EXAMEN PARCIAL D'EC
20 d'abril de 2021

- L'examen consta de 7 preguntes, que s'han de contestar als mateixos fulls de l'enunciat. No oblidis posar el teu nom i cognoms a tots els fulls.
- La durada de l'examen és de 1:30 hores (90 minuts)
- Les notes, la solució i el procediment de revisió es publicaran al Racó el dia 3 de maig.

Pregunta 1 *(1 punt)*
Donada la següent subrutina en assemblador MIPS:

```
acces_aleatori:
        li      $t0, 400
        mult    $t0, $a1
        mflo    $t0
        addu    $t0, $t0, $a0
        lw      $v0, -80($t0)
        jr      $ra
```

Sabem que és la traducció de la següent funció en C (de la qual desconeixem els valors dels requadres). Completa els requadres amb les expressions vàlides en C per tal que la traducció sigui correcta:

```
int acces_aleatori (int M[][100], int i)
{
        return M[ i-1 ][ 80 ];
}
```

Pregunta 2 *(1.25 punts)*
Donada la següent funció `foo` en alt nivell correcte (no cal comprovar si se surt de rang), on `M` i `V` són declarades com a variables globals:

```
int M[100][100]; /* suposarem que està inicialitzada */
int V[100];
void foo() {
        int i,aux; /* ocupen els registres $t1 i $t3 respectivament */
        for (i=0; i<97; i++) {
                aux = M[i][i+3];
                M[i+1][i+3] = V[aux];
        }
}
```

Completa el següent codi MIPS omplint les caselles en blanc perquè sigui equivalent a l'anterior codi en alt nivell, tenint en compte que els elements de la matriu s'accedeixen utilitzant la tècnica **d'accés seqüencial**, usant el registre `$t0` com a punter per accedir els elements de la matriu.

```
        la      $t0, M + 12
        li      $t1, 0
        li      $t2, 97
        la      $t7, V
bucle:  bge     $t1, $t2, fibuc
        lw      $t3, 0($t0)     #hi ha variants: aquesta casella i la primera han de sumar 12
        sll     $t3, $t3, 2
        addu    $t3, $t3, $t7
        lw      $t4, 0($t3)
        sw      $t4, 400($t0)   #hi ha variants: aquesta casella i la primera han de sumar 412
        addiu   $t0, $t0, 404
        addiu   $t1, $t1, 1
        b bucle
fibuc:
```

Pregunta 3 *(0.75 punts)*
Donades les següents declaracions de variables locals

```
unsigned int a, b, c;
int d;
```

i el següent codi en alt nivell

```
if (((a<=b) && (a>c)) || (d!=0))
      d = -1;
else
      d = 0;
```

Sabent que les variables a, b, c, d estan guardades en els registres `$t0`, `$t1`, `$t2`, `$t3` respectivament, completa el següent fragment de codi MIPS omplint les caselles en blanc perquè sigui equivalent a l'anterior codi en alt nivell (*nota: posa atenció al tipus de les variables!*):

```
et1:   bgtu  $t0, $t1, et3
et2:   bgtu  $t0, $t2, et4
et3:   beq   $t3, $zero, et6
et4:   li    $t3, -1
et5:   b     et7
et6:   move  $t3, $zero
et7:
```

Pregunta 4 *(2 punts)*
Donat el següent fragment de codi en assemblador MIPS:

```
li     $t1, 0xC0580000
mtc1   $t1, $f2
li     $t2, 0x42000002
mtc1   $t2, $f4
li     $t3, 0xC0000009
mtc1   $t3, $f6
add.s  $f8, $f4, $f6
```

a) Quin valor representa, en decimal en notació científica normalitzada, el contingut que hi ha a `$f2`? *(0.5 punts)*

-3.375

b) Quina és la mantissa (en binari) i l'exponent (en decimal) dels nombres que hi ha a `$f4` i `$f6`? *(0.25 punts)*

		mantissa(binari)	exponent (decimal)
$f4:	1 ,	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0	5
$f6:	- 1 ,	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1	1

c) Volem fer la suma entre els registres `$f4` i `$f6`. Què valdrien els bits GRS i el seu càlcul en la operació? Omple les caselles que falten dels bits GRS amb els valors corresponents. Els valors de les operacions ja estan ajustats al nombre amb major exponent. *(0.25 punts)*

	G	R	S
1 , 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0	0	0	0
– 0 , 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1	0	1
x x , x x x x x x x x x x x x x x x x x x x x x x x x	0	1	1

d) Quin és el contingut de `$f8` en hexadecimal després d'executar la instrucció `add.s`? *(1 punt)*

`$f8 =` 0x**41F00003**

Pregunta 5 *(1.5 punts)*
Donada la següent declaració de variables globals, que s'ubica a memòria a partir de l'adreça 0x10010000:

```
     .data
v1: .byte    -5
v2: .half    0x80
v3: .dword   0xFEDCBA9876543210
v4: .asciiz  "EC"  # codi ASCII 'A' = 0x41
    .align   2
v5: .space   8
v6: .word    v1
```

a) Omple la següent taula amb el contingut de memòria **en hexadecimal**. Les variables s'emmagatzemen a partir de l'adreça 0x10010000. Les posicions de memòria sense inicialitzar es deixen en blanc.

@Memòria	Dada	@Memòria	Dada	@Memòria	Dada	@Memòria	Dada
0x10010000	**0xFB**	0x10010008	**0x10**	0x10010010	**0x45**	0x10010018	**0x00**
0x10010001		0x10010009	**0x32**	0x10010011	**0x43**	0x10010019	**0x00**
0x10010002	**0x80**	0x1001000A	**0x54**	0x10010012	**0x00**	0x1001001A	**0x00**
0x10010003	**0x00**	0x1001000B	**0x76**	0x10010013		0x1001001B	**0x00**
0x10010004		0x1001000C	**0x98**	0x10010014	**0x00**	0x1001001C	**0x00**
0x10010005		0x1001000D	**0xBA**	0x10010015	**0x00**	0x1001001D	**0x00**
0x10010006		0x1001000E	**0xDC**	0x10010016	**0x00**	0x1001001E	**0x01**
0x10010007		0x1001000F	**0xFE**	0x10010017	**0x00**	0x1001001F	**0x10**

b) Calcula el valor final del registre $t0 en hexadecimal després d'executar el següent codi.

```
la   $t1, v1
lb   $t1, 0($t1)
sra  $t2, $t1, 4
sltu $t0, $t1, $t2
```

$t0 = 0x**00000001**

Pregunta 6 *(1 punt)*
S'executa un programa de test en un ordinador que té 3 tipus d'instruccions (A,B,C), amb CPI diferents. La següent taula especifica el nombre d'instruccions de cada tipus que executa el programa i el seu CPI:

Tipus d'instrucció	Nombre d'instruccions	CPI
A	$2*10^9$	1
B	$1*10^9$	1
C	$1*10^9$	2

a) Sabem que la freqüència de rellotge és de 1GHz i que la potència dissipada és P=100W. Calcula el temps d'execució en segons i l'energia consumida en Joules durant l'execució del programa de test.

t_{exe} = **5 s**

E = **500 J**

b) S'implementen unes millores al CPU que permeten incrementar la freqüència de rellotge a 2GHz. No obstant, el CPI de les instruccions de tipus B passa a ser de 4 cicles. Quin és el guany (speedup) de rendiment obtingut?.

Guany = **1.25**

Pregunta 7 *(2.5 punts)*
Donat el següent programa en C:

```
char G[7] = "EXAMEN";

int main() {
  char L[7];
  int i = 0;
  char x = 1;
  while(G[i] != '\0') {
    x = func(&G[i], L, i, x);
    i++;
  }
}

char func(char *a, char P[7], int b, char c) {
    P[b] = *a + c;
    return c + 1;
}
```

Completa la següent traducció a llenguatge assemblador MIPS omplint les caselles en blanc:

```
.data
G: .asciiz "EXAMEN"
.text
main:
  addiu $sp, $sp, -16
  sw    $s0, 8($sp)
  sw    $ra, 12($sp)
  li    $s0, 0
  li    $v0, 1
while:
  la    $a0, G
  addu  $a0, $a0, $s0
  lb    $t0, 0($a0)
  beq   $t0, $zero, endwhile
  move  $a1, $sp
  move  $a2, $s0
  move  $a3, $v0
  jal   func
  addiu $s0, $s0, 1
  b     while
endwhile:
  lw    $s0, 8($sp)
  lw    $ra, 12($sp)
  addiu $sp, $sp, 16
  jr $ra


func:
   lb    $t0, 0($a0)
   addu  $t0, $t0, $a3
   addu  $t1, $a1, $a2
   sb    $t0, 0($t1)
   addiu $v0, $a3, 1
   jr    $ra
```