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**

I. 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.

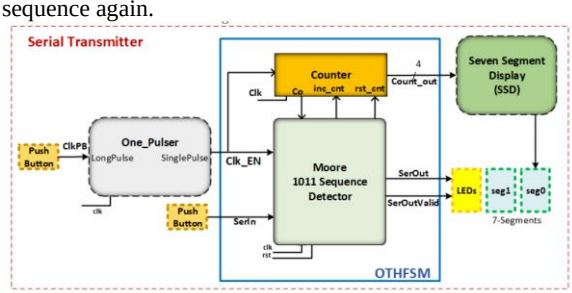

Fig. 1 Serial transmitter

II. Seguential Synthesis and FPGA Device Programming

A. Onepulser

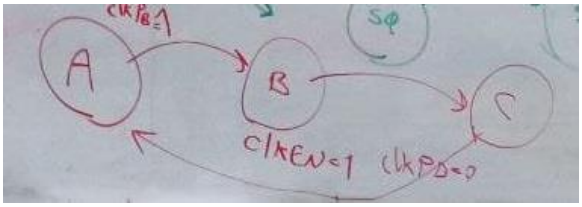

Fig. 2 State machine of onepulser

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 :) .

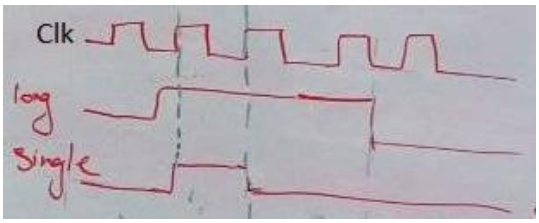

Fig. 3 Input and ouput waveforms of onepulser

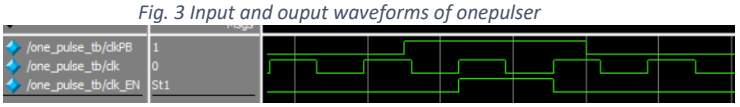

Fig. 4 Simulation of onepulser

B. *Orthogonal Finite State Machine*

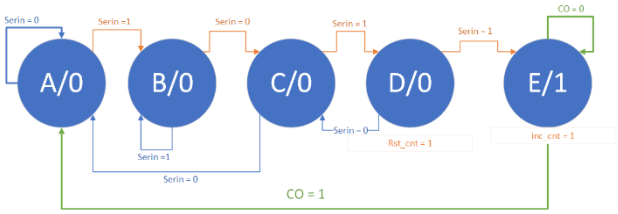

Fig. 5 State diagram of the sequence detector

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

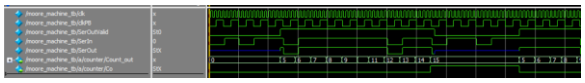

Fig. 6 Serial transmitter simulation

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

C. *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

Inputs				Segments							
A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	1	1	1	1	1	1	0	For display 0
0	0	0	1	0	1	1	0	0	0	0	For display 1
0	0	1	0	1	1	0	1	1	0	1	For display 2
0	0	1	1	1	1	1	1	0	0	1	For display 3
0	1	0	0	0	1	1	0	0	1	1	For display 4
0	1	0	1	1	0	1	1	0	1	1	For display 5
0	1	1	0	1	0	1	1	1	1	1	For display 6
0	1	1	1	1	1	1	0	0	0	0	For display 7
1	0	0	0	1	1	1	1	1	1	1	For display 8
1	0	0	1	1	1	1	1	0	1	1	For display 9
1	0	1	0	1	1	1	0	1	1	1	For display A
1	0	1	1	0	0	1	1	1	1	1	For display b
1	1	0	0	1	0	0	1	1	1	0	For display C
1	1	0	1	0	1	1	1	1	0	1	For display d
1	1	1	0	1	0	0	1	1	1	1	For display E
1	1	1	1	1	0	0	0	1	1	1	For display F

Fig. 7 Table

Fig. 8 positions of segments and Examples

III. 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.

Flow Status	Successful - Tue Nov 29 17:48:24 2022
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	OP_moore
Top-level Entity Name	OP_moore
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Total logic elements	26 / 18,752 (< 1 %)
Total combinational functions	26 / 18,752 (< 1 %)
Dedicated logic registers	11 / 18,752 (< 1 %)
Total registers	11
Total pins	20 / 315 (6 %)
Total virtual pins	0
Total memory bits	0 / 239,616 (0 %)
Embedded Multiplier 9-bit elements	0 / 52 (0 %)
Total PLLs	0 / 4 (0 %)

Fig. 6 Flow Summary

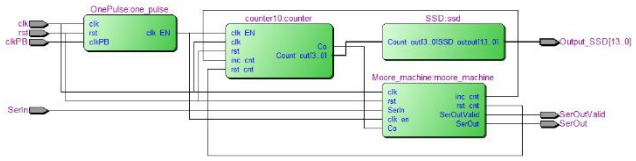

Fig. 7 RTL viewer

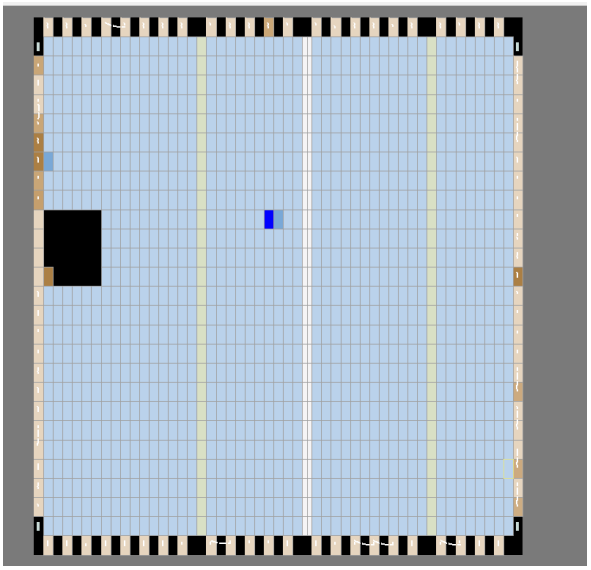

Fig. 8 Chip Planner

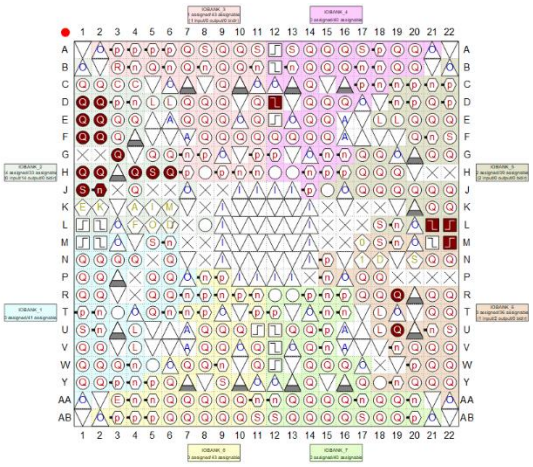

Fig. 9 Pin planner

Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Differer
Output_SSD[13]	Output	PIN_D1	2	B2_N0	PIN_D1	3.3-V L...efault)		24mA (default)	
Output_SSD[12]	Output	PIN_D2	2	B2_N0	PIN_D2	3.3-V L...efault)		24mA (default)	
Output_SSD[11]	Output	PIN_G3	2	B2_N0	PIN_G3	3.3-V L...efault)		24mA (default)	
Output_SSD[10]	Output	PIN_H4	2	B2_N0	PIN_H4	3.3-V L...efault)		24mA (default)	
Output_SSD[9]	Output	PIN_H5	2	B2_N0	PIN_H5	3.3-V L...efault)		24mA (default)	
Output_SSD[8]	Output	PIN_H6	2	B2_N0	PIN_H6	3.3-V L...efault)		24mA (default)	
Output_SSD[7]	Output	PIN_E1	2	B2_N1	PIN_E1	3.3-V L...efault)		24mA (default)	
Output_SSD[6]	Output	PIN_E2	2	B2_N1	PIN_E2	3.3-V L...efault)		24mA (default)	
Output_SSD[5]	Output	PIN_F1	2	B2_N1	PIN_F1	3.3-V L...efault)		24mA (default)	
Output_SSD[4]	Output	PIN_F2	2	B2_N1	PIN_F2	3.3-V L...efault)		24mA (default)	
Output_SSD[3]	Output	PIN_H1	2	B2_N1	PIN_H1	3.3-V L...efault)		24mA (default)	
Output_SSD[2]	Output	PIN_H2	2	B2_N1	PIN_H2	3.3-V L...efault)		24mA (default)	
Output_SSD[1]	Output	PIN_J1	2	B2_N1	PIN_J1	3.3-V L...efault)		24mA (default)	
Output_SSD[0]	Output	PIN_J2	2	B2_N1	PIN_J2	3.3-V L...efault)		24mA (default)	
SerIn	Input	PIN_L21	5	B5_N1	PIN_L21	3.3-V L...efault)		24mA (default)	
SerOut	Output	PIN_R19	6	B6_N0	PIN_R19	3.3-V L...efault)		24mA (default)	
SerOutValid	Output	PIN_U19	6	B6_N1	PIN_U19	3.3-V L...efault)		24mA (default)	
clk	Input	PIN_D12	3	B3_N0	PIN_D12	3.3-V L...efault)		24mA (default)	
clkPB	Input	PIN_M22	6	B6_N0	PIN_M22	3.3-V L...efault)		24mA (default)	
rst	Input	PIN_L22	5	B5_N1	PIN_L22	3.3-V L...efault)		24mA (default)	
<<new node>>									

Fig. 10 Pin Connection

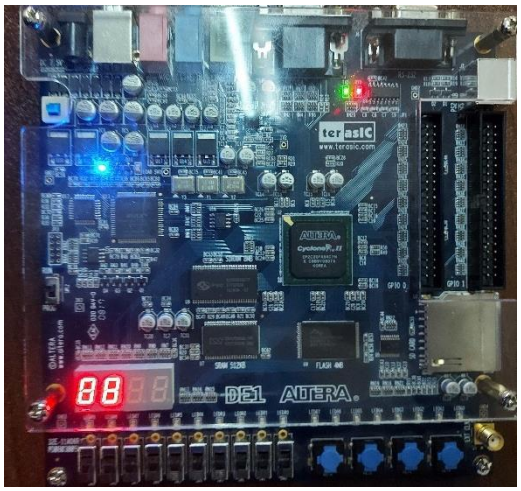

Fig. 11 FPGA before start

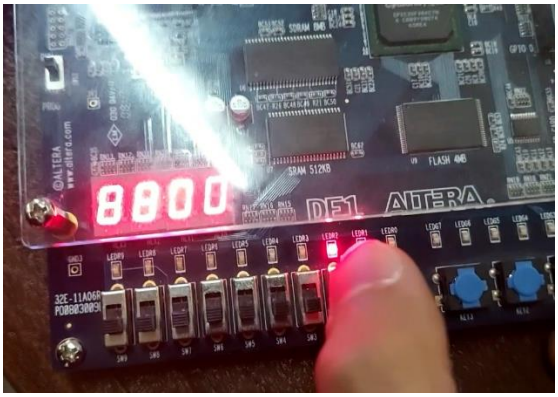

Fig. 12 FPGA detected the sequence

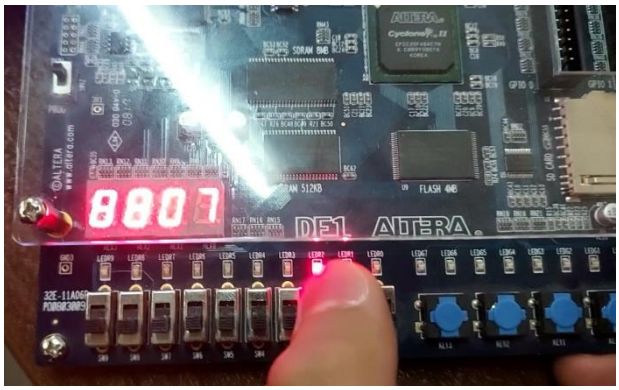

Fig. 12 FPGA while counting bits

Fig. 12 FPGA when counting is finished

IV. 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.