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

Duración del examen: 1 hora 30 minutos. La solución se tiene que escribir en el espacio reservado para ello en el enunciado. No podéis utilizar calculadora, móvil, apuntes, etc. La solución se publicará mañana por la tarde y las notas el 10 de mayo.

Ejercicio 1 (3 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: `PC=0x03DE`; `Ri=2*i` para `i=0`,... `7`; `MEM`$_w$`[@]=@+2` para `@=0`, 2, 4, 6... `(2^16)-2`. Escribid solo el contenido, en hexadecimal, de los registros (incluido el `PC`), palabras de la memoria de datos, `MEM`$_w$ (si se modifica un byte debe indicarse el valor de la palabra a la que pertenece) y puertos de entrada salida, `IN[p]` y `OUT[p]`, que se modifican al ejecutar cada instrucción).

15 14 13 12	11 10 9	8 7 6	5 4 3	2 1 0	Name	Mnemonic
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		Add Immediate	ADDI
0 0 1 1	a a a	d d d	n n n n n n		Load	LD
0 1 0 0	a a a	b b b	n n n n n n		Store	ST
0 1 0 1	a a a	d d d	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		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 OP	WrD	Byte	TknBr	N (hexa)	4) Estado después de su ejecución
a)	`MOVHI R6, 0xB6`							
b)	`STB 0x34(R4),R0`							
c)		`89F8`						
d)		`392E`						

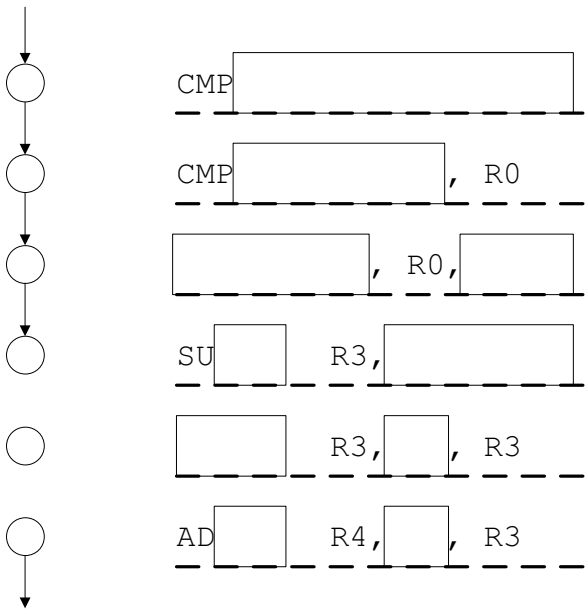
Ejercicio 2 (2 puntos)

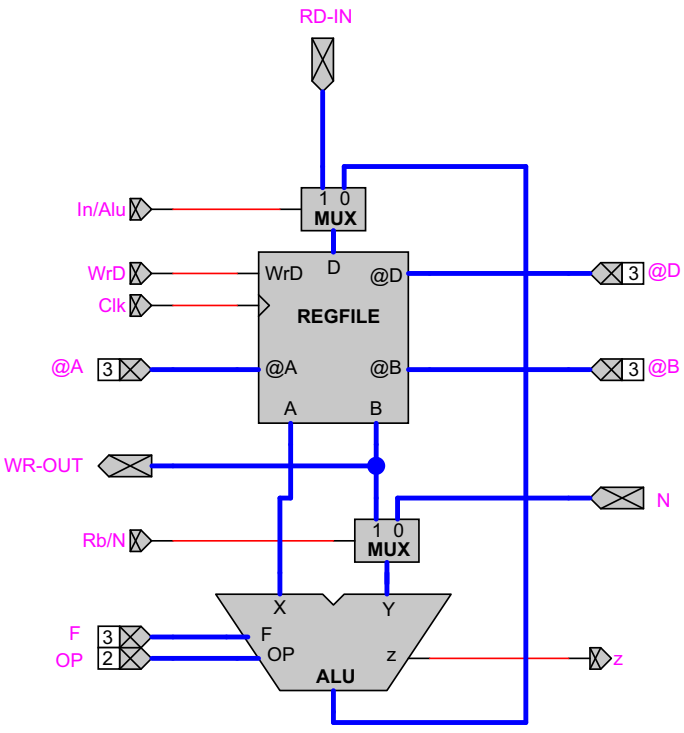
Dado el computador SISC Harvard uniciclo (formado por UCG+UPG+IO$_{key\text{-}print}$+MEM), ver pag. 4, escribid un fragmento de código ensamblador SISA que lea un dato del teclado (que se interpreta como un número natural), lo multiplique por 2 elevado a R3<3..0>u (siendo R3<3..0>u el valor de los 4 bits de menor peso del registro R3 interpretados como un número natural) y lo escriba en la posición de memoria cuya dirección se obtiene sumándole 12 al contenido de R5. Usad las direcciones de simbólicas de los puertos de entrad/salida. El acceso al puerto de datos del teclado tiene un efecto lateral sobre su registro de estado. El código solo debe modificar la memoria y los registros R0 y R1 que se usarán como registros temporales. **El código no puede tener más de 7 instrucciones. Si tiene más se obtendrá un 0.**

Ejercicio 3 (3.5 puntos)

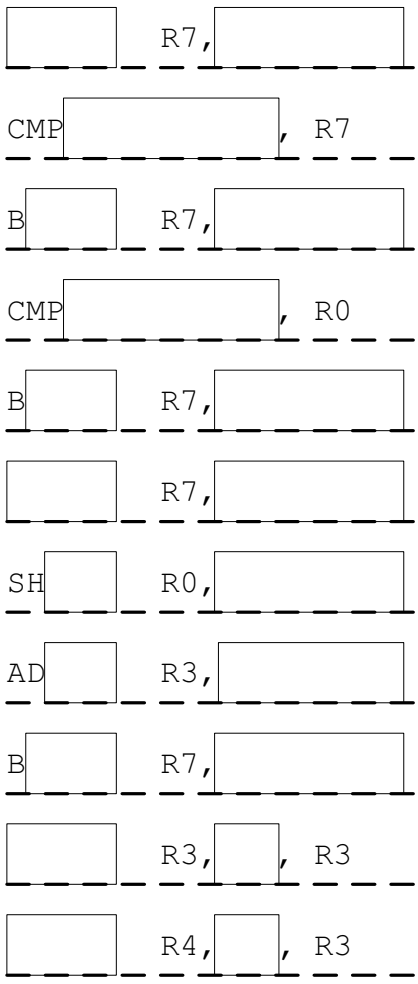
3.a) (2 puntos) Completad el fragmento de grafo de estados de la UC de **propósito específico** para que junto con la UPG formen un procesador que realice lo que indica el código en C. Completad los arcos que faltan, sus etiquetas (z, !z, o nada) y las casillas de cada palabra de control especificada con mnemotécnicos a la derecha de cada nodo del grafo. El contenido de los registros se interpreta como números naturales.

```
if (R3 > 50)
   while (R0 >= R2) {
      R0 = R0 / 2;
      R3 = R3 - 1;
   }
else R3 = R2 | R3;    /* OR bit a bit */
R4 = R0 + R3;
```



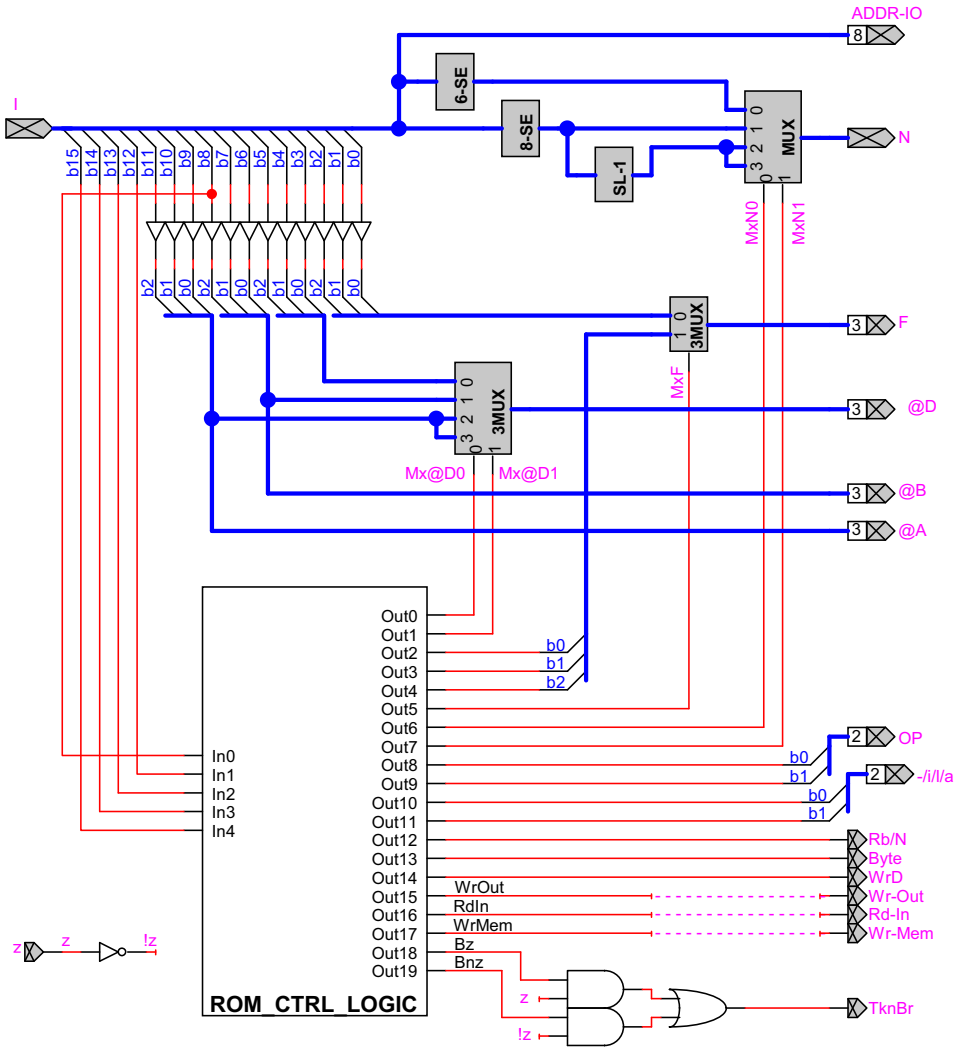


3.b) (1.5 puntos) Completad el programa en lenguaje ensamblador SISA para que el procesador formado por la unidad de control de propósito general (PC + secuenciador + memoria de instrucciones + lógica de control) junto con la UPG realicen la misma funcionalidad que el procesador del apartado 3a. El código solamente escribe en los registros `R0`, `R3`, `R4` (para implementar la funcionalidad descrita) y `R7` (para valores temporales).



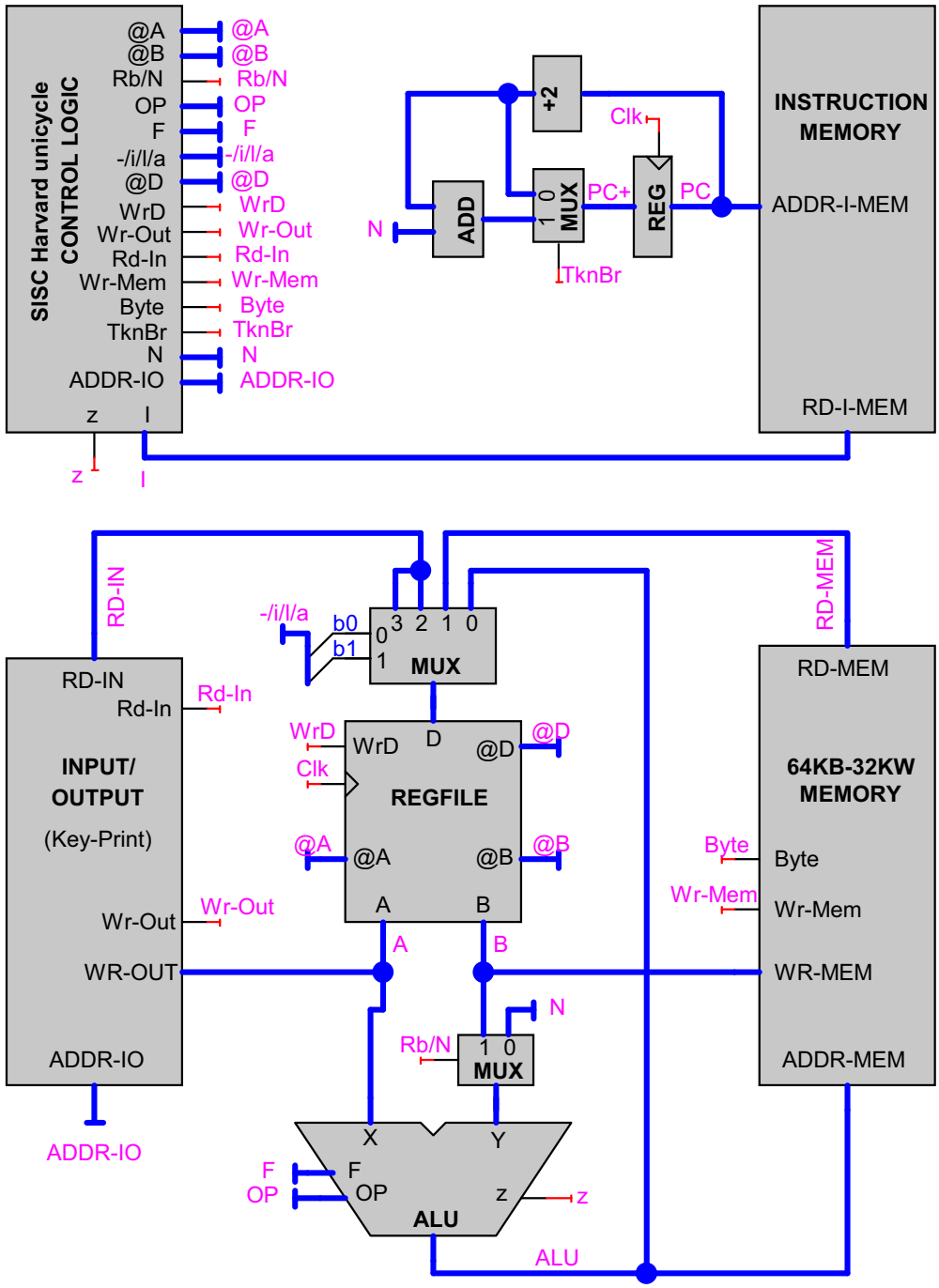
Ejercicio 4 (1,5 puntos)

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 (UCG+UPG+IO$_{key\text{-}print}$+MEM). La figura muestra parte de la lógica de control. Indicad con x los valores de los bits que pueden ser indistintamente 0 o 1.



Dirección					Contenido																					
I<15>	I<14>	I<13>	I<12>	I<8>	Bnz	Bz	Wr-Mem	RdIn	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	0	x																					ADDI
0	1	0	1	x																					LDB
1	0	0	1	1																					MOVHI

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)