# CS223 - DIGITAL DESIGN

**SECTION 3 - LAB 4**

**ARDA TAVUSBAY**

**21902722**

**21.11.2021**

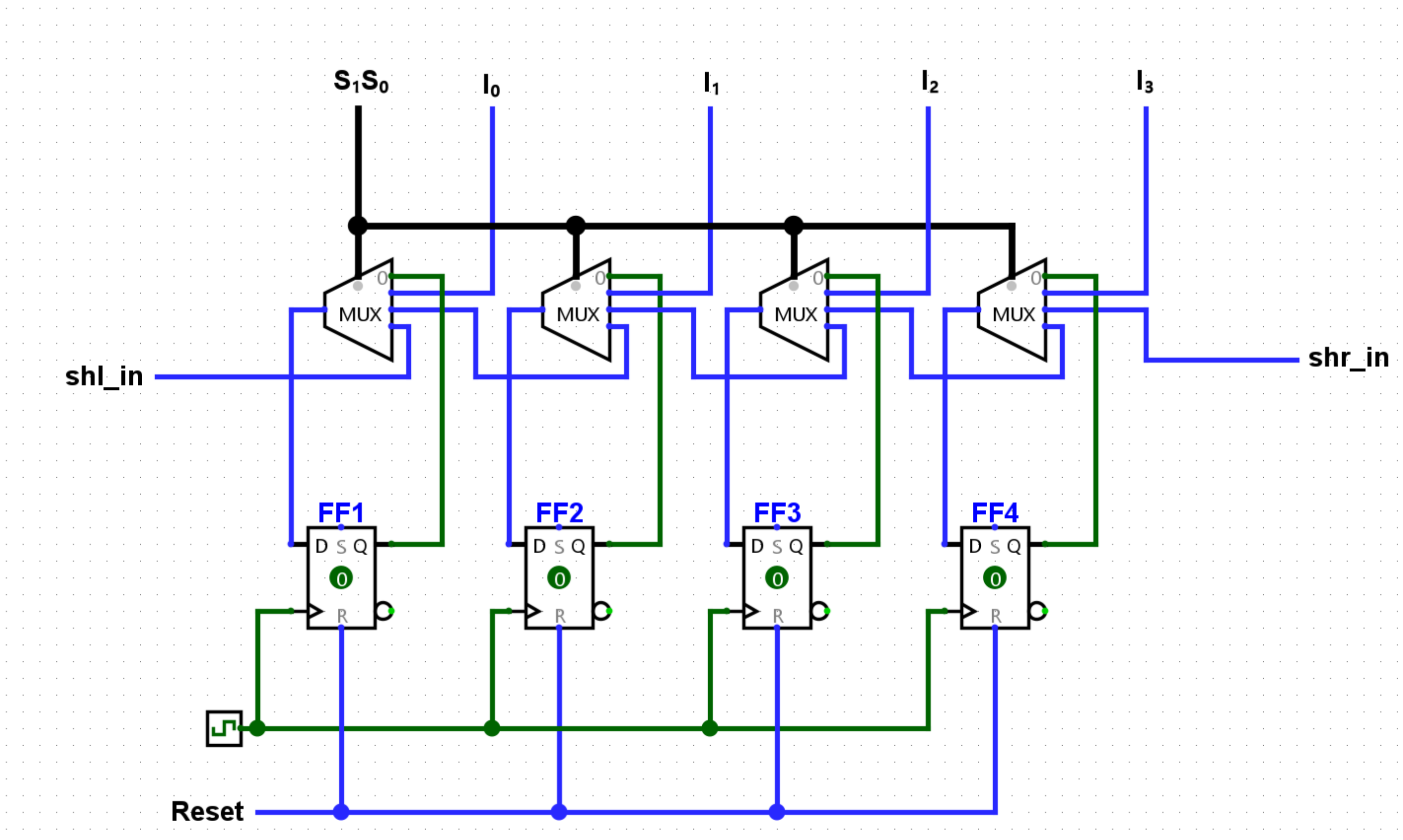
## (b) Write a SystemVerilog module for synchronously resettable D flip-flop

```
`timescale 1ns / 1ps


module ResDFlipFlop(input logic d, clk, reset, output reg q);


   always @(posedge clk)
   begin
   if(reset==1'b1)
      q <= 1'b0;
   else
      q <= d;
   end
endmodule
```

## (c) Draw a circuit schematic (block diagram) for the internal design of the multifunction register by using 4:1 multiplexers and synchronously resettable D flip-flops.



**(d) Write a Structural SystemVerilog module for the multifunction register you designed in part (c) and a testbench for it.**

```
`timescale 1ns / 1ps

module MultifuncReg(input i0, i1, i2, i3, s0, s1, shr_in, shl_in, reset, clk, output [3:0]out);

wire[3:0]w;

Mux4_1 m1( out[0], i0, out[1], shl_in, s0, s1, w[0]);
Mux4_1 m2( out[1], i1, out[2], out[0], s0, s1, w[1]);
Mux4_1 m3( out[2], i2, out[3], out[1], s0, s1, w[2]);
Mux4_1 m4( out[3], i3, shr_in, out[2], s0, s1, w[3]);

ResDFlipFlop d1(w[0], clk, reset, out[0]);
ResDFlipFlop d2(w[1], clk, reset, out[1]);
ResDFlipFlop d3(w[2], clk, reset, out[2]);
ResDFlipFlop d4(w[3], clk, reset, out[3]);

endmodule

module Mux4_1( input x0, x1, x2, x3, s0, s1, output reg y);
always@(x0, x1, x2, x3, s0, s1)
begin
if (s1 == 0 && s0 == 0 )
        y = x0;
if (s1 == 0 && s0 == 1 )
        y = x1;
if (s1 == 1 && s0 == 0 )
        y = x2;
if (s1 == 1 && s0 == 1 )
        y = x3;
end
endmodule
```

## Testbench for structural systemverilog module multifunctional register:

```
`timescale 1ns / 1ps

module TestbenchMultifuncReg();

reg i0, i1, i2, i3, s0, s1, shr_in, shl_in, reset, clk;
wire [3:0]out;
MultifuncReg dut(i0, i1, i2, i3, s0, s1, shr_in, shl_in, reset, clk, out);

always #50clk=~clk;
initial begin

// INITIALIZE VALUES
clk = 1'b0;
reset = 1'b0;
shl_in = 1'b0;
shr_in = 1'b0;
s0 = 1'b0;
s1 = 1'b0;
i0 = 0;
i1 = 0;
i2 = 0;
i3 = 1;

//CHECK PARALLEL LOAD
//SELECT LINE: 01 (s0 = 1, s1 = 0)
// ----------------------------------------------------------
#100;
s0 = 1;
s1 = 0;
```

```
#100;
i0 = 1;
i1 = 0;
i2 = 1;
i3 = 0;
// ------------------------------------------------------------

// CHECK IF IT STAYS THE SAME
// SELECT LINE: 00 (s0 = 0, s1 = 0)
// ------------------------------------------------------------
#100;
s0 = 0;
s1 = 0;
// ------------------------------------------------------------

// CHECK SHIFT RIGHT
// SELECT LINE: 11, [3:0]i = 0010 (i3 = 0, i2 = 0, i1= 1, i0 = 0)
// EXPECTED OUTPUT = [3:0]i = 0100 (i3 = 0, i2 = 1, i1= 0, i0 = 0)
// ------------------------------------------------------------
#100;
i0 = 0;
i1 = 1;
i2 = 0;
i3 = 0;

#100 // first load 0010
s0 = 1;
s1 = 0;

#100; // check shift right
s0 = 1;
s1 = 1;
```

```
// ------------------------------------------------------------

// CHECK SHIFT LEFT
// SELECT LINE: 10, [3:0]i = 0100 (i3 = 0, i2 = 1, i1= 0, i0 = 0)
// EXPECTED OUTPUT = [3:0]i = 0010 (i3 = 0, i2 = 0, i1= 1, i0 = 0)
// ------------------------------------------------------------
#100;
i0 = 0;
i1 = 0;
i2 = 1;
i3 = 0;

#100 // first load 0100
s0 = 1;
s1 = 0;

#100; // check shift left
s0 = 0;
s1 = 1;
// ------------------------------------------------------------

end
endmodule
```