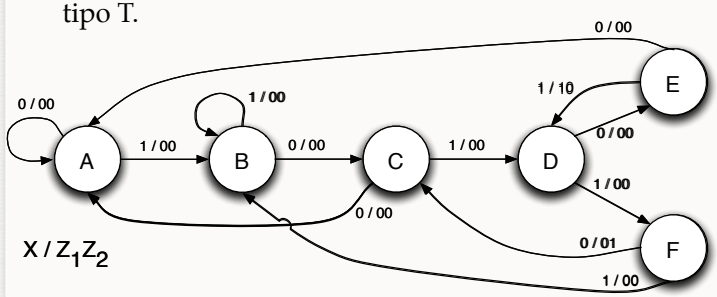
Diseño de Circuitos Secuenciales (2)

Circuitos Digitales EC1723

Universidad Simón Bolívar
Departamento de Electrónica y Circuitos
Prof. Juan. C. Regidor

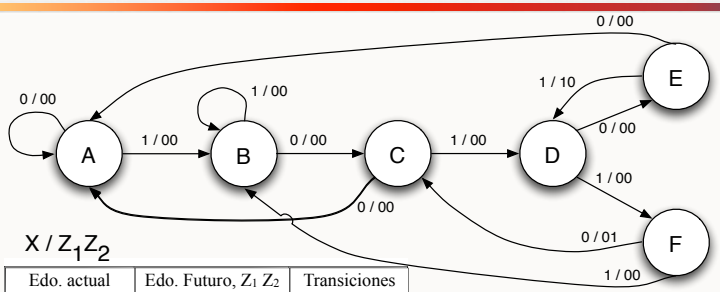
Ejercicio: Detector de secuencia (1)

- Se necesita un circuito detector de secuencias que active su salida Z_1 cuando se reciba la secuencia 10101, y su salida Z_2 cuando se reciba la secuencia 10110. Se considera que las secuencias pueden venir superpuestas. Utilizar flip-flops tipo T.



A	"...0"
B	"1"
C	"10"
D	"101"
E	"1010"
F	"1011"

Ejercicio: Detector de secuencia (2)



Edo. actual		Edo. Futuro, Z_1 Z_2		Transiciones	
	Codificado	X = 0	X = 1	X = 0	X = 1
A	000	000, 00	001, 00	000	00α
B	001	010, 00	001, 00	0αβ	001
C	010	000, 00	011, 00	0β0	01α
D	011	100, 00	101, 00	αββ	αβ1
E	100	000, 00	011, 10	β00	βαα
F	101	010, 01	001, 00	βαβ	β01

Ejercicio: Detector de secuencia (3)

Edo. actual		Edo. Futuro, Z_1 Z_2		Transiciones	
	Codificado	X = 0	X = 1	X = 0	X = 1
A	000	000, 00	001, 00	000	00α
B	001	010, 00	001, 00	0αβ	001
C	010	000, 00	011, 00	0β0	01α
D	011	100, 00	101, 00	αββ	αβ1
E	100	000, 00	011, 10	β00	βαα
F	101	010, 01	001, 00	βαβ	β01

Q1 Q0 \ X Q2	00	01	11	10
00	0	β	β	0
01	0	β	β	0
11	α	x	x	α
10	0	x	x	0

T2 = Q1·Q0 + Q2

Q1 Q0 \ X Q2	00	01	11	10
00	0	0	α	0
01	α	α	0	0
11	β	x	x	β
10	β	x	x	1

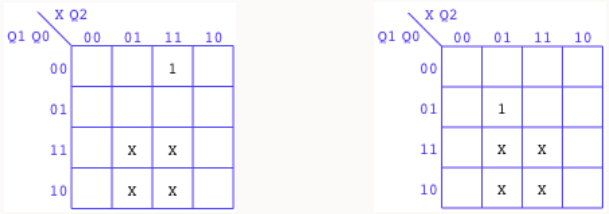
T1 = X'·Q0 + X·Q2·Q0' + X'·Q1 + Q1·Q0

Q1 Q0 \ X Q2	00	01	11	10
00	0	0	α	α
01	β	β	1	1
11	β	x	x	1
10	0	x	x	α

T0 = X·Q0' + X'·Q0 = X⊕Q0

Ejercicio: Detector de secuencia (4)

	Edo. actual	Edo. Futuro, Z_1 Z_2		Transiciones	
	Codificado	X = 0	X = 1	X = 0	X = 1
A	000	000, 00	001, 00	000	00α
B	001	010, 00	001, 00	0αβ	001
C	010	000, 00	011, 00	0β0	01α
D	011	100, 00	101, 00	αββ	αβ1
E	100	000, 00	011, 10	β00	βαα
F	101	010, 01	001, 00	βαβ	β01

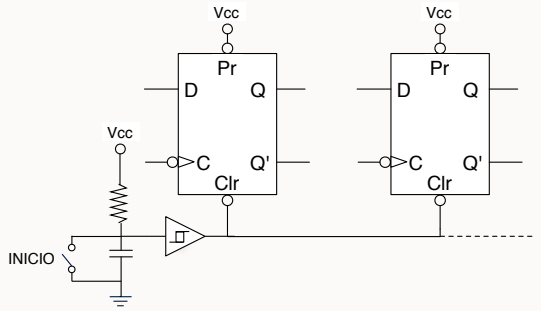


$Z_1 = X \cdot Q2 \cdot Q0'$

$Z_2 = X' \cdot Q2 \cdot Q0$

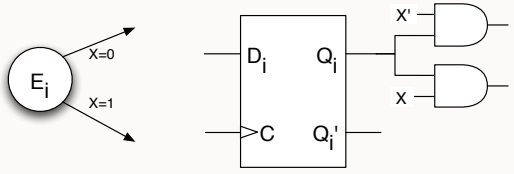
Inicio

- El estado inicial suele asignarse de tal modo que todos los flip-flops estén en cero o en uno, y se usan las entradas de *Clear* o *Preset* para iniciar el circuito.

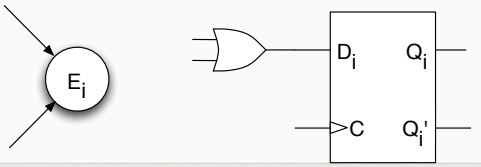


Diseño con un flip-flop por estado (*one-hot*)

- La idea es usar un flip-flop tipo D para representar cada estado de la máquina. El flip-flop que esté en uno corresponde al estado actual.
- Bifurcación:

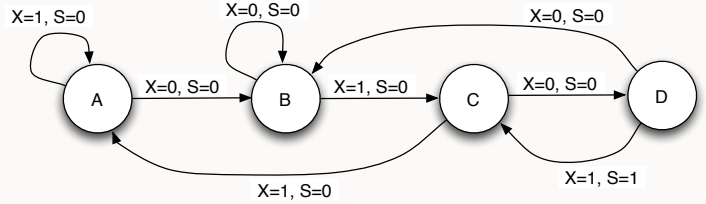


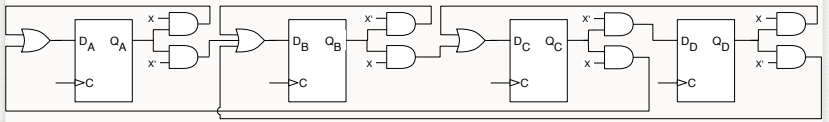
- Confluencia:



Diseño con un flip-flop por estado

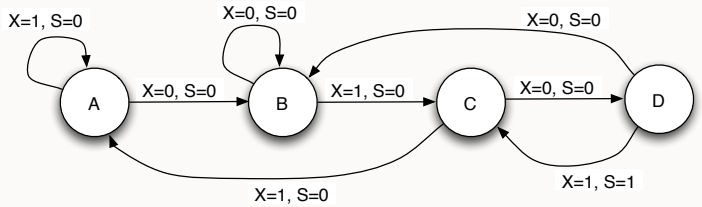
- El diseño del circuito es una copia directa del diagrama de estados:





Diseño con un flip-flop por estado

- Las ecuaciones de excitación de cada flip-flop pueden leerse de las "flechas" que llegan a cada estado:



- $D_A = X \cdot Q_A + X \cdot Q_C$ $D_B = X' \cdot Q_A + X' \cdot Q_B + X' \cdot Q_D$
- $D_C = X \cdot Q_B + X \cdot Q_D$ $D_D = X' \cdot Q_C$ $S = X \cdot Q_D$

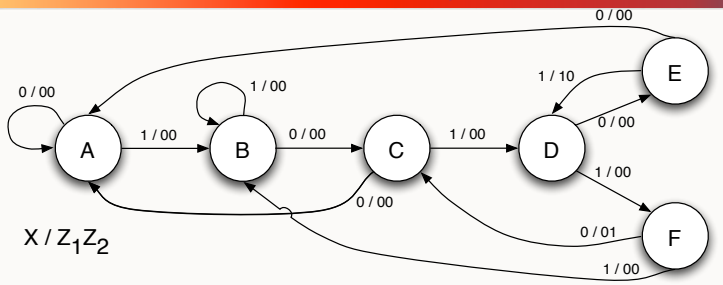
Diseño con un flip-flop por estado: Inicio

- El flip-flop que represente al estado inicial debe cargarse con un "uno" y todos los demás deben ponerse en "cero".
- La inicialización puede hacerse asíncronamente mediante las entradas de preset y clear, o de manera síncrona con compuertas adicionales a la entrada de los flip-flops.

Diseño con un flip-flop por estado

- Ventajas:
  - Simplicidad y rapidez del diseño.
  - Es más fácil depurar el circuito.
  - Especialmente útil cuando hay muchas entradas que no están activas todo el tiempo.
- Desventajas:
  - Cantidad excesiva de flip-flops.

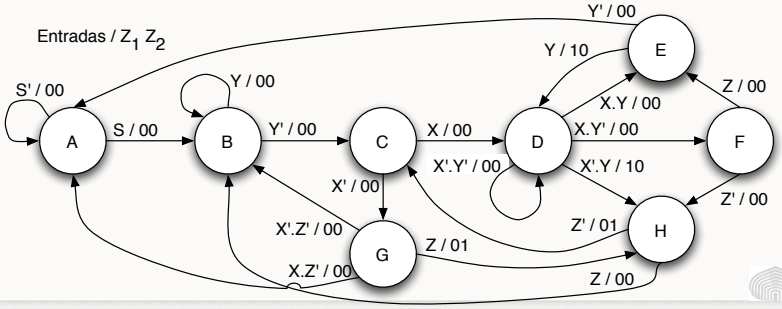
Diseño con un flip-flop por estado: Ejemplo



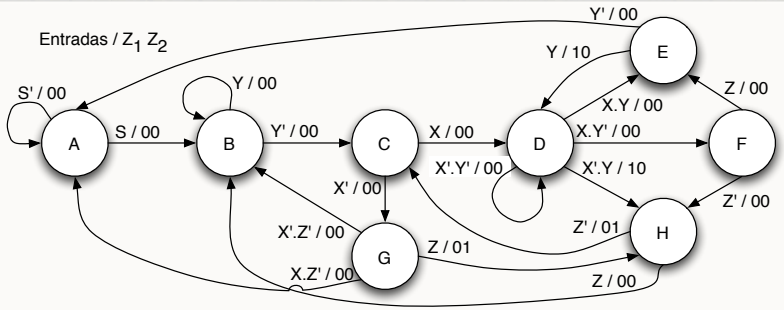
- $D_A = X' \cdot Q_A + X' \cdot Q_C + X' \cdot Q_E$
- $D_B = X \cdot Q_A + X \cdot Q_B + X \cdot Q_F$
- $D_C = X' \cdot Q_B + X' \cdot Q_F$
- $D_D = X \cdot Q_C + X \cdot Q_E$
- $D_E = X' \cdot Q_D$
- $D_F = X \cdot Q_D$
- $Z_1 = X \cdot Q_E$
- $Z_2 = X' \cdot Q_F$

Ejercicio con un flip-flop por estado (1)

- Implementar el diagrama de estados de la figura mediante el método de un flip-flop por estado. Escribir las ecuaciones de entrada de los flip-flops tipo D y las expresiones para las salidas Z_1 y Z_2.



Ejercicio con un flip-flop por estado (2)



$D_A = Q_G \cdot X \cdot Z' + Q_E \cdot Y' + Q_A \cdot S'$

$D_B = Q_H \cdot Z + Q_G \cdot X' \cdot Z' + Q_B \cdot Y + Q_A \cdot S$

$D_C = Q_H \cdot Z' + Q_B \cdot Y'$

$D_D = Q_E \cdot Y + Q_D \cdot X' \cdot Y' + Q_C \cdot X$

$D_E = Q_F \cdot Z + Q_D \cdot X \cdot Y$

$D_F = Q_D \cdot X \cdot Y'$

$D_G = Q_C \cdot X'$

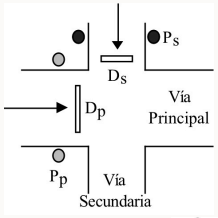
$D_H = Q_G \cdot Z + Q_F \cdot Z' + Q_D \cdot X' \cdot Y$

$Z_1 = Q_E \cdot Y + Q_D \cdot X' \cdot Y$

$Z_2 = Q_H \cdot Z' + Q_G \cdot Z$

Control de Semáforo (1)

- La figura muestra el esquema de una intersección de dos calles, una principal y otra secundaria. Hay dos detectores de vehículos, Dp y Ds, los cuales indican la presencia de un automóvil esperando en la vía principal o en la secundaria, respectivamente.
- Hay también pulsadores que pueden ser operados por un peatón que desee cruzar la calle principal (los pulsadores Pp) o la secundaria (los Ps); los pulsadores correspondientes se conectan a compuertas OR, de modo que cada grupo se puede tratar como una señal única.



Control de Semáforo (2)

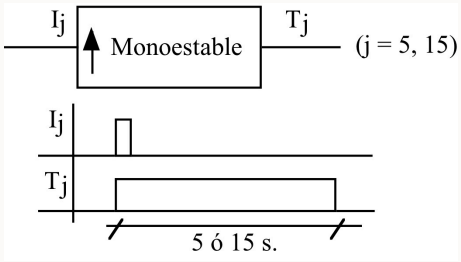
- Especificaciones:
  - La luz verde principal Vp se debe mantener encendida por un mínimo de 30 segundos y continuar encendida hasta que el detector Ds señale la presencia de un automóvil en la vía secundaria o hasta que un peatón accione el pulsador Pp. Se enciende simultáneamente la luz roja secundaria, Rs.
  - La luz amarilla principal Ap se enciende durante 5 segundos, manteniéndose encendida Rs.

Control de Semáforo (3)

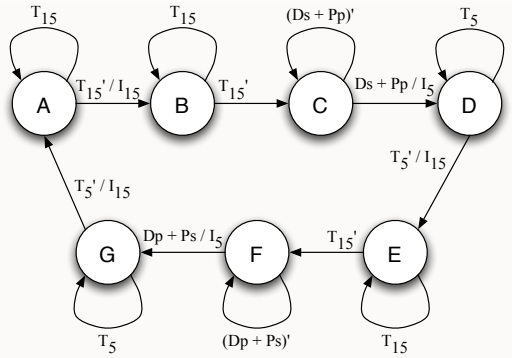
- Especificaciones (cont.):
  - La luz verde secundaria Vs se debe mantener encendida por un mínimo de 15 segundos y continuar encendida hasta que el detector Dp señale la presencia de un automóvil en la vía principal o hasta que un peatón accione el pulsador Ps. Se enciende simultáneamente la luz roja principal, Rp.
  - La luz amarilla secundaria As se enciende durante 5 segundos, manteniéndose encendida Rp.
  - Se repite el ciclo indefinidamente.

Control de Semáforo (4)

- Se dispone de dos temporizadores (monoestables no redisparables, activados por frente de subida), uno de 5 segundos (entrada I_5, salida T_5) y otro de 15 segundos (entrada I_{15}, salida T_{15})

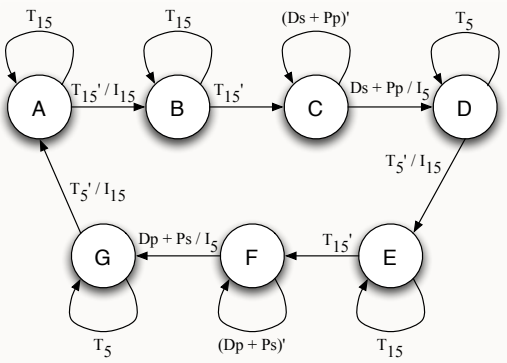


Control de Semáforo (5)



Estado	A	B	C	D	E	F	G
Salidas	Vp, Rs	Vp, Rs	Vp, Rs	Ap, Rs	Vs, Rp	Vs, Rp	As, Rp

Control de Semáforo (6)



$D_A = Q_A \cdot T_{15} + Q_G \cdot T_5'$

$D_B = Q_A \cdot T_{15}' + Q_B \cdot T_{15}$

$D_C = Q_B \cdot T_{15}' + Q_C \cdot (D_s + P_p)'$

$D_D = Q_D \cdot T_5 + Q_C \cdot (D_s + P_p)$

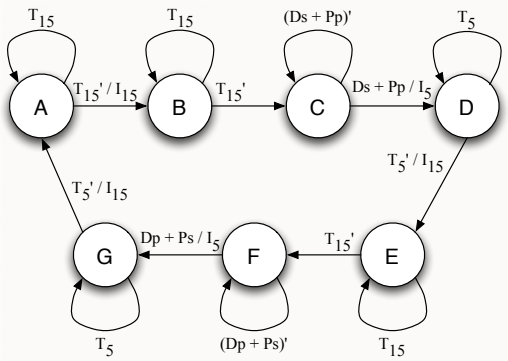
$D_E = Q_E \cdot T_{15} + Q_D \cdot T_5'$

$D_F = Q_E \cdot T_{15}' + Q_F \cdot (D_p + P_s)'$

$D_G = Q_G \cdot T_5 + Q_F \cdot (D_p + P_s)$

Estado	A	B	C	D	E	F	G
Salidas	Vp, Rs	Vp, Rs	Vp, Rs	Ap, Rs	Vs, Rp	Vs, Rp	As, Rp

Control de Semáforo (7)



Vp = QA + QB + QC

Ap = QD

Rp = QE + QF + QG

Vs = QE + QF

As = QG

Rs = QA + QB + QC + QD

I_5=QC·(Ds+Pp)+QF·(Dp+ Ps)

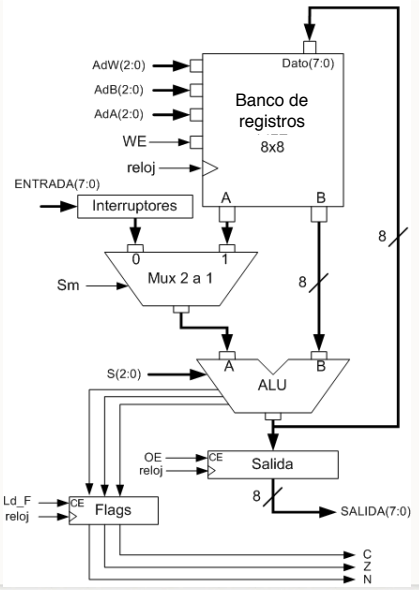
I_{15}=QA·T15'+QD·T5'+QG·T5'

Estado	A	B	C	D	E	F	G
Salidas	Vp, Rs	Vp, Rs	Vp, Rs	Ap, Rs	Vs, Rp	Vs, Rp	As, Rp

Camino de Datos (*Data Path*)

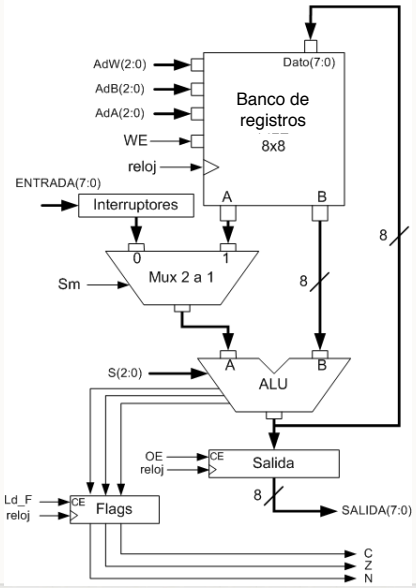
Control de la ULA

$S_2S_1S_0$	Operación
0 0 0	A + B
0 0 1	A – B
0 1 0	A + 1
0 1 1	A – 1
1 0 0	A AND B
1 0 1	A OR B
1 1 0	NOT A
1 1 1	A

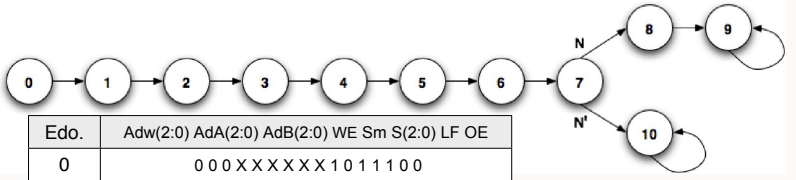


Camino de Datos (*Data Path*)

- Los registros (Banco, Salida, Flags) se cargan con el frente de subida del reloj. El período de éste debe ser lo bastante largo como para permitir que se completen todas las operaciones.



Camino de Datos



Edo.	Adw(2:0) AdA(2:0) AdB(2:0) WE Sm S(2:0) LF OE
0	000XXXXXX1011100
1	001001001110010 0
2	001001XXX1101000
3	010001XXX1101000
4	011001010110000 0
5	100011010110000 0
6	111100100110000 0
7	XXX0001111010011 0
8	110000000110000 0
9	110110XXX1101001
10	110110XXX1101101

Control de la ULA

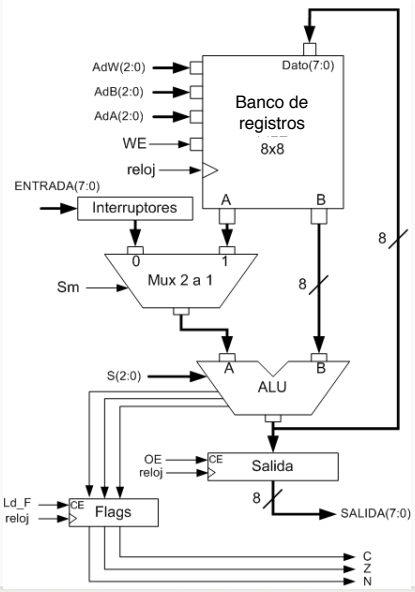
$S_2S_1S_0$	Operación
0 0 0	A + B
0 0 1	A – B
0 1 0	A + 1
0 1 1	A – 1
1 0 0	A AND B
1 0 1	A OR B
1 1 0	NOT A
1 1 1	A

Camino de Datos

- Dibujar un diagrama de estados para ejecutar la operación: Salida = Entrada*R4+R3;

Control de la ULA

$S_2S_1S_0$	Operación
0 0 0	A + B
0 0 1	A – B
0 1 0	A + 1
0 1 1	A – 1
1 0 0	A AND B
1 0 1	A OR B
1 1 0	NOT A
1 1 1	A



Camino de Datos: Ejercicio

- El siguiente algoritmo, escrito en lenguaje "C", produce la parte entera de la raíz cuadrada de un número. Hacer un diagrama de estados que lo materialice, de la manera más fiel posible, sobre el "camino de datos" mostrado antes. El "NUMERO" se supone cargado previamente en el registro R7.

Camino de Datos: Ejercicio

```
#define  NUMERO 38    // Número está cargado en el registro R7

int Sqrt( int numero )
{
   int suma, raiz, impar, DOS;

   suma  = raiz  = 0;
   impar = suma  + 1;
   DOS   = impar + 1;

wloop:
   if( suma < numero ) {
       impar += DOS;
       suma  += impar;
       raiz  ++;
       if( suma < 0 )
           return raiz ;   // Puede ser un 'Halt'
       goto wloop;
   }
   return raiz ;           // Puede ser un 'Halt'
}
```


Camino de Datos: Ejercicio

- En cierta fábrica, una banda trasportadora que corre a 1 m/s lleva dos tipos de cajas: las tipo "A" de 50 cm. de longitud, y las tipo "B", de 80 cm. Usando el camino de datos anterior, se desea diseñar un sistema que realice las siguientes operaciones:
  - Cada vez que haya pasado una caja se debe activar la salida del camino de datos, con el valor 0 si pasó una caja "A" y 1 si fue una tipo "B".
  - Se desea llevar la cuenta del número total de cajas (en R5), número de cajas tipo "A" (en R6) y número de cajas tipo "B" (en R7).

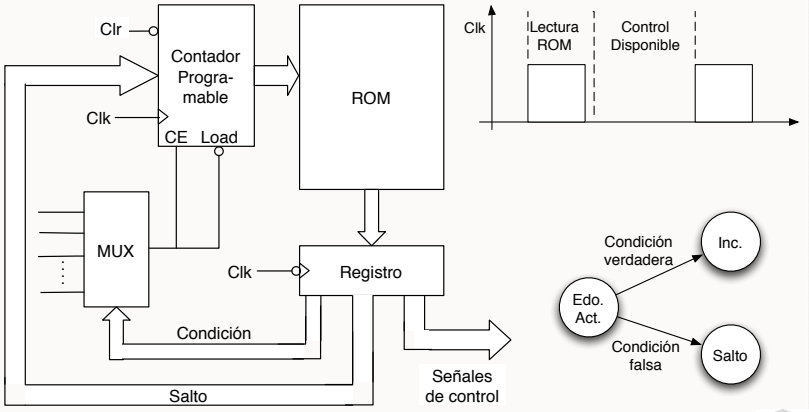
Camino de Datos: Ejercicio

- La cinta tiene un sensor óptico que detecta la interrupción de un rayo de luz al paso de una caja. La salida de este sensor se lleva a la entrada del camino de datos (bit 0). El reloj del controlador tiene un período de 0,01 s. Escriba un diagrama de estados que cumpla con las condiciones pedidas.

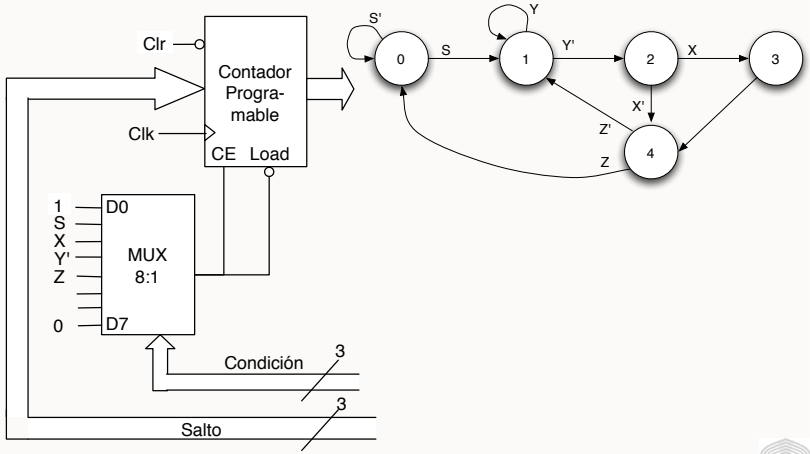
ROM

- Memoria de sólo lectura (*Read Only Memory*)
  - ROM: programable en la fabricación
  - PROM: programable una vez por el usuario
  - EPROM: borrable con luz ultravioleta
  - EEPROM o E²PROM: borrable eléctricamente; el ciclo de borrado es mucho más lento que el de escritura
  - *Flash memory*: E²PROM borrable y programable por bloques

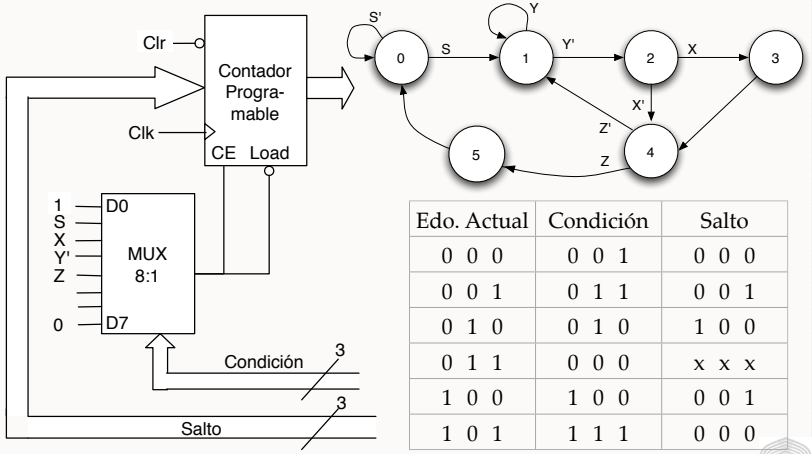
Control microprogramado



Control microprogramado 2



Control microprogramado 2



Edo. Actual	Condición	Salto
0 0 0	0 0 1	0 0 0
0 0 1	0 1 1	0 0 1
0 1 0	0 1 0	1 0 0
0 1 1	0 0 0	x x x
1 0 0	1 0 0	0 0 1
1 0 1	1 1 1	0 0 0