

# Experiment 2 - Sequential Synthesis and FPGA Programming

Ali Banihashemi - MohammadSadeqh Aghili

810100245 - 810100274

**Abstract:** In this experiment we got familiar with Concepts of state machines and Sequence detectors, FPGA devices and implementation and designing a MSSD component.

**Keywords:** state machine – Sequence detector – FPGA – MSSD– Huffman coding

## I. EXPERIMENTS

### 2-1. Onepulser

This component is used to freeze the clock until we make changes to the serIn and then after releasing it, circuit will use our new input.

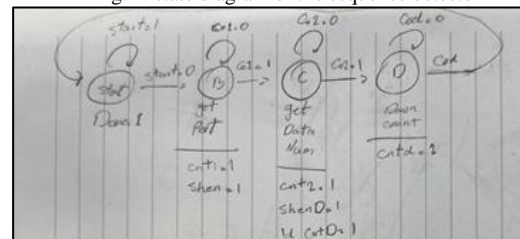
Fig.1: Verilog description of the one-pulser module

```
module onePulser(clkPB,clk,rst,clkEn);
input clkPB,clk,rst;
output reg clkEn;
parameter[1:0] A= 2'b00,B = 2'b01,C = 2'b10;
reg[1:0]ns,ps;
always@(ps,clkPB)
begin
ns = 2'b00;
case(ps)
A : ns = clkPB?B:A;
B : ns = C;
C : ns = clkPB?C:A;
endcase
end
always@(ps)
begin
case(ps)
A:clkEn = 1'b0;
B:clkEn = 1'b1;
C:clkEn = 1'b0;
endcase
end
always@(posedge clk , posedge rst)
begin
if(rst)
ps <= 2'b0;
else
ps <= ns;
end
end
endmodule
```

### 2-2. Finite State Machine and the counters

This state machine is implemented using Huffman coding style, in each state we decide to whether change present state or not based on the inputs of the state machine. Some of inputs are generated in the data-path such as counters' carry-out and some are inputs of data-path as well(serIn).

Fig. 2: state diagram of the sequence detector



Then writing the Verilog description of the FSM module is required. This description includes two always blocks. One of them changes next state and control signals based on present state. The other one changes present state to next state based on clock and reset signals.

Fig.3: Verilog description of the FSM module

```
module controller(input serIn,co1,co2,coD,clk,
rst,clkEn, output reg cnt1,cnt2,cntD,output ldcntD,
output reg shen,shenD,output seroutvalid, done);
parameter[1:0] A= 2'b00,B = 2'b01,
C = 2'b10, D = 2'b11;
reg[1:0]ns,ps;
always@(ps,co1,co2,coD,serIn)
begin
ns = 2'b00;
{cnt1,cnt2,cntD,shen,shenD} = 5'b0;
case(ps)
A :begin ns = serIn ? A : B; end
B :begin ns = co1 ? C : B;
{cnt1,shen} = 2'b11; end
C :begin ns = co2 ? D : C;
{cnt2,shenD} = 2'b11; end
D :begin ns = coD ? A : D; {cntD} = 1'b1;
end
endcase
end
always@(posedge clkEn , posedge rst)
begin
if(rst)
ps <= 2'b0;
else
ps <= ns;
end
assign ldcntD = co2;
assign done = coD;
assign seroutvalid = (ps==D && ~coD);
endmodule
```

The two counters has the same logic but one of them count to 2 (decimal) and the other one counts two 4 , because there are 2 bits that represent the port number and 4 bits that determine number of inputs. So first one can be 1-bit and the second one can be 2-bits. In each one carry-out signal shows if it reached the end of counting or not:

Fig. 4: Verilog description of the counter module

```
module cnt1bit(cnt,clk,rst,clkEn,co);
input cnt,clk,rst,clkEn;
output co;
reg count;
always@(posedge clkEn, posedge rst)
begin
count = rst ? 1'b0:~cnt?count:
(count==1'b1)?1'b0:count+1;
end
assign co = (count == 1'b1 && cnt)?
1'b1 : 1'b0;
endmodule

module cnt2bit(cnt,clk,rst,clkEn,co);
input cnt,clk,rst,clkEn;
output co;
reg [2:0] count;
always@(posedge clkEn, posedge rst)
begin
count = rst ? 3'b000:~cnt?count:
(count==3'b100)?3'b000:count+1;
end
assign co = (count == 3'b100 && cnt)?
1'b1 : 1'b0;
endmodule
```

### 2-3. Shift Registers and Demultiplexers

Shift registers are used to store port and data number so one of them is storing a 2-bit value and the other one is storing a 4-bit value, but their logic are exactly the same and they are just different in number of bits.

Fig. 5: Verilog description of the 4-bit shift register module

```
module shr4bit(serIn, shen, clk, rst, clkEn, portnum);
input serIn, shen, clk, rst, clkEn;
output [3:0] portnum;
reg [3:0] shift;
always@(posedge clkEn, posedge rst)
begin
if(rst)
shift <= 4'b0;
else if(shen)
shift <= {shift[2:0], serIn};
end
assign portnum=shift;
endmodule
```

In this DMUX, there is a 2-bit select signal which chooses one of the 4 available ports, then we assign serIn input to the selected port and the Z-value to the other 3 ports.

Fig. 6: Verilog description of the demultiplexer module

```
module Dmux(input [1:0] sel,
serIn, output [3:0] p);
assign p = (sel==2'b00) ? {serIn,3'bz} :
(sel==2'b01) ? {1'bz,serIn,2'bz} :
(sel==2'b10) ? {2'bz,serIn,1'bz} :
{3'bz,serIn};
endmodule
```

### 2-4. Seven Segment Display

In the Verilog description of this module, there is an always block we for every change of 4-bit input which is a 4-bit binary number, changes the 7-bit output that this 7-bit number represent seven segment display number.

Fig. 7: Verilog description of the seven-segment display module

```
module SSD(input[3:0] in, output reg [6:0] out);
always@(in) begin
out = 7'b0;
case(in)
4'b0000: out = 7'b1000000;
4'b0001: out = 7'b11111001;
4'b0010: out = 7'b01001100;
4'b0011: out = 7'b01100000;
4'b0100: out = 7'b0011001;
4'b0101: out = 7'b0010010;
4'b0110: out = 7'b0000010;
4'b0111: out = 7'b11111000;
4'b1000: out = 7'b0000000;
4'b1001: out = 7'b0010000;
4'b1010: out = 7'b0000100;
4'b1011: out = 7'b0000011;
4'b1100: out = 7'b1000110;
4'b1101: out = 7'b0100001;
4'b1110: out = 7'b0000110;
4'b1111: out = 7'b0001110;
endcase
end
endmodule
```

### 3. MSSD Implementation

To make a top-level design, there are two major parts; controller and data-path. Controller has been written so

after writing data-path, these two should be wired together in a top-level design which is called MSSD.

Data-path includes all the components such as counters and shift registers and it wire them together:

Fig. 8: Verilog description of the data-path

```
module datapath(input clkPB,clk,rst,
cnt1,cnt2,cntD,ldcntD,shen,
shenD,seroutvalid,done,serIn,
output co1,co2,coD,output [6:0] ssdout
,output [3:0] p,output clkEn);
onePulser onep(clkPB,clk,rst,clkEn);
wire[1:0] portnum;
cnt2bit cnt22(cnt2,clk,rst,clkEn,co2);
shr2bit portsh(serIn,shen,clk,rst,clkEn,portnum);
wire[3:0] portnum2;
shr4bit datash(serIn,shenD,clk,rst,clkEn,portnum2);
cnt1bit cnt11(cnt1,clk,rst,clkEn,co1);
wire[3:0] count2;
Dcnt dcnt(cntD,portnum2,clk,rst,
clkEn,ldcntD,coD,count2);
Dmux dmux(portnum,serIn,p);
SSD ssd(count2,ssdout);
endmodule
```

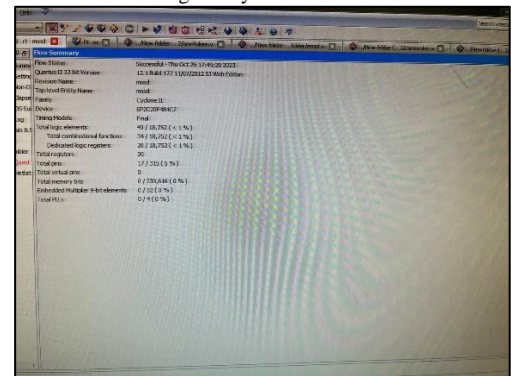
Then to finish the top level design, only wiring the data-path which contains all the components with the controller is required.

Fig. 9: top-level Verilog description

```
module mssd(input clk,rst,serIn,clkPB,
output done,seroutvalid,
output [6:0] ssdout,output [3:0] p);
wire cnt1,cnt2,cntD,ldcntD,shen,shenD,co1,co2,coD;
datapath dt(clkPB,clk,rst,cnt1,cnt2,cntD,
ldcntD,shen,shenD,seroutvalid,done,serIn,
co1,co2,coD,ssdout,p,clkEn);
controller ct( serIn,co1,co2,coD,clk,rst,clkEn,
cnt1,cnt2,cntD,ldcntD,
shen,shenD,seroutvalid,done);
endmodule
```

Next step is performing the synthesis of the top level design in the Quartus.

Fig. 10: synthesis result



Then pin assignments should be done. In this step signals that are in Verilog description of MSSD, will be assigned to actual components on a physical board which in this experiment is a Cyclone II device:

Report not available

Run Analysis and Optimization

- Early Pin Placement
- Run I/O Assignment and Analysis
- Export the Assignments
- Change View

Name	Direction	Location	IO Style	VIO Drive	Filter Location	IO Standard	Filtered
PIN_001	External	PCB_001	6	10.2k	PIN_001	3.3V-tx default	200k (default)
PIN_002	External	PCB_002	6	10.2k	PIN_002	3.3V-tx default	200k (default)
PIN_003	External	PCB_003	6	10.2k	PIN_003	3.3V-tx default	200k (default)
PIN_004	External	PCB_004	6	10.2k	PIN_004	3.3V-tx default	200k (default)
PIN_005	External	PCB_005	6	10.2k	PIN_005	3.3V-tx default	200k (default)
PIN_006	External	PCB_006	6	10.2k	PIN_006	3.3V-tx default	200k (default)
PIN_007	External	PCB_007	6	10.2k	PIN_007	3.3V-tx default	200k (default)
PIN_008	External	PCB_008	6	10.2k	PIN_008	3.3V-tx default	200k (default)
PIN_009	External	PCB_009	6	10.2k	PIN_009	3.3V-tx default	200k (default)
PIN_010	External	PCB_010	6	10.2k	PIN_010	3.3V-tx default	200k (default)
PIN_011	External	PCB_011	6	10.2k	PIN_011	3.3V-tx default	200k (default)
PIN_012	External	PCB_012	6	10.2k	PIN_012	3.3V-tx default	200k (default)
PIN_013	External	PCB_013	6	10.2k	PIN_013	3.3V-tx default	200k (default)
PIN_014	External	PCB_014	6	10.2k	PIN_014	3.3V-tx default	200k (default)
PIN_015	External	PCB_015	6	10.2k	PIN_015	3.3V-tx default	200k (default)
PIN_016	External	PCB_016	6	10.2k	PIN_016	3.3V-tx default	200k (default)
PIN_017	External	PCB_017	6	10.2k	PIN_017	3.3V-tx default	200k (default)
PIN_018	External	PCB_018	6	10.2k	PIN_018	3.3V-tx default	200k (default)
PIN_019	External	PCB_019	6	10.2k	PIN_019	3.3V-tx default	200k (default)
PIN_020	External	PCB_020	6	10.2k	PIN_020	3.3V-tx default	200k (default)
PIN_021	External	PCB_021	6	10.2k	PIN_021	3.3V-tx default	200k (default)
PIN_022	External	PCB_022	6	10.2k	PIN_022	3.3V-tx default	200k (default)
PIN_023	External	PCB_023	6	10.2k	PIN_023	3.3V-tx default	200k (default)
PIN_024	External	PCB_024	6	10.2k	PIN_024	3.3V-tx default	200k (default)
PIN_025	External	PCB_025	6	10.2k	PIN_025	3.3V-tx default	200k (default)
PIN_026	External	PCB_026	6	10.2k	PIN_026	3.3V-tx default	200k (default)
PIN_027	External	PCB_027	6	10.2k	PIN_027	3.3V-tx default	200k (default)
PIN_028	External	PCB_028	6	10.2k	PIN_028	3.3V-tx default	200k (default)
PIN_029	External	PCB_029	6	10.2k	PIN_029	3.3V-tx default	200k (default)
PIN_030	External	PCB_030	6	10.2k	PIN_030	3.3V-tx default	200k (default)
PIN_031	External	PCB_031	6	10.2k	PIN_031	3.3V-tx default	200k (default)
PIN_032	External	PCB_032	6	10.2k	PIN_032	3.3V-tx default	200k (default)
PIN_033	External	PCB_033	6	10.2k	PIN_033	3.3V-tx default	200k (default)
PIN_034	External	PCB_034	6	10.2k	PIN_034	3.3V-tx default	200k (default)
PIN_035	External	PCB_035	6	10.2k	PIN_035	3.3V-tx default	200k (default)
PIN_036	External	PCB_036	6	10.2k	PIN_036	3.3V-tx default	200k (default)
PIN_037	External	PCB_037	6	10.2k	PIN_037	3.3V-tx default	200k (default)
PIN_038	External	PCB_038	6	10.2k	PIN_038	3.3V-tx default	200k (default)
PIN_039	External	PCB_039	6	10.2k	PIN_039	3.3V-tx default	200k (default)
PIN_040	External	PCB_040	6	10.2k	PIN_040	3.3V-tx default	200k (default)
PIN_041	External	PCB_041	6	10.2k	PIN_041	3.3V-tx default	200k (default)
PIN_042	External</						

In this experiment, we learnt Concepts of state machines and Sequence detectors, Designing simulations, Synthesis and FPGA programming and implementation.

[1] Katayoon Basharkhah and Zain Navabi, *Digital Logic Laboratory*, University of Tehran, Fall 1402.

The screenshot shows the Samsung UCC7000 TV service menu. The 'Run' menu is expanded, showing options like 'Run Analysis and Fabrication', 'Early Pin Planning', 'Early Pin Placement', 'Run I/O Assignment Analysis', and 'Export Pin Assignments'. The 'Run I/O Assignment Analysis' option is selected, displaying a table of pin assignments for various components.

Pin Name	Device	Location	IO Type	Width (mm)	Pin Function	IO Standard	Category	Current Strength	Pin Type
U1_56	TV-IC	FP1_11	7	0.2540	FP1_11	5.0 V I/O - 40mA	General	40mA	FP1_11
U1_56	TV-IC	FP1_12	7	0.2540	FP1_12	5.0 V I/O - 40mA	General	40mA	FP1_12
U1_56	TV-IC	FP1_13	6	0.2540	FP1_13	5.0 V I/O - 40mA	General	40mA	FP1_13
U1_56	TV-IC	FP1_14	6	0.2540	FP1_14	5.0 V I/O - 40mA	General	40mA	FP1_14
U1_56	TV-IC	FP1_15	6	0.2540	FP1_15	5.0 V I/O - 40mA	General	40mA	FP1_15
U1_56	TV-IC	FP1_16	6	0.2540	FP1_16	5.0 V I/O - 40mA	General	40mA	FP1_16
U1_56	TV-IC	FP1_17	6	0.2540	FP1_17	5.0 V I/O - 40mA	General	40mA	FP1_17
U1_56	TV-IC	FP1_18	6	0.2540	FP1_18	5.0 V I/O - 40mA	General	40mA	FP1_18
U1_56	TV-IC	FP1_19	6	0.2540	FP1_19	5.0 V I/O - 40mA	General	40mA	FP1_19
U1_56	TV-IC	FP1_20	6	0.2540	FP1_20	5.0 V I/O - 40mA	General	40mA	FP1_20
U1_56	TV-IC	FP1_21	6	0.2540	FP1_21	5.0 V I/O - 40mA	General	40mA	FP1_21
U1_56	TV-IC	FP1_22	6	0.2540	FP1_22	5.0 V I/O - 40mA	General	40mA	FP1_22
U1_56	TV-IC	FP1_23	6	0.2540	FP1_23	5.0 V I/O - 40mA	General	40mA	FP1_23
U1_56	TV-IC	FP1_24	6	0.2540	FP1_24	5.0 V I/O - 40mA	General	40mA	FP1_24
U1_56	TV-IC	FP1_25	6	0.2540	FP1_25	5.0 V I/O - 40mA	General	40mA	FP1_25
U1_56	TV-IC	FP1_26	6	0.2540	FP1_26	5.0 V I/O - 40mA	General	40mA	FP1_26
U1_56	TV-IC	FP1_27	6	0.2540	FP1_27	5.0 V I/O - 40mA	General	40mA	FP1_27
U1_56	TV-IC	FP1_28	6	0.2540	FP1_28	5.0 V I/O - 40mA	General	40mA	FP1_28
U1_56	TV-IC	FP1_29	6	0.2540	FP1_29	5.0 V I/O - 40mA	General	40mA	FP1_29
U1_56	TV-IC	FP1_30	6	0.2540	FP1_30	5.0 V I/O - 40mA	General	40mA	FP1_30
U1_56	TV-IC	FP1_31	6	0.2540	FP1_31	5.0 V I/O - 40mA	General	40mA	FP1_31
U1_56	TV-IC	FP1_32	6	0.2540	FP1_32	5.0 V I/O - 40mA	General	40mA	FP1_32
U1_56	TV-IC	FP1_33	6	0.2540	FP1_33	5.0 V I/O - 40mA	General	40mA	FP1_33
U1_56	TV-IC	FP1_34	6	0.2540	FP1_34	5.0 V I/O - 40mA	General	40mA	FP1_34
U1_56	TV-IC	FP1_35	6	0.2540	FP1_35	5.0 V I/O - 40mA	General	40mA	FP1_35
U1_56	TV-IC	FP1_36	6	0.2540	FP1_36	5.0 V I/O - 40mA	General	40mA	FP1_36
U1_56	TV-IC	FP1_37	6	0.2540	FP1_37	5.0 V I/O - 40mA	General	40mA	FP1_37
U1_56	TV-IC	FP1_38	6	0.2540	FP1_38	5.0 V I/O - 40mA	General	40mA	FP1_38
U1_56	TV-IC	FP1_39	6	0.2540	FP1_39	5.0 V I/O - 40mA	General	40mA	FP1_39
U1_56	TV-IC	FP1_40	6	0.2540	FP1_40	5.0 V I/O - 40mA	General	40mA	FP1_40
U1_56	TV-IC	FP1_41	6	0.2540	FP1_41	5.0 V I/O - 40mA	General	40mA	FP1_41
U1_56	TV-IC	FP1_42	6	0.2540	FP1_42	5.0 V I/O - 40mA	General	40mA	FP1_42
U1_56	TV-IC	FP1_43	6	0.2540	FP1_43	5.0 V I/O - 40mA	General	40mA	FP1_43
U1_56	TV-IC	FP1_44	6	0.2540	FP1_44	5.0 V I/O - 40mA	General	40mA	FP1_44
U1_56									

Fig. 13: Physical board visualization