

INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE COMPUTO

DISEÑO DE SISTEMAS DIGITALES

Práctica 6. Detector de secuencias con modelo de mealy

**PROFESOR:**

Testa Nava Alexis

**Alumno:**

Ramírez Cotonieto Luis Fernando

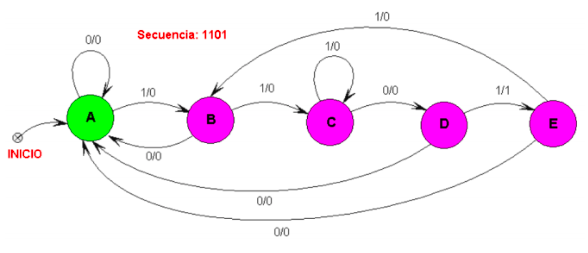
**GRUPO:**

2CV18

**FECHA:**

25/Mayo/2021

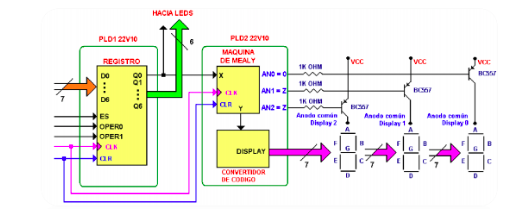
Práctica 6. Detector de secuencia con modelo de Mealy



• Aplicar toda la metodología de diseño para circuitos secuenciales y diseñar un circuito que detecte la secuencia 1 → 1→ 0 → 1.

• Observe la máquina de Mealy que se muestra en la ilustración anterior.

• Realizar el diseño por medio de FF de su preferencia que permitan implementar la FSM Mealy anterior en Falstad. Además de realizar un programa en HDL utilizando los tipos de datos enumerados (TYPE) y el diagrama a bloques que se muestra a continuación.





■ Los bits de salida (Q6, Q5, ..., Q0) del registro genérico deben colocarse en LEDs para observar el valor que contiene el registro.

■ El bloque de la máquina de Mealy y el convertidor de código. La señal de entrada X toma los bits del registro genérico a través del bit Q0.

■ El convertidor de código convierte el valor de salida 0 ó 1 de la máquina de estados en un valor de siete bits para mostrar en un display de siete segmentos de ánodo común. En el display se deben mostrar las letras A (Acierto) y E (Error). La letra A, debe mostrarse cuando se detecte la secuencia deseada y la letra E cuando no se haya detectado la secuencia.

• Verificar el correcto funcionamiento del sistema de la siguiente forma:

Cargar un número de siete bits en el registro genérico. El valor de siete bits debe contener la secuencia 1101.

Por ejemplo: 0011010, 1110110, 1011011, etc

• Colocar la operación de corrimiento a la derecha en el registro genérico. Con esta operación empezarán a introducirse los bits de forma serial a la máquina Mealy para que comience a detectar la secuencia.

• Verificar la detección de la secuencia en el display.

**Tabla de estados:**

|  |  |  |  |
| --- | --- | --- | --- |
| ENTRADA | ESTADO PRESENTE | ESTADO FUTURO | SALIDA |
| 0 | A | B | 0 |
| 1 | A | B | 0 |
| 0 | B | A | 0 |
| 1 | B | C | 0 |
| 0 | C | D | 0 |
| 1 | C | C | 0 |
| 0 | D | A | 0 |
| 1 | D | E | 1 |
| 0 | E | A | 0 |
| 1 | E | B | 0 |

**Asignación de estados:**

A= 000

B= 001

C=010

D=011

E=100

**Tablas de verdad de los Flip-Flop a utilizar**

|  |  |  |
| --- | --- | --- |
| Q | Q+1 | D |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

**Tabla de transición de estados:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ESTADO PRESENTE** | | | **ENTRADA** | **PROXIMO ESTADO** | | | **SALIDA** |
| **Q2** | **Q1** | **Q0** | **X** | **Q2+** | **Q1+** | **Q0+** | **Z** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |

Q2+ = D2 Q1+=D1 Q0+=D0

D2(Q2 Q1 Q0 X) = 7) + (12,13,14,15,10,11)

D1(Q2 Q1 Q0 X) = (3,4,5)

D0(Q2 Q1 Q0 X) = (1,4,7)

Z(Q2 Q1 Q0 X) = Q2 Q1 Q0 X

**Mapas de Karnaugh:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Q0,X |  |  |  |  |
| Q2,Q1\ | 00 | 01 | 11 | 10 |
| 00 |  |  |  |  |
| 01 |  |  | 1 |  |
| 11 | X | X | X | X |
| 10 |  |  | X | X |
| Q1,Q0 |  |  |  |  |
| X,Q2\ | 00 | 01 | 11 | 10 |
| 00 |  |  | 1 |  |
| 01 | 1 | 1 |  |  |
| 11 | X | X | X | X |
| 10 |  |  | X | X |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Q1,Q0 |  |  |  |  |
| X,Q2\ | 00 | 01 | 11 | 10 |
| 00 |  | 1 |  |  |
| 01 | 1 |  | 1 |  |
| 11 | X | X | X | X |
| 10 |  | 1 | X | X |

**D2** **D1**

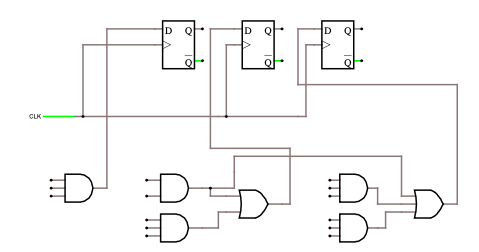
**D0**

D2(Q2 Q1 Q0 X) = Q1 Q0 X

D1(Q2 Q1 Q0 X) = Q1 Q0’ + Q1’ Q0 X

D0(Q2 Q1 Q0 X) = Q1 Q0’ X + Q1’ Q0’ X + Q1 Q0 X

**Circuito:**



<https://tinyurl.com/yzavd2zx>

**Codigo:**

library ieee;

use ieee.std\_logic\_1164.all;

entity Mealy is port (

clock, clear, x : in std\_logic;

y : out std\_logic\_vector ( 6 downto 0 )

);

attribute PIN\_NUMBERS of Mealy : entity is (

"clock:1 x:2 clear:13 "&

"y(0):14 y(1):15 y(2):16 y(3):17 y(4):18 y(5):19 y(6):20"

);

end Mealy;

architecture a\_Mealy of Mealy is

signal Qa, Qb : std\_logic;

begin

process ( clock, clear )

begin

if ( clear = '1' ) then

Qa <= '0';

Qb <= '0';

elsif ( clock'event and clock = '1' ) then

Qa <= ( Qa and ( not Qb ) ) or ( ( not Qa ) and Qb and x );

Qb <= ( not Qb ) and ( Qa xor x );

end if;

end process;

y <= "1110111" when ( ( Qa and Qb and x ) = '1' ) else

"1001111";

end architecture;