Espai grapa

Examen 2012/13-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	22/06/2013	12:00

⊂05.573ℜ22ℜ06ℜ13ℜEΞ8∈
05.573 22 06 13 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 en la qual estàs matriculat.**
- **Només has d'enganxar una etiqueta d'estudiant a l'espai corresponent d'aquest full.**
- **No es poden adjuntar fulls addicionals.**
- **No es pot realitzar la prova en llapis ni en retolador gruixut.**
- **Temps total:** 2 **h.**
- **En cas que els estudiants puguin consultar algun material durant l'examen, quin o quins materials poden consultar?**
  No es pot utilitzar calculadora ni material auxiliar
- **Valor de cada pregunta:** Pregunta 1 (20%); Pregunta 2 (40%); Pregunta 3 (40%)
- **En cas que hi hagi preguntes tipus test: Descompten les respostes errònies?** NO **Quant?**

- **Indicacions específiques per a la realització d'aquest examen:**

Enunciats

Examen 2012/13-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	22/06/2013	12:00

Pregunta 1 (20%)

Pregunta sobre la pràctica.

Només s'han de completar les instruccions marcades.
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 (40%)

2.1: 15%

2.2: 15%

2.3: 10%

Pregunta 3 (40%)

3.1: 20%

3.1.1: 10%

3.1.2: 10%

3.2: 20%

3.2.1: 10%

3.2.2: 10%

Examen 2012/13-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	22/06/2013	12:00

Pregunta 1

1.1

```
;;;;;
; Comprovar que el vector rebut com a paràmetre (l'adreça del vector)
; de 5 elements contingui només caràcters vàlids, comparant amb els
; caràcters del vector validChar.
; Per cada element del vector rebut com a paràmetre, comprovar que
; estigui en el vector validChar, si el troba passar al següent element
; si no el troba retornar un 1 i sortir.
; Si els troba tots, retornar un 0 i sortir.
;
; Variables utilitzades:
; validChar
;
; Paràmetres d'entrada :
; ebx: adreça del vector a comprovar
;
; Paràmetres de sortida:
; eax: resultat de la comprovació: 0 combinació correcta,
; 1 caràcters invàlids
;;;;;
CheckChars:
push rbp
mov rbp, rsp

...

mov edi, 0
mov esi, 0
mov eax, 0
CC_loopSecret:
mov al,[__ebx__+__edi__]
CC_loopChar:
cmp al,[__validChar__+__esi__]
je CC_nextSecretChar
__inc__ esi
cmp esi, 10
je __CC_invalidChar__
jmp __CC_loopChar__

CC_nextSecretChar:
mov esi, 0
inc edi
cmp edi, 5
jl CC_loopSecret
mov eax, 0 ;combinació correcta
jmp CC_endCheckChars

CC_invalidChar:
mov eax, 1 ;caràcter invàlid
CC_endCheckChars:

...

mov rsp, rbp
pop rbp
ret
```

Examen 2012/13-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	22/06/2013	12:00

1.2

```
;;;;;
; Llegir una jugada.
;
; Netejar l'espai on es llegeix la jugada amb espais en blanc,
; cridant la funció clearArea_C passant com a paràmetre row i col
; ...
; Llegir la jugada, cridant la subrutina GetCode, passant com a paràmetre
; l'adreça del vector play a través del registre ebx, posar showChar=1, per a
; indicar que GetCode que mostri els caràcters llegits.
; Si GetCode retorna un 6, s'ha premut ESC, imprimir el missatge corresponent
; cridant la funció printMessage_C i sortir.
; en cas contrari ...
; i repetir la lectura fins que la combinació sigui correcta.
; ...
;
; Variables utilitzades:
; play : vector que emmagatzema la jugada
; row : fila de la pantalla on llegirem la combinació
; col : columna inicial de la pantalla on llegirem la combinació
; showChar: 0: mostrar un * per al caràcter llegit, 1: mostrar el caràcter llegit
;
; Paràmetres d'entrada :
; Cap
;
; Paràmetres de sortida:
; eax: codi de sortida: 6 si s'ha premut ESC, 0 en cas contrari
;;;;;
GetPlay:
push rbp
mov rbp, rsp

...

GP_readPlay:
mov __edi__, [row] ;fila
mov __esi__, [col] ;columna
call clearArea_C

...

mov ebx, __[play]__
mov dword __[showChar]__, 1
call GetCode
cmp eax, 6
__je__ GP_finish

...

jmp GP_readPlay

GP_finish:

...

mov rsp, rbp
pop rbp
ret
```

Examen 2012/13-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	22/06/2013	12:00

Pregunta 2

2.1

Suposeu el següent estat inicial de la CISCA (abans de cada apartat):

- Registres: Ri = 2 * i per a i = 0, 1, …,15.
- Memòria: M(i)=(i+16) per a i = 0, 4, 8,…, 2^{32}-4.

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

Suposeu que l'adreça simbòlica A val 400h i l'adreça simbòlica B val 800h

a)

```
XOR   R1, 10h
SUB   R1, [A]
MOV   [B], 100h
```

```
R1:= 12H
R1:= 12H - 410h = FFFFFC02h
M(800H):= 100H
```

```
R1 = FFFFFC02h
M( 800h ) = 100h
Z =  0, C =  0, S =  1, V = 0
```

b)

```
MOV   R1, FFFFh
MOV   R2, 10H
SAR   R1, R2
```

```
R1:= FFFFH
R2:= 10H
R1:= 0H
```

```
R1 = 0H
R2 = 10H
Z =  1, C =  0, S = 0, V = 0
```

2.2

En la memòria d'un computador CISCA tenim emmagatzemada una matriu de 10 x 10 elements (10 files per 10 columnes) a partir de l'adreça simbòlica M. Cada element és un nombre enter codificat en complement a 2 amb 32 bits.

Completeu els buits del fragment de codi CISCA per assignar a la variable A el doble del contingut de l'element M[i][i], la qual cosa en C s'especificaria amb la sentència:

A= M[i][i] x 2;

La matriu està emmagatzemada per files en posicions consecutives de memòria, com és habitual quan es tradueix codi en C. Per exemple, els elements M[0][0], M[0][1], M[1][0] i M[7][4] es troben emmagatzemats en les adreces de memòria M, M+4, M+40 i M+296 respectivament.

Se sap que en R1 es troba emmagatzemat el valor de la variable "i", i que després d'executar-se el fragment de codi tots els registres han de mantenir els valors originals.

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	22/06/2013	12:00

```
PUSH  R1
PUSH  R2
MOV   R2, R1
MUL   R1, 40
SAL   R2, 2
ADD   R1, R2
MOV   R2, [M+R1]
SAL   R2, 1
MOV   [A], R2
POP   R2
POP   R1
```

2.3

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

```
MUL   R1, 4
SAL   R2, 2
MOV   R3, [B+R2]
```

Traduïu-ho a llenguatge màquina i expresseu-ho en la següent taula. Suposeu que la primera instrucció del codi es troba a partir de l'adreça 0009A600h (que és el valor del PC abans de començar l'execució del fragment de codi). Suposeu també que l'adreça simbòlica B val 00011250h. En la taula per al resultat 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	Instrucció
10h	MOV
22h	MUL
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

Examen 2012/13-2

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	22/06/2013	12:00

5h	Indexat
6h	Relatiu a PC

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

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

		Bk per a k=0..10										
@	**Assamblador**	0	1	2	3	4	5	6	7	8	9	10
0009A600h	`MUL R1, 4`	22	11	00	04	00	00	00				
0009A607h	`SAL R2, [B]`	35	12	20	50	12	01	00				
0009A60Eh	`MOV R3, [B+R2]`	10	13	52	50	12	01	00				
0009A615h												

		Bk per a k=0..10										
@	**Assamblador**	0	1	2	3	4	5	6	7	8	9	10
0009A600h	`MUL R1, 4`	22	11	00	04	00	00	00				
0009A607h	`SAL R2, 2`	35	12	00	02	00	00	00				
0009A60Eh	`MOV R3, [B+R2]`	10	13	52	50	12	01	00				
0009A615h												

Examen 2012/13-2

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	22/06/2013	12:00

Pregunta 3

3.1

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 4 paraules. Cada bloc comença a una adreça múltiple de 4. Així, el bloc 0 conté les adreces 0, 1, 2, i 3; el bloc 1, les adreces 4, 5, 6 i 7, i el bloc N les adreces 4*N, 4*N+1, 4*N+2 i 4*N+3.

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, 4 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 i 3).

3.1.1 Memòria Cau d'Accés Directe

Suposem que el sistema fa servir una **política d'assignació directa**, de manera que cada bloc de la memòria principal només es pot portar a una línia determinada de la memòria cau.

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

4, 1, 8, 25, 2, 48, 22, 53, 2, 12, 21, 32, 22, 23, 13

3.1.1.a) La següent taula mostra l'estat inicial de la cau, que conté les primeres 16 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 en la cau cal omplir una nova columna indicant quina referència a memòria ha provocat la fallada i el canvi que es produeix en l'estat de la memòria cau (la línia que es modifica).

	Estat Inicial	Fallada: 25	Fallada: 48	Fallada: 22
Línia 0	0, 1, 2, 3	0, 1, 2, 3	48, 49, 50, 51	48, 49, 50, 51
Línia 1	4, 5, 6, 7	4, 5, 6, 7	4, 5, 6, 7	20, 21, 22, 23
Línia 2	8, 9, 10, 11	24, 25, 26, 27	24, 25, 26, 27	24, 25, 26, 27
Línia 3	12, 13, 14, 15	12, 13, 14, 15	12, 13, 14, 15	12, 13, 14, 15

	Fallada: 53	Fallada: 2	Fallada: 21	Fallada: 32
Línia 0	48, 49, 50, 51	0, 1, 2, 3	0, 1, 2, 3	32, 33, 34, 35
Línia 1	52, 53, 54, 55	52, 53, 54, 55	20, 21, 22, 23	20, 21, 22, 23
Línia 2	24, 25, 26, 27	24, 25, 26, 27	24, 25, 26, 27	24, 25, 26, 27
Línia 3	12, 13, 14, 15	12, 13, 14, 15	12, 13, 14, 15	12, 13, 14, 15

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	22/06/2013	12:00

3.1.1.b) Quina és la taxa de fallades (T_f) ?

T_f = 7 fallades / 15 accessos = 0,46

3.1.1.c) Suposem que el temps d'accés a la memòria cau, o temps d'accés en cas d'encert (t_e), és de 4 ns i el temps total d'accés en cas de fallada (t_f) és de 20 ns. Considerant la taxa de fallades obtinguda a la pregunta anterior, quin és el temps mitja d'accés a memòria (t_m) ?

$$t_m = T_f \times t_f + (1\text{-}T_f) \times t_e = 0{,}46 * 20 \text{ ns} + 0{,}53 * 4 \text{ ns} = 9{,}33 \text{ ns} + 2{,}13 \text{ ns} = 11{,}46 \text{ ns}$$

3.1.2 Memòria Cau d'Accés Completament Associatiu

Ara suposem que el mateix 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, de manera que traurem de la memòria cau aquell bloc que fa més temps que no es referència.

Considerem la mateixa llista de lectures a memòria:

4, 1, 8, 25, 2, 48, 22, 53, 2, 12, 21, 32, 22, 23, 13

3.1.2.a) 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.

	Estat Inicial	Fallada: 25	Fallada: 48	Fallada: 22
Línia 0	0, 1, 2, 3	0, 1, 2, 3	0, 1, 2, 3	0, 1, 2, 3
Línia 1	4, 5, 6, 7	4, 5, 6, 7	48, 49, 50, 51	48, 49, 50, 51
Línia 2	8, 9, 10, 11	8, 9, 10, 11	8, 9, 10, 11	20, 21, 22, 23
Línia 3	12, 13, 14, 15	24, 25, 26, 27	24, 25, 26, 27	24, 25, 26, 27

	Fallada: 53	Fallada: 12	Fallada: 32	Fallada:
Línia 0	0, 1, 2, 3	0, 1, 2, 3	0, 1, 2, 3	
Línia 1	48, 49, 50, 51	12, 13, 14, 15	12, 13, 14, 15	
Línia 2	20, 21, 22, 23	20, 21, 22, 23	20, 21, 22, 23	
Línia 3	52, 53, 54, 55	52, 53, 54, 55	32, 33, 34, 35	

3.1.2.b) Quina és la taxa de fallades (T_f) ?

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	22/06/2013	12:00

T_f = 6 fallades / 15 accessos = 0,4

3.1.2.c) Suposem que el temps d'accés a la memòria cau, o temps d'accés en cas d'encert (t_e), és de 4 ns i el temps total d'accés en cas de fallada (t_f) és de 20 ns. Considerant la taxa de fallades obtinguda a la pregunta anterior, quin és el temps mitja d'accés a memòria (t_m) ?

$$t_m = T_f \times t_f + (1\text{-}T_f) \times t_e = 0{,}4 * 20 \text{ ns} + 0{,}6 * 4 \text{ ns} = 8 \text{ ns} + 2{,}4 \text{ ns} = 10{,}4 \text{ ns}$$

3.2

Es vol realitzar la següent comunicació de dades entre la memòria d'un computador i un port USB, que tenen les següents característiques:

- La CPU funciona amb un rellotge de 2GHz de freqüència i executa 1 instrucció per cada 2 cicles de rellotge
- Adreces dels **registres de dades** i **d'estat** del controlador d'E/S: 0A0h i 0A4h
- El bit del **registre d'estat** que indica que el controlador del port d'E/S està disponible és el bit 3, o el quart bit menys significatiu (quan val 1 indica que està disponible)
- Transferència de **escriptura** des de memòria al port d'E/S
- Transferència de $\boldsymbol{N}_{\text{dades}}$=400.000 dades, és a dir, 400.000 **×** 4 Bytes = 1.600.000 Bytes
- Adreça inicial de memòria on resideixen les dades: 20000000h
- La velocitat de transferència del port és de 10.000 Bytes per segon

3.2.1 E/S programada

Completar el següent codi realitzat amb el repertori CISCA que realitza la transferència descrita abans mitjançant la tècnica d'E/S programada.

```
1.              MOV   R3, 400000
2.              MOV   R2, 20000000h
3. Bucle:       IN    R0, [_0A4h_]        ; llegir 4 bytes
4.              AND   R0, _00001000b_
5.              _JE__ Bucle
6.              MOV   R0,[_R2_]           ; llegir 4 bytes
7.              ADD   _R2_, 4
8.              OUT   0A0h, _R0_          ; escriure 4 bytes
9.              SUB   R3, __1__
10.             _JNE_ Bucle
```

Quin és el percentatge de temps que dedica la CPU a la tasca d'Entrada/Sortida?

100%

3.2.2 E/S per Interrupcions

Completar el següent codi CISCA que és una rutina de servei a les interrupcions (RSI) per a transferir a través del dispositiu d'E/S anterior, el mateix nombre de dades que abans amb E/S

Examen 2012/13-2

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	22/06/2013	12:00

programada, però ara mitjançant la tècnica de E/S per interrupcions. Suposeu:

- Es fa servir una variable global que es representa amb l'etiqueta **Dir,** i que al principi del programa conté l'adreça inicial de memòria on resideixen les dades a transferir

```
1.          _CLI_
2.          PUSH  __R0___
3.          _PUSH_ R1
4.          MOV   _R1_, [Dir]
5.          MOV   _R0_, [R1]
6.          OUT   _[0A0h]_,R0  ; escriure 4 bytes
7.          ADD   R1, _4_
8.          MOV   _[Dir]__, R1
9.          POP   __R1__
10.         _POP_ R0
11.         STI
12.         IRET
```

Quin és el percentatge de temps que dedica la CPU a la tasca d'Entrada/Sortida?

1.600.000 Bytes a transferir. 10.000 Bytes per segon. Això fa que el temps total de la transferència sigui de 160 segons.

Cada cicle de rellotge és de 0,5ns. Per tant, cada instrucció triga 1 ns.

Una interrupció necessita 12 instruccions, per tant són 12 ns.

Hi ha 400.000 interrupcions, per tant són 4.800.000 ns. o 4,8 ms.

Això representa un 0,003% del temps total de la transferència.