# Estructura de Computadores - Examen Final - 18/01/2020 - CAT

Estructura de computadors (Universitat Oberta de Catalunya)

# **Examen** 2019/20-1

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

⊂05.573ℜ18ℜ01ℜ20ℜEΞρ∈
**05.573 18 01 20 EX**

Enganxeu en aquest espai una etiqueta identificativa
amb el vostre codi personal
Examen

## Fitxa tècnica de l'examen

- Comprova que el codi i el nom de l'assignatura corresponen a l'assignatura matriculada.
- Només has d'enganxar una etiqueta d'estudiant a l'espai corresponent d'aquest full.
- No es poden adjuntar fulls addicionals, ni realitzar l'examen en llapis o retolador gruixut.
- Temps total: **2 hores** Valor de cada pregunta: **S'indica a l'enunciat**
- En cas que els estudiants puguin consultar algun material durant l'examen, quins són?

  **CAP** En cas de poder fer servir calculadora, de quin tipus? CAP
- Si hi ha preguntes tipus test: Descompten les respostes errònies? NO Quant?
- Indicacions específiques per a la realització d'aquest examen:

# **Examen** 2019/20-1

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

## Enunciats

No es pot utilitzar calculadora. Cal saber interpretar un valor en binari, decimal o hexadecimal per a realitzar l'operació que es demani. I el resultat s'ha d'expressar en el format corresponent.

# Valoració de les preguntes de l'examen

# Pregunta 1 (20%)

Pregunta sobre la pràctica.
**Cal completar les instruccions marcades o afegir el codi que es demana.**
**Els punts suspensius indiquen que hi ha més codi però no l'heu de completar.**
NOTA: En cas que el codi proposat en cada pregunta no es correspongui amb la forma que vosaltres plantejaríeu la resposta, podeu reescriure el codi o part del codi segons el vostre plantejament.

**1.1 : 10%**
**1.2 : 10%**

## Pregunta 2 (35%)

**2.1 : 10%**
**2.2 : 15%**
**2.3 : 10%**

## Pregunta 3 (35%)

**3.1: 15%**
**3.1.1 : 10%**
**3.1.2 : 5%**

**3.2: 20%**
**3.2.1 : 10%**
**3.2.2 : 5%**
**3.2.3 : 5%**

## Pregunta 4 (10%)

**4.1 : 5%**
**4.2 : 5%**

# Examen 2019/20-1

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

# Pregunta 1

## 1.1 Pràctica – 1a Part

**Escriure dos fragments de codi assemblador de la subrutina moveCursorP1, que calculi la fila (row) i la columna (col) a partir de la posició del cursor indicada per (indexMat), i actualitzi la posició del cursor indicada per (indexMat) suposant que s'ha premut la tecla 'i'. (No s'ha d'escriure el codi de tota la subrutina).**

```
;;;;;
; Actualitzar la posició del cursor al tauler, que tenim indicada
; amb la variable (indexMat), en funció a la tecla premuda,
; que tenim a la variable (charac).
; Si es surt fora del tauler no actualitzar la posició del cursor.
; (i:amunt, j:esquerra, k:avall, l:dreta)
; Amunt i avall:    ( indexMat = indexMat +/- 10 )
; Dreta i esquerra: ( indexMat = indexMat +/- 1 )
; No s'ha de posicionar el cursor a la pantalla.
; Variables globals utilitzades:
; indexMat : Índex per a accedir a les matrius mines i marks.
; charac   : Caràcter llegit de teclat.
;;;;;
moveCursorP1:
      push rbp
      mov  rbp, rsp
      ...
      mov rdx, 0
      mov rax, QWORD[indexMat]    ;int row = indexMat/10;
                                  ;int col = indexMat%10;
      mov rsi, 10
      div rsi           ;RAX = RDX:RAX / EDI, RDX = RDX:RAX mod EDI
                        ;RAX=(indexMat/10) RDX=(indexMat%10)


      cmp BYTE[charac], 'i'     ;case 'i':
      je moveCursorP1_Up
      cmp BYTE[charac], 'j'     ;case 'j':
      je moveCursorP1_Left
      cmp BYTE[charac], 'k'     ;case 'k':
      je moveCursorP1_Down
      cmp BYTE[charac], 'l'     ;case 'l':
      je moveCursorP1_Right

      moveCursorP1_Up:          if (row>0) indexMat=indexMat-10;
        cmp rax, 0                          ;if (row>0)
        jle moveCursorP1_End
        sub QWORD[indexMat], 10 ;indexMat=indexMat-10;
        jmp moveCursorP1_End ;break;
```

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

```
moveCursorP1_Left:
      ... ; AQUESTA PART DEL CODI NO S'HA D'IMPLEMENTAR
moveCursorP1_Down:
      ... ; AQUESTA PART DEL CODI NO S'HA D'IMPLEMENTAR
moveCursorP1_Right:
      ... ; AQUESTA PART DEL CODI NO S'HA D'IMPLEMENTAR
moveCursorP1_End:
...       ; AQUESTA PART DEL CODI NO S'HA D'IMPLEMENTAR
mov rsp, rbp
pop rbp
ret
```

## 1.2 Pràctica – 2a part

**Completar el codi de la subrutina updateBoardP2, per a que mostri el contingut de la matriu marks. (Només completar els espais marcats, no es poden afegir o modificar altres instruccions).**

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Aquesta subrutina es dóna feta. NO LA PODEU MODIFICAR.
; Mostrar un caràcter (dil) a la pantalla, rebut com a paràmetre,
; en la posició on està el cursor cridant la funció printchP2_C.
; Variables globals utilitzades: Cap
; Paràmetres d'entrada : rdi(dil): (c) Caràcter que volem mostrar
; Paràmetres de sortida: Cap
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
printchP2:

;;;;;
; Actualitzar el contingut del Tauler de Joc amb les dades de la matriu
; (marks) i el nombre de mines que queden per marcar que es rep
; com a paràmetre i anomenem (nMines).
; S'ha de recórrer tota la matriu (marks), i per a cada element de la matriu
; posicionar el cursor a la pantalla i mostrar els caràcters de a matriu.
; Després mostra el valor de (nMines) a la part inferior del tauler.
; Per a posicionar el cursor s'ha de cridar a la subrutina gotoxyP2,
; per a mostrar els caràcters s'ha de cridar a la subrutina printchP2 i
; per a mostrar el nombre de mines s'ha de cridar a la subrutina
; ShowMinesP2, implementant correctament el pas de paràmetres.
; Variables globals utilitzades:
; marks    : Matriu amb les mines marcades i les mines de les obertes.
; Paràmetres d'entrada : rdi(edi) : (nMines) Mines que queden per marcar.
; Paràmetres de sortida: Cap
;;;;;
updateBoardP2:

      ...

      mov eax, edi     ;Guardem el nombre de mines
      mov rdi, 7       ;rowScreen = 7;
      mov rbx, 0       ;fila (0-9)
      mov rdx, 0       ;índex per a accedir a la matriu marks. (0-99)

      updateBoardP2_for1:          ;for (i=0;i<DimMatrix;i++){
        cmp rbx, DimMatrix
        jge updateBoardP2_endfor1
        mov rsi, 7                 ;colScreen = 7;
        mov rcx, 0                 ;columna (0-9)
```

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

```
  updateBoardP2_for2:     ;for (j=0;j<DimMatrix;j++){
    cmp rcx, __DimMatrix__
    jge updateBoardP2_endfor2
    call ___gotoxyP2__          ;Posicionar cursor
    __push__ rdi
    mov  dil, __BYTE[marks+rdx]__;marks[i][j];
    call printchP2                      ;Mostrar caràcter
    pop __rdi__
    add rsi, 4              ;colScreen = colScreen + 4
    inc __rdx__  ;incrementem l'índex de la matriu
    inc rcx      ;Actualitzem la columna.
    jmp updateBoardP2_for2
  updateBoardP2_endfor2:
  add rdi, 2                ;rowScreen = rowScreen + 2;
  inc rbx                   ;Actualitzem la fila.
  jmp updateBoardP2_for1
updateBoardP2_endfor1:

;Mostrar nombre de mines que queden per marcar
...
updateBoardP2_end:

...

ret
```

# Examen 2019/20-1

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

# Pregunta 2

## 2.1

L'estat inicial del computador CISCA just abans de començar l'execució de cada fragment de codi (a cada apartat) és el següent:

| | | |
|---|---|---|
| R0= 100h<br>R1= 200h<br>R2= 300h<br>R3= 400h | M(00000200h)=FFFFF600h<br>M(00000300h)=00000600h<br>M(00000400h)=0000FFFFh<br>M(00000500h)=60000000h | Z = 0, C = 0, S = 0, V = 0 |

Completeu l'estat del computador després d'executar cada codi (indiqueu els valors dels registres en hexadecimal).
**Suposeu que l'adreça simbòlica A val 200h.**

a)

```
      MOV R0, [R1]
PLUS: CMP 0, R0
      JG ELSE
      ADD R0, 1
ELSE: ...
```

R0:= [00000200h] = FFFFFF600h
CMP 0H, FFFFFF600h

Z=0, S=0, C=1, V=0

b)

```
SAL [A+R2], 1
MOV [R2], 0
```

[00000500h]= C0000000h
[00000300h]= 0

C=0, V=0, S=1, Z=0

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

## 2.2

Completar la traducció del programa en assemblador CISCA perquè executi l'algorisme d'alt nivell mostrat. Suposem que tenim el vector V de 5 elements de 32 bits. (Hem deixat 8 espais per omplir)

```
if (A > B) && (A == C) {
      C = B;
      if (C != 100) {
            A = V[C];
      }
}
```

```
      MOV R1,[A]
COMP: CMP R1,[B]
      JLE EXIT
      CMP R1,[C]
      JNE EXIT
 OK:  MOV R2, [B]
      MOV [C], R2
      CMP R2, 100
      JE EXIT
      MUL R2, 4
      MOV R1, [V+R2]
      MOV [A], R1
EXIT:
```

# Examen 2019/20-1

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

## 2.3

Donat el següent fragment de codi d'un programa en llenguatge assemblador de CISCA:

```
FOREVER:    MOV [R10], 0
            ADD R0, 4
            JMP FOREVER
```

Traduïu-ho a llenguatge màquina i expresseu-ho en la següent taula. Suposeu que la primera instrucció del codi s'assembla a partir de l'adreça 00032D00h (que és el valor del PC abans de començar l'execució del fragment de codi). En la següent taula useu una fila per codificar cada instrucció. Si suposem que la instrucció comença en l'adreça @, el valor Bk de cadascun dels bytes de la instrucció amb adreces @+k per a k=0, 1,… s'ha d'indicar en la taula en hexadecimal en la columna corresponent (recordeu que els camps que codifiquen un desplaçament en 2 bytes o un immediat o una adreça en 4 bytes ho fan en format little endian, això cal tenir-ho en compte escrivint els bytes de menor pes, d'adreça més petita, a l'esquerra i els de major pes, adreça major, a la dreta). Completeu també la columna @ que indica per a cada fila l'adreça de memòria del byte B0 de la instrucció que es codifica en aquesta fila de la taula.

A continuació us donem com a ajuda les taules de codis:

Taula de codis d'instrucció

| B0 | Instrucción |
|---|---|
| 20h | ADD |
| 10h | MOV |
| 40h | JMP |

Taula de modes d'adreçament (Bk<7..4>)

| Camp mode Bk<7..4> | mode |
|---|---|
| 0h | Immediat |
| 1h | Registre |
| 2h | Memòria |
| 3h | Indirecte |
| 4h | Relatiu |
| 5h | Indexat |
| 6h | Relatiu a PC |

Taula de modes d'adreçament (Bk<3..0>)

| Camp mode Bk<3..0> | Significat |
|---|---|
| Nº registro | Si el mode ha d'especificar un registre |
| 0 | No s'especifica registre. |

| | | Bk per a k=0..10 | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| @ | Assemblador | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 00032D00h | MOV [R10], 0 | 10 | 3A | 00 | 00 | 00 | 00 | 00 | | | | |
| 00032D07h | ADD R0, 4 | 20 | 10 | 00 | 04 | 00 | 00 | 00 | | | | |
| 00032D0Eh | JMP FOREVER | 40 | 00 | 00 | 2D | 03 | 00 | | | | | |
| 00032D14h | | | | | | | | | | | | |

# Examen 2019/20-1

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

# Pregunta 3

## 3.1. Memòria cau

Tenim un sistema de memòria en el que tots els accessos es fan a paraula (no ens importa quina és la mida d'una paraula). Suposarem que l'espai d'adreces de memòria es descompon en blocs de 8 paraules. Cada bloc comença en una adreça múltiple de 8. Així, el bloc 0 conté les adreces 0, 1, 2, 3, 4, 5, 6, 7, el bloc 1, les adreces 8, 9, 10, 11, 12, 13, 14, 15, i el bloc N les adreces 8*N, 8*N+1, 8*N+2, 8*N+3, 8*N+4, 8*N+5, 8*N+6, 8*N+7.

Suposem que el sistema també disposa d'una memòria cau de 4 línies (on cada línia té la mida d'un bloc, es a dir, 8 paraules). Aquestes línies s'identifiquen com a línies 0, 1, 2 i 3. Quan es fa referència a una adreça de memòria principal, si aquesta adreça no es troba a la memòria cau, es porta tot el bloc corresponent des de la memòria principal a una línia de la memòria cau (així si fem referència a l'adreça 2 de memòria principal portarem el bloc format per les paraules 0, 1, 2, 3, 4, 5, 6, 7).

Suposem que el sistema fa servir una **política d'emplaçament completament associativa**, de manera que qualsevol bloc de la memòria principal es pot portar a qualsevol bloc de la memòria cau. Si trobem que la cau ja està plena, es fa servir un **algorisme de reemplaçament LRU.**

L'execució d'un programa genera la següent llista de lectures a memòria:

12, 13, 25, 26, 17, 8, 22, 3, 23, 62, 5, 63, 64, 17, 18, 19, 57, 58, 20, 25

### 3.1.1.

La següent taula mostra l'estat inicial de la cau, que conté les primeres 32 paraules de la memòria (organitzades en 4 blocs).

Completar la taula per a mostrar l'evolució de la cau durant l'execució del programa. Per a cada accés cal omplir una columna indicant si es tracta d'un encert o una fallada.

Si és un encert escriurem E en la línia corresponent davant de les adreces del bloc, si és una fallada escriurem F i s'indicarà el nou bloc que es porta a la memòria cau en la línia que li correspongui, expressat de la forma b ($a_0$ - $a_7$) on b:número de bloc, i ($a_0$ - $a_7$) són les adreces del bloc, on $a_0$ és la primera adreça del bloc i $a_7$ és la vuitena (darrera) adreça del bloc.

| Línia | Estat Inicial | | 12 | | 13 | | 25 | | 26 | | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 (0 - 7) | | 0 (0 - 7) | | 0 (0 - 7) | | 0 (0 - 7) | | 0 (0 - 7) | | 0 (0 - 7) |
| 1 | 1 (8 - 15) | E | 1 (8 - 15) | E | 1 (8 - 15) | | 1 (8 - 15) | | 1 (8 - 15) | | 1 (8 - 15) |
| 2 | 2 (16 - 23) | | 2 (16 - 23) | | 2 (16 - 23) | | 2 (16 - 23) | | 2 (16 - 23) | E | 2 (16 - 23) |
| 3 | 3 (24 - 31) | | 3 (24 - 31) | | 3 (24 - 31) | E | 3 (24 - 31) | E | 3 (24 - 31) | | 3 (24 - 31) |

# Examen 2019/20-1

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

| Línia | 8 | 22 | 3 | 23 | 62 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 (0 - 7) | 0 (0 - 7) | E 0 (0 - 7) | 0 (0 - 7) | 0 (0 - 7) | E 0 (0 - 7) |
| 1 | E 1 (8 - 15) | 1 (8 - 15) | 1 (8 - 15) | 1 (8 - 15) | 1 (8 - 15) | 1 (8 - 15) |
| 2 | 2 (16 - 23) | E 2 (16 - 23) | 2 (16 - 23) | E 2 (16 - 23) | 2 (16 - 23) | 2 (16 - 23) |
| 3 | 3 (24 - 31) | 3 (24 - 31) | 3 (24 - 31) | 3 (24 - 31) | F 7 (56 - 63) | 7 (56 - 63) |

| Línia | 63 | 64 | 17 | 18 | 19 | 57 |
|---|---|---|---|---|---|---|
| 0 | 0 (0 - 7) | 0 (0 - 7) | 0 (0 - 7) | 0 (0 - 7) | 0 (0 - 7) | 0 (0 - 7) |
| 1 | 1 (8 - 15) | F 8 (64 - 71) | 8 (64 - 71) | 8 (64 - 71) | 8 (64 - 71) | 8 (64 - 71) |
| 2 | 2 (16 - 23) | 2 (16 - 23) | E 2 (16 - 23) | E 2 (16 - 23) | E 2 (16 - 23) | 2 (16 - 23) |
| 3 | E 7 (56 - 63) | 7 (56 - 63) | 7 (56 - 63) | 7 (56 - 63) | 7 (56 - 63) | E 7 (56 - 63) |

| Línia | 58 | 20 | 25 | | | |
|---|---|---|---|---|---|---|
| 0 | 0 (0 - 7) | 0 (0 - 7) | F 3 (24 – 31) | | | |
| 1 | 8 (64 - 71) | 8 (64 - 71) | 8 (64 - 71) | | | |
| 2 | 2 (16 - 23) | E 2 (16 - 23) | 2 (16 - 23) | | | |
| 3 | E 7 (56 - 63) | 7 (56 - 63) | 7 (56 - 63) | | | |

## 3.1.2 a)

Quina és la taxa d'encerts ($T_e$) ?

$T_e$ = 17 encerts / 20 accessos = 0,85

## 3.1.2 b)

Suposem que el temps d'accés a la memòria cau, o temps d'accés en cas d'encert ($t_e$), és de 5 ns i el temps total d'accés en cas de fallada ($t_f$) és de 40 ns. Considerant la taxa d'encerts obtinguda a la pregunta anterior, quin és el temps mitja d'accés a memòria ($t_m$) ?

$t_m = T_e \times t_e + (1-T_e) \times t_f$ = 0,85 * 5 ns + 0,15 * 40 ns = 4,25 ns + 6 ns = 10,25 ns

# Examen 2019/20-1

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

## 3.2 Sistema d'E/S

Es vol analitzar el rendiment de la comunicació de dades entre la memòria d'un processador i un port USB, utilitzant E/S programada, amb les següents característiques:

- Velocitat de transferència del dispositiu d'E/S $v_{transf}$ = 2 MBytes/s = 2000 Kbytes/s
- Temps de latència mitjà del dispositiu $t_{latència}$ = 0
- Adreces dels **registres d'estat** i **dades** del controlador d'E/S: 0A00h i 0A04h
- El bit del **registre d'estat** que indica que el controlador del port d'E/S està disponible és el bit 2, o sigui el tercer bit menys significatiu (quan val 1 indica que està disponible)
- Processador amb una freqüència de rellotge de 2 GHz, el temps de cicle $t_{cicle}$ = 0,5 ns.
- El processador pot executar 2 instruccions per cicle de rellotge
- Transferència de **escriptura** des de memòria al port d'E/S
- Transferència de $N_{dades}$= 160000 dades
- La mida d'una dada és $m_{dada}$ = 4 bytes
- Adreça inicial de memòria on resideixen les dades: C0000000h

### 3.2.1

El següent codi realitzat amb el joc d'instruccions CISCA realitza la transferència descrita abans mitjançant la tècnica d'E/S programada. Completeu el codi.

```
1.       MOV R3, _160000_
2.       MOV R2, C0000000h
3.Bucle: _IN_     R0, [0A00h]  ; llegir 4 bytes
4.       AND R0, _00000100b_
5.       _JE_Bucle
6.       MOV R0,_[R2]_         ; llegir 4 bytes
7.       ADD R2, 4
8.       OUT _[0A04h]_, R0     ; escriure 4 bytes
9.       SUB R3, 1
10.      JNE Bucle
```

### 3.2.2

Quant temps dura la transferència del bloc de dades $t_{transf\_bloc}$?

$t_{transf\_bloc} = t_{latència} + (N_{dades} * t_{transf\_dada})$
$t_{latència} = 0$
$N_{dades} = 160000$

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

$t_{transf\_dada} = m_{dada} / v_{transf}$ = 4 Bytes / 2000 Kbytes/s = 0,002 ms
$t_{transf\_bloc}$ = 0 + (160000 * 0,002 ms) = 320 ms = 0,32 s

## 3.2.3

Si volguéssim fer servir el mateix processador i el mateix programa però amb un dispositiu d'E/S més ràpid, quina és la màxima taxa o velocitat de transferència del nou dispositiu que es podria suportar sense que el dispositiu s'hagués d'esperar?

$t_{cicle}$ = 0,5 ns (nanosegons)
$t_{instr}$ = 0,5 ns / 2 = 0,250 ns

El mínim nombre d'instruccions que ha d'executar el programa per a cada dada transferida són les 8 instruccions: 3, 4, 5, 6, 7, 8, 9 i 10. Executar les 8 instruccions requereix 8 * $t_{instr}$ = 8 * 0,250 ns = 2 ns

Per tant, el temps mínim per a transferir una dada és: 2 ns

Es poden transferir 4 bytes cada 2 ns, es a dir: 4 / 2 * $10^{-9}$ = 2000 Mbyte/s = 2 Gbytes/s

# Examen 2019/20-1

| Assignatura | Codi | Data | Hora inici |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

# Pregunta 4

## 4.1

Què són les instruccions de ruptura de seqüència? De quants tipus hi ha?

Permeten canviar la seqüència d'execució del programa, és a dir, decidir la instrucció que s'executarà a continuació. Hi ha 3 grups:

- Instruccions de salt
- Instruccions de crida i retorn de subrutines
- Instruccions d'interrupció

## 4.2

### 4.2.1

Quines són les tres polítiques d'assignació per a emmagatzemar dades dins d'una memòria cau? En que consisteixen?

1) Política d'assignació directa: un bloc de la memòria principal només potser en una única línia de la memòria cau. La memòria cau d'assignació directa és la que té la taxa de fallades més alta, però s'utilitza molt perquè és la més barata i fàcil de gestionar.

2) Política d'assignació completament associativa: un bloc de la memòria principal pot ser en qualsevol línia de la memòria cau. La memòria cau completament associativa és la que té la taxa de fallades més baixa. No obstant això, no se sol utilitzar perquè és la més cara i complexa de gestionar.

3) Política d'assignació associativa per conjunts: un bloc de la memòria principal pot ser en un subconjunt de les línies de la memòria cau, però dins del subconjunt pot trobar-se en qualsevol posició. La memòria cau associativa per conjunts és una combinació

### 4.2.2

En un sistema d'E/S gestionat per DMA. Explica quan i perquè es produeix una interrupció. Serveix per indicar l'inici o el final d'una transferència? Qui la genera?

Finalització de l'operació d'E/S: quan s'ha acabat la transferència del bloc
el controlador de DMA envia una petició d'interrupció al processador per informar que s'ha acabat la transferència de dades.

# **Examen** 2019/20-1

| **Assignatura** | **Codi** | **Data** | **Hora inici** |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |

# **Examen** 2019/20-1

| **Assignatura** | **Codi** | **Data** | **Hora inici** |
|---|---|---|---|
| Estructura de computadors | 05.573 | 18/01/2020 | 12:00 |