



**University of Tehran**  
College of Engineering  
School of Electrical & Computer Engineering

---

Experiment 2  
Sessions 3,4  
**Sequential Synthesis and FPGA Programming**

---

Digital Logic Laboratory  
ECE 045  
Laboratory Manual

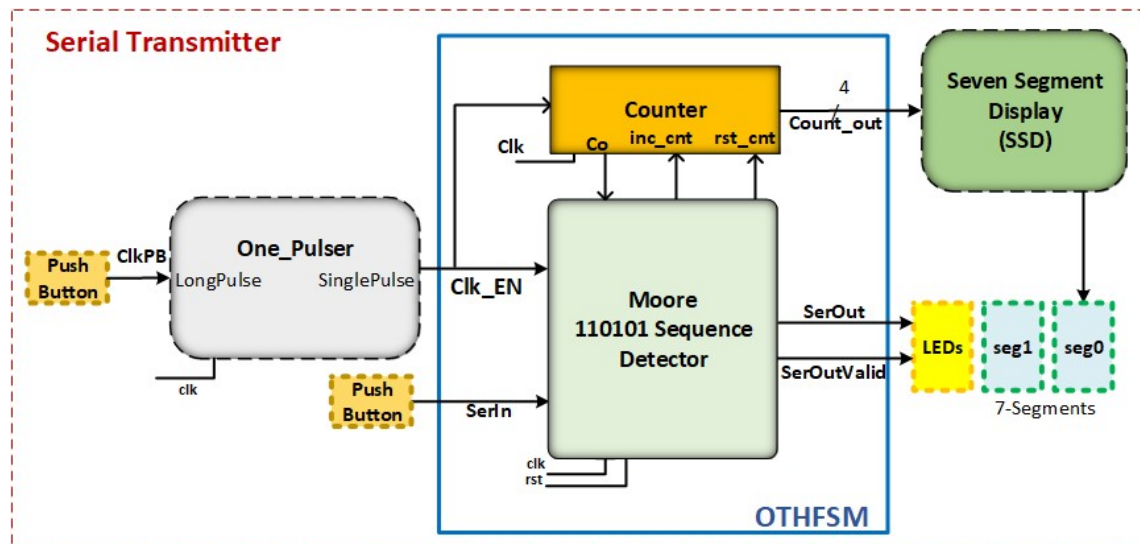
Spring 1402



## Contents

<b>Contents</b>	<b>1</b>
<b>Introduction</b>	<b>2</b>
<b>1 Serial Transmitter</b>	<b>2</b>
1.1 Onepulser . . . . .	3
1.2 Orthogonal Finite State Machine . . . . .	3
1.3 Seven Segment Display . . . . .	3
<b>Design Synthesis and FPGA Programming</b>	<b>3</b>
<b>2 Serial Transmitter Implementation</b>	<b>3</b>
<b>Appendices</b>	<b>5</b>
<b>A Using Quartus II</b>	<b>5</b>
A.1 Create the Project . . . . .	5
A.2 Compilation . . . . .	5
A.3 Pin Assignment . . . . .	5
A.4 Program the Design . . . . .	6
A.5 Examine the Timing and Resources . . . . .	6
<b>Acknowledgment</b>	<b>6</b>

Figure 1: Serial Transmitter



## Introduction

The first goal of this experiment is to introduce the concepts of state machines that are mostly used for controllers. The second goal is to get familiar with FPGA devices and implementation.

By the end of this experiment, you will learn:

- Concepts of state machines and Sequence detectors
- Huffman coding style
- Design simulation
- Synthesis
- FPGA programming and implementation

## 1 Serial Transmitter

In this experiment you are to design a serial transmitter circuit that searches on its *serIn* input for a start sequence of 110101 to begin transmitting its *serIn* on its *serOut*. When the start sequence is received, the *serOutValid* is asserted and for the next 10 clock cycles whatever appears on *serIn* will be transmitted on *serOut*. After the entire 10 bits are transmitted, the circuit returns to the state where search for the start sequence begins again. The initial state of this circuit is where it searches for the start sequence.

## 1.1 Onepulser

The overall design of this experiment is shown in figure 1. As shown three main components are needed for this design. The Onepulser module provides a clock-enable input for the counter and one for the sequence detector part. This common input (clkEn) is used for controlling the clock when the circuit is implemented on an FPGA board. The one-pulser connects to a push-button on your board (clkPB) and when pressed it creates a single pulse that is synchronized with the system clock. The output of this circuit connects to the clock enable input (clkEn) of the sequence detector and the counter.

1. Write the Verilog description of the One-Pulser module.
2. Test your design in Modelsim.

## 1.2 Orthogonal Finite State Machine

The second part of this experiment is called an Orthogonal FSM. It consists of a Moore state machine and a counter. When the clock Enable push-button is pushed the Moore sequence detector checks its input and decides which state to go. The sequence detector waits for the sequence of 110101 on its *serIn* input and when this is received, the *serOutvalid* output becomes one. The push button is pressed for every input that the state machine receives. After that, the counter will be enabled and will count for the next 10 consecutive clock cycles. During all these time the *serOutvalid* remains one.

1. Show the state diagram of the sequence detector.
2. Write the top-level Verilog description of the transmitter circuit. In the top-level description, write the Verilog description of the sequence detector and the Verilog description of the counter.
3. Write a testbench for the serial transmitter unit in Verilog that tests the correctness of your circuit and show the results in ModelSim.

## 1.3 Seven Segment Display

The last part of this experiment is a display module that is used for displaying the counter output on the seven segments of your FPGA board. Each seven segment receives a 4-bit input and displays the HEX value on its 7-bit output.

1. Write the Verilog description of the seven segment display module.

# Design Synthesis and FPGA Programming

## 2 Serial Transmitter Implementation

In this part you will use DE1 development board for your design implementation. Before implementing your design, make a top-level Verilog description that includes all three parts (Onepulser, sequence detector and the seven segment display modules) connected together.

1. Make a Quartus project.
2. Add the top-level Verilog codes to your project.
3. Connect the main FPGA clock to the clock input of your circuit.
4. Connect a push-button to the *serIn* of the serial transmitter circuit, another push-button to the input of the one-pulser circuit.
5. Perform the synthesis of your design.
6. Program the Cyclone II device.
7. For the output display, use an output LED for displaying the *serOutvalid* and one LED for *serOut*. Use one seven segment for displaying the counter output.
8. When applying a serial input, push the push-button corresponding to the *serIn* according to the value you want on the input, and then press the ClkPB push button once.
9. Attach images of implementation result into your report.

# Appendices

## Appendix A Using Quartus II

### A.1 Create the Project

1. Click on *File* ▷ *New Project Wizard*
2. Create an appropriate directory for your project and complete the form
3. Select the FPGA device as *Cyclone EP2C20F484C7* and then click *Finish*
4. From *File* ▷ *New* select the *Verilog HDL File*
5. Write your Verilog code in this window

### A.2 Compilation

- Select *Processing* ▷ *Start Compilation*
- Click on quick shortcut ►

There may be lots of warnings and some errors after compilation. The warnings are not so important while the errors should be completely removed.

### A.3 Pin Assignment

After you successfully compile your design, you should set your design with physical pins of the Cyclone II FPGA on DE1 board. You can find the position and the index of each pin from the DE1 user manual.

Figure 2: Creating new project

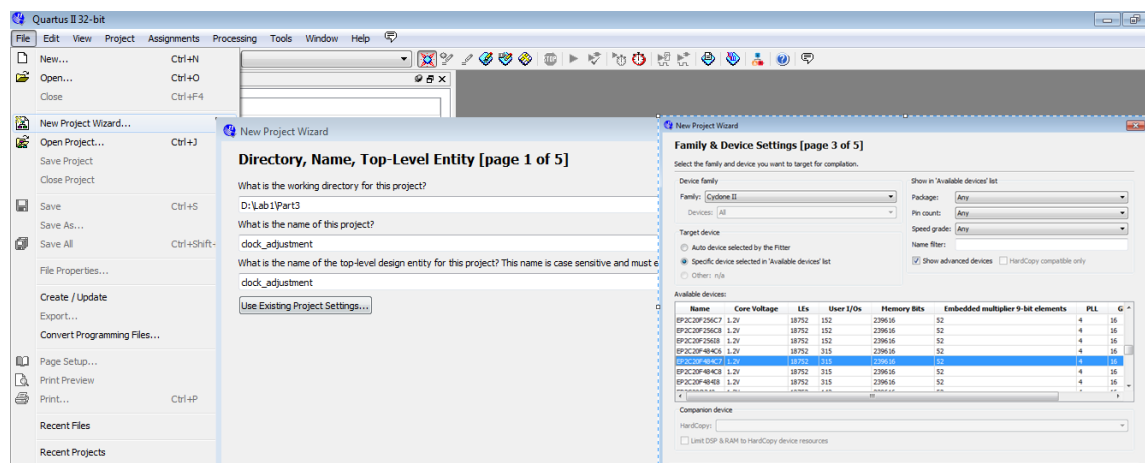
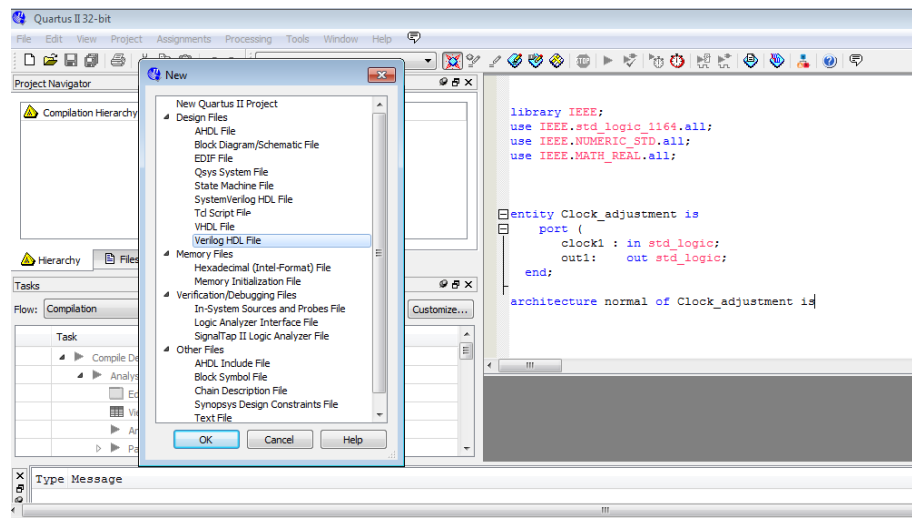



Figure 3: Writing Verilog HDL code




From *Assignments* ▷ *Pin Planner*, **Pin Planner window** will appear and you can select the corresponding locations from the list. When you are done with this, recompile your project.

#### A.4 Program the Design

1. Click on the  icon on the toolbar to open *Programmer*
2. Choose *USB Blaster* from the *Hardware setup* in the new window
3. Click on the start button

Your design will be programmed on the board when it is finished.

#### A.5 Examine the Timing and Resources

Now you can examine the resource usage of your design from *Processing* ▷ *Compilation Report* or by clicking on quick shortcut , and the timing from the *Tools* ▷ *TimeQuest Timing Analyzer*.

### Acknowledgment

This lab manual was prepared and developed by **Katayoon Basharkhah**, PHD student of Digital Systems at University of Tehran, under the supervision of professor Zain Navabi.

This manual has been revised and edited by **Zahra Jahanpeima**, PHD student of Digital Systems at University of Tehran.

Figure 4: Pin Planner window

