COGNOMS, NOM:

EXAMEN FINAL D'EC

19 de gener de 2021

L'examen consta de 10 preguntes, que s'han de contestar als mateixos fulls de l'enunciat. No oblideu posar el nom i cognoms a tots els fulls. La durada de l'examen és de 180 minuts. Les notes, la solució i el procediment de revisió es publicaran al Racó.

Problema 1. (1 punt)

Donat el següent codi MIPS

```
f1:  addiu     $sp, $sp, -16
     sw        $s0, 8($sp)
     sw        $ra, 12($sp)
     move      $s0, $a1
     move      $a1, $a0
     move      $a0, $sp
     jal       f2
     lw        $v0, 0($sp)
     lw        $t0, 4($sp)
     addu      $v0, $v0, $t0
     addu      $v0, $v0, $s0
     lw        $s0, 8($sp)
     lw        $ra, 12($sp)
     addiu     $sp, $sp, 16
     jr        $ra
```

i el prototipus en llenguatge C de la funció `f2`

```
void f2(int a[], int b);
```

Completeu el següent codi C amb una funció `f1`, que ha de correspondre amb la subrutina `f1` abans esmentada.

```
int f1(int a, int b) {
    int v[2];

    f2(v, a);

    return v[0] + v[1] + b;


}
```

COGNOMS, NOM:

Problema 2. (1 punt)

Donat el següent programa escrit en llenguatge C, traduïu el programa principal a codi MIPS de forma que s'optimitzi el nombre total d'instruccions executades. La matriu i el vector no cal declarar-los i considereu que ja estan inicialitzats.

```
int M[200][128];
int V[50];

main() {
int sum=0;
    for (i=0; i<50; i++) sum += M[V[i]][2*i];
}
```

```
main:
          li        $t0, 0          # sum
          la        $t1, V          # punter V: val ini
          la        $t2, V+200      # punter V: val fi
          la        $t3, M          # @base M
          li        $t4, 0          # despl. 2*i
bucle:
          lw        $t5, 0($t1)     # accés seq. V
          sll       $t5, $t5, 9
          addu      $t5, $t3, $t5
          addu      $t5, $t5, $t4
          lw        $t5, 0($t5)     # accés aleatori M
          addu      $t0, $t0, $t5
          addiu     $t1, $t1, 4     # act. punter V
          addiu     $t4, $t4, 8     # act. despl. 2*i
          bne       $t1, $t2, bucle

          jr        $ra
```

COGNOMS, NOM:

Problema 3. (1 punt)

Suposem un programa que s'executa sobre un procesador funcionant a una freqüència de 5 GHz, el qual dissipa una potencia de 100W. La següent taula mostra, per a cada tipus d'instrucció, el nombre d'instruccions executades i el CPI, referents a l'execució d'aquest programa:

	Nombre d'instruccions	CPI
Memòria	$10 \cdot 10^{12}$	10
Salts	$5 \cdot 10^{12}$	4
Resta d'instr.	$35 \cdot 10^{12}$	2

Es demana que calculeu

a) El CPI promig de tot el programa

3,8

b) El temps d'execució del programa, en segons

$3{,}8 * 10^4$ s

c) L'energia total consumida durant l'execució del programa, en Joules

$3{,}8 * 10^6$ J

Volem millorar el rendiment del procesador optimitzant la gestió de les instruccions de memòria, amb tècniques més eficients de gestió de la memòria cache.

d) Quin hauria de ser el nou CPI promig de les instruccions de memòria per a obtener un guany de rendiment (speed-up) de 1.25x?

6,2

COGNOMS, NOM:

Problema 4. (1 punt)

Feu un programa en MIPS que modifiqui el registre `$t1` de la següent forma:

- Posi a 1 els bits del 0 al 7
- Intercanvïi els bytes de les posicions 23 a 16 i 15 a 8.
- Posi a 0 els bits del 24 al 31

Recordeu que els bits es numeren del 0 al 31, sent el de menys pes el 0 i el de més pes el 31.

Es valorarà que el programa s'executi amb el menor nombre possible d'instruccions.

```
main:
     andi      $t0, $t1, 0xFF00
     sll       $t0, $t0, 8
     sll       $t1, $t1, 8
     srl       $t1, $t1, 16
     ori       $t1, $t1, 0xFF
     or        $t1, $t0, $t1

     jr        $ra
```

COGNOMS, NOM:

Problema 5. (1 punt)

Un sistema computador amb procesador MIPS gestiona memoria virtual paginada on les pàgines són de 4KB de mida i el reemplaçament usat és LRU. Un programa pot tenir un màxim de 4 pàgines carregades a memoria física.

Considerem que s'està executant un programa que fa servir dades que ocupen 128 KB des de l'adreça base 0x10010000. En un moment donat de l'execució d'aquest programa es tenen les 4 pàgines permeses carregades a memoria física. El contingut de les 4 entrades de la TP que indiquen que la seva página virtual és carregada a memoria física són els següents:

VPN:	PPN	P	D
00400:	10000	1	0
10010:	10001	1	0
10011:	10002	1	1
10012:	10003	1	1

El TLB que s'usa en el sistema és de 2 entrades, completament associatiu i amb reemplaçament LRU. L'ordre en què s'han referenciat les pàgines presents a memoria física és (de la que fa més temps a la que menys i indicades per VPN): 0x10011, 0x10010, 0x10012, 0x00400.

En aquest moment el MIPS és a punt d'executar la instrucció `lw $t0, -8($t1)`, situada a l'adreça 0x00400140. Respecte l'execució d'aquesta instrucció, es demana que indiqueu el contingut en hexadecimal de `$t1` que compleixi la condició establerta en cada apartat. Si hi ha més d'un posible valor indiqueu l'adreça **més baixa** possible. Si no n'hi ha cap de possible, poseu **CAP**.

a) Es detecta un encert de TLB

0x10010008

b) No provoca una fallada de página

0x10010008

c) No es produeix una lectura de página de disc cap a memòria física

0x10010008

d) Es produeix una escriptura de página de memoria física a disc

0x10013008

COGNOMS, NOM:

Problema 6. (1 punt)

Considera la declaració de les variables `u`, `w`, `x`, `y` guardades en `$t1`, `$t2`, `$t3`, `$t4` respectivament:

```
int u, w, x, y;
```

i el següent fragment de codi en llenguatge C:

```
if (u < w)
     y = (u >= x);
else
     y = (u != x);
```

L'hem traduït a assemblador MIPS sense fer servir macros. Completa els següents requadres amb els corresponents mnemònics i operands a fi que la traducció sigui correcta. Tingues en compte que, per la semàntica del llenguatge C, el resultat final de la variable *y* només pot ser 0 o 1.

```
     slt      [ $t0 ] , $t1, $t2
    [ beq  ]  $t0, $zero, else
if:
     slt      $t0, $t1, $t3
    [ sltiu ] $t4, $t0, [ 1 ]
     b        endif
else:
    [ subu ]  $t0, $t1, $t3
     sltu     $t4, [ $zero ] , $t0
endif:
```

COGNOMS, NOM:

Problema 7. (1 punt)

Disposem d'un processador de 32 bits (tant per adreces com per a dades) amb memòria principal adreçable a nivell de byte i dues memòries cache associatives per conjunts, una per a instruccions (MCI) i una altra per a dades (MCD), amb els següents paràmetres:

- Capacitat de la cache de dades: **512 KB**
- Capacitat de la cache d'instruccions: **1024 KB**
- Blocs per conjunt, en totes dues caches: **4 blocs**
- Mida de bloc: **32 paraules**
- Freqüència del rellotge: **200 MHz**
- Temps d'accés a memòria cache, per totes dues caches, en cas d'encert: **2 cicles**
- Temps d'accés a memòria principal per llegir/escriure blocs: **18 cicles**
- Temps d'accés a memòria principal per llegir/escriure paraules: **10 cicles**
- Memòria cache de dades amb una política **d'escriptura retardada amb assignació**.

Quan s'executen un conjunt de programes representatius (benchmark) observem que:

- CPI_{ideal}: **2,5**
- Taxa d'encerts en memòria cache d'instruccions**: 95%**
- Taxa d'encerts en memòria cache de dades: **70%**
- De cada 3 blocs reemplaçats a memòria cache de dades se n'ha modificat 1
- En mitjana, les instruccions que s'executen són
  - load: 10%
  - store: 5%
  - moviment de dades (immediats i registres): 20%
  - aritmètico-lògiques: 35%
  - salt: 20%
  - comparació: 10%

a) Quants bits ocuparà l'etiqueta (tag) dels blocs emmagatzemats a memòria cache de dades?

15

b) En mitjana, quants cicles per instrucció de penalització són motivats per les fallades a la memòria cache d'instruccions? (Indiqueu el nombre arrodonit a dècimes)

1,0

c) En mitjana, quants cicles per instrucció de penalització són motivats per les fallades a la memòria cache de dades? (Indiqueu el nombre arrodonit a dècimes)

1,2

d) Quin és el CPI promig d'aquest sistema processador-memòria obtingut en executar el benchmark? (Indiqueu el nombre arrodonit a dècimes)

3,7

COGNOMS, NOM:

Problema 8. (1 punt)

Considera la següent declaració MIPS de variables globals:

```
a:   .word 0xc4000000
b:   .word 0x43800000
c:   .float 128.0
```

Suposant que s'executa el següent codi:

```
la   $t0, a
lwc1 $f0, 0($t0)
la   $t0, b
lwc1 $f2, 0($t0)
la   $t0, c
lwc1 $f4, 0($t0)
```

Es demana que contesteu quin serà el valor final a `$f6` en hexadecimal després de l'execució dels següents codis:

a)

```
add.s $f6, $f4, $f4
add.s $f6, $f6, $f2
add.s $f6, $f6, $f0
```

$f6 => **0x00000000**

b)

```
add.s $f6, $f2, $f2
sub.s $f6, $f6, $f0
```

$f6 => **0x44800000**

COGNOMS, NOM:

Problema 9. (1 punt)

Donades les següents declaracions de variables globals en assemblador del MIPS:

```
.data                       # adreça base = 0x10010000
a:  .half -1, -7, 8
b:  .word a
.align 3
x:  .space 4
d:  .asciiz "abcde"         # codi ascii ‘a’ = 0x61
e:  .word 256
```

a) Ompliu la següent taula amb el contingut de la memòria, indicant el valor de cada byte EN HEXADECIMAL, i deixant EN BLANC les posicions no ocupades per cap dada.

@Memòria	Dada
0x10010000	**FF**
0x10010001	**FF**
0x10010002	**F9**
0x10010003	**FF**
0x10010004	**08**
0x10010005	**00**
0x10010006	
0x10010007	

@Memòria	Dada
0x10010008	**00**
0x10010009	**00**
0x1001000A	**01**
0x1001000B	**10**
0x1001000C	
0x1001000D	
0x1001000E	
0x1001000F	

@Memòria	Dada
0x10010010	**00**
0x10010011	**00**
0x10010012	**00**
0x10010013	**00**
0x10010014	**61**
0x10010015	**62**
0x10010016	**63**
0x10010017	**64**

@Memòria	Dada
0x10010018	**65**
0x10010019	**00**
0x1001001A	
0x1001001B	
0x1001001C	**00**
0x1001001D	**01**
0x1001001E	**00**
0x1001001F	**00**

b) Quin és el valor de `$t0` en hexadecimal després d’executar el següent codi?

```
la   $t0, b
lw   $t0, 0($t0)
lh   $t0, 2($t0)
```

$t0 => 0xFFFFFFF9

c) Quin és el valor final de `$t0` i de `$t1` en hexadecimal després d’executar el següent codi?

```
li   $t0, -7
li   $t1, 3
div  $t0, $t1
mflo $t0
mfhi $t1
```

$t0 => 0xFFFFFFFE
$t1 => 0xFFFFFFFF

d) Quin és el valor final de `$t0` i de `$t1` en hexadecimal després d’executar el següent codi?

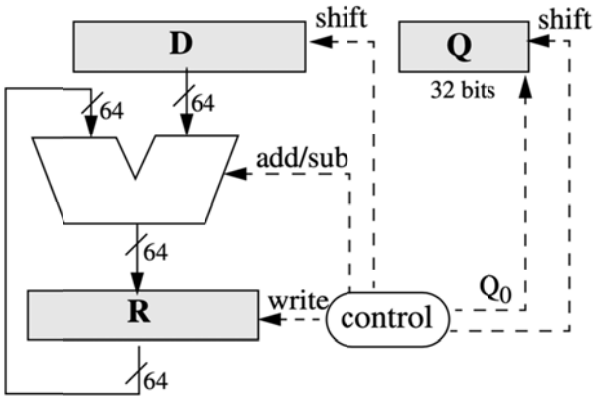
```
li   $t0, 1056
sra  $t1, $t0, 5
xor  $t0, $t0, $t1
subu $t0, $t0, $t1
```

$t0 => 0x000003E0
$t1 => 0x00000021

COGNOMS, NOM:

Problema 10. (1 punt)

Donat el següent diagrama que representa el divisor seqüencial de nombres naturals de 32 bits estudiat a classe i que realitza la divisió X/Y, calculant alhora el quocient i el residu, completeu l'algorisme iteratiu que en descriu el funcionament cicle a cicle:



$D_{63:32}$ = **Y** ;

$D_{31:0}$ = **0** ;

Q = **0** ;

$R_{63:32}$ = **0** ;

$R_{31:0}$ = **X** ;

```
for (i=1; i<=32 ; i++) {
```

```
D = D >> 1;
R = R - D;
if (R63 == 0) {
     Q = (Q << 1) | 1;
} else {
     R = R + D;
     Q = Q << 1;
}
```

```
}
```