

Examen Final (1ª parte)

- a) Duración del examen: 2 horas.
- b) La solución se tiene que escribir en el espacio reservado para ello en el propio enunciado.
- c) No podéis utilizar calculadora, móvil, apuntes, etc.
- d) La solución del examen se publicará en Atenea mañana.

Ejercicio 1 (0,5 puntos)

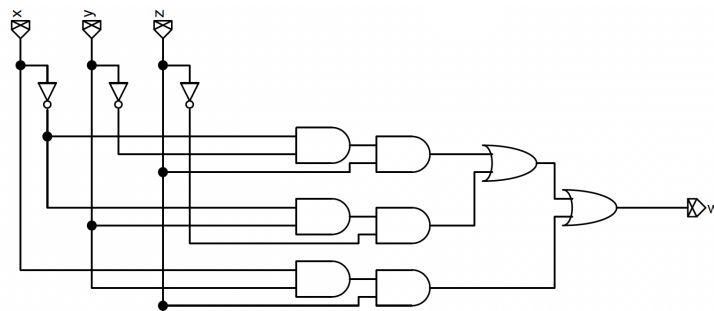
Cada fila de la tabla tiene 4 columnas con: el vector X de 8 bits, X expresado en hexadecimal, el valor en decimal, Xu, que representa X interpretado como un número natural codificado en binario, y el valor en decimal, Xs, que representa X interpretado como un número entero codificado en binario en Ca2. Completa todas las casillas vacías. **(cada fila con 1 error -0,2. Min 0)**

X (binario)	X (hexadecimal)	Xu	Xs
1111 0000	0xF0	240	-16
0011 0011	0x33	51	51
1111 0100	0xF4	244	-12

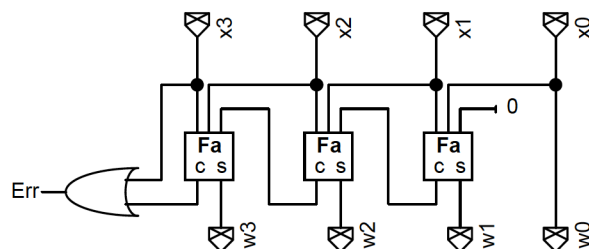
Ejercicio 2 (1 puntos)

Dada la siguiente tabla de verdad, sintetiza el circuito lógico combinacional que define la función lógica w como suma de minterms utilizando sólo puertas NOT, AND-2 y OR-2 **(corrección binaria)**

x	y	z	w
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

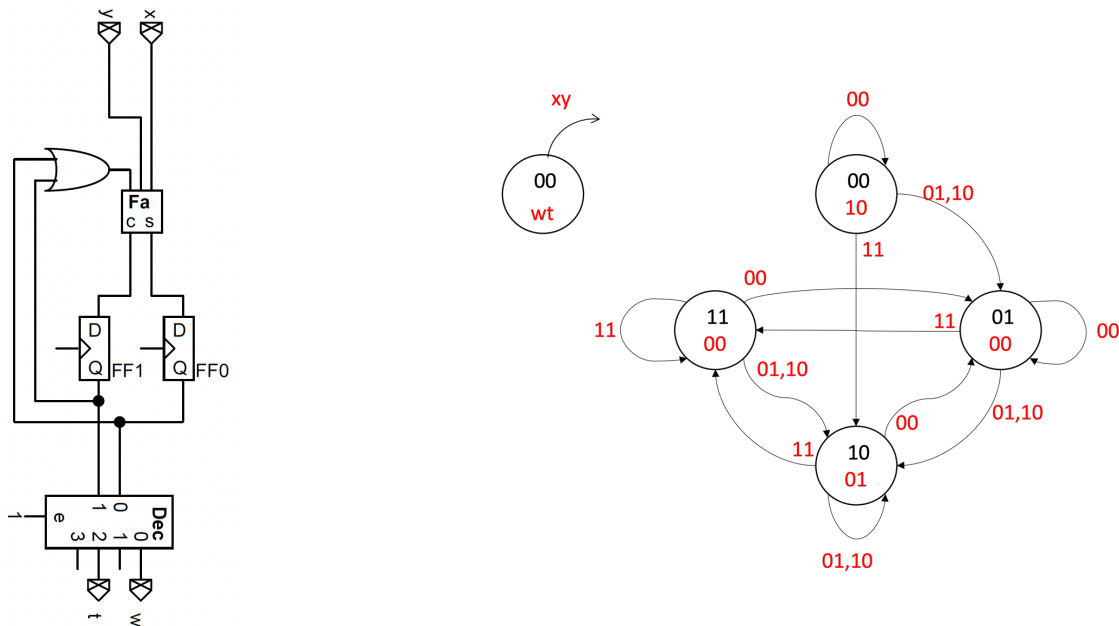
**Ejercicio 3 (1 puntos)**

Utilizando sólo 3 FA y una puerta OR-2, implementa un circuito que calcula 3X donde X es un número natural codificado con 4 bits y el resultado también es número natural codificado con 4 bits. El Circuito debe indicar también si el resultado no es representable en 4 bits. **(Suma correcta 0,5, si suma correcta, señal de error 0,5)**



Ejercicio 4 (1,5 puntos)

Dibujad el grafo de estados del siguiente circuito secuencial. No olvidéis la leyenda. **(-0,5 por cada estado con errores. Un error puede estar en la salida o en cualquiera de las aristas que salen del nodo)**

**Ejercicio 5 (1,5 puntos)**

Si el CLS del ejercicio 4 (no hace falta haberlo resuelto para hacer este problema) se hubiese implementado con una **única ROM** y un **único multiplexor** de buses:

- a) ¿Cuántos biestables serían necesarios? ¿Cuántas palabras tendría la ROM? ¿cuántos bits por palabra? ¿Cuánto valdría E si el multiplexor de buses fuese un MUX-E-1? ¿Cuántos bits de anchura los buses del multiplexor de buses? **(0,5 puntos binario)**

# Biestables	# Palabras ROM	# Bits por/ Palabra ROM	E	# Bits/Bus
2	$2^2 = 4$	$(2^2) * 2 + 2 = 10$	4	2

- b) ¿Cuál sería el camino crítico y el tiempo de propagación de ese circuito en su implementación original (la del problema 4)? Suponiendo los siguientes tiempos de propagación: **(0,5 puntos binario)**

TP (FA) = 120 u.t.

TP (OR-2) = 20 u.t.

TP (Biestable) = 100 u.t.

TP (MUX) = 100 u.t.

TP (Dec) = 40 u.t.

TP (ROM) = 50 u.t.

Las **entradas "x" y "y"** tardan 175 u.t. en estabilizarse.

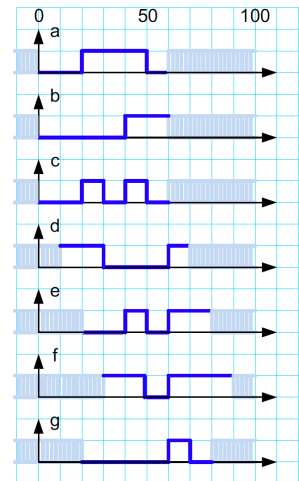
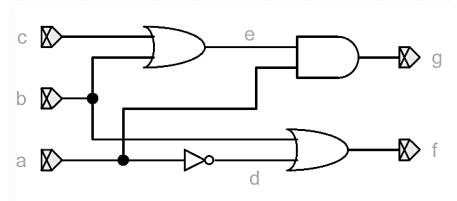
La **salida w** debe estar estable 30 u.t. antes del final de ciclo y la salida t 75 u.t.

Camino crítico	Tiempo propagación
x/y - FA - FF0/1	$175 + 120 = 295 \text{ u.t.}$

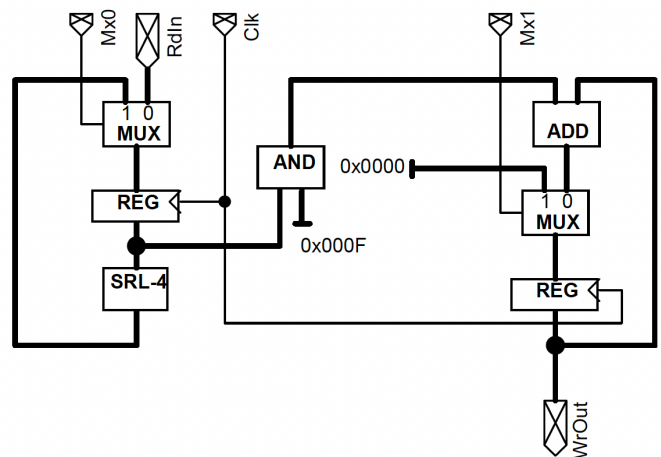
- c) ¿Cuál sería el camino crítico y el tiempo de propagación de ese circuito en su implementación original (la del problema 5)? Suponiendo los mismos tiempos de propagación que el apartado anterior. **(0,5 puntos binario)**

Camino crítico	Tiempo propagación
x/y - MUX - Reg	$175 + 100 = 275 \text{ u.t.}$

Completad el siguiente cronograma de las señales del esquema lógico sabiendo que los tiempos de propagación de las puertas son: $T_p(\text{Not}) = 10$, $T_p(\text{And}) = 20$, $T_p(\text{Or}) = 20$ 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. **(Cada señal con algún error -0,2. Mín = 0)**

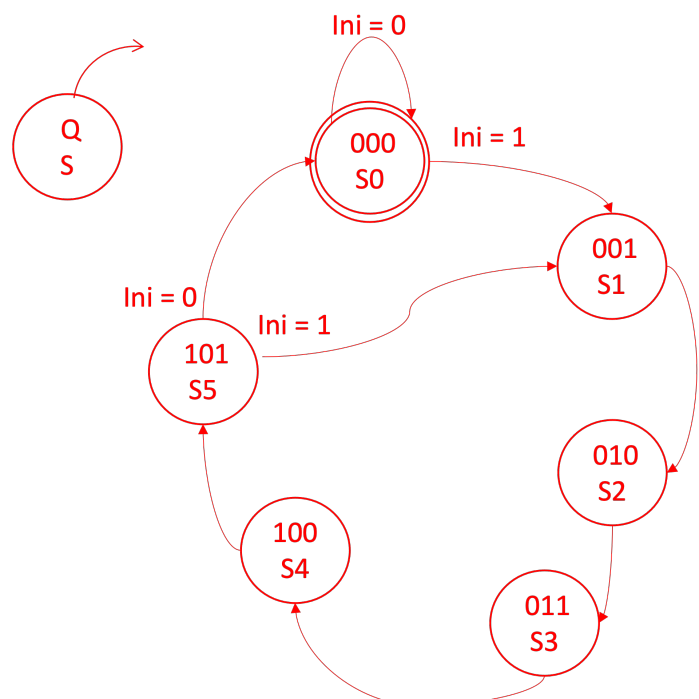


Cada cierto tiempo, un determinado dispositivo envía 4 valores naturales codificados en 4 bits cada uno. Para ahorrar espacio, el dispositivo envía los 4 valores a la vez por único bus de 16 bits. Es decir, para enviar los siguientes valores, de 4 bits cada uno 1, 2, 3, y 4, los empaquetará en el valor de 16 bits 0x1234 antes de enviarlos. Tenemos que implementar un PPE que reciba esos 4 valores por RdIN en el mismo ciclo que Ini valga 1 y dé el resultado de sumarlos por WrOut a la vez que Fi valga 1. Una vez iniciado el cálculo ignoraremos ini=1 hasta que este haya finalizado. Por ejemplo, si en el ciclo c, ini vale uno y RdIN vale 0x1234, a cabo de algunos ciclos Fi=1 y WrOut = $0x1+0x2+0x3+0x4 = 0xA$. Para ello tenemos la UP de la figura y debéis implementar la UC que la utilice para resolver el problema. **Puede que sobren estados en la tabla. Se debe permitir una nueva entrada lo antes posible.**



(-0,5 por cada estado con errores. Un error puede estar en la salida o en cualquiera de las aristas que salen del nodo)

S	Mx0	Mx1	Fi
S0	0	1	0
S1	1	0	0
S2	1	0	0
S3	1	0	0
S4	1	0	0
S5	0	1	1
S6			
S7			



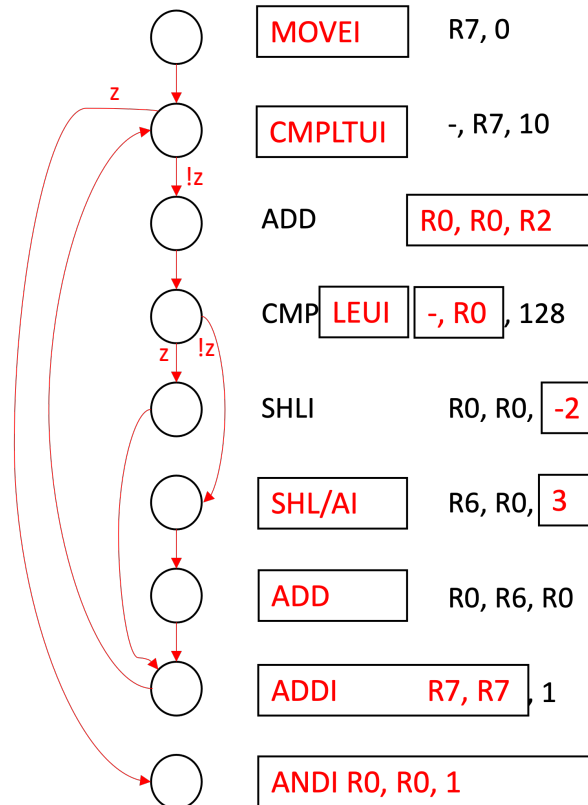
Ejercicio 8 (1 puntos)

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. Todos los valores se representarán como naturales. Podéis utilizar los registros que queráis para los valores temporales.

(-0,25 por cada estado con errores.

Un error puede estar en la salida o en cualquiera de las aristas que salen del nodo. Errores en I y U primera vez.)

```
For (i=0; i<10; i=i+1) do
  R0 = R0+R2
  If (R0>128) {
    R0 = R0 / 4
  } else {
    R0 = R0 * 9
  }
}
R0 = AND (R0, 0x1)
```



sólo se cuentan la

Ejercicio 9 (1 puntos)

Indicad los bits **relevantes** de la palabra de control de la **máquina SISC Harvard Uniciclo** para las siguientes instrucciones.

X = Mínimo número de filas y/o columnas que agrupen todos los errores. Nota = Max (0, 1-X+0,5)

	@A	@B	Rb/N	OP	F	-/!/a	@D	Wd	Rd-In	Wr-Out	Wr-Mem	byte	N (Hexa)	ADDR-IO (Hexa)
OUT 6, R5	101	xxx	x	xx	xxx	xx	xxx	0	0	1	0	x	0xFFFF	0x06
BZ R0, 0xF0	000	xxx	x	10	000	xx	xxx	0	0	0	0	x	0xFFE0	0xFF
ADD R3, R2, R1	010	001	1	00	100	00	011	1	0	0	0	x	0xFFFF	0xFF

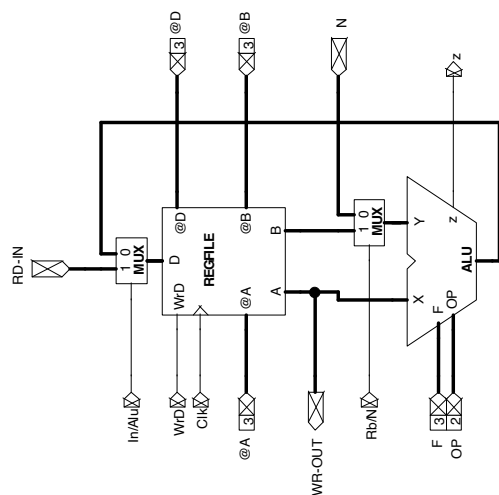
Funcionalidades ALU

F			OP			
b_2	b_1	b_0	11	10	01	00
0	0	0	0000	X	CMPLE(X,Y)	AND(X,Y)
0	0	1	0001	Y	CMPLE(X,Y)	OR(X,Y)
0	1	0	0010	MOVHI(X,Y)	----	XOR(X,Y)
0	1	1	0011	----	CMPEQ(X,Y)	NOT(X)
1	0	0	0100	----	CMPLE(X,Y)	ADD(X,Y)
1	0	1	0101	----	CMPLE(X,Y)	SUB(X,Y)
1	1	0	0110	0000	----	SHA(X,Y)
1	1	1	0111	0000	0000	SHL(X,Y)

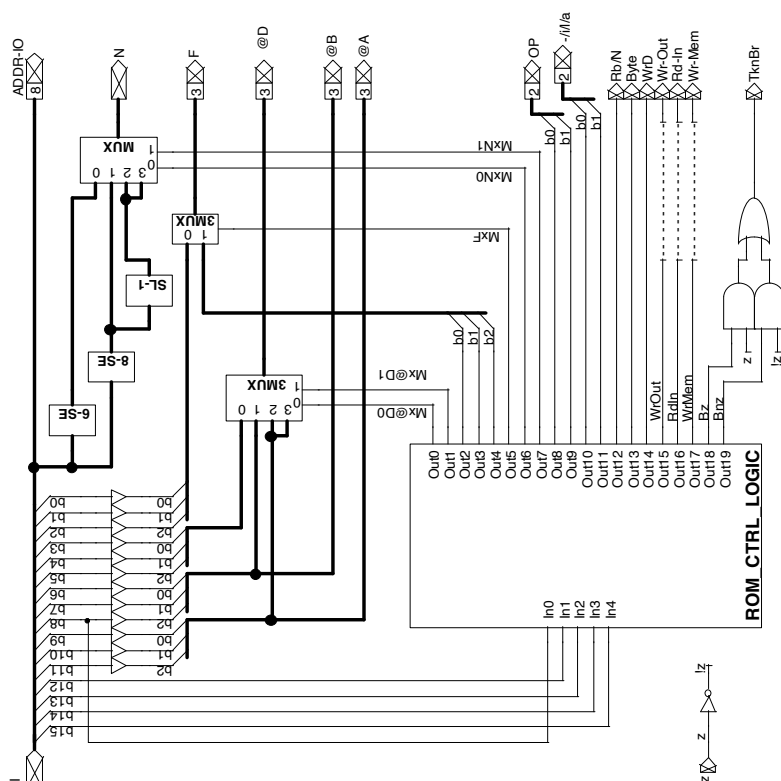
Formato Instrucciones SISA-I

Name												Mnemonic
Logic and Arithmetic Operations												AND, OR, XOR, NOT, ADD, SUB, SHA, SHL
Compare Signed and Unsigned												CMPLT, CMPLF, -, CMPEQ, CMPELT, CMPELU, -, -
Add Immediate												ADDI
Load												LD
Store												ST
Load Byte												LDB
Store Byte												STB
Branch on Zero												BZ
Branch on Not Zero												BNZ
Move Immediate												MOVI
Move Immediate High												MOVHI
Input												IN
Output												OUT
Future extensions												Future extensions

UPG básica



Lógica de control del SISC Harvard Uniciclo



Computador SISC Harvard Uniciclo (UCG + UPG + IO + MEM)

