SOLUCIÓN

PROBLEMA 1

Diagrama de estados

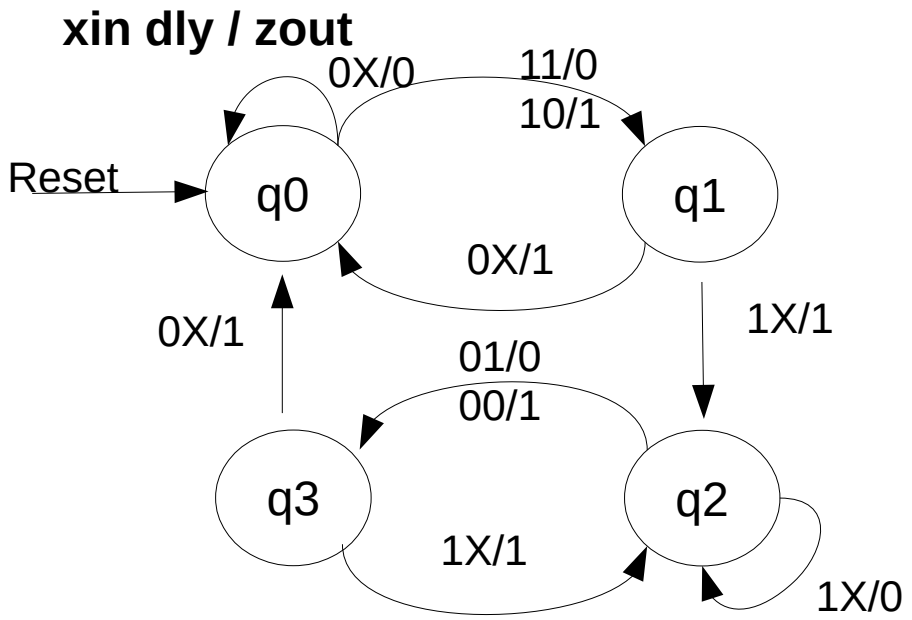


Tabla de Estados

	00	01	11	10	00	01	11	10
q0	q0	q0	q1	q1	0	0	0	1
q1	q0	q0	q2	q2	1	1	1	1
q2	q3	q3	q2	q2	1	0	0	0
q3	q0	q0	q2	q2	1	1	1	1

Minimización de Estados

q2	✘		
q1	✓	✘	
q0	✘	✘	✘
	q3	**q2**	**q1**

q1 y q3 son equivalentes.

Codificación de Estados

		00	01	11	10	00	01	11	10
q0	**00**	00	00	01	01	0	0	0	1
q1~q3	**01**	00	00	11	11	1	1	1	1
q2	**11**	01	01	11	11	1	0	0	0
q3	**10**	--	--	--	--	-	-	-	-

SOLUCIÓN

Minimización – Mapas K

Q1Q0\xin dly	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	0	0	1	1
10	-	-	-	-

D1 = Q0.xin

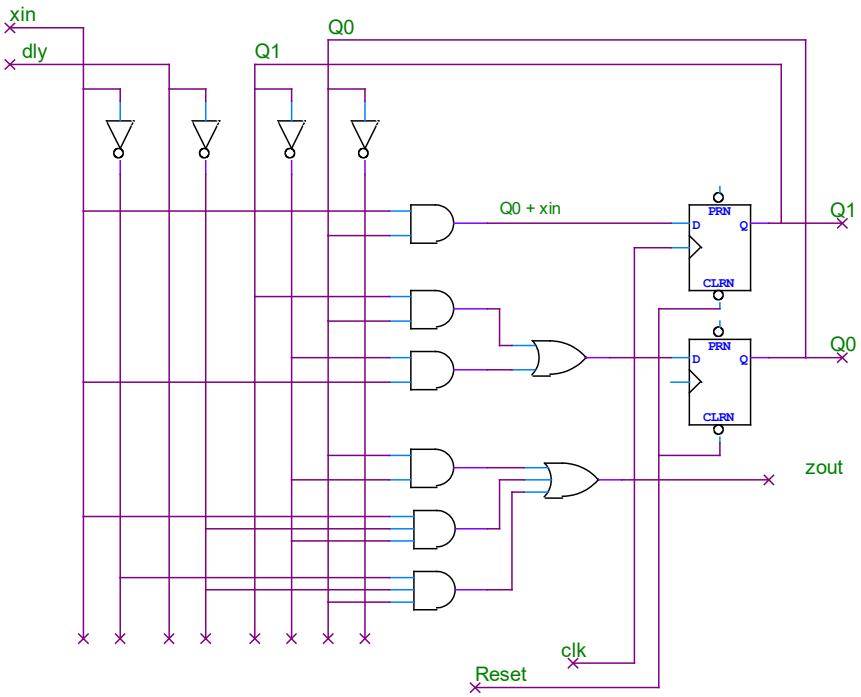
Q1Q0\xin dly	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	1	1	1	1
10	-	-	-	-

D0 = Q1.Q0 + !Q1.xin

Q1Q0\xin dly	00	01	11	10
00	0	0	0	1
01	1	1	1	1
11	1	0	0	0
10	-	-	-	-

zout = !Q1.Q0 + !Q1.xin.!dly + Q0.!xin.!dly

Circuito



SOLUCIÓN

Problema 2

SECUENCIA

```
MODULE:     Examen
INPUT:      Nivel[4], Stb0, Act[4], Stb1
OUTPUT:     Bot_Tot[8], Bot_MIN[8], Res
MEMORY:     N_Min[4], N_aux[4], Bot_Total[8], Bot_menor[8]


   0. Bot_Total[] ← 0000 0000
      Bot_menor[] ← 0000 0000

   1. Bot_Total[] * Stb0 ← INC(Bot_Total[])
      Bot_menor[] * Stb0 . MENOR(Nivel[],N_Min[]) ← INC(Bot_menor[])
      N_Min[] * Stb1 . /Stb0 ← Act[]
      → (Stb0, /Stb0) / (2,1)

   2. Bot_Total[] * Stb0 . /Fin ← INC(Bot_Total[])
      Bot_menor[] * Stb0 . /Fin. MENOR(Nivel,N_Min) ← INC(Bot_menor[])
      → (Fin,/Fin) / (3,2)

   3. Res =  1
      N_Min[] ← N_aux[] . /Stb1 + Act[] . Stb1
      Bot_Total[] ← 0000 0000
      Bot_menor[] ← 0000 0000
      → (1)

END SEQUENCE
CONTROLRESET (0)
      Fin[] = Stb0 . (IGUAL(Nivel[], 0xF))
      N_aux * Stb1 ← Act[]
      Bot_Tot[] = Bot_Total[]
      Bot_MIN[] = Bot_menor[]
END
```

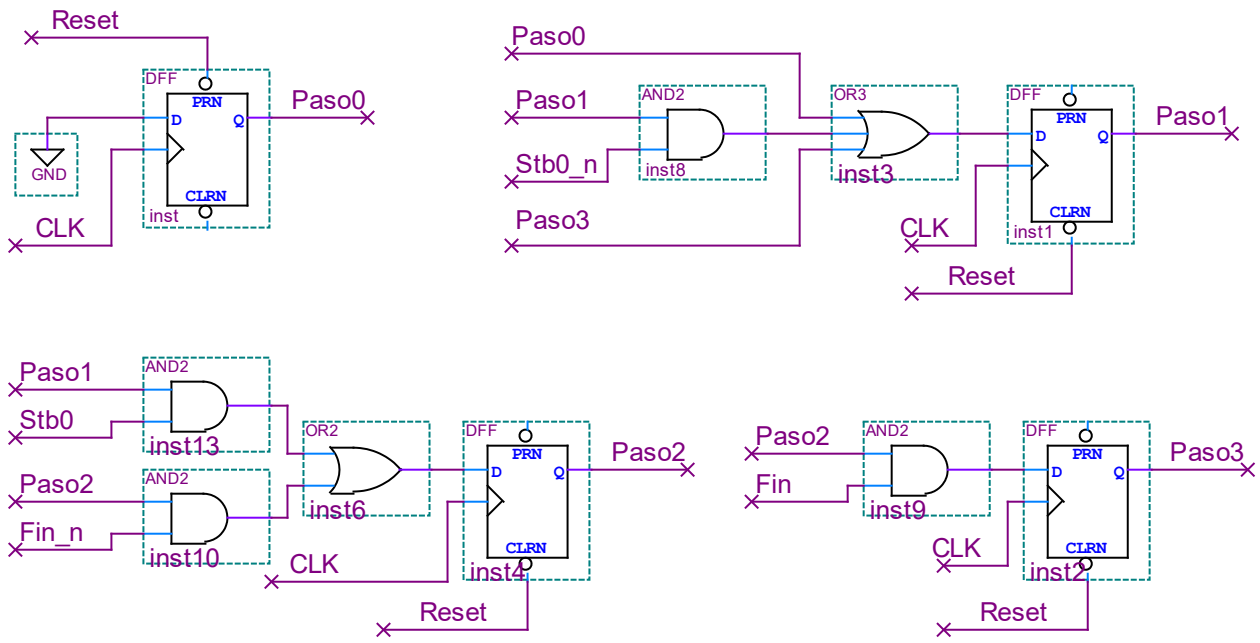
En el paso 1 se espera que comience un nuevo lote y mientras se puede actualizar el N_Min hasta que Stb0=1.
En el paso 2 se utiliza el último N_Min que llegó, pero si llega un nuevo N_Min, se lo guarda en N_aux[] para el siguiente lote.
En el paso 3 se actualiza el N_Min con el último valor ingresado, teniendo en cuenta que puede venir un nuevo valor en ese mismo instante.

FACULTAD DE INGENIERIA
UNIVERSIDAD DE LA REPUBLICA

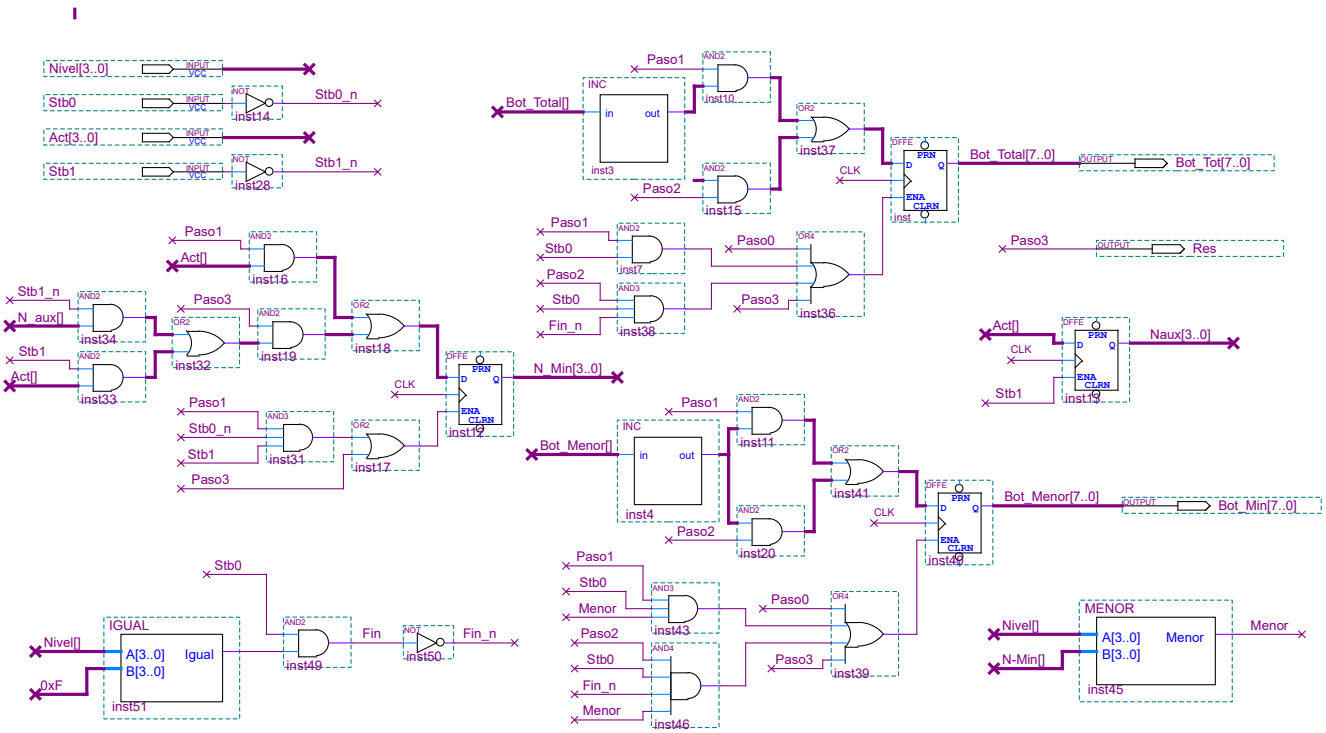
DISEÑO LÓGICO
JULIO 2019

SOLUCIÓN

BLOQUE DE CONTROL



BLOQUE DE DATOS



SOLUCIÓN

Ejercicio 1

a)

T - tp0 >= tsu1 T1 = tsu1 + tp0max

T - tp1 >= tsu2 T2 = tsu2 + tp1max

T - tp0 – txor - tor >= tsu0 T3 = tsu0 + tp0max + txormax + tormax

T - tp1 – txor - tor >= tsu0 T4 = tsu0 + tp1max + txormax + tormax

fmax = 1/max{T1, T2, T3, T4}

b) X debe permanecer constante desde (tsu0 + tormax) antes del flanco del reloj, hasta (tormin + th0) después del flanco de reloj

Ejercicio 2

```
A = 63 -> representación binaria del módulo: 11 1111 6 bits
B = -11 -> representación binaria del módulo: 1011 4 bits
C = -1 -> representación binaria: 1 1 bit
Alcanza con 7 bits

Representaciones en código complemento a 2:
A = 011 1111
B = 111 0101 (000 1011 -> 111 0100 + 1 = 111 0101)
C = 111 1111

-A = 100 0001 (100 0000 + 1 = 100 0001)

X = B - A = 111 0101 + 100 0001 = 011 0110
Resultado NO válido, hay cambio de signo del resultado: sumando dos negativos  da
resultado positivo (o también se puede ver que solo hay Carry out)

Z = C – A = 111 1111 + 100 0001 = 100 0000
Resultado válido, sumando 2 negativos da resultado negativo (o también que hay
Cyin y Cyout). El resultado es -64, justo el caso límite que se puede representar
en código de complemento a 2 de 7 bits.
```