ELEX 2117 : Digital Techniques 2
2020 Fall Term

Combinational Logic Design with Verilog

Please read and follow the course Verilog Coding Guidelines when writing your solution.

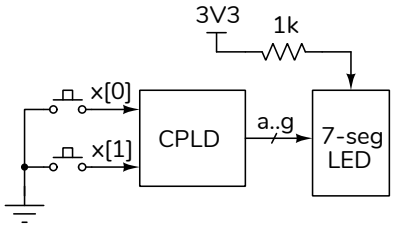
Introduction

This lab introduces logic synthesis using Verilog. You will need the following from your ELEX 1117 parts kit:

- solderless breadboard
- EPM240T100C5 CPLD board, Byte Blaster JTAG interface and coaxial power connector,
- a 7-segment LED
- a 1 kΩ resistor
- two SPDT pushbutton switches and machine-pin DIP socket for mounting the switches (two SPST N.O. switches will also work)
- 11 pin-header jumpers

Circuit Description

The following block diagram shows how the pushbutton switches and LED are connected to the CPLD:



The two pushbuttons, `x[0]` and `x[1]`, are the inputs to your design. The seven LED segments, labelled `a` through `g` are the outputs. All signals are active-low[1]. The two inputs will be configured with internal pull-up resistors.

Your circuit should display the last four digits of *your* BCIT ID when the switch inputs, treated as a binary number, have values of 0 through 3. For example, if you BCIT ID is A00123456 then when both switches are false (not pressed) the LED should show 3 and when `x[0]` is active (low) and `x[1]` is not true (high) then the LED should show 4, etc. The truth table defining the input-to-output relationship for an ID of A00123456 is:

`x[1]`	`x[0]`	LED
0	0	3
0	1	4
1	0	5
1	1	6

Note that the table above shows the *logical* values of `x`. This means that a `1` indicates "true" which corresponds to a *low* logic level. Verilog uses the values 0 and 1 for low and high logic levels respectively when specifying input and output values. But it also uses 0 and 1 for arithmetic and truth values. This can be confusing so you need to pay attention to the context.

CPLD I/O

The tables and photos below show how the CPLD can be connected to the switches and the LED.

The internal pull-up resistor and the N.O. (normally open) pushbutton switch contacts will result in a low logic level when a button is pressed. Mount the pushbuttons on a machined-pin DIP socket to secure it.

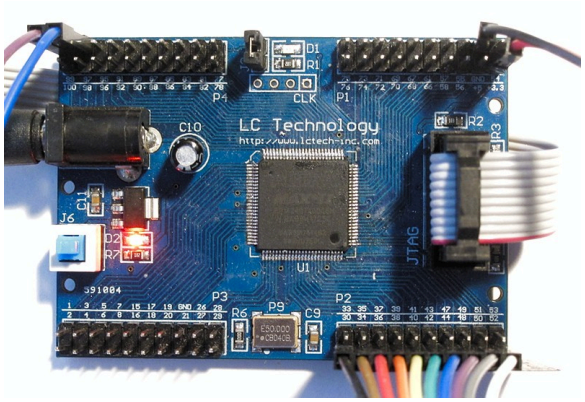
I connected the LED segments to the CPLD using the bottom row of pins on connector P2:

wire color	CPLD pin	segment	LED pin
black (0)	30	`e`	1
brown (1)	34	`d`	2
red (2)	36	`c`	4
green (5)	42	`b`	6
blue (6)	44	`a`	7
violet (7)	48	`f`	9
gray (8)	50	`g`	10

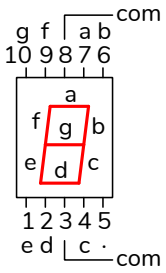
The pushbutton switches are connected to pins 99 and 97 on connector P4 at the top left of the CPLD board:

[1] Active-low means an input or output is at a low logic level when it's true. We will consider a switch to be "true" when pressed and an LED segment is "true" when lit.

wire color	CPLD pin	switch
violet (7)	99	`x[1]`
blue (6)	97	`x[0]`

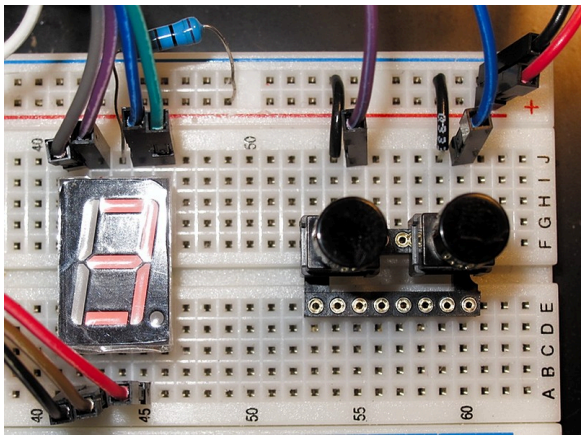


The conventional segment labelling and the pinouts for the LED in your parts kit is shown below:



The display in your parts kit[2] has a `com`mon anode. This pin should be connected to the 3.3 V supply (available on pin labelled +3.3 at the top right of the CPLD board) through a 1 kΩ resistor[3].

The connections to the switches and LED on my breadboard are shown below.



[2]Datasheet on course web site.

[3]Typically a current-limiting resistor is used on each segment to obtain the same brightness regardless of the number of segments that are lit. But for our purposes one resistor will do.

Procedure

Follow the general procedure in Appendix A to create a project, compile it and configure your CPLD. Connect the CPLD board to the switches and LED. Test your design and fix any errors.

Internal Pull-Up Resistors

In step 6 you'll also need to configure internal pull-up resistors on the two input pins. Open the Assignment Editor (**Assignments** > **Assignment Editor**). Double-click on «**new**» in the **To** column and enter the pin name (`x[0]`). Select **Weak Pull-Up Resistor** from the drop-down menu in the **Assignment Name** column. Select **On** from the drop-down menu in the **Value** column. Repeat for `x[1]`.

If you used the pin assignments above you should end up with the following:

From	To	Assignment Name	Value	Enabled	Entity
	a	Location	PIN_44	Yes	
	b	Location	PIN_42	Yes	
	c	Location	PIN_36	Yes	
	d	Location	PIN_34	Yes	
	e	Location	PIN_30	Yes	
	f	Location	PIN_48	Yes	
	g	Location	PIN_50	Yes	
	x[1]	Location	PIN_99	Yes	
	x[0]	Location	PIN_97	Yes	
	x[0]	Weak Pull-Up Resistor	On	Yes	lab1
	x[1]	Weak Pull-Up Resistor	On	Yes	lab1
<<...>>	<<...>>	<<new>>			

Hints

It might be helpful to prepare a table showing the switch input levels and the segment output levels. For the example above:

x[1]	x[0]	digit	a	b	c	d	e	f	g	hex
1	1	3	0	0	0	0	1	1	0	06
1	0	4	1	0	0	1	1	0	0	4c
0	1	5	0	1	0	0	1	0	0	24
0	0	6	0	1	0	0	0	0	0	20

Verilog has a concatenation operator (`{,}`) that can be used in the target of an assignment as well as part of an expression. For example, `assign {a,b,c} = 3'b011 ;` assigns `0` to `a` and `1` to `b` and `c`.

Follow the Verilog Coding Guidelines as found on the course website.

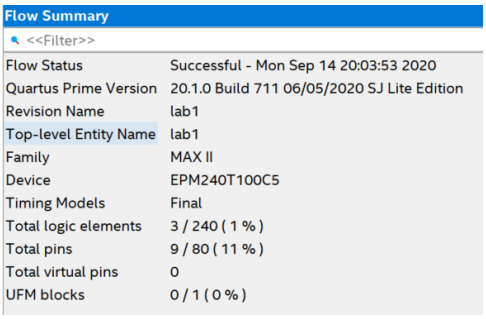
Submissions

To get credit for completing this lab, submit the following to the Assignment folder for Lab 1 on the course website:

1. A PDF document containing:

   - Your name, BCIT ID, course number and lab number.
   - A listing of your Verilog code. You must follow the coding guidelines given on the "Course Information" section of the course website.

     The listings should be easy to read. For example, follow one of the suggestions in the document on the course web site under **Resources** > **Other** > **Including Code in Reports**.
   - a screen capture of your compilation report (**Window** > **Compilation Report**) similar to this:



Flow Summary	
<<Filter>>	
Flow Status	Successful - Mon Sep 14 20:03:53 2020
Quartus Prime Version	20.1.0 Build 711 06/05/2020 SJ Lite Edition
Revision Name	lab1
Top-level Entity Name	lab1
Family	MAX II
Device	EPM240T100C5
Timing Models	Final
Total logic elements	3 / 240 (1 %)
Total pins	9 / 80 (11 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)

2. A video showing the pushbuttons and the LED display as you test the four pushbutton states. The LED should, of course, show the last four digits of *your* BCIT ID in order. It should take less than 10 seconds to go through all four switch combinations.

A Logic Synthesis with Quartus

1. start Quartus
2. select **File** > **New...** > **New Quartus Prime Project** > **OK**
3. in the dialog boxes that follow: select a new folder for your lab (e.g. `C:\ELEX2117\lab1`), enter a project name (e.g. `lab1`), select an empty project, don't add any files, select the Max II Family, select the specific device `EPM240T100C5`, and leave other settings at their defaults
4. add any existing design files using **Project** > **Add/Remove Files in Project...**, or create new ones using **File** > **New...** > **System Verilog HDL File**. One of the files must have a module with the top-level name specified above (in this example, `lab1`). The port names of this module will correspond to the names assigned to the CPLD's pins.
5. after all the design files have been created and added to the project, select **Processing** > **Start Compilation**. Correct any errors and recompile as necessary
6. select **Assignments** > **Pin Planner** and select the correct pin in the **Location** drop-down box for each I/O pin. Note that you must compile the project before the pin names are visible in Pin Planner. Recompile the project (**Processing** > **Start Compilation**) for the assignments to take effect.
7. connect the CPLD board's coaxial power connector to a USB port and use the pushbutton on the board to turn on the power (the power LED should go on). Connect the "USB Blaster" to the CPLD and a free USB port. The POWER and ACT lights on the USB-Blaster should turn on.
8. select **Tools** > **Programmer**, click on **Hardware Setup...**, select **USB-Blaster** from the drop-down and **Close**. **USB-Blaster** should appear next to **Hardware Setup...**
9. if necessary, click on **Add File...**, navigate to the location of the generated `.pof` file (typically in the `output_files` folder of the project folder) and select the `.pof` file
10. check that the **Program/Configure** checkboxes are checked and press **Start** to program the device. The progress bar should show 100%.
11. test your design.