Examen Final P1 *(14 de Enero del 2020)* IC - 19-20-Q1

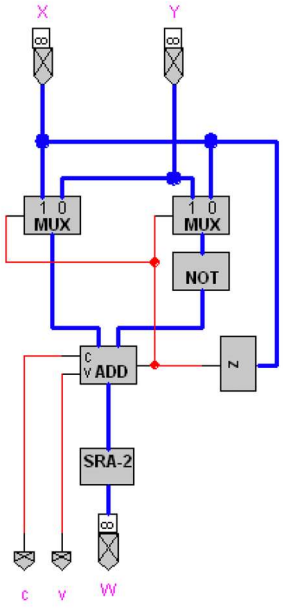
Apellidos y nombre: …… Grupo:……

Examen Final (1ª parte)

- Duración del examen: 2 hora y 30 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 del examen se publicará en el Racó de la FIB y en Atenea mañana.
- La revisión será el **XX** de **Enero** a las **11:00** en el aula **C6-E101** Las notas definitivas se publicarán en el Racó de la FIB al día siguiente.

Ejercicio 1 *(1,5 puntos)*

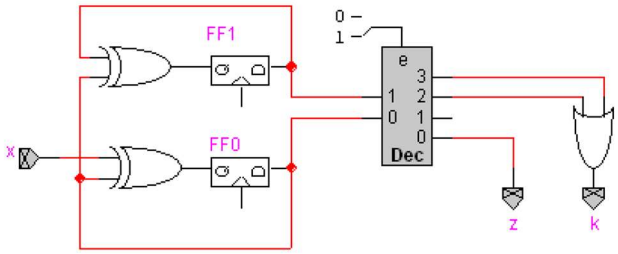
Dado el siguiente circuito combinacional, los valores de los vectores de 8 bits de entrada X e Y, rellenad la tabla indicando el valor de las señales de salida c, v, el valor del vector de 8 bits de la salida W para cada combinación de valores de entrada (cada fila de la tabla). También tenéis que escribir el valor que representa W interpretado como un número natural codificado en binario (W_u) y el valor que representa W interpretado como un número entero codificado en complemento a dos (W_s). No os descuidéis de la señal de entrada carry del sumador. *(Corrección: 0,5 cada fila correcta, 0,3 si hay 1 error, 0 si hay mas errores)*

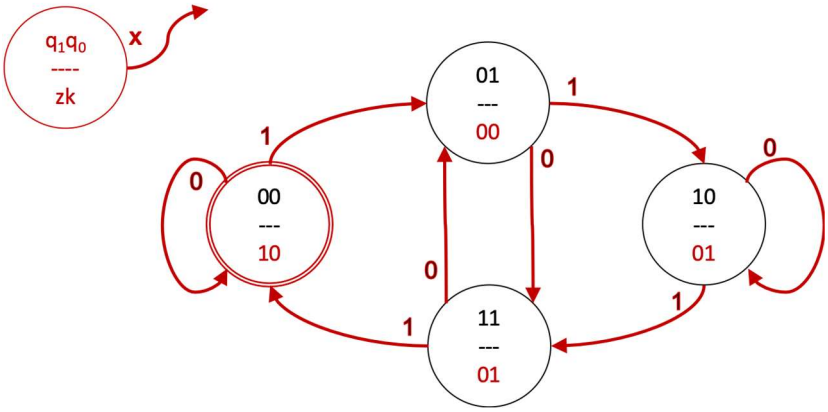


X	Y	c	v	W	W_u	W_s
10001000	00010001	0	1	1110 0010	226	-30
11111111	00011111	0	0	0000 0111	7	7
00000000	10000000	0	1	1110 0000	224	-32

Ejercicio 2 *(0,5 puntos)*

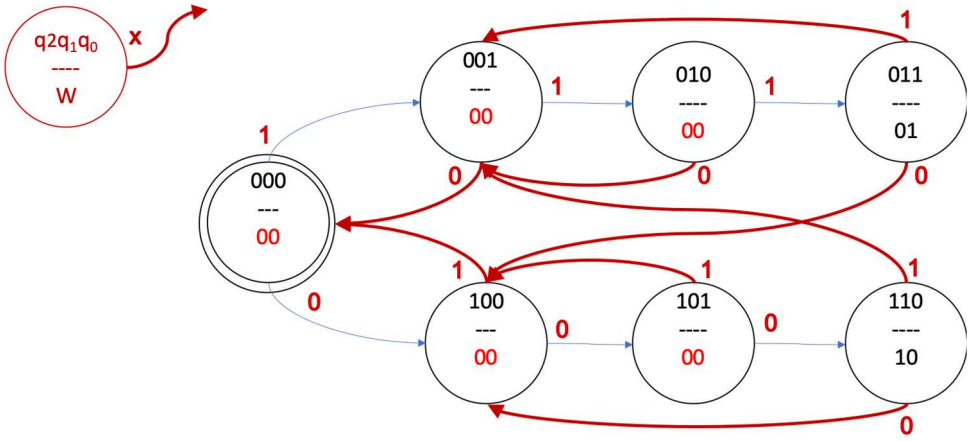
Tenemos el siguiente circuito. Sabiendo que el valor inicial de los biestables es q1=0 i q0=0, completa el grafo de estados del circuito (**no olvidéis la leyenda, ni marcar el estado inicial**). *(Corrección: 0 si no hay leyenda o es incorrecta. -0,2 por cada estado incorrecto ya sea por las salidas o por las transiciones que de él salen)*





Ejercicio 3 *(1,5 puntos)*

Queremos implementar un circuito que detecte si el número entre 0's y 1's de la entrada (x) se desequilibra demasiado. Se considera desequilibrado si llegan 3 ceros o más que unos o 3 unos o mas que ceros. Una vez detectado el desequilibrio se dará una señal de alarma durante un ciclo codificada de la siguiente forma: w=10 si hay más ceros que unos y W=01 si hay mas unos que ceros. Una vez dada la alarma se volverá a iniciar la comprobación del equilibrio teniendo ya en cuenta la entrada (x) en el ciclo que hemos dado la señal de alarma. Mientras no demos alarma, el valor de W ha de ser 00. Completad el grafo del circuito (no olvidéis la leyenda). Por ejemplo, la secuencia 010111011 tiene 3 ceros y 6 unos por lo que al llegar el último uno debería activar la alarma de demasiados unos y volver a iniciar la comprobación a partir de siguiente valor de x, el que ha llegado en ciclo en que se ha activado la alarma. *(Corrección: 0 si no hay leyenda o es incorrecta. -0,3 por cada estado incorrecto ya sea por las salidas o por las transiciones que de él salen)*



Ejercicio 4 *(0,75 puntos)*

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

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. *(0.5 puntos)*
(Corrección tabla: 0,4 si la tabla está bien, 0,2 si las agrupaciones son correctas, pero no mínimas)
(Corrección expresión: 0,1 - binaria y sólo si la tabla esta bien)

x\yz	00	01	11	10
0	1	x	1	x
1	0	1	1	0

W = ¡x + z

x	*y*	*z*	*w*
0	*0*	*0*	***1***
0	*0*	*1*	***X***
0	*1*	*0*	***X***
0	*1*	*1*	***1***
1	*0*	*0*	***0***
1	*0*	*1*	***1***
1	*1*	*0*	***0***
1	*1*	*1*	***1***

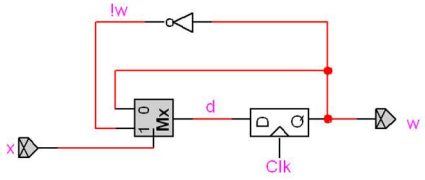
b) Si implementamos **directamente** la expresión (**sin optimizar**) en suma de minterms de la función w considerando las x como 0, ¿Cuántas puertas And y OR y con cuantas entradas hacen falta? *(0.1 puntos)* *(Corrección: 0,1 - binaria)*

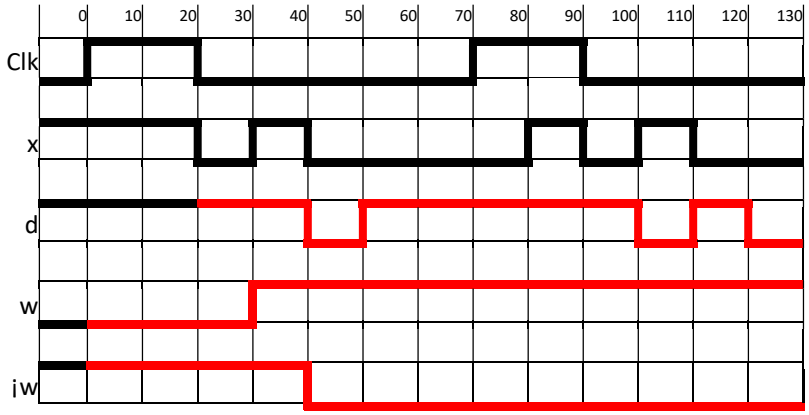
4 puertas AND de 3 entradas y una OR de 4 entradas

c) Si implementamos la función w con una ROM. ¿Cuántas palabras y cuántos bits por palabra tiene la ROM? *(0.15 puntos)*
(Corrección: 0,15 - binaria)

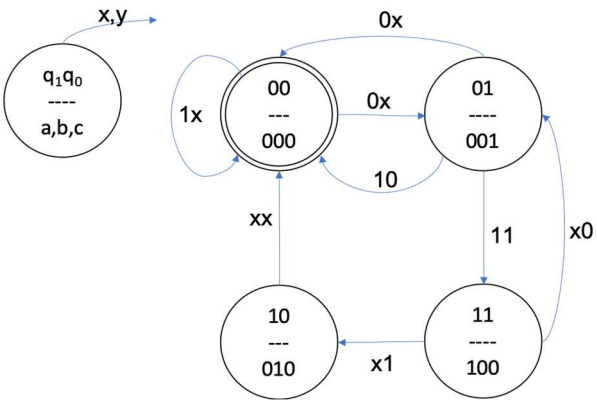
8 palabras de 1 bit

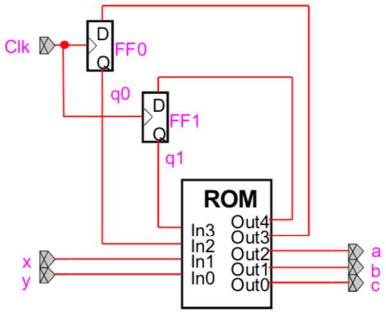
Ejercicio 5 *(0,75 puntos)*
Completad el siguiente cronograma de las señales del esquema lógico considerando que los tiempos de propagación son: Tp(Not)=10u.t., Tp(Mx)=20.t., Tp(biestable)=30.t. *(Corrección: 0,25 por cada señal correcta)*





Ejercicio 6 *(0,75 puntos)*
Si se quiere implementar el grafo de estados siguiente con un circuito secuencial y una sola ROM tal y como se muestra en la figura.





Indicad, en hexadecimal, el contenido de las siguientes direcciones de la ROM.
(Corrección: 0,25 por cada posición de la ROM correcta)

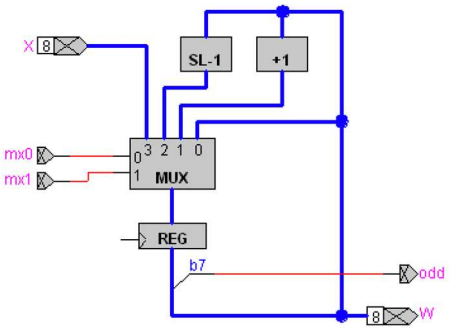
ROM[0x2]= 0x00	ROM[0x7]= 0x19	ROM[0xF]= 0x14

Ejercicio 7 *(0,5 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. *(Corrección: -0,2 por cada casilla incorrecta)*

Lenguaje máquina SISA	Lenguaje ensamblador SISA
0x97AA	MOVHI R3,0xAA
0x71C6	JALR R7, R0
0x2B3F	ADDI R4, R5, -1

Ejercicio 8 *(1,5 punto)*

Queremos implementar un PPE que nos ayude a encriptar una entrada de 8 bits que llega por el bus X en el mismo ciclo que se activa la señal ini=1. La encriptación que haremos, muy sencilla, rotará dos bits hacia la izquierda, es decir, que si la entrada es $b_7b_6b_5b_4b_3b_2b_1b_0$ la salida será $b_5b_4b_3b_2b_1b_0b_7b_6$. Un ejemplo con números es el siguiente: si X=10001010 entonces W debería ser 00101010.

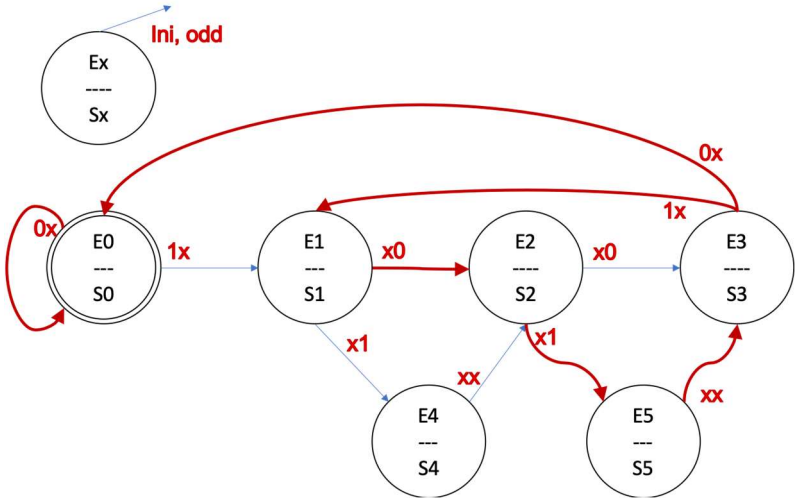


Una vez tengamos la salida disponible en el bus W, activaremos la señal de fi.

Durante la encriptación de una palabra se ignorará la señal ini, pero sí se tendrá en cuenta en el ciclo en que fi valga 1.

Os pedimos que completéis el grafo de la UC que implemente este circuito utilizando la siguiente UP. No os olvidéis de completar la leyenda. *(Corrección: 0 si no hay leyenda o es incorrecta. -0,3 por cada estado incorrecto ya sea por las salidas o por las transiciones que de él salen)*

	mx1	mx0	fi
S0	1	1	0
S1	1	0	0
S2	1	0	0
S3	1	1	1
S4	0	1	0
S5	0	1	0



Ejercicio 9 *(0,75 puntos)*
Calcula el camino crítico (si hay más de uno, basta con uno) y el tiempo mínimo de ciclo del circuito del Ejercicio 8 suponiendo que la UC se ha implementado con una ROM y un multiplexor de buses. Asume los siguientes tiempos de propagación: Tp(ADD)=660u.t., Tp(MUX)=50 u.t. (cualquier multiplexor de cualquier medida), Tp (SHL-1) = 0 u.t., Tp(Reg)=100 u.t., Tp(ROM)=60 u.t., todas las entradas necesitan 100u.t. para estabilizarse y las salidas deben estar 50u.t. estables antes del final del ciclo.
(Corrección: 0,5 si el camino crítico es correcto. 0,25 si el tiempo de clico es correcto (sólo si el camino crítico era correcto)

Caminos críticos: Registro – incrementador – Multiplexor (– Regsitro)

Tiempo mínimo de ciclo: 100 + 660 + 50 = 810 u.t.

Ejercicio 10 *(1,5 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 fragmento de código en C. El código SISA ya escrito **siempre utiliza el registro R7 para valores temporales**. Todos los datos son **naturales**. Rellenad la parte subrayada que falta. **Utilizad las etiquetas que os hemos puesto siempre que podáis**. *(Corrección: -0,2 por cada instrucción con error)*

```
while (R2==R5) {
   if (R2 <= R3) {
         R5 = R5 - R2
   } else {
         R2 = R2 + 0xff
   }
}
MEMb[R2+10] = 10
```

```
.text
        while:  CMPEQ R7, R2, R5
                BZ R7, while_end
                CMPLTU R7, R3, R2
                BNZ R7, else
        then:   SUB R5, R5, R2
                BZ R7, while
        else:   MOVI R7, 0xFF
                MOVHI R7, 0x00
                ADD R2, R2, R7
                BNZ R7, while
   while_end:   MOVI R7, 10
                STB 10(R2), R7
.end
```