Laboratory Exercise 1

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_{9-0} on the DE0 board as inputs to the circuit. We will use light emitting diodes (LEDs) and 7-segment displays as output devices.

Part I

The DE0 board provides 10 toggle switches, called SW_{9-0}, that can be used as inputs to a circuit, and 10 green lights, called $LEDG_{9-0}$, 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 10 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 10 *LEDG* outputs, which is equivalent to the individual assignments

> **assign** LEDG[9] = SW[9];
> **assign** LEDG[8] = SW[8];
> . . .
> **assign** LEDG[0] = SW[0];

The DE0 board has hardwired connections between its FPGA chip and the switches and lights. To use SW_{9-0} and $LEDG_{9-0}$ it is necessary to include in your Quartus II project the correct pin assignments, which are given in the *DE0 User Manual*. For example, the manual specifies that SW_0 is connected to the FPGA pin *J6* and $LEDG_0$ is connected to pin *J1*. A good way to make the required pin assignments is to import into the Quartus II software directory the file called *DE0_pin_assignments.qsf*, which is provided on the *DE0 System CD* and in the University Program section of Altera's web site. The procedure for making pin assignments is described in the tutorial *Quartus II Introduction using Verilog Design*, which is also available from Altera.

It is important to realize that the pin assignments in the *DE0_pin_assignments.qsf* 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*[9] and *LEDG*[0] . . . *LEDG*[9] for the switches and lights, which is the reason we used these names in Figure 1.

```
// Simple module that connects the SW switches to the LEDG lights
module  part1 (SW, LEDG);
    input [9:0] SW;         // toggle switches
    output [9:0] LEDG;      // green LEDs

    assign LEDG = SW;
endmodule
```

Figure 1. Verilog code that uses the DE0 board switches and lights.

Perform the following steps to implement a circuit corresponding to the code in Figure 1 on the DE0 board.

1. Create a new Quartus II project for your circuit. Select Cyclone III EP3C16F484C6 as the target chip, which is the FPGA chip on the Altera DE0 board.

2. Create a Verilog module for the code in Figure 1 and include it in your project.

3. Include in your project the required pin assignments for the DE0 board, as discussed above. Compile the project.

4. Download the compiled circuit into the FPGA chip. Test the functionality of the circuit by toggling the switches and observing the LEDs.
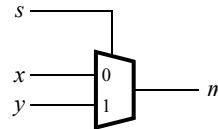
Part II

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.
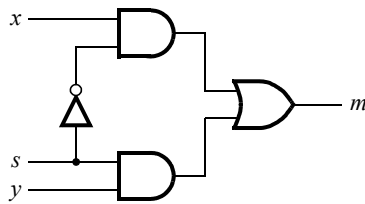
a) Circuit

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 = ($\sim$s & x) | (s & y);

You are to write a Verilog module that includes four assignment statements like the one shown above to describe the circuit given in Figure 3a. This circuit has two three-bit inputs, X and Y, and produces the three-bit output M. If $s = 0$ then $M = X$, while if $s = 1$ then $M = Y$. We refer to this circuit as a three-bit wide 2-to-1 multiplexer. It has the circuit symbol shown in Figure 3b, in which X, Y, and M are depicted as three-bit wires. Perform the steps shown below.

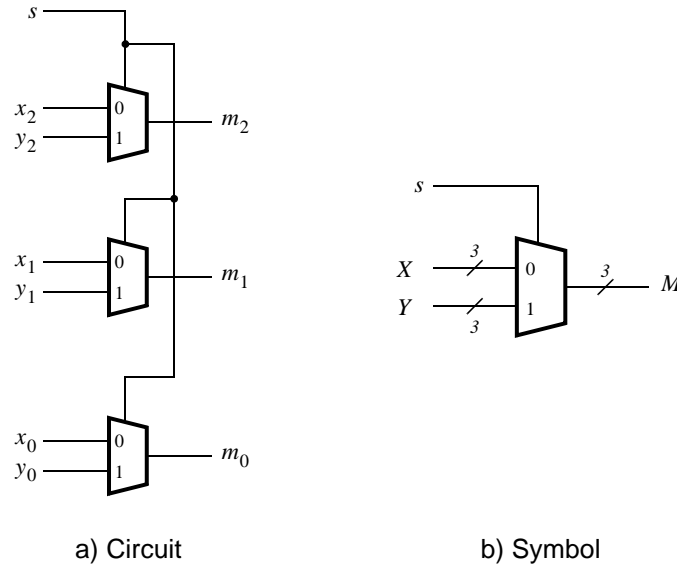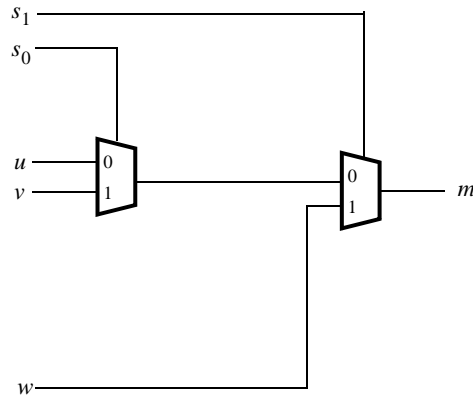a) Circuit   b) Symbol

Figure 3. A three-bit wide 2-to-1 multiplexer.

1. Create a new Quartus II project for your circuit.

2. Include your Verilog file for the three-bit wide 2-to-1 multiplexer in your project. Use switch SW_9 on the DE0 board as the s input, switches SW_{5-3} as the X input and SW_{8-6} as the Y input. Connect the SW_{9-3} to the green lights $LEDG_{9-3}$ and connect the output M to the green lights $LEDG_{2-0}$.

3. Include in your project the required pin assignments for the DE0 board. As discussed in Part I, these assignments ensure that the input ports of your Verilog code will use the pins on the Cyclone III FPGA that are connected to the SW switches, and the output ports of your Verilog code will use the FPGA pins connected to the $LEDG$ lights.

4. Compile the project.

5. Download the compiled circuit into the FPGA chip. Test the functionality of the three-bit wide 2-to-1 multiplexer 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 three inputs u, v, and w. Part a of Figure 4 shows how we can build the required 3-to-1 multiplexer by using two 2-to-1 multiplexers. The circuit uses a 2-bit select input $s_1 s_0$ and implements the truth table shown in Figure 4b. A circuit symbol for this multiplexer is given in part c of the figure.
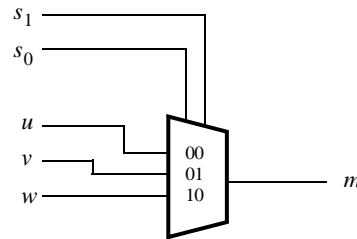
Recall from Figure 3 that a three-bit wide 2-to-1 multiplexer can be built by using three instances of a 2-to-1 multiplexer. Figure 5 applies this concept to define a two-bit wide 3-to-1 multiplexer. It contains two instances of the circuit in Figure 4a.

a) Circuit

s_1 s_0	m
0 0	u
0 1	v
1 0	w
1 1	w

b) Truth table   c) Symbol

Figure 4. A 3-to-1 multiplexer.

Perform the following steps to implement the two-bit wide 3-to-1 multiplexer.

1. Create a new Quartus II project for your circuit.

2. Create a Verilog module for the two-bit wide 3-to-1 multiplexer. Connect its select inputs to switches SW_{9-8}, and use switches SW_{7-2} to provide the three 2-bit inputs U to W. Connect the SW_{9-2} switches to the green lights $LEDG_{9-2}$ and connect the output M to the green lights $LEDG_{1-0}$.

3. Include in your project the required pin assignments for the DE0 board. Compile the project.

4. Download the compiled circuit into the FPGA chip. Test the functionality of the two-bit wide 3-to-1 multiplexer by toggling the switches and observing the LEDs. Ensure that each of the inputs U to W can be properly selected as the output M.

Part IV

Figure 6 shows a *7-segment* decoder module that has the two-bit input c_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 valuation of c_1c_0. To keep the design simple, only three characters are included in the table (plus the 'blank' character, which is selected for codes 11).

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
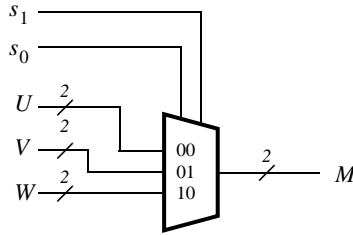
Figure 5. A two-bit wide 3-to-1 multiplexer.

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.
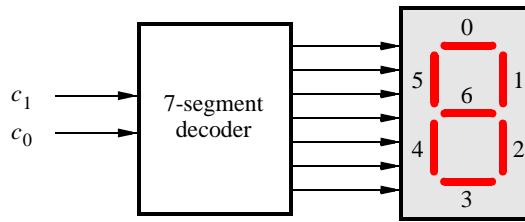

Figure 6. A 7-segment decoder.

c_1c_0	Character
00	d
01	E
10	0
11	

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_1c_0 inputs to switches SW_{1-0}, and connect the outputs of the decoder to the *HEX0* display on the DE0 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 [0:6] HEX0;

   in your Verilog code so that the names of these outputs match the corresponding names in the *DE0 User Manual* and the *DE0_pin_assignments.qsf* file.

3. After making the required DE0 board pin assignments, compile the project.

4. Download the compiled circuit into the FPGA chip. Test the functionality of the circuit by toggling the SW_{1-0} switches and observing the 7-segment display.

Part V

Consider the circuit shown in Figure 7. It uses a two-bit wide 3-to-1 multiplexer to enable the selection of three 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 D, E, 0, and 'blank'. The character codes are set according to Table 1 by using the switches SW_{5-0}, and a specific character is selected for display by setting the switches SW_{9-8}.

An outline of the Verilog code that represents this circuit is provided in Figure 8. Note that we have used the circuits from Parts III and IV as subcircuits in this code. You are to extend the code in Figure 8 so that it uses three 7-segment displays rather than just one. You will need to use three instances of each of the subcircuits. The purpose of your circuit is to display any word on the four displays that is composed of the characters in Table 1, and be able to rotate this word in a circular fashion across the displays when the switches SW_{9-8} are toggled. As an example, if the displayed word is dE0, then your circuit should produce the output patterns illustrated in Table 2.
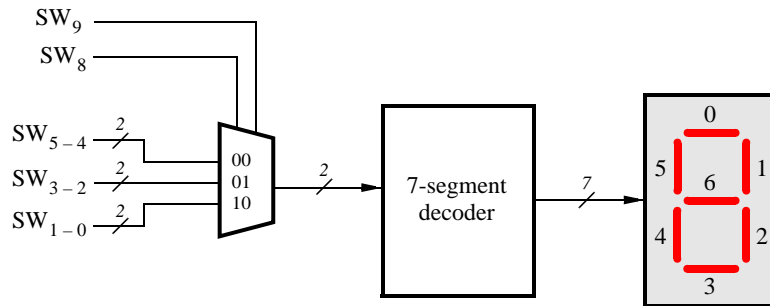

Figure 7. A circuit that can select and display one of three characters.

```
module part5 (SW, HEX0);
    input [9:0] SW;         // toggle switches
    output [0:6] HEX0;      // 7-seg displays

    wire [1:0] M;

    mux_2bit_3to1 M0 (SW[9:8], SW[5:4], SW[3:2], SW[1:0], M);
    char_7seg H0 (M, HEX0);
endmodule

// implements a 2-bit wide 3-to-1 multiplexer
module mux_2bit_3to1 (S, U, V, W, M);
    input [1:0] S, U, V, W;
    output [1:0] M;

    ... code not shown

endmodule

// implements a 7-segment decoder for d, E, 0 and 'blank'
module char_7seg (C, Display);
    input [1:0] C;          // input code
    output [0:6] Display;   // output 7-seg code

    ... code not shown

endmodule
```

Figure 8. Verilog code for the circuit in Figure 7.

$SW_9\ SW_8$	Character pattern		
00	d	E	0
01	E	0	d
10	0	d	E

Table 2. Rotating the word dE0 on three displays.

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 switches SW_{9-8} to the select inputs of each of the three instances of the three-bit wide 3-to-1 multiplexers. Also connect SW_{5-0} to each instance of the multiplexers as required to produce the patterns of characters shown in Table 2. Connect the outputs of the three multiplexers to the 7-segment displays *HEX2*, *HEX1*, and *HEX0*.

3. Include the required pin assignments for the DE0 board for all switches, LEDs, and 7-segment displays. Compile the project.

4. Download the compiled circuit into the FPGA chip. Test the functionality of the circuit by setting the proper character codes on the switches SW_{5-0} and then toggling SW_{9-8} to observe the rotation of the characters.

Part VI

Extend your design from Part V so that is uses all four 7-segment displays on the DE0 board. Your circuit should be able to display words with three (or fewer) characters on the four displays, and rotate the displayed word when the switches SW_{9-8} are toggled. If the displayed word is dE0, then your circuit should produce the patterns shown in Table 3.

$SW_9\ SW_8$	Character pattern				
00			d	E	0
01		d	E	0	
10	E	0			d
11	0			d	E

Table 3. Rotating the word dE0 on eight displays.

Perform the following steps:

1. Create a new Quartus II project for your circuit and select the target chip as Cyclone III EP3C16F484C6.

2. Include your Verilog module in the Quartus II project. Connect the switches SW_{9-8} to the select inputs of each instance of the multiplexers in your circuit. Also connect SW_{5-0} to each instance of the multiplexers as required to produce the patterns of characters shown in Table 3. (Hint: for some inputs of the multiplexers you will want to select the 'blank' character.) Connect the outputs of your multiplexers to the 7-segment displays *HEX3, ..., HEX0*.

3. Include the required pin assignments for the DE0 board for all switches, LEDs, and 7-segment displays. Compile the project.

4. Download the compiled circuit into the FPGA chip. Test the functionality of the circuit by setting the proper character codes on the switches SW_{5-0} and then toggling SW_{9-8} to observe the rotation of the characters.