Apellidos y Nombre: .....Grupo:.....DNI: .....

Examen 3 (temas 8, 9, 10 y 11)

Duración del examen: 1h 45min. Escribid la solución de cada ejercicio en el espacio reservado para ello en el enunciado. No podéis utilizar calculadora, móvil, apuntes, etc. La solución se publicará en Atenea mañana por la tarde y las notas antes de una semana.

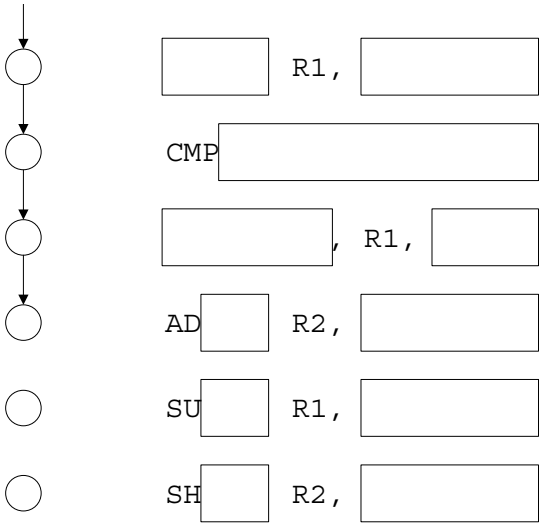
Ejercicio 1 (2 puntos)

Para ejecutar el código `for(R1=133;R1>-133;R1--){if(R1<0>==1)R2=R2+R1}R2=R2/4;` donde los registros contienen números enteros en Ca2, completad: (En ambos apartados se usa `R7` para almacenar valores temporales) (`R1<0>` es el bit de menor peso de `R1`). (Ver chuleta en página 4)

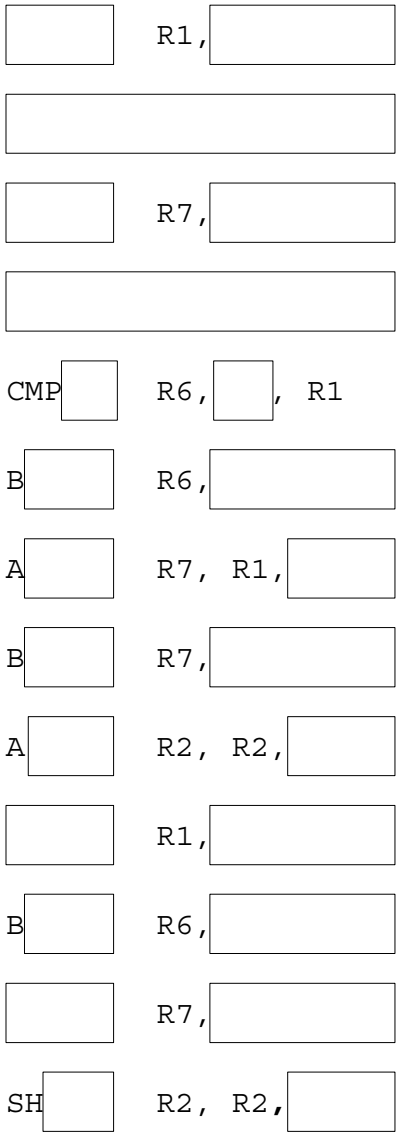
a) El grafo de estados de la UC de propósito específico para que, junto con la UPG, formen un PPE que lo ejecute. No faltan nodos, faltan arcos y etiquetas (z, !z, o nada) y la palabra de control en mnemotécnicos de cada nodo.

b) El código en ensamblador SISA para ejecutarlo en el computador SISC Harvard uniciclo (UCG+UPG+IO$_{key\text{-}print}$+MEM).

a) (1 punto)

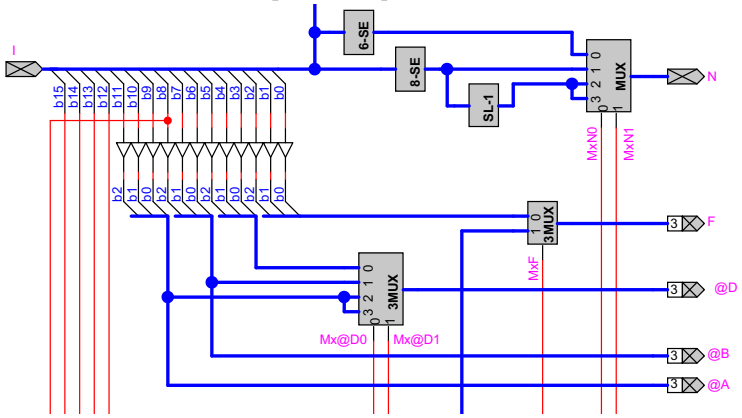


b) (1 punto)



Ejercicio 2 (2 puntos)

Escribid el contenido de las filas que se muestran de la tabla de la ROM del bloque ROM-CTRL-LOGIC del computador SISC Harvard uniciclo, pag.4 (UCG+UPG+IO$_{key\text{-}print}$+MEM). La figura muestra parte de la lógica de control. Las señales de la palabra de control que no se muestran salen directamente de la ROM (excepto TknBr que se forma así: TknBr=Bz·z+Bnz·!z).



Dirección					Contenido																	
I<15>	I<14>	I<13>	I<12>	I<8>	Bnz	Bz	Wr-Mem	Rdln	WrOut	WrD	Byte	Rb/N	-/i/l/a1	-/i/l/a0	OP1	OP0	MxN1	MxN0	MxF	F2	F1	F0	Mx@D1	Mx@D0
0	0	1	1	x																				
0	1	1	0	x																				
1	0	0	0	0																				
1	0	0	1	1																				

LD
STB
BZ
MOVHI

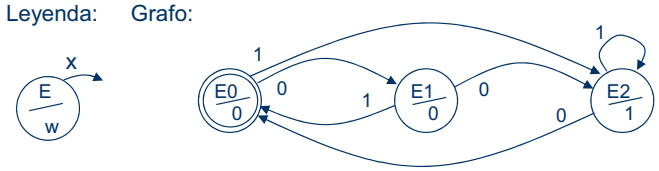
Ejercicio 3 (**2,5 puntos**)

Vamos a simular el comportamiento de un circuito secuencial sencillo que tiene, ademas de la señal de reloj, una señal de entrada de un bit, x, y otra de salida, w. La figura muestra el grafo de estados que especifica el funcionamiento del circuito secuencial. Realizamos dos simulaciones, apartados a y b, y en ambas la secuencia de valores de entrada, x, se realiza por el teclado y la secuencia de valores de salida, w, se realiza por la impresora, siguiendo el protocolo asíncrono usual en el que la lectura/escritura del puerto de datos del teclado/impresora (KEY-DATA/PRINT-DATA) produce un efecto lateral en el puerto de estado del teclado/impresora (KEY-STATUS/PRINT-STATUS). Cada señal binaria x/w se implementará leyendo/escribiendo por el teclado/impresora una palabra de 16 bits con valor 0 o 1 según corresponda. (Leer la nota explicativa antes de realizar ningún apartado)

a) En este apartado la simulación se realiza en un PPE con una UCE+UPG+$IO_{Key\text{-}Print}$ (ver esquema de la UPG en pag.3). Completad, en la página 3, el grafo de estados de la Unidad de Control Específica (UCE) del PPE que simula **indefinidamente** el comportamiento del circuito. **(1,5 puntos)**

b) En este apartado la simulación se realiza en el computador SISC Harvard uniciclo (UCG+UPG+$IO_{Key\text{-}Print}$) Completad, en la página 3, el código ensamblador SISA para que el SISC Harvard uniciclo (ver esquema en pag.4) simule indefinidamente el comportamiento del circuito. **(1 punto)**

Nota explicativa. Para ambos apartados la secuencia de acciones/instrucciones sigue el mismo patrón; un primer bloque de inicialización de constantes (`R0` y `R1` mantienen las constantes 0 y 1 de 16 bits para simular los valores binarios 0 y 1 que pueden tomar cada una de las señales x y w) seguido de tres bloques, cada uno de ellos simula un nodo del grafo del circuito (E0, E1 y E2). La estructura interna de cada bloque que simula un nodo es la misma: Las primeras 3 acciones/instrucciones realizan la impresión de la salida del circuito, w. Las siguientes 4 acciones/instrucciones realizan la entrada por el teclado de la entrada del circuito, x, y pasan a simular el siguiente nodo del grafo según el valor entrado (y el estado actual que se está simulando). De los tres nodos del grafo solo se pide que escribáis los bloques que simulan el E0 y el E2, para simplificar el ejercicio. Los cuatro bloques se han separado por líneas discontinuas. En el bloque E2 hay que considerar que están las 7 acciones (nodos)/instrucciones, aunque no se muestran para ahorrar espacio. Para ambos apartados solo se usa el registro `R7`, ademas del R0 y `R1` ya mencionados.



Ejercicio 4 (3,5 puntos)

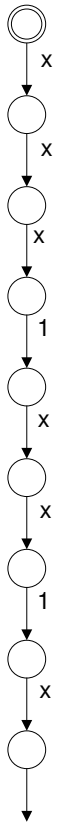
Completad la tabla en la que cada fila es un apartado diferente que contiene 4 columnas para una misma instrucción: 1) Instrucción en ensamblador SISA, 2) Instrucción en lenguaje máquina (LM) en hexadecimal, 3) algunos bits de la palabra de control del SISC Harvard uniciclo (UCG+UPG+$IO_{key\text{-}print}$+MEM), ver pag. 4, (poned x siempre que no se sepa el valor del bit al no saber cómo se han implementado las x en la ROM de la Lógica de Control) y 4) estado del computador después de ejecutar la instrucción suponiendo que el estado antes de su ejecución es (El símbolo % denota la operación módulo): `PC=0x03DE`; `Ri=2*i` para `i=0`,... `7`; MEM_w`[@]=(@+2)%(2^16)` para `@=0`, 2, 4, 6... `(2^16)-2`
Escribid solo el contenido, en hexadecimal, de los registros (incluido el `PC`) y la palabra de la memoria de datos, MEM_w (si se modifica un byte debe indicarse el valor de la palabra a la que pertenece) que se modifican al ejecutar cada instrucción.

0 0 0 0	a a a	b b b	d d d	f f f	Logic and Aritmetic Operations	AND, OR, XOR, NOT, ADD, SUB, SHA, SHL
0 0 0 1	a a a	b b b	d d d	f f f	Compare Signed and Unsigned	CMPLT, CMPLE, -, CMPEQ, CMPLTU, CMPLEU, -, -
0 0 1 0	a a a	d d d	n n n n n n n		Add Immediate	ADDI
0 0 1 1	a a a	d d d	n n n n n n n		Load	LD
0 1 0 0	a a a	b b b	n n n n n n n		Store	ST
0 1 0 1	a a a	d d d	n n n n n n n		Load Byte	LDB
0 1 1 0	a a a	b b b	n n n n n n n		Store Byte	STB
0 1 1 1						Branch future extension
1 0 0 0	a a a	0	n n n n n n n n		Branch on Zero	BZ
		1			Branch on Not Zero	BNZ
1 0 0 1	d d d	0	n n n n n n n n		Move Immediate	MOVI
	a a a d d d	1			Move Immediate High	MOVHI
1 0 1 0	d d d	0	n n n n n n n n		Input	IN
	a a a	1			Output	OUT
1 0 1 1	x x x x x x x x x x x x					Future extensions
1 1 x x						

	1) Ensamblador	2) LM (Hexa)	3) Bits Pal. Control -/i/l/a		WrD	Byte	TknBr	N (hexa)	4) Estado después de su ejecución
a)	`STB -7(R4), R0`								
b)	`BZ R0, -7`								
c)		`596D`							
d)		`9D86`							

Respuesta apartado a)

o:



R0,

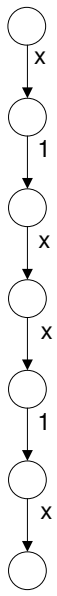
R7,

–,

R7,

–,

Aquí 7 nodos para E1



R7,

–,

Respuesta apartado b)

R0,

R7,

B R7,

R7,

B R7,

B R7,

Aquí 7 instrucciones para E1

R7,

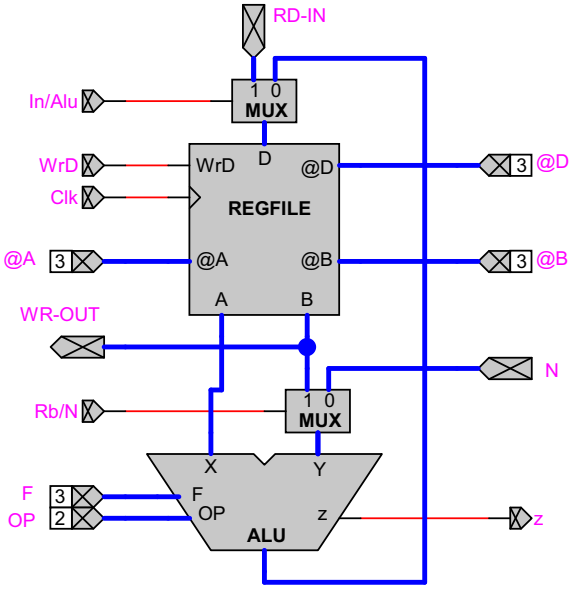
B R7,

R7,

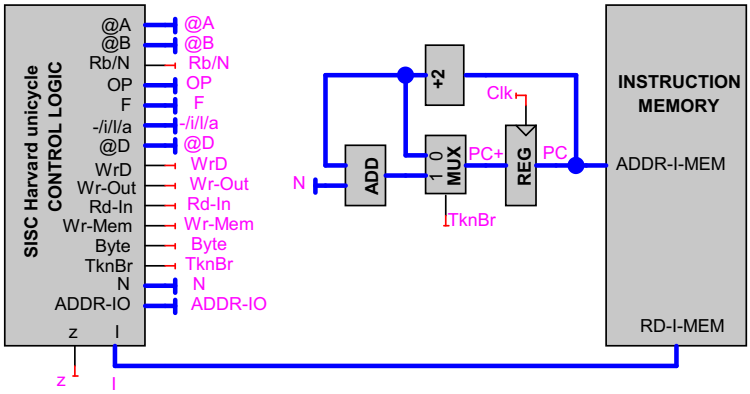
B R7,

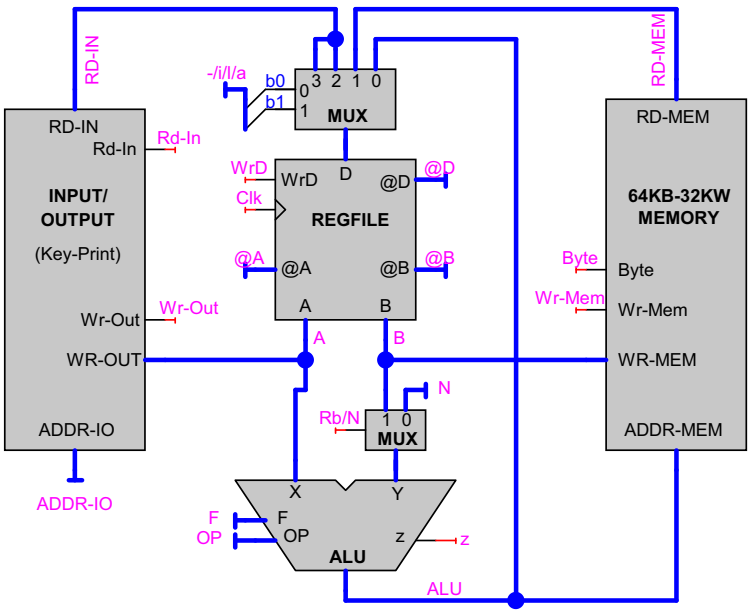
BZ R7,

Esquema a bloques de la UPG sin espacio de Entrada/salida



Estructura a bloques del SISC Harvard uniciclo (UCG+UPG+IO$_{key-print}$+MEM)





Funcionalidad de la ALU

F			OP			
b_2	b_1	b_0	11	10	01	00
0	0	0	---	X	CMPLT (X, Y)	AND (X, Y)
0	0	1	---	Y	CMPLE (X, Y)	OR (X, Y)
0	1	0	---	MOVHI(X, Y)	---	XOR(X, Y)
0	1	1	---	---	CMPEQ (X, Y)	NOT (X)
1	0	0	---	---	CMPLTU (X, Y)	ADD (X, Y)
1	0	1	---	---	CMPLEU (X, Y)	SUB (X, Y)
1	1	0	---	---	---	SHA(X, Y)
1	1	1	---	---	---	SHL(X, Y)