Examen 2020/21-2

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	19/6/2021	16:00

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	19/6/2021	16:00

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	19/6/2021	16:00

Pregunta 1

1.1 Pràctica – 1a Part

Modifiqueu la subrutina showDigitsP1 perquè comprovi que el valor que volem mostrar està entre 0 i 99 (té dos dígits). Si és més petit que 0 posar el valor a mostrar a 0, si és més gran que 99 posar el valor a mostrar a 99. 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).

```
;;;;;
; Converteix un valor (value) de tipus short int(2 bytes) (entre 0 i 99) en
; dos caràcters ASCII que representin aquest valor. (27 -> '2' '7').
; S'ha de dividir el valor entre 10, el quocient representarà les
; desenes i el residu les unitats, i després s'han de convertir a ASCII
; sumant '0' o 48(codi ASCII de '0') a les unitats i a les desenes.
; Mostrar els dígits (caràcter ASCII) a partir de la fila indicada
; per la variable (rowScreen) i a la columna indicada per la variable
; (colScreen).
; Per a posicionar el cursor es cridar a la subrutina gotoxyP1 i per a
; mostrar els caràcters a la subrutina printchP1.
;
; Variables globals utilitzades:
; (rowScreen): Fila de la pantalla on posicionem el cursor.
; (colScreen): Columna de la pantalla on posicionem el cursor.
; (charac)   : Caràcter a escriure a pantalla.
; (value)    : Valor que volem mostrar.
;;;;;
showDigitsP1:
   push rbp
   mov  rbp, rsp

   push rax
   push rbx
   push rdx

   mov rax, 0
   mov rdx, 0
   mov ax, WORD[value] ;Valor que volem mostrar
   cmp ax, 0
   jl  showDigits_es zero
   cmp ax, 99
   jle  showDigits_div
   mov ax, 99
   jmp showDigits_div
   showDigits_es_zero:
   mov ax, 0
   showDigits_div:
   mov bx,10
   div bx           ;AX=DX:AX/BX, DX=DX:EAX mod BX
                    ;eax = d = value / 10; edx = u = value % 10;
   add al,'0'       ;d = d + '0';
   add dl,'0'       ;u = u + '0';

   call gotoxyP1              ;gotoxyP1_C();
   mov  BYTE[charac], al
   call printchP1             ;printchP1_C();
   inc  DWORD[colScreen]      ;colScreen++;
   call gotoxyP1              ;gotoxyP1_C();
```

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/6/2021	16:00

```
   mov  BYTE[charac], dl
   call printchP1             ;printchP1_C();

   showDigitsP1_End:
   pop rdx
   pop rbx
   pop rax

   mov rsp, rbp
   pop rbp
   ret
```

1.2 Pràctica – 2a part

Feu els canvis necessaris al codi assemblador d'aquesta subrutina considerant que las variables moves i pairs s'ha declarat de tipus long int (8 byte), no es poden afegir instruccions, només modificar les instruccions que sigui necessari.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Mostrar els valors de la matriu (mOpenCards) dins el tauler,
; a les posicions corresponents, eels moviments que es poden fer (moves)
; i les parelles que s'han de fer (pairs), rebuts com a paràmetre.
; S'ha de recórrer tota la matriu (mOpenCards), d'esquerra a dreta i
; de dalt a baix, cada posició és de tipus char(BYTE)1byte,
; i per a cada element de la matriu fer:
; Posicionar el cursor en el tauler en funció de les variables
; (rScreen) fila i (cScreen) columan cridant la subrutina gotoxyP2.
; Les variables (rScreen) i (cScreen) s'inicialitzaran, a 10 i 14,
; respectivament i que és la posició a pantalla de la casella [0][0].
; Mostrar els caràcters de cada posició de la matriu (mOpenCards)
; cridant la subrutina printchP2.
; Després, mostrar els moviments (moves) de tipus int(DWORD)4bytes,
; a partir de la posició [19,15] de la pantalla i mostrar les parelles
; fetes (pairs) de tipus int(DWORD)4bytes, a partir de la
; posició [19,24] de la pantalla cridant la subrutina showDigitsP2.
; Variables globals utilitzades: (mOpenCards9): Matriu on guardem les targetes del joc.
; Paràmetres d'entrada : (moves): rdi(di): Intents que queden.
;                        (pairs): rsi(si): Parelles que falten fer.
; Paràmetres de sortida: Cap.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
updateBoardP2:
   push rbp
   mov  rbp, rsp
   ...
   mov rdx, rdi                        ;edx: moves
   mov rbx, rsi                        ;ebx: pairs
   mov rcx, 0                          ;ecx: indexMat
   mov edi, 10                         ;rScreen=10;
   mov r10d, 0                         ;i=0
   updateBoardP2_bucle_Row:
   cmp r10d, ROWDIM                    ;i<ROWDIM
   jge updateBoardP2_show
      mov esi, 12                      ;cScreen=12;
      mov r11d, 0                      ;j=0;
      updateBoardP2_bucle_Col:
      cmp r11d, COLDIM                 ;j<COLDIM
      jge updateBoardP2_Col_end
         call gotoxyP2                 ;gotoxyP2_C(rScreen, cScreen);
         push rdi
         mov  dil, BYTE[mOpenCards+rcx];c = mOpenCards[i][j];
         call printchP2                ;printchP2_C(c);
         pop  rdi
         inc  rcx
         inc  r11d                     ;j++;
```

Examen 2020/21-2

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	19/6/2021	16:00

```
        add  esi, 4                    ;cScreen = cScreen + 4;
      jmp updateBoardP2_bucle_Col

      updateBoardP2_Col_end:
      inc  r10d                        ;i++;
      add  edi, 2                      ;rScreen = rScreen + 2;
   jmp updateBoardP2_bucle_Row

   updateBoardP2_show:
   mov edi, 19
   mov esi, 15
   mov rdx, rdx
   call showDigitsP2                   ;showDigitsP2_C(moves, 19, 15);
   mov esi, 24
   mov rdx, rbx
   call showDigitsP2                   ;showDigitsP2_C(pairs, 19, 24);

   updateBoardP2_End:
   ...
   mov rsp, rbp
   pop rbp
   ret


fdssadsf
```

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/6/2021	16:00

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 = 00000A10h R1 = 00000B20h R2 = 00000C30h R3 = 00000D40h R4= 00000E50h	M(00000A10h) = 0000F00Fh M(00000B20h) = 0000F000h M(00000C30h) = 00000001h M(00000D40h) = 11111111h M(00000E50h) = 77777777h	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).

a)	b)
SUB R0, R1 XOR R1, [R1]	MOV R1, [R4] SAL R1, [R2]
R0= FFFFFEF0 R1= 0000FB20h Z = 0 , S = 0 , C = 0 , V = 0	R1= 77777777h R1= EEEEEEEEh Z = 0 , S = 1 , C = 0 , V = 1

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/6/2021	16:00

2.2

Donat el següent codi d'alt nivell:

```
i = 10;
while (i>=MIN)  {
        A[i] = i+5;
        i = i - 1;
};
```

Es proposa la següent traducció a CISCA on hem deixat 8 espais per què els ompliu.

```
        MOV R0, Ah
        MOV R1, R0
        MUL R1,  4
PLUS:   CMP R0,[MIN]
        JL END
        MOV [A+R1], R0
        ADD [A+R1], 5
        SUB R1, 4
        DEC R0
        JMP PLUS
END:    MOV [I], R0
```

Examen 2020/21-2

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	19/6/2021	16:00

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 **00044A00h** (que és el valor del PC abans de començar l'execució del fragment de codi). L'etiqueta A representa l'adreça **00000400h**.

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

Taula de codis d'instrucció

B0	Instrucció
10h	MOV
26h	SUB
41h	JE
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
Nº registro	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
00044A00h	E1:	SUB R6,[A+R4]	26	16	54	00	04	00	00				
00044A07h		MOV R3,[100h]	10	13	20	00	01	00	00				
00044A0Eh		JE E1	41	60	EE	FF							
00044A12h		SAL [R6], 2h	35	36	00	02	00	00	00				
00044A19h													

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/6/2021	16:00

Pregunta 3

3.1. Memòria cau

Memòria cau d'assignació directa

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'assignació directa**, de manera que cada bloc de la memòria principal sols es pot portar a una línia determinada de la memòria cache.
L'execució d'un programa genera la següent llista de lectures a memòria:

7, 8, 5, 24, 3, 18, 55, 6, 17, 18, 32, 40, 4, 6, 63, 40, 18, 53, 42, 50

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 cache durant l'execució del programa. Indicar únicament aquells accessos que provoquen una fallada de cau. Per a cada fallada ompliu una columna indicant el nou bloc que es porta a la memòria cau en la línia que correspongui, expressat de la forma b:e (a_0 - a_7) on b: número de bloc, e:etiqueta i (a_0 - a_7) són les adreces del bloc, sent a_0 la primera adreça del bloc i a_7 la vuitena (última) adreça del bloc.

Línia	Estat Inicial	Fallada: 55	Fallada: 17	Fallada: 32		Fallada: 40	Fallada: 4
0	0:0 (0 - 7)			4:1 (32 - 39)			0:0 (0 – 7)
1	1:0 (8 - 15)					5:1 (40 - 47)	
2	2:0 (16 - 23)	6:1 (48 - 55)	2:0 (16 - 23)				
3	3:0 (24 - 31)						

Examen 2020/21-2

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	19/6/2021	16:00

Línia	Fallada: 63	Fallada: 53	Fallada:	Fallada:		Fallada:	Fallada:
0							
1							
2		6:1 (48 - 55)					
3	7:1 (56 – 63)						

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 = 13 encerts / 20 accessos = 0,65

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)?

$t_m = T_e \times t_e + (1-T_e) \times t_f = 0,65 * 5 \text{ ns} + 0,35 * 25 \text{ ns} = 3,25 \text{ ns} + 8,75 \text{ ns} = 12 \text{ ns}$

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/6/2021	16:00

3.2 Sistema d'E/S

E/S programada

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 = 2.000 Kbytes/s.
- Temps de latència mig del dispositiu $t_{latència}$ = 0.
- Adreces dels **registres d'estat** i **de dades** del controlador d'E/S: 0200h y 0204h.
- El bit del **registre de estat** que indica que el controlador del port d'E/S està disponible és el bit 6, o el setè bit menys significatiu (quan val 1 indica que està disponible).
- Processador amb una freqüència de rellotge de 4 GHz, el temps de cicle t_{cicle} = 0,25 ns. El processador pot executar 1 instrucció cada 2 cicles de rellotge.
- Transferència d'**escriptura** des de la memòria al port d'E/S.
- Transferència de N_{dades}= 1.600.000 dades
- La mida d'una dada és m_{dada} = 4 bytes
- Adreça inicial de memòria on es troben les dades: 20000000h

3.2.1

El següent codi realitzat amb el repertori d'instruccions CISCA realitza la transferència descrita a dalt mitjançant la tècnica d'E/S programada. Completar el codi.

```
1.        MOV     R3, _1600000_
2.        MOV     R2, 20000000h
3.Bucle:  IN      R0, _[0200h]_     ; llegir 4 bytes
4.        AND     R0, _0100000b_
5.        _JE_    Bucle
6.        MOV     R0,_[R2]_         ; llegir 4 bytes
7.        _OUT_   [0204h], R0       ; escriure 4 bytes
8.        ADD     R2, _4_
9.        DEC     R3
10.       _JNE_   Bucle
```

Examen 2020/21-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/6/2021	16:00

3.2.2

Quan temps dura la transferència del bloc de dades t_{transf_bloc}?

$t_{transf\_bloc} = t_{latència} + (N_{dades} * t_{transf\_dada})$
$t_{latencia} = 0$
$N_{dades} = 1600000$
$t_{transf\_dada} = m_{dada} / v_{transf} = 4$ Bytes / 2000 Kbytes/s = 0,002 ms = 2 us
$t_{transf\_bloc} = 0 + (1.600.000 * 0{,}002$ ms) = 3.200 ms = 3,2 s

3.2.3

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

t_{cicle} = 0,25 ns (nanosegons)

t_{instr} = 0,25ns * 2 = 0,5ns

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 * 0,5 ns = 4 ns

Per tant, el temps mínim per a transferir una dada és: 4 ns

Es poden transferir 4 bytes cada 4 ns, és a dir: $4 / 4 * 10^{-9} =$ 1000 Mbyte/s = 1 Gbytes/s

Examen 2020/21-2

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	19/6/2021	16:00

Enunciats