Examen final IC (*Parte 1)*

- Duración del examen: 2:30 horas.
- Los problemas tienen que resolverse en las **HOJAS DE RESPUESTAS**.
- No podéis utilizar calculadora, móvil, apuntes, etc.
- La solución del examen se publicará en Atenea mañana por la mañana.
- La revisión será el **30** de **enero** a las **12:00** en el aula **C6-E101**. Las notas definitivas se publicarán en Atenea al día siguiente.

Para algunos de los ejercicios son útiles las figuras y tablas que se muestran en la hoja aparte
La puntuación de las preguntas de esta parte del examen corresponde a 6 puntos de la nota final del examen.

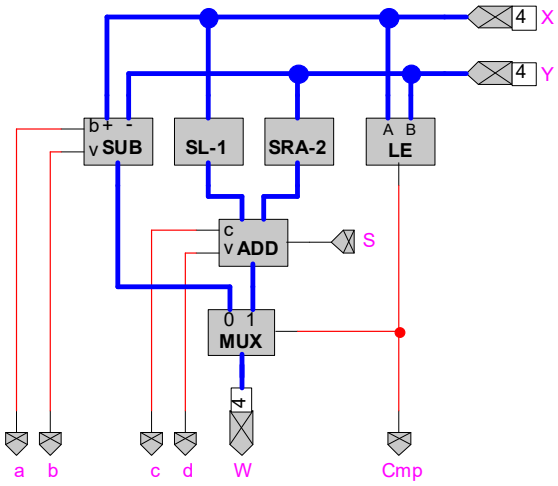
Pregunta 1) *(0.2 puntos)*

Cada fila de la tabla tiene 3 columnas con: el vector de 8 bits X escrito en hexadecimal, el valor que representa X interpretado como un número natural codificado en binario (X_u) y el valor que representa X interpretado como un número entero codificado en complemento a dos (X_s). Completa las casillas que faltan.

X (hexa)	X_u	X_s
	162	
AB		

Pregunta 2) *(0.4 puntos)*

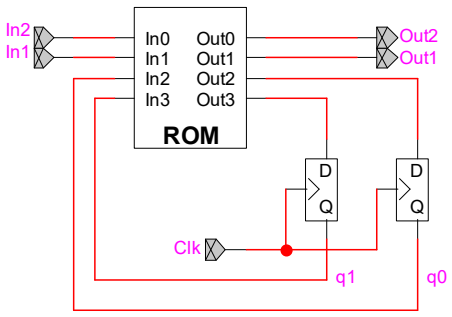
Dado el siguiente circuito combinacional, los valores de los vectores de 4 bits de entrada **X** e **Y** y el valor de la señal **s**, rellenad la tabla indicando el valor de las señales de salida a, b, c, d y Cmp, el valor del vector de 4 bits de la salida W y su interpretación como número natural (W_u) y número entero (W_s) para cada combinación de valores de entrada (cada fila de la tabla). El bloque LE realiza la comparación con signo $A \leq B$ activando el bit de salida a 1 cuando es cierta.



s	X	Y	a	b	c	d	Cmp	W *(4 bits)*	W_u	W_s
1	1100	1011								
1	0110	0101								

Pregunta 3) *(0.3 puntos)*

Dibuja el grafo de estados del siguiente circuito (no olvidéis la leyenda) sabiendo que el valor inicial de los biestables es q_1=0 i q_0=0.



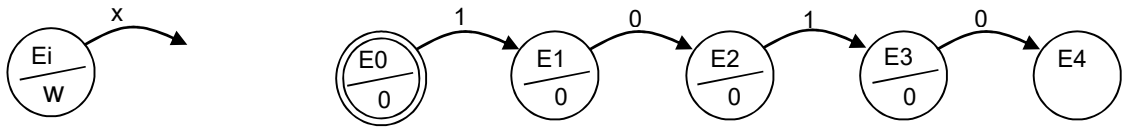
Contenido ROM_Q+

	q_1^+	q_0^+	Out1	Out2
M[0]	0	0	0	0
M[1]	0	0	0	0
M[2]	0	1	0	0
M[3]	0	1	0	0
M[4]	0	1	1	1
M[5]	1	0	1	1
M[6]	0	1	1	1
M[7]	1	0	1	1
M[8]	1	1	0	1
M[9]	1	1	0	1
M[10]	1	1	0	1
M[11]	1	1	0	1
M[12]	0	0	1	0
M[13]	1	1	1	0
M[14]	0	1	1	0
M[15]	0	1	1	0

Pregunta 4) *(0.2 puntos)*

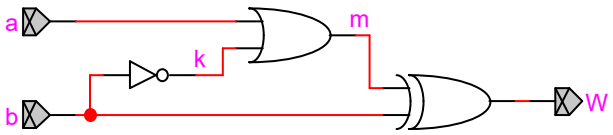
A partir del siguiente texto que describe el funcionamiento de un circuito lógico secuencial (CLS) de Moore, completa el grafo de estados que describe su funcionamiento. El grafo está incompleto y le faltan sólo algunos arcos.

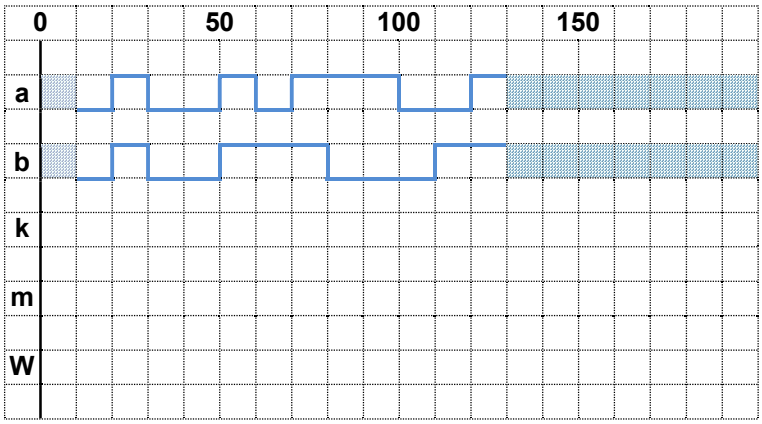
El CLS a diseñar es un reconocedor de una secuencia con solapamiento con una entrada x y una salida w. La salida w vale 1 al ciclo siguiente de que la entrada haya recibido el último bit de la secuencia "1010". En cualquier otro caso la salida w vale 0.



Pregunta 5) *(0.3 puntos)*

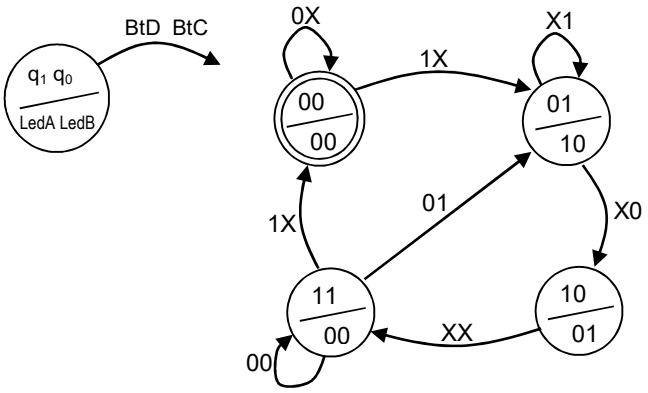
Completad el siguiente cronograma de las señales del esquema lógico sabiendo que los tiempos de propagación de las puertas son: $Tp_{(Not)} = 10$, $Tp_{(Or\text{-}2)} = 20$, $Tp_{(Xor\text{-}2)} = 30$ u.t. Debéis operar adecuadamente con las zonas sombreadas (no se sabe el valor que tienen) y dibujar la señal sombreada cuando no se pueda saber si vale 0 o 1.

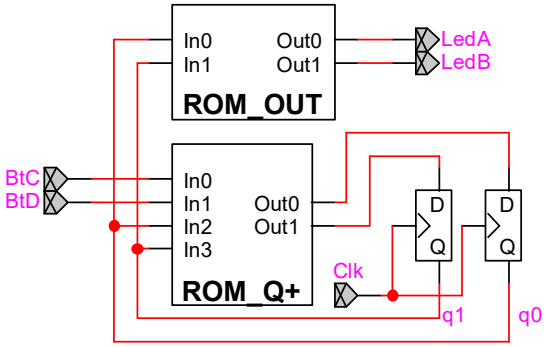




Pregunta 6) *(0.3 puntos)*

El grafo de estados de Moore se implementa mediante el circuito secuencial con dos ROMs de la figura. Responded a los siguiente:





a) ¿Cuál es el contenido, en binario, de las direcciones 0x3 y 0xA de la ROM_Q+?

ROM_Q+ [0x3]= , ROM_Q+ [0xA]=

b) ¿Cuál es el contenido, en binario, de las direcciones 0x1 y 0x2 de la ROM_OUT?

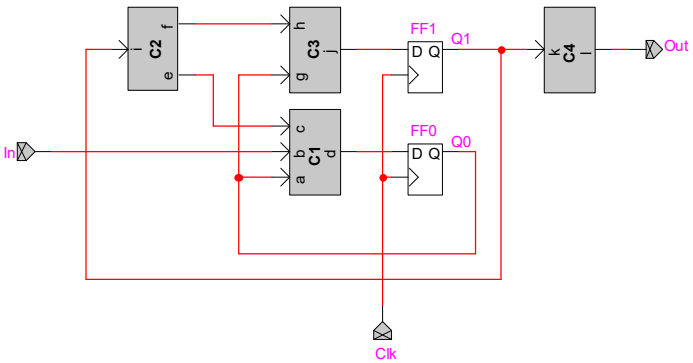
ROM_OUT [0x1]= , ROM_OUT [0x2]=

c) ¿Cuáles son las direcciones, en binario, y sus contenidos, en binario, de la ROM_Q+ que implementan el arco del grafo que va del estado q_1q_0=01 al estado q_1q_0=10? Si se necesitan más de 4 posiciones de la ROM_Q+ escribe solo las 4 primeras.

ROM_Q+ [ ]= , ROM_Q+ [ ]= , ROM_Q+ [ ]= , ROM_Q+ [ ]=

Pregunta 7) *(0.3 puntos)*

El siguiente circuito secuencial está inmerso en un sistema más complejo. Se sabe que la entrada In se estabiliza pasadas 70 u.t. desde la llegada del flanco ascendente de reloj y la salida Out tiene que estar estable 60 u.t. antes de la llegada del flanco ascendente de reloj. Deseamos calcular el tiempo de ciclo mínimo del sistema suponiendo que el camino crítico pasa por este circuito. Los bloques C1, C2, C3 y C4 son circuitos combinacionales cuya implementación interna desconocemos.



Los tiempos de propagación de los bloques combinacionales que hay en este circuito son los siguientes ($TCi_{n\text{-}m}$ indica el tiempo de propagación del bloque Ci desde la entrada *n* a la salida *m* medido en u.t.):

T_{C1}		T_{C2}		T_{C3}		T_{C4}	
a-d	80	i-e	70	g-j	80	k-l	30
b-d	90	i-f	50	h-j	20		
c-d	20						

El tiempo de propagación de cualquier biestable del sistema es de 100 u.t.

7.a) Escribid **todos los caminos críticos** que haya listando la secuencia de bloques que los forman e indicando la entrada y salida de cada bloque por la que pasa el camino. *(0.2 puntos)*

7.b) ¿Cuál es el tiempo de ciclo mínimo del sistema? *(0.1 puntos)*

Pregunta 8) *(0.4 puntos)*

x_2	x_1	x_0	w
0	0	0	1
0	0	1	1
0	1	0	X
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	X
1	1	1	1

Dada la tabla de verdad correspondiente a una función que calcula w, responde a las siguientes preguntas.

8.a) Dibujad el mapa de Karnaugh marcando claramente las agrupaciones de unos adecuadas para obtener la expresión mínima en suma de productos de la función w. Escribid la expresión mínima en suma de productos de w. *(0.2 puntos)*

8.b) Si implementamos directamente la expresión en suma de minterms de la función w considerando las x como 0, ¿Cuántas puertas And y OR y de cuántas entradas hacen falta? *(0.1 puntos)*

8.c) Si implementamos la función w con una ROM. ¿Cuántas palabras y cuántos bits por palabra tiene la ROM? *(0.1 puntos)*

Pregunta 9) *(0.4 puntos)*

9.a) Escribid la fórmula que da el valor de un número natural en función de los 3 dígitos que lo representan en el sistema convencional en base 7. *(0.1 puntos)*

9.b) Escribid la fórmula que da el valor de un número entero en función de los n bits en complemento a dos que lo representan. *(0.1 puntos)*

9.c) Expresad el rango de los números enteros que se pueden representar en el sistema convencional en base 5 para el caso de un vector X de 4 dígitos. *(0.1 puntos)*

9.d) ¿Cuál es el número mínimo de bits necesarios para representar los siguientes números enteros en complemento a dos? *(0.1 puntos)*

32: 1: -1:

Pregunta 10) *(1 punto)*

A partir de la descripción de la funcionalidad del procesador de propósito específico (PPE) que se da a continuación, diseña una implementación formada por una unidad de proceso y una unidad de control, ambas de propósito específico. Para la unidad de proceso (UP) haz un diseño ad-hoc usando los bloques combinacionales y secuenciales vistos en clase. Especifica la unidad de control (UC) mediante un grafo de estados de Moore.

El PPE a diseñar debe realizar indefinidamente la siguiente operación. En el ciclo en que la señal de entrada Inicio vale 1, simultáneamente por la entrada X llega un dato y en los dos ciclos siguientes llegan dos datos más a razón de uno por ciclo. Al ciclo siguiente de la llegada del último dato, durante un ciclo la salida Fin vale 1 y por la salida W sale el resultado de la siguiente operación:

```
if (Inicio(c) == 1) {
   if ((X(c)-2*X(c+1))<=0) {
          W(c+3) = X(c) - 2*X(c+1);
   } else {
          W(c+3) = X(c) - 2*X(c+1) + X(c+2);
   }
   Fin(c+3)= 1;
}
```

En el resto de ciclos el valor de la señal W no importa. En el ciclo que Fin vale 1 y los siguientes, hay que comprobar si Inicio vale 1, lo que indica que comienza una nueva secuencia. Si en algún momento durante la recepción de los datos la señal Inicio vuelve a valer 1 se debe abortar el proceso y volver a procesar una nueva secuencia. Todos los buses son de 8 bits y los valores son números **enteros**. En el diseño no hace falta tener en cuenta si durante las operaciones se producen desbordamientos (carrys, borrows y overflows).

10.a) Rellena el siguiente cronograma. Escribe la secuencia de valores para Fin y W para los ciclos del 4 al 13. Cuando el valor de la señal W no importe indícalo con XX. *(0.2 puntos)*

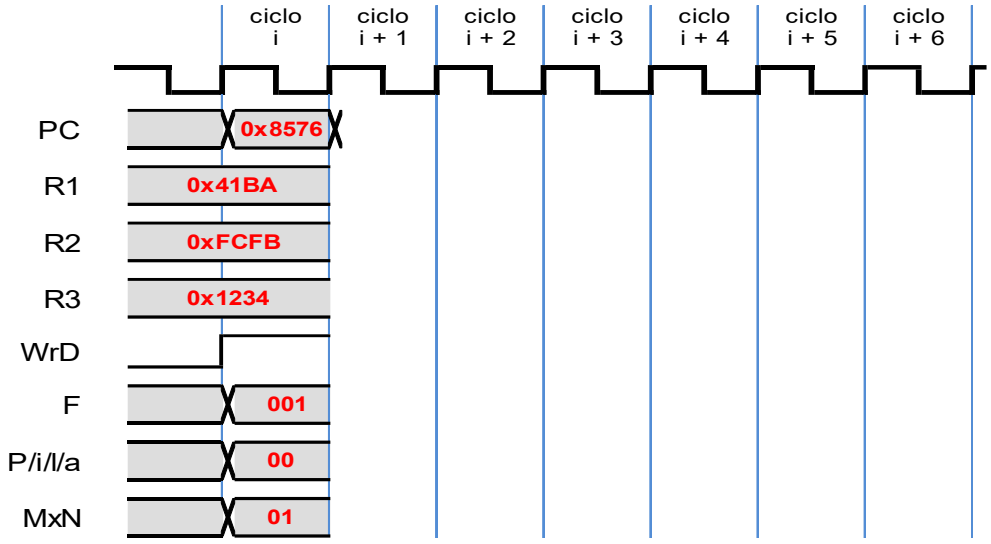
Ciclo	00	01	02	03	04	05	06	07	08	09	10	11	12	13
X	0x03	0x02	0x04	0x7A	0x18	0x06	0x02	0x12	0x09	0x21	0x03	0xFE	0x9E	0x9E
Inicio	1	0	0	0	1	0	0	1	0	1	0	0	1	0
Fin	0	0	0	1										
W	XX	XX	XX	0xFF										

10.b) Dibuja el circuito de la unidad de proceso (UP) usando los bloques de la documentación de la asignatura. Dibuja también el grafo de estados de la unidad de control (UC) correspondiente. *(0.8 puntos)*

Pregunta 11) *(0.3 puntos)*

Dado el siguiente fragmento de código donde en el *ciclo i* se ejecuta la instrucción `MOVI R3,0x9C`, rellenad el siguiente cronograma indicando el valor de las señales de la UCG o UPG y los valores de los registros. *(nota: para facilitar la lectura del cronograma no hace falta que pongáis el **0x** delante de los valores hexadecimales de los registros. Ya se sobreentienden)*

```
MOVI R3,0x9C
MOVHI R3,0x00
AND R2, R2, R1
SHA R1, R2, R3
BNZ R1, -3
OUT 12, R2
```



Pregunta 12) *(0.3 puntos)*

Completa la siguiente 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. 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 0xCAFE, el contenido de todos los registros es 0xFOFA, el byte contenido en todas las direcciones pares de la memoria es 0x24 y el de todas las impares es 0x91 y el contenido de todos los puertos de entrada es 1 y el de los de salida es 2. Utilizad la notación MEMb[0x...]=0x... para indicar cualquier cambio en la memoria y PORTIN[0x...]=0x... para los puertos de entrada y PORTOUT[0x...]=0x... para los de salida.

Lenguaje máquina SISA	Lenguaje ensamblador SISA	Cambios en el estado del computador
`0x145C`		
	`ST -2(R6), R1`	
`0x7800`		

Pregunta 13) *(0.3 puntos)*

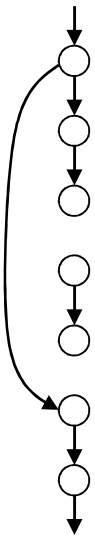
Indica el valor que debe tener cada uno de los bits de la palabra de control de la UPG (sin subsistema de I/O ni memoria, ver anexo) para que realice, durante un ciclo, la acción concreta especificada mediante el mnemotécnico. Indicad con x las casillas cuyo valor **no importe** para la ejecución de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

Mnemotécnico	@A	@B	Rb/N	OP	F	In/Alu	@D	WrD	N *(hexa)*
XOR R2, R7, R4									
OUT R6 // MOVEI R1, 0x82									
SHAI –, R2, -8									

Pregunta 14) *(0.6 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 la funcionalidad descrita mediante el siguiente código en C (el código no tiene que hacer algo útil). 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. En las comparaciones, hay que interpretar los datos como valores **enteros**.

```
if (R6<714) {
    while (R4 <= 50) {
        R5 = (R5 - 1) - R4;
        R4 = R4 + 5;
    }
} else {
    R6 = R6 + 2
}
R5 = R6 / 8;
```



```
CMP [      ] ,714
CMPLEI  - , [      ]
SUBI    R7, [      ]
        [      ] ,R4
        [      ] ,5
ADDI    R6, [      ]
SH  [  ]    R5, [      ]
```

Pregunta 15) *(0.7 puntos)*

Completad el fragmento de programa en lenguaje ensamblador SISA para que el procesador formado por la unidad de control de propósito general (UCG) junto con la UPG realice la funcionalidad descrita en el siguiente fragmento de código en C. El código SISA ya escrito solamente escribe en los registros `R0`, `R1`, `R2` (para implementar la funcionalidad descrita) y `R7` (para valores temporales). En las comparaciones, hay que interpretar los datos como valores **enteros**. Rellenad la parte que falta. No se pueden utilizar etiquetas para los saltos.

```
for (R0=-17; R0<=128; R0++) {
    while ((R1>R4) && (R3<R2)) {
        R2=R2-R3;
        R1=R1-1;
    }
}
R2=R2/8;
```

Nota: el símbolo && corresponde a la AND lógica (booleana).

```
MOVI    [      ]
[      ] R7, [      ]
[      ] R7, [      ]
CMP [      ] , R0 , R7
B   [      ] R7, [      ]
[      ] R7 , R1 , R4
B   [      ] R7, [      ]
CMP [      ] R7 , R3 , R2
B   [      ] R7, [      ]
SUB     [      ]
[      ] R1 , R1 , -1
[      ] R7, [      ]
ADDI    R0, [      ]
[      ] R7, [      ]
B   [      ] R7, [      ]
MOVI    R7, [      ]
SH  [      ] R2, [      ]
```