Examen 2018/19-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/01/2019	15:30

⊂05.573ℜ19ℜ01ℜ19ℜEΞ&∈
05.573 19 01 19 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 2018/19-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/01/2019	15:30

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 2018/19-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/01/2019	15:30

Pregunta 1

1.1 Pràctica – 1a Part

Escriure un fragment de codi assemblador de la subrutina calcIndexP1, per calcular el valor de l'índex (indexMat) per a accedir a una matriu (4x5) de ROWDIM * COLDIM posicions de tipus char(BYTE)1byte, cadascuna, a partir de la fila (row) i la columna (col) especificades. (No s'ha d'escriure el codi de tota la subrutina)

```
;;;;;
; Calcular el valor de l'índex (indexMat) per a accedir a una matriu
; (4x5) de ROWDIM * COLDIM posicions de tipus char(BYTE)1byte, cadascuna,
; a partir de la fila (row) i la columna (col) especificades.
; indexMat=((row*COLDIM)+(col))
; m[i][j] en C, és equivalent a BYTE[m+eax] en assemblador,
; si eax = indexMat = ((row*COLDIM)+(col)).
; m[1][2] en C, és DWORD[m+7] en assemblador.
;
; Aquesta subrutina no té una funció en C equivalent.
;
; Variables globals utilitzades:
; row     : Fila que volem accedir de la matriu (4x5).
; col     : Columna que volem accedir de la matriu (4x5).
; indexMat: Índex per a accedir a la matriu (4x5).
;;;;;
calcIndexP1:
   push rbp
   mov  rbp, rsp

   push rax
   push rbx
   push rcx
   push rdx

   mov rax, 0
   mov rbx, 0
   mov rcx, 0
   mov rdx, 0

   mov eax, DWORD[row]
   mov ebx, COLDIM
   mul ebx              ;(EDX:EAX = EAX; font): eax = (row*COLDIM)
   add eax, DWORD[col]   ;eax = (row*COLDIM)+(col)
   mov DWORD[indexMat], eax

   calcIndexP1_End:
   pop rdx
   pop rcx
   pop rbx
   pop rax

   mov rsp, rbp
   pop rbp
   ret
```

Examen 2018/19-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/01/2019	15:30

1.2 Pràctica – 2a part

Completar el codi de la subrutina checkPairsP2. (Només completar els espais marcats, no es poden afegir, ni modificar altres instruccions).

```
;;;;;
; Calcular el valor de l'índex (eax) per a accedir a una matriu
; (4x5) de ROWDIM * COLDIM posicions de tipus char(BYTE)1byte, cadascuna,
; que guardarem al registre (eax) a partir de la fila (edi) i la
; columna (esi), rebuts com a paràmetre.
; eax=((edi*COLDIM)+(esi))
; m[i][j] en C, és equivalent a BYTE[m+eax] en assemblador,
; si eax = ((edi*COLDIM)+(esi)).
; m[1][2] en C, és DWORD[m+7] en assemblador.
;
; Aquesta subrutina no té una funció en C equivalent.
;
; Variables globals utilitzades:
; Cap.
; Paràmetres d'entrada :
; rsi(esi): Fila de la matriu (5x4).
; rdi(edi): Columna de la matriu (5x4).
; Paràmetres de sortida:
; rax(eax) : Índex per a accedir a matrius (5x4).
;;;;;
calcIndexP2:

;;;;;
; Comprovar si les dues targetes obertes són iguals.
; Si les targetes són iguals canviat l'estat 'hi ha parella' (status=3).
; Si no són iguals tornar a girar-les. Per fer-ho cal tornar a posar
; els  valors de targetes que ara tenim a la matriu (mOpenCards) a la
; matriu (mCards) i a la matriu (mOpenCards) posar-hi una 'X' per a
; indicar que estan tapades. Canviar l'estat a 'no hi ha parelles' (status=4).
; La matriu (mMoves) de tipus short(WORD)2bytes conté la fila i
; la columna de les targetes obertes.
; mMoves[0][0]:[mMoves+0]: Fila de la 1a targeta;
; mMoves[0][1]:[mMoves+2]: Columna de la 1a targeta;
; mMoves[1][0]:[mMoves+4]: Fila de la 2a targeta;
; mMoves[1][1]:[mMoves+6]: Columna de la 2a targeta;
; Per accedir a les matrius (mOpenCards) i (mCards) en assemblador
; s'ha de calcular l'índex cridant la subrutina calcIndexP2,
; implementant correctament el pas de paràmetres.
;
; Variables globals utilitzades:
; mMoves     : Matriu amb les posicions de les targetes obertes.
; mCards     : Matriu on guardem les targetes del joc.
; mOpenCards : Matriu on tenim les targetes obertes del joc.
;
; Paràmetres d'entrada:
; Cap
;
; Paràmetres de sortida:
; status :rax(eax): Estat del joc.
;;;;;
checkPairsP2:
   push rbp
   mov  rbp, rsp

   push rsi
   push rdi
   push r10
   push r11
   push r12
   push r13
   push r14
   push r15

   mov r10, 0
   mov r11, 0
   mov r12, 0
   mov r13, 0
```

Examen 2018/19-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/01/2019	15:30

```
mov r10w, WORD[mMoves+0]   ;int i0 = mMoves[0][0];
mov r11w, WORD[mMoves+2]   ;int j0 = mMoves[0][1];
mov r12w, WORD[mMoves+4]   ;int i1 = mMoves[1][0];
mov r13w, WORD[mMoves+6]   ;int j1 = mMoves[1][1];

mov  edi, r10d
mov  esi, r11d
call calcIndexP2
mov  r14d, _eax_           ;index0 = (i0*5)+j0
mov  edi, r12d
mov  esi, r13d
call calcIndexP2
mov  r15d, _eax_           ;index1 = (i1*5)+j1

mov al, _BYTE[mOpenCards+r14d]_ ;mOpenCards[i0][j0]
_cmp_ al, BYTE[mOpenCards+r15d]
                           ;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+r14d], al      ;mCards[i0][j0]   = mOpenCards[i0][j0];
   mov BYTE[mOpenCards+r14d], 'X' ;mOpenCards[i0][j0] = 'X';
   mov al, _BYTE[mOpenCards+r15d]_
   mov BYTE[mCards+r15d], al      ;mCards[i1][j1]   = mOpenCards[i1][j1];
   mov BYTE[mOpenCards+r15d], 'X' ;mOpenCards[i1][j1] = 'X';
   mov _eax_, 4            ;status = 4; //No hi ha parella

checkPairs_End:
pop r15
pop r14
pop r13
pop r12
pop r11
pop r10
pop rdi
pop rsi

mov rsp, rbp
pop rbp
ret
```

Examen 2018/19-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/01/2019	15:30

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:

R2 = 00000020h R4 = 00000040h R8 = 00000080h	M(00000290h) = 810077E0h M(00000060h) = 00000810h M(00000250h) = 00001810h	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 250h.

a)

```
ADD R2, R4
MOV R8, [R2]
SUB R8, [A]
```

R2 = 00000060h
R8 = 00000810h
R8 = FFFFF000h

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

b)

```
    MOV R2,[A+R4]
    CMP [A],R2
    JNE EXIT
    ...
EXIT:
```

R2 = 810077E0h

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

Examen 2018/19-1

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	19/01/2019	15:30

2.2

Donat el següent codi d'alt nivell:

```
if (A<B) {
     if (C>=B) A=C;
     else B= C;
}
else A=B;
```

Es proposa la següent traducció a CISCA on hem deixat 6 espais per què els ompliu.

```
INI:   MOV  R0, [A]
       MOV  R1, [B]
       MOV  R2, [C]
       CMP  R0, R1
       JGE  LSE1
       CMP  R2, R1
       JL   LSE2
       MOV  R0, R2
       JMP  END
LSE2:  MOV  R1, R2
       JMP  END
LSE1:  MOV  R0, R1
END:   MOV  [A], R0
       MOV  [B], R1
       MOV  [C], R2
```

Examen 2018/19-1

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	19/01/2019	15:30

2.3

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

```
        TEST R0,[M]
        JE   BEGIN
        INC  [R0]
BEGIN:  CMP  [M+R2], 10h
```

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 00023C00h (que és el valor del PC abans de començar l'execució del fragment de codi). Suposeu que l'adreça simbòlica M val 00000400h. En la següent taula useu una fila per a 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 codifiquin un desplaçament en 2 bytes o un immediat o una adreça en 4 bytes ho fan en format little endian, això calç 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 és codifica en aquesta fila de la taula.

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

Taula de codis d'instrucció

B0	Instrucció
43h	JE
26h	CMP
33h	TEST
24h	INC

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 a k=0..10										
@	**Assemblador**	0	1	2	3	4	5	6	7	8	9	10
00023C00h	TEST R0, [M]	33	10	20	00	04	00	00				
00023C07h	JE BEGIN	43	60	02	00							
00023C0Bh	INC [R0]	24	30									
00023C0Dh	CMP [M+R2], 10h	26	52	00	04	00	00	00	10	00	00	00
00023C18h												

Examen 2018/19-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/01/2019	15:30

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:

7, 8, 5, 24, 3, 18, 55, 6, 17, 18, 32, 40, 4, 6, 63, 40, 48, 56, 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 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	7	8	5	24	3
0	0 (0 - 7)	E 0 (0 - 7)	0 (0 - 7)	E 0 (0 - 7)	0 (0 - 7)	E 0 (0 - 7)
1	1 (8 - 15)	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)	2 (16 - 23)
3	3 (24 - 31)	3 (24 - 31)	3 (24 - 31)	3 (24 - 31)	E 3 (24 - 31)	3 (24 - 31)

Línia	18	55	6	17	18	32
0	0 (0 - 7)	0 (0 - 7)	E 0 (0 - 7)	0 (0 - 7)	0 (0 - 7)	0 (0 - 7)
1	1 (8 - 15)	F 6 (48 - 55)	6 (48 - 55)	6 (48 - 55)	6 (48 - 55)	6 (48 - 55)
2	E 2 (16 - 23)	2 (16 - 23)	2 (16 - 23)	E 2 (16 - 23)	E 2 (16 - 23)	2 (16 - 23)
3	3 (24 - 31)	3 (24 - 31)	3 (24 - 31)	7 (56 - 63)	7 (56 - 63)	F 4 (32 - 39)

Examen 2018/19-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/01/2019	15:30

Línia	40		4		6		63		40		48	
0		0 (0 - 7)	E	0 (0 - 7)	E	0 (0 - 7)		0 (0 - 7)		0 (0 - 7)		0 (0 - 7)
1	F	5 (40 - 47)		5 (40 - 47)		5 (40 - 47)		5 (40 - 47)	E	5 (40 - 47)		5 (40 - 47)
2		2 (16 - 23)		2 (16 - 23)		2 (16 - 23)	F	7 (56 - 63)		7 (56 - 63)		7 (56 - 63)
3		4 (32 - 39)		4 (32 - 39)		4 (32 - 39)		4 (32 - 39)		4 (32 - 39)	F	6 (48 - 55)

Línia	56		42		50							
0		0 (0 - 7)		0 (0 - 7)		0 (0 - 7)						
1		5 (40 - 47)	E	5 (40 - 47)		5 (40 - 47)						
2	E	7 (56 - 63)		7 (56 - 63)		7 (56 - 63)						
3		6 (48 - 55)		6 (48 - 55)	E	6 (48 - 55)						

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

T_e = 15 encerts / 20 accessos = 0,75

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) ?

$t_m = T_e \times t_e + (1-T_e) \times t_f$ = 0,75 * 5 ns + 0,25 * 40 ns = 3,75 ns + 10 ns = 13,75 ns

Examen 2018/19-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/01/2019	15:30

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
```

Examen 2018/19-1

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	19/01/2019	15:30

3.2.2
Quant temps dura la transferència del bloc de dades t_{transf_bloc}?

$t_{transf\_bloc} = t_{latència} + (N_{dades} * t_{transf\_dada})$
$t_{latència} = 0$
$N_{dades} = 160000$
$t_{transf\_dada} = m_{dada} / v_{transf} = 4$ 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 2018/19-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	19/01/2019	15:30

Pregunta 4

4.1

Què entenem per «segmentació dels instruccions»?

La segmentació dels instruccions (pipeline) consisteix a dividir el cicle d'execució dels instruccions en un conjunt d'etapes. Aquestes etapes poden coincidir, o no, amb els fases del cicle d'execució dels instruccions.

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.