Pregunta 1

101 100 → -20
010 100 → 20

a) 15 14 13 12 11 10 9
1 1 0 0 0 0 1 | 101 | 011 | 100

if (R3 < 0) PC ← PC − 20
else PC ← PC + 1

			MB	FS	MD	RW	MW	PL	JB	BC
101	011	100	1	0000	0	0	0	1	0	1

b) 15 14 13 12 11 10 9
0 1 0 0 0 0 0 | 000 | 011 | 010

M[R3] ← R2

			MB	FS	MD	RW	MW	PL	JB	BC
000	011	010	0	0000	0	0	1	0	0	0

c) 15 14 13 12 11 10 9
1 0 0 0 0 1 0 | 010 | 111 | 101

R2 ← R7 + 6

			MB	FS	MD	RW	MW	PL	JB	BC
010	111	101	1	0010	0	1	0	0	0	0

d) 15 14 13 12 11 10 9
1 1 1 0 0 0 0 | 111 | 000 | 010

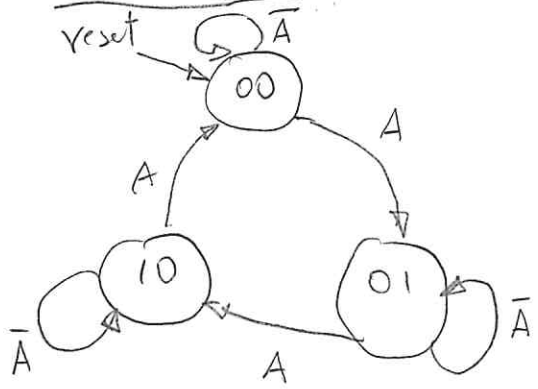
PC ← R0

			MB	FS	MD	RW	MW	PL	JB	BC
111	000	010	1	0000	1	0	0	1	1	0

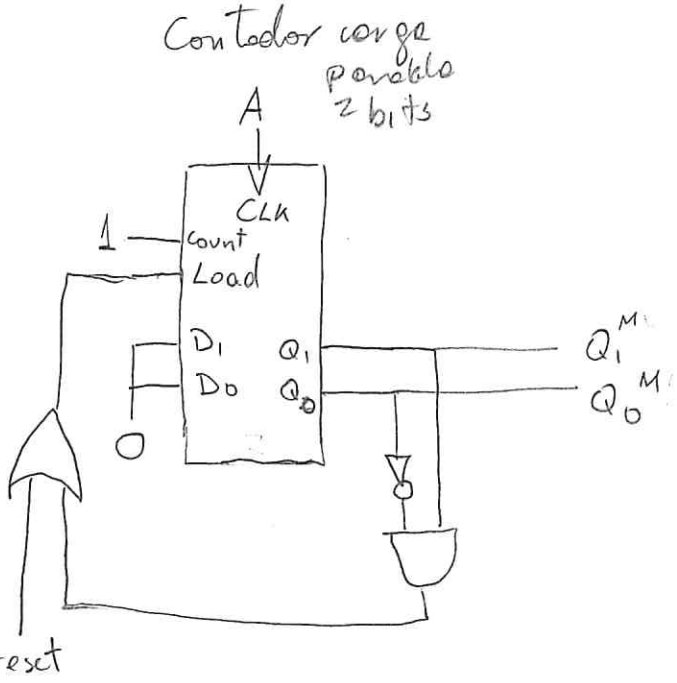
0.75 puntos por cada micro operación
0.75 puntos por cada palabra de control a la salida del decodificador

Pregunta 2

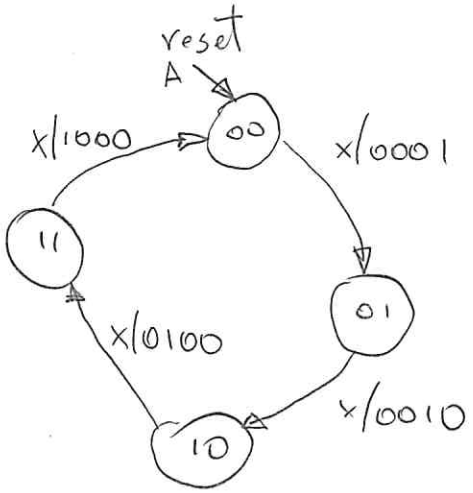
Selector de modo



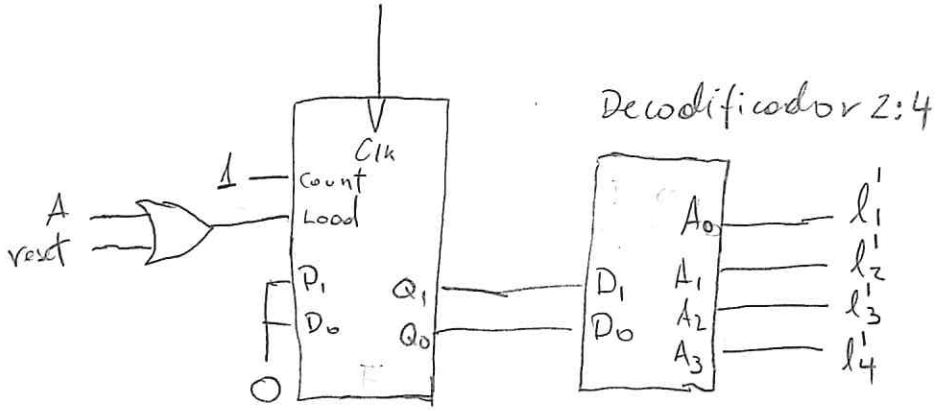
Contador carga paralelo 2 bits



Luces modo 1

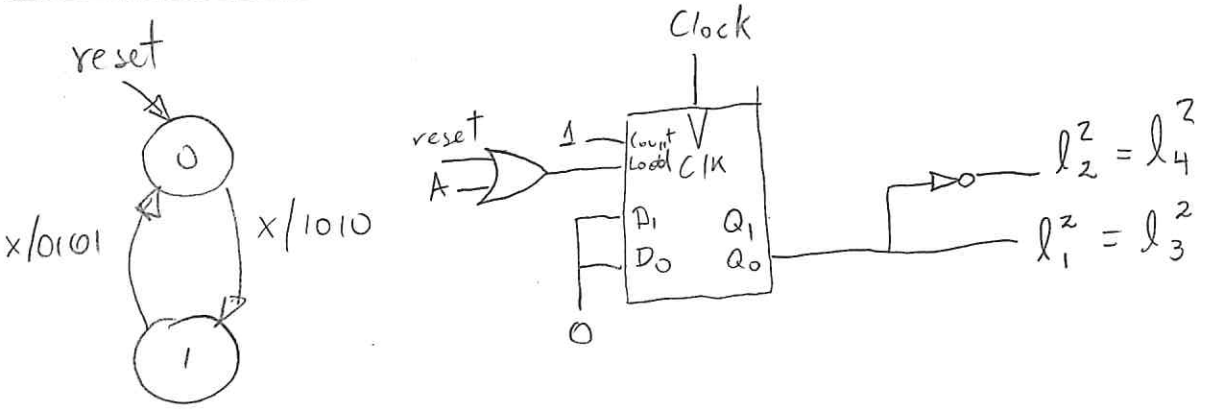


Contador carga paralelo 2 bits clock

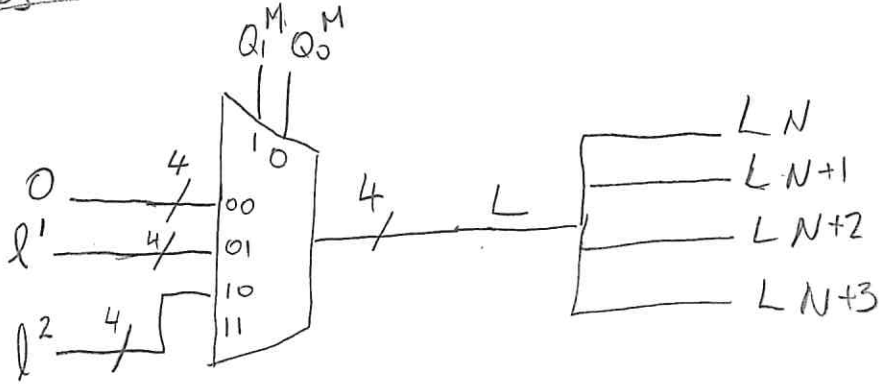


Luces modo 2

contador carga paralela 2 bits



Luces de salida



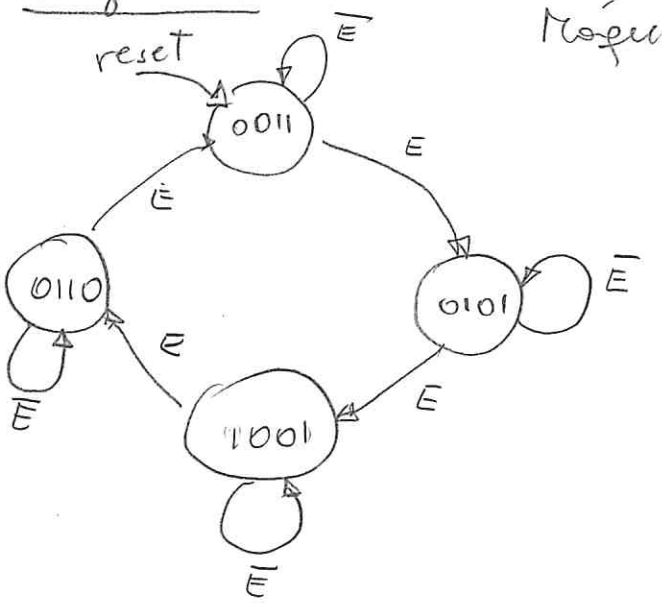
2 pts Descripción general de las maquinas de estado usadas

1 pto circuito selector de modo

2 pts circuitos de modos de las luces

1 pto circuito o gestión de la salida del sistema

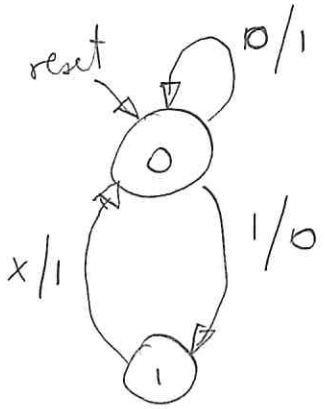
Pregunta 3



Máquina 1: contador

Entradas: Enable (E)

Estados = Salidas = cuenta



Máquina 2: divisor de frecuencia

Entradas: selector (s)

Salidas: Enable (E)

Pauta P3

Contador

```
module maquina1 (input clk,
                 input reset,
                 input E,
                 output [3:0] C);

reg [4:0] state, nextstate;

parameter S3 = 4'b0011;
parameter S5 = 4'b0101;
parameter S9 = 4'b1001;
parameter S6 = 4'b0110;

always @ (posedge clk, posedge reset)
        if (reset) state <= S3;
        else       state <= nextstate;

always @ (*)
        case (state)
                S3: nextstate = S5;
                S5: nextstate = S9;
                S9: nextstate = S6;
                S6: nextstate = S3;
                default: nextstate = S3;
        endcase

assign C = state;
```

Divisor clock (2x)

```
module maquina2 (input clk,
                 input reset,
                 input S,
                 output E);

reg state, nextstate;

parameter S0 = 1'b0;
parameter S1 = 1'b1;

always @ (posedge clk, posedge reset)
        if (reset) state <= S0;
        else       state <= nextstate;

always @ (*)
        case (state)
                S0: if (S) nextstate = S1;
                    else nextstate = S0;
                S1: nextstate = S0;
        endcase

assign E = (state == 0) & (S == 0) | (state == 1);
```

-1pto por cada diagrama de estados de cada maquina
-2pto por casa código
Si no hay diagrama de estado y el código está correcto
-Cada código vale 3ptos
Se descontó 0,25 por cada pequeño error en el código
y 0,5 en caso de que el error sea más grave

Pregunta 4

```
0   LDI  0,4
1   LD   1,0     N guardado en R1
2   ADI  0,0,5
3   LD   2,0     D guardado en R2
4   LDI  3,5     linea 5 guardada en R3
5   INC  4,4     P se ve guardando en R4
6   SUB  1,1,2   R se ve guardando en R1
7   BRN  1,2
8   JMP  3
9   ADI  0,0,7
10  INC  0,0
11  ST   0,4     Guarda P en 17
12  ADI  0,0,3
13  ST   0,1     Guarda R en 20
```

Puntajes en carpeta