Experiment **2**  
 Sequential Synthesis and FPGA Device

Programming

Kamyar Rahmani

810199422

Mohammad Mashreghi

810199492

Abstract**—In this document we are going to design a serial transmitter and after that we are going to use quartus and then use FPGA and uplode in it.**

Keywords**— Serial Transmitter, Concepts of state machines and Sequence detectors, Huffman coding style, Design simulation, Synthesis, FPGA, Onepulser Seven Segment Display**

1. Introduction

In this document we are going to design a serial transmitter circuit that search on its serin input and when it find a sequence of 1011 and when it found the sequence, it starts to pass whatever is in serin input to serOut for 10 clock cycles and then go to the first state and try to find the sequence again.



Fig. 1 Serial transmitter

1. SEGUENTIAL SYNTHESIS AND FPGA DEVICE PROGRAMMING

1. Onepulser

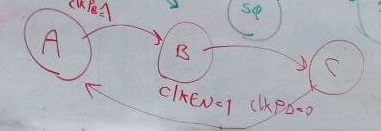


Fig. 2 State machine of onepulser

In this Expriment, we use FPGA and its frequency of the clock signal is about 50 MHz that is too high and we can’t control it, for example if set Serin on 1 and push the botton the duration is too long that FPGA sends a lot of cycle and all of them works with 1 that it's not what we want so if use only FPGA clk we can’t control our serial transmitter so we use one pulser.

One pulser converts our long pulse to a single pulse that we call it clk\_en and its duration is only one cycle.

So in this way when we push the botton our serial transmitter works only with one rising edge clk and not anymore :) .

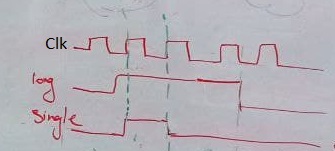


Fig. 3 Input and ouput waveforms of onepulser

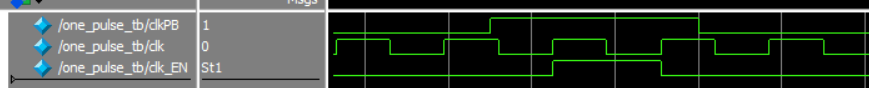


Fig. 4 Simulation of onepulser

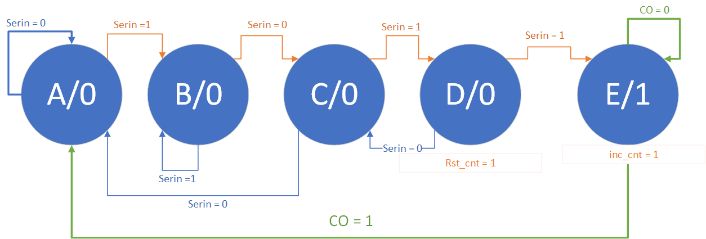
1. Orthogonal Finite State Machine

Fig. 5 State diagram of the sequence detector

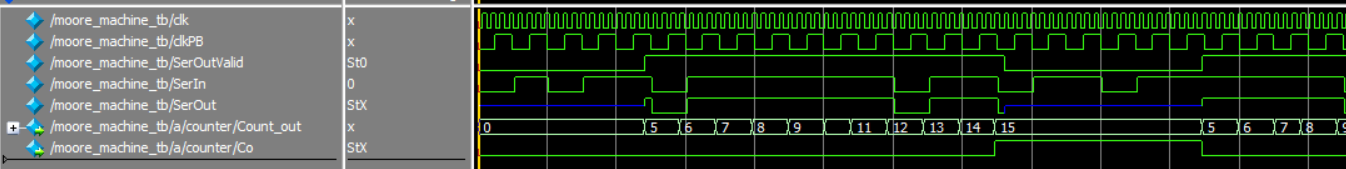
 As we can in state diagram (Fig. 5) when we see 1011 in Serin, the FPGA starts to count.

Fig. 6 Serial transmitter simulation

Well as you can see in the pic (Fig. 6), when the sequence detector dected 1011 the counter starts to count.

1. Seven Segment Display

In this part, the output of the SSD module is hex,  
and we have 4-bit counter and we must convert it to 7-bit somehow that the specific LED of SS light up and show us the number

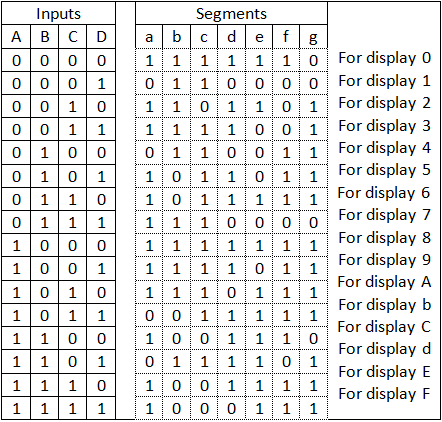


Fig. 7 Table

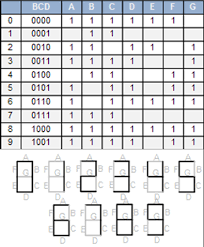


Fig. 8 positions of segments and Examples

1. SEGUENTIAL SYNTHESIS AND FPGA DEVICE PROGRAMMING

In this part, we simulated in Quartus II and after that we implemented it on FPGA.

In this part, we two LEDs for SerOut & SerOutValid and we use 3 push botton for clk\_PB , reset, SerIn and a SSD for counter.

And finally we connect our clk signal to the FPGA clk.

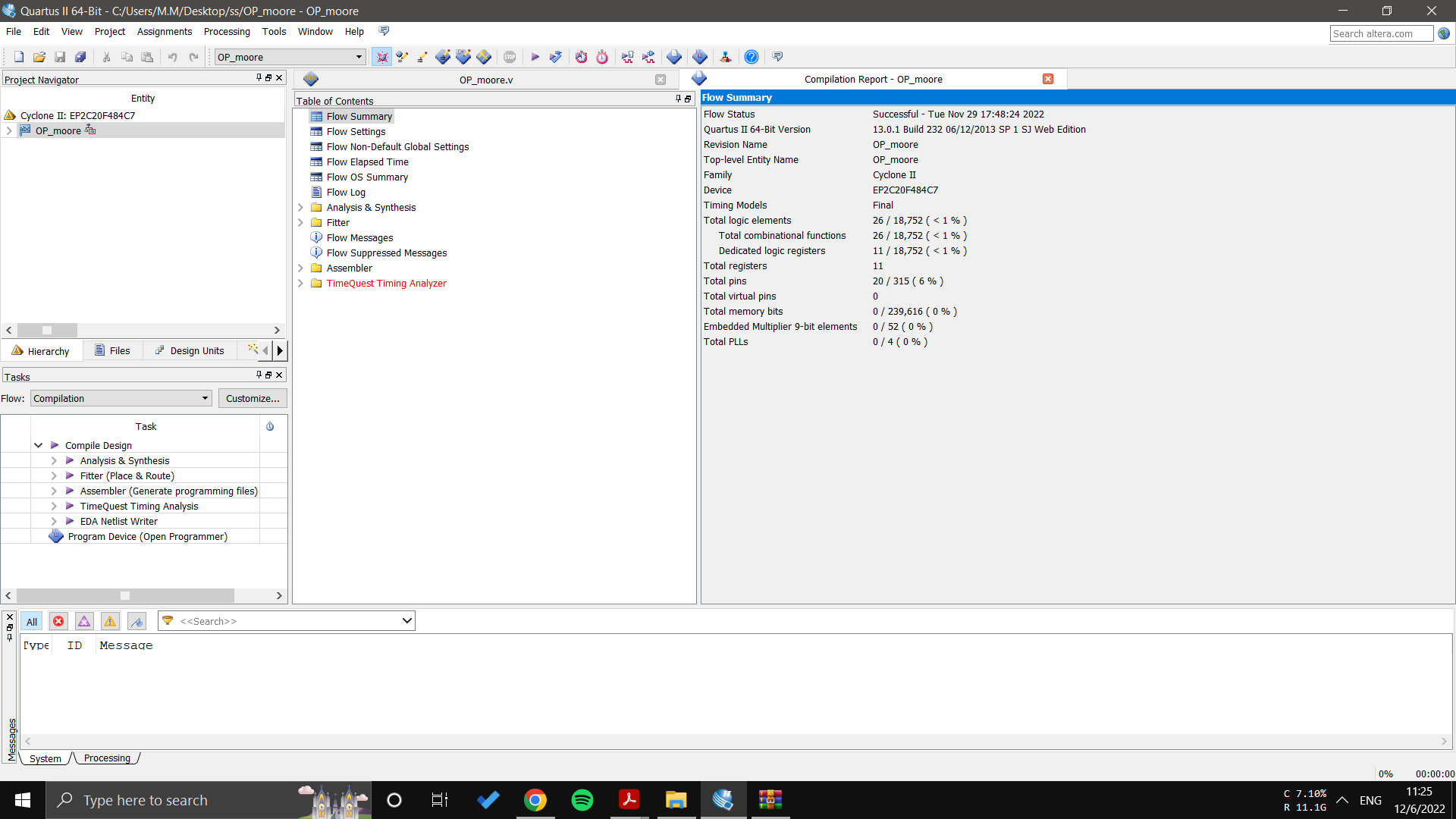


Fig. 6 Flow Summary

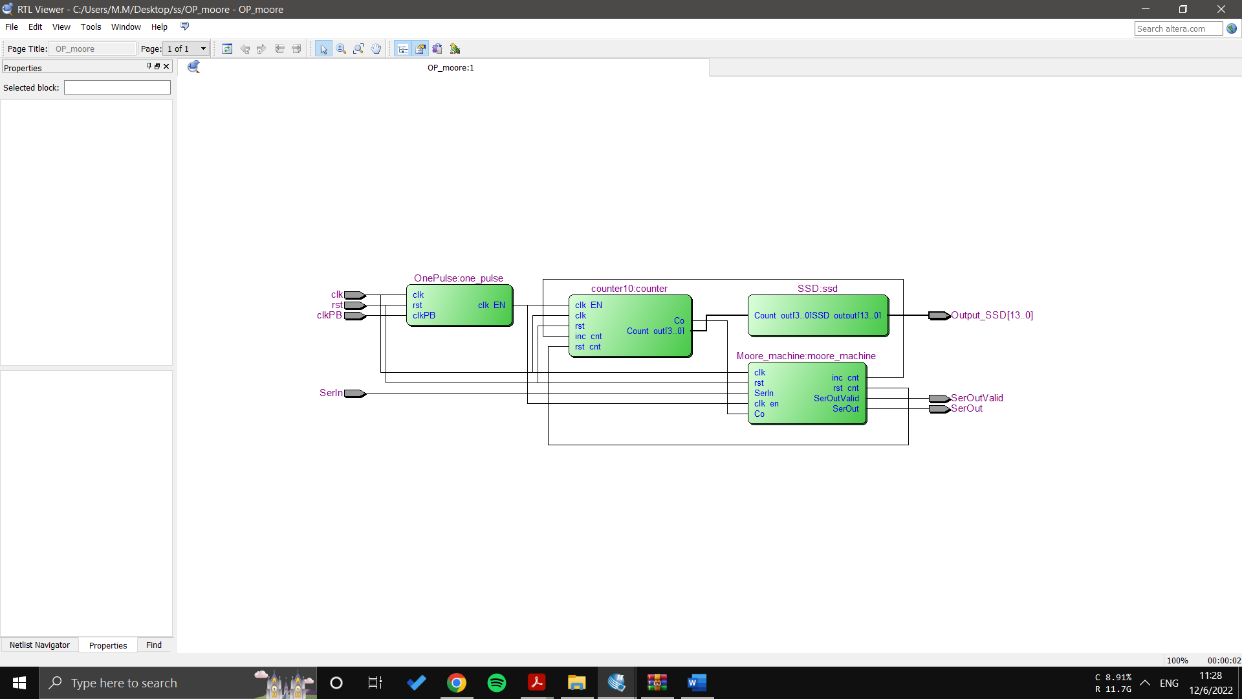


Fig. 7 RTL viewer

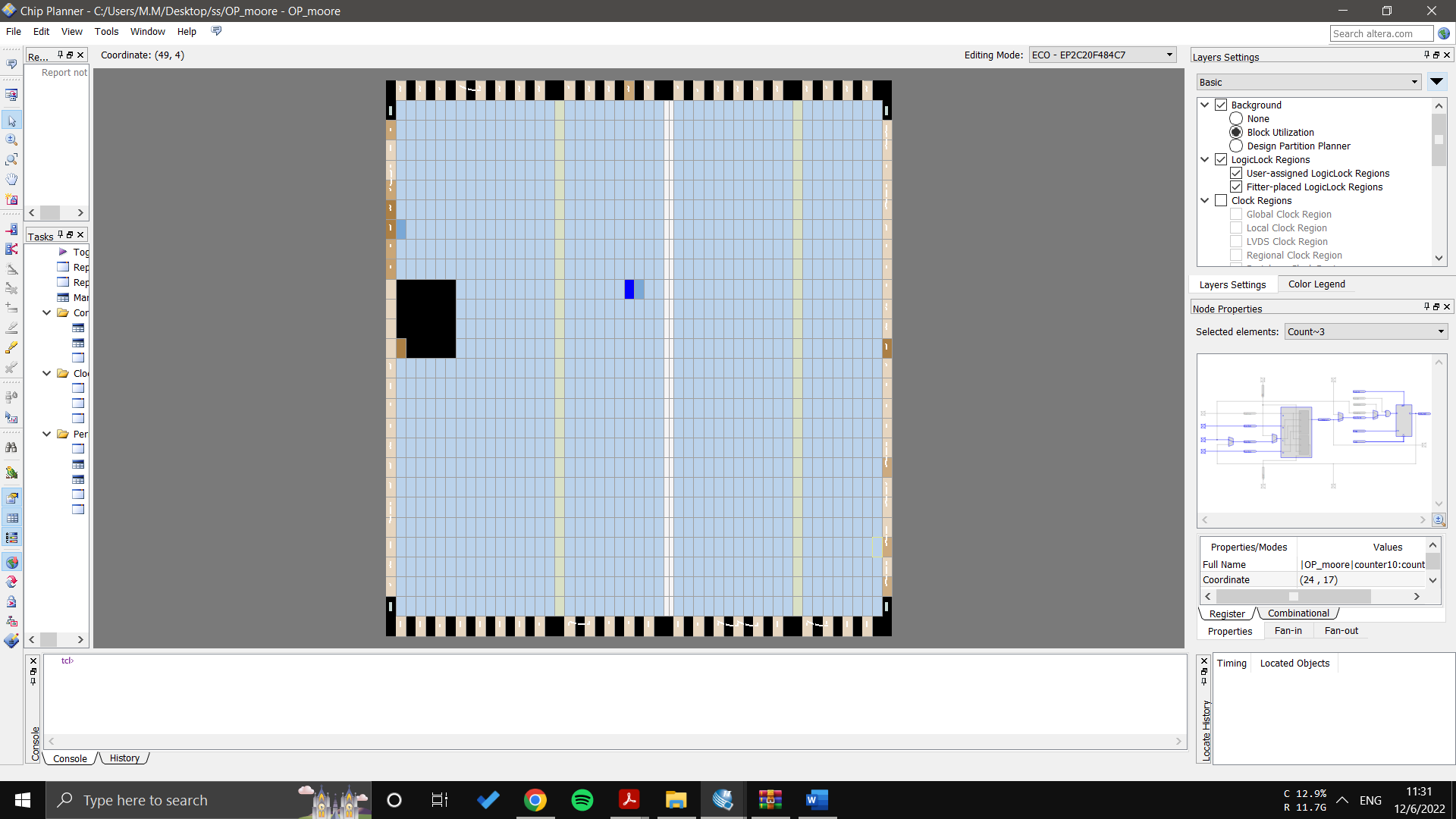


Fig. 8 Chip Planner

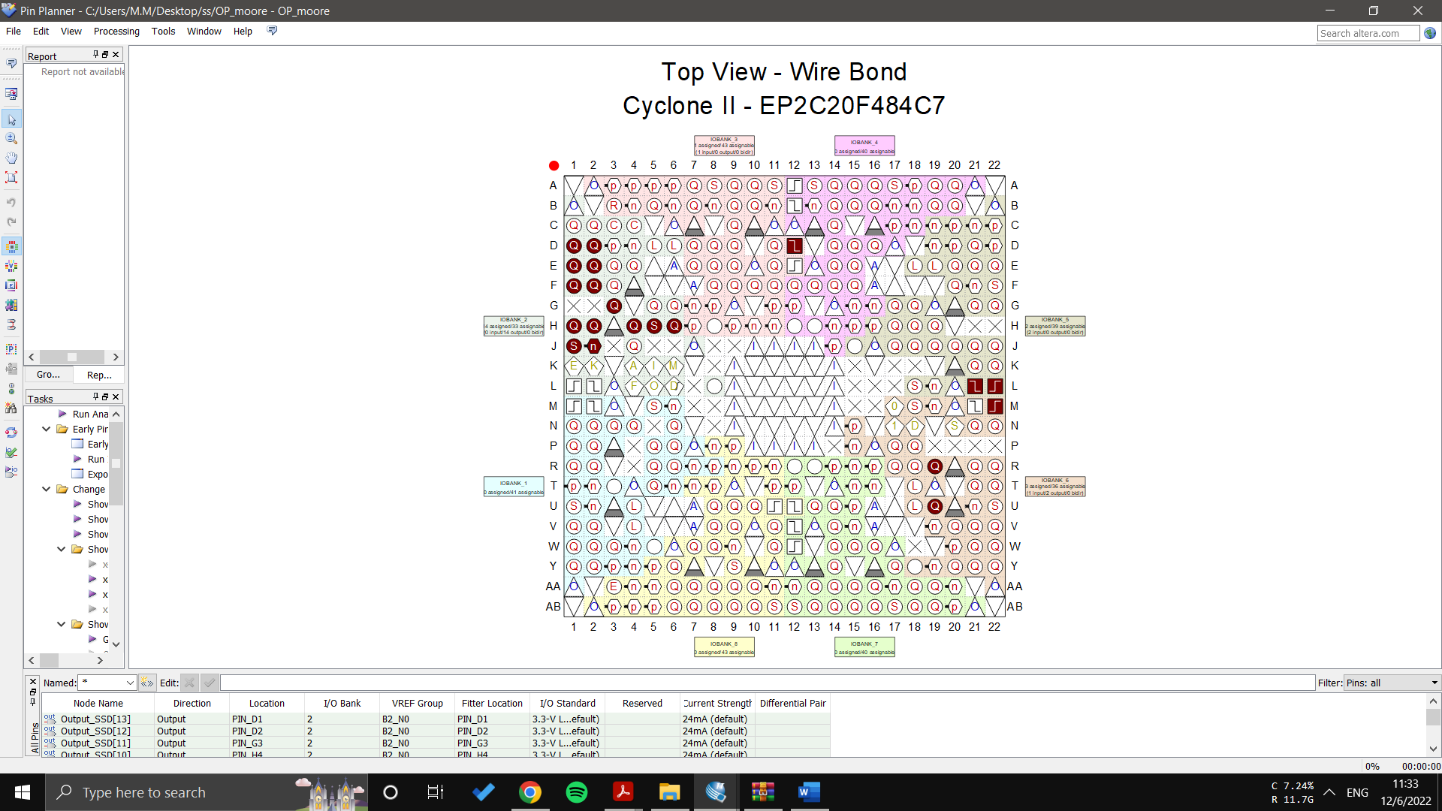


Fig. 9 Pin planner

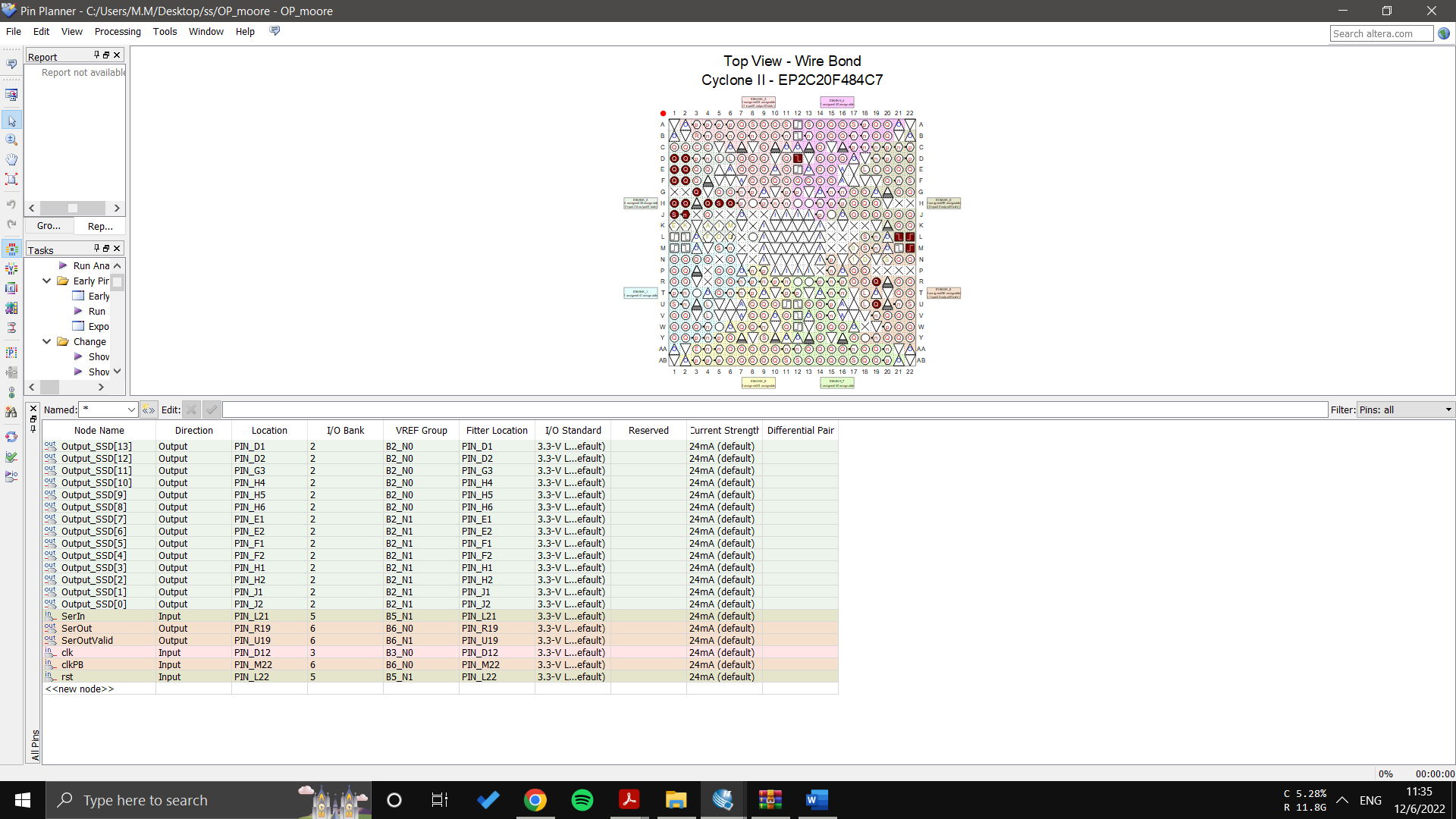


Fig. 10 Pin Connection

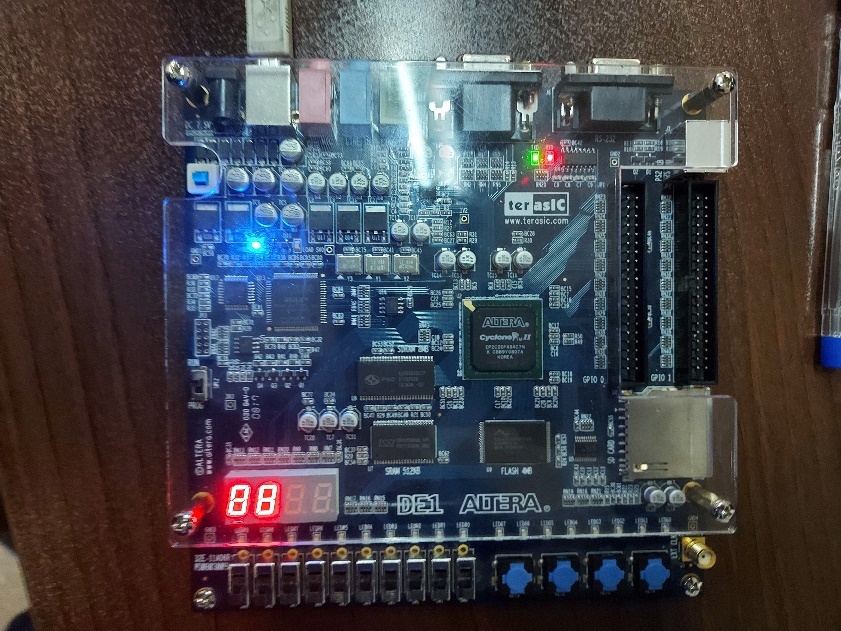


Fig. 11 FPGA before start

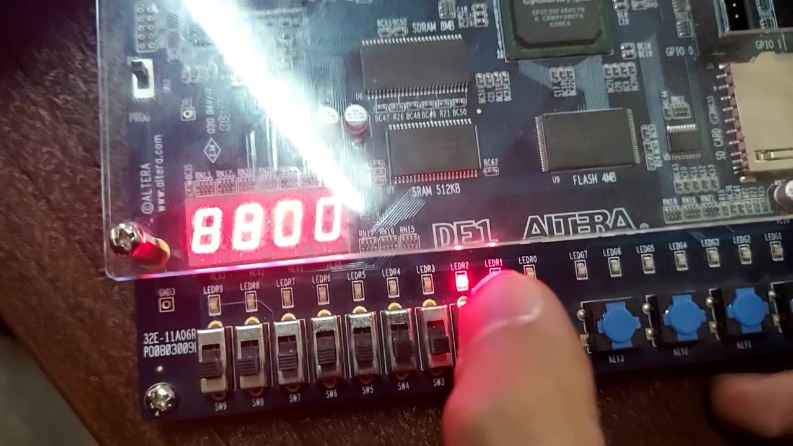


Fig. 12 FPGA detected the sequence

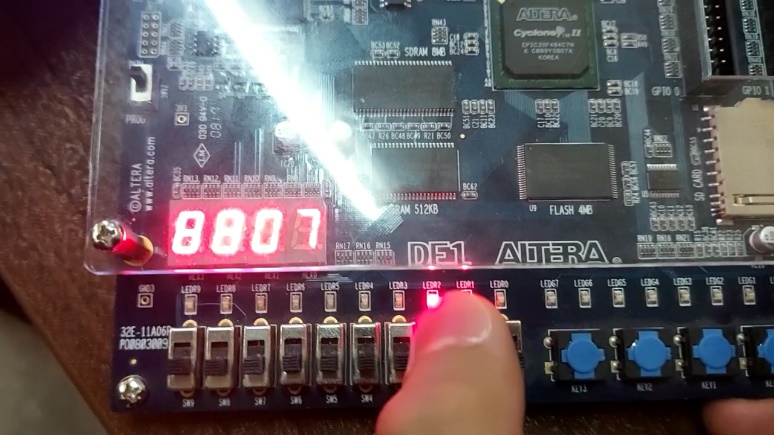


Fig. 12 FPGA while counting bits

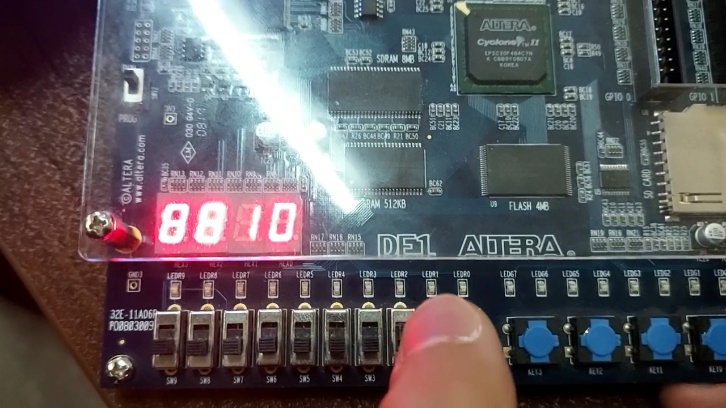


Fig. 12 FPGA when counting is finished

1. **Conclusion**

In this experiment, we programmed a FPGA with modelsim and quartus II and we designed onepulser, SSD, Counter , sequence detectore and combine them all together.