CS223

Digital Design

Section: 1

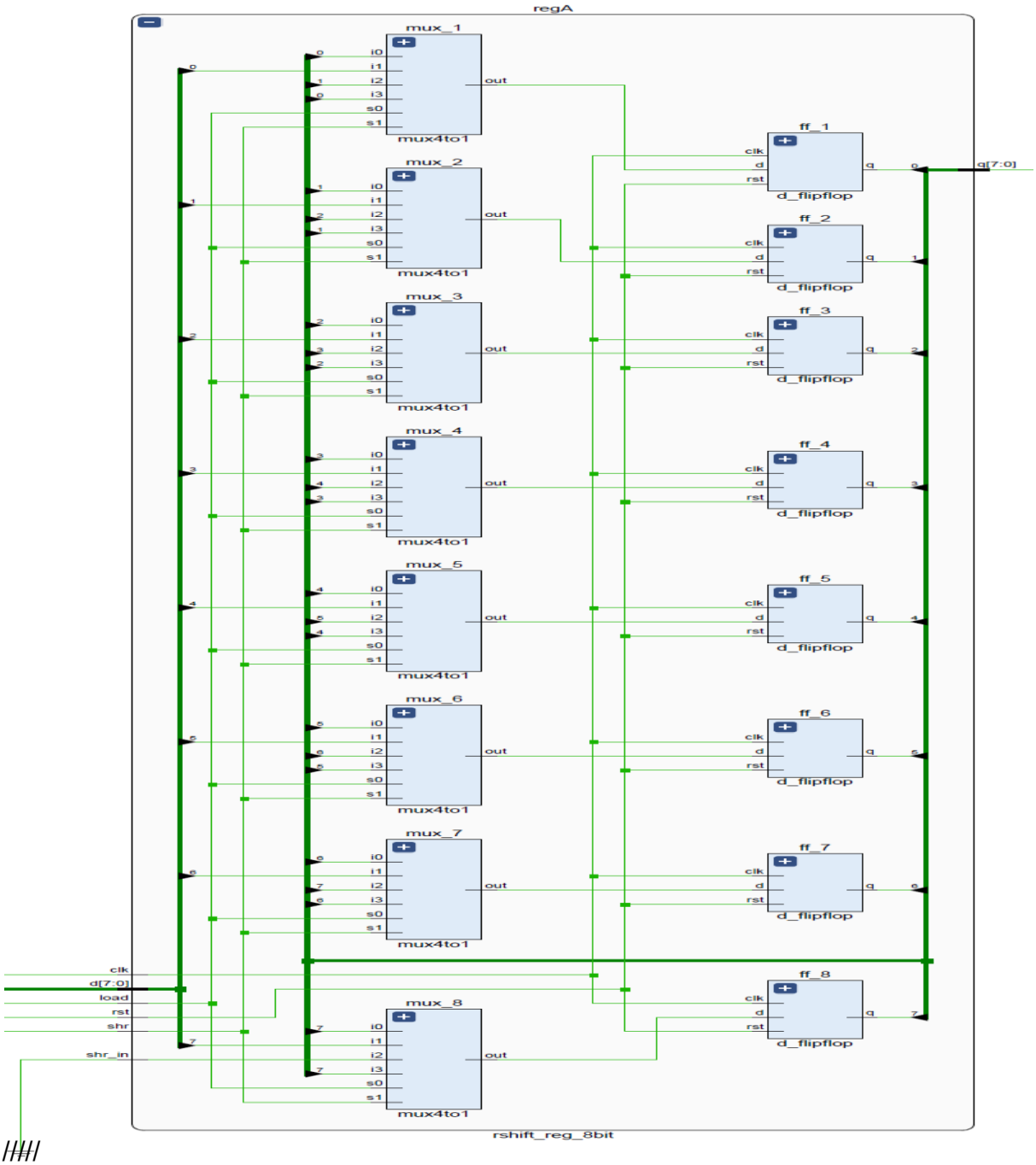
Lab-05

Name: Tolga Han Arslan

ID: 22003061

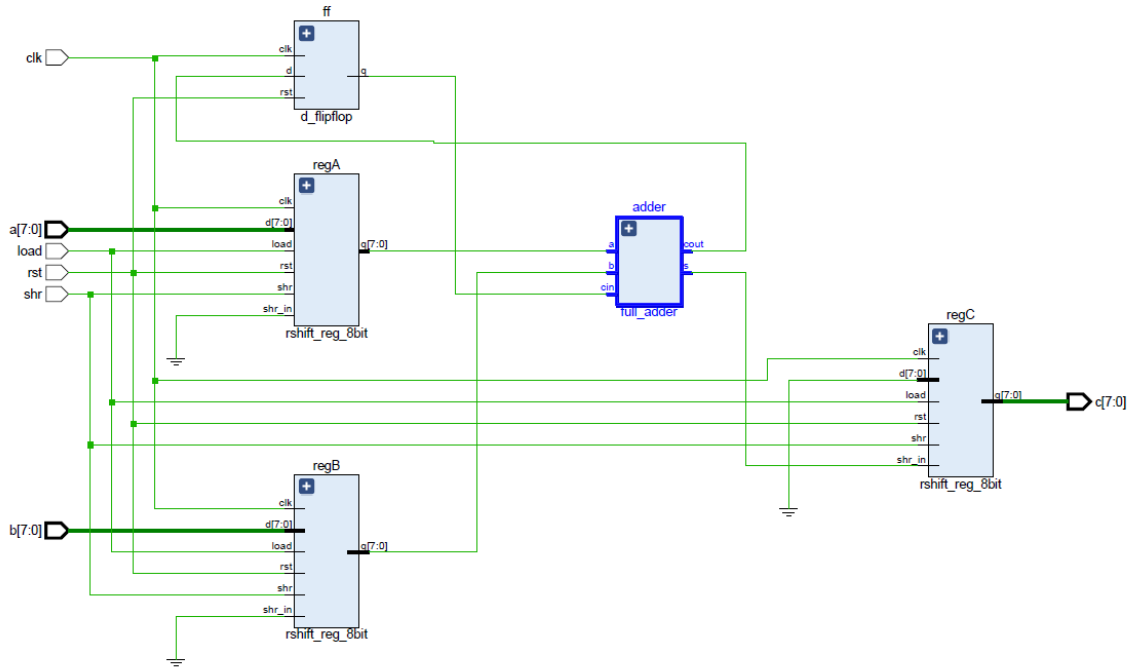
5.12.2022

(b) Circuit schematic for your shift register design using D flip-flops.



////

(c) Circuit schematic for your serial adder using the shift registers, full adder, and D flip-flop.



(d) SystemVerilog module for synchronously resettable D flip-flop.

```
module d_flipflop(
input logic d, clk, rst,
output logic q
   );
   always_ff @(posedge clk)
   begin
   if(rst) q<=0;
   else q<=d;
   end

endmodule
```

(e) Structural SystemVerilog module for your shift register using the D flip-flop module along with the testbench.

```
module rshift_reg_8bit(
```

```
input logic clk, rst, load, shr, shr_in, [7:0]d,
output logic [7:0] q
   );
   logic out[7:0];
   mux4to1 mux_8(q[7],d[7],shr_in,q[7],load,shr,out[7]);
   d_flipflop ff_8(out[7],clk,rst,q[7]);
   mux4to1 mux_7(q[6],d[6],q[7],q[6],load,shr,out[6]);
   d_flipflop ff_7(out[6],clk,rst,q[6]);
   mux4to1 mux_6(q[5],d[5],q[6],q[5],load,shr,out[5]);
   d_flipflop ff_6(out[5],clk,rst,q[5]);
   mux4to1 mux_5(q[4],d[4],q[5],q[4],load,shr,out[4]);
   d_flipflop ff_5(out[4],clk,rst,q[4]);
   mux4to1 mux_4(q[3],d[3],q[4],q[3],load,shr,out[3]);
   d_flipflop ff_4(out[3],clk,rst,q[3]);
   mux4to1 mux_3(q[2],d[2],q[3],q[2],load,shr,out[2]);
   d_flipflop ff_3(out[2],clk,rst,q[2]);
   mux4to1 mux_2(q[1],d[1],q[2],q[1],load,shr,out[1]);
   d_flipflop ff_2(out[1],clk,rst,q[1]);
   mux4to1 mux_1(q[0],d[0],q[1],q[0],load,shr,out[0]);
   d_flipflop ff_1(out[0],clk,rst,q[0]);

endmodule
module rshift_reg_8bit_tb();
logic clk,rst,load,shr,shr_in;
logic [7:0]d;
logic [7:0]q;
rshift_reg_8bit dut(clk,rst,load,shr,shr_in,d,q);
```

```
  always
    begin
    clk=0; #5;
    clk=1; #5;
    end
    initial begin
    d = 8'b10001000;
    shr_in = 1;
    rst = 1; #100;
    load = 0; shr = 0; #100;
    load = 0; shr = 1; #100;
    load = 1; shr = 0; #100;
    load = 1; shr = 1; #100;
    rst =0;#100;
    load = 0; shr = 0; #100;
    load = 0; shr = 1; #100;
    load = 1; shr = 0; #100;
    load = 1; shr = 1; #100;
    rst = 1; #100;
end
endmodule
```

(f) Structural SystemVerilog module for your serial adder using the shift register, full adder, and D flip-flop modules along with the testbench.

```
module serial_adder(input logic clk,rst,load,shr, input logic [7:0]a, input logic [7:0]b,
output logic [7:0]c);
logic [7:0] x;
logic [7:0] y;
logic sum,cin,cout;
```

```
logic [3:0]count;
logic run;
rshift_reg_8bit regA(clk,rst,load,shr,0,a,x);
rshift_reg_8bit regB(clk,rst,load,shr,0,b,y);
full_adder adder(x[0],y[0],cin,sum,cout);
d_flipflop ff(cout,clk,rst,cin);
rshift_reg_8bit regC(clk,rst,load,run,sum,8'b00000000,c);
always@(posedge clk)
if(rst|load)
count <= 8;
else if(run) count <= count -1;
assign run =|count;

endmodule
module serial_adder_tb();
logic clk,rst,load,shr;
logic [7:0]a; logic [7:0]b; logic [7:0]c;
   serial_adder dut(clk,rst,load,shr,a,b,c);
   always
      begin
      clk = 0; #5;
      clk = 1; #5;
      end
      initial begin
      a=8'b01000000;b=8'b00000010;
            rst = 1; #100;
            load = 0; shr = 0; #100;
```

```
        load = 1; shr=0; #100;
        load = 0; shr=1; #100;
        load = 1; shr=1; #100;
        rst = 0; #100;
        load = 0; shr = 0; #100;
        load = 1; shr=0; #100;
        load = 0; shr=1; #100;
        load = 1; shr=1; #100;
        rst = 1; #100;
    end
endmodule
```