COGNOMS:	GRUP:
NOM:	

EXAMEN FINAL D'EC
15 de juny de 2015

L'examen consta de 10 problemes, que s'han de contestar als mateixos fulls de l'enunciat. No oblidis posar el teu nom i cognoms a tots els fulls. La duració de l'examen és de 180 minuts. Les notes i la solució es publicaran al Racó el dia 26 de juny. La revisió es farà el 29 de juny a les 17h.

Problema 1. Accés Seqüencial (0,75 punts)

Donat el següent codi en alt nivell:

```
int M[4][10];

main() {
int i;
       for (i=0; i<3; i++) {
              M[i][9-i] = M[i+1][9-i];
       }
}
```

Completa el següent codi en assemblador del MIPS, que fa el mateix que el programa donat, utilitzant la tècnica d'accés seqüencial.

```
main:
        li $t0, 0
        li $t1, [          ]
        la $t2, [          ]
bucle:
        bge     $t0, $t1, fibucle
        lw      $t4, [          ]     ($t2)
        sw      $t4, 0($t2)
        addiu   $t2, $t2, [          ]
        addiu   $t0, $t0, 1
        b       bucle
fibucle:
        jr $ra
```

Problema 2. Anàlisi (1,5 punts)

Donades les següents declaracions de dades globals en assemblador del MIPS, que s'ubiquen en memòria a partir de l'adreça 0x10010000

```
        .data
a:      .ascii "abc"            # El codi ascii de la 'a' és 0x61
b:      .half  -7
c:      .word  a
d:      .byte  0x03
        .align 1
e:      .byte  0xA1, 0xA2, 0xA3, 0xA4, 0xA5
f:      .float 3.25
```

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

@Memòria	Dada
0x10010000	
0x10010001	
0x10010002	
0x10010003	
0x10010004	
0x10010005	
0x10010006	
0x10010007	

@Memòria	Dada
0x10010008	
0x10010009	
0x1001000A	
0x1001000B	
0x1001000C	
0x1001000D	
0x1001000E	
0x1001000F	

@Memòria	Dada
0x10010010	
0x10010011	
0x10010012	
0x10010013	
0x10010014	
0x10010015	
0x10010016	
0x10010017	

@Memòria	Dada
0x10010018	
0x10010019	
0x1001001A	
0x1001001B	
0x1001001C	
0x1001001D	
0x1001001E	
0x1001001F	

b) Quin és el contingut final de `$t0` en hexadecimal després d'executar el següent codi?

```
li      $t0, 5123
li      $t1, 10
div     $t0, $t1
mflo    $t0
mfhi    $t1
addu    $t0, $t0, $t1
```

`$t0 =` ____

c) Quin és el contingut final de `$t0` en hexadecimal després d'executar el següent codi?

```
li      $t0, -3
sra     $t1, $t0, 31
xor     $t0, $t0, $t1
subu    $t0, $t0, $t1
sll     $t1, $t1, 31
or      $t0, $t0, $t1
```

`$t0 =` ____

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

```
la      $t0, c
lw      $t0, 0($t0)
lh      $t0, 4($t0)
```

`$t0 =` ____

COGNOMS:	GRUP:
NOM:	

Problema 3. Test (1 punt)

Posa una X al costat de cada una de les següents afirmacions (a la columna V si és Verdadera o a la columna F si és Falsa). Suposem en tots els casos que es fa referència a un processador MIPS com l'estudiat a classe. Cada resposta correcta suma 0,1 punts; les respostes no contestades no es tenen en compte; cada resposta incorrecta resta 0,1 punts; i la puntuació total mínima és 0.

	Afirmació	V	F
1.-	Si l'accés a dades d'un *store* produeix un encert al TLB, però el bit D val 0, llavors es produeix una excepció		
2.-	Si al traduir una adreça amb el TLB es troba una entrada amb el mateix VPN però amb el bit V que val 0, llavors es produeix una fallada de pàgina		
3.-	Si el bit EXL val 1, les excepcions seran ignorades		
4.-	Si s'executa la instrucció tlbwr en mode usuari, es produeix una excepció		
5.-	Després d'una fallada de TLB la instrucció causant de l'excepció s'ha de reexecutar		
6.-	Una rutina de servei a excepcions pot acabar amb `jr $epc` doncs al registre `$epc` es guarda l'adreça de la instrucció que cal seguir executant un cop s'ha atès l'excepció		
7.-	Segons la representació de números en coma flotant IEEE-754 amb simple precisió el valor més gran representable just per sota del +infinit es representa `0x7f7fffff`		
8.-	El bit D del TLB en el processador MIPS estudiat al tema 8 és sempre coherent amb el bit D de la TP.		
9.-	Una excepció no pot ser atesa fins que la instrucció en curs hagi finalitzat		
10.-	En el MIPS, una mateixa instrucció pot causar durant la seva execució 2 fallades de pàgina		

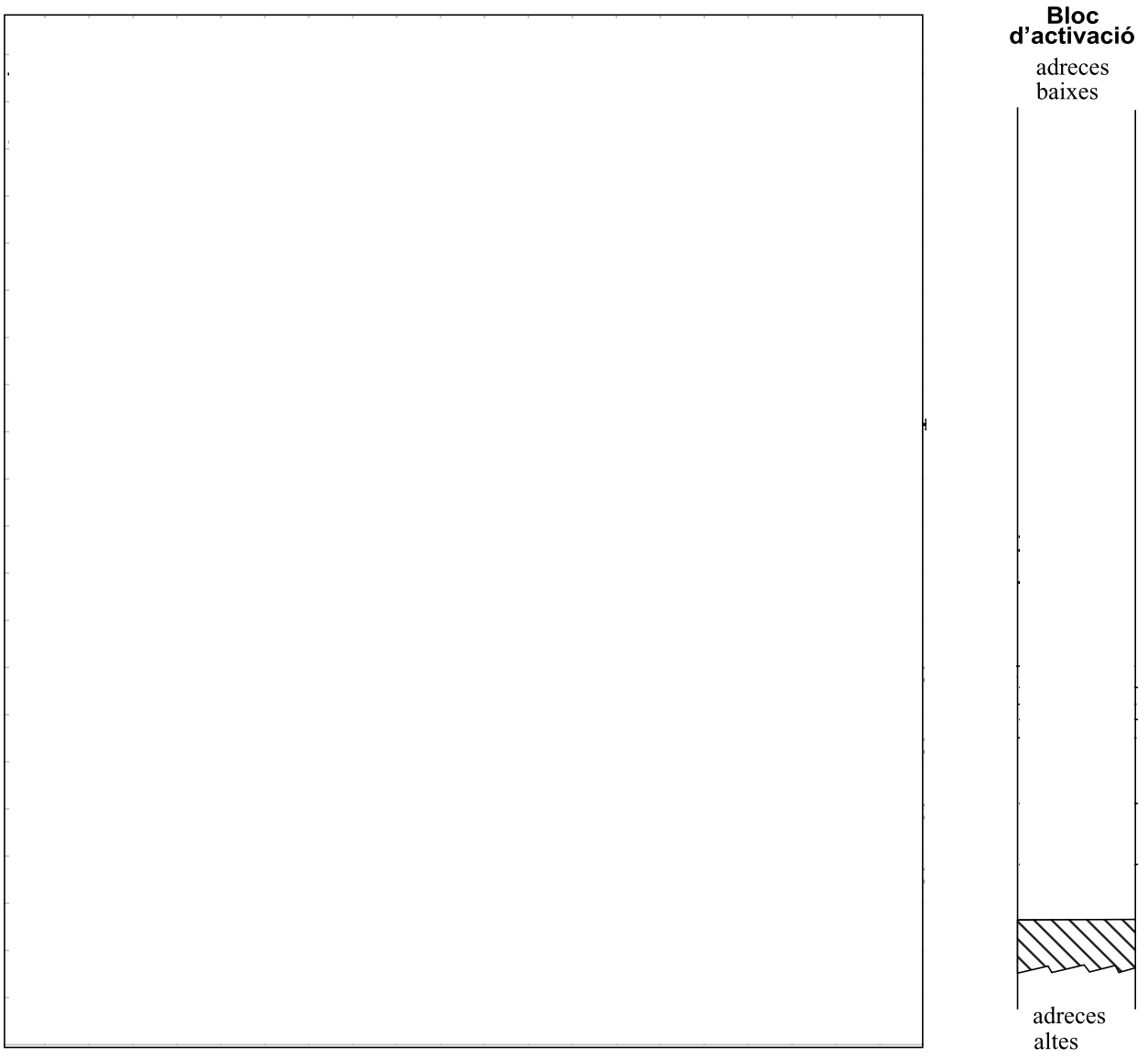
Problema 4. Subrutines (1,5 punts)

Donada la següent declaració de funcions en C:

```
char f(int j, char *p);
char g(char c);
char examen(int i, char vec[]) {
    char val, aux;
       aux = f(i, &val);
       return val + vec[g(aux) + aux];
}
```

a) Seguint les regles de l'ABI estudiades, si volem salvar a la pila el mínim nombre de registres, ¿quines variables, paràmetres o càlculs intermedis de la funció `examen` cal guardar obligatòriament en registres segurs (`$s`) per garantir que no siguin alterats per les crides a `f` o `g`?

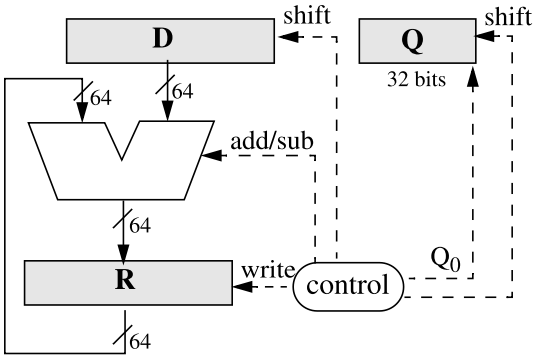
b) Tradueix a assemblador MIPS la subrutina `examen`, i dibuixa el bloc d'activació, especificant-hi la posició on apunta el registre `$sp` un cop reservat l'espai corresponent a la pila, així com el nom de cada registre i/o variable guardada, i la seva posició (desplaçament relatiu al `$sp`).



COGNOMS:	GRUP:
NOM:	

Problema 5. Aritmètica (0,5 punts)

Sigui el següent diagrama simplificat del circuit per a la divisió de nombres naturals de 32 bits, anàleg al que has estudiat a classe:



Completa el següent algorisme de divisió, en pseudocodi amb operadors en C, el qual expressa el funcionament del circuit per a obtenir el quocient QUO = X/Y i el residu RES = X%Y, on X,Y són operands naturals de 32 bits:

```
/* En quins bits de R i D cal escriure X i Y inicialment? */
R[63:32] = [          ] ;
R[31:0]  = [          ] ;
D[63:32] = [          ] ;
D[31:0]  = [          ] ;
Q        = 0;
for (i=1; i<=32; i++) {
    [                                                    ]
}
QUO      = Q;
RES      = R[      ] /* En quins bits de R està el residu? */
```

Problema 6. Condicional (0,5 punts)

Donades les següents variables locals

```
int aux, j;   /* guardades a $t0 i $t1 respectivament */
```

i el següent codi en alt nivell

```
if ((0 <= aux) && (aux <= j))
       aux = aux+1;
```

Completa el següent fragment de codi MIPS omplint les caselles en blanc perquè sigui equivalent a l'anterior codi en alt nivell:

```
        [          ]    $t0, $zero, fi
        [          ]    $t0, $t1, fi
          addiu          $t0, $t0, 1
fi:
```

Problema 7. Memòria Cache (1,25 punts)

Suposem que tenim un processador de 32 bits amb una memòria cache de dades de 512 bytes, on cada bloc té 16 bytes. Suposem que executem els següents programes.

```
//programa A

int M[4][128];

void main() {
int i, j; //en registres
  for (i=0;i<4;i++)
    for (j=0;j<128;j++)
      M[i][j]= 0;
}
```

```
//programa B

int M[4][128];

void main() {
int i, j; //en registres
  for (j=0;j<128;j++)
    for (i=0;i<4;i++)
      M[i][j]= M[i][j]+1;
}
```

Calcula el nombre de fallades de la cache suposant que la memòria cache és inicialment buida. L'adreça base de la matriu `M` és 0.

a) Suposant que la cache és de correspondència directa i té la política d'escriptura retardada amb assignació.

Fallades A = [ ] Fallades B = [ ]

b) Suposant que la cache és associativa per conjunts de 4 vies (algorisme de reemplaçament LRU), i que té la política d'escriptura immediata sense assignació.

Fallades A = [ ] Fallades B = [ ]

COGNOMS:	GRUP:
NOM:	

Problema 8. Coma Flotant (1 punt)

Donada la instrucció `add.s $f0, $f2, $f4` i que els continguts de `$f2` i `$f4` són `0x44400004` i `0x42400002`, respectivament.

a) Quin és el contingut de `$f0` en hexadecimal després d'executar la instrucció donada?

`$f0 =` [ ]

b) Quin és el valor de l'error que s'ha produït, mesurat com el valor absolut de la diferència entre el valor correcte de la suma i el valor finalment representat. Expressa aquest error en notació científica i en decimal, omplint les caselles corresponents a la mantissa i a l'exponent.

`error =` [ ] `* 2` [ ]

Problema 9. Rendiment (0,75 punts)

Un processador disposa de 4 tipus d'instruccions diferents: A, B, C i D. La següent taula mostra quin és el número d'instruccions executades per a un programa sota consideració i el CPI de cada tipus d'instrucció. El processador té un rellotge a 2GHz.

tipus d'instrucció	nombre d'instruccions	CPI
A	$8*10^9$	1
B	$6*10^9$	2
C	$4*10^9$	1
D	$2*10^9$	8

a) Calcula el CPI mitjà del programa sota consideració.

CPI = [ ]

b) Indica quin és el temps d'execució (en segons) del programa sota consideració.

temps = [ ] s.

c) Indica quin seria el guany (speed-up) obtingut si s'aconseguís reduir el CPI de les instruccions de tipus D a 4 cicles.

guany = [ ]