ES204 Digital Systems
LAB Assignment - 6

Indian Institute of Technology, Gandhinagar
February 14, 2024

Submission deadline: February 14, 2024
Marks : 40

Submission instructions:

Only one student from the team will submit with the word doc name Rollno1_Rollno2.pdf. The PDF will contain the code, testbench and simulation results, XDC and FPGA snapshots.
After synthesis, report power, LUT utilization and timing reports. You can take snapshots or save them as text files and add.

Design a 4-bit universal shift register using behavioral style of coding (always block). Implement it on FPGA and show all the contents of the shift register on the LEDs. Use Reset to reset the shift register to all 0s. Parallel load can be done from switches. Assume that 0 is padded during shifting.

Universal shift register allows shift left, shift right, parallel load and no-shift options.

Lab 6 ES 204

Birudugadda Srivibhav (22110050)
Reddybathuni Venkat (22110220)

#Behavioural implementation code of Universal Shift Register

1. Code

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 14.02.2024 03:05:15
// Design Name:
// Module Name: universal_shift_register
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////

module clock_divider(
    input main_clk,
    output slow_clk);
    reg [31: 0] counter;
    always@ (posedge main_clk)
    begin
    counter <= counter + 1;
    end
    assign slow_clk= counter[27];
endmodule
module universal_shift_register(
    input [3:0]Q,
    input [1:0] mode,
```

```
    // mode = 2'b00 is no shift
    // mode = 2'b01 is parallel load
    //mode = 2'b10 is left shift
    // mode = 2'b11 is right shift
    input reset,
    input clk,
    output reg [3:0]OUT
    );
    wire slow_clk;
    clock_divider inst(.main_clk(clk), .slow_clk(slow_clk));
    always@(posedge slow_clk)
    begin
    if(reset == 0)
    OUT <= 4'b0000;
    else if(mode == 2'b01)
    OUT <= Q;
    else if(mode == 2'b11)
    begin
    OUT <= {1'b0,OUT[3:1]};
    end
    else if (mode ==2'b10)
    begin
        OUT <= {OUT[2:0],1'b0};
    end
    else if(2'b00)
    OUT <= OUT;

    //when en = 0 OUT will be 0000
    end
endmodule
```

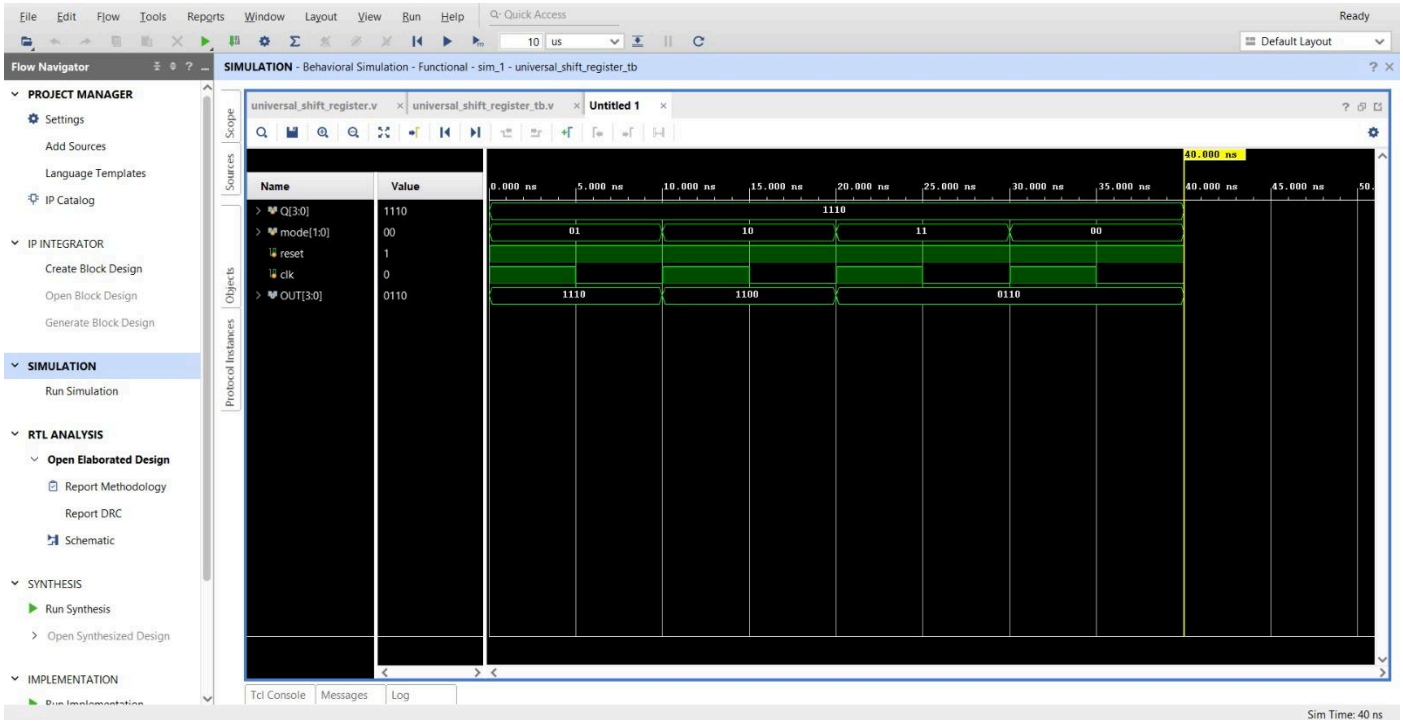
2. Test bench

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 14.02.2024 03:11:40
// Design Name:
// Module Name: universal_shift_register_tb
// Project Name:
// Target Devices:
```

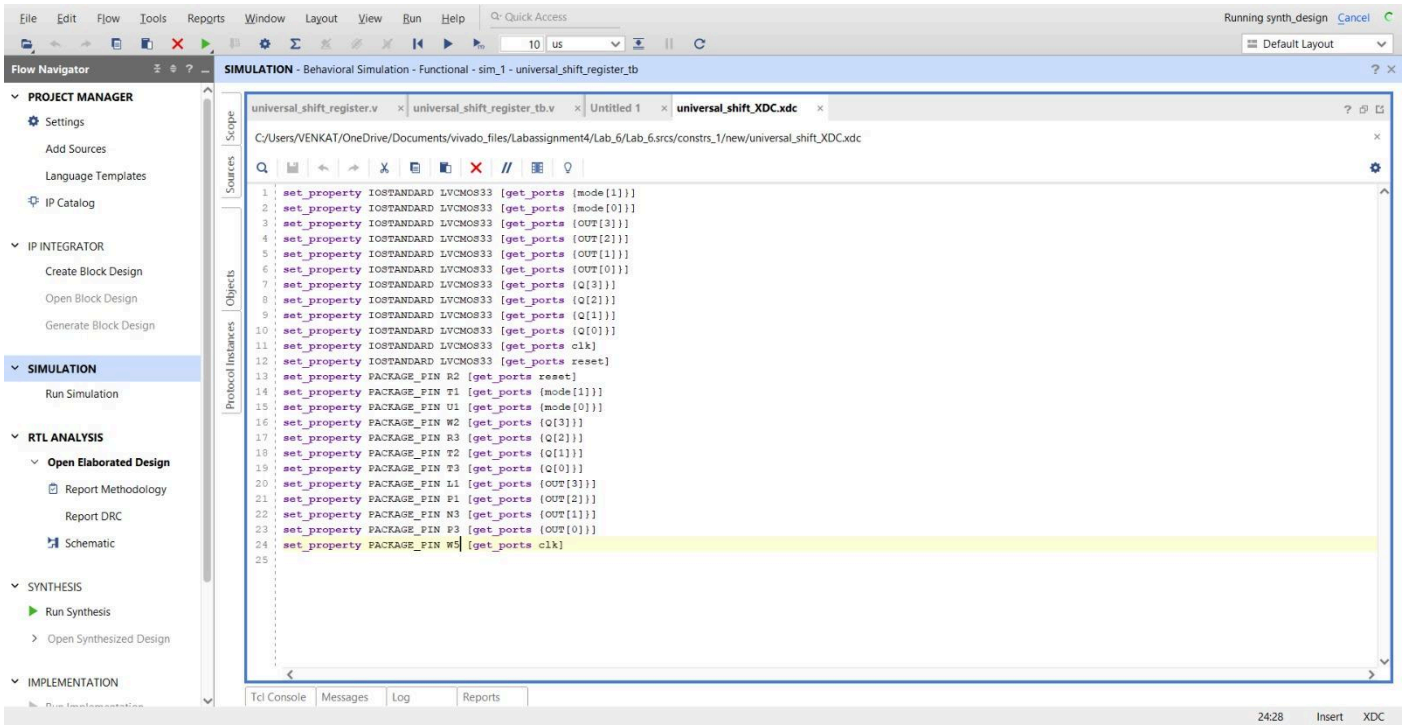
```
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module universal_shift_register_tb();
    reg [3:0]Q;
    reg [1:0] mode;
    reg reset,clk;
    wire [3:0]OUT;
    universal_shift_register uut(.Q(Q), .mode(mode), .reset(reset), .clk(clk), .OUT(OUT));
    initial
    begin
    clk = 1;
    forever #5 clk = ~clk;
    end
    initial
    begin
    Q = 4'b1110; mode = 2'b01; reset = 1;
    #10;
    Q = 4'b1110; mode = 2'b10; reset = 1;
    #10;
    Q = 4'b1110; mode = 2'b11; reset = 1;
    #10;
    Q = 4'b1110; mode = 2'b00; reset = 1;
    #10;
    $finish();
    end
endmodule
```

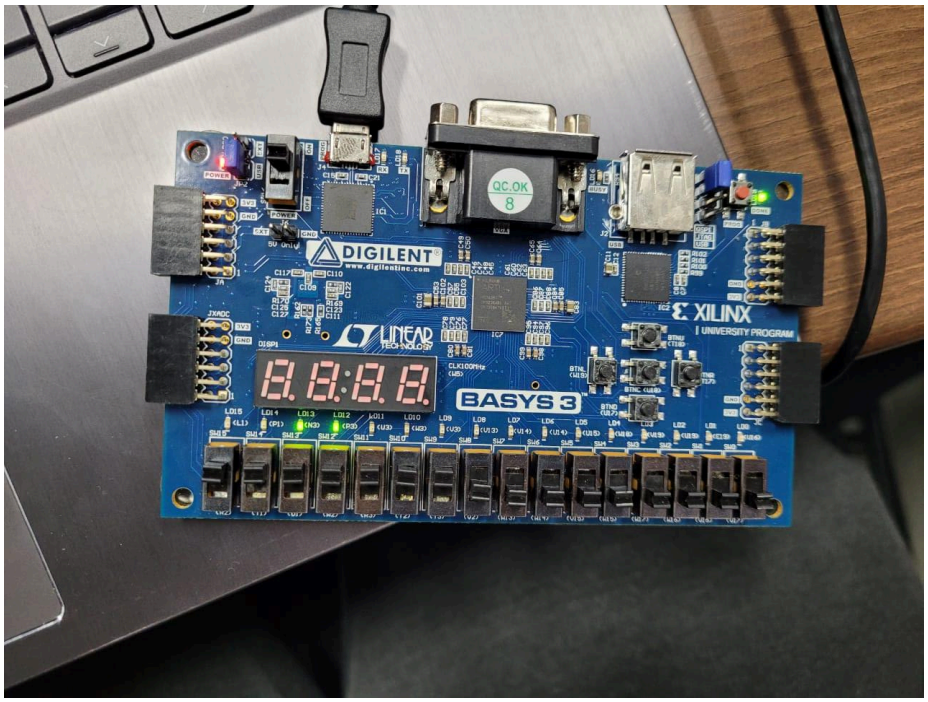
3. Simulation



4. XDC File



5. Circuit Image



6. Power Utilization