Universidad Autónoma De Querétaro
Facultad de Ingeniería

Autonomous University of Querétaro

Faculty of Engineering.

Career: Automation Engineer.

Subject: Systems with reconfigurable logic.

Student: Martínez Murillo Omar Yarif.
Student: Diego Joel Zúñiga Fragoso.
Student: Daniela del Carmen Manríquez Navarro.
Student: Joselyn Gallegos Abreo.

Practice 3: Switch case,IF,When

Field-Programmable Gate Arrays (FPGAs)
Revolutionizing Digital Design
FPGA
CIRCUITS DIY

UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
FACULTAD DE INGENIERÍA

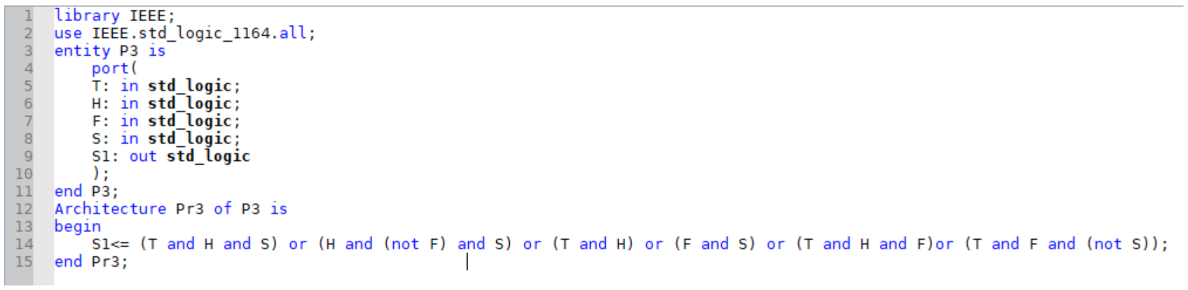
Introduction

The practice is divided into three parts. The first part involves the use of Karnaugh maps, where the task is to design a system for monitoring environmental conditions. A truth table is generated, and the Karnaugh map is used to obtain the most simplified equation for the Boolean operations. The second part focuses on the use of the switch-case statement in VHDL. Here, the goal is to display the alphabet on an FPGA by creating a truth table, performing a simulation, and then programming it. Lastly, the practice covers the use of the if statement in VHDL to display words, again creating a truth table, performing a simulation, and then programming it.

Methodology

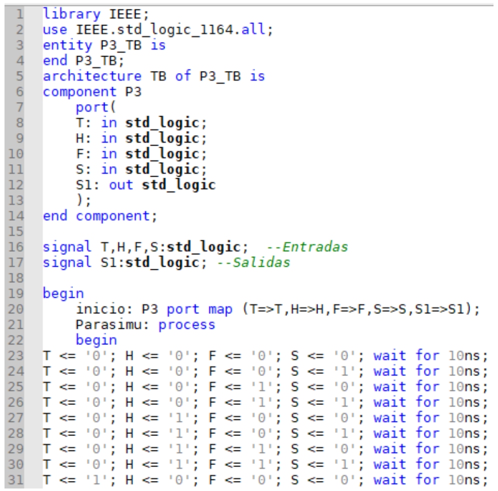
- First, we create the Karnaugh map based on the truth table derived from the given conditions.

- Second, the equation is extracted, and the base code is programmed in Aldec-Active-VHDL.



```vhdl
library IEEE;
use IEEE.std_logic_1164.all;
entity P3 is
    port(
    T: in std_logic;
    H: in std_logic;
    F: in std_logic;
    S: in std_logic;
    S1: out std_logic
    );
end P3;
Architecture Pr3 of P3 is
begin
    S1<= (T and H and S) or (H and (not F) and S) or (T and H) or (F and S) or (T and H and F)or (T and F and (not S));
end Pr3;
```

- Third, a testbench is created for simulation.



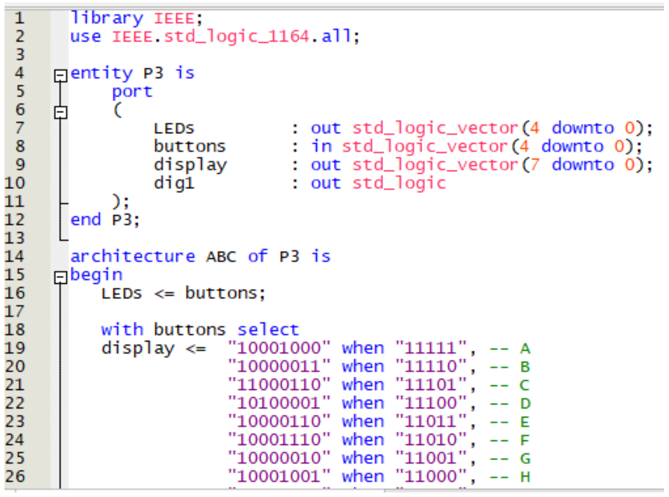
```vhdl
library IEEE;
use IEEE.std_logic_1164.all;
entity P3_TB is
end P3_TB;
architecture TB of P3_TB is
component P3
    port(
    T: in std_logic;
    H: in std_logic;
    F: in std_logic;
    S: in std_logic;
    S1: out std_logic
    );
end component;

signal T,H,F,S:std_logic;  --Entradas
signal S1:std_logic; --Salidas

begin
    inicio: P3 port map (T=>T,H=>H,F=>F,S=>S,S1=>S1);
    Parasimu: process
    begin
T <= '0'; H <= '0'; F <= '0'; S <= '0'; wait for 10ns;
T <= '0'; H <= '0'; F <= '0'; S <= '1'; wait for 10ns;
T <= '0'; H <= '0'; F <= '1'; S <= '0'; wait for 10ns;
T <= '0'; H <= '0'; F <= '1'; S <= '1'; wait for 10ns;
T <= '0'; H <= '1'; F <= '0'; S <= '0'; wait for 10ns;
T <= '0'; H <= '1'; F <= '0'; S <= '1'; wait for 10ns;
T <= '0'; H <= '1'; F <= '1'; S <= '0'; wait for 10ns;
T <= '0'; H <= '1'; F <= '1'; S <= '1'; wait for 10ns;
T <= '1'; H <= '0'; F <= '0'; S <= '0'; wait for 10ns;
```

- Fourth, a truth table is created for the switch-case code, where specific combinations are defined to represent letters of the alphabet.

	s1	s2	s3	s4	b1	letra	
1	0	0	0	0	0	A	10001000
2	0	0	0	0	1	B	10000011
3	0	0	0	1	0	C	11000110
4	0	0	0	1	1	D	10100001
5	0	0	1	0	0	E	10000110
6	0	0	1	0	1	F	10001110
7	0	0	1	1	0	G	10000010
8	0	0	1	1	1	H	10001001
9	0	1	0	0	0	I	11001111
10	0	1	0	0	1	J	11110001
11	0	1	0	1	0	L	11000111
12	0	1	0	1	1	M	10110000
13	0	1	1	0	0	N	11001000
14	0	1	1	0	1	O	11000000
15	0	1	1	1	0	P	10001100
16	0	1	1	1	1	Q	10011000
17	1	0	0	0	0	R	11001110
18	1	0	0	0	1	S	10010010
19	1	0	0	1	0	T	10000111
20	1	0	0	1	1	U	11000001
21	1	0	1	0	0	W	10000110
22	1	0	1	0	1	Z	10100100

- Fifth, the configuration is set up in Aldec-Active-VHDL, followed by simulation.

- Sixth, the configuration is transferred to Quartus, and the pins are assigned.



```
library IEEE;
use IEEE.std_logic_1164.all;

entity P3 is
     port
     (
          LEDs          : out std_logic_vector(4 downto 0);
          buttons       : in std_logic_vector(4 downto 0);
          display       : out std_logic_vector(7 downto 0);
          dig1          : out std_logic
     );
end P3;

architecture ABC of P3 is
begin
    LEDs <= buttons;

    with buttons select
    display <=   "10001000" when "11111", -- A
                 "10000011" when "11110", -- B
                 "11000110" when "11101", -- C
                 "10100001" when "11100", -- D
                 "10000110" when "11011", -- E
                 "10001110" when "11010", -- F
                 "10000010" when "11001", -- G
                 "10001001" when "11000", -- H
```

- Seventh, the program is loaded onto the FPGA.

- Eighth, a truth table is created for the if statement, which should display 4-letter words.

#	I1	I2	I3	I4	I5	h	g	f	e	d	c	b	a
A	0	0	0	0	0	1	0	0	0	1	0	0	0
B	0	0	0	0	1	1	0	0	0	0	0	1	1
C	0	0	0	1	0	1	1	0	0	0	1	1	0
D	0	0	0	1	1	1	0	1	0	0	0	0	1
E	0	0	1	0	0	1	0	0	0	0	1	1	0
F	0	0	1	0	1	1	0	0	0	1	1	1	0
G	0	0	1	1	0	1	0	0	0	0	0	1	0
H	0	0	1	1	1	1	0	0	0	1	0	1	1
I	0	1	0	0	0	1	1	1	1	1	0	0	1
J	0	1	0	0	1	1	1	1	1	0	0	0	1
K	0	1	0	1	0	1	0	0	0	0	1	0	1
L	0	1	0	1	1	1	1	0	0	0	1	1	1
M	0	1	1	0	0	1	0	1	1	0	0	0	0
N	0	1	1	0	1	1	0	1	0	1	0	1	1
Ñ	0	1	1	1	0	1	0	1	0	1	0	1	0
O	0	1	1	1	1	1	1	0	0	0	0	0	0
P	1	0	0	0	0	1	0	0	0	1	1	0	0
Q	1	0	0	0	1	1	0	0	1	1	0	0	0
R	1	0	0	1	0	1	1	0	0	1	1	1	0
S	1	0	0	1	1	1	0	0	1	0	0	1	0
T	1	0	1	0	0	1	0	0	0	0	1	1	1
U	1	0	1	0	1	1	1	1	0	0	0	1	1
V	1	0	1	1	0	1	1	0	0	0	0	0	1
W	1	0	1	1	1	1	0	0	0	0	1	1	0
X	1	1	0	0	0	1	0	0	0	1	0	0	1
Y	1	1	0	0	1	1	0	0	1	0	0	0	1
Z	1	1	0	1	0	1	0	1	0	0	1	0	0

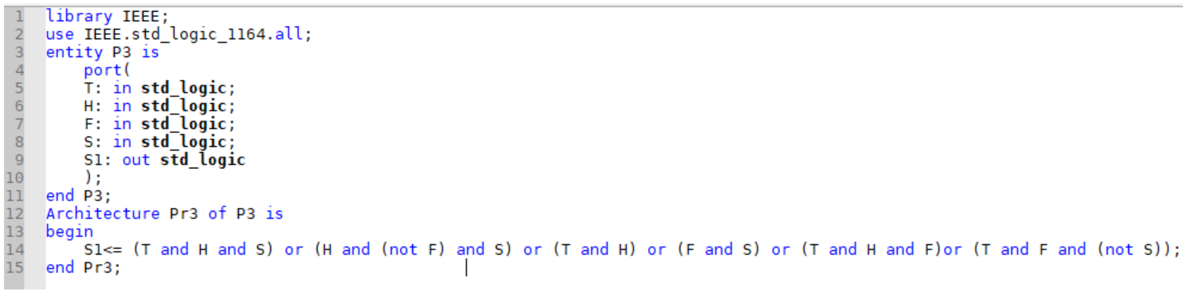
- Ninth, the configuration is set up in Aldec-Active-VHDL, followed by simulation.

- Tenth, the configuration is transferred to Quartus, and the pins are assigned.

- Eleventh, the program is loaded onto the FPGA.

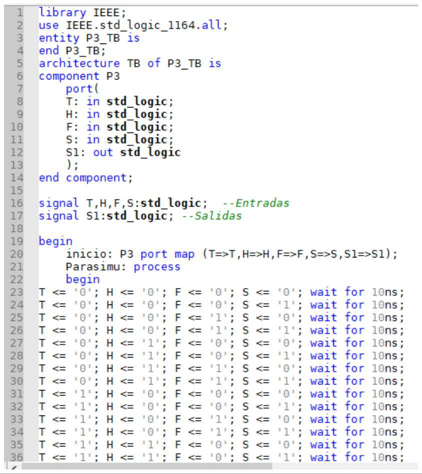
Results

Base algorithm Karnaugh

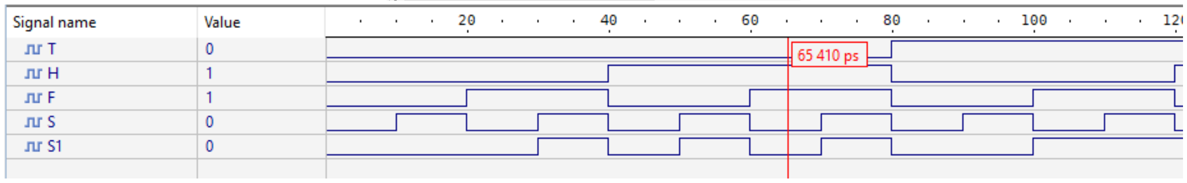


```vhdl
library IEEE;
use IEEE.std_logic_1164.all;
entity P3 is
    port(
    T: in std_logic;
    H: in std_logic;
    F: in std_logic;
    S: in std_logic;
    S1: out std_logic
    );
end P3;
Architecture Pr3 of P3 is
begin
    S1<= (T and H and S) or (H and (not F) and S) or (T and H) or (F and S) or (T and H and F)or (T and F and (not S));
end Pr3;
```

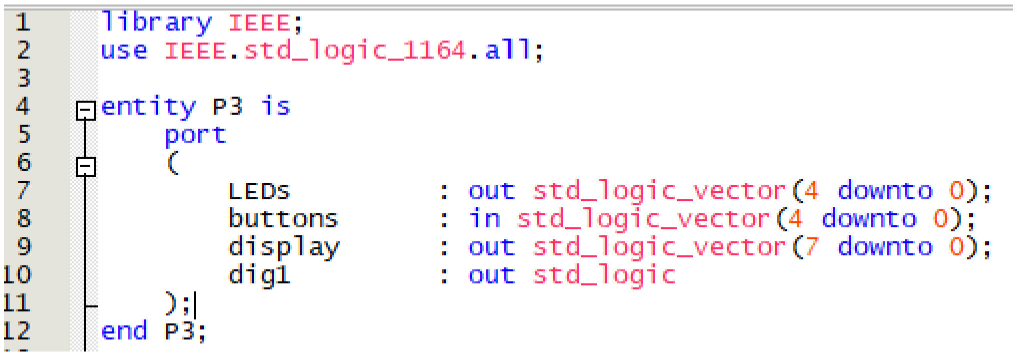
Test Bench



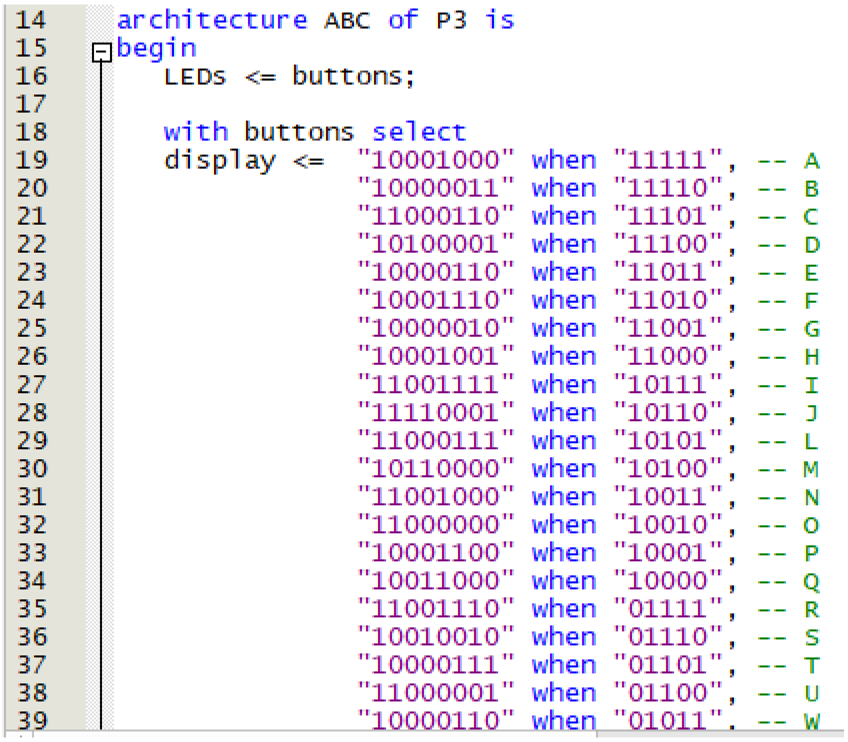
```vhdl
library IEEE;
use IEEE.std_logic_1164.all;
entity P3_TB is
end P3_TB;
architecture TB of P3_TB is
component P3
    port(
    T: in std_logic;
    H: in std_logic;
    F: in std_logic;
    S: in std_logic;
    S1: out std_logic
    );
end component;

signal T,H,F,S:std_logic;  --Entradas
signal S1:std_logic; --Salidas

begin
    inicio: P3 port map (T=>T,H=>H,F=>F,S=>S,S1=>S1);
    Parasimu: process
    begin
T <= '0'; H <= '0'; F <= '0'; S <= '0'; wait for 10ns;
T <= '0'; H <= '0'; F <= '0'; S <= '1'; wait for 10ns;
T <= '0'; H <= '0'; F <= '1'; S <= '0'; wait for 10ns;
T <= '0'; H <= '0'; F <= '1'; S <= '1'; wait for 10ns;
T <= '0'; H <= '1'; F <= '0'; S <= '0'; wait for 10ns;
T <= '0'; H <= '1'; F <= '0'; S <= '1'; wait for 10ns;
T <= '0'; H <= '1'; F <= '1'; S <= '0'; wait for 10ns;
T <= '0'; H <= '1'; F <= '1'; S <= '1'; wait for 10ns;
T <= '1'; H <= '0'; F <= '0'; S <= '0'; wait for 10ns;
T <= '1'; H <= '0'; F <= '0'; S <= '1'; wait for 10ns;
T <= '1'; H <= '0'; F <= '1'; S <= '0'; wait for 10ns;
T <= '1'; H <= '0'; F <= '1'; S <= '1'; wait for 10ns;
T <= '1'; H <= '1'; F <= '0'; S <= '0'; wait for 10ns;
T <= '1'; H <= '1'; F <= '0'; S <= '1'; wait for 10ns;
```



Base algorithm of Switch



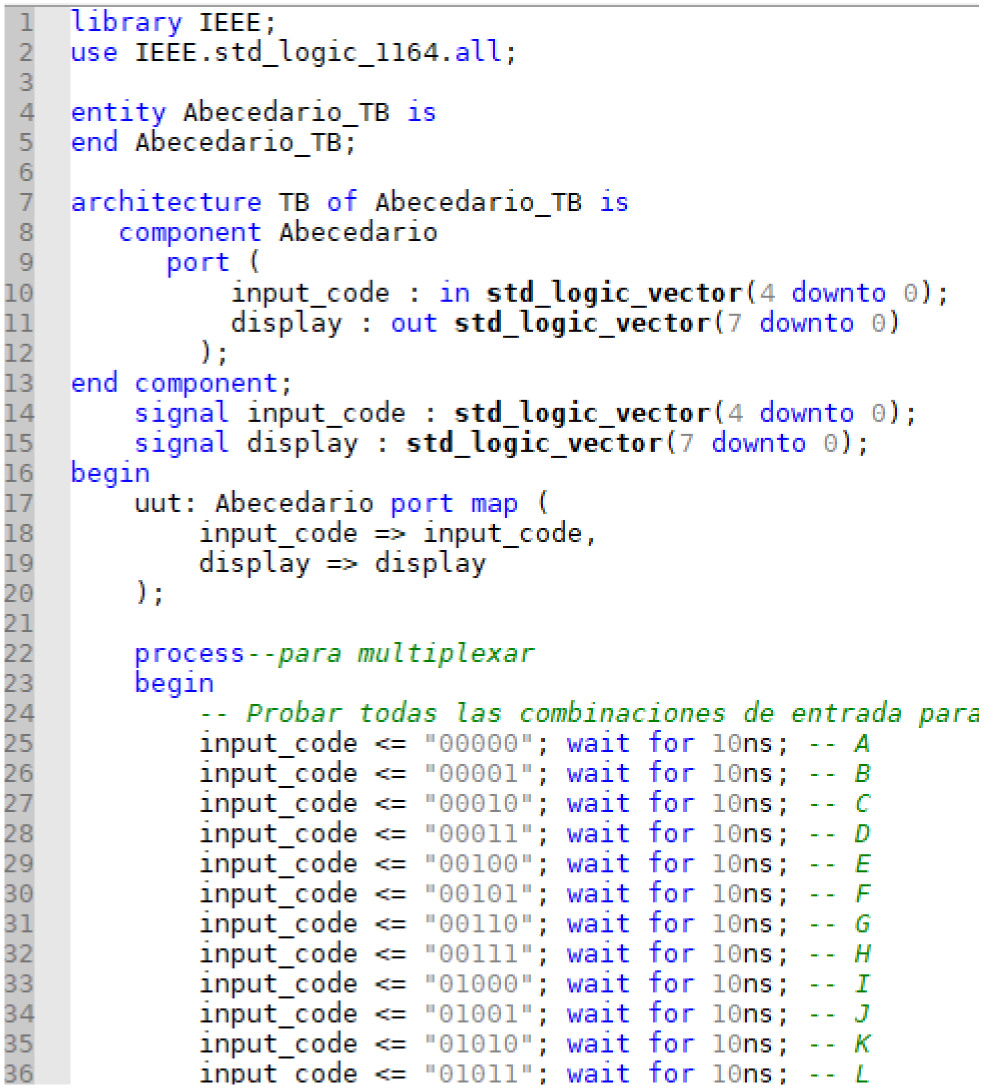
```
library IEEE;
use IEEE.std_logic_1164.all;

entity P3 is
    port
    (
        LEDs        : out std_logic_vector(4 downto 0);
        buttons     : in std_logic_vector(4 downto 0);
        display     : out std_logic_vector(7 downto 0);
        dig1        : out std_logic
    );
end P3;
```



```
architecture ABC of P3 is
begin
    LEDs <= buttons;

    with buttons select
    display <=   "10001000" when "11111", -- A
                 "10000011" when "11110", -- B
                 "11000110" when "11101", -- C
                 "10100001" when "11100", -- D
                 "10000110" when "11011", -- E
                 "10001110" when "11010", -- F
                 "10000010" when "11001", -- G
                 "10001001" when "11000", -- H
                 "11001111" when "10111", -- I
                 "11110001" when "10110", -- J
                 "11000111" when "10101", -- L
                 "10110000" when "10100", -- M
                 "11001000" when "10011", -- N
                 "11000000" when "10010", -- O
                 "10001100" when "10001", -- P
                 "10011000" when "10000", -- Q
                 "11001110" when "01111", -- R
                 "10010010" when "01110", -- S
                 "10000111" when "01101", -- T
                 "11000001" when "01100", -- U
                 "10000110" when "01011", -- W
```

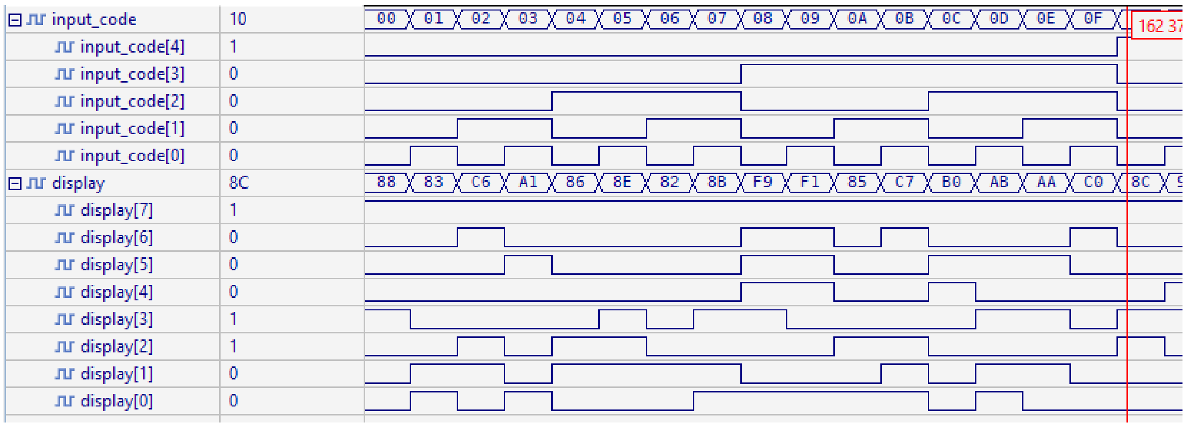
Universidad Autónoma De Querétaro
Facultad de Ingeniería

Test Bench



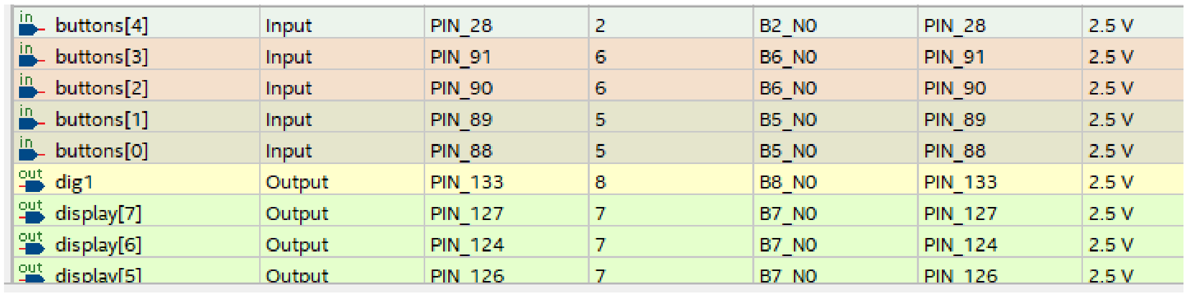
```vhdl
library IEEE;
use IEEE.std_logic_1164.all;

entity Abecedario_TB is
end Abecedario_TB;

architecture TB of Abecedario_TB is
   component Abecedario
      port (
          input_code : in std_logic_vector(4 downto 0);
          display : out std_logic_vector(7 downto 0)
        );
end component;
    signal input_code : std_logic_vector(4 downto 0);
    signal display : std_logic_vector(7 downto 0);
begin
    uut: Abecedario port map (
        input_code => input_code,
        display => display
    );

    process--para multiplexar
    begin
        -- Probar todas las combinaciones de entrada para
        input_code <= "00000"; wait for 10ns; -- A
        input_code <= "00001"; wait for 10ns; -- B
        input_code <= "00010"; wait for 10ns; -- C
        input_code <= "00011"; wait for 10ns; -- D
        input_code <= "00100"; wait for 10ns; -- E
        input_code <= "00101"; wait for 10ns; -- F
        input_code <= "00110"; wait for 10ns; -- G
        input_code <= "00111"; wait for 10ns; -- H
        input_code <= "01000"; wait for 10ns; -- I
        input_code <= "01001"; wait for 10ns; -- J
        input_code <= "01010"; wait for 10ns; -- K
        input_code <= "01011"; wait for 10ns; -- L
```

Simulation



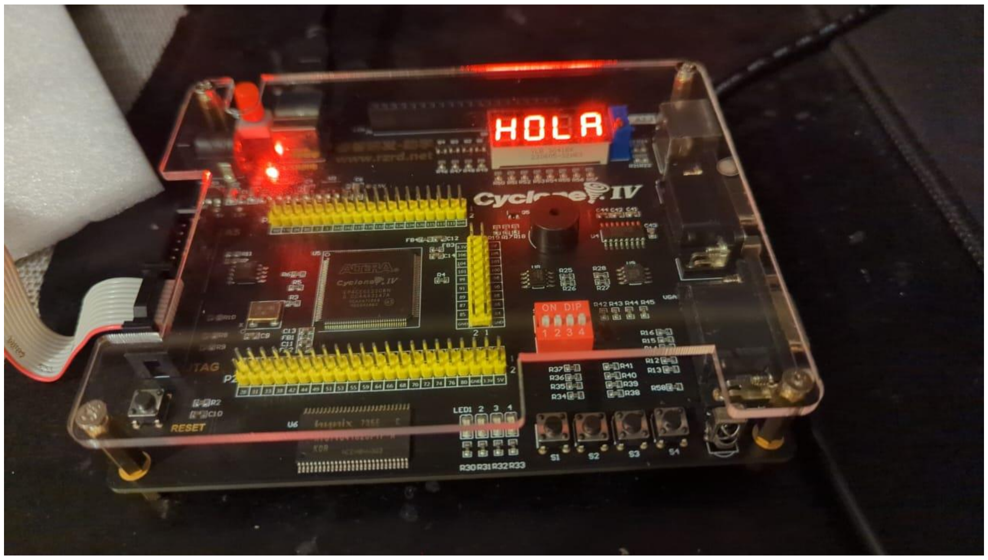
input_code	10	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
input_code[4]	1	
input_code[3]	0	
input_code[2]	0	
input_code[1]	0	
input_code[0]	0	
display	8C	88 83 C6 A1 86 8E 82 8B F9 F1 85 C7 B0 AB AA C0 8C
display[7]	1	
display[6]	0	
display[5]	0	
display[4]	0	
display[3]	1	
display[2]	1	
display[1]	0	
display[0]	0	

Asigned PIN's of Switch

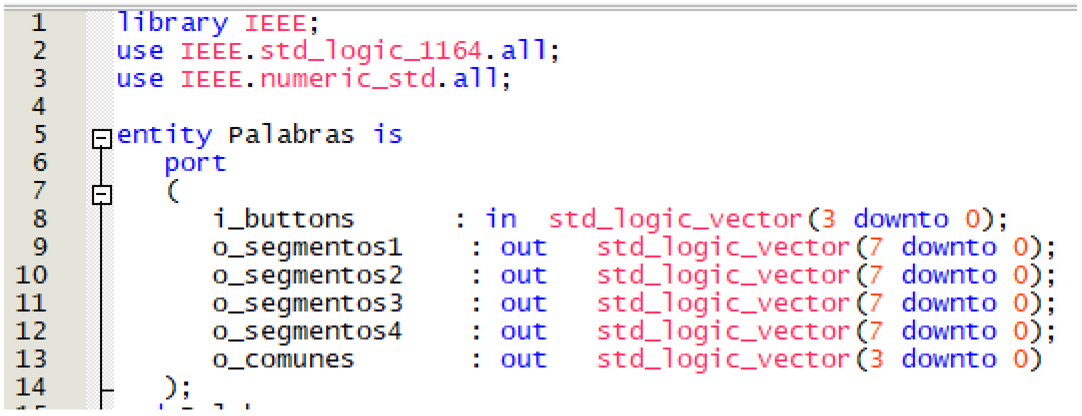


buttons[4]	Input	PIN_28	2	B2_N0	PIN_28	2.5 V
buttons[3]	Input	PIN_91	6	B6_N0	PIN_91	2.5 V
buttons[2]	Input	PIN_90	6	B6_N0	PIN_90	2.5 V
buttons[1]	Input	PIN_89	5	B5_N0	PIN_89	2.5 V
buttons[0]	Input	PIN_88	5	B5_N0	PIN_88	2.5 V
dig1	Output	PIN_133	8	B8_N0	PIN_133	2.5 V
display[7]	Output	PIN_127	7	B7_N0	PIN_127	2.5 V
display[6]	Output	PIN_124	7	B7_N0	PIN_124	2.5 V
display[5]	Output	PIN_126	7	B7_N0	PIN_126	2.5 V

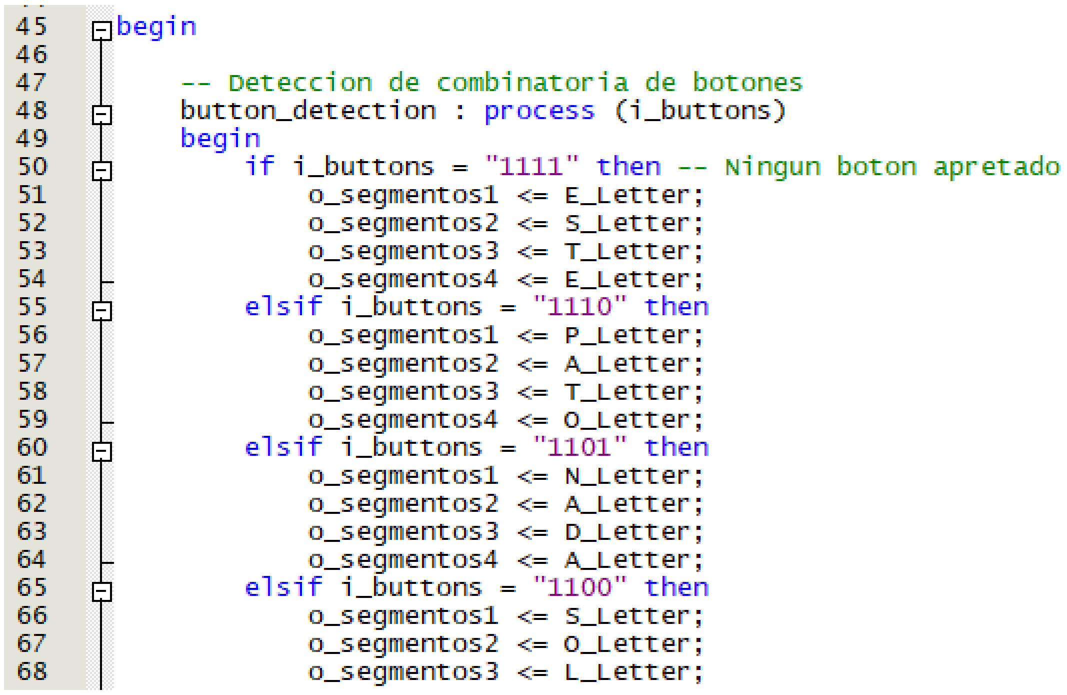
Hardware images



Base algorithm of IF

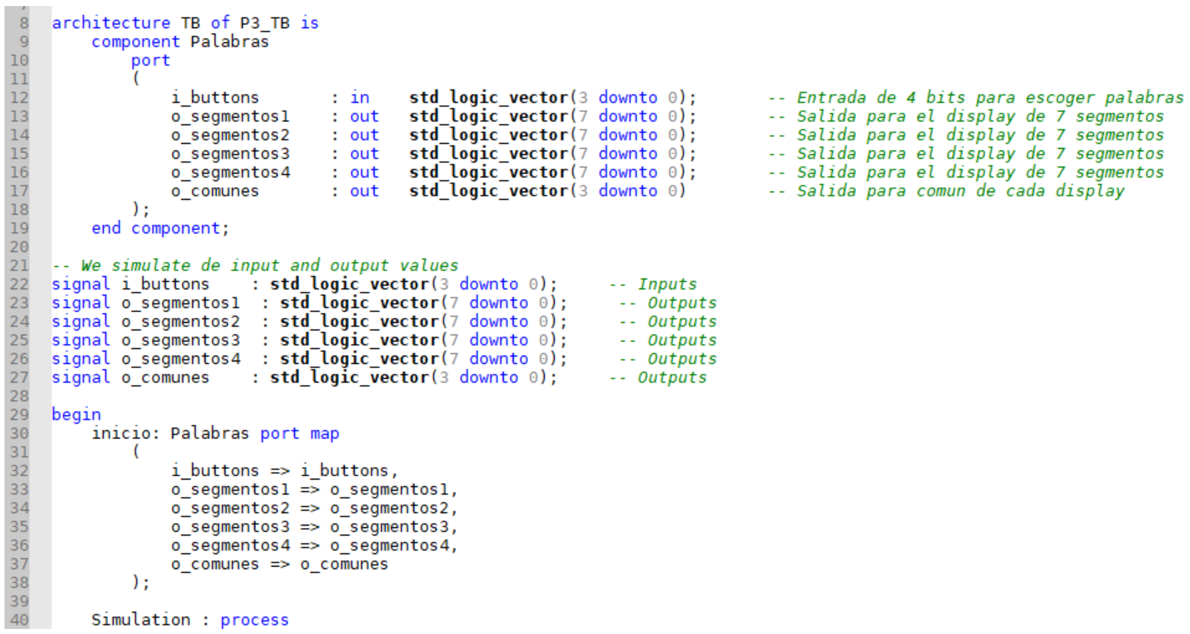


```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity Palabras is
   port
   (
      i_buttons       : in  std_logic_vector(3 downto 0);
      o_segmentos1     : out   std_logic_vector(7 downto 0);
      o_segmentos2     : out   std_logic_vector(7 downto 0);
      o_segmentos3     : out   std_logic_vector(7 downto 0);
      o_segmentos4     : out   std_logic_vector(7 downto 0);
      o_comunes        : out   std_logic_vector(3 downto 0)
   );
```

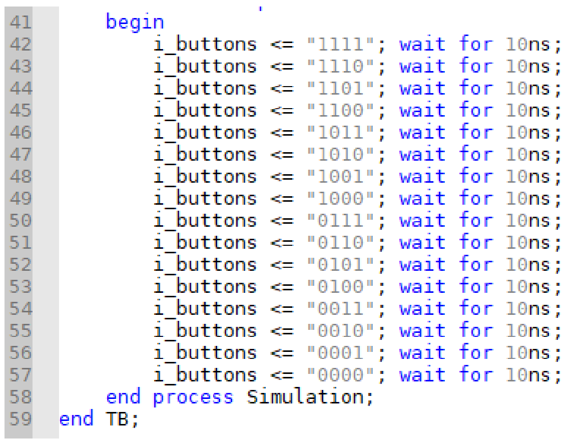


```
begin

    -- Deteccion de combinatoria de botones
    button_detection : process (i_buttons)
    begin
        if i_buttons = "1111" then -- Ningun boton apretado
            o_segmentos1 <= E_Letter;
            o_segmentos2 <= S_Letter;
            o_segmentos3 <= T_Letter;
            o_segmentos4 <= E_Letter;
        elsif i_buttons = "1110" then
            o_segmentos1 <= P_Letter;
            o_segmentos2 <= A_Letter;
            o_segmentos3 <= T_Letter;
            o_segmentos4 <= O_Letter;
        elsif i_buttons = "1101" then
            o_segmentos1 <= N_Letter;
            o_segmentos2 <= A_Letter;
            o_segmentos3 <= D_Letter;
            o_segmentos4 <= A_Letter;
        elsif i_buttons = "1100" then
            o_segmentos1 <= S_Letter;
            o_segmentos2 <= O_Letter;
            o_segmentos3 <= L_Letter;
```

Test Bench



```
architecture TB of P3_TB is
    component Palabras
        port
        (
            i_buttons       : in    std_logic_vector(3 downto 0);       -- Entrada de 4 bits para escoger palabras
            o_segmentos1    : out   std_logic_vector(7 downto 0);       -- Salida para el display de 7 segmentos
            o_segmentos2    : out   std_logic_vector(7 downto 0);       -- Salida para el display de 7 segmentos
            o_segmentos3    : out   std_logic_vector(7 downto 0);       -- Salida para el display de 7 segmentos
            o_segmentos4    : out   std_logic_vector(7 downto 0);       -- Salida para el display de 7 segmentos
            o_comunes       : out   std_logic_vector(3 downto 0)        -- Salida para comun de cada display
        );
    end component;

-- We simulate de input and output values
signal i_buttons     : std_logic_vector(3 downto 0);      -- Inputs
signal o_segmentos1   : std_logic_vector(7 downto 0);      -- Outputs
signal o_segmentos2   : std_logic_vector(7 downto 0);      -- Outputs
signal o_segmentos3   : std_logic_vector(7 downto 0);      -- Outputs
signal o_segmentos4   : std_logic_vector(7 downto 0);      -- Outputs
signal o_comunes     : std_logic_vector(3 downto 0);      -- Outputs

begin
    inicio: Palabras port map
        (
            i_buttons => i_buttons,
            o_segmentos1 => o_segmentos1,
            o_segmentos2 => o_segmentos2,
            o_segmentos3 => o_segmentos3,
            o_segmentos4 => o_segmentos4,
            o_comunes => o_comunes
        );

    Simulation : process
```

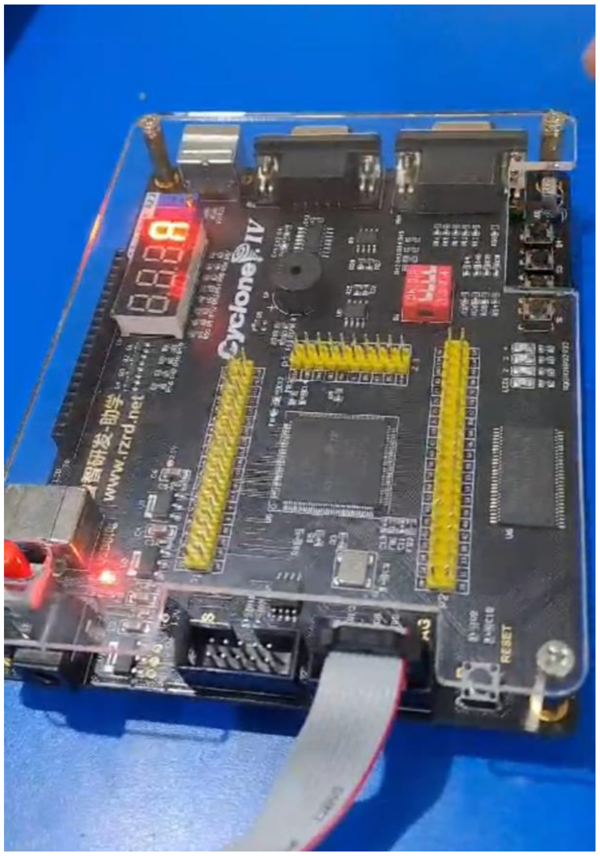


```
    begin
        i_buttons <= "1111"; wait for 10ns;
        i_buttons <= "1110"; wait for 10ns;
        i_buttons <= "1101"; wait for 10ns;
        i_buttons <= "1100"; wait for 10ns;
        i_buttons <= "1011"; wait for 10ns;
        i_buttons <= "1010"; wait for 10ns;
        i_buttons <= "1001"; wait for 10ns;
        i_buttons <= "1000"; wait for 10ns;
        i_buttons <= "0111"; wait for 10ns;
        i_buttons <= "0110"; wait for 10ns;
        i_buttons <= "0101"; wait for 10ns;
        i_buttons <= "0100"; wait for 10ns;
        i_buttons <= "0011"; wait for 10ns;
        i_buttons <= "0010"; wait for 10ns;
        i_buttons <= "0001"; wait for 10ns;
        i_buttons <= "0000"; wait for 10ns;
    end process Simulation;
end TB;
```

Truth table

I 1	I 2	I 3	I 4	S1								S2								S3								S4							
1	1	1	1	1	0	0	0	0	1	1	0	1	0	0	1	0	0	1	0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	0
1	1	1	0	1	0	0	0	1	1	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
1	1	0	1	1	0	1	0	1	0	1	1	1	0	0	0	1	0	0	0	1	0	1	0	0	0	0	1	1	0	0	0	1	0	0	0
1	1	0	0	1	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0	1	1	0	0	0	1	1	1	1	1	0	0	0	0	0	0
1	0	1	1	1	1	0	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	0	0	1	1	1	0	1	1	0	0	0	0	0	0
1	0	1	0	1	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
1	0	0	1	1	0	0	0	0	0	1	1	1	0	0	0	0	1	1	0	1	0	0	0	0	0	1	1	1	0	0	0	0	1	1	1
1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	1	1	1	0	0	0	1	1	1	0	0	0	1	0	0	0
0	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	1	0	0	1	1	0	0	1	0	0	1	0
0	1	1	0	1	0	0	0	1	0	1	1	1	0	0	0	1	0	0	0	1	1	0	0	0	1	1	0	1	0	0	0	0	1	1	0
0	1	0	1	1	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	1	0	1	0	0	0	0	1	1	1	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0	0	1	1	1	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0	1	0	1	0	1	0	1	1
0	0	1	1	1	1	0	0	0	1	1	0	1	0	0	0	1	0	0	0	1	0	1	0	0	0	0	1	1	0	0	0	1	0	0	0
0	0	1	0	1	1	0	0	0	1	1	0	1	0	0	0	1	0	0	0	1	0	0	1	0	0	1	0	1	0	0	0	1	0	0	0
0	0	0	1	1	0	0	0	0	1	1	0	1	0	0	1	0	0	1	0	1	0	0	0	0	1	1	1	1	0	0	0	1	0	0	0
0	0	0	0	1	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0	1	1	0	0	0	1	1	1	1	0	0	0	1	0	0	0

Hardware images



Conclusión

Throughout the practice, we successfully represented letters on a 7-segment display, configuring each letter through a combination of 5 inputs. Additionally, we gained experience using the Quartus software, which allowed us to become more familiar with VHDL and its selective structures, such as `IF` and `SWITCH`. We also learned to use the `process` block to generate sequential actions, such as multiplexing the displays. This practice gave us a deeper understanding of VHDL programming and the control of displays through combinational and sequential logic.