Course Code: CS223

Course Name: Digital Design

Section: 1

Lab 4

Mustafa Mert Gülhan

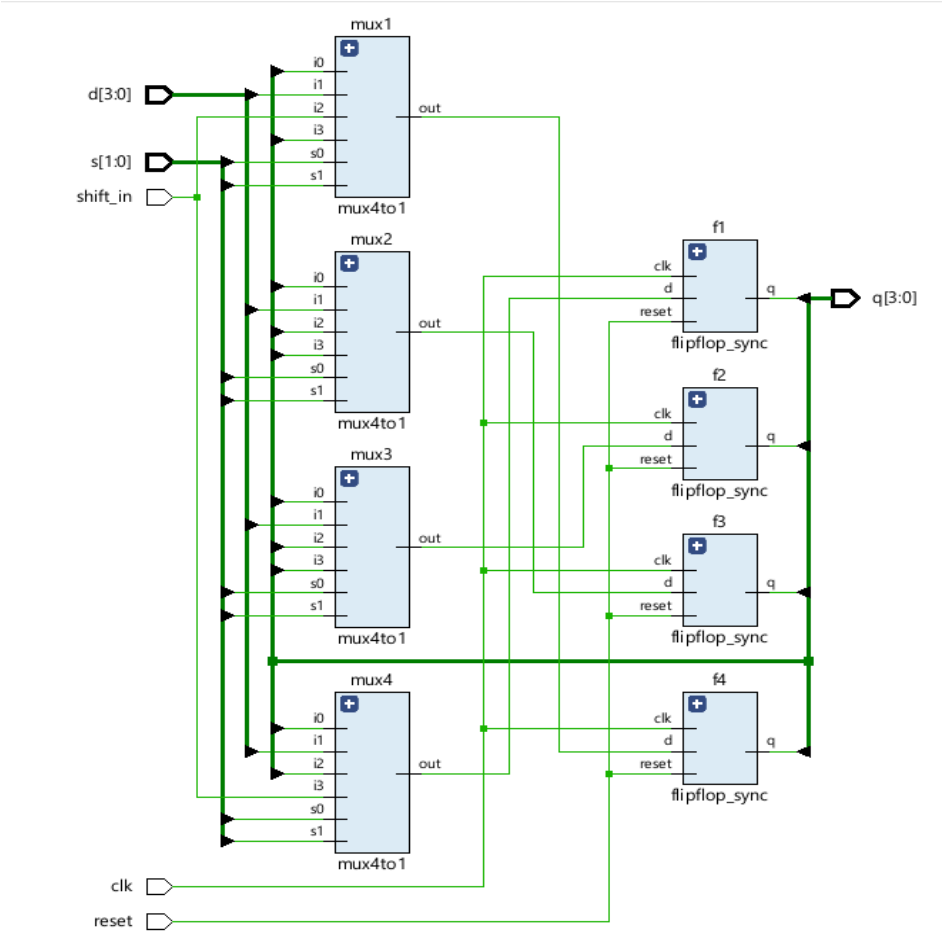
22201895

Date: 11.04.2024

SystemVerilog module for synchronously resettable D flip-flop

```
module flipflop_sync(input logic clk,reset,d,
output logic q);
always_ff @(posedge clk)
        if(reset)
            q <= 0;
        else
            q <= d;
endmodule
```

Circuit Schematic for Multifunction Register



Structural SystemVerilog module for the multifunction register

```
module multi_reg(input logic clk,reset,shift_in,
[3:0]d, [1:0]s,  output logic [3:0]q);

logic w1,w2,w3,w4;

mux4to1 mux1(q[0],d[0],shift_in,q[1],s[0],s[1],w1);
mux4to1 mux2(q[1],d[1],q[0],q[2],s[0],s[1],w2);
mux4to1 mux3(q[2],d[2],q[1],q[3],s[0],s[1],w3);
mux4to1 mux4(q[3],d[3],q[2],shift_in,s[0],s[1],w4);

flipflop_sync f1(clk,reset,w4,q[3]);
flipflop_sync f2(clk,reset,w3,q[2]);
flipflop_sync f3(clk,reset,w2,q[1]);
flipflop_sync f4(clk,reset,w1,q[0]);


endmodule
```

Multifunction register testbench

```
module multi_reg_tb();
    logic clk;
    logic reset;
    logic [3:0] d = 0;
    logic [1:0] s;
    logic shift_in;
    logic [3:0] q;
```

```
    multi_reg dut(clk,reset,shift_in,d,s,q);

    always begin
        clk = 0; #10;
        clk = 1; #10;
    end

    initial begin
        reset = 1; #20 reset = 0;
        s = 2'b00; d = 4'b0000; shift_in = 0; #20;
        s = 2'b01; d = 4'b1010; shift_in = 0; #20;
        s = 2'b10; d = 4'b1010; shift_in = 0; #20;
        s = 2'b11; d = 4'b1010; shift_in = 1; #20;
        s = 2'b00; d = 4'b1111; shift_in = 1; #20;
        s = 2'b01; d = 4'b1011; shift_in = 1; #20;
        s = 2'b11; d = 4'b1011; shift_in = 0; #20;
        s = 2'b10; d = 4'b1011; shift_in = 0; #20;
        s = 2'b00; d = 4'b1011; shift_in = 0; #20;
        s = 2'b01; d = 4'b1111; shift_in = 0; #20;
    end
endmodule
```

SystemVerilog module for BCD-to-7-segment code converter

```
module bcd_to7(input logic [3:0]w, output logic
a,b,c,d,e,f,g);
    always @(*)
    begin
        case(w)
        4'b0000:
```

```
begin
    a=0; b=0; c=0; d=0; e=0; f=0; g=1;
end
4'b0001:
begin
    a=1; b=0; c=0; d=1; e=1; f=1; g=1;
end
4'b0010:
begin
    a=0; b=0; c=1; d=0; e=0; f=1; g=0;
end
4'b0011:
begin
    a=0; b=0; c=0; d=0; e=1; f=1; g=0;
end
4'b0100:
begin
    a=1; b=0; c=0; d=1; e=1; f=0; g=0;
end
4'b0101:
begin
    a=0; b=1; c=0; d=0; e=1; f=0; g=0;
end
4'b0110:
begin
    a=0; b=1; c=0; d=0; e=0; f=0; g=0;
end
4'b0111:
begin
    a=0; b=0; c=0; d=1; e=1; f=1; g=1;
end
4'b1000:
```

```
        begin
            a=0; b=0; c=0; d=0; e=0; f=0; g=0;
        end
        4'b1001:
        begin
            a=0; b=0; c=0; d=0; e=1; f=0; g=0;
        end
        default:
        begin
            a=1; b=1; c=1; d=1; e=1; f=1; g=0;
        end
        endcase
    end
endmodule
```

BCD-to-7-segment code converter testbench

```
module bcd_to7_tb();
    logic [3:0]w;
    logic a,b,c,d,e,f,g;
    bcd_to7 dut (w,a,b,c,d,e,f,g);
    initial begin
        w= 4'b0000; #20;
        w= 4'b0001; #20;
        w= 4'b0010; #20;
        w= 4'b0011; #20;
        w= 4'b0100; #20;
        w= 4'b0101; #20;
        w= 4'b0110; #20;
        w= 4'b0111; #20;
        w= 4'b1000; #20;
        w= 4'b1001; #20;
```

```
        w= 4'b1101; #20;
    end
endmodule
```

Disadvantages of using BCD to represent numbers

1. Efficiency

It uses 4 bits to represent the only 1 decimal of the number. Thus, when dealing with large numbers it uses much more memory than the normal binary representation.

2. Arithmetic Operations

Dealing with arithmetic operations on BCD numbers is much more complex than dealing with normal binary numbers. Thus, it needs more electronics which makes it less efficient and slower.

Full four-digit seven-segment display on Basys3 Board

Firstly, cathodes for each digit are connected together for every digit on the

display. Thus to display different digits, it is needed to display every digit at different times and to see them that are changing at the same time with our eyes the frequency period of the 4 digit must be between 1KHz and 60Hz. Furthermore, the cathodes of 7-segments (a,b,c,d,e,f,g) must be on low logic level (0) to illuminate that LED. Thus, the truth table is reversed.