ELEX 2117 : Digital Techniques 2
2021 Winter Term

Sequential Logic Design with Verilog

Updated Jan23: description of behaviour in Submission section.

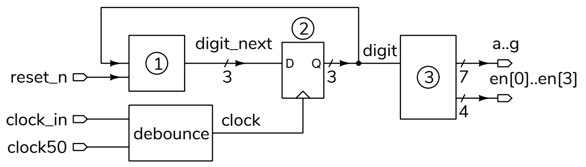
Introduction

In this lab you will implement a circuit that displays successive digits of your BCIT ID on a four-digit 7-segment LED display.

You will use the same components, input/output devices and wiring as the previous lab.

Requirements

Your design should implement the following circuit.



It has two pushbutton switch inputs (**reset_n** and **clock_in**), a 50 MHz clock input (**clock50**), seven active-low segment outputs, (**a** through **g**), and four active-high digit-enables (**en[3]** through **en[0]**).

The functions of the blocks labelled with circled numbers are:

① Is a combinational logic circuit that outputs the next digit position based on the current digit and the active-low **reset_n** input. The function is described below. Digit values have values from 0 to 7 because BCIT ID's have 8 digits. This signal is thus 3 bits wide.

② Is a 3-bit register whose value is the digit position being displayed (between 0 and 7). The register's **clock** signal is the "debounced" **clock_in** input.

③ Is a combinational logic circuit that outputs signals that drive the LED segments and digit enables so that the correct number is displayed for the current digit position.

Your circuit should operate as follows:

- The output should change on the rising edge of the clock (that is, when the **clock_in** pushbutton is *released*).
- If the **reset_n** input is asserted (low) at the rising edge of the clock, the first digit of your BCIT ID should be displayed on the leftmost LED digit.
- If the reset input is not asserted at the rising edge of the clock then the next digit of your BCIT ID should be displayed on the next LED digit position.

  When you reach the rightmost digit position display should "wrap around" and display again on the leftmost digit for the fifth digit of your ID.
- if your ID ends with:
  - an odd number: your design should continue from the first digit (position 0) when last digit is reached,
  - an even number: your design should remain on the last digit (7) when last digit is reached.

Debouncing

Most mechanical switches briefly interrupt the connection when they switch. This "switch bounce" produces multiple signal edges. So you'll need to "debounce" the **clock_in** switch input [1].

Include a module instantiation statement such as:

```
debounce debounce0 ( clock_in, clock50, clock ) ;
```

in your Verilog code where **clock_in** and **clock** are the signal names for the un-debounced and debounced clock signals respectively and **clock50** is the 50 MHz clock on the CPLD board connected to CPLD pin 12.

The module instantiation statement above adds an instance of the **debounce** module to your design with

[1] But not **reset_n** since it's level-sensitive rather than edge-sensitive.

the name **debounce0**, and connects the module inputs and output to the specified signals.

CPLD I/O

The CPLD should be wired up to two pushbutton switches and the four-digit seven-segment LED display as in the previous lab.

Procedure

Follow the general procedure in the Software Installation and Use document on the course web site 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.

Note the following instructions specific to this lab.

Debouncing

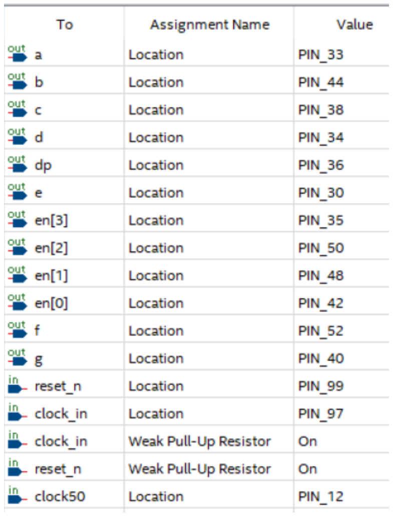
Download **debounce.sv** from the course website to your project folder and add it using **Project > Add/Remove Files in Project...**, select the file, **Add > OK**).

You can use **Assignments > Import Assignments** to import your **lab1.qsf** file instead of doing all the pin assignments manually. However, you'll also need to:

- change the names **x[0]** to **clock_in** and **x[1]** to **reset_n**,
- update the signal and entity names for the pull-up assignments,
- assign pin 12 to the **clock50** 50 MHz oscillator input.

In addition the previous inputs and outputs, your module will also need the 50 MHz clock input.

You should end up with the following assignments, possibly using different signal names:



To	Assignment Name	Value
out a	Location	PIN_33
out b	Location	PIN_44
out c	Location	PIN_38
out d	Location	PIN_34
out dp	Location	PIN_36
out e	Location	PIN_30
out en[3]	Location	PIN_35
out en[2]	Location	PIN_50
out en[1]	Location	PIN_48
out en[0]	Location	PIN_42
out f	Location	PIN_52
out g	Location	PIN_40
in reset_n	Location	PIN_99
in clock_in	Location	PIN_97
in clock_in	Weak Pull-Up Resistor	On
in reset_n	Weak Pull-Up Resistor	On
in clock50	Location	PIN_12

On each release of the pushbutton (rising clock edge) the count value should change to the next digit of your BCIT ID. Holding the reset button and releasing the block button should display the first digit of your BCIT ID. On reaching the last digit of your ID, the display should change (or not) as described above.

Hints

1. Verilog has an addition operator you can use to compute the next digit position. For example, **assign digit_next = digit+1'b1;**.

2. You'll need to define an internal **clock** signal that the debounced version of the **clock_in** input.

3. Examples of the output digit sequences are shown below for the IDs **A00123456** and **A01234567**. The bottom rows show the outputs after the rising edge of the clock ("↑" represents a rising edge of the clock):

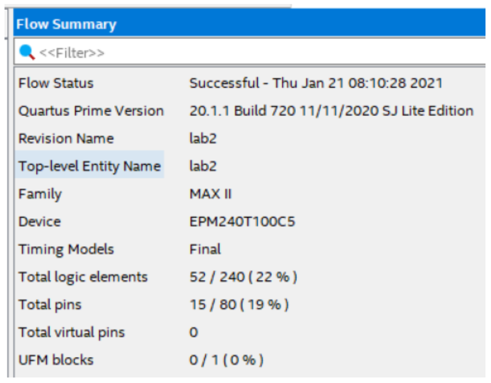
reset_n	L	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	...
clock	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	...
A00123456	0	0	0	1	2	0	0	1	2	3	4	5	6	6	6	6	6	...
A01234567	0	0	1	2	3	0	1	2	3	4	5	6	7	0	1	2	3	...

Submission

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

1. A PDF document containing:

- A listing of your Verilog code.
- A screen capture of your compilation report (**Window** > **Compilation Report**) similar to this:



Flow Summary

<<Filter>>

Flow Status	Successful - Thu Jan 21 08:10:28 2021
Quartus Prime Version	20.1.1 Build 720 11/11/2020 SJ Lite Edition
Revision Name	lab2
Top-level Entity Name	lab2
Family	MAX II
Device	EPM240T100C5
Timing Models	Final
Total logic elements	52 / 240 (22 %)
Total pins	15 / 80 (19 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)

2. The PDF file containing the schematic created by **Tools > Netlist Viewers > RTL Viewer** and then **File > Export...** . The file might look like Figure 1 or Figure 2.

3. A video showing the pushbuttons and the LED display as you test your design by doing the following:

   - push & hold `reset_n`
   - push & release `clock_in` once (shows 0)
   - release `reset_n`
   - push & release `clock_in` three times (shows first three digits)
   - push & hold `reset_n`
   - push & release `clock_in` once (shows 0)
   - release `reset_n`
   - push & release `clock_in` eleven times (shows all digits and either stops at the last digit if it's even or starts again if it's odd)

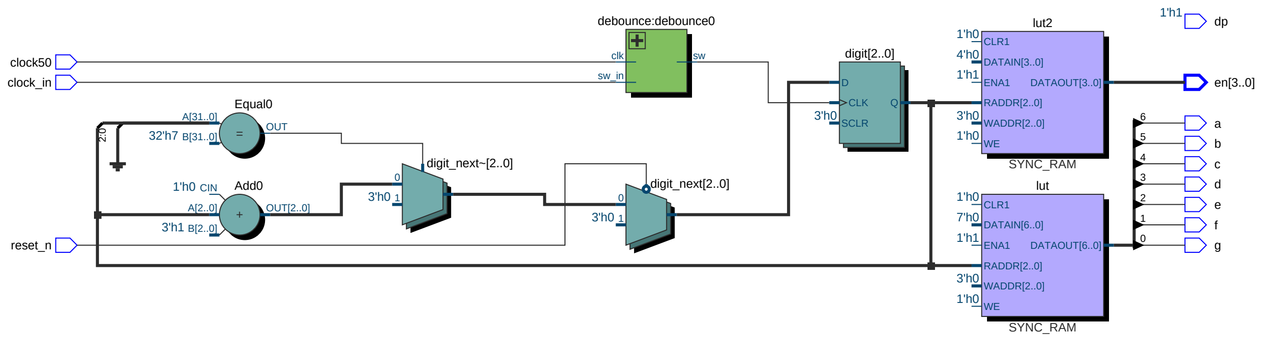


Figure 1: Example RTL Schematic for Lab 2 (look-up table implementation).

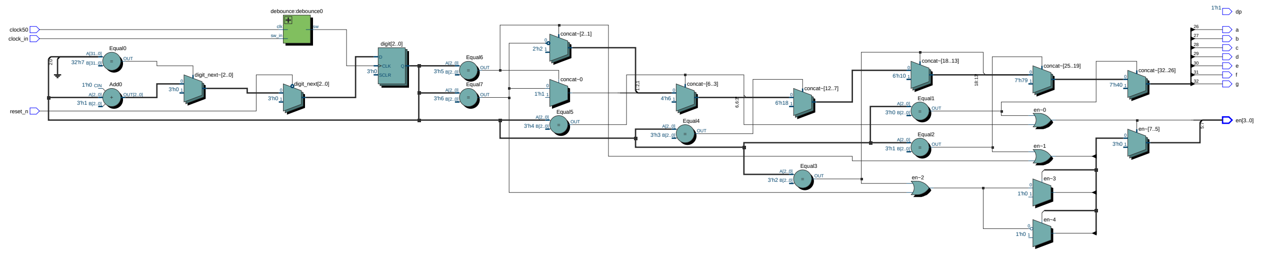


Figure 2: Example RTL Schematic for Lab 2 (multiplexer implementation).