Universitat Autònoma de Barcelona
Escola d'Enginyeria
Fonaments d'Informàtica **COMPUTACIÓ** **16 de gener de 2020**

Cognoms:	**NIU**:
Nom:	**Grup**:

Responeu les preguntes de la prova en l'espai habilitat, no s'acceptaran fulls apart.

L'examen es fa sobre 12 punts.

Llegiu l'enunciat de cada pregunta amb cura i contesteu només el que es pregunta, però justifiqueu sempre la vostra resposta.

Teoria 1. ***(1.5 punts)***

Responeu breument les següents preguntes:

a) Què és el Registre Comptador de Programa (PC)?

Registre que conté l'adreça de la següent instrucció a ser executada.

b) Què és un Mode d'adreçament (enumerar-los **no** és una resposta)

És la funció que genera l'adreça efectiva de la posició de memòria on és l'operand d'una instrucció, a partir de l'adreça explicita que apareix a la instrucció.

e) Què són el mode usuari i el mode supervisor?

Estats en que pot operar una CPU amb diferents capacitats. El mode usuari està pensat per executar els programes dels usuaris i el mode supervisor per executar el sistema operatiu.

En mode usuari no es pot accedir a tot el mapa de memòria, ni es pot executar qualsevol instrucció del repertori, n'hi ha de privilegiades, no es pot accedir als ports d'entrada/sortida i no es té accés a determinats registres de la CPU.

En mode supervisor no hi ha cap de les limitacions esmentades en el mode usuari.

Teoria 2. ***(1,5 punts)***

Apliqueu els conceptes estudiats al curs per respondre les preguntes següents:

a) Quin és el nombre màxim (en Megues) de cel·les (posicions) que pot tenir la memòria principal si el bus d'adreces té 32 línies (bits)? **(0,5 punts)**

32 bits de bus ➔ 2^{32} posicions de memòria. si 1 Mega = 2^{20} ➔ 2^{12} Megues = 4096 Megues

b) Quin és l'efecte d'executar el següent programa? (El PC = 50 inicial) (**1 punt**). Si descriviu el programa línia per línia serà considerat incorrecte. Volem l'efecte general, global. Calcula alguna cosa? Quan acaba? Com canvia la pila?

```
50      MOV RPG, #25     ;
51      SHL RPG                  ;
52      CALL(JSR)  1000 ;
53      MOV $100, RPG            ;

1000    PUSH RPG
1001    RTN              (RETORN)
```

Guarda un 25 al registre RPG que el converteix en un 50 (SHL) i va a una rutina on posa el 50 a la capçalera de la pila, queda per sobre del PC guardat pel "CALL". D'aquesta manera quan retorna de la rutina, "RET", en lloc de tornar a la 53 torna a la 50 i això ens fa un bucle indefinit.

Teoria 3. ***(1 punt)***

Descriviu, a nivell de transferència de registres, l'execució de les instrucció següent (seguiu tots els passos del model de màquina de von Neumann).

a. **MOV (4500) , RPG** (RPG és un registre del processador i (4500) vol dir que es fa servir el mode d'adreçament indirecte, MAR i MBR són els registres de transferència amb els busos de memòria)

MAR ← <PC>
Read + wait memory accés time
Load MBR
IR ← <DR> + PC ← <PC> + 1

Cicle de cerca de la instrucció

Decoding

MAR ← <IR>Address (4500)
Read + wait memory accés time
Load MBR
MAR ← <MBR>

Es resol la indirecció

MBR ← <RPG>

Write + wait memory accés time

S'escriu l'operand a memòria

Teoria 4. ***(1 punt)***

En un moment determinat el processador es troba executant la instrucció CMP RPG0, RPG1 que pertany a un cert programa P1. Coneixem les dades següents:

- Aquesta instrucció es troba a la posició 4000 de memòria,
- RPG0 i RPG1 són registres de 8 bits i en el moment d'executar-la RPG0 conté el valor 00110010 i RPG1 el valor 11100111
- En aquest moment el registre SP conté el valor 1000. En aquesta màquina SP apunta a la primera posició lliure de la pila i la pila creix cap a posicions majors de memòria.

- En el moment en que la ALU (Unitat Aritmètica-Lògica) està fent l'operació entre RPG0 i RPG1, s'activa l'únic senyal d'interrupció que té el processador.
- El registre d'estat té 8 bits, però només s'emmagatzemen els bits C, Z, S i O (en aquest ordre i en els bits menys significatius del registre), els altres 4 bits sempre valen 0.
- La rutina de servei que s'executarà en cas d'acceptar la interrupció és a l'adreça 400 de memòria.

Suposant que el processador acceptarà i servirà la interrupció que s'ha produït, responeu les preguntes següents:

i) Indiqueu els valors emmagatzemats en els registres comptador de programa (PC), estat (SR) i apuntador a la pila (SP) just abans de saltar a la rutina de servei d'interrupció.

R0 00110010

R1 11100111

01001011 Resta

C=0, Z=0, S=0 i O(V)=0 **SR = 00000000** **PC = 4001** **SP = 1000**

ii) Indiqueu els valors emmagatzemats en els registres comptador de programa (PC) i apuntador a la pila (SP) just després de saltar a la rutina de servei d'interrupció.

PC = 400 **SP = 1002**

Teoria 5. ***(1 punt)***

Descriviu breument els tres mecanismes d'E/S introduïts en el curs (E/S programada amb llaços d'espera, E/S programada amb interrupcions i Accés Directe a Memòria). Indiqueu les raons que justifiquen l'existència dels dos darrers mecanismes.

L'E/S programada de dades (TPD) amb llaços d'espera es fa mitjançant un programa executat per la CPU on:

1 es selecciona el dispositiu

2 es determina si està disponible el dispositiu

3 s'accedeix a l'i/o port i s'envia (escriu) o es rep (llegeix) la dada a transmetre.

4 si no hem acabat tornar a 2

El pas "2" es fa amb un bucle on es llegeix sistemàticament un i/o port d'estat, esperant que estigui disponible el dispositiu. En aquest bucle podem consumir el 99% dels temps de la TPD.

Si canviem aquest bucle pel fet que el controlador enviï una interrupció a la CPU quan estigui disponible serà una E/S programada amb interrupcions. Això comporta que a partir del punt 3 la CPU pot anar a executar altres programes i anirà fent el pas 3 cada cop que rebi una interrupció. El pas 3 serà la rutina de tractament de la interrupció. Això evita el buble del 99% de temps d'espera.

L'Accés Directe a Memòria consisteix en incloure un controlador de DMA. Aquest dispositiu substitueix completament la CPU en fer la transferència, mentre la CPU fa altres tasques. El controlador de DMA quan acabi la feina que se li ha encomanat d'entrada o sortida, interromprà la CPU per a que li programi una altra tasca d'e/s. Per tant la intervenció de la CPU es limita a programar el que ha de fer el DMA i el DMA fa la transferència.

*Problema 1. (1 **punt**)*

Donat el fragment de codi següent, en llenguatge C, escriviu el fragment de codi equivalent en llenguatge màquina. Cas que sigui necessari podeu assignar a cada variable l'adreça que considereu adient.

```
#define N 1000

int i, max, V[N];
...
max = V[0];
for ( i = 1; i < N; i++ )
        if ( max < V[i] ) max = V[i];
```

```
i <- 999      max  <- 998     v <- 1000

        mov  R0,v
        mov  X, #1
for:    cmp X, #1000
        jge fifor
        cmp R0, v[X]
        jge fiif
        mov R0, v[X]
fiif:   inc X
        jmp for
fifor:  mov 998, R0
        mov 999, X
```

*Problema 2. (1 **punt**) **Mostreu el procediment per calcular-ho***

Quin és el valor en decimal (base 10) de la seqüència de bits següent 11010110 si la interpretem com:

a) Un número natural en binari **(0,25 punts)**

$11010110 = 2^7+2^6+2^4+2^2+2^1 = 128+64+16+4+2 = 214_{(10}$

b) Un número enter representat en signe magnitud i 8 bits (SM) (**0,25 punts**)

$11010110 = -\ (\ 2^6+2^4+2^2+2^1\ ) = -\ (64+16+4+2) = -86_{(10}$

c) Un número enter representat en complement a 2 i 8 bits (Ca2) (**0,5 punts**)

$11010110 = -\ (00101010) = -\ (2^5+2^3+2^1) = -\ (32+8+2) = -42_{(10}$

*Problema 3. (**1 punt**)*

Quin és el valor en binari (base 2) de la seqüència de dígits 321 si la interpretem com:

a) Un número natural en base 16 (és a dir, $321_{(16)}$) **(0,25 punts)**

0011 (3) 0010 (2) 0001 (1) 001100100001

b) Un número natural en base 8 (és a dir, 321(8) (**0,25 punts**)

011 (3) 010 (2) 001 (1) 011010001

c) Un número natural en base 10 (és a dir, 321(10) (**0,5 punts**)

321 div 2 = 160, Residu = 1

160 div 2 = 80, Residu = 0

80 div 2 = 40, Residu = 0

40 div 2 = 20, Residu 0 . . . **101000001**

*Problema 4. (1 **punt**)*

Donats els valors A = 11110101 i B = 11100001 que representen dos números enters binaris en complement a 2 (C2) i 8 bits. (Opereu sempre en C2)

a) Quin és el valor de A + B? Es produeix desbordament (overflow) (expliqueu el perquè)?

1 1 1 1 0 1 0 1

1 1 1 0 0 0 0 1

1 1 0 1 0 1 1 0 C=1 Overflow = 0 perquè el signe dels operands és igual al del resultat

b) Quin és el valor de A – B? Es produeix desbordament (overflow) (expliqueu el perquè)?

-B = 00011111

1 1 1 1 0 1 0 1

0 0 0 1 1 1 1 1

0 0 0 1 0 1 0 0 C=1 Overflow = 0 perquè estem sumant un positiu i un negatiu.

*Problema 5. (2 **punts**)*

Donat el següent fragment de programa

1.-	MOV X, #0	50.-	PUSH R0
2.-	MOV R0, 200[X]	51.-	INC R0
3.-	MOV (301), R0	52.-	INC X
4.-	CALL(JSR) 50	53.-	PULL R0
5.-	CMP X, #2	54.-	RTN
6.-	BNE 2		

Mostreu l'evolució dels registres, la memòria i la pila.

INSTRUCCIO	PC	SP	R0	X	100	101	102	200	201	202	300	301	302	400	401
	1	400			200	201	202	10	11	12	100	101	102		
mov X, #0	2			0											
mov R0, 200[X]	3		10												
mov (301), R0	4					10									
call (jsr) 50	50	401												5	
push R0	51	402													10
inc R0	52		11												
inc X	53			1											
pull R0	54	401	10												
Rtn	5	400													
cmp X, #2	6														
bne 2	2														
mov R0, 200[X]	3		11												
mov (301), R0	4					11									
Call (jsr) 50	50	401												5	
push R0	51	402													11
inc R0	52		12												
inc X	53			2											
pull R0	54	401	11												
rtn	5	400													
cmp X, #2	6														
bne 2	7														
. . .															

Els modes d'adreçament utilitzats són:

n Directe #n immediat n[X] Indexat (n) Indirecte

L'SP apunta a la 1a posició lliura de la pila i aquesta creix cap a posicions majors de memòria.