desde rápido a lento (registros) y lento a lento (discos).

Se intenta se aprovecha p/ un comportamiento equivalente al de una memoria única, rápida y grande (la real sería considerada ideal).

○ Jerarquía de memorias forma en g se organizan los tipos de memoria.



R + const., pequeña, + rápida y > frecuencia de acceso

& lento pero barato.

Puesta directa tamb. se usa el cache

Tipo

Registro

Tiempo

1ns

Tamaño

1MB

Caché

520ns

1MB

Memoria

60-80ns

1GB

Memoria del computador:

Velocidad > Fundamentos físicos > Localización >

Se refiere p/ lo rapido de almacenamiento.

Velocidad de acceso reducida.

Características

Dirac de la info.

Modo de direccionamiento.

Velocidad RAM

Acceso a palabra: mem. p/pal.

No volátil discos, raras.

"x" bloques discos rápidos

Permanentes: ROM, EEPROM

Velocidad

Memoria semiconductora

Tiempo acceso temporal

modo de operación alternado

relación tiempo memoria / instrucciones

+ ciclo > t. acceso

Memoria magnética

acceso: pax, tablas,

modo de acceso

V. transferencia bytes/seg.

Métodos de acceso:

- Absoluto: tiempo p/ acceder a una locación es independiente de la secuencia de accesos anteriores y es constante. Ej: memoria ppal.
- Secuencial: acceso en secuencia lógica específica.
- Directo: los bloques e registros tienen una dirección q se establece en la locación física.
- Asociativo: memoria cache

Memoria de oración sacerdotal

RAM (Random Access Memory): Almacena síntesis q se puede acceder a cualquier celda de memoria en el mismo tiempo, independiente de la posic en la estructura de memoria.

- SRAM: memoria estructura basada en flip flops.
 - DRAM: - Dinamico basada en transistores (capacitores) (el de q' el memoria 0 como una electricidad).

ROM: mismo tipo de acceso.

DRAM: olvidaría + info q' SRAM es lo mismo, superficie pequeña, liviana, lenta.

hoy q' reescritura + memoria u flash. Se refresca & tendencia natural a desaparecer.

SRAM: + rápido, usado como cache.

Organizar el plan básico de una presentación de seminarios sobre la cultura artística.

Los celdas de memoria de semiconductores contienen 3 o más nodos.

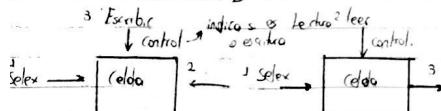
- Biestable: representa 1 y 0.
 - Se puede escribir en ellos al menos 1 vez

So greater force of reaction of E.
of addition force & force of the reaction force is equal

- Solox:** solerrina una colada de manzana

- Cont.

E/Lectura de D



Escritura de D



Lectura de D



Organización de la memoria

- Una memoria de 1 bit lo implementaremos con flip-flops y "formaremos" seguidas de n bits de ff .
 - Para construir memorias "grandes" se requiere una ay. \Rightarrow , en la real sea posible disconectar peldanes individuales.

Organizer del ch.p

- Cada chip contiene un arreglo de relojes de memoria.
 - En las memorias de semiconductor se han empleado 2 enfoques: 2D y 3D.

Original 2D

- El arreglo este organizado en 2^o polobios de 8 bits de 1/lineas horizontales (una de 2^o) se conecta a el posic de memoria, se leva un renglon.
 - Los buses verticales conectan el lat a los.
 - El decodificador q esta en el chip tiene 2^o salidas p/ w E. (bits del bus de direcciones).

Organic Zoox 2 1/2 D

- El orzuelo es "cuadrado" y funciona = q. 20
 - Los bits de una misma palabra están dispersos en <> chips.
 - Lo directo se divide en 2 partes. 1 setex de renglón y una setex de columna.
Hoy 2 directores.

Conspiracy

- En 2D todos los bits están en el mismo chip.
 - En 2-1/2D los bits de una misma palabra estarán en \gg chips.
 - 2D es lento, ∇^* grande de parámetros de picos bits. Clínica de selección de palabra tiene q tener un manejo duro y controlarse al decodificar. Cuppen mucha superficie
 - 2D dificulta el uso eficaz de los circuitos correctores de errores. El 2-1/2D d. tener los bits dispersos en \gg chips hoy \leq probabilidad de error

En 2560, los filos y colores indican la complejidad de los subcomponentes

Problemas 1) /módulo tiene el espacio de direccionamiento requerido, pero solo cubre una parte de la palabra.

Solución: varios módulos en paralelo.

2) La longitud de la palabra es lo deseado, pero los módulos no tienen la respuesta.

Solución: cubrir un campo de direcciones de módulos de memoria "en serie".

1 módulo estaria en dirección \leftrightarrow

Nuevas técnicas RAM

• La DRAM es la misma desde los 1º chips de RAM

• Enhanced DRAM (tamaño pequeño SRAM)

La SRAM guarda la última linea leída (como una cache).

• Cache DRAM (cache una SRAM + grande)

Se usa la SRAM como cache o como buffer serial.

• Synchronous DRAM (SDRAM) p/ compensar la latencia de RAM

Adelantos en DRAMs. Acceso sincronizado al reloj externo. Se presenta una linea a la RAM. RAM encierra D y no expone la DRAM.

SDRAM maneja D neto en tiempo del reloj.

CPU puede hacer otra cosa mientras opera.

Bus de permito al SDRAM trabajar en bloques.

Capa 4: memoria interna.

4 = dirección

32 bits

32 bits

2 bits

ATA
SATA

Práctica 5 - Lenguaje Assembly

Procedimientos:

• Interpretar 1 y 0

• Resetea op. de elementos como suma, resta,

• Copiar octetos p/ el tamano de D

Uso de D: Dificil recordar el código binario p/ d/ instrucción

Requiere resolver op. compuestas.

Uso D de tamaño variable y grande.

Lenguaje assembly: Aprende sintaxis recordando lo que "migra" mediante mnemónicos.

• Implementa E de control \rightarrow Q, Iyrics + saltos condicionales y E/S.

• Uso D de tamano o \rightarrow Modos de direccionamiento

una palabra \leftrightarrow teclear

Instrucciones: codigos binarios q' dicen q' hace el q' tiene sus inst.

q' inst \leftrightarrow representación binaria (muchas inst. cortas + largas)

q' modo de direccionamiento \rightarrow + modos = códigos de inst + largos.

tmb. es una inst \leftrightarrow

Sx88 \rightarrow p/lo practica \rightarrow 4 registros: de 32 bits: Ax, Bx, Cx, Dx

A su vez pueden dividirse en 8 bits: AH, AL, BH, BL, etc.

Tamano de inst. variable (multiplos de 8 bits)

de palabra de 32 bits

Bus de D de 8 bits de ancho (2 relojes p/ recuperar una palabra)

Direc de 32 bits (2^{32} : 4GB de direcciones + 1 byte: 64KB de memoria)

Modo de d/c conmutante. Inmediato-Directo (registro)-Indirecto (memoria)-Indir x reg

Tener en cto: • Línea ENTRADA final del activo si o si

• Nombre de archivo máximo 8 caracteres

Estructura de un programa 1º instrucciones basadas en directo programando (2000 h)

Las d se dividen a partir de la direc. 1000h

Otra instr. a partir de q diras se dividen las sig. instr. y declaras

Las d e inst. declaradas en cualquier posic., siempre q abriremos en 2000h la 1º instr. del programa.

Flags 8 banderas en total

Solo afectado a op. aritméticas y op. lógicas de la ALU

Al resto de op. no afectan a los flags

JMP altera los flags tras restar/líz operando pero no almacena d ellos.

Señal importants el Z,S,O,C.

Transferencia de control Sig. inst. alteran el flujo normal del programa modificando el sig. IP si → condiciones

Instrucción cond. x operador

JZ F Z=true JNZ

JS F S,ng=true JNS

JO F overflow=true JNO

JC Flag carry=true JNC

JUP Incond. normal No opera

CPU 8088 tiene su propio ensamblador ASME8.

Plantea el problema, leección M388 <cas>

Instrucciones

• Inst. transf. de D: NOP

• cálculos lógicos: Inst. cálculos ADD, ADC, SUB, SBB

Lógicas AND, OR, XOR, NEG, NOT

• comparar CMP

• incremento / decremento INC, DEC

• cambio d flags de programa Sellos condicionales JZ, JNZ, JS, JNS, JC, JNC, JO, JNO

Sellos condicionales JMP

ejecutar a startine CALL, RET

• manejo d pila PUSH, POP, PUSHF, POPF

• " gestión d los interrupciones: INT, IRET, STI, CLI

• de control NOP, HLT

• E/S: IN, OUT

W: width tamaño de operandos W: 0 → 8 bits DB → 1 byte / 8 bits

W: 1 → 16 bits DW → word → 16 bits

ID = dirección de la próxima instr. a ejecutar

SP = dirección del topo de la pila

Instrucciones de salto

JZ salto si el flag Z=1 JNZ salto si Z=0; ~ngno

JS " S:1 JNS " :D: → ngno

JC " C:1 JNC " C:0; ~ngno

JO " O:1 JNO " O:0; ~omflow

JMP dirección ; salto siempre

Select if then else

JF AL=4 then CMP AL,4 then MOV BL,1

begin JZ then INC CL

BL:1; JMP Else

CL=CL+1; JMP Fin_IF

end Else: MOV BL,2

DFC CL

Fin - JF: HLT