Laboratory 1:

GET STARTED WITH FPGA

OBJECTIVES

- The purpose of this lab 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.
- Besides, we learn how to write simple digital circuits (multiplexer, decoder, …).

PREPARATION FOR LAB 1

- Finish Lab 0 at home.
- Finish Pre Lab 1 at home.
- Students have to simulate all the exercises in Lab 0 and Pre Lab 1 at home. All results (codes, waveform, RTL viewer, …) have to be captured and submitted to instructors prior to the lab session.

  If not, students will not participate in the lab and be considered absent this session.

REFERENCE

1. Intel FPGA training

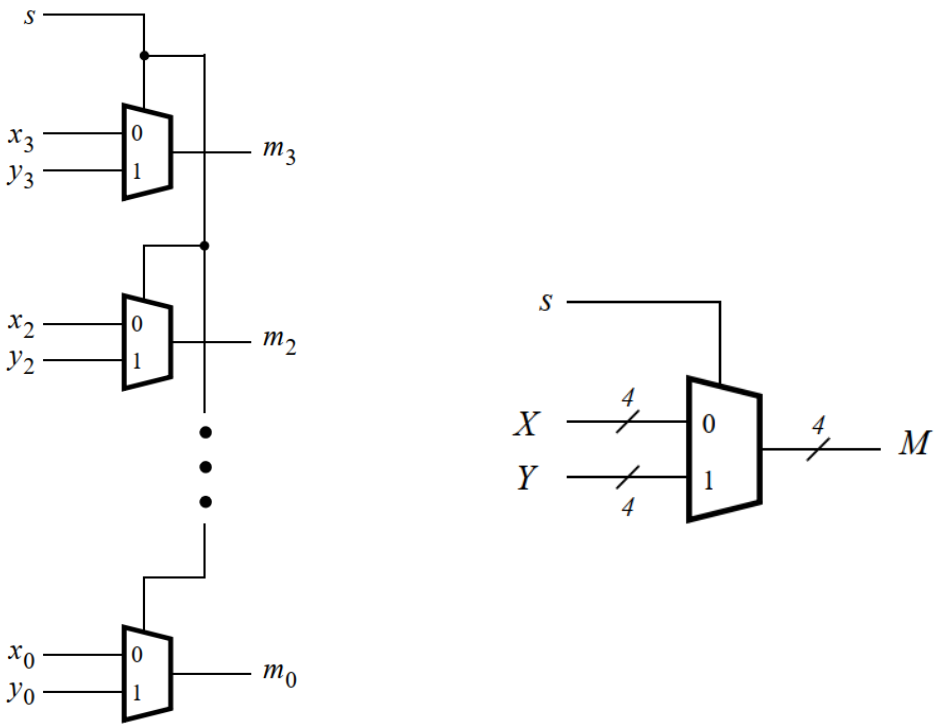
Laboratory 1:

GET STARTED WITH FPGA

EXPERIMENT 1

Objective: Known how to program n-bit wide 2-to-1 multiplexer.

Requirement: Write a VHDL entity to describe the circuit given in Figure 1a. 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 . This circuit is called four-bit wide 2-to-1 multiplexer. It has the circuit symbol shown in Figure 1b, in which X, Y, and M are depicted as four-bit wires.



(a) Circuit (b) Symbol

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

Instruction:

1. Create a new Quartus project for your circuit.
2. In your Pre Lab, exercise 1, the multiplexer can be described by the following VHDL statement:

```
m <= (NOT (s) AND x) OR (s AND y)
```

Write a VHDL entity that includes four assignment statements like the one shown above to describe the circuit given in Figure 1a.

3. Use switch SW9 as the *s* input, switches SW3−0 as the X input and SW7−4 as the Y input. Display the value of the input s on LEDR9, connect the output M to LEDR3−0, and connect the unused LEDR lights to the constant value 0.
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.

Check: Your report has to show two results:

- The waveform to prove the circuit works correctly.
- The result of RTL viewer.

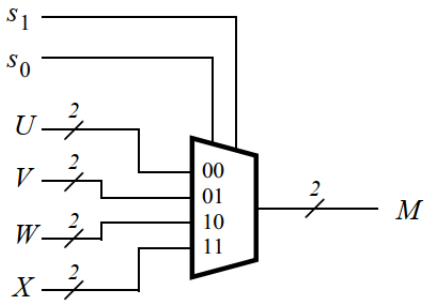
Laboratory 1:

GET STARTED WITH FPGA

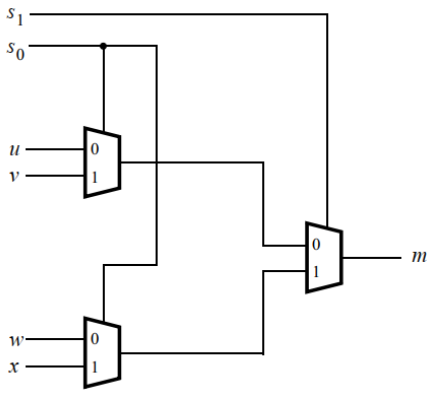
EXPERIMENT 2

Objective: Known how to program n-bit wide 4-to-1 multiplexer.

Requirement: Write a VHDL entity to describe the circuit in Figure 2a called two-bit wide 4-to-1 multiplexer. For this part consider a circuit in which the output *m* has to be selected from four inputs *u*, *v*, *w*, and *x*. Part b of Figure 2 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 2c. A circuit symbol for this multiplexer is given in part d of the figure.



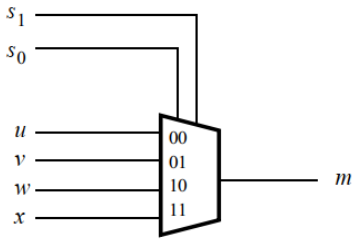
(a) Symbol two-bit wide 4-to-1 multiplexer



(b) Circuit one-bit wide 4-to-1 multiplexer

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

(c) Truth table one-bit wide 4-to-1 multiplexer



(d) Symbol one-bit wide 4-to-1 multiplexer

Figure 2: Instruction to design a two-bit wide 4-to-1 multiplexer.

Instruction:

- In experiment 1, a four-bit wide 2-to-1 multiplexer can be built by using four instances of a 2-to-1 multiplexer. Figure 2a applies this concept to define a two-bit wide 4-to-1 multiplexer. It contains two instances of the circuit in Figure 2b.

Laboratory 1:

GET STARTED WITH FPGA

- Create a VHDL entity for the two-bit wide 4-to-1 multiplexer. Connect its select inputs *s*1*s*0 to switches SW9-8, and use switches SW7-0 to provide the four 2-bit inputs U to X. Connect the output M to the red lights LEDR1-0.

Check: Your report has to show two results:

- The waveform to prove the circuit works correctly.
- The result of RTL viewer.

Laboratory 1:

GET STARTED WITH FPGA

EXPERIMENT 3

Objective: Known how to interface 7-segment LED.

Requirement: Consider the circuit shown in Figure 3. It uses a two-bit wide 4-to-1 multiplexer to enable the selection of four characters that are displayed on a 7-segment display. Using the 7-segment decoder from exercise 3 (Pre Lab 1), this circuit can display the characters d, E, 0, 1 on your DE-series board. The character codes are set according to Table 1 by using the switches *SW*7-0, and a specific character is selected for display by setting the switches *SW*9-8.

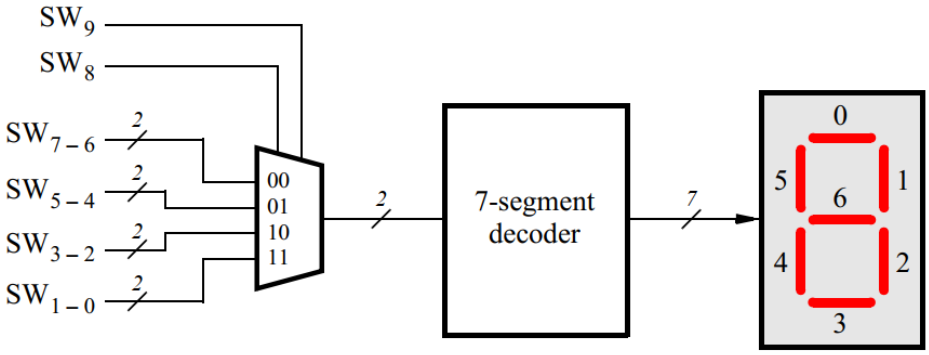
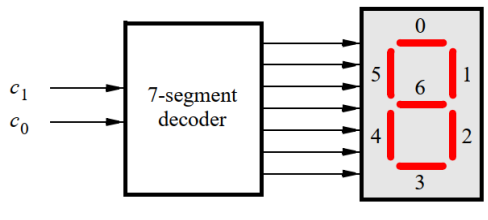


Figure 3: A circuit that can select and display one of four characters.

SW_{9-8}	Characters			
00	d	E	1	0
01	E	1	0	d
10	1	0	d	E
11	0	d	E	1

Table 1: Rotating the word dE10 on four displays.



(a)

$c_1 c_0$	DE10
00	d
01	E
10	1
11	0

(b)

Figure 4: Character codes for the DE-10 boards.

Instruction:

- An outline of the VHDL code that represents this circuit is provided below. Note that we have used the circuits from experience 2 and exercise 3 (Pre Lab 1) as subcircuits in this code. You are to extend the following code so that it uses four 7-segment displays rather than just one. You will need to use four instances of each of the subcircuits. The purpose of your circuit is to display any word on the three 7-segment displays that is composed of the characters in Table of Figure 4, 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 dE10, then your circuit should produce the output patterns illustrated in Table (Figure 4).

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY experience3 IS
PORT ( SW : IN STD_LOGIC_VECTOR(9 DOWNTO 0);
LEDR : OUT STD_LOGIC_VECTOR(9 DOWNTO 0));
HEX0 : OUT STD_LOGIC_VECTOR(0 TO 6) );
END experience3;
ARCHITECTURE Behavior OF experience3 IS
COMPONENT mux_2bit_4to1
PORT (S, U, V, W, X : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
M : OUT STD_LOGIC_VECTOR(1 DOWNTO 0));
END COMPONENT;
COMPONENT char_7seg
PORT ( C : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
Display : OUT STD_LOGIC_VECTOR(0 TO 6));
END COMPONENT;
SIGNAL M0 : STD_LOGIC_VECTOR(1 DOWNTO 0);
BEGIN
U0: mux_2bit_4to1 PORT MAP (SW(9 DOWNTO 8), SW(7 DOWNTO 6), SW(5
DOWNTO 4),
SW(3 DOWNTO 2), SW(1 DOWNTO 0), M0);
H0: char_7seg PORT MAP (M0, HEX0);
……
```

```
END Behavior;
---- implements a 2-bit wide 4-to-1 multiplexer ---------
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY mux_2bit_4to1 IS
PORT ( S, U, V, W, X : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
M : OUT STD_LOGIC_VECTOR(1 DOWNTO 0));
END mux_2bit_4to1;
ARCHITECTURE Behavior OF mux_2bit_4to1 IS
… code not shown
END Behavior;
-------------------------------------------------------
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY char_7seg IS
PORT ( C : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
Display : OUT STD_LOGIC_VECTOR(0 TO 6));
END char_7seg;
ARCHITECTURE Behavior OF char_7seg IS
… code not shown
END Behavior;
```

- Include your VHDL entity in the Quartus project. Connect the switches *SW*9-8 to the select inputs of each of the three instances of the two-bit wide 4-to-1 multiplexers. Also connect *SW*7-0 to each instance of the multiplexers as required to produce the patterns of characters. Connect the SW switches to the red lights LEDR, and connect the outputs of the four multiplexers to the 7-segment displays *HEX3*, *HEX2*, *HEX1*, and *HEX0*.

Check: Your report has to show two results:

- The waveform to prove the circuit works correctly.
- The result of RTL viewer.

Laboratory 1:

GET STARTED WITH FPGA

EXPERIMENT 4

Objective: Known how to interface 7-segment LED and multiplexer.

Requirement: Extend your design from expermient 3 so that is uses all 7-segment displays on your DE-10 board. Your circuit needs to display letter H, E, L, L, O and blank. The character codes are set according to Table 2. Note that for the DE-10 you will need to use 3-bit codes for your characters, because five characters are needed when including the 'blank' character (your 7-segment decoder will have to use 3-bit codes, and you will need to use 3-bit wide 6-to-1 multiplexers). The character codes are shown in Table 3.

SW9-7	Character pattern
000	H E L L O
001	H E L L O
010	H E L L O
011	O H E L L
100	L O H E L
101	L L O H E
100	E L L O H

Table 2: Rotating the word HELLO on six displays.

c2c1c0	Character
000	H
001	E
010	L
011	O
100	
101	
100	

Table 3: Character codes of 7-segment LED.

Instruction:

- The code is extended from the previous experience. You will need to use three select lines for each of the multiplexers: connect the select lines to switches *SW*9-7. In your VHDL code connect constants to the 6-to-1 (or 8-to-1) multiplexers that select each character, because there are not enough *SW* switches.
- Remember to connect the SW switches to the red lights LEDR, and connect the outputs of the six multiplexers to the 7-segment displays *HEX7 to HEX0*.

Check: Your report has to show two results:

- The waveform to prove the circuit works correctly.
- The result of RTL viewer.

Laboratory 1:

GET STARTED WITH FPGA

EXPERIMENT 5

Objective: Known how to interface 7-segment LED and multiplexer.

Requirement: You are to design a circuit that converts a four-bit binary number $V = v_3v_2v_1v_0$ into its two-digit decimal equivalent $D = d_1d_0$. Table 4 shows the required output values. A partial design of this circuit is given in Figure 5. It includes a comparator that checks when the value of V is greater than 9, and uses the output of this comparator in the control of the 7-segment displays. You are to complete the design of this circuit.

$v_3v_2v_1v_0$	d_1	d_0
0000	0	0
0001	0	1
0010	0	2
...	...	...
1001	0	9
1010	1	0
1011	1	1
1100	1	2
1101	1	3
1110	1	4
1111	1	5

Table 4: Binary-to-decimal conversion values.

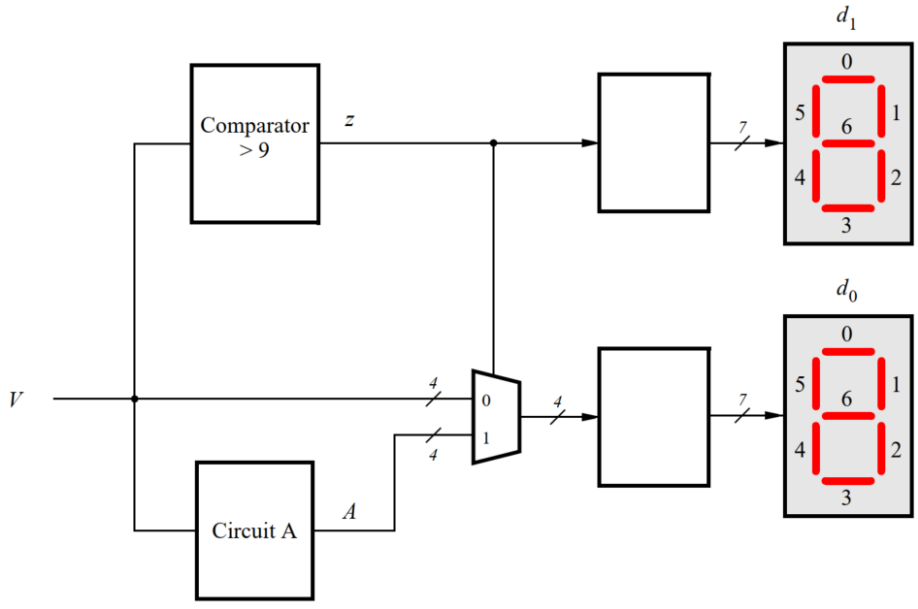


Figure 5: Partial design of the binary-to-decimal conversion circuit.

Laboratory 1:

GET STARTED WITH FPGA

Instruction:

- The output z for the comparator circuit can be specified using a single Boolean expression, with the four inputs $V_{3\text{-}0}$. Design this Boolean expression by making a truth table that shows the valuations of the inputs $V_{3\text{-}0}$ for which z has to be 1.
- Notice that the circuit in Figure 5 includes a 4-bit wide 2-to-1 multiplexer. The purpose of this multiplexer is to drive digit d_0 with the value of V when $z = 0$, and the value of A when $z = 1$. To design circuit A consider the following. For the input values $V \leq 9$, the circuit A does not matter, because the multiplexer in Figure 5 just selects V in these cases. But for the input values $V > 9$, the multiplexer will select A. Thus, A has to provide output values that properly implement Table 4 when $V > 9$. You need to design circuit A so that the input $V = 1010$ gives an output $A = 0000$, the input $V = 1011$ gives the output $A = 0001$, ..., and the input $V = 1111$ gives the output $A = 0101$. Design circuit A by making a truth table with the inputs $V_{3\text{-}0}$ and the outputs $A_{3\text{-}0}$.
- Write VHDL code to implement your design. The code should have the 4-bit input $SW_{3\text{-}0}$, which should be used to provide the binary number V, and the two 7-bit outputs *HEX1* and *HEX0*, to show the values of decimal digits d_1 and d_0. The intent of this exercise is to use simple VHDL assignment statements to specify the required logic functions using Boolean expressions. Your VHDL code should not include any IF-ELSE, CASE, or similar statements.

Check:

- Adjust the waveform to prove the circuit works correctly and capture it.
- Capture the result of RTL viewer.
- Your report has to show two above results.