Laboratory Exercise 2 – ECE241 Fall 2014

Switches, Lights, and Multiplexers

The purpose of this exercise is to learn how to connect simple input and output devices to an FPGA chip and implement a circuit that uses these devices. We will use the switches SW_{17-0} and keys KEY_{2-0} on the DE2 board as inputs to the circuit. We will use light emitting diodes (LEDs) and 7-segment displays as output devices.

Preparation Before the Lab

You are required to write the Verilog code for Parts I to IV of the lab. For marking of preparation by the teaching assistants, you are required to show the teaching assistants your Verilog code for Parts II and IV. This code should be printed out and pasted into your lab book. For Parts I to IV of the lab, you must also simulate your circuit with QSim. You are required to show the teaching assistant the simulated timing diagrams for Part II of the lab, printed out and pasted in your lab book.

In-lab Work

You are required to implement and test all of Parts I to V of the lab. You need to demonstrate to the teaching assistants Parts II and V.

Part I

The DE2 board provides 18 toggle switches, called SW_{17-0}, that can be used as inputs to a circuit, and 18 red lights, called $LEDR_{17-0}$, that can be used to display output values. Figure 1 shows a simple Verilog module that uses these switches and shows their states on the LEDs. Since there are 18 switches and lights it is convenient to represent them as vectors in the Verilog code, as shown. We have used a single assignment statement for all 18 *LEDR* outputs, which is equivalent to the individual assignments:

```
assign LEDR[17] = SW[17];
assign LEDR[16] = SW[16];
...
assign LEDR[0] = SW[0];
```

```
// Simple module that connects the SW switches to the LEDR lights
module  part1 (SW, LEDR);
    input [17:0] SW;          // toggle switches
    output [17:0] LEDR;    // red LEDs

    assign LEDR = SW;
endmodule
```

Figure 1: Verilog code that uses the DE2 board switches and lights

The DE2 board has hardwired connections between its FPGA chip and the switches and lights. To use SW_{17-0} and $LEDR_{17-0}$ it is necessary to include in your Quartus II project the correct pin assignments, which are given in the *DE2 User Manual*. For example, the manual specifies that SW_0 is connected to the FPGA pin *N25* and $LEDR_0$ is connected to pin *AE23*. A good way to make the required pin assignments is to import into the Quartus II software the file called *DE2_pin_assignments.csv*, which can be found at

`http://www.eecg.toronto.edu/~pc/courses/241/DE2/DE2_pin_assignments.csv`.

The procedure for making pin assignments is described in the tutorial *Quartus II Introduction (using Verilog Design)*, which you have performed previously (see announcement on course webpage).

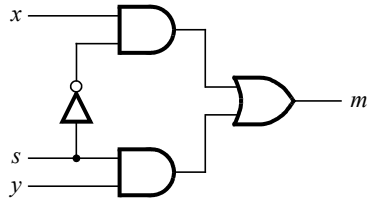
It is important to realize that the pin assignments in the *DE2_pin_assignments.csv* file are useful only if the pin names given in the file are exactly the same as the port names used in your Verilog module. The file uses the names *SW*[0] ... *SW*[17] and *LEDR*[0] ... *LEDR*[17] for the switches and lights, which is the reason we used these names in Figure 1.

Perform the following steps to implement a circuit corresponding to the code in Figure 1 on the DE2 board.

1. Create a new Quartus II project for your circuit. Select Cyclone II EP2C35F672C6 as the target chip, which is the FPGA chip on the Altera DE2 board.
2. Create a Verilog module for the code in Figure 1 and include it in your project.
3. Simulate your circuit with QSim for different values of *SW*. Ensure the output waveforms are correct for the different input values.
4. Include in your project the required pin assignments for the DE2 board, as discussed above. Compile the project.
5. Download the compiled circuit into the FPGA chip. Test the functionality of the circuit by toggling the switches and observing the LEDs.

Part II

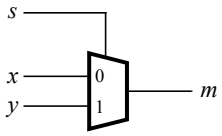
Figure 2a shows a sum-of-products circuit that implements a 2-to-1 *multiplexer* with a select input s. If $s = 0$ the multiplexer's output m is equal to the input x, and if $s = 1$ the output is equal to y. Part b of the figure gives a truth table for this multiplexer, and part c shows its circuit symbol.



a) Circuit

s	m
0	x
1	y

b) Truth table



c) Symbol

Figure 2. A 2-to-1 multiplexer.

The multiplexer can be described by the following Verilog statement:

assign m = (~s & x) | (s & y);

You are to write a Verilog module to describe the circuit given in Figure 3a. This circuit has two eight-bit inputs, X and Y, and produces the eight-bit output M. If $s = 0$ then $M = X$, else if $s = 1$ then $M = \sim (X \mid Y)$. That is, s is used to select either the bitwise X input or bitwise NOR of the 8-bit-wide inputs. A concise view of the circuit is shown in Figure 3b, in which X, Y, and M are depicted as 8-bit-wide wires. Each bit of the output can be described by a Verilog statement of the following form (your Verilog will contain 8 statements like the one shown below):

assign m = (∼s & x) | (s & ∼(x | y));

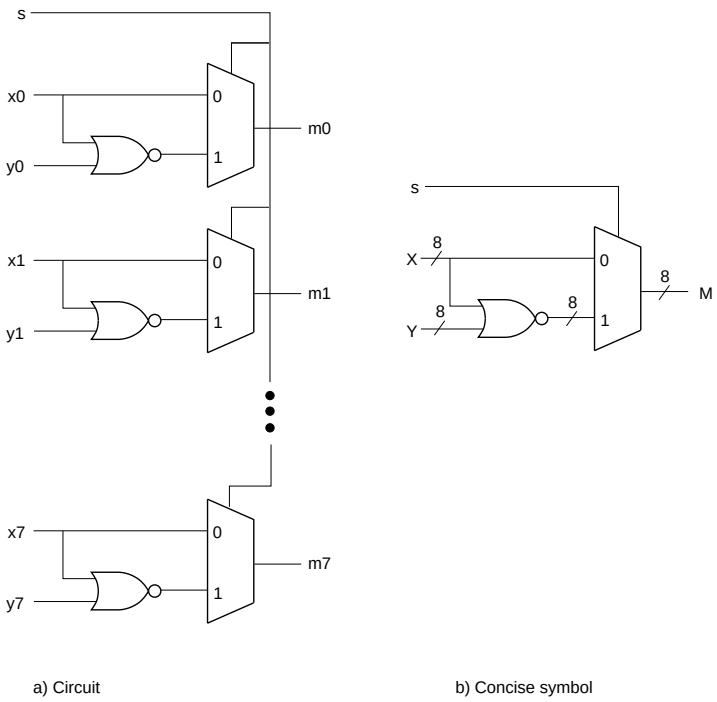


Figure 3. An eight-bit wide 2-to-1 multiplexer selecting either the X input or the NOR of the two inputs.

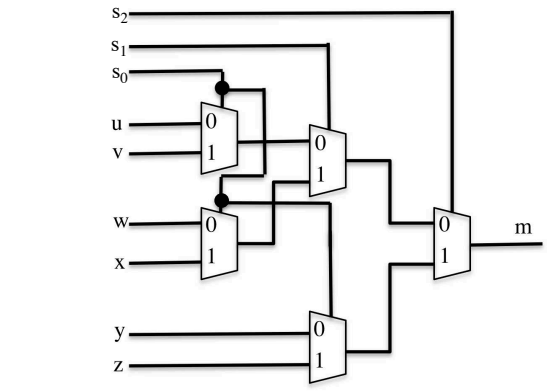
Perform the steps shown below.

1. Create a new Quartus II project for your circuit.
2. Include your Verilog file for the circuit in your project. Use switch SW_{17} on the DE2 board as the s input, switches SW_{7-0} as the X input and SW_{15-8} as the Y input. Connect the *SW* switches to the red lights *LEDR* and connect the output M to the green lights $LEDG_{7-0}$.
3. Simulate your circuit with QSim for different values of s, X, and Y. Print the simulation waveforms and paste them into your lab book. You must show these to the TA as part of your pre-work.
4. Include in your project the required pin assignments for the DE2 board. As discussed in Part I, these assignments ensure that the input ports of your Verilog code will use the pins on the Cyclone II FPGA that are connected to the *SW* switches, and the output ports of your Verilog code will use the FPGA pins connected to the *LEDR* and *LEDG* lights.
5. Compile the project.
6. Download the compiled circuit into the FPGA chip. Test the functionality of the circuit by toggling the switches and observing the LEDs.

Part III

In Figure 2 we showed a 2-to-1 multiplexer that selects between the two inputs *x* and *y*. For this part consider a circuit in which the output *m* has to be selected from six inputs u, v, w, x, y, and z. Part a of Figure 4 shows how we can build the required 6-to-1 multiplexer by using five 2-to-1 multiplexers. The circuit uses a 3-bit select input $s_2 s_1 s_0$ and implements the truth table shown in Figure 4b. A circuit symbol for this multiplexer is given in part c of the figure.

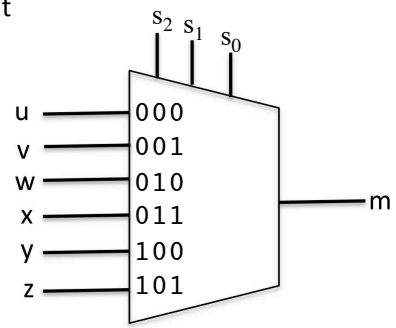
Recall from Figure 3 that an eight-bit wide 2-to-1 multiplexer can be built by using eight instances of a 2-to-1 multiplexer. Figure 5 applies this concept to define a three-bit wide 6-to-1 multiplexer. It contains three instances of the circuit in Figure 4a.



a) circuit

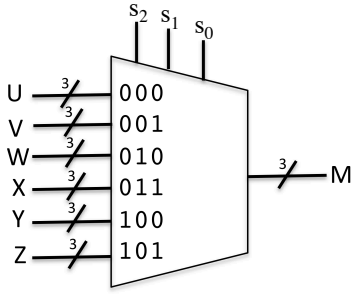
s_2	s_1	s_0	m
0	0	0	u
0	0	1	v
0	1	0	w
0	1	1	x
1	0	0	y
1	0	1	z
1	1	0	y
1	1	1	z

b) truth table



c) Symbol

Figure 4. A 6-to-1 multiplexer.



Three-bit wide 6-to-1 multiplexer

Figure 5. A three-bit wide 6-to-1 multiplexer.

Perform the following steps to implement the three-bit wide 6-to-1 multiplexer.

1. Create a new Quartus II project for your circuit.
2. Create a Verilog module for the three-bit wide 6-to-1 multiplexer. Connect its select inputs to keys KEY_{2-0} and use the 18 switches SW_{17-0} to provide the six 3-bit inputs U to Z. Connect the *SW* switches to the red lights *LEDR* and connect the output M to the green lights $LEDG_{2-0}$.
3. Simulate your circuit with QSim for different values of the inputs, ensuring the output waveforms are correct.
4. Include in your project the required pin assignments for the DE2 board. Compile the project.
5. Download the compiled circuit into the FPGA chip. Test the functionality of the three-bit wide 6-to-1 multiplexer by toggling the switches and observing the LEDs. Ensure that each of the inputs U to Z can be properly selected as the output M.

Part IV

Figure 6 shows a *7-segment* decoder module that has the three-bit input $c_2c_1c_0$. This decoder produces seven outputs that are used to display a character on a 7-segment display. Table 1 lists the characters that should be displayed for each value of $c_2c_1c_0$. To keep the design simple, only six characters are included in the table (plus the 'blank' character, which is selected for codes $110 - 111$).

The seven segments in the display are identified by the indices 0 to 6 shown in the figure. Each segment is illuminated by driving it to the logic value 0. You are to write a Verilog module that implements logic functions that represent circuits needed to activate each of the seven segments. Use only simple Verilog **assign** statements in your code to specify each logic function using a Boolean expression.

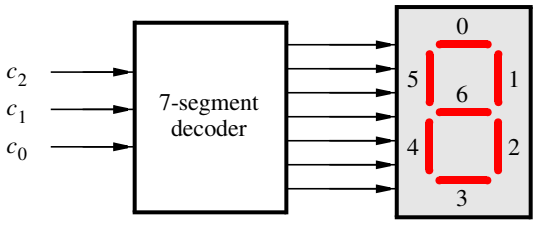


Figure 6. A 7-segment decoder.

$c_2c_1c_0$	Character
000	L
001	E
010	A
011	F
100	6
101	7
110	
111	

Table 1. Character codes.

Perform the following steps:

1. Create a new Quartus II project for your circuit.
2. Create a Verilog module for the 7-segment decoder. Connect the $c_2c_1c_0$ inputs to switches SW_{2-0}, and connect the outputs of the decoder to the *HEX0* display on the DE2 board. The segments in this display are called $HEX0_0$, $HEX0_1$, . . ., $HEX0_6$, corresponding to Figure 6. You should declare the 7-bit port

   output [6:0] HEX0;

   in your Verilog code so that the names of these outputs match the corresponding names in the *DE2 User Manual* and the *DE2_pin_assignments.csv* file.
3. Simulate your circuit with QSim for a variety of input settings, ensuring the output waveforms are correct.
4. After making the required DE2 board pin assignments, compile the project.
5. Download the compiled circuit into the FPGA chip. Test the functionality of the circuit by toggling the SW_{2-0} switches and observing the 7-segment display.

Part V

Consider the circuit shown in Figure 7. It uses a three-bit wide 6-to-1 multiplexer to enable the selection of six characters that are displayed on a 7-segment display. Using the 7-segment decoder from Part IV this circuit can display any of the characters L, E, A, F, 6, 7. The character codes are set according to Table 1 by using the switches SW_{17-0}, and a specific character is selected for display by setting the keys KEY_{2-0}.

In the next lab, we will use multiple seven-segment displays to form words from the letters. So, do not delete your Verilog, as you will need to use it next week!

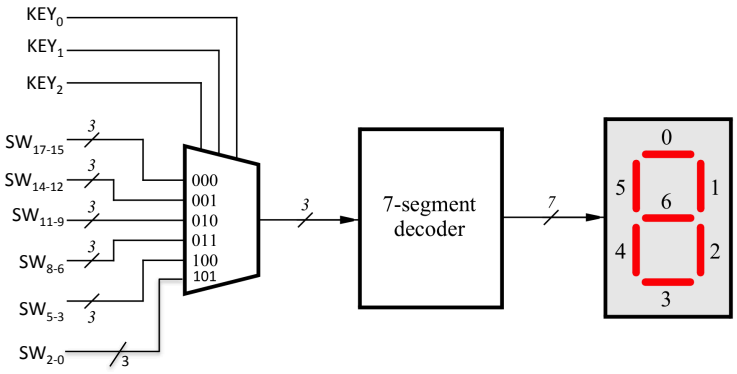


Figure 7. A circuit that can select and display one of six characters.

Perform the following steps.

1. Create a new Quartus II project for your circuit.
2. Include your Verilog module in the Quartus II project. Connect the keys KEY_{2-0} to the select inputs of the three-bit wide 6-to-1 multiplexer. Also connect SW_{17-0} to the multiplexer as required to produce the patterns of characters shown in Table 1. Connect the outputs of the multiplexer to the 7-segment display *HEX0*.
3. Simulate your circuit with QSim; ensure the output waveforms are correct.
4. Include the required pin assignments for the DE2 board for all switches, LEDs, and 7-segment displays. Compile the project.
5. Download the compiled circuit into the FPGA chip. Test the functionality of the circuit by setting the proper character codes on the switches SW_{17-0} and then toggling KEY_{2-0} to observe the different characters.