

RELACIONES.pdf



charlieangel



Arquitectura de Computadores



2º Grado en Ingeniería Informática



Escuela Politécnica Superior de Córdoba
Universidad de Córdoba



[Accede al documento original](#)



Escuela de
Organización
Industrial

Contigo que evoluciones.
Contigo que lideras. Contigo que transformas.

**Esto es EOI.
Mismo propósito,
nueva energía.**



Descubre más aquí



EOI Escuela de
Organización
Industrial

Importante

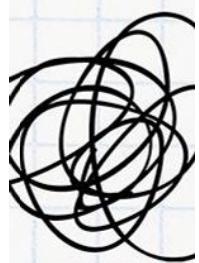
Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

pierdo
espacio



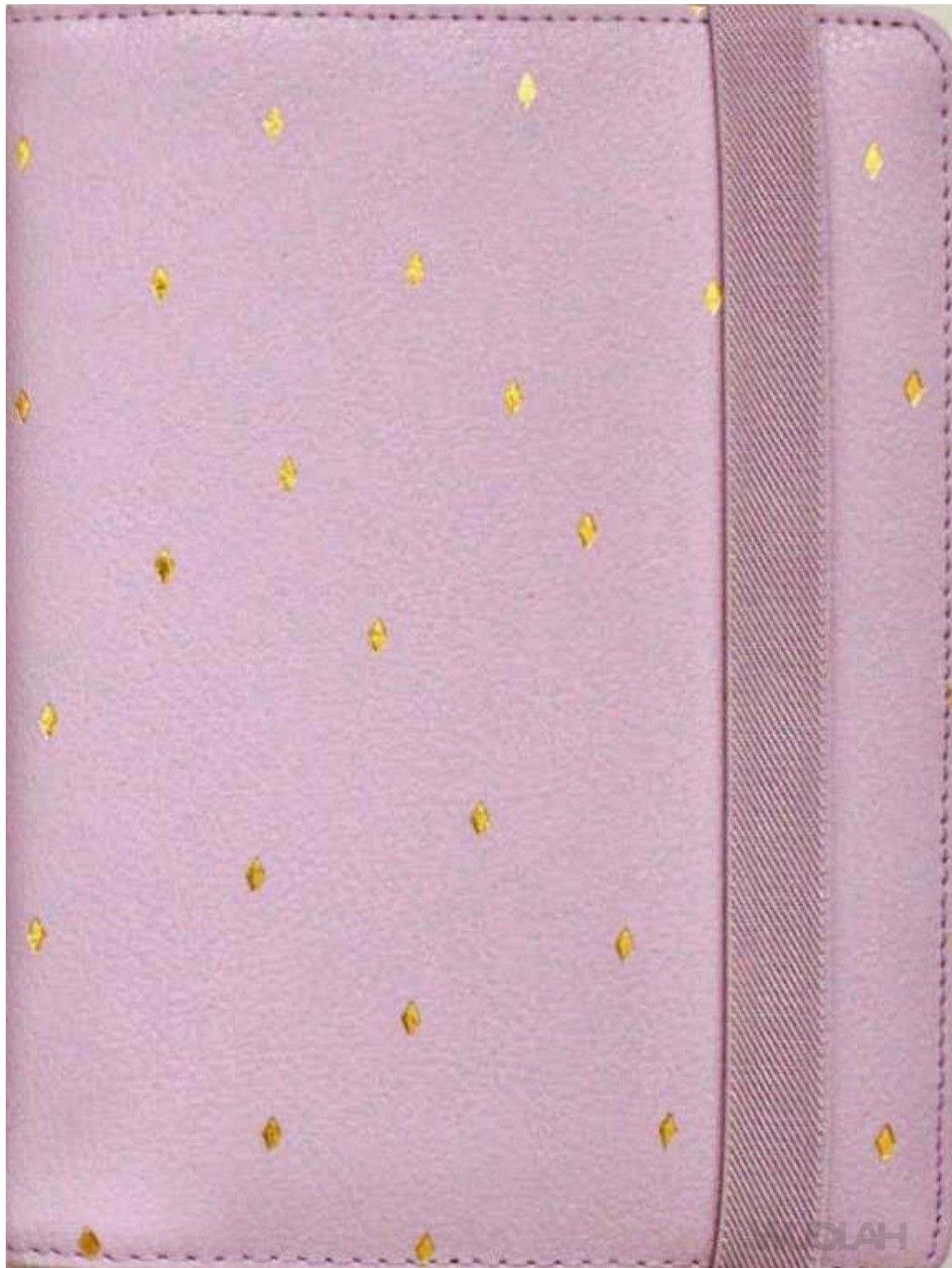
(=)



Necesito
concentración

ali ali oooh
esto con 1 coin me
lo quito yo...

wuaAH
XXXXX



COMPONENTES PRINCIPALES

1. PC (Program Counter)

- Guarda la dirección de la siguiente instrucción a ejecutar.
- Se puede incrementar automáticamente ($PC+1 \rightarrow PC$) o cargar con una dirección ($GPR \rightarrow PC$).

2. MAR (Memory Address Register)

- Contiene la dirección de memoria a la que se quiere acceder.
- Se carga desde el PC, SP o GPR según la operación.

3. SP (Stack Pointer)

- Maneja la pila.
- Se incrementa ($SP+1 \rightarrow SP$) o decrementa ($SP-1 \rightarrow SP$) cuando se hacen operaciones de pila.

4. OPR (Operando) $GPR(COP) \rightarrow OPR$

- Almacena el código de operación actual, que indica qué tipo de instrucción ejecutar.

5. Controlador (CONTROLADOR)

$SC-1 \rightarrow SC$
LOAD SC

- Interpreta el código de operación y genera las señales de control para el resto de los bloques.
- Recibe el código desde OPR y activa el flujo adecuado.

6. GPR/BR (General Purpose Register / Buffer Register)

MEM → GPR
ACC → GPR
PC → GPR
GPR+1 → GPR
QR → GPR
GPR+1 → GPR

- Registro intermedio entre otros bloques y la memoria.
- Puede recibir datos desde la memoria, el acumulador (ACC), PC, o SP.
- También puede escribir hacia la memoria.

7. MEMORIA

- Almacena datos e instrucciones.
- Puede recibir datos desde GPR o QR, y enviar datos hacia GPR.

ALU (Unidad Aritmético Lógica)

Contiene:

- Complementador:** Para operaciones de complemento a uno o a dos.
- SUM (Sumador):** Realiza operaciones aritméticas básicas.
- F:** Registro de banderas (overflow, negativo, cero, etc.).
- ACC (Acumulador):** Almacena resultados intermedios.
- QR:** Registro temporal para operaciones entre registros.
- Operaciones soportadas (algunas):**
 - SHR F A-Q: Desplazamiento a la derecha.
 - SHL F A-Q: Desplazamiento a la izquierda.
 - ROR, ROL: Rotaciones.
 - MEM → QR: Carga de datos desde memoria.

○ Acc → GPR: Transferencia de datos.

○ QR → MEM: Guardado de datos.

STATUS REGISTER

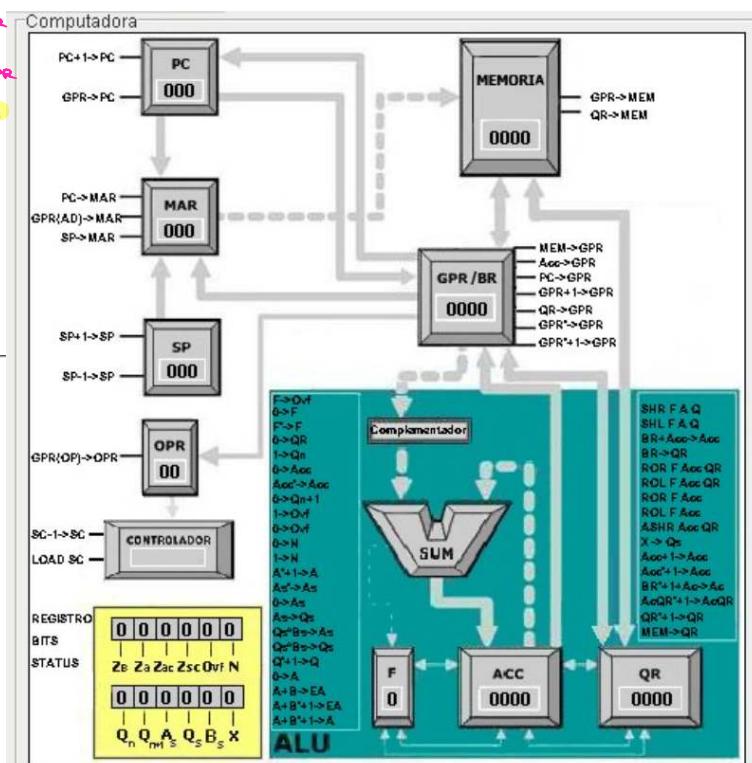
Contiene banderas que informan del resultado de operaciones:

- Zc: cero
- Ov: overflow
- N: negativo

Estas ayudan a tomar decisiones lógicas (ej. saltos condicionales).

FLUJO GENERAL DE UNA INSTRUCCIÓN

- PC indica la dirección de la siguiente instrucción.
- Se carga en MAR.
- Se accede a la MEMORIA y se carga la instrucción.
- OPR extrae el código de operación.
- El CONTROLADOR interpreta y activa la operación.
- Los operandos se mueven (por ejemplo, de memoria a ACC).
- La ALU ejecuta la operación.
- El resultado va al ACC y/o GPR, y puede actualizarse MEMORIA.



Operaciones de ALU

OPERACIONES LÓGICAS Y DE DESPLAZAMIENTO

1. SHR F A Q

- Shift Right (desplazamiento a la derecha): Mueve todos los bits del acumulador (A) una posición a la derecha.
- El bit que sale puede ir a F (flags), y podría venir un bit desde Q.

2. SHL F A Q

- Shift Left (desplazamiento a la izquierda): Lo contrario de SHR.
- Se mueven los bits de A a la izquierda, y el bit que sale puede ir a F.

TRANSFERENCIAS Y ARITMÉTICA

3. BR + Acc → Acc

- Suma el contenido del buffer register (BR) con el acumulador (Acc), y guarda el resultado en el acumulador.

4. BR → QR

- Copia el contenido de BR (registro de propósito general) hacia QR (registro auxiliar para la ALU).

ROTACIONES Y DESPLAZAMIENTOS

5. ROR F Acc QR

- Rotate Right: Rota los bits de Acc hacia la derecha, posiblemente usando QR como entrada/salida y F para la bandera.

6. ROL F Acc QR

- Rotate Left: Igual que arriba, pero hacia la izquierda.

7. ROR F Acc

- Rotación del acumulador sin QR.

8. ROL F Acc

- Rotación del acumulador hacia la izquierda sin QR.

9. ASHR Acc QR

- Arithmetic Shift Right: Desplazamiento aritmético (conserva el signo) de Acc, posiblemente afectando QR.

OTROS OPERADORES

10. X → Qs

- Carga el contenido de algún registro (probablemente externo o inmediato) a Qs, que puede ser parte del estado o parte del QR.

11. Acc + 1 → Acc

- Incrementa el acumulador en 1.

12. Acc - 1 → Acc

- Decrementa el acumulador en 1.

13. BR + 1 + Acc → Acc

- Suma el contenido de BR, más 1, más Acc, y guarda el resultado en Acc.

14. Acc QR + 1 → Acc QR

- Suma Acc con QR y 1, y guarda el resultado en ambos registros.

15. QR + 1 → QR

- Incrementa el QR en 1.

16. MEM → QR

- Carga un dato desde memoria hacia el registro QR.

Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

tema 1:

①

1: INC m incrementar contenido en m

GPR(m) → MAR apunto al dato

MEM → GPR manda el dato al GPR

GPR + 1 → GPR incrementar dato

GPR → MEM devuelvo el dato a memoria

2: DEC m decrementar contenido en m

GPR(m) → MAR ; 0 → ACC apunto al dato , ACC = 0

MEM → GPR ; ACC → ACC obtengo el dato de memoria, 0 a $C_2 = -1$

GPR + ACC → ACC al sumar obtengo $x - 1$

ACC → GPR resultado ($x - 1$) a GPR

GPR → MEM

3: LDA m cargar en ACC lo contenido en m

GPR(m) → MAR ; 0 → ACC

MEM → GPR

GPR + ACC → ACC

4: JPC m saltar a la dirección PC + m

0 → ACC limpio el acumulador

GPR + ACC → ACC GPR tiene m, lo manda a ACC

PC → GPR(CAD) PC es mandado a GPR

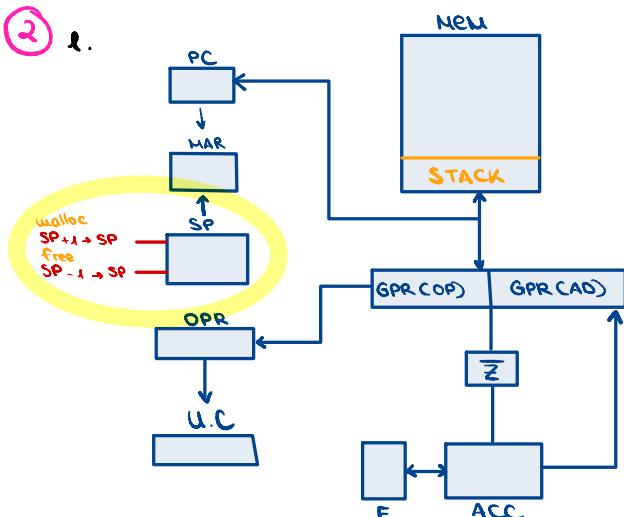
GPR + ACC → ACC GPR + ACC(m) se carga en el ACC (PC + 1)

ACC → GPR (PC + m) se manda a GPR

GPR(CAD) → PC Se indica que lo guardado en GPR es por donde tiene

que seguir PC

②



SP solo tiene conexión con MAR y recorre la pila

CSR m } 1. PC apila

2. PC ← m

SP - 1 → SP

SP → MAR; GPR(CAD) → PC; PC → GPR(CAD)

Solo puede hacerse con GPR y PC

GPR → MEM

RET m

SP → MAR

MEM → GPR; SP + 1 → SP

GPR(CAD) → PC

pierdo espacio



Necesito concentración

ali ali ooooh
esto con 1 coin me
lo quito yo...

wuolah

(3)

PSH introducir en la pila contenido ACC

$SP - 1 \rightarrow SP$; $ACC \rightarrow GPR$ Al introducir tam SP decremente, lo que hay en ACC se pasa a GPR

$SP \rightarrow MAR$ la dirección a la que debe apuntar MAR es la de la Pila

$GPR \rightarrow NEM$ en esa dirección se almacena lo pasado por GPR

POP sacar de la pila el contenido y almacenar en ACC

$SP \rightarrow MAR$; $O \rightarrow ACC$ indica a MAR la última posición

$SP + 1 \rightarrow SP$; $NEM \rightarrow GPR$ aumenta el espacio en SP, manda lo contenido en MAR a GPR

$GPR + ACC \rightarrow ACC$ lo guarda en ACC

tema 1. Repaso

1. INC m

$GPR(m) \rightarrow MAR$
 $NEM \rightarrow GPR$
 $GPR + 1 \rightarrow GPR$
 $GPR \rightarrow NEM$

DEC m

$GPR(m) \rightarrow MAR$; $O \rightarrow ACC$
 $NEM \rightarrow GPR$; $\overline{ACC} \rightarrow ACC$
 $GPR + ACC \rightarrow ACC$
 $ACC \rightarrow GPR$
 $GPR \rightarrow NEM$

LDA m

$GPR(m) \rightarrow MAR$; $O \rightarrow ACC$
 $NEM \rightarrow GPR$
 $GPR \rightarrow ACC$

JPC m

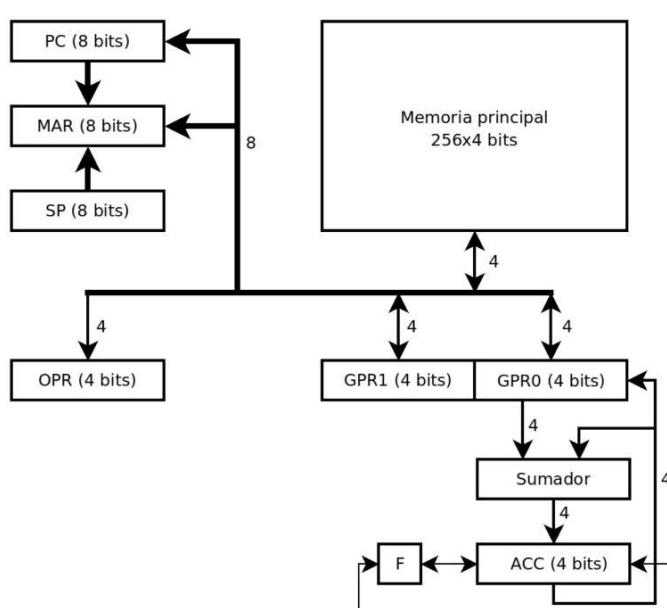
$O \rightarrow ACC$
 $GPR + ACC \rightarrow ACC$
 $PC \rightarrow GPR(m)$
 $GPR + ACC \rightarrow ACC$
 $ACC \rightarrow GPR$
 $GPR \rightarrow PC$

3. PSH

$SP - 1 \rightarrow SP$; $ACC \rightarrow GPR$
 $SP \rightarrow MAR$
 $GPR \rightarrow NEM$

POP

$SP \rightarrow MAR$; $O \rightarrow ACC$
 $SP + 1 \rightarrow SP$; $NEM \rightarrow GPR$
 $GPR + ACC \rightarrow ACC$

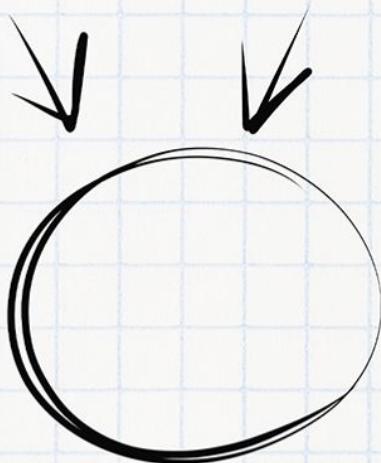


Imagínate aprobando el examen

Necesitas tiempo y concentración

| Planes | PLAN TURBO | PLAN PRO | PLAN PRO+ |
|---|--------------------------------|--------------|--------------|
| diamond Descargas sin publi al mes | 10 🟡 | 40 🟡 | 80 🟡 |
| clock Elimina el video entre descargas | ✓ | ✓ | ✓ |
| folder Descarga carpetas | ✗ | ✓ | ✓ |
| download Descarga archivos grandes | ✗ | ✓ | ✓ |
| circle Visualiza apuntes online sin publi | ✗ | ✓ | ✓ |
| glasses Elimina toda la publi web | ✗ | ✗ | ✓ |
| € Precios | Anual <input type="checkbox"/> | 0,99 € / mes | 3,99 € / mes |
| | | | 7,99 € / mes |

Ahora que puedes conseguirlo,
¿Qué nota vas a sacar?



WUOLAH

ciclo búsqueda

PC → MAR

NEM → GPR1 ; PC + 1 → PC

ADD m

1 PC → MAR

2 NEM → GPR1 ; PC + 1 → PC

3 PC → MAR

4 NEM → GPRO ; PC + 1 → PC

5 GPR → MAR

6 NEM → GPRO

7 ACC + GPRO → ACC

ADDI m

1 PC → MAR

2 NEM → GPR1 ; PC + 1 → PC

3 PC → MAR

4 NEM → GPRO ; PC + 1 → PC

5 GPR → MAR

6 NEM → GPR1 ; MAR + 1 → MAR

7 NEM → GPRO

8 GPR → MAR

9 NEM → GPRO

10 Acc + GPRO → ACC

STA m

1 GPR1 → MAR posiciona MAR en m

2 ACC → GPRO guarda en GPRO el contenido de ACC

3 GPR → NEM guarda en dicha dir lo de ACC

JMP m

1 NEM → GPR

2 GPR → PC

INC m

- 1 SP - 1 → SP ; ACC → GPRO reserva espacio en la pila , guardar en GPRO lo de ACC ya preservarlo
- 2 SP → MAR apunta nueva dir en la pila donde se almacenará ACC
- 3 GPRO → NEM Guarda en NEM lo que tenía ACC
- 4 PC → MAR ; O → ACC MAR tiene m
- 5 NEM → GPR1 ; PC + 1 → PC ; ACC + 1 → ACC Se carga desde memoria m y guardado en GPR1, ACC = 1
- 6 PC → MAR para leer contenido de m
- 7 NEM → GPRO ; PC + 1 → PC guarda lo de m en GPRO
- 8 GPR → MAR para escribir en memoria
- 9 GPRO + ACC → ACC se hace lo de m + 1 y guardar en ACC
- 10 ACC → GPRO res pasa a GPRO
- 11 GPRO → NEM res se guarda en NEM
- 12 SP → MAR ; O → ACC
- 13 NEM → GPRO ; SP + 1 → SP } para restaurar el estado original
- 14 GPRO + ACC → ACC

DEC m

1 SP - 1 → SP ; ACC → GPRO

2 SP → MAR

3 GPRO → NEM

4 PC → MAR ; O → ACC

5 NEM → GPR1 ; PC + 1 → PC ; ACC → ACC

6 PC → MAR

7 NEM → GPRO ; PC + 1 → PC

8 GPR → MAR

9 GPRO + ACC → ACC

10 ACC → GPRO

11 GPRO → NEM

12 SP → MAR ; O → ACC

13 NEM → GPRO ; SP + 1 → SP

14 GPRO + ACC → ACC

Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

pierdo
espacio



Necesito
concentración

ali ali ooooh
esto con 1 coin me
lo quito yo...

wuolah

STA m

- 1 PC → MAR leer la instrucción en memoria
- 2 MEM → GPR1 ; PC + 1 → PC lee contenido de m (instrucción) y guarda GPR1
- 3 PC → MAR vuelve a leer
- 4 MEM → GPRO ; PC + 1 → PC guarda en GPRO el contenido m
- 5 GPR → MAR lo de GPR es la nueva dir donde se almacenará ACC
- 6 ACC → GPRO GPRO almacena contenido en ACC
- 7 GPRO → MEM se guarda en memoria lo de ACC

LDA m

- 1 PC → MAR ; O → ACC
- 2 MEM → GPR1 ; PC + 1 → PC
- 3 PC → MAR
- 4 MEM → GPRO ; PC + 1 → PC
- 5 GPRO + ACC → ACC

PSH m

- 1 SP - 1 → SP
- 2 SP → MAR
- 3 ACC → GPRO
- 4 GPRO → MEM

POP m

- 1 SP → MAR
- 2 SP + 1 → SP ; O → ACC
- 3 MEM → GPRO
- 4 GPRO + ACC → ACC

Tema 3 y 4: unidad cálculo y unidad control

1.

$$\begin{array}{r} \text{s.m.} \\ 0+ | 1011101 \\ + 0+ | 011111 \\ \hline 1001100 \end{array}$$

overflow

cuando la operación es $+ + ó -- \Rightarrow$ O. ADITIVA y tiene AVF (overflow)
cuando la operación es $+ - ó - + \Rightarrow$ O. SUSTRATIVA y tiene E
Si $E=0$ res = C_2 res

Operación aditiva

2.

$$\begin{array}{r} \text{s.m.} \\ 0+ | 000011011 \\ - 1- | 0100111 \\ \hline 0011000 \end{array}$$

NO overflow

Operación aditiva

3.

$$\begin{array}{r} \text{s.m.} \\ 0+ | 1011011 \\ - 0+ | 0111111 \\ \hline C_2 1000001 \end{array}$$

Operación Sustractiva

4.

$$\begin{array}{r} \text{s.m.} \\ 1- | 110111011 \\ - 1- | 1011101 \\ \hline C_2 0100011 \end{array}$$

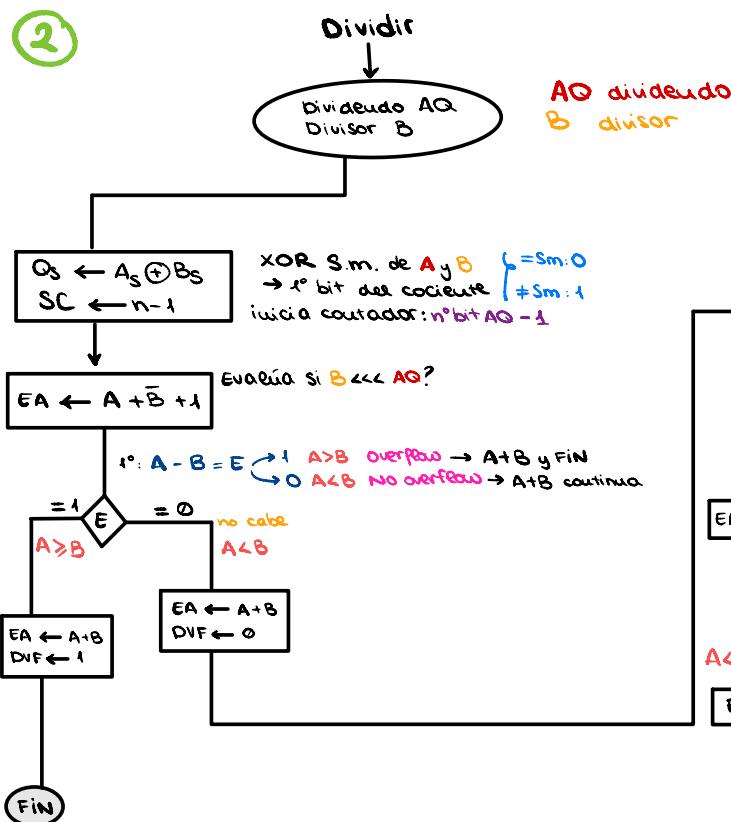
Operación Sus tractiva

5.

$$\begin{array}{r} \text{s.m.} \\ 1- | 0111111 \\ - 1- | 1011011 \\ \hline C_2 0100110 \end{array}$$

Operación sus tractiva

2



OBJ: dividir nº binario entre otro usando desplazamientos y sumas/resta

A: resto parcial (parte alta)

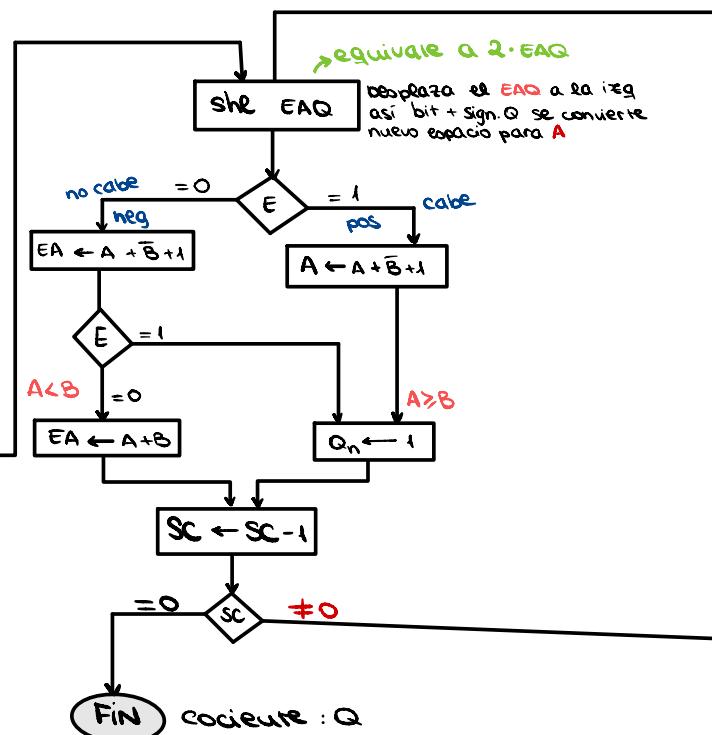
Q: cociente (parte baja)

B: divisor

EA: extensión de A para cálculos (+1 bit)

SC: contador ciclos ($n = \text{nº bits de AQ}$)

$$A - B = A + \bar{B} + 1$$



ejemplo de clase:

$$\begin{array}{r}
 \begin{array}{c} A \\ \underline{101001} \\ 10 \rightarrow \text{cabe} \\ 001001 \\ 010 \rightarrow \text{no cabe} \\ 01001 \\ 010 \rightarrow \text{cabe} \\ 0001 \\ 010 \rightarrow \text{no cabe} \\ 0001 \\ 010 \rightarrow \text{no cabe} \\ \hline 01 \rightarrow \text{resto}
 \end{array}
 \quad
 \begin{array}{c} B \\ \underline{1010} \\ 10 \rightarrow \text{cabe} \\ 10100 \\ 010 \rightarrow \text{no cabe} \\ 0101 \\ 010 \rightarrow \text{no cabe} \\ 0001 \\ 010 \rightarrow \text{no cabe} \\ 0001 \\ 010 \rightarrow \text{no cabe} \\ \hline
 \end{array}
 \end{array}$$

$1 \oplus 0 = 1 \rightarrow \text{puede continuar}$
 $6-1 = 5 \text{ repeticiones}$

cociente : Q
resto : A

ejemplo chat gpt:

$$\begin{array}{r}
 \begin{array}{c} A \\ \underline{110101} \\ 011 \rightarrow \text{cabe} \\ 10001 \\ 011 \rightarrow \text{no cabe} \\ 000101 \\ 011 \rightarrow \text{no cabe} \\ 000101 \\ 011 \rightarrow \text{no cabe} \\ 000101 \\ 011 \rightarrow \text{no cabe} \\ \hline 010 \rightarrow \text{resto}
 \end{array}
 \quad
 \begin{array}{c} B \\ \underline{1011} \\ 101 \rightarrow \text{cabe} \\ 001100 \\ 010100 \\ 101 \\ 00000 \\ \hline
 \end{array}
 \end{array}$$

ejemplo II chat gpt:

$$\begin{array}{r}
 \begin{array}{c} 60 \\ 111100 \\ 101 \rightarrow \text{cabe} \\ 001100 \\ 010100 \\ 101 \\ 00000 \\ \hline
 \end{array}
 \quad
 \begin{array}{c} 5 \\ | \\ 101 \\ 00000 \\ \hline
 \end{array}
 \end{array}$$

residuo = 0

Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

pierdo
espacio



Necesito
concentración

ali ali ooooh
esto con 1 coin me
lo quito yo...

wuolah

Tema 2: MIPS



| | |
|----------------------|-------------------------|
| $f \rightarrow \$S0$ | $A[0] \rightarrow \$S4$ |
| $g \rightarrow \$S1$ | $B[0] \rightarrow \$S5$ |
| $h \rightarrow \$S2$ | |
| $i \rightarrow \$S3$ | |

$$\text{fragmento 1} \rightarrow f = g + h + i$$

1. add \$S0, \$S1, \$S2 $f = g + h$
2. add \$S0, \$S0, \$S3 $f = g + h + i$

$$\text{fragmento 3} \rightarrow f = g + h + B[4]$$

1. lw \$t0, 16(\$S5) $t_0 = B[4]$
2. add \$S0, \$S1, \$S2 $f = g + h$
3. add \$S0, \$S0, \$t0 $f = g + h + B[4]$

$$\text{fragmento 5} \rightarrow f = g - A[B[4]]$$

1. lw \$t0, 16(\$S5) $t_0 = B[4]$
2. sll \$t0, \$t0, 2 $t_0 = 4 \cdot B[4]$
3. add \$t0, \$S4, \$t0 $t_0 = A[0] + 4 \cdot B[4]$
4. lw \$t0, 0(\$t0) $t_0 = A[B[4]]$
5. sub \$S0, \$S1, \$t0 $f = g - A[B[4]]$

$$\text{fragmento 2} \rightarrow f = g - h - i$$

1. add \$S0, \$S2, \$S3 $f = h + i$
2. sub \$S0, \$S1, \$S0 $f = g - (h + i)$

$$\text{fragmento 4} \rightarrow f = -g + h - B[4]$$

1. lw \$t0, 16(\$S5) $t_0 = B[4]$
2. add \$t1, \$S1, \$t0 $t_1 = g + B[4]$
3. sub \$S0, \$S2, \$t1 $f = h - (g + B[4])$

$$\text{fragmento 6} \rightarrow f = g + A[B[h] + 1]$$

1. sll \$t0, \$S2, 2 $t_0 = 4h$
2. add \$t0, \$S5, \$t0 $t_0 = B[0] + 4h$
3. lw \$t0, 0(\$t0) $t_0 = B[h]$
4. addi \$t0, \$t0, 1 $t_0 = B[h] + 1$
5. sll \$t0, \$t0, 2 $t_0 = 4[B[h] + 1]$
6. add \$t0, \$S4, \$t0 $t_0 = A[0] + 4[B[h] + 1]$
7. lw \$t0, 0(\$t0) $t_0 = A[B[h] + 1]$
8. add \$S0, \$S1, \$t0 $f = g + A[B[h] + 1]$

2

$$\text{fragmento 1} \rightarrow f = g + h - i$$

1. $f = g + h$
2. $f = g + h - i$

$$\text{fragmento 2} \rightarrow A[i] = B[2] + g$$

1. $t_0 = B[2]$
2. $t_0 = B[2] + F$
3. $B[2] + F = A[i]$

$$\text{fragmento 3} \rightarrow f = A[4] - B[5] - g$$

1. $t_0 = B[5]$
2. $f = B[5] + g$
3. $t_0 = A[4]$
4. $F = A[4] - B[5] - g$

$$\text{fragmento 4} \rightarrow f = g + 4(h+i) - A[2i]$$

1. $t_0 = h + i$
2. $t_0 = 4(h + i)$
3. $f = g + 4(h + i)$
4. $t_0 = 8i$
5. $t_0 = A[0] + 8i$
6. $t_0 = A[2i]$
7. $F = g + 4(h + i) - A[2i]$

5

$$\begin{aligned} a &= \$s0 \\ i &= \$s1 \\ F[0] &= \$s2 \end{aligned}$$

fragmento 1

$$\begin{aligned} a &= 0; \\ \text{for } (i=0; i < 5; i++) \\ &\quad a = a + i; \end{aligned}$$

MIPS

1. move \$s0, \$zero $a = 0$
2. move \$s1, \$zero $i = 0$
3. loop:

add \$s0, \$s0, \$s1 $a = a + i$
 addi \$s1, \$s1, 1 $i++$
 slti \$t0, \$s1, 5 $t_0 = \begin{cases} i < 5 \rightarrow t_0 = 1 \\ i \geq 5 \rightarrow t_0 = 0 \end{cases}$
 bne \$t0, \$zero, loop $\text{si } t_0 \neq 0, \text{ bucle}$
 end;

fragmento 2

$$\begin{aligned} a &= 0; \\ i &= 5; \\ \text{do } h & F[i-1] = a++; \\ \text{while } (i >= 0) \end{aligned}$$

MIPS

1. move \$s0, \$zero $a = 0$
 2. addi \$s1, \$zero, 5 $i = 5$
 3. addi \$t0, \$zero, 1 $t_0 = 1$
 4. loop:
 5. add \$t3, \$s0, \$t0 $t_3 = a + 1$
 6. sub \$t1, \$s1, \$t0 $t_1 = i - 1$
 7. sll \$t2, \$t1, 2 $t_2 = 4(i-1)$
 8. add \$t4, \$s2, \$t2 $t_4 = F[0] + 4(i-1)$
 9. lw \$t4, 0(\$t4) $t_4 = F[i-1]$
 10. sw \$t4, \$t3 $F[i-1] = a + 1$
 11. sgei \$t5, \$t1, \$zero $t_5 = \begin{cases} i >= 0 \rightarrow t_5 = 1 \\ i < 0 \rightarrow t_5 = 0 \end{cases}$
 12. bne \$t5, \$zero, loop
- end;

fragmento 3

```

for(i=0; i<5; i++)
    F[i] = rst2(4*i, i);
}

int rst2 (int a1, int a2)
{
    return 2*(a1-a2);
}

```

rst2:

```

Sub $v0, $a0, $a1  v0=a0-a1
Sll $v0, $v0, 1 v0=2(a0-a1)
jr $ra

```

MIPS

```

1 move $s1,$zero i=0
2 loop:
3 sll $t0,$s1,2 t0=4i
4 add $t1,$s2,$t0 t1=F[0]+4i
5 lw $t1,0($t1) t1=F[i]
6 move $a0,$t0 a0=4i
7 move $a1,$s1 a1=i
8 jal rst2 return 2(a0-a1)
9 sw $v0,0($t1) F[i]=rst2(4i,i)
10 addi $s1,$s1,1 i=i+1
11 seti $t2,$s1,5 t2
12 bne $t2,$zero,loop
end;

```

fragmento 4

```

for (i=0; i<5; i++)
    swap(F,i,9-i);
}

void swap(int v[],int a1,int a2)
{
    int temp;
    temp=v[a1];
    v[a1]=v[a2];
    v[a2]=temp;
}

```

MIPS

```

1 move $s1,$zero i=0
2 addi $t0,$zero,9 t0=9
3 add $a0,$zero,$s2 a0=dir F[0]
loop:
4 seti $t1,$s1,5 t1
5 beq $t1,$zero,end
6 sub $a2,$t0,$s1 a2=9-i
7 move $a1,$s1 a1=i
8 jal swap
9 addi $s1,$s1,1 i=i+1
end;

```

Swap:

```

10 move $t2,$zero t2=temp=0
11 Sll $a1,$a1,2 a1=4i
12 Sll $a2,$a2,2 a2=4(9-i)
13 add $t3,$a0,$a1 t3=v[0]+4a1
14 lw $t3,0($t3) t3=v[a1]
15 sw $t2,$t3 temp=v[a1]
16 add $t4,$a0,$a2 t4=v[0]+4a2
17 lw $t4,0($t4) t4=v[a2]
18 sw $t3,$t4 v[a1]=v[a2]
19 sw $t4,$t2 v[a2]=temp
20 jr $ra

```

Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

pierdo
espacio



Necesito
concentración

ali ali ooooh
esto con 1 coin me
lo quito yo...

wuolah

6

$\$s0 = i$
 $\$s1 = \text{dir } A[0]$

programa en C:
for ($i=0; i \leq 10; i++$)
 $A[i] = 2i + 1;$

MIPS

move \$s0, \$zero $i=0$
loop:
 sll \$t0, \$s0, 1 $t_0 = 2i$
 addi \$t1, \$t0, 1 $t_1 = 2i + 1$
 sll \$t0, \$t0, 1 $t_0 = 4i$
 add \$t2, \$s1, \$t0 $t_2 = A[0] + 4i$
 lw \$t2, 0(\$t2) $t_2 = A[i]$
 sw \$t2, \$t1 $A[i] = 2i + 1$
 addi \$s0, \$s0, 1 $i++$
 slti \$t3, \$s0, 10 $t_3 \rightarrow 0 \text{ si } i \geq 10$
 t3 \uparrow si $i < 10$
 bne \$t3, \$zero, loop
end

7

programa en C:

max_y_min (v[], dim);

int max_y_min (int v[], int dim)
 int valores[2] = {v[0], v[0]};
 for (i=1; i < dim; i++)
 if (v[i] > valores[0])
 valores[0] = v[i];
 if (v[i] < valores[1])
 valores[1] = v[i];
 return valores[];

MIPS

move \$a0, \$s0
move \$a1, \$s1
jal max_y_min

max_y_min:
 addi \$s2, \$zero, 1 $i=1$
 sw \$s3, \$a1 $\text{valores}[0] = v[0]$
 addi \$t0, \$zero, 1 $t_0=1$
 sll \$t0, \$t0, 2 $t_0=4$
 add \$t1, \$s3, \$t0 $t_1 = \text{valores}[0] + 4$
 lw \$t1, 0(\$t1) $t_1 = \text{valores}[1]$
 sw \$t1, \$a0 $\text{valores}[1] = v[0]$
 sll \$t2, \$s2, 2 $t_2 = 4i$
 add \$t3, \$a0, \$t2 $t_3 = v[0] + 4i$
 lw \$t3, 0(\$t3) $t_3 = v[i]$

loop:

 bgt \$t3, \$s3, L30
 blt \$t3, \$t1, L31
 sw \$v0, \$s3
 sw \$v1, \$t1
 addi \$s2, \$s2, 1 $i=i+1$
 seti \$t4, \$s2, \$dim $t_4 \rightarrow 0 \text{ si } i \geq \text{dim}$
 t4 \uparrow si $i < \text{dim}$
 bne \$t4, \$zero, loop
end
jal \$ra

$v[0] = \$s0$
 $\text{dim} = \$s1$
 $i = \$s2$
 $\text{valores}[0] = \$s3$

30 sw \$s3, \$t3 $\text{val}[0] = v[i]$
31 sw \$t1, \$t3 $\text{val}[1] = v[i]$

Reparo NIPS

```

A → $s0    for Ci=0; i < N; i++)
B → $s1      ↳ if C A[i] < 0)
N → $s2      ↳ B[i] = -1;
i → $t0
    ↳ else if C A[i] > 0)
    ↳ B[i] = A[i];
    ↳ else A[i] = 0
    ↳ B[i] = 0;
    ↳
{
}
  
```

NIPS

```

move $t0,$zero to=i=0
loop:
  set $t4,$t0,$s2 i < N? ↳ si t4=1
  beq $t4,$zero,
  see $t1,$t0,2 t1=4i
  add $t2,$s0,$t1 t2=A[0]+4i
  lw $t2,0($t2) t2=A[i]
  add $t3,$s1,$t1 t3=B[0]+4i
  lw $t3,0($t3) t3=B[i]
  set $t5,$t2,$zero t5 ↳ si A[i] < 0
  beq $t5,$zero,else ↳ o si A[i] >= 0
  sub $t3,$zero,$t5 B[i]=0-i
end_if

else:
  sgt $t6,$t2,$zero t6 ↳ si A[i] > 0
  beq $t6,$zero,be_zero ↳ o si A[i] = 0
  sw $t3,0($t2) B[i]=A[i]
end_if;

be_zero:
  sw $t3,0($zero) B[i]=0
end_if;
addi $t0,$t0,1 i++
end_loop;
  
```

$$B[0] = A[0]$$

```

for( i=1; i < N; i++)
  ↳ B[i] = 4 · A[i-1] - 2 · A[i];
  ↳
}
  
```

A → \$s0
B → \$s1
N → \$s2

NIPS

```

lw $t0,0($s0)
sw $t0,0($s1) ↳ B[0]=A[0]
addi $t1,$zero,1 i=t1=1
loop:
  set $t2,$t1,$s2 t2 ↳ i < N
  beg $t2,$zero,eud
  see $t3,$t1,2 t3=4i
  addi $t4,$zero,1 t4=1
  sub $t5,$t1,$t4 t5=i-1
  add
  
```

Tema 5: MEMORIA

1 M.P. $64K \cdot 16$ → tam. mem. principal

M.C. 1K palabras de M.P.

Tam bloque caché → 4 palabras

1. El número bloques de la caché

$$\frac{n \text{ líneas caché}}{\log_2 \text{ ancho palabra}} = \frac{1K}{\log_2 16} = \frac{2^10}{2^4} = 2^8 \rightarrow \text{bits de línea}$$



2. Longitud dirección de la caché → etiqueta [etiqueta] [línea] [palabra] 16 b total

Para calcular la longitud de palabra

Sabiendo que línea + palabra = long. dirección de caché ($L.D.C = 10 \text{ bits } (1K = 2^{10})$)

Por tanto

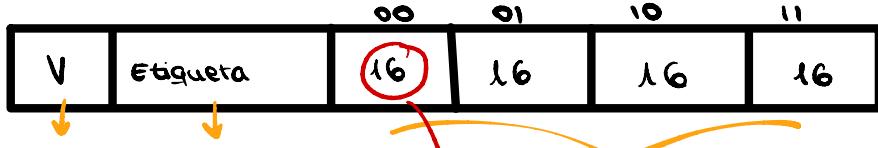
$$\text{palabra} = L.D.C - \text{línea}$$

$$\text{palabra} = 10 - 8 = 2^3$$

$$L.D.M.P = \log_2 64 + \log_2 K = 6 + 10 = 16 \text{ bits en total}$$

$$\text{etiqueta} = L.D.M.P - L.D.C = 16 - 10 = 6 \text{ bits}$$

3. Cantidad bits cada línea caché + funciones, incluyendo bit de validación



Ancho palabra × tam bloq (4)

$$\text{Bit de línea} \rightarrow 1 + 6 + (16 \times 4) = 71 \text{ b}$$

Bits total de la mem. caché → bit de linea × n° de bloques

$$71 \times 2^8 \text{ bits}$$

2

Mapo directo → 16 palabras

MP: FF1DE $5 \times 4 = 20 \text{ b}$
1111 1111 0001 1101 1110

MC: $732_{10} \rightarrow 3 \cdot 8 = 9 \text{ b}$
111 011 010

1. Longitud mem. física



Para saber e y p:

Palabra: $\log_2 16 = 4 \rightarrow p = 4$
 $e = 9 - 4 = 5$

2. Tam MP y Tam caché

$$\text{Tam MP} = 2^{20} \times 16 = 2^{24} \text{ b}$$

$$\text{Tam MC} = 2^5 \times (1 + 11 + (16 \times 16)) \text{ b}$$

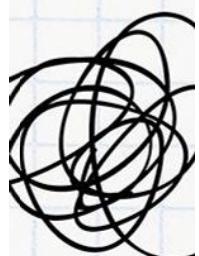


Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

pierdo
espacio



Necesito
concentración

ali ali ooooh
esto con 1 coin me
lo quito yo...

wuolah

③ 2 vías bloques de 4 palabras, almacenar 2048 palabras MP

$$MP: 128K \times 32$$

1. Organización caché

$$1 \text{ vía} = 1024 \text{ w}$$

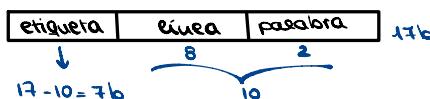
$$\text{Nº bloques} = \frac{1024}{4} = 256 \text{ líneas caché} = 2^8$$

$$\text{Palabra} = \log_2 4 = 2$$



MP:

$$128K = 2^{17} \rightarrow \text{bits totales}$$



| V | etiqueta | W ₀ | W ₁ | W ₂ | W ₃ | V | etiqueta | W ₀ | W ₁ | W ₂ | W ₃ |
|---|----------|----------------|----------------|----------------|----------------|---|----------|----------------|----------------|----------------|----------------|
| | | | | | | | | | | | |

$$TMC = 2 \times (2^8 \times (1 + 7 + (4 \times 32))) b$$

④

1. 2^{32} direcciones

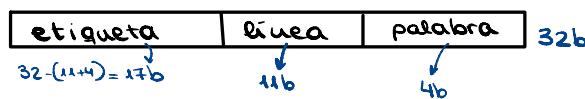
$$W = 4 \text{ bits}$$

Mapa directo → 32K palabras

Tam bloques → 16 palabras

$$\log_2 16 = 4$$

$$\text{Nº bloques} = \frac{32K}{16} = 2K \rightarrow 2^{11} \rightarrow \text{bits}$$



| elemento | Dirección MP | Etiqueta | Línea | Palabra | En caché | F/A | A caché |
|----------|--------------|------------------|-------|---------|----------|-----|---------|
| A [0] | 00101AFO | 0000001000000000 | 100 | 0000 | — | | |

9

mem. virtual \rightarrow 24 bits
 mem. física \rightarrow 16 bits

1. Número palabras existentes en mem. virtual y en mem. física

Para mem. virtual $\rightarrow 2^{24}$ palabras

Para mem. física $\rightarrow 2^{16}$ palabras

2. Utiliza paginación de 2K palabras

$$\log_2 2K = 11 \text{ bits}$$



10

mem. virtual \rightarrow Pág. 3 Despl. 10

$$2^3 \text{ págs}$$

$$\log_2 1K$$

mem. física \rightarrow Bloque 2 Despl. 10 total = 12

$$2^2 \text{ bloques}$$

| Página | 0 | 1 | 4 | 6 |
|--------|---|---|---|---|
| Bloque | 3 | 1 | 2 | 0 |

| Página | Desplazamiento | Direcciones |
|--------|-----------------------------|-------------|
| 0 10 | 000000000000 - 111111111111 | 0800 - 0BFF |
| 0 11 | 001000000000 - 111111111111 | 0C00 - 0FFF |
| 1 01 | 001000010000 - 111111111111 | 1400 - 17FF |
| 1 11 | 001000010000 - 111111111111 | 1C00 - 1FFF |

11 Tamaño página \rightarrow 1K palabras

Direcciones \rightarrow 8K palabras

Memoria \rightarrow 4K palabras

FIFO first in, first out

$$\frac{\text{Virtual}}{\text{Física}} = \frac{8K}{1K} = 2^3$$

$$\frac{\text{Virtual}}{\text{Física}} = \frac{4K}{1K} = 2^2$$



| |
|-------|
| X Ø 3 |
| Ø X 5 |
| 2 Ø 7 |
| X 2 |

quedan: 3, 5, 7, 2

LRU least recently used & menos usado

4, 0, 2, 0, 1, 2, 6, 1, 4, 0, 1, 1, 0, 2, 6, 3, 3, 5, 7

| | |
|---------|---|
| X 4 2 7 | 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, 0, 1, 2, 3, 4 |
| 0 4 6 | 0, 1, 0, 1, 2, 3, 0, 1, 2, 3, 4, 0, 1, 2, 3 |
| 2 0 5 | 0, 1, 2, 0, 1, 2, 0, 1, 0, 1, 2, 3, 4, 0 |
| X 3 | 0, 1, 0, 1, 2, 0, 1, 2, 3, 0, 0, 1 |

quedan: 3, 5, 6, 7

LFU least frequently used si hay empate FIFO

4, 0, 2, 0, 1, 2, 6, 1, 4, 0, 1, 1, 0, 2, 6, 3, 3, 5, 7

| | |
|----------------|---------------|
| X 4 X 0 3 \$ 7 | X X X 1, 2 |
| 0 | 1, 2, 3, 4 |
| 2 | 1, 2, 3 |
| 1 | 1, 2, 3, 4, 5 |

quedan: 0, 1, 2, 7

12

Memoria Virtual → paginación: $4MB = 2^2 \cdot 2^{20} = 2^{22}$

memoria: $32KB = 2^5 \cdot 2^{10} = 2^{15}$
 tam página: 512 bytes = 2^9

1. LDV = $\log_2 2^{22} = 22$ bits

LDF = $\log_2 2^9 = 9$ bits

2. N° págs $\rightarrow \frac{\text{páginas}}{\text{tam pág}} = \frac{2^{22}}{2^9} = 2^3$ págs \rightarrow DV:

| | |
|--------|----------------|
| 13 | 9 |
| Página | Desplazamiento |

Nº Bloques $\rightarrow \frac{\text{memoria}}{\text{tam pág}} = \frac{2^{15}}{2^9} = 2^6$ bloques \rightarrow DF:

| | |
|--------|----------------|
| 6 | 9 |
| Bloque | Desplazamiento |

3. Memoria Asociativa = $2^6 \times (1+13+6) = 2^6 \cdot 20$ bits

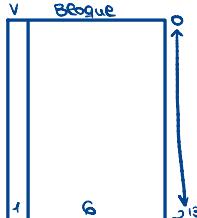


Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

Memoria de acceso aleatorio = $2^{13} \cdot (1+6) = 2^{18} \cdot 7$ bits



pierdo espacio



4. Extensión de 32 páginas

A. 000020



Desplazamiento 9b f Página: 0 → Bloque: 10

10 = 

+ Desplz.

Resultado  Desplazamiento 9b

Bloque 6b

B. 0E8000



Desplazamiento (9b)

no existe página almacenada

C. 0011A4



Página=0008 → página no existente en la memoria

$$\hookrightarrow 0 \times 8 = 8 \cdot (16^0) = 8$$

D. 0020A3



Desplazamiento

$$\hookrightarrow 0 \times 10 = 1 \cdot (16^1) + 0 \cdot (16^0) = 16$$

→ corresponde

Bloque: 7 → 000111 01010 0011

+ desplaz.

13

128 segmentos

Cada segmento → 32 págs de 4K palabras cada 1

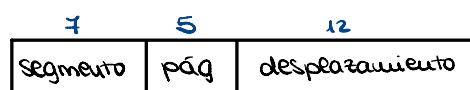
Memoria física → 4K bloques de 4K palabras cada 1

Formato de dirección lógica:

Segmento = $\log_2 128 = 7$ bits

página = $\log_2 32 = 5$ bits

palabras = $\log_2 4K = 12$ bits

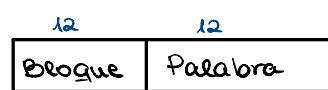


{ 24 bits total

Formato dirección física

bloque = $\log_2 4K = 12$ bits

palabra = $\log_2 4K = 12$ bits



{ 24 bits en total

ali ali 0000
esto con 1 coin me
lo quito yo...

wuolah

(14)

1. Formato dir LÓGICA

$$\begin{aligned} \text{palabra} &= \log_2 256 = 8 \text{ bits} \\ \text{página} &= 2 \times 4 \text{ bits} = 8 \text{ bits} \\ \text{segmento} &= \log_2 16 = 4 \text{ bits} \end{aligned}$$

8 8 4

total = 20 bits

Formato dir física

$$\begin{aligned} \text{bloque} &= 3 \times 4 = 12 \text{ bits} \\ \text{palabra} &= \log_2 256 = 8 \text{ bits} \end{aligned}$$

12 8

total 20 bits

2. Tamaño memoria lógica = 2^{20}
" física = 2^{20}

3. Rango direcciones memoria virtual