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 7: LCD

UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
FACULTAD DE INGENIERÍA

Introduction

LCD screens use liquid crystals that electrically align to allow or block light, thereby creating images. These crystals do not emit light by themselves, so LCD screens require a backlight source, such as fluorescent lamps or LEDs.

In this project, we programmed an FPGA to control a 16x2 arithmetic LCD using a state machine. It was implemented so that the LCD would display the names of the team members, updating whenever a change in the clock cycle was detected.

Methodology

Step 1: State Machine Base.

The state machine code was used as the foundation, where the clock was first set up and the number of states to be used was adjusted.

Step 2: LCD Variable Initialization.

Variables were defined and associated with all the LCD pins, as well as internal signals to assist in the state machine process.

Step 3: Constants were established to switch between names on the LCD based on a combination of button inputs and to reset the clock.

Step 4: The pins were assigned to the variables, and the code was compiled.

Step 5: The FPGA was programmed, and its functionality was verified.

Results

Base algorithm LCD

```
1   library IEEE;
2   use IEEE.std_logic_1164.all;
3   use IEEE.numeric_std.all;
4
5   entity LCD is
6       port
7       (
8           CLK          : in STD_LOGIC;
9           RS, E, RW    : out STD_LOGIC;
10          Datos        : out std_logic_vector(7 downto 0);
11
12          LEDS         : out std_logic_vector(1 downto 0);
13            Botones      : in std_logic_vector(1 downto 0);
14            ST           : in std_logic
15      );
16  end LCD;
17  --------------------------------------------------------------------------------
```

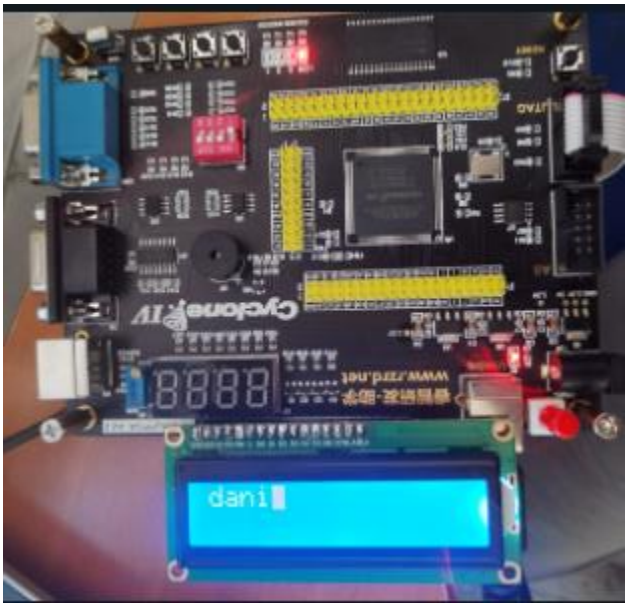
```
18  architecture logica of LCD is
19
20      -- Señales para transicion de estados
21      type Estados is (E0, E1, E2, E3, E4, E5, E6, E7, E8, E9, E10, E11, E12, E13, E14, E15,
22      signal Estado_pre, Estado_fut: Estados := E0;       --Ponemos en estado inicial la maqu
23
24      -- Señal para division de reloj
25      signal count: integer range 0 to (2500000 * 2 - 1);
26
27      -- Señal para transicion de palabra en funcion de botones presionados
28      type array_4_t is array(0 to 3) of std_logic_vector(7 downto 0);
29      type nombres_t is array(0 to 3) of array_4_t;
30
31      constant nombres : nombres_t :=
32      (
33          0 => (0 => X"4F", 1 => X"6D", 2 => X"61", 3 => X"72"), -- Omar
34          1 => (0 => X"44", 1 => X"61", 2 => X"6E", 3 => X"69"), -- Dani
35          2 => (0 => X"4A", 1 => X"6F", 2 => X"65", 3 => X"6C"), -- Joel
36          3 => (0 => X"4A", 1 => X"6F", 2 => X"73", 3 => X"73")  -- Joss
37      );
38      signal nombres_index: integer range 0 to 3;
```

```
40  begin
41      LEDS <= not Botones;
42      RW <= '0';
43      nombres_index <= to_integer(unsigned(Botones));
44
45  ----------------------------------------------------------------------------
46          -- DIVISION DE RELOJ PARA RETARDO ENTRE TRANSICIONES DE SM --
47  ----------------------------------------------------------------------------
48      process (CLK)
49      begin
50          if rising_edge(CLK) then
51              if count = (2500000 * 2 - 1) then
52                  Estado_pre <= Estado_fut;
53                  count <= 0;
54              else
55                  count <= count + 1;
56              end if;
57          end if;
58      end process;
```

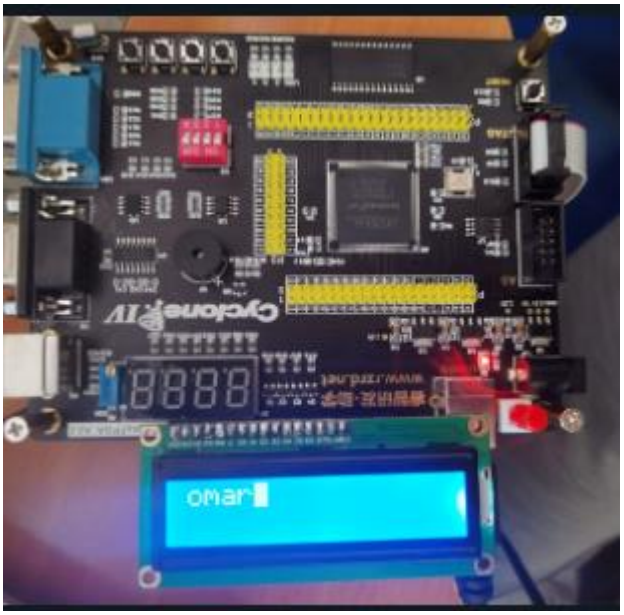
```
60      process (Estado_pre)
61      begin
62          case Estado_pre is
63              -- Configuración del LCD
64              when E0 =>
65                  Datos <= X"38"; -- Set(dato, matriz)
66                  E <= '1'; -- Sube señal
67                  RS <= '0'; -- Configuración del RS
68                  Estado_fut <= E1;
69
70              when E1 =>
71                  E <= '0'; -- Baja la señal
72                  Estado_fut <= E2;
73
74              -- Limpiado del LCD
75              when E2 =>
76                  Datos <= X"01"; -- Clear LCD
77                  E <= '1';
78                  RS <= '0';
79                  Estado_fut <= E3;
80
81              when E3 =>
82                  E <= '0'; -- Baja la señal
83                  Estado_fut <= E4;
```

```
125             when E11 =>
126                 E <= '0'; -- Baja la señal
127                 Estado_fut <= E12;
128
129             -- Caracter 4
130             when E12 =>
131                 Datos <= nombres(nombres_index)(3); -- Letra R en ASCII y en hexadecimal
132                 E <= '1';
133                 RS <= '1';
134                 Estado_fut <= E13;
135
136             when E13 =>
137                 E <= '0'; -- Baja la señal
138                 Estado_fut <= IDLE;
139
140             when IDLE =>        -- Estado inactivo cuando esperando boton de start
141                 if ST = '1' then
142                     Estado_fut <= E0;
143                 else
144                     Estado_fut <= IDLE;
145                 end if;
146             when others => null;
147         end case;
148     end process;
149
150 end logica;
```

Hardware of FPGA with LCD




joss


omar

Conclusion

The use and management of the LCD on the FPGA proved to be an effective task for understanding the interaction between hardware and software through a state machine. The implementation allowed precise control and updating of the display based on clock and button inputs. This type of integration facilitates interaction with user interfaces, displaying relevant information in real-time. Programming the FPGA to control the LCD not only strengthens knowledge of digital circuits but is also essential for applications in embedded projects and control systems.