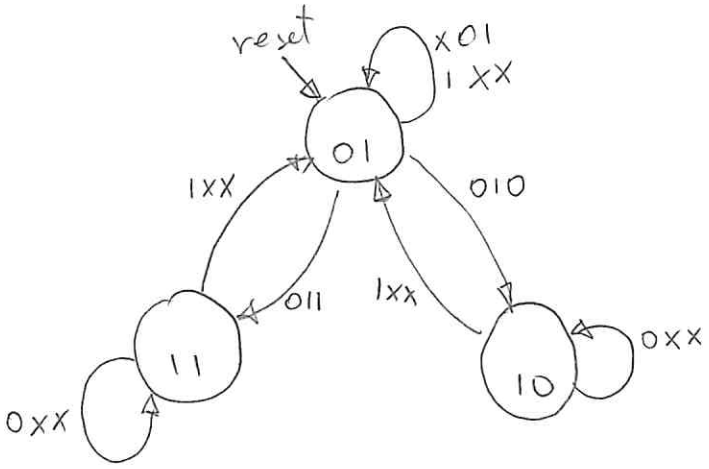
I2 Sistemas Digitales I 2019

Pregunta 1

Máquina 1 Guarda el valor del dado del primer flanco de clock

Entradas: dado D (2 bits)
Fin (cuarto intento) F (1 bit) } FD

Salidas = Estados = valor del Dado en primer intento C (2 bits)



F	P_1	D_0	Q_1	Q_0	Q_1^{t+1}	Q_0^{t+1}
X	0	1	0	1	0	1
0	1	0	0	1	1	0
0	1	1	0	1	1	1
1	X	X	0	1	0	1
0	X	X	1	0	1	0
1	X	X	1	0	0	1
0	X	X	1	1	1	1
1	X	X	1	1	0	1

F=0

D_1D_0 \ Q_1Q_0	00	01	11	10
00	X	X	X	X
01	X	1	1	0
11	X	1	1	0
10	X	0	1	0

F=1

D_1P_0 \ Q_1Q_0	00	01	11	10
00	X	X	X	X
01	X	1	1	1
11	X	1	1	1
10	X	1	1	1

Bit menos significativo

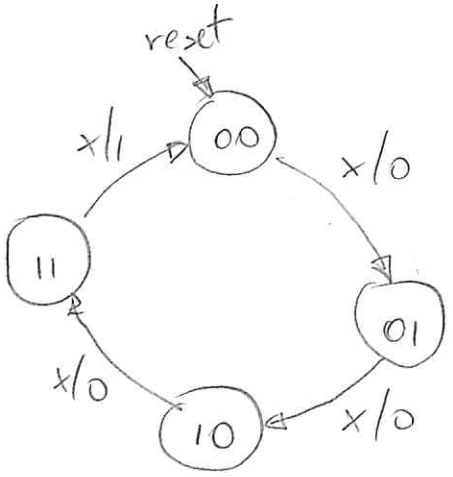
$$Q_0^{t+1} = F + Q_1 Q_0 + D_0 Q_0$$

Máquina 2 lleva la cuenta del número de lanzamientos

Entradas: no hay (solo reset)

Salidas: Fin F (1 bit)

Estados: número de lanzamientos



$Q_1 Q_0$	Q_1^{t+1}	Q_0^{t+1}	F
0 0	0	1	0
0 1	1	0	0
1 0	1	1	0
1 1	0	0	1

Q_1 \ Q_0	0	1
0	1	0
1	1	0

Bit menos significativo $\boxed{Q_0^{t+1} = \overline{Q_0}}$

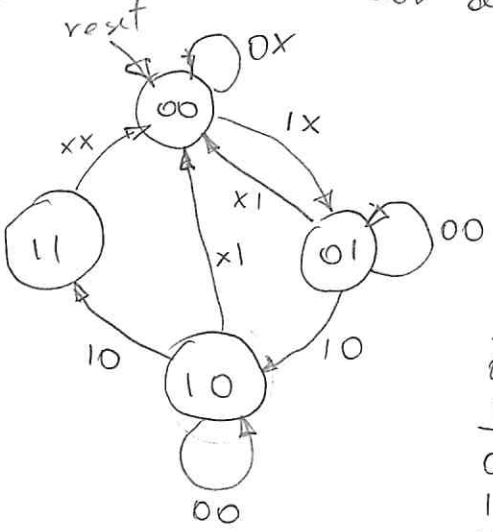
Máquina 3 Contador de repeticiones

Entradas: comparación de números P (1 bit)

Fin (F)

PF

Salidas = Estados = número repeticiones

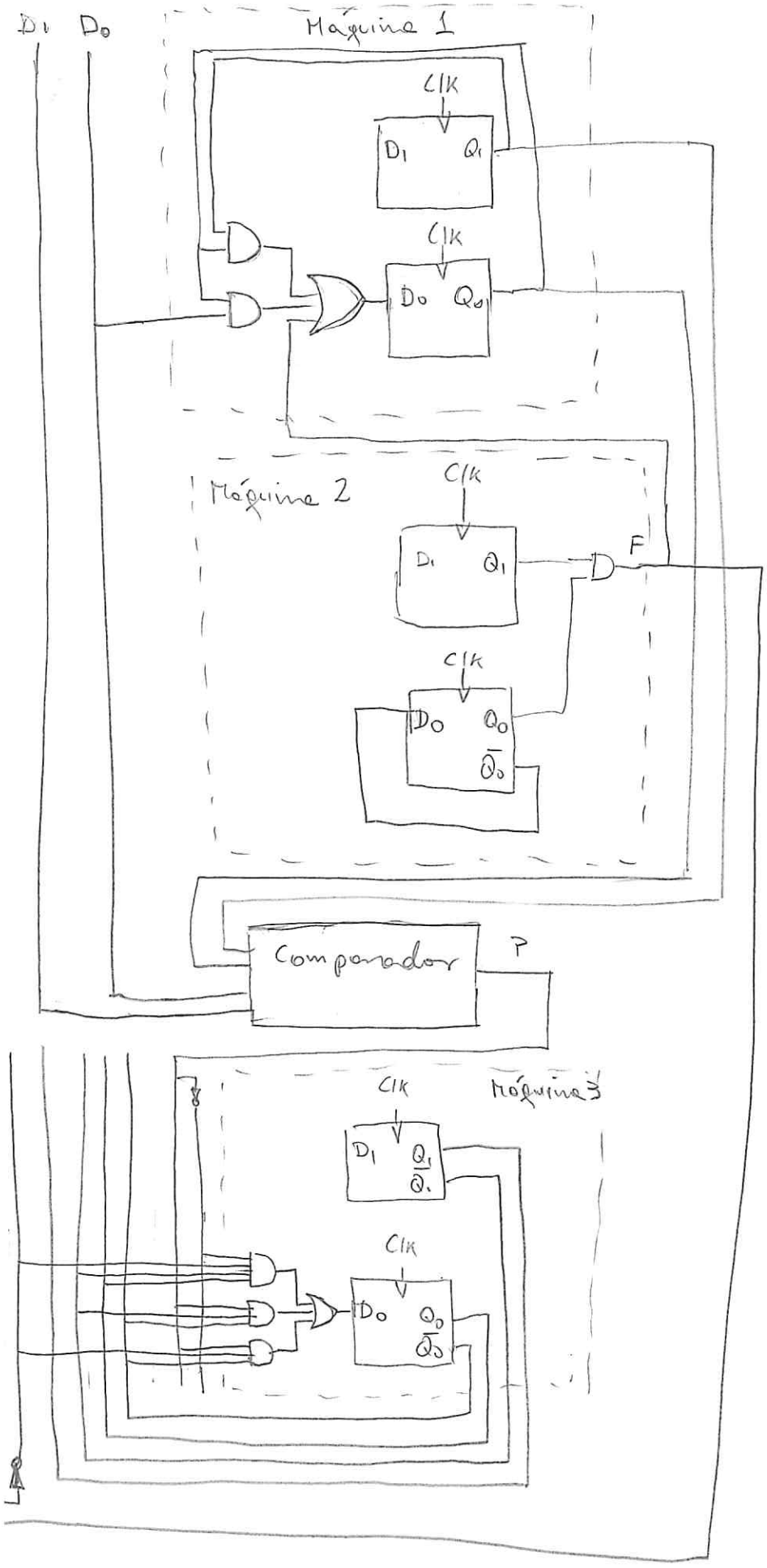


PF	$Q_1 Q_0$	$Q_1^{t+1} Q_0^{t+1}$
0X	0 0	0 0
1X	0 0	0 1
0 0	0 1	0 1
1 0	0 1	1 0
X 1	0 1	0 0
0 0	1 0	1 0
1 0	1 0	1 1
X 1	1 0	0 0
X X	1 1	0 0

PF \ $Q_1 Q_0$	00	01	11	10
00	0	1	0	0
01	0	0	0	0
11	1	0	0	0
10	1	0	0	1

Bit menos significativo

$$Q_0^{t+1} = \overline{P}\,\overline{F}\,\overline{Q_1}\,Q_0 + P\,\overline{Q_1}\,\overline{Q_0} + P\,\overline{F}\,\overline{Q_0}$$

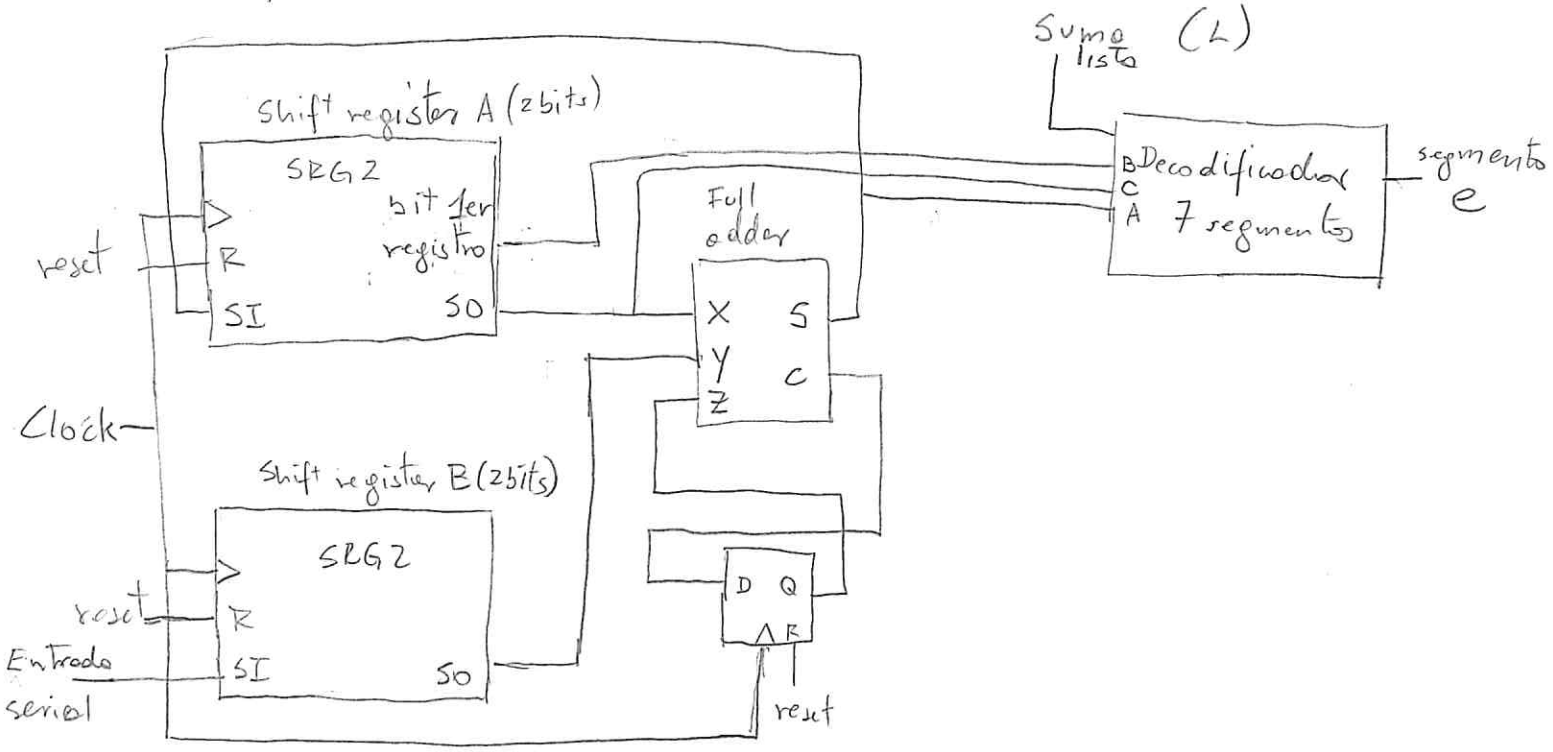


Contador de repeticiones (1,5pts): 0.5 por diagramas de estado, 1pt por tablas, kmaps y funciones. Idem para contador de turnos y memoria

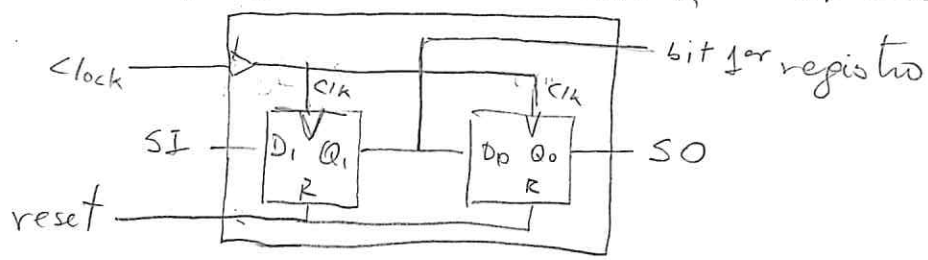
Circuiteria (1.5pts): 0,5 por diagrama de interconexiones (considerar comparador), 1pt por los 3 circuitos de las máquinas anteriormente mencionadas

Pregunta 2

Se utilizará el sumador serie visto en clase



Shift registar A: Tenemos que tener acceso a los 2 registros



Decodificador 7 segmentos

Entradas: suma lista (L)
ABC (suma binaria)

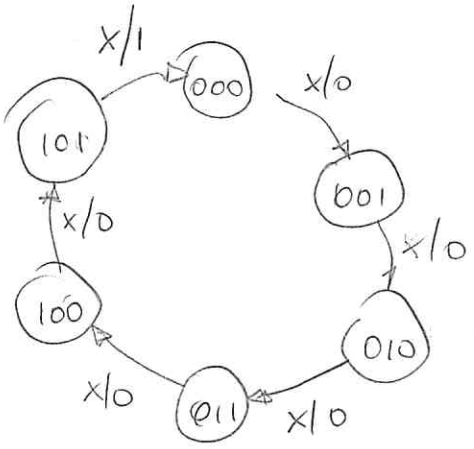
Salida: e

L	A	B	C	e
0	X	X	X	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1

LA \ BC	00	01	11	10
00	0	0	0	0
01	0	0	X	0
11	0	0	X	1
10	1	0	0	1

$e = L B \bar{C} + L \bar{A} \bar{C}$

Suma lista (L): es un contador que se prende cuando la suma está lista para desplegarse. Esto ocurre al sexto flanco de subida



$Q_2 Q_1 Q_0$	Q_2^{t+1}	Q_1^{t+1}	Q_0^{t+1}	L
0 0 0	0	0	1	0
0 0 1	0	1	0	0
0 1 0	0	1	1	0
0 1 1	1	0	0	0
1 0 0	1	0	1	0
1 0 1	0	0	0	1

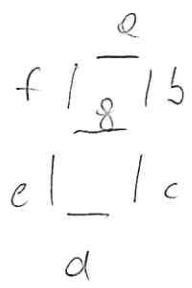
$L = Q_2 Q_0$

L también se tiene que conectar al reset de los registros

Diagrama general 1.5puntos
receptor o equivalente 1.5puntos
sumador o equivalente 1puntos
7-segmentos 1punto
suma lista 1punto

Pregunta 3

A B C D	resultado resto	a b c d e f g
0 0 0 0	0	1 1 1 1 1 1 0
0 0 0 1	1	0 1 1 0 0 0 0
0 0 1 0	2	1 1 0 1 1 0 1
0 0 1 1	3	1 1 1 1 0 0 1
0 1 0 0	1	0 1 1 0 0 0 0
0 1 0 1	0	1 1 1 1 1 1 0
0 1 1 0	1	0 1 1 0 0 0 0
0 1 1 1	2	1 1 0 1 1 0 1
1 0 0 0	2	1 1 0 1 1 0 1
1 0 0 1	1	0 1 1 0 0 0 0
1 0 1 0	0	1 1 1 1 1 1 0
1 0 1 1	1	0 1 1 0 0 0 0
1 1 0 0	3	1 1 1 1 0 0 1
1 1 0 1	2	1 1 0 1 1 0 1
1 1 1 0	1	0 1 1 0 0 0 0
1 1 1 1	0	1 1 1 1 1 1 0




```
module resta (input logic A,B,C,D,
              output logic a,b,c,d,e,f,g);

assign a = ~A&~B&~C&~D |
           ~A&~B& C&~D |
           ~A&~B& C& D |
           ~A& B&~C&~D |
           ~A& B& C& D |
            A&~B&~C&~D |
            A&~B& C&~D |
            A& B&~C&~D |
            A& B&~C& D |
            A& B& C& D;

assign b = 1;
```

2 puntos por sintaxis correcta
1 punto por definir las entradas y salidas correctas
2 punto por restar
2 puntos por implementar display

```
assign c = ~A&~B&~C&~D |
           ~A&~B&~C& D |
           ~A&~B& C& D |
           ~A& B&~C&~D |
           ~A& B&~C& D |
           ~A& B& C&~D |
            A&~B&~C& D |
            A&~B& C&~D |
            A&~B& C& D |
            A& B&~C&~D |
            A& B&~C& D |
            A& B& C&~D |
            A& B& C& D ;

assign d = ~A&~B&~C&~D |
           ~A&~B& C&~D |
           ~A&~B& C& D |
           ~A& B&~C&~D |
           ~A& B& C& D |
            A&~B&~C&~D |
            A&~B& C&~D |
            A&~B& C& D |
            A& B&~C&~D |
            A& B&~C& D |
            A& B& C& D ;

assign e = ~A&~B&~C&~D |
           ~A&~B&~C& D |
           ~A&~B& C&~D |
           ~A& B&~C& D |
           ~A& B& C& D |
            A&~B&~C&~D |
            A&~B&~C& D |
            A&~B& C&~D |
            A& B&~C& D |
            A& B& C& D ;
```

```
assign f = ~A & ~B & ~C & ~D |
           ~A &  B & ~C &  D |
            A & ~B &  C & ~D |
            A &  B &  C &  D ;

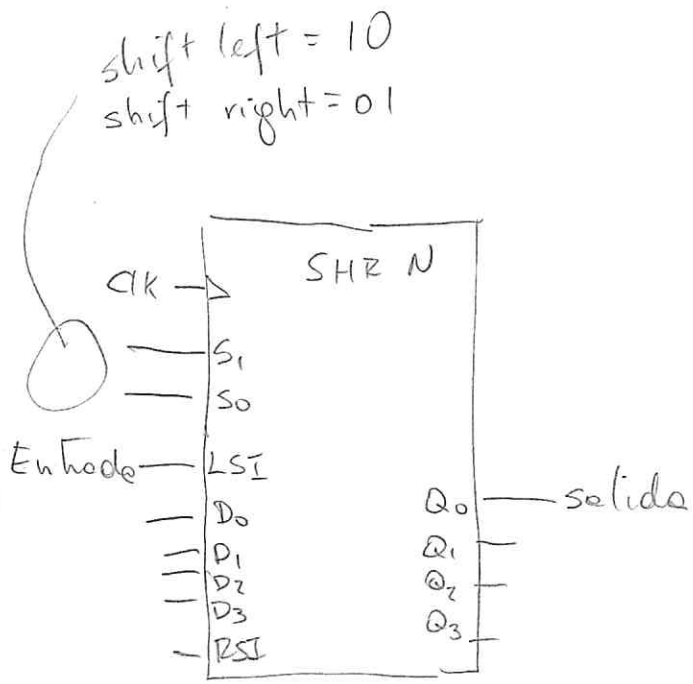
assign g = ~A & ~B &  C & ~D |
           ~A & ~B &  C &  D |
           ~A &  B &  C &  D |
            A & ~B & ~C & ~D |
            A &  B & ~C & ~D |
            A &  B & ~C &  D ;

endmodule
```

Problema 4

Este problema se puede resolver utilizando un shift registr bi-direccional visto en clase.

Para cargar en forma serial un número de N bits se necesitan N flip flops y N flancos de clock, mediante la operación shift left. Luego, necesitamos N-1 flancos de subida para mover hacia la derecha los bits guardados. Por lo tanto necesitamos N flip flops y 2N-1 flancos de subida para hacer todo el proceso.



N flip-flops: 3ptos
Distinto a N o N-1 flip-flops pero buen entendimiento: 1,5/3 ptos

2N-1 o 2N flancos: 3ptos
Distinto de 2n-1 o 2n flancos pero buen entendimiento: 1,5/3ptos