Attachment_file:

Experiment **2**
Sequential Synthesis and FPGA Device Programming

Verilog code

Kamyar Rahmani - Mohammad Mashreghi

810199422 - 810199492

A. Onepulser

```
`timescale 1 ns/1ns
module OnePulse(input clk,rst , clkPB, output clk_EN);
  reg [1:0] ns,ps;
  parameter [2:0] A = 0 ,B = 1 , C = 2  ;
  always @(ps,clkPB) begin
      ns = A;
      case (ps)
          A : ns = clkPB ? B : A;
          B : ns = C ;
          C : ns = clkPB ? C : A;
          default: ns = A;
      endcase
  end
  assign clk_EN = (ps == B)? 1'b1 :1'b0;
  always @(posedge clk,posedge rst) begin
    if(rst)
      ps <= A;
    else
      ps <= ns;
  end
endmodule
```

B. Onepulse-Testbench

```
module one_pulse_tb;
  reg clkPB , clk , rst ;
  wire clk_EN;
  wire [1:0]state;
 OnePulse must( clk , rst, clkPB, clk_EN);
  initial begin
    #6
    clkPB = 1'b0;
    rst = 1'b1;
    clk = 1'b0;
  end
  initial #15 rst = 1'b0;
  always #13  clk = ~clk;
  always #50 clkPB = ~clkPB;
  initial begin
    #1000
  $stop;
  end
endmodule
```

C. Moore_Machine

```
`timescale 1 ns/1ns
module Moore_machine(input clk ,rst , SerIn , clk_en , Co ,
output reg inc_cnt , rst_cnt , SerOutValid ,output SerOut );
  reg [2:0] ps , ns;
  parameter [2:0] A = 0 ,B = 1 , C = 2 , D = 3 , E = 4 ;
  always @(clk_en , ps, SerIn , Co) begin
      ns = A ;
      rst_cnt = 0;
      inc_cnt = 0;
      SerOutValid = 0;

      case(ps)
          A: ns = clk_en ? (SerIn? B : A) : A;

          B: ns = clk_en ? (SerIn? B : C) : B;

          C: ns = clk_en ? (SerIn? D : A) : C;

          D: begin ns = clk_en ? (SerIn? E : D) : D ; rst_cnt =1 ; end

          E: begin ns = Co ? A : E ; inc_cnt = 1 ; SerOutValid = 1 ;end

          default : ns = A;
      endcase
  end

  always @(posedge clk , posedge rst)begin
      if(rst)
        ps <= A;
      else
        ps <= ns;
  end

  assign SerOut = (ps==E) ? SerIn : 1'bz;
endmodule
```

D. Counter

E. Onepulser

```
module counter10(input clk_EN , clk ,rst, inc_cnt , rst_cnt ,
output [3:0] Count_out , output Co );
  reg [3:0]Count;
  always @(posedge clk  , posedge rst) begin
    if(rst) Count <= 4'd0 ;
    else if(clk_EN && rst_cnt) Count <= 4'd5;
    else if(clk_EN && inc_cnt) Count <= Count + 1  ;
  end
  assign Count_out = Count;
  assign Co = &Count;
endmodule
```

Attachment_file:

F. SSD

```
module SSD(input [3:0] Count_out , output reg[6:0] SSD_output);
  always@(Count_out) begin
      case(Count_out)
        4'd5 : SSD_output = 7'b0010010;
        4'd6 : SSD_output = 7'b0000010;
        4'd7 : SSD_output = 7'b1111000;
        4'd8 : SSD_output = 7'b0000000;
        4'd9 : SSD_output = 7'b0010000;
        4'd10 : SSD_output = 7'b0001000;
        4'd11 : SSD_output = 7'b0000011;
        4'd12 : SSD_output = 7'b1000110;
        4'd13 : SSD_output = 7'b0100001;
        4'd14 : SSD_output = 7'b0000110;
        4'd15 : SSD_output = 7'b0001110;
        default : SSD_output = 7'b1111111;
      endcase
  end
endmodule
```

G. Serial transmitter

```
module OP_moore(input clkPB , clk , rst , SerIn ,
output SerOutValid,SerOut, output[6:0]Output_SSD, output [15:0] bcd);
  wire inc_cnt;
  wire [3:0]Count_out;
  wire clk_EN , Co ,  rst_cnt;
OnePulse   one_pulse(clk,rst , clkPB,  clk_EN);
Moore_machine   moore_machine(clk ,rst , SerIn , clk_EN , Co , inc_cnt , rst_cnt , SerOutValid ,SerOut);
counter10 counter(clk_EN , clk ,rst, inc_cnt , rst_cnt , Count_out , Co );
SSD ssd(Count_out , Output_SSD );
endmodule
```

H. Serial transmitter - Testbench

```
`timescale 1 ns/1ns
module moore_machine_tb;
  reg clkPB , clk , rst , SerIn;
  wire SerOutValid,SerOut;
  wire [6:0] SSD;

  OP_moore a(clkPB , clk , rst , SerIn , SerOutValid ,SerOut ,SSD);
  initial begin
    #2
    clkPB = 1'b0;
    rst = 1'b1;
    clk = 1'b0;
  end
  initial #15 rst = 1'b0;
  always #13  clk = ~clk;
  always #50 clkPB = ~clkPB;
  initial begin
    SerIn = 1'b0;
    #5

    #100 SerIn = 1'b1;
    #100 SerIn = 1'b0;
    #100 SerIn = 1'b1;
    #100 SerIn = 1'b1;
      repeat (15)
          #100 SerIn = $random ;

      repeat (15)
          #100 SerIn = $random ;
    #500
  $stop;
end
endmodule
```

Attachment_file:

Attachment_file: