An-Najah National University

Faculty of Engineering and IT

جامعة النجاح الوطنية

كلية الهندسة وتكنولوجيا المعلومات

Digital Circuit Design2 Lab

Dr. Ashraf Armoush

Wednesday 2:00 pm – 5:00 pm

First Semester

Experiment Information		
Experiment Name: LCD Driver	Experiment Number: #5	
Performed: 11 Nov 2020	Submitted: 19 Nov 2020	
Partner Students		
1- Taher Anaya	2- Hadi Jml	3 – Ameed Omar

Introduction:

An LCD is integrated into the Spartan 3E kit to display any text/numeric information. And to be able to use the LCD, we need to get our hands on user guide of the Sparten 3E starter kit. The user guide holds a lot of important information regarding the timing and about how to use it in general.

Objectives:

- Learning how to write a drive using VHDL.
- Using the integrated LCD in the Spartan 3E kit.
- Displaying our name on the LCD.
- Displaying a counter on the LCD.
- Displaying our name as animated text on the LCD.

Procedure:

This experiment has three parts:

- Displaying our name on the LCD.
- Displaying a counter on the LCD.
- Displaying our name as animated text on the LCD.

But there are two important steps we need to do before we can display anything on the LCD. First, we need to clear it. Then, we need to configure it.

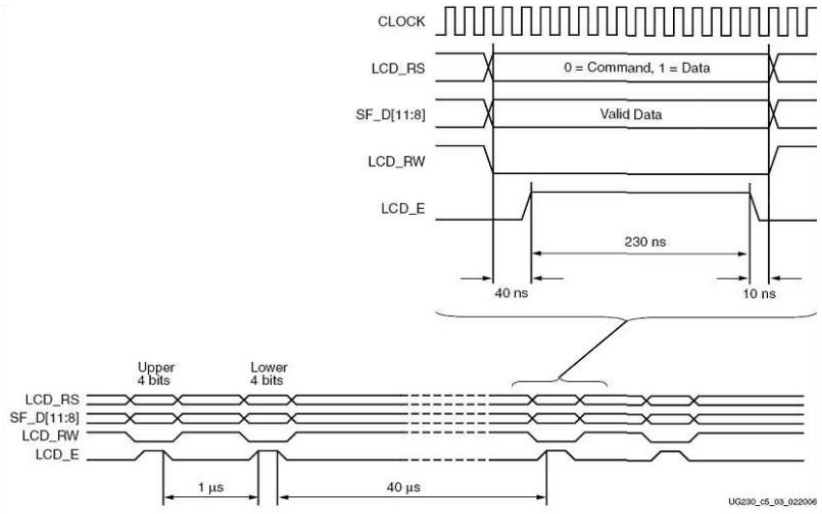
We also need to understand how does the LCD work and how does it accept commands. All of this information is mentioned in the user guide of the Spartan 3E starter kit.

Procedure: *(cont.)*

Clearing the LCD Display

The first step we must do is to clear the LCD Display from the initial written text. And to do that, we need to send a set of commands to the LCD with a prespecified time delay.

There's a general timing diagram (in the user guide) showing how to send commands and the prespecified time delays. The diagram is very important because we can't use the LCD without knowing any of the information it holds.



We can conclude from this diagram that to send a certain command, we need to do the following (assuming that our kit runs at 50MHz frequency):

1. Set the RS, RW, and first 4 bits of the data value, then wait for 2 clock cycles.
2. Set the enable value to 1 for 12 clock cycles, then again to 0, and wait for 50 clock cycles.
3. Set the last 4 bits of the data value and wait for 2 clock cycles.
4. Set the enable value to 1 for 12 clock cycles, then again to 0, and wait for 1 clock cycle.
5. Wait for 2000 clock cycles so the LCD can accept other commands and repeat.

Procedure: *(cont.)*

After understanding the timing diagram and summarizing it into clear steps, we can now perform our first task, which clears the LCD from the initial written text, so we need to send a set of commands.

Note: Since the clearing is a special case, there is a timing difference between it and the regular commands. The clearing commands show in the following figure:

1. Wait 15 ms or longer, although the display is generally ready when the FPGA finishes configuration. The 15 ms interval is 750,000 (B71B0H) clock cycles at 50 MHz
2. Write SF_D<11:8> = 0x3, pulse LCD_E High for 12 (0CH) clock cycles.
3. Wait 4.1 ms or longer, which is 205,000 (320C8H) clock cycles at 50 MHz
4. Write SF_D<11:8> = 0x3, pulse LCD_E High for 12 (0CH) clock cycles.
5. Wait 100 µs or longer, which is 5,000 (1388H) clock cycles at 50 MHz
6. Write SF_D<11:8> = 0x3, pulse LCD_E High for 12 (0CH) clock cycles.
7. Wait 40 µs or longer, which is 2,000 (7D0H) clock cycles at 50 MHz
8. Write SF_D<11:8> = 0x2, pulse LCD_E High for 12 (0CH) clock cycles.
9. Wait 40 µs or longer, which is 2,000 (7D0H) clock cycles at 50 MHz

Now, all we have to do follow the attached commands in the figure and turn it into a VHDL code to clear the LCD.

Our VHDL entity code will seem like this:

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  entity LCDDriverTest is
4      Port ( CLK : in  STD_LOGIC;
5             Data : out  STD_LOGIC_VECTOR (3 downto 0);
6             E : out  STD_LOGIC;
7             RW : out  STD_LOGIC;
8             RS : out  STD_LOGIC);
9  end LCDDriverTest;
```

Procedure: *(cont.)*

And our VHDL code for clearing the LCD will seem like this:

```
11  architecture Behavioral of LCDDriverTest is
12
13  begin
14
15     p1: process ( CLK )
16        variable flag, counter: integer := 0;
17     begin
18        if ( CLK'event and CLK = '1' ) then
19           if ( flag = 0 ) then -- First Step: Clearing The LCD Display
20              counter := counter + 1;
21
22              if ( counter = 750000 ) then
23                 RS <= '0';
24                 RW <= '0';
25                 Data <= "0011";
26              elsif ( counter = (750000 + 2) )then
27                 E <= '1';
28              elsif ( counter = (750000 + 2 + 12) )then
29                 E <= '0';
30
31              elsif ( counter = (750000 + 2 + 12 + 205000) ) then
32                 Data <= "0011";
33              elsif ( counter = (750000 + 2 + 12 + 205000 + 2) )then
34                 E <= '1';
35              elsif ( counter = (750000 + 2 + 12 + 205000 + 2 + 12) )then
36                 E <= '0';
37
38              elsif ( counter = (750000 + 2 + 12 + 205000 + 2 + 12 + 5000) ) then
39                 Data <= "0011";
40              elsif ( counter = (750000 + 2 + 12 + 205000 + 2 + 12 + 5000 + 2) )then
41                 E <= '1';
42              elsif ( counter = (750000 + 2 + 12 + 205000 + 2 + 12 + 5000 + 2 + 12) )then
43                 E <= '0';
44
45              elsif ( counter = (750000 + 2 + 12 + 205000 + 2 + 12 + 5000 + 2 + 12 + 2000) ) then
46                 Data <= "0010";
47              elsif ( counter = (750000 + 2 + 12 + 205000 + 2 + 12 + 5000 + 2 + 12 + 2000 + 2) )then
48                 E <= '1';
49              elsif ( counter = (750000 + 2 + 12 + 205000 + 2 + 12 + 5000 + 2 + 12 + 2000 + 2 + 12) )then
50                 E <= '0';
51              elsif ( counter = (750000 + 2 + 12 + 205000 + 2 + 12 + 5000 + 2 + 12 + 2000 + 2 + 12 + 2000) )then
52                 flag := 1;
53                 counter := 0;
54              end if;
```

Note: Our code structure works in different steps. So, we set a Flag that can take us from one stage to another. If the flag was set to 0, that means we are in the clearing process. After we are done, the flag becomes 1, and the counter is reset, which means that we are done with our clearing and ready to move on to the next step, which is the configuration part.

Now, our display is clean and ready to move on to the next step, which is the configuration part.

Procedure: *(cont.)*

Configuration the LCD

After clearing our LCD, we need to configure it so it can be ready to receive data from the user.

To configure the LCD, we need to follow the given steps:

1. Issue a Function Set command, 0x28, to configure the display for operation on the Spartan-3E Starter Kit board.
2. Issue an Entry Mode Set command, 0x06, to set the display to automatically increment the address pointer.
3. Issue a Display On/Off command, 0x0C, to turn the display on and disables the cursor and blinking.
4. Finally, issue a Clear Display command. Allow at least 1.64 ms (82,000 clock cycles) after issuing this command.
10. The third and last step involves the actual process of writing data to the DD-RAM.

The steps are requiring us to send a different set of commands. There's a table in the user guide that tells us more about the commands and how to deal with it.

Function	LCD_RS	LCD_RW	Upper Nibble				Lower Nibble			
			DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Function Set	0	0	0	0	1	0	1	0	-	-
Set CG RAM Address	0	0	0	1	A5	A4	A3	A2	A1	A0
Set DD RAM Address	0	0	1	A6	A5	A4	A3	A2	A1	A0
Read Busy Flag and Address	0	1	BF	A6	A5	A4	A3	A2	A1	A0
Write Data to CG RAM or DD RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0
Read Data from CG RAM or DD RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0
Clear Display	0	0	0	0	0	0	0	0	0	1
Return Cursor Home	0	0	0	0	0	0	0	0	1	-
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S
Display On/Off	0	0	0	0	0	0	1	D	C	B
Cursor and Display Shift	0	0	0	0	0	1	S/C	R/L	-	-

Now, all we have to do follow the attached commands in the figure and turn it into a VHDL code to configure the LCD.

Procedure: *(cont.)*

```
55
56      elsif ( flag = 1 ) then
57         counter := counter + 1;
58
59         if ( counter = 1) then
60            RS <= '0';
61            RW <= '0';
62            Data <= "0010";
63         elsif ( counter = 3 )then
64            E <= '1';
65         elsif ( counter = 15 )then
66            E <= '0';
67         elsif ( counter = 65) then
68            Data <= "1000";
69         elsif ( counter = 67 )then
70            E <= '1';
71         elsif ( counter = 82 )then
72            E <= '0';
73
74         elsif ( counter = 2082) then
75            Data <= "0000";
76         elsif ( counter = 2084 )then
77            E <= '1';
78         elsif ( counter = 2096 )then
79            E <= '0';
80         elsif ( counter = 2146 )then
81            Data <= "0110";
82         elsif ( counter = 2148 )then
83            E <= '1';
84         elsif ( counter = 2160 )then
85            E <= '0';
```

```
86
87         elsif ( counter = 4160) then
88            Data <= "0000";
89         elsif ( counter = 4162 )then
90            E <= '1';
91         elsif ( counter = 4174 )then
92            E <= '0';
93         elsif ( counter = 4224) then
94            Data <= "1100";
95         elsif ( counter = 4226 )then
96            E <= '1';
97         elsif ( counter = 4238 )then
98            E <= '0';
99
100        elsif ( counter = 6238) then
101           Data <= "0000";
102        elsif ( counter = 6240 )then
103           E <= '1';
104        elsif ( counter = 6252 )then
105           E <= '0';
106        elsif ( counter = 6302) then
107           Data <= "0001";
108        elsif ( counter = 6304 )then
109           E <= '1';
110        elsif ( counter = 6316 )then
111           E <= '0';
112        elsif ( counter = 88316 )then
113           flag := 2;
114           counter := 0;
115        end if;
116
```

Now our LCD is configured to receive a text. So, we set our flag to 2 and reset our counter to move on to the third step, which is the LCD Text display/Counter display/ Animated text Display.

The first two steps will repeat for all parts, so I will not mention them again in this report.

Procedure: *(cont.)*

LCD Text Display (Static Text)

In this part, we will display our name on the LCD after clearing/configuration. In our case, we displayed "TAHERO".

To display our text, we first need to set our DD Ram address to tell the LCD where we will start displaying our names. Then we will start sending the letters one by one in ASCII code using the following table:

Figure 1: LCD Character Set

Now, all we have to do turn every letter in our name into the corresponding binary code, then typing our VHDL code.

Procedure: *(cont.)*

```
116
117       elsif ( flag = 2 ) then
118          counter := counter + 1;
119
120          -- This Command To Set The Location
121          -- That We Want To Start Typing From It.
122          if ( counter = 1) then
123             Data <= "1000";
124          elsif ( counter = 3 )then
125             E <= '1';
126          elsif ( counter = 15 )then
127             E <= '0';
128          elsif ( counter = 65) then
129             Data <= "0000";
130          elsif ( counter = 67 )then
131             E <= '1';
132          elsif ( counter = 82 )then
133             E <= '0';
134
135          --  T Character
136          elsif ( counter = 2082) then
137             RS <= '1';
138             Data <= "0101";
139          elsif ( counter = 2084 )then
140             E <= '1';
141          elsif ( counter = 2096 )then
142             E <= '0';
143          elsif ( counter = 2146 )then
144             Data <= "0100";
145          elsif ( counter = 2148 )then
146             E <= '1';
147          elsif ( counter = 2160 )then
148             E <= '0';
```

```
150          --  A Character
151          elsif ( counter = 4160) then
152             Data <= "0100";
153          elsif ( counter = 4162 )then
154             E <= '1';
155          elsif ( counter = 4174 )then
156             E <= '0';
157          elsif ( counter = 4224) then
158             Data <= "0001";
159          elsif ( counter = 4226 )then
160             E <= '1';
161          elsif ( counter = 4238 )then
162             E <= '0';
163
164          --  H Character
165          elsif ( counter = 6238) then
166             Data <= "0100";
167          elsif ( counter = 6240 )then
168             E <= '1';
169          elsif ( counter = 6252 )then
170             E <= '0';
171          elsif ( counter = 6302) then
172             Data <= "1000";
173          elsif ( counter = 6304 )then
174             E <= '1';
175          elsif ( counter = 6316 )then
176             E <= '0';
```

```
177
178          --  E Character
179          elsif ( counter = 8316) then
180             Data <= "0100";
181          elsif ( counter = 8318 )then
182             E <= '1';
183          elsif ( counter = 8330 )then
184             E <= '0';
185          elsif ( counter = 8380) then
186             Data <= "0101";
187          elsif ( counter = 8382 )then
188             E <= '1';
189          elsif ( counter = 8394 )then
190             E <= '0';
191
192          --  R Character
193          elsif ( counter = 10394) then
194             Data <= "0101";
195          elsif ( counter = 10396 )then
196             E <= '1';
197          elsif ( counter = 10408 )then
198             E <= '0';
199          elsif ( counter = 10458) then
200             Data <= "0010";
201          elsif ( counter = 10460 )then
202             E <= '1';
203          elsif ( counter = 10472 )then
204             E <= '0';
```

```
205
206          --  O Character
207          elsif ( counter = 12472) then
208             Data <= "0100";
209          elsif ( counter = 12474 )then
210             E <= '1';
211          elsif ( counter = 12486 )then
212             E <= '0';
213          elsif ( counter = 12536) then
214             Data <= "1111";
215          elsif ( counter = 12538 )then
216             E <= '1';
217          elsif ( counter = 12550 )then
218             E <= '0';
219          end if;
220       end if;
221    end if;
222 end process;
223
224 end Behavioral;
```

Procedure: *(cont.)*

Now our VHDL code becomes ready. As a final step, we need to type our constraint file, which will be the same for all parts:

```
NET "CLK" LOC = C9;
NET "E" LOC = M18;
NET "RS" LOC = L18;
NET "RW" LOC = L17;
NET "Data[3]" LOC = M15;
NET "Data[2]" LOC = P17;
NET "Data[1]" LOC = R16;
NET "Data[0]" LOC = R15;
```

Up to here, we will have finished the first part of the experiment, and now let's move on to the second part, which is LCD Counter Display.

LCD Counter Display (BCD Counter)

After clearing/configuring the display, we can start showing our 00 – 99 counter.
In the previously shown ASCII code table, we noticed that the numbers are consecutive, which will make our code a lot easier.

We need to initialize one signal for every Digit:

```
14
15    Signal FirstDigit : STD_LOGIC_VECTOR (3 downto 0) := "0000";
16    Signal SecondDigit: STD_LOGIC_VECTOR (3 downto 0) := "0000";
17
```

And, we also need another counter and a flag that informs us that 1 second has passed:

```
23
24    variable checkSecond: integer := 0;
25    variable secondPassed: STD_LOGIC := '0';
26
```

Procedure: *(cont.)*

Now we can start typing our VHDL code.
Note: The clearing and the configuration steps are the same as the ones from the first part.

```
140
141     if ( secondPassed = '1' ) then
142
143        -- This Command To Set The Location
144        -- That We Want To Start Typing From It.
145        if ( counter = 1) then
146           Rs <= '0';
147           RW <= '0';
148           Data <= "1000";
149        elsif ( counter = 3 )then
150           E <= '1';
151        elsif ( counter = 15 )then
152           E <= '0';
153        elsif ( counter = 65) then
154           Data <= "1000";
155        elsif ( counter = 67 )then
156           E <= '1';
157        elsif ( counter = 82 )then
158           E <= '0';
159
160        -- This Command To Show
161        -- The Second Digit On The Screen.
162        elsif ( counter = 2082) then
163           RS <= '1';
164           Data <= "0011";
165        elsif ( counter = 2084 )then
166           E <= '1';
167        elsif ( counter = 2096 )then
168           E <= '0';
169        elsif ( counter = 2146 )then
170           Data <= SecondDigit;
171        elsif ( counter = 2148 )then
172           E <= '1';
173        elsif ( counter = 2160 )then
174           E <= '0';
175
```

```
175
176        -- This Command To Show
177        -- The First Digit On The Screen.
178        elsif ( counter = 4160) then
179           Data <= "0011";
180        elsif ( counter = 4162 )then
181           E <= '1';
182        elsif ( counter = 4174 )then
183           E <= '0';
184        elsif ( counter = 4224) then
185           Data <= FirstDigit;
186        elsif ( counter = 4226 )then
187           E <= '1';
188        elsif ( counter = 4238 )then
189           E <= '0';
190
191           -- After We Show The number On Display
192           -- Now We Want To Increment It By One.
193           elsif ( counter = 4240) then
194              secondPassed := '0';
195              if (FirstDigit = "1001") then
196
197                 FirstDigit <= "0000";
198                 if (SecondDigit = "1001") then
199                    SecondDigit <= "0000";
200                 else
201                    SecondDigit <= SecondDigit + 1;
202                 end if;
203              else
204                 FirstDigit <= FirstDigit + 1;
205              end if;
206
```

Note: the counter is displayed in the middle of the screen (start at location 8), and whenever a second passed, we reset our DD Ram address value again to the middle of the screen.

Up to here, we will have finished the second part of the experiment, and now let's move on to the third (Bonus) part, which is LCD Animated Text.

Procedure: *(cont.)*

LCD Animated Text Display (Bonus Part)

After clearing/configuring the display, we can start showing our animated text.
In this part, we want to scroll the text - that we showed on the screen in the first part- along the LCD screen.

We need to initialize shiftFlag signal:

```
15
16      signal shiftFlag: STD_LOGIC:= '0';
17
```

Now we can start typing our VHDL code.
Note: The clearing and the configuration steps are the same as the ones from the first part.

```
124
125  elsif ( flag = 2 ) then
126
127     counter := counter + 1;
128     if (shiftFlag = '0')  then
129
130     -- This command To Set The Location
131     -- That We Want to Start Typing From It.
132        if ( counter = 1) then
133           Data <= "1001";
134        elsif ( counter = 3 )then
135           E <= '1';
136        elsif ( counter = 15 )then
137           E <= '0';
138        elsif ( counter = 65) then
139           Data <= "0000";
140        elsif ( counter = 67 )then
141           E <= '1';
142        elsif ( counter = 82 )then
143           E <= '0';
144
145        -- T Character
146        elsif ( counter = 2082) then
147           RS <= '1';
148           Data <= "0101";
149        elsif ( counter = 2084 )then
150           E <= '1';
151        elsif ( counter = 2096 )then
152           E <= '0';
153        elsif ( counter = 2146 )then
154           Data <= "0100";
155        elsif ( counter = 2148 )then
156           E <= '1';
157        elsif ( counter = 2160 )then
158           E <= '0';
159
```

```
159
160     -- A Character
161     elsif ( counter = 4160) then
162        Data <= "0100";
163     elsif ( counter = 4162 )then
164        E <= '1';
165     elsif ( counter = 4174 )then
166        E <= '0';
167     elsif ( counter = 4224) then
168        Data <= "0001";
169     elsif ( counter = 4226 )then
170        E <= '1';
171     elsif ( counter = 4238 )then
172        E <= '0';
173
174     -- H Character
175     elsif ( counter = 6238) then
176        Data <= "0100";
177     elsif ( counter = 6240 )then
178        E <= '1';
179     elsif ( counter = 6252 )then
180        E <= '0';
181     elsif ( counter = 6302) then
182        Data <= "1000";
183     elsif ( counter = 6304 )then
184        E <= '1';
185     elsif ( counter = 6316 )then
186        E <= '0';
187
```

Procedure: *(cont.)*

```
187
188     -- E Character
189     elsif ( counter = 8316) then
190        Data <= "0100";
191     elsif ( counter = 8318 )then
192        E <= '1';
193     elsif ( counter = 8330 )then
194        E <= '0';
195     elsif ( counter = 8380) then
196        Data <= "0101";
197     elsif ( counter = 8382 )then
198        E <= '1';
199     elsif ( counter = 8394 )then
200        E <= '0';
201
202     -- R Character
203     elsif ( counter = 10394) then
204        Data <= "0101";
205     elsif ( counter = 10396 )then
206        E <= '1';
207     elsif ( counter = 10408 )then
208        E <= '0';
209     elsif ( counter = 10458) then
210        Data <= "0010";
211     elsif ( counter = 10460 )then
212        E <= '1';
213     elsif ( counter = 10472 )then
214        E <= '0';
215
```

```
215
216     -- O Character
217     elsif ( counter = 12472) then
218        Data <= "0100";
219     elsif ( counter = 12474 )then
220        E <= '1';
221     elsif ( counter = 12486 )then
222        E <= '0';
223     elsif ( counter = 12536) then
224        Data <= "1111";
225     elsif ( counter = 12538 )then
226        E <= '1';
227     elsif ( counter = 12550 )then
228        E <= '0';
229        counter := 0;
230        shiftFlag <= '1';
231     end if;
232
233  elsif ( shiftFlag = '1' ) then
234     RS <= '0';
235     counter := counter + 1;
236     if ( counter = 50000000 ) then
237        Data <= "0001";
238        resetFlag := resetFlag + 1;
239        if ( resetFlag = 22 ) then
240           shiftFlag <= '0';
241           resetFlag := 0;
242        end if;
243     elsif ( counter = 50000002 ) then
244        E <= '1';
245     elsif ( counter = 50000014 ) then
246        E <= '0';
247     elsif ( counter = 50000064 ) then
248        Data <= "1000";
249     elsif ( counter = 50000066 ) then
250        E <= '1';
251     elsif ( counter = 50000078 ) then
252        E <= '0';
253     elsif ( counter = 50000080 ) then
254        counter := 0;
255     end if;
256  end if;
```

Conclusion:

In the end, we learned how to use the LCD integrated with the Spartan 3E kit. Additionally, I learned a lot about the timing and how to write a proper driver, which will help us further experiments.