Tarea 2

Implementación de un contador aleatorio

Acoidan Martín Conrado
Sistemas Electrónicos Digitales
1º Grado Ingeniería Informática
11/04/2024

Índice

Nota: La plantilla utilizada es una guía para la realización del informe, el estudiante la debe usar un punto de partida. Se valorará todas las explicaciones para el entendimiento del proceso seguido.

Tarea 2

Implementación de un contador aleatorio

1.- Funciones asignadas

El contador aleatorio asignado se muestra en la Tabla 1.

Alumno	Secuencia	Flip-flops				Cíclico
119 Acoidan Martín Conrado	3,5,8,1,7,4,6,9,0,2	T	D	D	T	No

Tabla 1. Contador aleatorio asignado

2.- Tabla de transiciones

La tabla de transiciones del contador aleatorio asignado es:

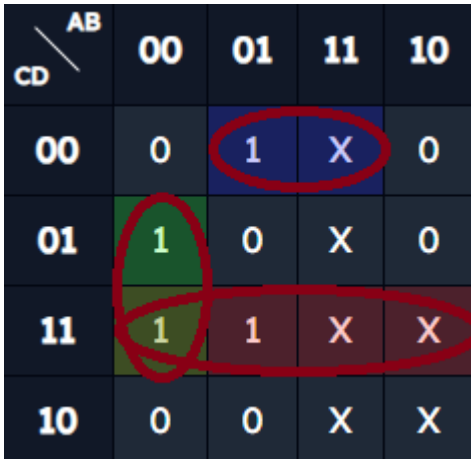
	Estado actual				Estado siguiente							
	Q_A	Q_B	Q_C	Q_D	Q_A^+	Q_B^+	Q_C^+	Q_D^+	T_A	D_B	D_C	T_D
0	0	0	0	0	0	0	1	0	0	0	1	0
1	0	0	0	1	0	1	1	1	0	1	1	0
2	0	0	1	0	0	0	1	0	0	0	1	0
3	0	0	1	1	0	1	0	1	0	1	0	0
4	0	1	0	0	0	1	1	0	0	1	1	0
5	0	1	0	1	1	0	0	0	1	0	0	1
6	0	1	1	0	1	0	0	1	1	0	0	1
7	0	1	1	1	0	1	0	0	0	1	0	1
8	1	0	0	0	0	0	0	1	1	0	0	1
9	1	0	0	1	0	0	0	0	1	0	0	1
10	1	0	1	0	X	X	X	X	X	X	X	X
11	1	0	1	1	X	X	X	X	X	X	X	X
12	1	1	0	0	X	X	X	X	X	X	X	X
13	1	1	0	1	X	X	X	X	X	X	X	X
14	1	1	1	0	X	X	X	X	X	X	X	X
15	1	1	1	1	X	X	X	X	X	X	X	X

3.- Mapas de Karnaugh

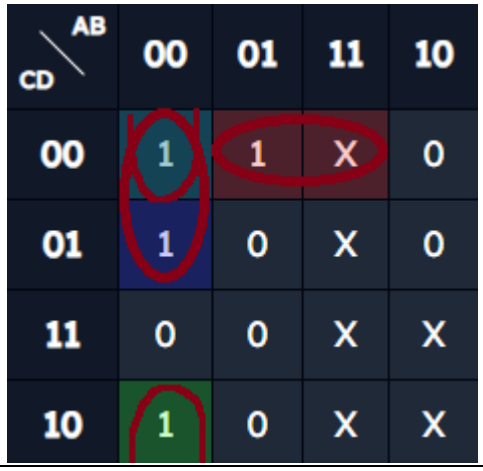
Los mapas de Karnaugh de las entradas de los biestables son:

$T_A = (Q_A + Q_B Q_C \bar{Q}_D + Q_B \bar{Q}_C Q_D)$

$D_B = (Q_C Q_D + \bar{Q}_A \bar{Q}_B Q_D + Q_B \bar{Q}_C \bar{Q}_D)$



$D_C = (Q_B \bar{Q}_C \bar{Q}_D + \bar{Q}_A \bar{Q}_B \bar{Q}_D + \bar{Q}_A \bar{Q}_B \bar{Q}_C)$



$T_D = (Q_A + Q_B Q_C + Q_B Q_D)$

4.- Circuito del contador implementado

A continuación, se muestra el circuito del contador que se ha implementado:

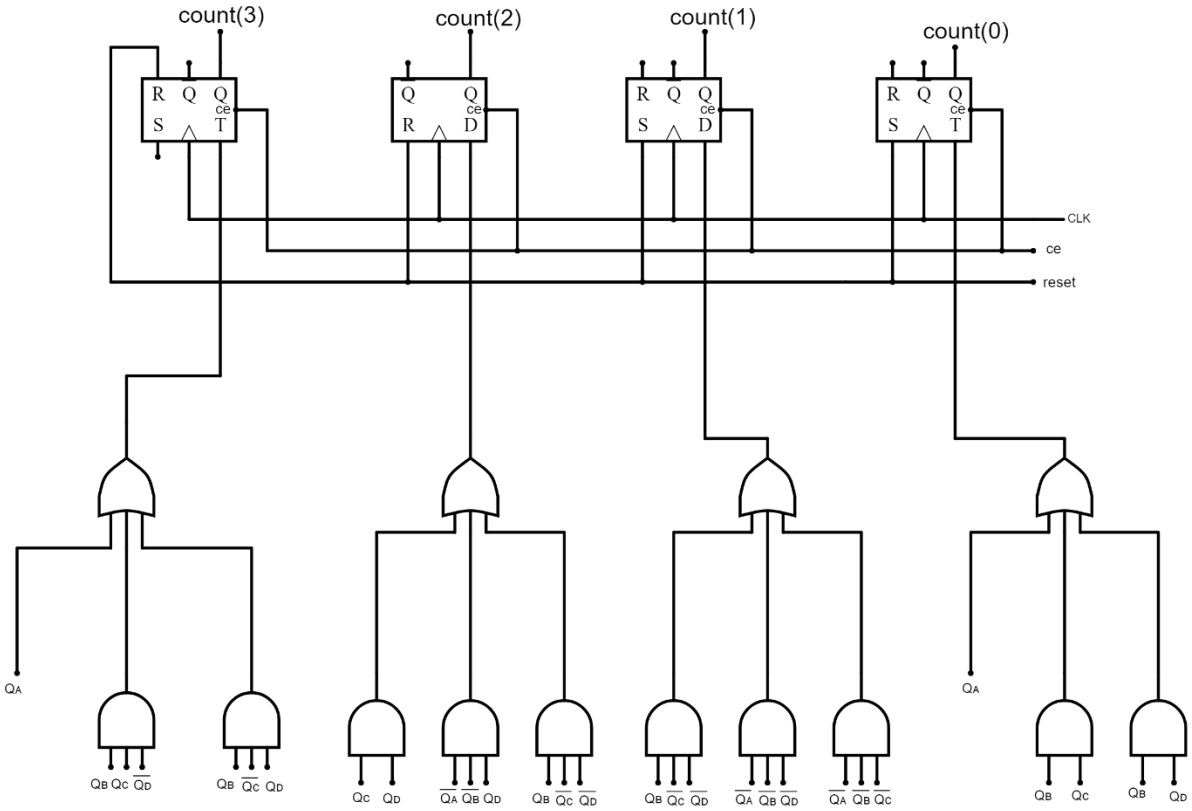


Figura 1. Circuito implementado del contador 119.

Nota: La **R** significa reset, y la **S** significa preset, esta escrito de esa manera por el software utilizado.

5.- Códigos VHDL

En este apartado se muestra el código VHDL del contador implementado:

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity contador119 is
    Port ( clk   : in std_logic;
           ce    : in std_logic;
           reset : in std_logic;
           count : out std_logic_vector (3 downto 0));
end contador119;

```

```vhdl
architecture Behavioral of contador119 is

    component ffD_reset
        port (
              clk   : in std_logic;
              ce    : in std_logic;
              reset : in std_logic;
              d     : in std_logic;
              q     : out std_logic
              );
    end component;

    component ffT_preset
        port (
              clk     : in std_logic;
              ce      : in std_logic;
              preset  : in std_logic;
              t       : in std_logic;
              q       : out std_logic
              );
    end component;

    component ffT_reset
      port (
            clk   : in std_logic;
            ce    : in std_logic;
            reset : in std_logic;
            t     : in std_logic;
            q     : out std_logic
            );
    end component;

    component ffD_preset
      port (
            clk    : in std_logic;
            ce     : in std_logic;
            preset : in std_logic;
            d      : in std_logic;
            q      : out std_logic
            );
    end component;

    signal qa, qb, qc, qd : std_logic;
    signal ta, db, dc, td : std_logic;
```

```vhdl
begin

    unitA: ffT_reset port map (
        clk   => clk,
        ce    => ce,
        reset => reset,
        t     => ta,
        q     => qa
        );

    unitB: ffD_reset port map (
        clk   => clk,
        ce    => ce,
        reset => reset,
        d     => db,
        q     => qb
        );

    unitC: ffD_preset port map (
        clk    => clk,
        ce     => ce,
        preset => reset,
        d      => dc,
        q      => qc
        );

    unitD: ffT_preset port map (
        clk    => clk,
        ce     => ce,
        preset => reset,
        t      => td,
        q      => qd
        );

    ta <= (qa) or (qb and qc and not qd) or (qb and not qc and qd);
    db <= (qc and qd) or (not qa and not qb and qd) or (qb and not qc and not qd);
    dc <= (qb and not qc and not qd) or (not qa and not qb and not qd) or (not qa and not qb and not qc);
    td <= (qa) or (qb and qc) or (qb and qd);

    count <= qa & qb & qc & qd;

end Behavioral;
```

Figura 2. Código VHDL del contador 119.

También se muestra el código VHDL del test bench.

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity sistema119_tb is
--  Port ( );
end sistema119_tb;

architecture Behavioral of sistema119_tb is

    component sistema119
        port (clk   : in std_logic;
              ce    : in std_logic;
              reset : in std_logic;
              count : out std_logic_vector (3 downto 0);
              led   : out std_logic_vector (6 downto 0));
    end component;

    signal clk   : std_logic := '0';
    signal ce    : std_logic := '0';
    signal reset : std_logic := '0';
    signal count : std_logic_vector (3 downto 0);
    signal led   : std_logic_vector (6 downto 0);

begin

    dut : sistema119
    port map (clk => clk,
              ce => ce,
              reset => reset,
              count => count,
              led => led);

    -- Clock Generation
    clk <= not clk after 10 ns;

    stimuli : process
    begin

        ce <= '0'; reset <= '1';
        wait for 20 ns;
        reset <= '0'; ce <= '1';
        wait for 180 ns;
        reset <= '1';
    end process;

end Behavioral;
```

Figura 3. Código VHDL del test bench.

6.- Cronograma de simulación

A continuación, en la Figura 3, se muestra el cronograma de simulación del sistema implementado con las salidas del contador y del decodificador de 7 segmentos.

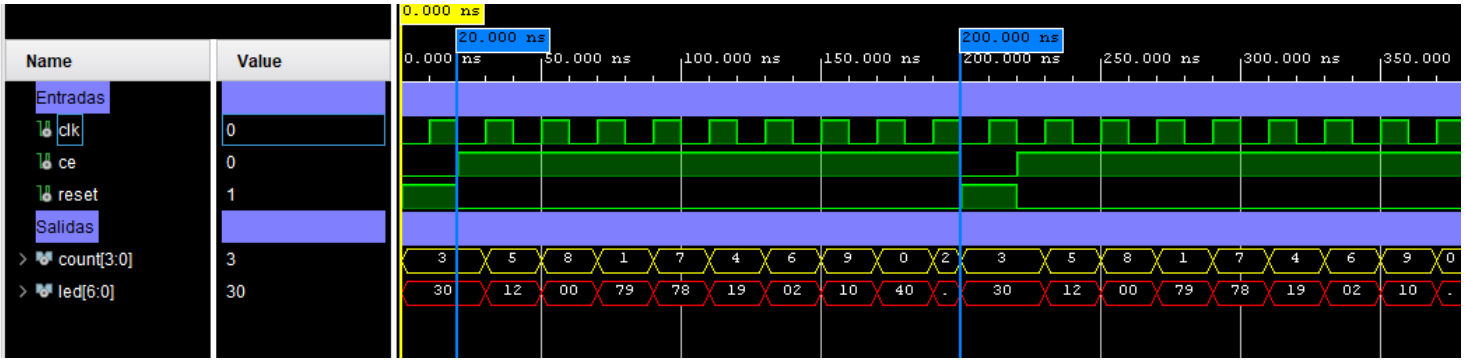


Figura 4. Cronograma de simulación del contador 119 con la secuencia completa recorrida.

Se aprecia que los resultados son consistentes con la secuencia que aparece en la tabla del apartado 1.