ELEX 2117 : Digital Techniques 2
2021 Fall Term

Practice Lab

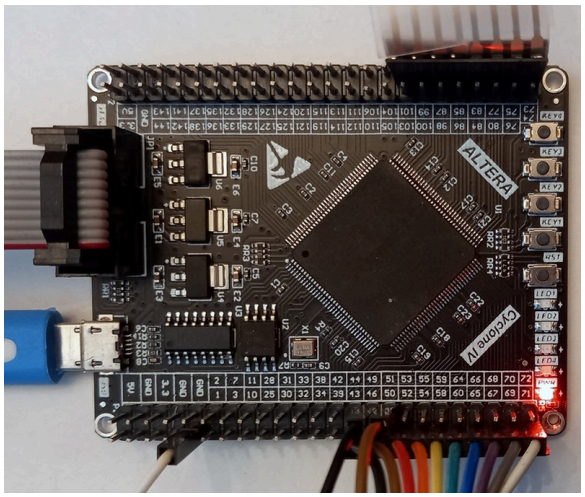
Version 2: corrected syntax errors in sample code.

Introduction

This lab will familiarize you with the FPGA board, the Quartus and Modelsim software, writing lab reports and creating video demonstrations. This lab will be marked but will not count towards your lab mark.

Intel Cyclone IV FPGA Board

This board contains an Intel (formerly Altera) EP4CE6E22C8N Cyclone IV field-programmable gate array (FPGA). An FPGA is a component that can be programmed to create many different products. This board will be used for most labs. The board's schematic is available on the course website.



The board has four active-low push-button switches (labelled **KEY1** through **KEY4**) connected to four FPGA pins and four active-low LED's (**LED1** through **LED4**) connected to both FPGA pins and to header pins. We will use these on-board components for this lab.

The board also contains a 50 MHz clock oscillator, a power-indicating LED, a reset button, three voltage regulators supplying 3.3, 2.5 and 1.2 volts and 16 Mbit flash memory that stores the FPGA's power-on configuration[1]

The board is powered through a micro-USB port which must be connected to a USB port (or charger). *The USB-Blaster cannot power the board (even though it lights the power LED).*

The board is programmed through a JTAG connector which must be connected to the USB-Blaster from your ELEX 1117 parts kit. The schematic for the USB-Blaster is also available on the course website.

Quartus Prime and ModelSim

Quartus is a logic synthesizer – software that converts a description of your circuit to a file that can program the FPGA over the JTAG port. We will use System Verilog to design digital circuits.

Quartus can add a function called Signal Tap to your design that lets you to monitor signals within the FPGA while it operates. This lets you check if inputs, outputs and other signals are behaving as you expect.

ModelSim is a program for simulation of digital logic circuits. Simulations compile faster and provide more visibility into the behaviour of designs. Simulations are often used to test a circuit before it's implemented.

Procedure

Install the Software

Follow the instructions in the Software Installation and Use document to install Quartus, ModelSim, Cyclone IV device support and the USB-Blaster driver dated 2009-04-21 (version 2.4.16.0) from the course web site.

[1] The factory configuration blinks the on-board LEDs.

Simulation

Create two files, **lab0.sv** and **lab0_tb.sv** containing the code in the Appendix. This simple design turns on a number of LEDs corresponding to the key that was pressed. The testbench simulates pressing key1 for 40 ns.

Follow the simulation instructions to run the testbench. Take a screen capture of the signals displayed in the Wave window.

Synthesis

Follow the synthesis instructions to synthesize **lab0.sv** and program the FPGA.

The signals should be assigned to the following pins:

signal	pin
led[0]	39
led[1]	42
led[2]	43
led[3]	44
clk50	23

signal	pin
key1	89
key2	88
key3	90
key4	91

Signal Tap Logic Analyzer

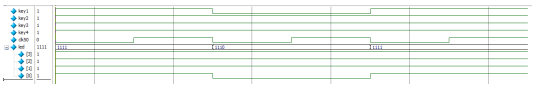
Follow the SignalTap instructions to add the SignalTap logic analyzer and monitor all of the **key** and **led** signals. Trigger off the falling edge of **key3**. Take a screen capture of the waveform for your report.

Practice Lab Report

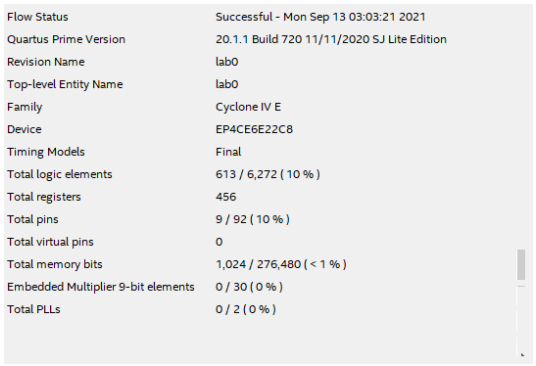
Follow the Report and Video Guidelines document to create a lab report that includes the following:

Screen Captures

- the ModelSim Wave window:

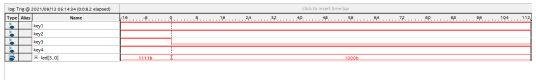


- the compilation report (**Window** > **Compilation Report**):



Flow Status	Successful - Mon Sep 13 03:03:21 2021
Quartus Prime Version	20.1.1 Build 720 11/11/2020 SJ Lite Edition
Revision Name	lab0
Top-level Entity Name	lab0
Family	Cyclone IV E
Device	EP4CE6E22C8
Timing Models	Final
Total logic elements	613 / 6,272 (10 %)
Total registers	456
Total pins	9 / 92 (10 %)
Total virtual pins	0
Total memory bits	1,024 / 276,480 (< 1 %)
Embedded Multiplier 9-bit elements	0 / 30 (0 %)
Total PLLs	0 / 2 (0 %)

- the Quartus SignalTap **Data** screen:



Block Diagram

A block diagram corresponding to the following Verilog:

```
module sample
  (
   input logic sel,
   input logic [3:0] a, b,
   output logic [3:0] y ) ;

   assign y = sel ? a : b ;

endmodule
```

Coding Guidelines

A corrected version of the following Verilog code[2]. The behaviour of the corrected version is immaterial.

```
module sample
  (
    input wire clk, enable,
    output reg [7:0] count, ecount
    ) ;

   always @(posedge clk) begin
      if ( !enable )
        count = count + 1'b1 ;
      end

      always @(posedge enable)
          ecount = ecount + 1'b1 ;

endmodule
```

[2]The code shown violates all six of the mandatory coding guidelines.

Submissions

Submit to the appropriate assignment folder on the course web site: (a) a report containing the items described above, and (b) a 10-second video of your FPGA board.

Appendix - Source Code

Sample Code

```
// lab0.sv
// Use pushbuttons to turn on 1 to 4 LEDs.
// Ed. Casas  2021-9-12

module lab0
  (
    input logic key1, key2, key3, key4,
    input logic clk50,
    output logic [3:0] led
    ) ;

   // each active-low key selects the appropriate
       ↪ number of active-low
   // LEDs

   assign led = !key1 ? 4'b1110 :
                !key2 ? 4'b1100 :
                !key3 ? 4'b1000 :
                !key4 ? 4'b0000 : 4'b1111 ;

endmodule
```

Sample Testbench

```
// lab0_tb.sv
// Testbench for Lab 0
// Ed. Casas  2021-9-12

module lab0_tb ;

 logic key1, key2, key3, key4, clk50 ;
 logic [3:0] led ;

  lab0 l0 ( key1, key2, key3, key4,
     clk50, led ) ;

  initial begin
     clk50 = '0 ;
     { key1, key2, key3, key4 } = 4'b1111 ;
     #40ns key1 = '0 ;
     #40ns key1 = '1 ;
     #40ns $stop() ;
  end ;

  always #20ns clk50 = !clk50 ;

endmodule
```