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

Examen 4 (temas 12, 13 y 14)

- Duración del examen: 2 horas. La solución de cada ejercicio se tiene que escribir en el espacio reservado para ello en el propio enunciado.
- No podéis utilizar calculadora, móvil, apuntes, etc. En hoja aparte se os da una "chuleta" con información útil para realizar los ejercicios.
- La solución del examen se publicará en Atenea mañana por la tarde y las notas antes del próximo lunes noche. En general, para revisar la nota de este examen debéis enviar un email a vuestro profesor de teoría el próximo miércoles, excepto que el profesor indique otra cosa.

Ejercicio 1 **(3 puntos)**

El programa en ensamblador, una vez completado, se traducirá a lenguaje máquina para ser ejecutado en el SISC Von Neumann situando la sección `.text` a partir de la dirección 256 de memoria y a continuación la sección `.data`. Puede ser que en el programa haya más etiquetas de las necesarias.

a) Completad la escritura del programa para que indice en el vector `D` en qué elementos del vector `F` se produciría overflow si se multiplicara cada elemento por 512, considerando los elementos de `F` y `D` como números enteros codificados en complemento a dos con 16 bits. Los elementos `D[0]` y `F[0]` se encuentran almacenados en memoria en las direcciones simbólicas `D` y `F` respectivamente. Solo se procesan los `N` primeros elementos de `F`, a razón de uno por iteración. Se puede responder al apartado c sin haber hecho los anteriores. El programa sigue el siguiente algoritmo: (**1 punto**)

```
for(i=0;i<N;i++)if(((F[i]*512)/512)!=F[i])D[i]=1;
```

b) Una vez cargado el programa en memoria:

- ¿A qué dirección de memoria corresponde cada una de las siguientes etiquetas, o direcciones simbólicas, y qué palabra (word) contiene? (**0.75 puntos**)

`F:`	=>	Mem_w[0x	] = 0x
`L0:`	=>	Mem_w[0x	] = 0x
`L1:`	=>	Mem_w[0x	] = 0x

- Una vez ejecutado el programa en el computador SISC Von Neumann ¿Cuál es la dirección de memoria donde se ha escrito por última vez la instrucción con etiqueta `L3` y cuál es su contenido? (**0,5 puntos**)

Mem_w[0x	] =	0x

c) ¿Cuántas instrucciones se ejecutan? ¿Cuánto tarda en ejecutarse el código en el Harvard uniciclo y en el Von Neumann suponiendo que los tiempos de ciclo son 3.000 y 1.000 u.t. respectivamente? ¿Cuánto vale x para que sea cierta la siguiente afirmación? "El computador Harvard uniciclo es un x% **más rápido** que el Von Neumann, ejecutando este código.(**0,75 puntos**)

```
.data
        N=0x0008     ; 0<N<=16
D:      .space 32
        .even
F:      .word 127,-128,512,63,64
        .word -64,250,29,-1024,56
        .space 12
.text
L0:     MOVI    R0,  N
        MOVI    R1,
        MOVHI   R1,
        MOVI    R7,
        M       R6,
        M       R5,
L1:     LD      R3,  0(R1)
        SH      R4,     ,  R7
        SH      R4,     ,  R6
L2:     CMP     R4,     ,  R4
        B       R4,
L3:     ST          (R1),  R5
L4:     A       R1,  R1,
        A       R0,  R0,
L5:     B       R0,
.end
```

InstrucEjec =	; Tejec(Harvard uniciclo) =	; Tejec(V.Neumann) =	; x =

Ejercicio 2 **(0,75 punto)**

Cada uno de los apartados pregunta sobre un ciclo concreto de la ejecución de una instrucción en el SISC Von Neumann. Escribid el valor de los bits de la **palabra de control** que genera el bloque **SISC CONTROL UNIT** durante el ciclo a que hace referencia cada apartado. **Poned x siempre que no se pueda saber el valor de un bit** (ya que no sabemos cómo se han implementado las x en la ROM_OUT). Para cada apartado/fila se indica el nodo/estado de la UC en ese ciclo y la instrucción (en ensamblador) que está almacenada en el IR en ese ciclo. Podéis ver el grafo de estados de Moore de la UC en el anexo. Suponed que el contenido de todos los registros, Rk para k=0,...7, antes de ejecutarse cada instrucción es 0.

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	D	`ST 18(R3), R2`																			
b	Bnz	`BNZ R5, -61`																			
c	Movhi	`MOVHI R6, 0x80`																			

Ejercicio 3 **(1,25 puntos)**

Completad la tabla que representa en forma compacta parte del contenido de la ROM_OUT de la unidad de control del SISC Von Neumann. Poned x siempre que un bit pueda valer tanto 0 como 1.

@ROM	Bnz	Ldlr	R@/Pc	Alu/R@	Pc/Rx	Ry/N	MxN1	MxN0	MxF	Mx@D1	Mx@D0	
4												Addi
8												Ldb
11												Bz
14												Movhi

Ejercicio 4 **(1 punto)**

Indicad qué cambios se producen en el estado del computador SISC Von Neumann después de ejecutar cada una de las instrucciones de la tabla suponiendo que antes de ejecutarse cada una de ellas el PC vale 0xF0F8, el contenido de todos los registros es 0x6789 y que el byte contenido en todas las direcciones pares de la memoria es 0x83 y el de todas las impares es 0x01. Utilizad la notación $MEM_b[\ldots]$=... y/o $MEM_w[\ldots]$=... para indicar los cambios en la memoria.

Instrucción a ejecutar	Cambios en el estado del computador
`LDB R0, -3(R5)`	
`MOVHI R2, 0x93`	
`BNZ R1, 0xF0`	

Ejercicio 5 **(3 puntos)**

Completad el diseño del SISC Von Neumann para que pueda ejecutar, además de las 25 instrucciones originales SISA, la nueva instrucción STRR+, que tiene el formato y codificación, la sintaxis ensamblador y la semántica siguientes:

```
Codificación:  1011 aaa bbb ddd xxx
Sintaxis:      STRR+ (Ra+Rd), Rb
Semántica:     Mem_w[(Ra+Rd)&(~1)]=Rb; Ra=Ra+2
```

La instrucción almacena una palabra en memoria en la dirección que se obtiene sumando los registros Ra y Rd y después incrementa en 2 el registro Ra.

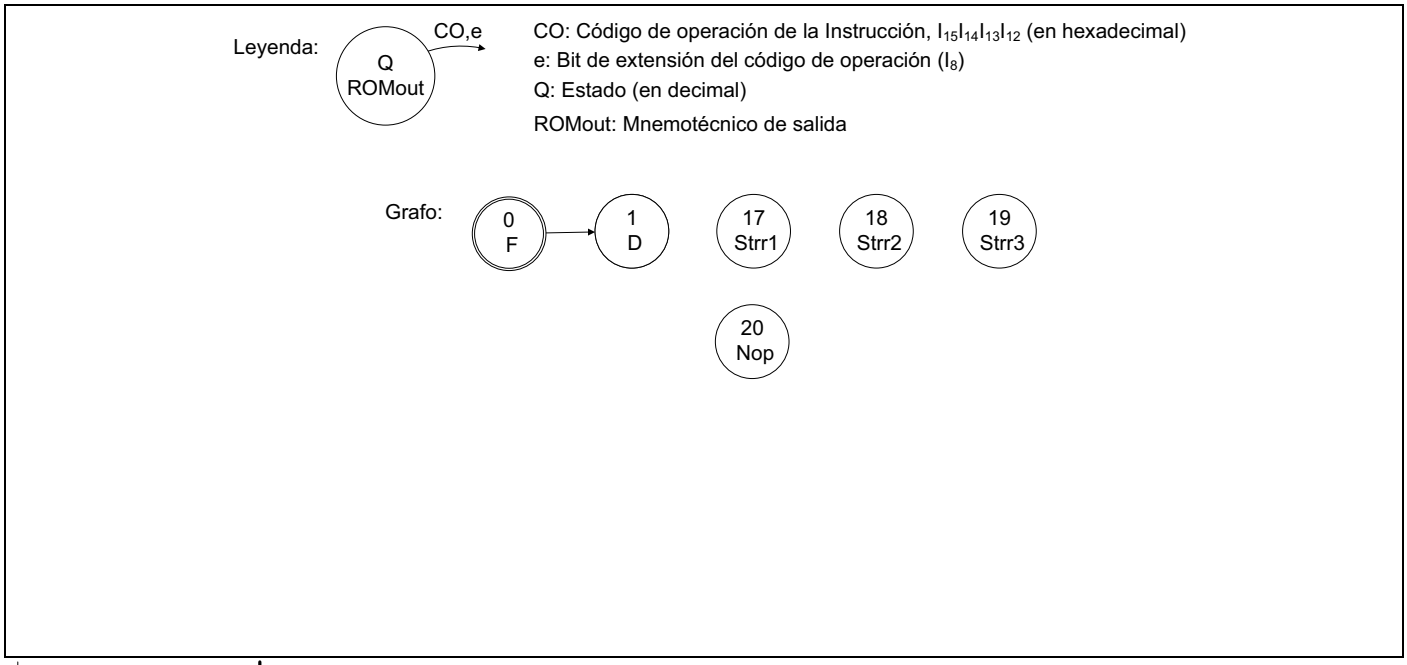
Para poder ejecutar la nueva instrucción solo hay que añadir un nuevo multiplexor de buses, MUX-2-1, en la unidad de control (con señal de selección Mx, que valdrá 0 en el nodo/ciclo Decode, D) y modificar parte del contenido de la ROM_Q+ y de la ROM_OUT.

Se pide:

a) Completad el fragmento del grafo de estados del circuito secuencial de la unidad de control necesario para ejecutar completamente la nueva instrucción STRR+ y todas las que tienen un código ilegal. Se da la leyenda del grafo y todos los nodos necesarios, pero solo un arco al que le falta la etiqueta. Dibujad todos los arcos que faltan y todas las etiquetas.

   Completad, también, el contenido de la tabla que indica, mediante una fila para cada nodo, la acción o acciones en paralelo que se realiza en el computador en cada uno de los 5 ciclos/nodos que requiere la ejecución de la nueva instrucción (Fetch, Decode, y los 3 ciclos/nodos de la ejecución propiamente dicha): F, D, Strr1, Strr2 y Strr3. Para especificar las acciones usad el mismo lenguaje de transferencia de registros que en la documentación. El nuevo multiplexor de la UC se usa para generar un campo/bit de la palabra de control ¿Qué campo/bit es este? Responded en la siguiente página. (**1,5 = 0,25 + 1 + 0,25 puntos**)

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



Nodo/Estado		Acciones
Número	Mnem.	
0	F	
1	D	
17	Strr1	
18	Strr2	
19	Strr3	

El nuevo MUX-2-1 con señal de selección Mx se usa para generar el campo/bit ..... de la palabra de control

b) Completad (poniendo 0, 1 o x en cada bit) las filas de la tabla que especifica parte del contenido de la ROM_OUT para que se ejecuten correctamente todas las instrucciones, poniendo el máximo número de x posibles. La dirección 0 de la ROM corresponde al estado 0 (F), la 1 al 1 (D)... la dirección 17 al estado 17 (Strr1)... la dirección 19 al estado 19 (Strr3) y de la 20 a la 31 el estado Nop (No operación, que se usa para que las instrucciones con un código de operación ilegal no modifiquen el estado del computador y pasen a ejecutar la siguiente instrucción en secuencia). (**1,25 puntos**)

@ROM	Mx	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	
0																										F
1	0																									D
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
17																										Strr1
18																										Strr2
19																										Strr3
20..31																										Nop

c) Indicad la dirección o las direcciones de la memoria ROM_Q+ y su contenido o sus contenidos para implementar correctamente el paso del nodo/estado 1 al 17. (**0,25 puntos**)

Ejercicio 6 **(1 punto)**

Especificad el camino crítico (escribiendo la suma ordenada de los tiempos de propagación de las puertas/bloques por los que pasa) y el tiempo de este camino (sumando los términos de la expresión anterior) para que el computador SISC von Neumann pueda implementar correctamente las acciones que se deben realizar en el nodo/ciclo Bnz. (Este camino sería el camino critico del computador y su tiempo el tiempo de ciclo mínimo si todos los nodos del grafo de estados del la unidad de control necesitaran menos tiempo que el nodo Bnz). Suponed que los tiempos de propagación de las puertas/bloques que forman el computador son los siguientes (los tiempos de la ALU son desde las entradas X e Y hasta el bus de salida W y la señal z):

T_p(And-2) = T_p(Or-2) = 25 u.t.

Tp(Not) = 10 u.t.

T_p(MUX-2-1) = 50 u.t.

T_p(MUX-4-1) = 100 u.t.

T_p(MUX-8-1) = 150 u.t.

T_p(ALU-slow) = 750 u.t; Tp de la ALU para las operaciones/funciones lentas: ADD, SUB, CMP* .

T_p(ALU-quick) = 200 u.t; Tp de la ALU para las operaciones/funciones rápidas: cualquier otra distinta de ADD, SUB, CMP* .

T_{acc}(MEM) = 1000 u.t; Tiempo de acceso (para la lectura o escritura) a la memoria del SISC, tanto para acceso a word como a Byte.

Tp(Dec-3-8) = 100 u.t.

T_p(REG) = 100 u.t; Tiempo de propagación de un registro.

T_p(ROM_OUT) = Tp(ROM_Q+) = 150 u.t; Tp de las dos ROMs que implementan el grafo de estados de la UC.

Recordad que un registro con señal de carga (Ld), REGwLd, está construido con un REG y un MUX-2-1 (no os damos el esquema interno del REGwLd, porque lo tenéis que saber).

Nodo 12, con mnemotécnico de salida del grafo Bnz;

Expresión de la suma de los tiempos que forman el camino:

Tiempo del camino =

