Cognoms: ..... Nom: ..... Grup: .....

Duració de l'examen: 2 hores

La solució a cada exercici cal escriure-la en l'espai reservat en el mateix enunciat. No podeu fer servir calculadora, mòbil, apunts, … La solució de l'examen es publicará demà a Atenea i les **notes** es publicaran el **Divendres 9 Juny per la tarda**.

Exercici 1 (1,5 punts)

Cada apartat pregunta sobre un cicle concret de l'execució d'una instrucció al SISC Von Neumann. Escriviu el valor dels bits de la paraula de control que genera el bloc SISC CONTROL UNIT durant el cicle a què fa referència cada apartat. **Poseu x sempre que un bit sigui irrellevant en aquest cicle (encara que se'n pogués saber el valor)**. Per a cada apartat/fila, s'indica el node/estat de la UC i la instrucció SISA emmagatzemada a l'IR en aquest cicle. Si us cal, podeu suposar que el contingut del registre Ri és i.

Criterio: 0,5 por la, 0,4 con una señal errónea, 0,25 con dos señales erróneas, 0 si hay 3 o más errores

Apartado	Nodo/Estado (Mnemo Salida)	Instrucción en IR (en ensamblador)	Palabra de Control																		
			@A	@B	Pc/Rx	Ry/N	OP	F	P/I/L/A	@D	WrD	Wr-Out	Rd-In	Wr-Mem	Ldlr	LdPc	Byte	Alu/R@	R@/Pc	N (hexa)	ADDR-IO (hexa)
a	In	IN R7, 65	xxx	xxx	x	x	xx	xxx	10	111	1	0	1	0	x	0	x	x	x	xxxx	41
b	D	BNZ R2, 16	010	xxx	1	0	00	100	xx	xxx	0	0	0	0	0	0	x	x	x	0020	xx
c	Stb	STB -3(R6), R5	xxx	xxx	x	x	xx	xxx	xx	xxx	0	0	0	1	x	0	1	x	1	xxxx	xx

Exercici 2 (1,5 punts) **Criterio: 0,25 puntos por la, correción binaria**

Especifiqueu el camí crític (indicant la suma ordenada dels temps de propagació dels blocs pels quals passa) i calculeu el temps de cicle mínim perquè el computador SISC Von Neumann pugui executar correctament el tipus d'instrucció SISA que s'indica a cada apartat (aquest seria el temps de cicle mínim del computador si només executa instruccions com la indicada o altres que requereixin menor temps). No heu d'afegir cap percentatge de seguretat al càlcul del temps de cicle mínim. Suposeu que els temps de propagació dels blocs que formen el computador són els següents:

Tp(ROM_Q+) = 50 u.t.
Tp(ROM_OUT) = 70 u.t.
Tp(MUX-2-1) = 50 u.t.
Tp(MUX-4-1) = 100 u.t.
Tp(REG) = 100 u.t. // Temps de propagació d'un registre.
Tp(REGFILE) = 250 u.t. // Temps de lectura del banc de registres
Tp(ALU-slow) = 700 u.t. // Tp de ALU per a les operacions/funcions lentes: ADD, SUB, CMP*.
Tp(ALU-quick) = 350 u.t. // Tp de ALU paper a les operacions/funcions ràpides: qualsevol diferent de ADD, SUB, CMP*.
Tacc(64Kb MEMORY) = 900 u.t. // Temps acces (lectura o escriptura) a la memòria
Tp(AND-2) = Tp(OR-2) = 20 u.t. Tp(NOT) = 10 u.t.

El temps de propagació d'un bloc combinacional (Tp) i el temps d'accés a memòria per fer una lectura (Tacc) és el temps des que estan estables totes les entrades necessàries fins que s'estabilitzen les sortides requerides al valor correcte per a les entrades aplicades . Desconeixem com s'han implementat internament els blocs (i podria ser de manera diferent dels vistos a classe). Recordeu que un registre amb senyal de càrrega (Ld), REGwLd, està construït amb un REG i un MUX-2-1 (no us donem l'esquema intern del REGwLd, perquè ho heu de saber).

a) Tc corresponent al node de **F (Fetch)**: ……..

Tc (Fetch)= 100+70+100+50+700+50+50 (REG->ROM_OUT->MUX2-1 ->MEMORY->MUX2-1->IR) = 1170 ut

b) Tc corresponent al node de **Addr**: ……..

Tc (Addr)= 100+70+100+50+700 (REG->ROM_OUT->MUX4-1->MUX2-1->ALU-slow->R@) = 1020 ut

Cognoms: ..... Nom: ..... Grup: .....

Exercici 3 (1,5 punts)

Indiqueu quins canvis es produeixen a l'estat al SISC Von Neumann després d'executar les instruccions de la taula suposant que abans d'executar-se cadascuna PC=0xACDC, Ri=0xABBA i que el contingut del byte de memòria i-ésim és igual a (i+2) mòdul 2^8.

Instrucción a ejecutar	Cambios en el estado del computador
LD R1, 1(R6)	R1 = 0xBDBC, PC = 0xACDE
MOVHI R2, 0x26	R2 = 0x26BA, PC = 0xACDE
BNZ R3, -7	PC = 0xACD0
JALR R4, R5	R4= 0xACDE, PC = 0xABBA

Utilitzeu el mnemotècnic MEMb[...] = ... i/o MEMw[...] = ... per indicar canvis a la memòria.

Criterio: 0,25 puntos por la, correción binaria

Exercici 4 (1 punt) **Criterio: 0,25 cada la, 0,1 si hay 1 error, 0 si hay 2 o más errores**

Completeu les files i les columnes de la taula següent que representa un subconjunt de la ROM_OUT de la unitat de control del SISC Von Neumann. Poseu x sempre que un bit pugui valer tant 0 com 1.

@ROM	Bz	WrOut	R@/PC	Alu/R@	OP1	OP0	MxN1	MxN0	MxF	Estado
0	1	0	0	1	0	0	1	1	1	Fetch
8	0	0	1	x	x	x	x	x	x	Ldb
10	1	0	x	1	1	0	x	x	1	Jalr
14	0	0	x	x	1	0	0	1	1	Movhi

Exercici 5 (2 punts) **Correcció: Tots els apartats, correcció binària**

El programa assemblador de la dreta s'ha traduït a llenguatge màquina per ser executat al SISC Von Neumann, situant la secció **.text** a partir de l'adreça 0xBE00 de memòria i tot seguit la secció **.data**.

```
        N = 8 ; Asumimos N par

.data
V:      .space 2*2*N

.text
        ; primera mitad
        MOVI    R0, lo(V)
        MOVHI   R0, hi(V)
        MOVI    R1, lo(N)
        MOVHI   R1, hi(N)
        MOVI    R2, 0
        MOVI    R3, 0
L1:     ADD     R2, R2, R3
        ST      0(R0), R2
        ADDI    R0, R0, 2
        ADDI    R3, R3, 1
        CMPLTU  R4, R3, R1
        BNZ     R4, L1

        ; segunda mitad
        MOVI    R2, 0
        MOVI    R3, 0
L2:     ADD     R2, R2, R3
        ST      0(R0), R2
        ADDI    R3, R3, 1
        ADD     R2, R2, R3
        ST      2(R0), R2
        ADDI    R0, R0, 4
        ADDI    R3, R3, 1
        CMPLTU  R4, R3, R1
        BNZ     R4, L2
.end
```

a) Un cop carregat el programa en memòria, a quina adreça de memòria corresponen les etiquetes, o direccions simbòliques, L1 i V? (**0,5 punts**)

L1 = 0xBE0C
V = 0xBE2E

b) Quina és la codificació SISA de les instruccions MOVHI R0, hi(V) i BNZ R4, L1? (**0,5 punts**)

MOVI R0, hi(V) = 0x91BE
BNZ R4, L1 = 0x89FA

c) Un cop executat el programa, quin és el contingut del vector V? Indicar la llista de valors com words, separats per comes i en l'ordre d'emmagatzament(**1 punt**)

0,1,3,6,10,15,21,0,1,3,6,10,15,21,28

Exercici 6 (1 punt) **Criterio: 0,25 puntos cada respuesta, criterio binario**

Cognoms: ..... Nom: ..... Grup: .....

a) Indica quin és el contingut de les següents adreces de de la ROM_Q+ del computador Von Neumann:
ROM_Q+[0x010] = 0x01
ROM_Q+[0x14F] = 0x00

b) Indica quina adreça/es de la ROM_Q+ contenen les següents transcions. Indica las adreces en format binari, indicant amb x els bits que no importin.

De Addr a Stb = 00101 0110 x
De Decode a Movi =00001 1001 0

Exercici 7 (3,5 punts)
El dissenyador del llenguatge màquina SISA considera que cal afegir una instrucció nova al repertori d'instruccions. És una instrucció de salt indexat (indexed jump, JI) que permet saltar a l'adreça de codi indicada per la posició enèsima d'un vector emmagatzemat en memòria, on n és un natural menor que 128. Com la instrucció JALR, aquesta nova instrucció també torna l'adreça de codi corresponent al seqüenciament implícit.

Sintaxis: JI Ra, N7
Codificació: 1111 aaa x 0nnn nnnn
Semántica: tmp = PC + 2; PC = Memw[Ra + 2 · N7]; Ra = tmp;

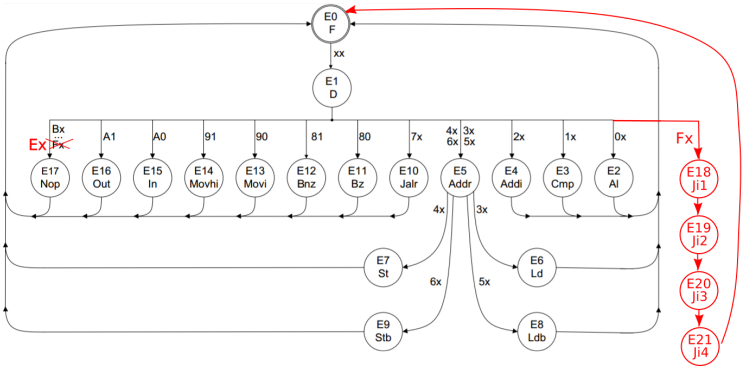
La codificació de la instrucció exigeix que el bit 7 valgui '0'; per contra, el valor del bit 8 és indiferent. Tingueu en compte que Ra és tant registre font com registre destinació.
Per exemple, si R3=0xBECA, PC=0xAD10 i el vector de words emmagatzemat a partir de l'adreça 0xBECA conté els words 0xABBA, 0xACDC, 0xCAFE, 0xBEBE, 0xBABE, 0xFACE, ... en aquest ordre, l'execució de JI R3 , 0 faria que PC=0xABBA i R3=0xAD12; en canvi, l'execució de JI R3, 5 faria que PC=0xFACE i R3=0xAD12.

a) Considerem un programa que executa 2.000 instruccions ràpides i 1.000 d'accés a memòria. Reescrivim el programa fent servir la instrucció JI i obtenim una nova versió del programa que executa 1.600 instruccions ràpides, 900 instruccions d'accés a memòria i 200 instruccions JI. Suposant que la implementació de la nova instrucció trigui 6 cicles (incloent-hi Fetch i Decode) i que no impacti al Tc, indiqueu quants cicles triga l'execució de cada versió del programa i quin percentatge de reducció en el temps d'execució observaríem respecte al temps d'execució del programa original. (Indiqueu el càlcul en funció del nombre i tipus d'instruccions així com el resultat final) **(0,25 punts)** **Criteri: binari**
NumCiclos-original = 3 2.000 + 4 1.000 = 10.000 ciclos
NumCiclos-nuevo = 3 1.600 + 4 900+ 6 200 = 4.800 + 3.600 + 1.200 = 9.600 ciclos
El programa con la nueva instrucción tardará un 4 % menos que el original

b) Sense modificar el maquinari i només modificant el contingut de les ROM's, completeu el disseny del computador perquè executi, a més de les instruccions originals, la instrucció JI en 6 cicles (incloent F i D).

b1) Indiqueu quines modificacions introduiríeu al graf d'estats de la unitat de control **(0,5 punts)** **Criteri: binari**

Cognoms: ..... Nom: ..... Grup: .....



b2) Indiqueu el contingut de les de la ROM_OUT que calgui modi car així com les accions a realitzar (la taula adjunta té el nombre suficient de files, fins i tot és possible que no siguin necessàries totes) **(1 punt)**

Criterio: 0,2 las acciones (0,05 cada una); 0,8 la romout (0,2 por la, 0,15 con un error, 0,1 con dos errores), también sería posible hacer Ra=PC en el estado 20

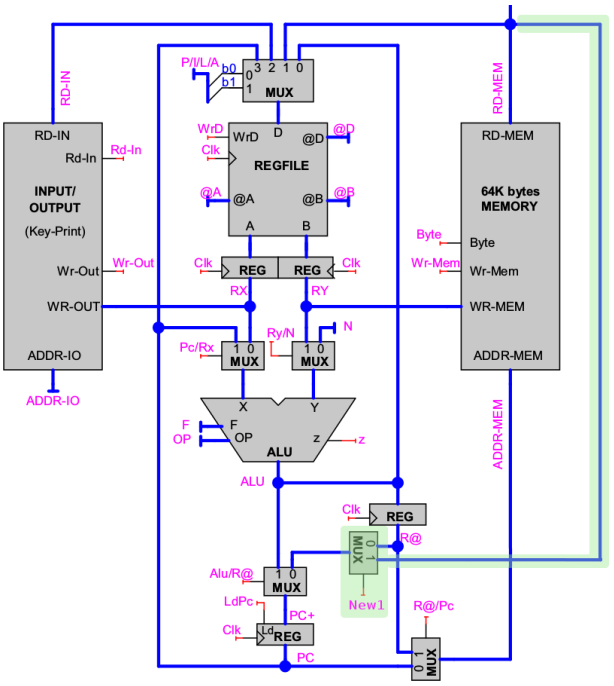
@ROM	Bnz	Bz	WrMem	RdIn	WrOut	WrD	LdIr	Byte	R@/Pc	Alu/R@	Pc/Rx	Ry/N	P/I/L/A1	P/I/L/A0	OP1	OP0	MxN1	MxN0	MxF	F2	F1	F0	Mx@D1	Mx@D0
18	0	0	0	0	0	0	0	x	x	x	0	0	x	x	0	0	1	0	1	1	0	0	x	x
19	0	0	0	0	0	1	0	0	1	x	x	x	0	1	x	x	x	x	x	x	x	x	1	0
20	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
21	1	1	0	0	0	1	x	x	x	1	0	x	1	1	1	0	x	x	1	0	0	0	1	0

Acciones asociadas al estado (en lenguaje de transferencia de registros)

R@ = RX + SE(N8)·2
Ra = Mem$_w$[R@]
RX = Ra
PC = RX || Ra = PC

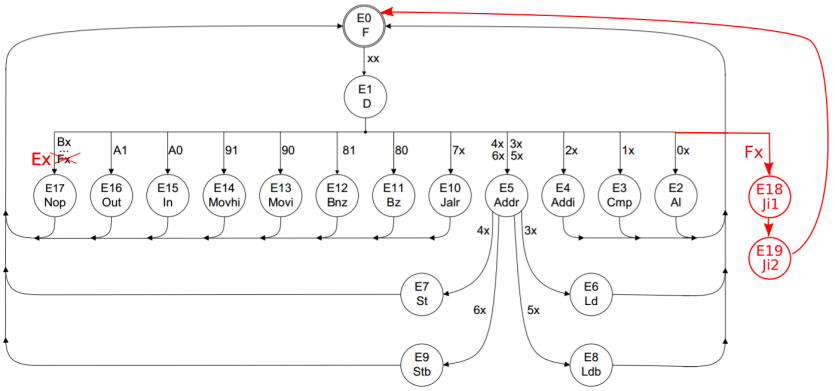
c) Si poguéssim modificar el maquinari de la Unitat de Procés i el de la Unitat de Control),

c1) Com modificaríeu el maquinari de la UPG per reduir el temps d'execució de la instrucció JI a 4 cicles (incloent-hi Fetch i Decode)? Podeu modificar busos i afegir multiplexor(s), busos i senyals de control però no podeu modificar ni els blocs ni l'ús dels senyals de control existents. Per poder mantenir el temps de cicle, no és vàlid calcular la direcció de memòria i accedir a memòria al mateix cicle. **(0,25 punts)** **Criterio: binario, existen otras soluciones correctas que ubican el multiplexor en otro punto**



Cognoms: ..... Nom: ..... Grup: .....

c2) Com modificaries conseqüentment el graf d'estats de la UC? **(0,25 puntos)** **Criterio: binario**



c3) Indiqueu el contingut total de les de la ROM_OUT que calgui afegir així com les accions a realitzar; disposeu d'espai per indicar dos nous senyals de control (és possible que no siguin necessaris tots). Indiqueu també el valor del(s) nou(s) senyal(s) en almenys dues de les de la ROM_OUT ja existents en què el(s) nou(s) senyal(s) de control no tingui(n) el valor x. La taula adjunta té el número suficient de files, fins i tot és possible que no siguin necessàries totes. **(1 punt)** **Criterio: como apartado b2), existen diversas soluciones**

@ROM	New1	New2	Bnz	Bz	WrMem	RdIn	WrOut	WrD	LdIr	Byte	R@/Pc	Alu/R@	Pc/Rx	Ry/N	P/I/L/A1	P/I/L/A0	OP1	OP0	MxN1	MxN0	MxF	F2	F1	F0	Mx@D1	Mx@D0	Acciones asociadas al estado (en lenguaje de transferencia de registros)
18	x		0	0	0	0	0	0	0	x	x	x	0	0	x	x	0	0	1	0	1	1	0	0	x	x	R@ = RX + SE(N8)·2
19	1		1	1	0	0	0	1	x	0	1	0	x	x	1	1	x	x	x	x	x	x	x	x	1	0	PC = Mem$_w$[R@] || Ra = PC
11	0																										(Bz)
12	0																										(Bnz)

c4) Actualitzeu el càlcul de l'apartat a) considerant que la instrucció JI tarda 4 cicles. **(0,25 punts)** **Criterio: como apartado a)**

NumCiclos-original = 3 2.000 + 4 1.000 = 10.000 ciclos

NumCiclos-nuevo = 3 1.600 + 4 900+ 4 200 = 4.800 + 3.600 + 800 = 9.200 ciclos

El programa con la nueva instrucción tardará un 8 % menos que el original