Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/6/2021	19:00

Fitxa tècnica de l'examen

- Comprova que el codi i el nom de l'assignatura corresponen a l'assignatura de què t'has matriculat.
- Temps total: **2 hores** Valor de cada pregunta: **S'indica a l'enunciat**
- Es pot consultar cap material durant l'examen? NO Quins materials estan permesos? **CAP**
- Es pot fer servir calculadora? NO 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:

Enunciat: L'enunciat de l'examen estarà en format PDF.
A l'ordinador des d'on fareu l'examen cal tindre instal·lat algun programari per a poder llegir documents en format PDF. Per exemple, es pot utilitzar el programari gratuït Adobe Acrobat Reader DC, però podeu utilitzar qualsevol altre programari.

Identificació de l'estudiant: No és necessari identificar-se, d'aquesta forma es garanteix que l'examen serà tractat de forma anònima.

Respostes:
S'ha d'identificar cada resposta dins l'examen. És ***obligatori indicar el número de pregunta i l'apartat***, opcionalment també es pot afegir tot o part de l'enunciat si això us ajuda en la resolució de la pregunta.
Si no s'identifica correctament a quina pregunta fa referència la resposta no s'avaluarà.

En cas de ser necessari aplicar un procediment per resoldre alguna pregunta, mostreu clarament i argumenteu el procediment aplicat, no només el resultat. En cas de dubte, si no es poden resoldre pels mecanismes establerts o per manca de temps, feu els supòsits que considereu oportuns i argumenteu-los.

Elaboració document a lliurar:
Utilitzar qualsevol editor de text per crear el document amb les respostes, sempre que després us permeti exportar el document a format PDF per fer el lliurament.

Lliurament: És obligatori lliurar les respostes de l'examen en un únic document ***en format PDF***.
No s'acceptaran altres formats.
És responsabilitat de l'estudiant que la informació que contingui el document PDF que es lliuri reflecteixi correctament les respostes donades a l'examen. Recomanem que obriu el fitxer PDF generat i reviseu atentament les respostes per evitar que s'hagi pogut perdre, canviar o modificar alguna informació al generar el document en format PDF.
El lliurament es pot fer tantes vegades com es vulgui, es corregirà el darrer lliurament que es faci dins l'horari especificat per realitzar l'examen.

COMPROMÍS D'AUTORESPONSABILITAT: aquest examen s'ha de resoldre de forma individual sota la vostra responsabilitat i seguint les indicacions de la fitxa tècnica (sense utilitzar cap material, ni calculadora).
En cas que no sigui així, l'examen s'avaluarà amb un zero. Per altra banda, i sempre a criteri dels Estudis, l'incompliment d'aquest compromís pot suposar l'obertura d'un expedient disciplinari amb possibles sancions.

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/6/2021	19:00

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 : 15%
1.2 : 5%

Pregunta 2 (40%)

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

Pregunta 3 (40%)

3.1: 20%
3.1.1 : 10%
3.1.2 : 10%

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

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/6/2021	19:00

Pregunta 1

1.1 Pràctica – 1a Part

Modifiqueu la subrutina moveCursorP1 perquè funcioni si en lloc de disposar de la variable 'pos' (Posició del cursor dins la matriu), tenim les variables 'row' i 'col', també de tipus int, que indiquen la fila i la columna del cursor dins la matriu, i per fer el moviment actualitzem les variables 'row' i 'col'.
; Per canviar de fila sumem o restem 1 a (row) i per canviar de
; columna sumem o restem 1 a (col) perquè cada posició de la matriu
; és de tipus char(BYTE)1byte i té ROWDIM files i COLDIM columnes.
(No s'ha d'escriure el codi de tota la subrutina, només cal modificar (afegir, eliminar o canviar) el codi per fer el què es demana).

```
;;;;;
; Actualitzar la posició del cursor dins la matriu indicada per la
; variable(pos), de tipus int(DWORD)4bytes, en funció de la tecla
; premuda (charac), de tipus char(BYTE)1byte,
; (i: amunt, j:esquerra, k:avall, l:dreta).
; Comprovar que no sortim de la matriu, (pos) només pot prendre valors
; de posicions dins de la matriu [0 : (ROWDIM*COLDIM)-1].
; Per comprovar-ho cal calcular la fila i columna dins la matriu:
; fila    = pos / COLDIM, que pot pendre valors [0 : (ROWDIM-1)].
; columna = pos % COLDIM, que pot pendre valors [0 : (COLDIM-1)].
; Per canviar de fila sumem o restem COLDIM a (pos) i per canviar de
; columna sumem o restem 1 a (pos) perquè cada posició de la matriu
; és de tipus char(BYTE)1byte i té ROWDIM files i COLDIM columnes.
; Si el moviment surt de la matriu, no fer el moviment.
; NO s'ha de posicionar el cursor a la pantalla cridant posCurScreenP1.
; Variables globals utilitzades:
; (charac): Caràcter que llegim de teclat.
; (pos)   : Posició del cursor dins la matriu.
;;;;;
moveCursorP1:
   push rbp
   mov  rbp, rsp

   push rax
   push rbx
   push rcx
   push rdx

   mov eax, DWORD[pos]
   mov edx, 0
   mov ebx, COLDIM
   div ebx      ;(eax) i = pos / COLDIM; j(edx) = pos % COLDIM;

   mov  ebx, DWORD[pos]
   mov eax, DWORD[row]
   mov edx, DWORD[col]
   mov  cl , BYTE[charac]
   moveCursorP1_i:
      cmp cl,'i'              ;case 'i':
      jne moveCursorP1_k
      cmp eax, 0              ;i > 0
      jle moveCursorP1_End
      sub ebx, COLDIM         ;pos=pos-COLDIM
      sub eax, 1
      jmp moveCursorP1_End
```

Examen 2020/21-2

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	13/6/2021	19:00

```
moveCursorP1_k:
   cmp cl,'k'              ;case 'k':
   jne moveCursorP1_j
   cmp eax, (ROWDIM-1)     ;i < (ROWDIM-1)
   jge moveCursorP1_End
   add bx, COLDIM          ;pos=pos+COLDIM;
   add eax,1
   jmp moveCursorP1_End
moveCursorP1_j:
   cmp cl,'j'              ;case 'j':
   jne moveCursorP1_l
   cmp edx, 0              ;j > 0
   jle moveCursorP1_End
   sub cbx, 1              ;pos=pos-1;
   sub edx, 1
   jmp moveCursorP1_End
moveCursorP1_l:
   cmp cl,'l'              ;case 'l':
   jne moveCursorP1_End
   cmp edx, (COLDIM-1)     ;j < (COLDIM-1)
   jge moveCursorP1_End
   add cbx, 1              ;pos=pos+1;
   add edx,1
moveCursorP1_End:
mov  DWORD[pos], cbx
mov DWORD[row], eax
mov DWORD[col], edx

pop rdx
pop rcx
pop rbx
pop rax

mov rsp, rbp
pop rbp
ret
```

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/6/2021	19:00

1.2 Pràctica – 2a part

Feu els canvis necessaris al codi assemblador d'aquesta subrutina considerant que el vector vPos és de tipus short(2 bytes), no es poden afegir instruccions, només modificar les instruccions que sigui necessari.

```
;;;;;
; Comprovar si les dues targetes obertes són iguals.
; Si les targetes són iguals canviar a l'estat de 'hi ha parella' (status=3).
; Si no són iguals tornar a girar-les. Per a fer-ho cal tornar a posar
; els valors de les targetes que ara tenim a la matriu (mOpenCards),
; a la matriu (mCards) i a la matriu (mOpenCards) posar-hi una 'X'
; (majúscula) per a indicar que estan tapades. Canviar a l'estat
; de 'no hi ha parella' (status=4). Retornar l'estat actualitzat.
; El vector (vPos) de tipus int(DWORD)4bytes, rebut com a paràmetre,
; conté les posicions de les targetes obertes.
; vPos[0]:[vPos+0]: Posició de la 1a targeta.
; vPos[1]:[vPos+4]: Posició de la 2a targeta.
; Variables globals utilitzades:
; (mCards)    : Matriu on guardem les targetes del joc.
; (mOpenCards): Matriu on tenim les targetes obertes del joc.
; Paràmetres d'entrada:
; (vPos)  : rdi(rdi): Adreça del vector amb les posicions de les targetes obertes.
; Paràmetres de sortida:
; (status): rax(eax): Estat del joc.
;;;;;
checkPairsP2:
   push rbp
   mov  rbp, rsp
   ...
   mov rbx, 0
   mov rcx, 0
   mov bx, WORD[rdi+0]         ;vPos[0]
   mov cx, WORD[rdi+2]         ;vPos[1]

   mov al, BYTE[mOpenCards+rbx] ;mOpenCards[i0][j0]
   cmp al, BYTE[mOpenCards+rcx] ;if(mOpenCards[i0][j0]==mOpenCards[i1][j1])
   jne checkPairsP2_Else
      mov eax, 3               ;status = 3; //Hi ha parella
      jmp checkPairs_End
   checkPairsP2_Else:
      mov BYTE[mCards+rbx], al      ;mCards[i0][j0]   = mOpenCards[i0][j0];
      mov BYTE[mOpenCards+rbx], 'X' ;mOpenCards[i0][j0] = 'X';
      mov al, BYTE[mOpenCards+rcx]
      mov BYTE[mCards+rcx], al      ;mCards[i1][j1]   = mOpenCards[i1][j1];
      mov BYTE[mOpenCards+rcx], 'X' ;mOpenCards[i1][j1] = 'X';
      mov eax, 4              ;status = 4; //No hi ha parella

   checkPairs_End:
   ...
   mov rsp, rbp
   pop rbp
   ret
```

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/6/2021	19:00

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 R1= 200h R2= 300h R3= 400h	M(00000100h)=00000100h M(00000200h)=FFFFF600h M(00000300h)=00000600h M(00000400h)=0000FFFFh M(00000500h)=60000000h	Z = 0, C = 0, S = 0, V = 0

Completeu l'estat del computador (registres i bits d'estat) després d'executar cada codi (indiqueu els valors dels registres en hexadecimal).

Suposeu que l'adreça simbòlica A val 100h.

a)

```
MOV R2, FFFFFFF0h
ADD R2, [A+R3]
```

```
R2:= FFFFFFF0h
R2:= FFFFFFF0h+[00000500h]=
FFFFFFF0h+60000000h = 5FFFFFF0h

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

b)

```
MOV R2, [A+100h]
SUB R2, R3
NOT R2
```

```
R2 = [200]= FFFFF600h
R2 = FFFFF200h
R2 = 00000DFFh

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

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/6/2021	19:00

2.2

Donat el següent codi d'alt nivell:

Si (A[j]>A[i]*2) A[i]= A[i]+A[j];

A és un vector de 8 elements de 4 bytes cadascun. Es proposa la següent traducció a CISCA on hem deixat 8 espais per omplir.

```
        MOV R0, [j]
        MUL R0, 4
        MOV R2, [A+R0]
        MOV R1, [i]
        MUL R1, 4
        MOV R3, [A+R1]
        SAL R3, 1
        CMP R2, R3
        JLE END
        ADD R2,[A+R1]
        MOV [A+R1], R2
END:
```

Examen 2020/21-2

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	13/6/2021	19:00

2.3

Traduïu a llenguatge màquina el fragment de codi en llenguatge assemblador que us proposem a la taula.

Suposeu que la primera instrucció del codi s'assembla a partir de l'adreça **00023C00h** (que és el valor del PC abans de començar l'execució del fragment de codi). L'etiqueta B representa l'adreça **00880000h**.

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

Taula de codis d'instrucció

B0	Instrucció
26h	CMP
43h	JE
20h	ADD
35h	SAL

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
Num. registre	Si el mode ha d'especificar un registre
0	No s'especifica registre

			Bk per k=0..10										
Adreça	**Eti**	**Assemblador**	0	1	2	3	4	5	6	7	8	9	10
00023C00h		CMP R1, 10h	26	11	00	10	00	00	00				
00023C07h		JE E1	43	60	12	00							
00023C0Bh		ADD [B+R10],2 00h	20	5A	00	00	88	00	00	00	02	00	00
00023C16h		SAL [R10], 4h	35	3A	00	04	00	00	00				
00023C1Dh	E1:												

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/6/2021	19:00

Pregunta 3

3.1. Memòria cau

Memòria cau d'accés completament associatiu
Tenim un sistema de memòria en el que tots els accessos es realitzen a paraules (no ens importa la mida de la paraula). Suposem 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, és a dir, 8 paraules). Aquestes línies s'identifiquen com línies 0, 1, 2 i 3. Quan es fa una 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 utilitza 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, s'utilitza un **algorisme de reemplaçament LRU**, de manera que traurem de la memòria cau el bloc que fa més temps que no es referencia.

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

7, 8, 9, 12, 4, 24, 18, 29, 15, 45, 51, 13, 73, 14, 52, 42, 28, 58, 20, 21

3.1.1

La següent taula mostra l'estat inicial de la cache, 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 fallada ompliu una columna indicant el nou bloc que es porta a la memòria cau a la línia que 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 i a_7 és la vuitena (última) adreça del bloc.

Línia	Estat Inicial	Fallada: 45	Fallada: 51	Fallada: 73	Fallada: 28	Fallada: 58
0	0 (0 - 7)	5 (40 – 47)				
1	1 (8 - 15)					7 (56 – 63)
2	2 (16 - 23)		6 (48 – 55)			
3	3 (24 - 31)			9 (72 – 79)	3 (24 – 31)	

Línia	Fallada: 20	Fallada:	Fallada:	Fallada:	Fallada:	Fallada:
0						
1						
2	2 (16 - 23)					
3						

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/6/2021	19:00

Línia	Fallada:	Fallada:	Fallada:	Fallada:	Fallada:	Fallada:
0						
1						
2						
3						

3.1.2 a)
Quina és la taxa d'encerts (T_e)?

T_e = 14 encerts / 20 accessos = 0,7

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 25 ns. Considerant la taxa d'encerts obtinguda en la pregunta anterior, quin és el temps mig d'accés a memòria (t_m)?

$t_m = T_e \times t_e + (1-T_e) \times t_f = 0,7 * 5 \text{ ns} + 0,3 * 25 \text{ ns} = 3,5 \text{ ns} + 7,5 \text{ ns} = 11 \text{ ns}$

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/6/2021	19:00

3.2 Sistema d'E/S

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

- Velocitat de transferència del dispositiu d'E/S v_{transf} = 10 MBytes/s = 10.000 Kbytes/s.
- Temps de latència mig del dispositiu $t_{latència}$ = 0.
- Adreça dels **registres d'estat** y **dades** del controlador de E/S: 0B00h i 0B04h.
- El bit del **registre d'estat** que indica que el controlador del port d'E/S està disponible ès el bit 4, o el cinquè bit menys significatiu (quan val 1 indica que està disponible).
- Procesador amb una freqüència de rellotge de 2 GHz, el temps de cicle t_{cicle} = 0,5 ns. El processador pot executar 1 instrucció per cada 2 cicles de rellotge.
- Transferència de **lectura** des del port d'E/S a la memòria.
- Transferència de N_{dades}= 400.000 dades.
- La mida d'una dada és m_{dada} = 4 bytes
- Adreça inicial de memòria on es troben les dades: A0000000h.
- El temps per a atendre a la interrupció (t_{rec_int}) és de 2 cicles de rellotge.

3.2.1

Completar el següent codi CISCA que es una rutina de servei a les interrupcions (RSI) per a transferir dades des del dispositiu de E/S anterior, mitjançant la tècnica d'E/S per interrupcions.

S'utilitza una variable global que es representa amb l'etiqueta **Addr,** i que al principi del programa conté l'adreça inicial de memòria on s'emmagatzemen les dades rebudes.

```
1.   CLI
2.   PUSH     _R0_
3.   PUSH     R1
4.   IN       R0, _[0B04h]_
5.   MOV      _R1_, [Addr]
6.   MOV      _[R1]_,R0
7.   ADD      _R1_,4
8.   MOV      _[Addr]_,R1
9.   _POP_    R1
10.  POP      R0
11.  STI
12.  _IRET_
```

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/6/2021	19:00

3.2.2

Quan temps dedica la CPU a la transferència del bloc de dades t_{transf_bloc}?

El temps d'un cicle, t_{cicle} = 0,5 ns (nanosegons)

Temps per a atendre la interrupció, t_{rec_int}: 2 cicles * 0,5 ns = 1 ns

Temps d'execució d'una instrucció, t_{instr}: t_{cicle} *2= 1 ns

Temps d'execució RSI, t_{rsi}: N_{rsi} * t_{instr} = 12 instr. * 1 ns = 12 ns

Temps consumit per la CPU en cada interrupció, t_{transf_dada}:

$t_{transf\_dada} = t_{rec\_int} + t_{rsi}$ = 1 + 12 = 13 ns

Número de interrupcions produïdes (número total de dades, N_{dades}): 400.000 interrupcions

Temps consumit en total en TOTES les interrupcions:

$t_{transf\_bloc} = t_{transf\_dada} * N_{dades}$ = 13 ns * 400.000 interrupcions = 5.200.000 ns = 5,2 ms (milisegons)

3.2.3

Quin és el percentatge d'ocupació del processador? Percentatge que representa el temps de transferència del bloc t_{transf_bloc} respecte al temps de transferència del bloc per part del perifèric t_{bloc}

$t_{bloc} = t_{latència} + (N_{dades} * t_{dada})$
$t_{latència}$ = 0
N_{dades} = 400.000
$t_{dada} = m_{dada} / v_{transf}$ = 4 / 10000 Kbytes/s = 0,0004 ms
t_{bloc} = 0 + (400.000 * 0,0004) ms = 160 ms
% ocupación = (t_{transf_bloc} * 100 / t_{bloc}) = (5,2 * 100) / 160 = 3,25%

Examen 2020/21-2

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	13/6/2021	19:00

Enunciats