Examen E3 *(21 de noviembre del 2019)* IC – 19-20-Q1
Apellidos y nombre: …….. Grupo:……

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

- Duración del examen: 1 hora y 45 minutos.
- La solución se tiene que escribir en el espacio reservado para ello en el propio enunciado.
- No podéis utilizar calculadora, móvil, apuntes, etc.
- La solución se publicará en Atenea mañana y las notas antes del 28 de noviembre del 2019

Ejercicio 1 *(0,5 puntos)*
Completa el siguiente fragmento de código ensamblador SISA para el procesador SISC Harvard uniciclo (UPG+I/O+MEM) para que guarde un 0 en la posición 0x0007 de memoria.

XOR R0, R0, __

ADDI R1, R0, 0x1

SHA R2, _____

ADD R2, _____

SHA R2, _____

ADD R2, _____

ST 0(___), _____

Ejercicio 2 *(1 punto)*

a) Indica el valor que debe tener cada uno de los bits de la palabra de control de la UPG básica (sin subsistema de I/O ni memoria) para que realice, durante un ciclo, la acción concreta especificada mediante el mnemotécnico. Indicad con ***x*** las casillas cuyo valor no importe para la ejecución de la instrucción. En caso de que no se pueda realizar la acción tachar **toda la línea** de señales. *(0.5 puntos)*

Mnemotécnico	@A	@B	Rb/N	OP	F	In/Alu	@D	WrD	N (hexa)
XORI R4, R5, 4									
IN R3 // AND -,R3,R3									

b) Indica el mnemotécnico que corresponde a cada una de las siguientes palabras de control de la UPG básica (sin subsistema de I/O ni memoria). *(0.5 puntos)*

Mnemotécnico	@A	@B	Rb/N	OP	F	In/Alu	@D	WrD	N (hexa)
	001	010	1	00	100	0	001	1	XXXX
	xxx	xxx	0	10	001	0	001	1	FF43

Ejercicio 3 *(1 punto)*
Completa la tabla ensamblando las instrucciones en ensamblador SISA o desensamblando las instrucciones en lenguaje máquina según sea necesario. Indica poniendo NA en la casilla aquellos casos en los que la instrucción no corresponda al lenguaje SISA.

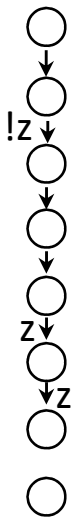
Lenguaje máquina SISA	Lenguaje ensamblador SISA
0x96AB	
0x0772	
	SHL R3, R1, R1
	OUT 0x0F, R5

Ejercicio 4 *(3 puntos)*

Dado el siguiente fragmento de código en C (el código no tiene que hacer algo útil) donde todos los datos son **naturales**, Completad el fragmento de…

```
…
while (R0<R5) {
   R5 = R5/4
   R0 = R0 XOR R4
   if ((R0 == R3) OR (R0 > 0x1F4)){
            R0 = 0xFF
   }
}
```

a) … grafo de estados de la UC de **propósito especifico** para que junto con la UPG formen un procesador que realice la funcionalidad descrita en los fragmentos de código anteriores. Indicad los arcos que faltan, las etiquetas de los arcos (z, !z, o nada) y completad las casillas de cada palabra de control que se especifica con mnemotécnicos a la derecha de cada nodo del grafo. *(1.5 punto)*



…
CMP
SH
CMPEQ
CMP
…

b) … programa en lenguaje ensamblador SISA para que el procesador formado por la unidad de control de propósito general (UCG) junto con la UPG realicen las funcionalidades descritas en los fragmentos de código en C (el código no tiene que hacer algo útil). En las comparaciones, hay que interpretar los datos como valores **naturales**. Rellenad la parte subrayada que falta. *(1.5 punto)*

@I-MEM	
	…
0x1000	CMP_____ R7,_____
0x1002	BNZ R7,_____
0x1004	MOV____ R7,_____
0x1006	SH _____
0x1008	XOR _____
0x100A	CMP_____ R7,_____
0x100C	B____ R7,_____
0x100E	MOV____ R6,_____
0x1010	MOV____ R6,_____
0x1012	CMP_____ R7, _____
0x1014	BZ R7,_____
0x1016	_____
0x1018	_____
0x101A	BNZ R7, _____
0x101C	…

Ejercicio 5 *(1 punto)*

Escribid sobre la siguiente tabla el valor de los bits que tiene la palabra de control del SISC-Harvard uniciclo (incluyendo la señal *TknBr*) durante el ciclo en que se ejecuta cada una de las instrucciones SISA. Indicad únicamente el valor (0 o 1) de los bits que son estrictamente necesarios para ejecutar correctamente cada instrucción. Para el resto de bits de la palabra de control, que pueden valer 0 o 1 indistintamente para la ejecución correcta de la instrucción, poned **x** (aunque se pueda saber el valor codificando la instrucción). Suponed que antes de ejecutar cada instrucción el contenido de los registros es cero.

Instrucción SISA	@A	@B	Rb/N	OP	F	-/i/I/a	@D	WrD	Wr-Out	Rd-In	Wr-Mem	Byte	TknBr	N (hexa)	ADDR-IO (hexa)
MOVHI R0, 0xFF															
XOR R1, R1, R1															
BZ R7, 0xA															
OUT 0x11, R2															

Ejercicio 6 *(1 puntos)*

Indica el contenido de la tabla de la ROM (solo filas indicadas) correspondiente al bloque ROM_CRTL_ LOGIC. Indica los valores que tomarían las señales para ejecutar correctamente las instrucciones. Indica con x los valores de los bits del contenido de la ROM que puedan valer 0 o 1.

Dirección ROM		Contenido ROM													
I_{15} I_{14} I_{13} I_{12}	I_8	Bnz	Bz	Wr-Mem	Rd-In	Wr-Out	Byte	Rb/n	$-i/I/a_1$ $-i/I/a_0$	OP_1 OP_0	MxN_1 MxN_0	MxF	F_2 F_1 F_0	MxD_1 MxD_0	
1 0 1 0	0														IN
1 0 0 1	0														MOVI
0 1 0 1	x														LDB
1 0 0 0	1														BNZ

Ejercicio 7 *(1 punto)*

Indicad qué cambios hay en el estado del computador después de ejecutar cada una de las instrucciones de la tabla suponiendo que **antes de ejecutarse cada una** de ellas el PC vale 0x1000, el contenido de todos los registros necesarios es R0=0x00FF, R3=0x000A, R5=0x0000, y que el contenido de todas las posiciones pares de la memoria de datos es 0x2 y el de todas las posiciones impares de la memoria de datos es 0x1. Asumid todos los registros de E/S contienen el valor 0x0001 y que el teclado está formado por un port de datos (Keydata) y uno de status (KeyStat) y que el sistema de ES tiene el efecto lateral del protocolo de cuatro fases. Utiliza el mnemotécnico MEMb[...], MEMw[...] y Port[...] para indicar los cambios en la memoria y los puertos de E/S respectivamente.

Instrucción	Cambios en el estado del computador
MOVHI R0, 0x1	
BZ R5, 5	
IN R3, KeyData	
ST 0xA(R5), R0	

Ejercicio 8 *(1,5 puntos)*

Dado el computador SISC Harvard uniciclo (formado por UCG+UPG+IOkey-print+MEM), escribid un fragmento de código ensamblador SISA que lea un dato de teclado (ports KeyData y KeyStatus). **Guarde el valor leído en el registro R0. En el Registro R1 guarde los 4 bits de menos peso extendiendo el signo a los 16 bits del contenido del R0.** Después guardaremos los 8 bits de menos peso de R1 en la posición de memoria almacenada en R5. Para cualquier registro temporal emplearemos el R7. No podéis emplear ninguna operación de comparación (CMP*). El Código no debe tener más de 11 líneas.