



Universidad Nacional Autónoma de México  
Facultad de Ingeniería  
Ingeniería en Computación



Asignatura: Diseño Digital VLSI

Grupo: 4

Practica 4: Uso De Un Teclado Matricial 4 X 4

Profesor: M.I. Alberto Navarrete Hernández

Alumnos:

- Jiménez Gutiérrez Miguel
- Martínez Ortiz Carlos Daniel

## USO DE UN TECLADO MATRICIAL 4 x 4

**Objetivo:** El alumno aprenderá a diseñar módulos con parámetros genéricos, lo que permitirá crear un proyecto con varias instancias de un mismo elemento pero con diferentes características de operación, con el fin de dar una mayor versatilidad a los módulos diseñados por el alumno.

### Introducción

Un sistema diseñado en VHDL debe simularse y probarse para la funcionalidad antes de que se convierta en hardware. En este pase de simulación, errores de diseño e incompatibilidad de los componentes utilizados en el diseño pueden ser detectados. La simulación de un diseño requiere la generación de datos de prueba y la observación de los resultados de la simulación.

### Especificaciones:

Diseñar un circuito utilizando un FPGA que se encargue de utilizar un teclado matricial 4x4, que podrá ser manejado en algún sistema de control controlar. Al presionar alguna tecla de la matriz, se deberá mostrar correctamente el valor de la tecla presionada en un display de 7 segmentos. La figura 1.1 muestra el diagrama de bloques de este sistema.

#### Diagrama de bloques:

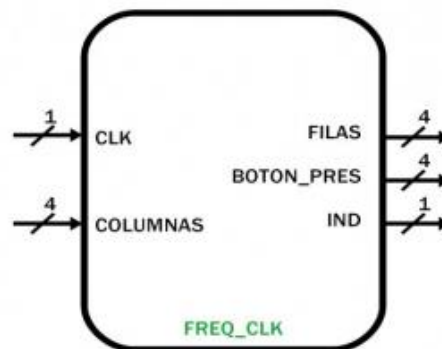


Figura 1.1 Diagrama de bloques del control de intensidad de encendido de leds.

El teclado matricial está diseñado como se muestra en la figura 1.2. Para poder detectar cuál de los botones ha sido presionado, es necesario implementar un circuito, en el cual se pueda detectar una entrada que permita ubicar la posición del botón presionado. En esta práctica lo que se realizará, es colocar las señales de las filas como salidas y las señales de las columnas como entradas. Se implementa un manejo de las filas (multiplexo) y al oprimir un botón se genera la entrada correspondiente a una columna, con lo cual se ubica la posición del botón presionado en un cierto instante, para ello las columnas se conectarán en configuración pull-down..

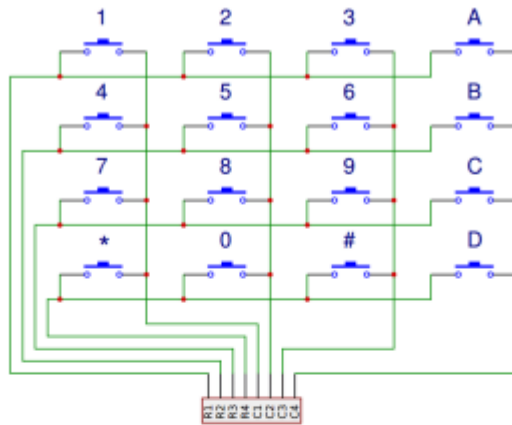


Figura 1.2 Circuito del teclado matricial 4x4.

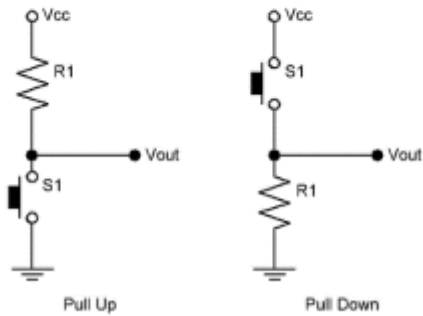
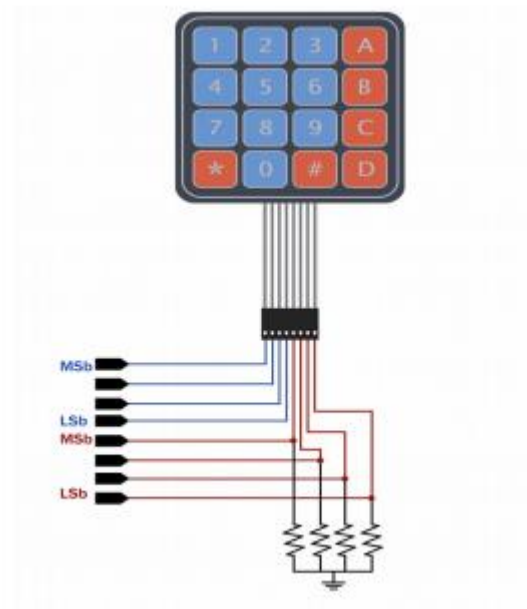


Figura 1.3 Configuraciones pull-down y pull-up.

Para la configuración básica del uso del teclado matricial se tiene el esquema siguiente



## Desarrollo

1.- Realizar un sistema que simule la introducción de un password de cuatro dígitos del teclado matricial, de tal manera, que cuando se tenga el password en forma correcta se activará una señal de salida y permanecerá en ese nivel hasta que se introduzca una nueva clave.

Código en VHDL

Library IEEE;

Use IEEE.Std\_logic\_1164.all;

Use IEEE.Std\_logic\_unsigned.all;

entity TEC\_MATRIX\_4x4 is

Generic (FREQ\_CLK: integer := 50\_000\_00);

Port( CLK: IN STD\_LOGIC;

COLUMNAS: in std\_logic\_vector(3 downto 0);

FILAS: out std\_logic\_vector(3 downto 0);

d: buffer std\_logic\_vector(6 downto 0);

IND: out std\_logic);

end TEC\_MATRIX\_4x4;

architecture behavioral of TEC\_MATRIX\_4x4 is

CONSTANT DELAY\_1MS : INTEGER := (FREQ\_CLK/1000)-1;

CONSTANT DELAY\_10MS : INTEGER := (FREQ\_CLK/100)-1;

SIGNAL BOTON\_PRES: std\_logic\_vector (3 downto 0);--3 down to 0

SIGNAL CONTA\_1MS: INTEGER RANGE 0 TO DELAY\_1MS := 0;

SIGNAL BANDERA: STD\_LOGIC := '0';

SIGNAL CONTA\_10MS: INTEGER RANGE 0 TO DELAY\_10MS := 0;

SIGNAL BANDERA2 : STD\_LOGIC := '0';

SIGNAL BOT\_1 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_2 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_3 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_4 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_5 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_6 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_7 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_8 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_9 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_A : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_B : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_C : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_D : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_0 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_AS : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

```

SIGNAL BOT_GA : STD_LOGIC_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');
SIGNAL FILA_REG_S : STD_LOGIC_VECTOR(3 DOWNT0 0) := (OTHERS=>'0');
SIGNAL FILA : INTEGER RANGE 0 TO 3 := 0;
SIGNAL IND_S : STD_LOGIC := '0';
SIGNAL EDO : INTEGER RANGE 0 TO 1 := 0;
--SIGNAL FILA_REG_S : STD_LOGIC_VECTOR(3 DOWNT0 0) :=
(OTHERS=>'0');
--SIGNAL FILA : INTEGER RANGE 0 TO 3 := 0;
--SIGNAL IND_S : STD_LOGIC := '0';
--SIGNAL EDO : INTEGER RANGE 0 TO 1 := 0;
BEGIN
FILAS <= FILA_REG_S;
--RETARDO 1 MS--
PROCESS(CLK)
BEGIN
IF RISING_EDGE(CLK) THEN
CONTA_1MS <= CONTA_1MS+1;
BANDERA <= '0';
IF CONTA_1MS = DELAY_1MS THEN
CONTA_1MS <= 0;
BANDERA <= '1';
END IF;
END IF;
END PROCESS;
-----
--RETARDO 10 MS--
PROCESS(CLK)
BEGIN
IF RISING_EDGE(CLK) THEN

```

```

CONTA_10MS <= CONTA_10MS+1;
BANDERA2 <= '0';
IF CONTA_10MS = DELAY_10MS THEN
CONTA_10MS <= 0;
BANDERA2 <= '1';
END IF;
END IF;
END PROCESS;

```

-----

--PROCESO EN LAS FILAS ----

```

PROCESS(CLK, BANDERA2)
BEGIN
IF RISING_EDGE(CLK) AND BANDERA2 = '1' THEN
FILA <= FILA+1;
IF FILA = 3 THEN
FILA <= 0;
END IF;
END IF;
END PROCESS;
WITH FILA SELECT
FILA_REG_S <= "1000" WHEN 0,
"0100" WHEN 1,
"0010" WHEN 2,
"0001" WHEN OTHERS;

```

-----

-----PROCESO EN EL TECLADO AL SELECCIONAR UN VALOR-----

```

PROCESS(CLK,BANDERA)
BEGIN

```

```

IF RISING_EDGE(CLK) AND BANDERA = '1' THEN
IF FILA_REG_S = "1000" THEN
BOT_1 <= BOT_1(6 DOWNT0 0)&COLUMNAS(3);
BOT_2 <= BOT_2(6 DOWNT0 0)&COLUMNAS(2);
BOT_3 <= BOT_3(6 DOWNT0 0)&COLUMNAS(1);
BOT_A <= BOT_A(6 DOWNT0 0)&COLUMNAS(0);
ELSIF FILA_REG_S = "0100" THEN
BOT_4 <= BOT_4(6 DOWNT0 0)&COLUMNAS(3);
BOT_5 <= BOT_5(6 DOWNT0 0)&COLUMNAS(2);
BOT_6 <= BOT_6(6 DOWNT0 0)&COLUMNAS(1);
BOT_B <= BOT_B(6 DOWNT0 0)&COLUMNAS(0);
ELSIF FILA_REG_S = "0010" THEN
BOT_7 <= BOT_7(6 DOWNT0 0)&COLUMNAS(3);
BOT_8 <= BOT_8(6 DOWNT0 0)&COLUMNAS(2);
BOT_9 <= BOT_9(6 DOWNT0 0)&COLUMNAS(1);
BOT_C <= BOT_C(6 DOWNT0 0)&COLUMNAS(0);
ELSIF FILA_REG_S = "0001" THEN
BOT_AS <= BOT_AS(6 DOWNT0 0)&COLUMNAS(3);
BOT_0 <= BOT_0(6 DOWNT0 0)&COLUMNAS(2);
BOT_GA <= BOT_GA(6 DOWNT0 0)&COLUMNAS(1);
BOT_D <= BOT_D(6 DOWNT0 0)&COLUMNAS(0);
END IF;
END IF;
END PROCESS;

-----

--SALIDA--
PROCESS(CLK)
BEGIN

```



```

IF RISING_EDGE(CLK) THEN
IF BOT_0 = "11111111" THEN BOTON_PRES <= X"0"; IND_S <= '1';
ELSIF BOT_1 = "11111111" THEN BOTON_PRES <= X"1"; IND_S <= '1';
ELSIF BOT_2 = "11111111" THEN BOTON_PRES <= X"2"; IND_S <= '1';
ELSIF BOT_3 = "11111111" THEN BOTON_PRES <= X"3"; IND_S <= '1';
ELSIF BOT_4 = "11111111" THEN BOTON_PRES <= X"4"; IND_S <= '1';
ELSIF BOT_5 = "11111111" THEN BOTON_PRES <= X"5"; IND_S <= '1';
ELSIF BOT_6 = "11111111" THEN BOTON_PRES <= X"6"; IND_S <= '1';
ELSIF BOT_7 = "11111111" THEN BOTON_PRES <= X"7"; IND_S <= '1';
ELSIF BOT_8 = "11111111" THEN BOTON_PRES <= X"8"; IND_S <= '1';
ELSIF BOT_9 = "11111111" THEN BOTON_PRES <= X"9"; IND_S <= '1';
ELSIF BOT_A = "11111111" THEN BOTON_PRES <= X"A"; IND_S <= '1';
ELSIF BOT_B = "11111111" THEN BOTON_PRES <= X"B"; IND_S <= '1';
ELSIF BOT_C = "11111111" THEN BOTON_PRES <= X"C"; IND_S <= '1';
ELSIF BOT_D = "11111111" THEN BOTON_PRES <= X"D"; IND_S <= '1';
ELSIF BOT_AS = "11111111" THEN BOTON_PRES <= X"E"; IND_S <= '1';
ELSIF BOT_GA = "11111111" THEN BOTON_PRES <= X"F"; IND_S <= '1';
ELSE IND_S <= '0';
END IF;
END IF;
END PROCESS;

```

-----  
--ACTIVACIÓN PARA LA BANDERA UN CICLO DE RELOJ--

```

PROCESS(CLK)
BEGIN
IF RISING_EDGE(CLK) THEN
IF EDO = 0 THEN
IF IND_S = '1' THEN

```

```

IND <= '1';
EDO <= 1;
ELSE
EDO <= 0;
IND <= '0';
END IF;
ELSE
IF IND_S = '1' THEN
EDO <= 1;
IND <= '0';
ELSE
EDO <= 0;
END IF;
END IF;
END IF;
END PROCESS;

```

with BOTON\_PRES select

```

d      <= "1000000" when "0000",--0
        "1001111" when "0001",--1
        "0100100" when "0010",--2
        "0110000" when "0011",--3
        "0011001" when "0100",--4
        "0010010" when "0101",--5
        "0000010" when "0110",--6
        "1111000" when "0111",--7
        "0000000" when "1000",--8
        "0011000" when "1001",--9

```

"0001000" when "1010",--A

"0000011" when "1011",--B

"1000110" when "1100",--C

"0100001" when "1101",--D

"0001001" when "1110",--H

"0000110" when "1111",--E

"1111111" when others;

-----  
END behavioral;

Type	ID	Message
> i	332146	worst-case setup slack is -0.596
> i	332146	worst-case hold slack is 0.148
i	332140	No Recovery paths to report
i	332140	No Removal paths to report
> i	332146	worst-case minimum pulse width slack is -3.000
i	332102	Design is not fully constrained for setup requirements
i	332102	Design is not fully constrained for hold requirements
> i		Quartus Prime Timing Analyzer was successful. 0 errors, 5 warnings
i	293000	Quartus Prime Full Compilation was successful. 0 errors, 11 warnings

Node Name	Direction	Location	
in CLK	Input	PIN_P11	3
in COLUMNAS[3]	Input	PIN_AB20	4
in COLUMNAS[2]	Input	PIN_Y19	4
in COLUMNAS[1]	Input	PIN_AA19	4
in COLUMNAS[0]	Input	PIN_AB19	4
out d[6]	Output	PIN_C17	7
out d[5]	Output	PIN_D17	7
out d[4]	Output	PIN_E16	7
out d[3]	Output	PIN_C16	7
out d[2]	Output	PIN_C15	7
out d[1]	Output	PIN_E15	7
out d[0]	Output	PIN_C14	7
out FILAS[3]	Output	PIN_AA12	4
out FILAS[2]	Output	PIN_AA11	4
out FILAS[1]	Output	PIN_Y10	3
out FILAS[0]	Output	PIN_AB9	3

Asignación del Pin Planner

Actividades complementarias:

**1.-** Realizar un sistema que simule la introducción de un password de cuatro dígitos del teclado matricial, de tal manera, que cuando se tenga el password en forma correcta se activará una señal de salida y permanecerá en ese nivel hasta que se introduzca una nueva clave.

Library IEEE;

Use IEEE.Std\_logic\_1164.all;

Use IEEE.Std\_logic\_unsigned.all;

entity TEC\_MATRIX\_4x4 is

Generic (FREQ\_CLK: integer := 50\_000\_00);

Port( CLK: IN STD\_LOGIC;

COLUMNAS: in std\_logic\_vector(3 downto 0);

FILAS: out std\_logic\_vector(3 downto 0);

d: buffer std\_logic\_vector(6 downto 0);

d1: buffer std\_logic\_vector(6 downto 0);

d2: buffer std\_logic\_vector(6 downto 0);

d3: buffer std\_logic\_vector(6 downto 0);

IND: out std\_logic);

end TEC\_MATRIX\_4x4;

architecture behavioral of TEC\_MATRIX\_4x4 is

CONSTANT DELAY\_1MS : INTEGER := (FREQ\_CLK/1000)-1;

CONSTANT DELAY\_10MS : INTEGER := (FREQ\_CLK/100)-1;

SIGNAL BOTON\_PRES: std\_logic\_vector (3 downto 0);--3 down to 0

SIGNAL BUTTON: std\_logic\_vector(6 downto 0);

SIGNAL CONTA\_1MS: INTEGER RANGE 0 TO DELAY\_1MS := 0;

SIGNAL BANDERA: STD\_LOGIC := '0';

SIGNAL CONTA\_10MS: INTEGER RANGE 0 TO DELAY\_10MS := 0;

SIGNAL BANDERA2 : STD\_LOGIC := '0';

SIGNAL BOT\_1 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_2 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_3 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_4 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_5 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_6 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_7 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_8 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_9 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_A : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_B : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_C : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_D : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_0 : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_AS : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL BOT\_GA : STD\_LOGIC\_VECTOR(7 DOWNT0 0) := (OTHERS=>'0');

SIGNAL FILA\_REG\_S : STD\_LOGIC\_VECTOR(3 DOWNT0 0) := (OTHERS=>'0');

SIGNAL FILA : INTEGER RANGE 0 TO 3 := 0;

SIGNAL IND\_S : STD\_LOGIC := '0';

```

SIGNAL EDO : INTEGER RANGE 0 TO 1 := 0;
--SIGNAL  FILA_REG_S  :  STD_LOGIC_VECTOR(3  DOWNT0  0)  :=
(Others=>'0');
--SIGNAL FILA : INTEGER RANGE 0 TO 3 := 0;
--SIGNAL IND_S : STD_LOGIC := '0';
--SIGNAL EDO : INTEGER RANGE 0 TO 1 := 0;
BEGIN
FILAS <= FILA_REG_S;
--RETARDO 1 MS--
PROCESS(CLK)
BEGIN
IF RISING_EDGE(CLK) THEN
CONTA_1MS <= CONTA_1MS+1;
BANDERA <= '0';
IF CONTA_1MS = DELAY_1MS THEN
CONTA_1MS <= 0;
BANDERA <= '1';
END IF;
END IF;
END PROCESS;
-----
--RETARDO 10 MS--
PROCESS(CLK)
BEGIN
IF RISING_EDGE(CLK) THEN
CONTA_10MS <= CONTA_10MS+1;
BANDERA2 <= '0';
IF CONTA_10MS = DELAY_10MS THEN
CONTA_10MS <= 0;

```

```
BANDERA2 <= '1';
```

```
END IF;
```

```
END IF;
```

```
END PROCESS;
```

```
-----
```

```
--PROCESO EN LAS FILAS ----
```

```
PROCESS(CLK, BANDERA2)
```

```
BEGIN
```

```
IF RISING_EDGE(CLK) AND BANDERA2 = '1' THEN
```

```
FILA <= FILA+1;
```

```
IF FILA = 3 THEN
```

```
FILA <= 0;
```

```
END IF;
```

```
END IF;
```

```
END PROCESS;
```

```
WITH FILA SELECT
```

```
FILA_REG_S <= "1000" WHEN 0,
```

```
"0100" WHEN 1,
```

```
"0010" WHEN 2,
```

```
"0001" WHEN OTHERS;
```

```
-----
```

```
-----PROCESO EN EL TECLADO AL SELECCIONAR UN VALOR-----
```

```
PROCESS(CLK,BANDERA)
```

```
BEGIN
```

```
IF RISING_EDGE(CLK) AND BANDERA = '1' THEN
```

```
IF FILA_REG_S = "1000" THEN
```

```
BOT_1 <= BOT_1(6 DOWNT0 0)&COLUMNAS(3);
```

```
BOT_2 <= BOT_2(6 DOWNT0 0)&COLUMNAS(2);
```

```

BOT_3 <= BOT_3(6 DOWNT0 0)&COLUMNAS(1);
BOT_A <= BOT_A(6 DOWNT0 0)&COLUMNAS(0);
ELSIF FILA_REG_S = "0100" THEN
BOT_4 <= BOT_4(6 DOWNT0 0)&COLUMNAS(3);
BOT_5 <= BOT_5(6 DOWNT0 0)&COLUMNAS(2);
BOT_6 <= BOT_6(6 DOWNT0 0)&COLUMNAS(1);
BOT_B <= BOT_B(6 DOWNT0 0)&COLUMNAS(0);
ELSIF FILA_REG_S = "0010" THEN
BOT_7 <= BOT_7(6 DOWNT0 0)&COLUMNAS(3);
BOT_8 <= BOT_8(6 DOWNT0 0)&COLUMNAS(2);
BOT_9 <= BOT_9(6 DOWNT0 0)&COLUMNAS(1);
BOT_C <= BOT_C(6 DOWNT0 0)&COLUMNAS(0);
ELSIF FILA_REG_S = "0001" THEN
BOT_AS <= BOT_AS(6 DOWNT0 0)&COLUMNAS(3);
BOT_0 <= BOT_0(6 DOWNT0 0)&COLUMNAS(2);
BOT_GA <= BOT_GA(6 DOWNT0 0)&COLUMNAS(1);
BOT_D <= BOT_D(6 DOWNT0 0)&COLUMNAS(0);
END IF;
END IF;
END PROCESS;

```

-----

--SALIDA--

```

PROCESS(CLK)
BEGIN
IF RISING_EDGE(CLK) THEN
IF BOT_0 = "11111111" THEN BOTON_PRES <= X"0"; IND_S <= '1';
ELSIF BOT_1 = "11111111" THEN BOTON_PRES <= X"1"; IND_S <= '1';
ELSIF BOT_2 = "11111111" THEN BOTON_PRES <= X"2"; IND_S <= '1';

```



```

ELSIF BOT_3 = "11111111" THEN BOTON_PRES <= X"3"; IND_S <= '1';
ELSIF BOT_4 = "11111111" THEN BOTON_PRES <= X"4"; IND_S <= '1';
ELSIF BOT_5 = "11111111" THEN BOTON_PRES <= X"5"; IND_S <= '1';
ELSIF BOT_6 = "11111111" THEN BOTON_PRES <= X"6"; IND_S <= '1';
ELSIF BOT_7 = "11111111" THEN BOTON_PRES <= X"7"; IND_S <= '1';
ELSIF BOT_8 = "11111111" THEN BOTON_PRES <= X"8"; IND_S <= '1';
ELSIF BOT_9 = "11111111" THEN BOTON_PRES <= X"9"; IND_S <= '1';
ELSIF BOT_A = "11111111" THEN BOTON_PRES <= X"A"; IND_S <= '1';
ELSIF BOT_B = "11111111" THEN BOTON_PRES <= X"B"; IND_S <= '1';
ELSIF BOT_C = "11111111" THEN BOTON_PRES <= X"C"; IND_S <= '1';
ELSIF BOT_D = "11111111" THEN BOTON_PRES <= X"D"; IND_S <= '1';
ELSIF BOT_AS = "11111111" THEN BOTON_PRES <= X"E"; IND_S <= '1';
ELSIF BOT_GA = "11111111" THEN BOTON_PRES <= X"F"; IND_S <= '1';
ELSE IND_S <= '0';
END IF;
END IF;
END PROCESS;

```

-----

--ACTIVACIÓN PARA LA BANDERA UN CICLO DE RELOJ--

PROCESS(CLK)

BEGIN

IF RISING\_EDGE(CLK) THEN

IF EDO = 0 THEN

IF IND\_S = '1' THEN

-- IND <= '1';

EDO <= 1;

ELSE

EDO <= 0;

```

    -- IND <= '0';
END IF;
ELSE
    IF IND_S = '1' THEN
        EDO <= 1;
        -- IND <= '0';
    ELSE
        EDO <= 0;
    END IF;
END IF;
END IF;
END PROCESS;

```

with BOTON\_PRES select

```

BUTTON    <= "1000000" when "0000",--0
           "1001111" when "0001",--1
           "0100100" when "0010",--2
           "0110000" when "0011",--3
           "0011001" when "0100",--4
           "0010010" when "0101",--5
           "0000010" when "0110",--6
           "1111000" when "0111",--7
           "0000000" when "1000",--8
           "0011000" when "1001",--9
           "0001000" when "1010",--A
           "0000011" when "1011",--B
           "1000110" when "1100",--C
           "0100001" when "1101",--D

```

"0001001" when "1110",--H

"0000110" when "1111",--E

"1111111" when others;

PROCESS(EDO)

BEGIN

IF EDO = 1 THEN

d<=BUTTON;

d1<=d;

d2<=d1;

d3<=d2;

END IF;

--password: AA69

IF d="0011000" AND d1="0000010" AND d2="0001000" AND  
d3="0001000" THEN

IND <= '1';

ELSE







IND <='0';

END IF;

END PROCESS;

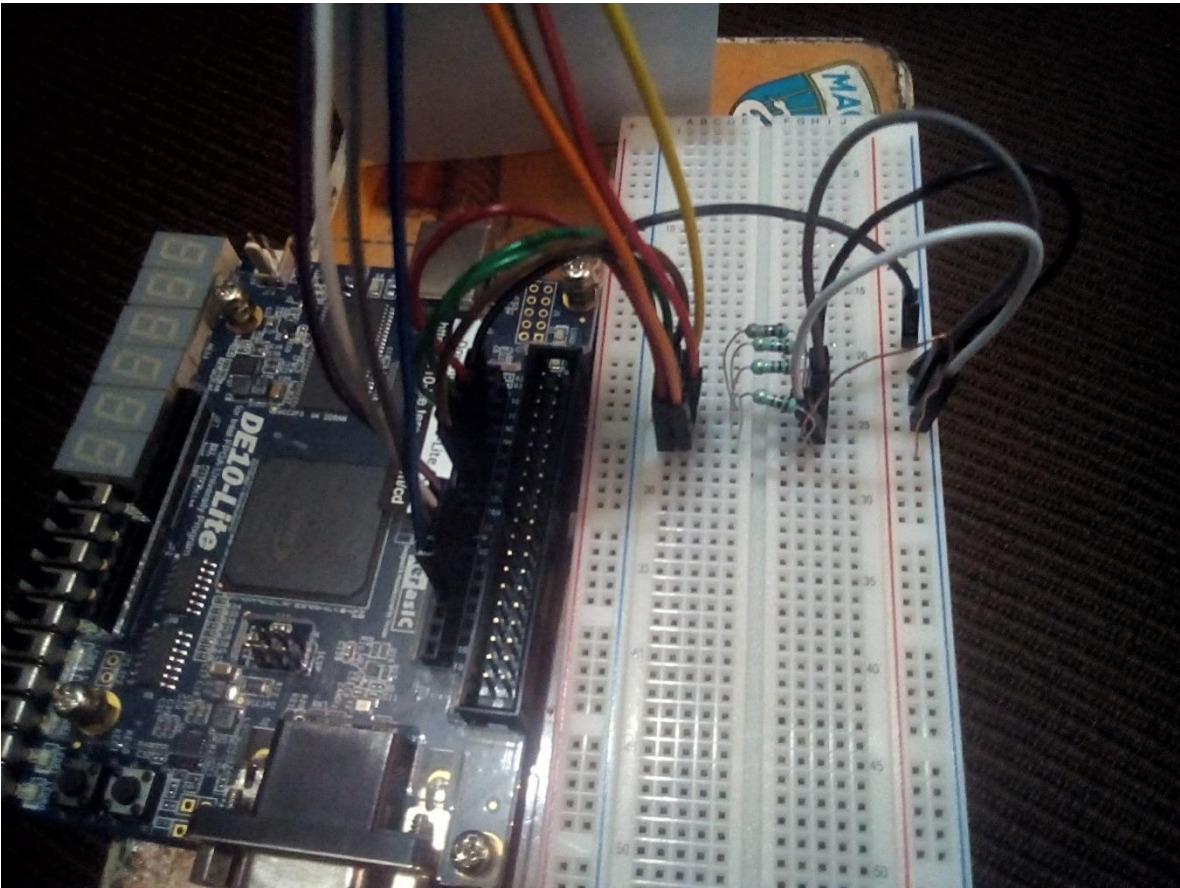
END behavioral;

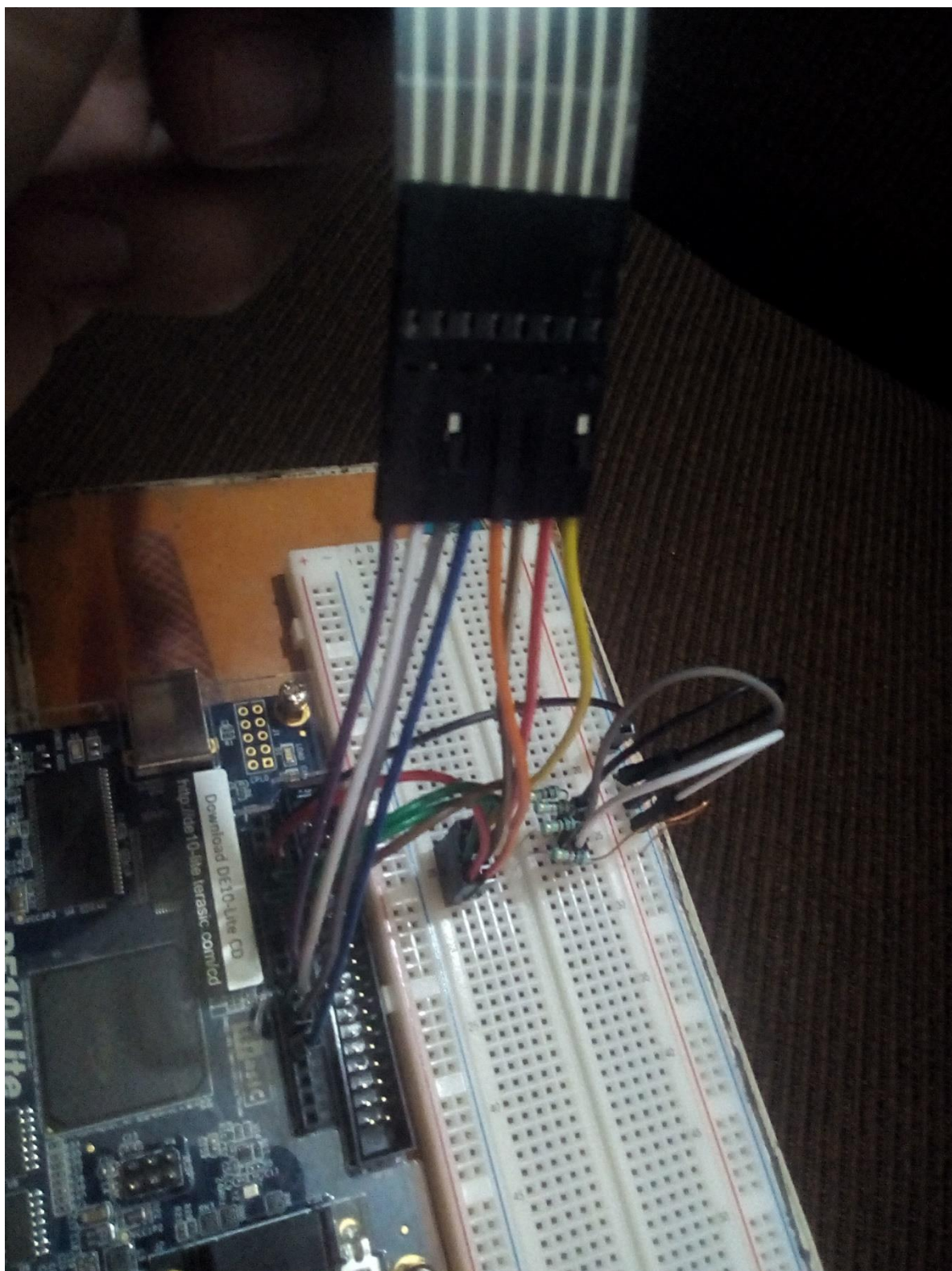
---

Type	ID	Message
	332140	No Removal paths to report
> 	332146	worst-case minimum pulse width slack is -3.000
	332102	Design is not fully constrained for setup requirements
	332102	Design is not fully constrained for hold requirements
> 	Quartus Prime Timing Analyzer was successful. 0 errors, 5 warnings	
	293000	Quartus Prime Full Compilation was successful. 0 errors, 15 warnings

	Node Name	Direction	Location	I/O
All Pins	in CLK	Input	PIN_P11	3
	in COLUMNAS[3]	Input	PIN_AB20	4
	in COLUMNAS[2]	Input	PIN_Y19	4
	in COLUMNAS[1]	Input	PIN_AA19	4
	in COLUMNAS[0]	Input	PIN_AB19	4
	out d[6]	Output	PIN_C17	7
	out d[5]	Output	PIN_D17	7
	out d[4]	Output	PIN_E16	7
	out d[3]	Output	PIN_C16	7
	out d[2]	Output	PIN_C15	7
	out d[1]	Output	PIN_E15	7
	out d[0]	Output	PIN_C14	7
	out d1[6]	Output	PIN_B17	7
	out d1[5]	Output	PIN_A18	7
	out d1[4]	Output	PIN_A17	7
	out d1[3]	Output	PIN_B16	7
	out d1[2]	Output	PIN_E18	6
	out d1[1]	Output	PIN_D18	6

	Node Name	Direction	Location
All Pins	out d1[0]	Output	PIN_C18
	out d2[6]	Output	PIN_B22
	out d2[5]	Output	PIN_C22
	out d2[4]	Output	PIN_B21
	out d2[3]	Output	PIN_A21
	out d2[2]	Output	PIN_B19
	out d2[1]	Output	PIN_A20
	out d2[0]	Output	PIN_B20
	out d3[6]	Output	PIN_E17
	out d3[5]	Output	PIN_D19
	out d3[4]	Output	PIN_C20
	out d3[3]	Output	PIN_C19
	out d3[2]	Output	PIN_E21
	out d3[1]	Output	PIN_E22
	out d3[0]	Output	PIN_F21
	out FILAS[3]	Output	PIN_AA12
	out FILAS[2]	Output	PIN_AA11
	out FILAS[1]	Output	PIN_Y10
	out FILAS[0]	Output	PIN_AB9
	out IND	Output	PIN_D13





## **Conclusión**

Para realizar esta práctica fue importante poner atención a las conexiones hechas en la protoboard entre la Fpga y el teclado, ya que de este depende el poder ver en el display el dígito o letra que queramos

Video demostrativos:

<https://drive.google.com/drive/folders/1ECV3buGaoOF642sjaKNgk7DNm3KaMJpC?usp=sharing>

## **Referencia:**

Navabi, Z. (2007). VHDL:Modular desing ans synthesis of cores and systems. USA: McGraw-Hill