Examen parcial d'FCO – Temes 5 i 6

10 de Gener de 2020

COGNOMS: _____ NOM: _____

DNI: _____GRUP: _____ SIGNATURA:

Normativa:

- La durada de l'examen és de 2 hores.
- Escriviu el nom i cognoms i GRUP en lletres MAJÚSCULES i signeu en TOTS els fulls.
- Heu de respondre dins l'espai assignat.
- No es permeten calculadores ni apunts.
- Heu de romandre en silenci durant la realització de l'examen.
- No es pot abandonar l'examen fins que el professor ho indique.
- Heu de tenir una identificació damunt la taula, a la vista del professor (DNI, carnet UPV, targeta de resident, etc.).

1.- **(1,5 punts)** Completeu la taula següent:

Decimal	Signe i Magnitud 8 bits	Complement a 2 8 bits	Excés 127 8 bits
-7	10000111	11111001	01111000
-13	10001101	11110011	01110010
Rang	[-127, +127]	[-128 ,+127]	[-127,+128]

Nota: Indiqueu el rang en decimal

2.- **(1 punt)** Per als nombres enters A = 111111_{Ca2} i B = 100000_{Ca2} representats ambdós en complement a 2 amb 8 bits, realitzeu les operacions següents sense canviar de representació. Indiqueu clarament i justifiqueu si hi ha o no desbordament. Mostreu el detall de la vostra solució.

a) A + B **(0,4 punts)**

Com que es tracta d0una suma, els nombres es sumen tal i com estan:

10 0000. (Bits deròssec)

111111 (A)

+100000 (B)

011111 (es descarta l'últim bit de ròssec)

Com que els dos últims bit de ròssec són diferents, hi ha desbordament i el resultat no es representable en complement a 2 amb 8 bits.

El resultat de A + B és: No representable, fora de rang, desbordament.

b) B - A **(0,6 punts)**.

Convertim la resta en una suma fent l'operació B+Ca2 (A), és a dir, canviem el signe al subtrahend fent-li el complement a dos, i per tant la resta es converteix en una suma B - A = B + (-A)

Ca2(A) = Ca2(111111) = 000001

Ara realitzem l'operació de suma A + Ca2(B)

00 0000

100000

+000001

100001

Com que els dos últims bit de ròssec són iguals, no hi ha desbordament i el resultat és vàlid, essent: B-A= 100001_{Ca2} y es igual a -31_{10}

3.- **(1 punt)** Per al nombre real la representació del qual en format IEEE-754 de simple precisió és la seqüència de bits R = 0xC1F10000, obteniu el seu valor en decimal. Mostreu els passos realitzats

Signo = 1, Negativo
Magnitud = 1,1110001

Exponente + Exceso = Número Representado

Exceso=127
Número Representado = 131

Exponente = 131 – 127 = 4

La solución es: (R= Signo Magnitud * $2^{Exponente}$)

Signo: -
Magnitud: 1,1110001
Exponente (en decimal): 4

-11110,001 x 2^0

SOLUCIÓN: R=$-30,125_{10}$

4.- **(5,5 punts)** Considerant el programa següent, escrit en llenguatge d'assemblador del MIPS R2000, responeu les qüestions que es presenten a continuació:

```
            .globl __start
            .data 0x10001000
       Tam: .byte 5
        A1: .half 5, 8, -7, 9, 4
  cartell: .asciiz "GII-fco"
        A2: .byte 4, 7, -6, 8, 5
resultat: .word 0

            .text 0x00400100
  __start: la $2, Tam
           lb $2,0($2)
           li $10, 0
           la $3, A1
           la $4, A2

 mentre:   beq $2, $0, fi
           lh $5, 0($3)
           lb $6, 0($4)
           sub $8, $5, $6
           mult $8, $8
           mflo $8
           add $10, $10, $8
           addi $2,$2,-1
           addi $3, $3, 2
```

```
        addi $4,$4, 1
        j mentre

fi:     la $2, Tam
        lb $2, 0($2)
        div $10, $2
        mflo $10
        la $2, resultatt
        sw $10,0($2)
        .end
```

a) **(0,5 punts**) Indiqueu el contingut del segment de dades abans d'executar el programa. Teniu en compte que les dades s'emmagatzemen en format "little endian". Heu d'indicar el contingut de cadascun dels bytes en hexadecimal. Indiqueu les zones de memòria de contingut desconegut amb un interrogant o un guionet.

31 … 24	23 … 16	15 … 8	7 … 0	adreça
0x00	0x05	-	0x05	0x10001000
0xff	0xf9	0x00	0x08	0x10001004
0x00	0x04	0x00	0x09	0x10001008
'-'	'l'	'l'	'G'	0x1000100c
0x00	'o'	'c'	'f'	0x10001010
0x08	0xfa	0x07	0x04	0x10001014
-	-	-	0x05	0x10001018
0x00	0x00	0x00	0x00	0x1000102c

b) **(1 punt)** Indiqueu el contigut en hexadecimal dels registres següents **després** d'executar per **primera** vegada la instrucció `j mentre`.

Registre	Contingut Hexadecimal
$2	0x00000004
$3	0x10001004
$4	0x10001015

$5	0x00000005
$6	0x00000004

c) **(1 punt)** Indiqueu el contingut en hexadecimal dels registres següents en finalitzar l'execució del codi

Registre	Contingut Hexadecimal
$3	0x1000100c
$4	0x10001019
$5	0x00000004
$6	0x00000005
$10	0x00000001

d) **(0,5 punts)** Indiqueu tan sols les zones de memòria que s'hagen modificat després de l'execució del programa. Expreseu el valor de cadascun dels bytes en hexadecimal.

31 … 24	23 … 16	15 … 8	7 … 0	Adreça
0x00	0x00	0x00	0x01	0x1000102c

e) **(0,5 punts)** Per medi d'una eqüació indiqueu quin és el resultat que s'emmagatzema al registre $10.

$$f(A1[\,], A2[\,]) = \frac{\left(\sum_{i=0}^{n-1} (\mathrm{A1[i]} - \mathrm{A2[i]})^2\right)}{5}$$

f) **(0,5 punts)** Quin és el resultat emmagatzemat en $10 si es canvien el valors del vector A1 pels següents bytes?

```
A1: byte 3,6,-9,11,6
```

El valor emmagatzemat en $10 és 4.

g) **(0,5 punts)** Indiqueu en hexadecimal les adreces associades a les etiquetes següents:

Etiqueta	Adreça
Tam	0x10001000
A1	0x10001002
A2	0x10001014
__start	0x00400100
mentre	0x00400120

h) **(0,5 punts)** Indiqueu la seqüència d'instruccions per les que l'assemblador del MIPS R2000 traduiria la pseudoinstrucció:

```
la $4, A2
```

lui $1, 0x1000

ori $4, $1, 0x1014

NOTA: Heu d'utilitzar el registre $1 per als càlculs intermedis.

i) **(0,5 punts)** Codifiqueu la instrucció `div $10,$2`. Indiqueu el resultat tant en binari com en hexadecimal i mostreu els passos realitzats.

Instrucció tipus R: COP Op1 Op2 Dest Desp funció

COP = 0x00 funció = 0x1A

Op1= $10 Op2=$2 dest = $0 Desp=0x00

Binari: 000000 01010 00010 00000 00000 011010

Hexadecimal: 0x0142001a

5.- (1 punt) Escriviu un programa en llenguatge assemblador del MIPS R2000 que realitze l'operació:

var_d= (var_a *var_b) / var_c

seguint les instruccions següents:

- Les dades "var_a", "var_b" i "var_c" han de definir-se com a enters de 8 bits a partir de l'adreça de memòria 0x10000000
- Seguidament es reservarà espai per a emmagatzemar el resultat, var_d, com a enter de 32 bits
- Considereu per a inicialitzar les variables els valors següents:
  - var_a= 20
  - var_b= 10
  - var_c= 5

- Etiqueteu en la memòria de dades cada dada amb el seu valor alfabètic, és a dir, var_a, var_b, var_c i var_d.

Solució

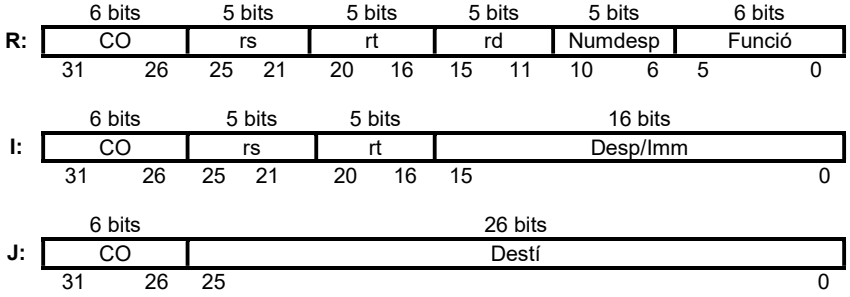
```
.globl __start

.data 0x10001000

var_a: .byte 20
var_b: .byte 10
var_c: .byte 5
res: .word 0

.text 0x00400000
__start:  la $2,var_a
     lb $4,0($2)
```

```
        la $2,var_b
        lb $5,0($2)
        la $2,var_c
        lb $6,0($2)

        mult $4,$5
        mflo $10
        div $10,$4

        mflo $10
        la $2,res
        sw $10,0($2)
.end
```

Sintaxi i tipus de les instruccions

Sintaxi	Format	Descripció
add rd, rs, rt	R	rd ← rs + rt
addi rt, rs, imm	I	rt ← rs + imm, la dada immediata és de 16 bits i s'estén el signe
sub rd, rs, rt	R	rd ← rs – rt
mult rs, rt	R	Multiplica rs per rt deixant els 32 bits de major pes en el registre HI i els 32 bits de menor pes en LO
div rs, rt	R	Dividix rs entre rt deixant el quocient en el registre LO i la resta en el registre HI
and rd, rs, rt	R	rd ← rs and rt, l'operació lògica indicada es realitza bit a bit
nor rd, rs, rt	R	rd ← rs nor rt, l'operació lògica indicada es realitza bit a bit
xor rd, rs, rt	R	rd ← rs xor rt, l'operació lògica indicada es realitza bit a bit
or rd, rs, rt	R	rd ← rs or rt, l'operació lògica indicada es realitza bit a bit
andi rt, rs, imm	I	rt← rs and imm, dada immediata de 16 bits que s'estén amb 16 zeros
ori rt, rs, imm	I	rt← rs or imm, dada immediata de 16 bits que s'estén amb 16 zeros
xori rt, rs, imm	I	rt← rs xor imm, dada immediata de 16 bits que s'estén amb 16 zeros
sll rd, rt, desp	R	rd← rt << desp, desplaçament a esquerres, conforme desplaça s'ompli amb 0
srl rd, rt, desp	R	rd← rt >> desp, desplaçament a dretes, conforme desplaça s'ompli amb 0
sra rd, rt, desp	R	rd← rt >> desp, desplaçament a dretes, conforme desplaça s'estén el bit de signe
Sintaxi	Format	Descripció
lw rt, desp(rs)	I	rt ← M[desp+rs], carrega una paraula de 32 bits. El desplaçament (desp), és de 16 bits i s'estén el signe
lh rt, desp(rs)	I	rt ← M[desp+rs], carrega mitja paraula (16 bits) i estén el signe
lb rt, desp(rs)	I	rt ← M[desp+rs], carrega un byte (8 bits) i estén el signe
sw rt, desp(rs)	I	M[desp+rs] ← rt
sh rt, desp(rs)	I	M[desp+rs] ← rt, emmagatzema la part baixa (16 bits) de rt en memòria
sb rt, desp(rs)	I	M[desp+rs] ← rt, emmagatzema el byte menys significatiu de rt en memòria
lui rt, imm	I	rt31...16 ← imm, rt15...0 ←0
Sintaxi	Format	Descripció
mfhi rd	R	rd ← HI
mflo rd	R	rd ← LO
mthi rs	R	HI ← rs
mtlo rs	R	LO ← rs
Sintaxi	Format	Descripció
slt rd, rs, rt	R	Si (rs < rt) llavors rd ← 1 si no rd ← 0
slti rt, rs, imm	I	Si (rs < imm) llavors rt ← 1 si no rt ← 0
Sintaxi	Format	Descripció
beq rs, rt, etiqueta	I	Si (rs == rt) PC← etiqueta. Si es complix la condició bota a l'adreça etiqueta
bne rs, rt, etiqueta	I	Si (rs != rt) PC← etiqueta. Si es complix la condició bota a l'adreça etiqueta
Sintaxi	Format	Descripció
j etiqueta	J	PC← etiqueta, bota a l'adreça etiqueta
jal etiqueta	J	$31← PC+4, PC← etiqueta, bota a l'adreça etiqueta guardant-se prèviament l'adreça de retorn en $31
jr rs	R	PC ← rs, bota a l'adreça continguda en el registre rs

Codificació segons el format



Codis d'operació i funció

Instrucció	CO
addi	0x08
andi	0x0C
beq	0x04
bne	0x05
j	0x02
jal	0x03
lb	0x20
lh	0x21
lui	0x0F
lw	0x23
ori	0x0D
sb	0x28

6 bits

Instrucció	CO
sh	0x29
sw	0x2B
xori	0x0E

6 bits

Instrucció	CO	Funció
add	0x00	0x20
and	0x00	0x24
div	0x00	0x1A
jr	0x00	0x08

6 bits 6 bits

Instrucció	CO	Funció
mfhi	0x00	0x10
mflo	0x00	0x12
mthi	0x00	0x11
mtlo	0x00	0x13
mult	0x00	0x18
nor	0x00	0x27
or	0x00	0x25
sll	0x00	0x00
slt	0x00	0x2A
srl	0x00	0x02
sub	0x00	0x22
xor	0x00	0x26

6 bits 6 bits

Pseudoinstruccions

Pseudoinstrucció (Sintaxi)	Descripció
li rd, imm	rd ← $imm_{32\ bits}$
la rd, etiqueta	rd ← $etiqueta_{adreça\ 32\ bits}$

Conveni MIPS

Nom	Nº
`$zero`	0
`$at`	1
`$v0,$v1`	2,3
`$a0,..,$a3`	4,..,7

Nom	Nº
`$t0,…,$t7`	8, …, 15
`$s0,…,$s7`	16,…,23
`$t8,$t,0`	24,25
`$k0,$k1`	26,27

Nom	Nº
`$gp`	28
`$sp`	29
`$fp`	30
`$ra`	31