Pre 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

- 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

Pre Laboratory 1:

GET STARTED WITH FPGA

EXERCISE 1:

Objective: Get started with FPGA

Instruction: The DE10-Standard boards provide ten switches and lights, called *SW*9-0 and *LEDR*9-0. These switches can be used to provide inputs, and the lights can be used as output devices. The code below shows a simple VHDL entity 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 VHDL code, as shown. We have used a single assignment statement for all *LEDR* outputs, which is equivalent to the individual assignments:

```
LEDR(9) <= SW(9);

LEDR(8) <= SW(8);

...

LEDR(0) <= SW(0);
```

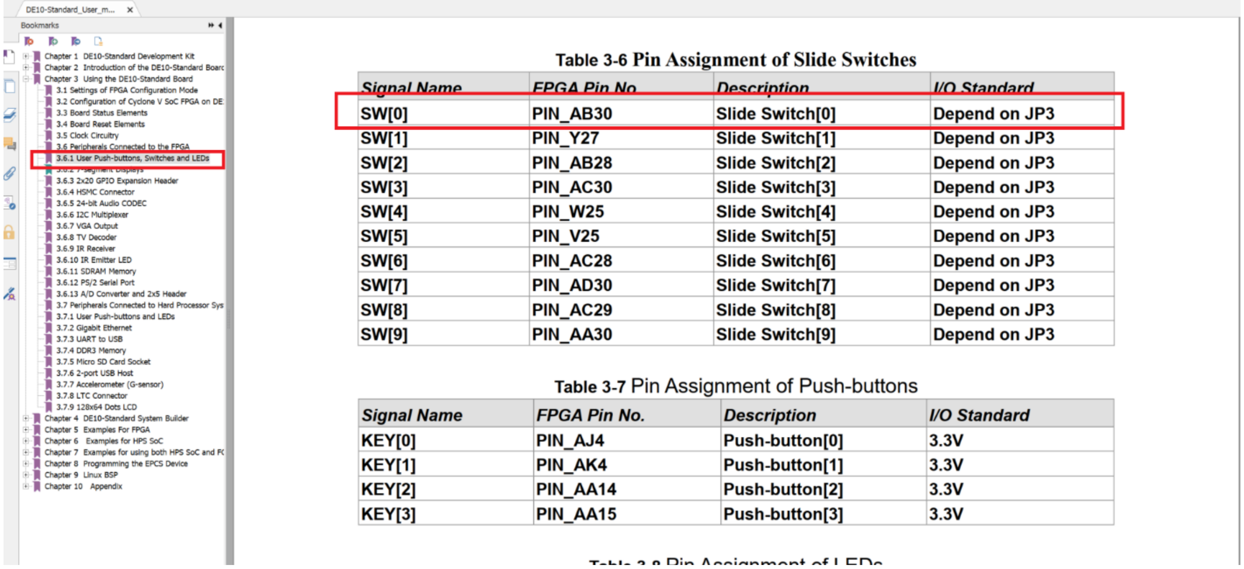
The DE-10 boards have hardwired connections between its FPGA chip and the switches and lights. To use thenswitches and lights it is necessary to include in your Quartus project the correct pin assignments, which are given in your board's user manual.

- The first 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 VHDL Design*, which is also available from Intel.

- The second way is using file *DE10-Standard_User_manual.pdf,* which is avaible for all of you. Open and file the PIN which you want to use. For example, the Figure 1 show which PIN is SW[0] connected to. In short, open the menu bar, choose the peripherals you want to use, then find the PIN in the corresponding table.

Pre Laboratory 1:

GET STARTED WITH FPGA



DE10-Standard_User_m...

Bookmarks

Chapter 1 DE10-Standard Development Kit
Chapter 2 Introduction of the DE10-Standard Boarc
Chapter 3 Using the DE10-Standard Board
3.1 Settings of FPGA Configuration Mode
3.2 Configuration of Cyclone V SoC FPGA on DE
3.3 Board Status Elements
3.4 Board Reset Elements
3.5 Clock Circuitry
3.6 Peripherals Connected to the FPGA
3.6.1 User Push-buttons, Switches and LEDs
3.6.2 7-segment Displays
3.6.3 2x20 GPIO Expansion Header
3.6.4 HSMC Connector
3.6.5 24-bit Audio CODEC
3.6.6 I2C Multiplexer
3.6.7 VGA Output
3.6.8 TV Decoder
3.6.9 IR Receiver
3.6.10 IR Emitter LED
3.6.11 SDRAM Memory
3.6.12 PS/2 Serial Port
3.6.13 A/D Converter and 2x5 Header
3.7 Peripherals Connected to Hard Processor Sys
3.7.1 User Push-buttons and LEDs
3.7.2 Gigabit Ethernet
3.7.3 UART to USB
3.7.4 DDR3 Memory
3.7.5 Micro SD Card Socket
3.7.6 2-port USB Host
3.7.7 Accelerometer (G-sensor)
3.7.8 LTC Connector
3.7.9 128x64 Dots LCD
Chapter 4 DE10-Standard System Builder
Chapter 5 Examples For FPGA
Chapter 6 Examples for HPS SoC
Chapter 7 Examples for using both HPS SoC and F(
Chapter 8 Programming the EPCS Device
Chapter 9 Linux BSP
Chapter 10 Appendix

Table 3-6 Pin Assignment of Slide Switches

Signal Name	FPGA Pin No.	Description	I/O Standard
SW[0]	PIN_AB30	Slide Switch[0]	Depend on JP3
SW[1]	PIN_Y27	Slide Switch[1]	Depend on JP3
SW[2]	PIN_AB28	Slide Switch[2]	Depend on JP3
SW[3]	PIN_AC30	Slide Switch[3]	Depend on JP3
SW[4]	PIN_W25	Slide Switch[4]	Depend on JP3
SW[5]	PIN_V25	Slide Switch[5]	Depend on JP3
SW[6]	PIN_AC28	Slide Switch[6]	Depend on JP3
SW[7]	PIN_AD30	Slide Switch[7]	Depend on JP3
SW[8]	PIN_AC29	Slide Switch[8]	Depend on JP3
SW[9]	PIN_AA30	Slide Switch[9]	Depend on JP3

Table 3-7 Pin Assignment of Push-buttons

Signal Name	FPGA Pin No.	Description	I/O Standard
KEY[0]	PIN_AJ4	Push-button[0]	3.3V
KEY[1]	PIN_AK4	Push-button[1]	3.3V
KEY[2]	PIN_AA14	Push-button[2]	3.3V
KEY[3]	PIN_AA15	Push-button[3]	3.3V

Table 3-8 Pin Assignment of LEDs

Figure 1: How to find the PIN connected to SW[0]

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 VHDL entity. 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 VHDL code below.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
- - Simple entity that connects the SW switches to the LEDR lights
ENTITY part1 IS
PORT ( SW : IN STD_LOGIC_VECTOR(9 DOWNTO 0);
            LEDR : OUT STD_LOGIC_VECTOR(9 DOWNTO 0));
END part1;
ARCHITECTURE Behavior OF part1 IS
BEGIN
    LEDR <= SW;
END Behavior
```

Perform the following steps to implement a circuit corresponding to the code above on the DE-10 boards.

Pre Laboratory 1:

GET STARTED WITH FPGA

1. Create a new Quartus project for your circuit. Select the target chip that corresponds to your DE-10 board (Intel Cyclone® V SE **5CSXFC6D6F31C6N** device).

2. Create a VHDL entity for the code and include it in your project.

3. Include in your project the required pin assignments for your DE-10 board, as discussed above. Compile the project.

4. Download the compiled circuit into the FPGA chip by using the Quartus Programmer tool. Test the functionality of the circuit by toggling the switches and observing the LEDs.

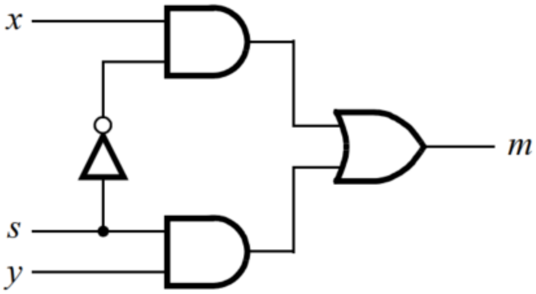
Pre Laboratory 1:

GET STARTED WITH FPGA

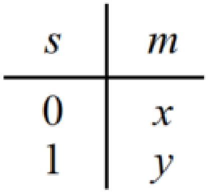
EXERCISE 2:

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

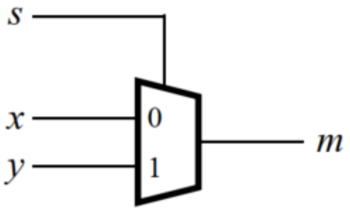
Requirement: Write a VHDL entity to describe the circuit in Figure 1a called one-bit wide 2-to-1 multiplexer. Figure 2*a* shows a sum-of-products circuit that implements a 2-to-1 *multiplexer* with a select input *s*. If *s* = 0 the multiplexer's output *m* is equal to the input *x*, and if *s* = 1 the 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) Symbol two-bit wide 4-to-1 multiplexer



s	m
0	x
1	y



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

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

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

Instruction: The multiplexer can be described by the following VHDL statement:

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

Check: Your report has to show two results:

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

Pre Laboratory 1:

GET STARTED WITH FPGA

EXERCISE 3

Objective: Known how to interface 7-segment LED.

Requirement: Write a program to display a character on a 7-segment display. The specific character displayed depends on a two-bit input. Figure 3 shows a *7-segment decoder* entity that has the two-bit input *c*1*c*0. This decoder produces seven outputs that are used to display a character on a 7-segment display. Table 2 lists the characters that should be displayed for each valuation of *c*1*c*0.

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 VHDL entity that implements logic functions to activate each of the seven segments. Use only simple VHDL assignment statements in your code to specify each logic function using a Boolean expression.

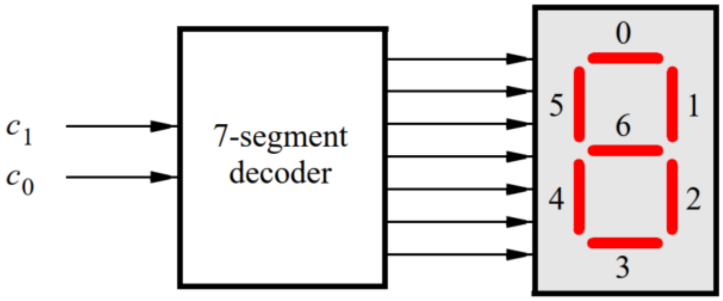


Figure 3: A circuit that can select and display one of four characters.

c_1c_0	DE10
00	d
01	E
10	1
11	0

Table 2: Character codes for the DE-10 boards.

Instruction:

- The segments in this display are called *HEX0*$_0$, *HEX0*$_1$, …, *HEX0*$_6$, corresponding to Figure 3. You should declare the 7-bit port:

  HEX0 : OUT STD_LOGIC_VECTOR(0 TO 6);

in your VHDL code so that the names of these outputs match the corresponding names in your board’s user manual and pin assignment file.

Check: Your report has to show two results:

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