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 on the DE-series boards as inputs to the circuit. We will use light emitting diodes (LEDs) and 7-segment displays as output devices.

Part I

The DE10-Lite, DE0-CV, and DE1-SoC boards provide ten switches and lights, called SW_{9-0} and $LEDR_{9-0}$. Similarly, the DE2-115 provides eighteen switches and lights. The switches can be used to provide inputs, and the lights can be used as output devices. Figure 1 shows a simple Verilog module that uses ten switches and shows their states on the LEDs. Since there are multiple 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 *LEDR* outputs, which is equivalent to the individual assignments:

```
. . .
assign LEDR[2] = SW[2];
assign LEDR[1] = SW[1];
assign LEDR[0] = SW[0];
```

The DE-series boards have hardwired connections between its FPGA chip and the switches and lights. To use the switches and lights it is necessary to include in your Quartus project the correct pin assignments, which are given in your board's user manual. For example, the DE1-SoC manual specifies that SW_0 is connected to the FPGA pin *AB12* and $LEDR_0$ is connected to pin *V16*. A good way to make the required pin assignments is to import into the Quartus software the pin assignment file for your board, which is provided on the FPGA University Program section of Intel's web site. The procedure for making pin assignments is described in the tutorial *Quartus Introduction using Verilog Design*, which is also available from Intel.

It is important to realize that the pin assignments in the file are useful only if the pin names that appear in this file are exactly the same as the port names used in your Verilog module. For example, if the pin assignment file uses the names *SW*[0], . . ., *SW*[9] and *LEDR*[0], . . ., *LEDR*[9], then these are the names that must be used for input and output ports in the Verilog code, as we have done in Figure 1.

```
// Module that connects ten switches and lights
module  part1 (SW, LEDR);
    input [9:0] SW;          // slide switches
    output [9:0] LEDR;       // red LEDs

    assign LEDR = SW;
endmodule
```

Figure 1: Verilog code that uses ten switches and lights.

Perform the following steps to implement a circuit corresponding to the code in Figure 1 on the DE-series boards.

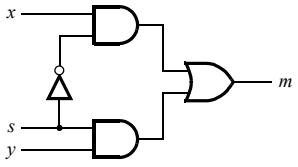
1. Create a new Quartus project for your circuit. Select the target chip that corresponds to your DE-series board. Refer to Table 1 for a list of devices.
2. Create a Verilog module for the code in Figure 1 and include it in your project.
3. Include in your project the required pin assignments for your DE-series board, as discussed above. Compile the project.
4. Download the compiled circuit into the FPGA chip by using the Quartus Programmer tool (the procedure for using the Programmer tool is described in the tutorial *Quartus Introduction*). Test the functionality of the circuit by toggling the switches and observing the LEDs.

Board	**Device Name**
DE10-Lite	MAX 10 10M50DAF484C6GES
DE0-CV	Cyclone V 5CEBA4F23C7
DE1-SoC	Cyclone V SoC 5CSEMA5F31C6
DE2-115	Cyclone IVE EP4CE115F29C7

Table 1: DE-series FPGA device names

Part II

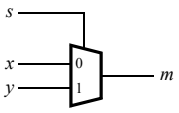
Figure 2a shows a sum-of-products circuit that implements a 2-to-1 *multiplexer* with a select input s. If $s = 0$ the multiplexer's output m is equal to the input x, and if $s = 1$ the output is equal to y. Part b of the figure gives a truth table for this multiplexer, and part c shows its circuit symbol.



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

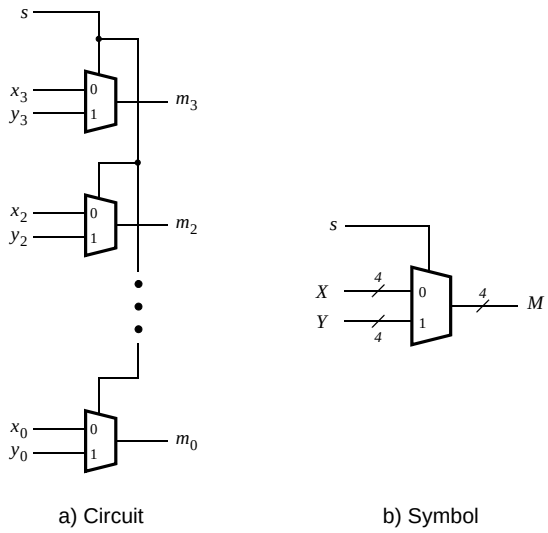


Figure 3: A four-bit wide 2-to-1 multiplexer.

Perform the steps listed below.

1. Create a new Quartus project for your circuit.
2. Include your Verilog file for the four-bit wide 2-to-1 multiplexer in your project. Use switch SW_9 as the s input, switches SW_{3-0} as the X input and SW_{7-4} as the Y input. Display the value of the input s on $LEDR_9$, connect the output M to $LEDR_{3-0}$, and connect the unused LEDR lights to the constant value 0.
3. Include in your project the required pin assignments for your DE-series board. As discussed in Part I, these assignments ensure that the ports of your Verilog code will use the pins on the FPGA chip that are connected to the *SW* switches and *LEDR* lights.
4. Compile the project, and then download the resulting circuit into the FPGA chip. Test the functionality of the four-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 four inputs u, v, w, and x. Part a of Figure 4 shows how we can build the required 4-to-1 multiplexer by using three 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 four-bit wide 2-to-1 multiplexer can be built by using four instances of a 2-to-1 multiplexer. Figure 5 applies this concept to define a two-bit wide 4-to-1 multiplexer. It contains two instances of the circuit in Figure 4a.

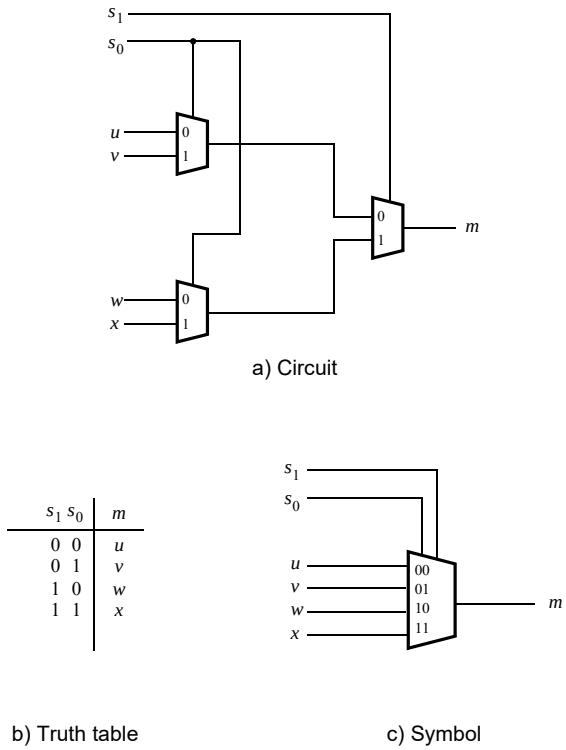


Figure 4: A 4-to-1 multiplexer.

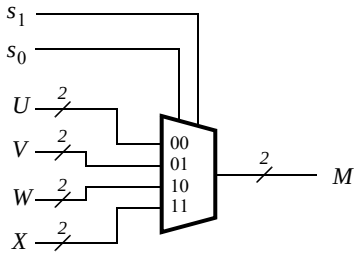


Figure 5: A two-bit wide 4-to-1 multiplexer.

Perform the following steps to implement the two-bit wide 4-to-1 multiplexer.

1. Create a new Quartus project for your circuit.
2. Create a Verilog module for the two-bit wide 4-to-1 multiplexer. Connect its select inputs to switches SW_{9-8}, and use switches SW_{7-0} to provide the four 2-bit inputs U to X. Connect the output M to the red lights $LEDR_{1-0}$.
3. Include in your project the required pin assignments for your DE-series board. Compile the project.
4. Download the compiled circuit into the FPGA chip. Test the functionality of the two-bit wide 4-to-1 multiplexer by toggling the switches and observing the LEDs. Ensure that each of the inputs U to X can be properly selected as the output M.