CS 223

Section – 6

Lab – 05

Mert Fidan

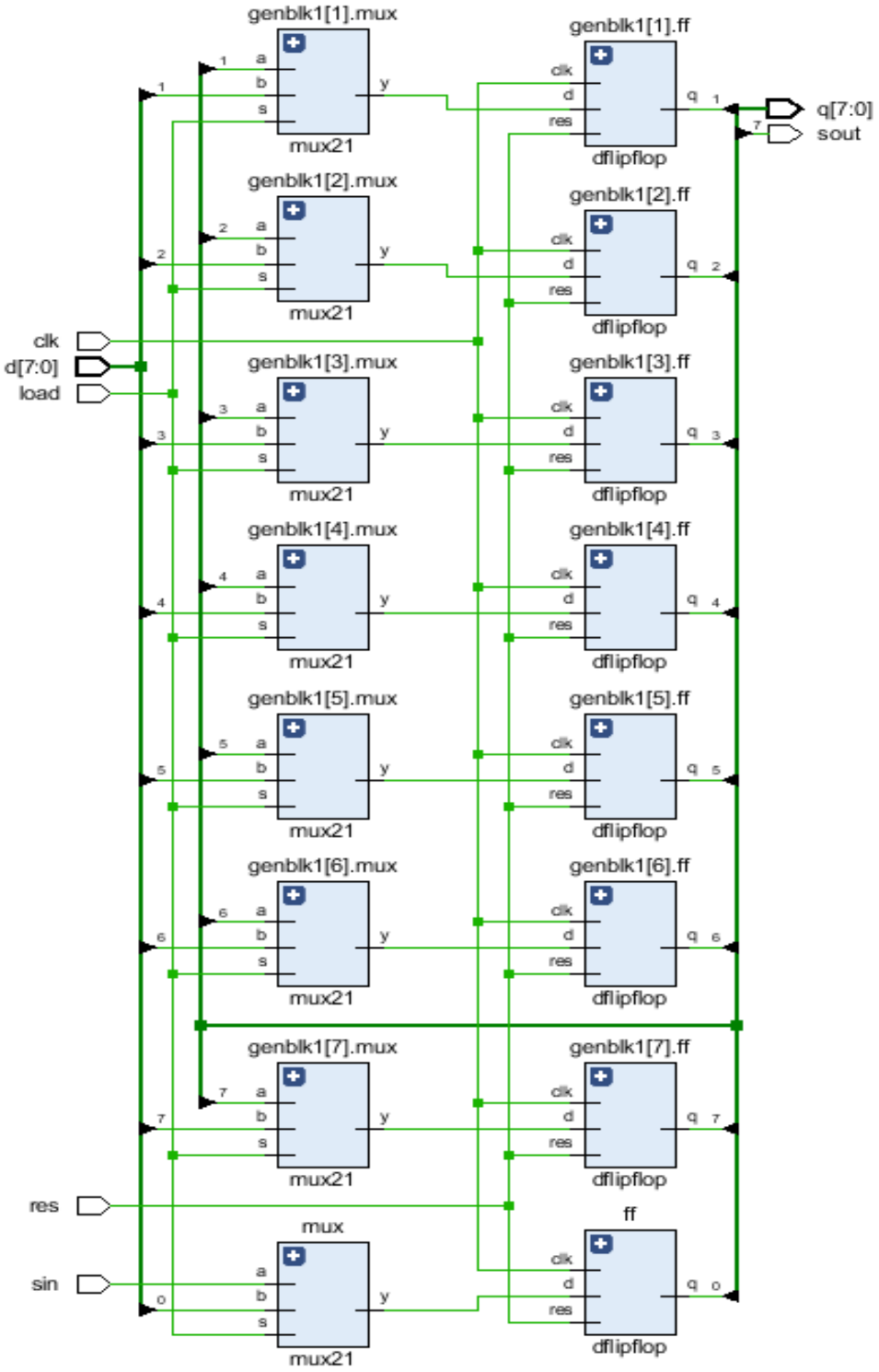
22101734

03.12.2022

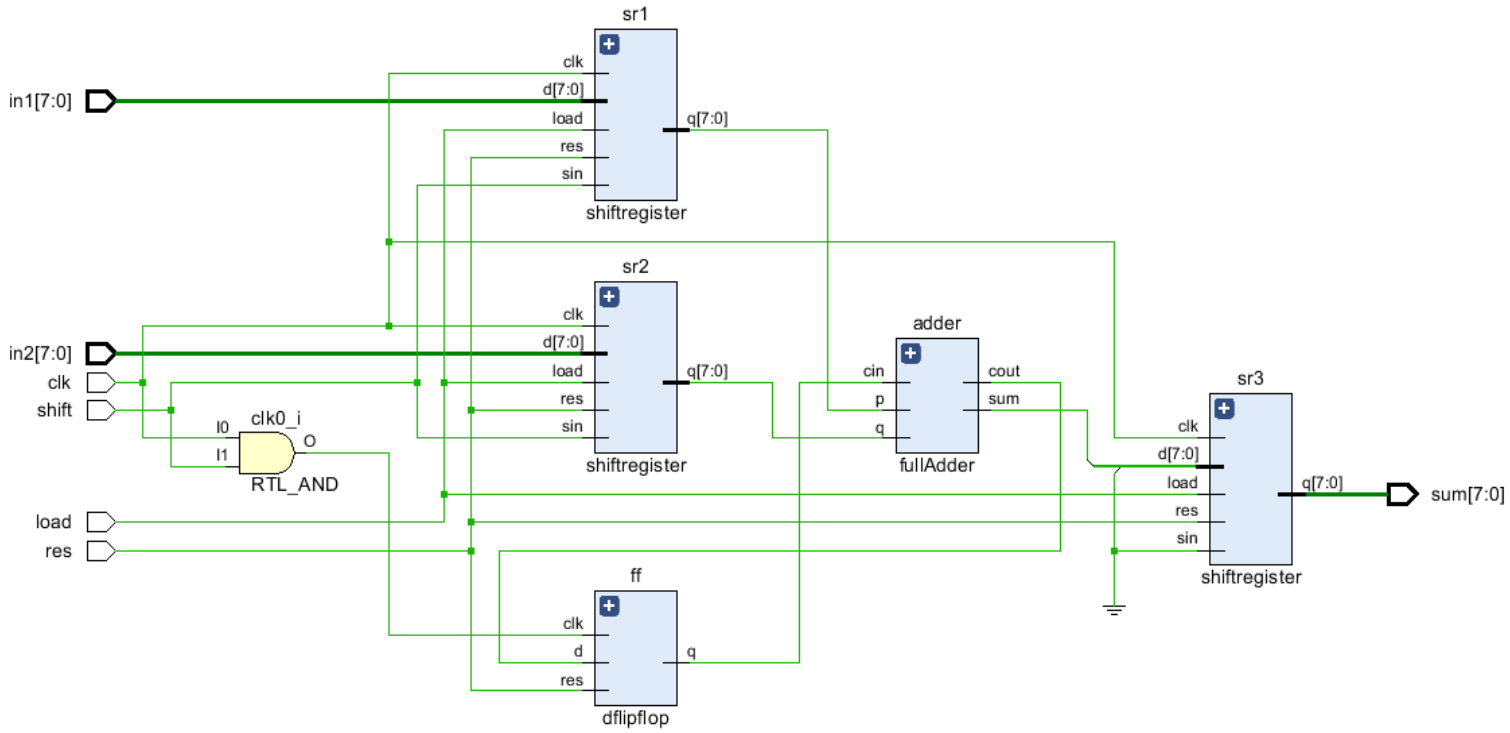
- **2-to-1 Multiplexer Module:**

```
module mux21(input logic a, b, s, output logic y);
    assign y = (~s & a) + (s & b);
endmodule
```

- **Shift Register Circuit Schematic:**



- **Serial Adder Circuit Schematic:**



- **Synchronously Resettable D Flip-Flop Module:**

```
module dflipflop(input logic clk, res, d, output logic q);
    always@(posedge clk)
        begin
            if(res) q <= 1'b0;
            else q <= d;
        end
endmodule
```

- **Shift Register Module:**

```
module shiftregister(input logic clk, res, load, sin, [7:0]d, output logic sout, [7:0]q);
    parameter n = 8;
    genvar i;
    logic muxOut;
    mux21 mux(sin, d[0], load, muxOut);
    dflipflop ff(clk, res, muxOut, q[0]);
    generate
      for(i = 1; i < n; i = i + 1)
        begin
         logic mOut;
         mux21 mux(q[i], d[i], load, mOut);
         dflipflop ff(.clk(clk), .res(res), .d(mOut), .q(q[i]));
        end
    endgenerate
    assign sout = q[n - 1];
endmodule
```

- **Serial Adder Module:**

```
module serialAdder(input logic clk, res, load, shift, [7:0]in1, in2, output logic [7:0] sum);
    logic sout1, sout2, sout3, cin, cout;
    logic  [7:0]res1, res2, res3;
    shiftregister sr1(clk, res, load, shift, in1, sout1, res1);
    shiftregister sr2(clk, res, load, shift, in2, sout2, res2);
    logic ffclk = clk & shift;
    dflipflop ff(ffclk, res, cout, cin);
    fullAdder adder(res1, res2, cin, res3, cout);
    shiftregister sr3(clk, res, load, 0, res3, sout3, sum);
endmodule
```

- **Shift Register Testbench:**

```
module shiftregister_tb();
    logic clk, res, load, sin, sout;
    logic [7:0]d, q;
    shiftregister sr(clk, res, load, sin, d, sout, q);
    always
        begin
            clk = 0; #10; clk = 1; #10;
        end
    integer i;
    initial
        begin
            sin = 0; res =1; load = 1;
            #2 sin = 0 ; res = 0;
            #2 res=1;
            for(i =0; i<=10; i=i+1)
                begin
                    #20 sin = ~sin;
                end
            #20 sin =1;
            #20 sin = 1;
            #20 sin =0;
            #20 sin =1;
            #20 sin = 1;
            #20 sin =0;
            #20 sin =1;
            #20 sin = 1;
            #20 sin =0;
         end
endmodule
```

- **Serial Adder Testbench:**

```
module serialAdder_tb();
    logic clk, res, load, shift;
    logic [7:0]in1, in2, sum;
    serialAdder sa(clk, res, load, shift, in1, in2, sum);
    always
        begin
            clk = 0; #5; clk = 1; #5;
        end
    initial
        begin
            res = 0; load = 1;
            in1[0] = 1; in2[0]=0; shift = 1;#10;
            in1[1] = 0; in2[1]=1; shift = 0;#10;
            in1[2] = 1; in2[2]=0; shift = 0;#10;
            in1[3] = 0; in2[3]=1; shift = 0;#10;
            in1[4] = 1; in2[4]=0; shift = 0;#10;
            in1[5] = 0; in2[5]=1; shift = 0;#10;
            in1[6] = 1; in2[6]=0; shift = 0;#10;
            in1[7] = 0; in2[7]=1; shift = 0;#10;
            res = 1; #10;
            res = 0;
            in1[0] = 0; in2[0]=0; shift = 1;#10;
            in1[1] = 0; in2[1]=0; shift = 0;#10;
            in1[2] = 0; in2[2]=0; shift = 0;#10;
            in1[3] = 0; in2[3]=0; shift = 0;#10;
            in1[4] = 1; in2[4]=0; shift = 0;#10;
            in1[5] = 0; in2[5]=1; shift = 0;#10;
            in1[6] = 1; in2[6]=0; shift = 0;#10;
            in1[7] = 0; in2[7]=1; shift = 0;#10;
            res = 1; #100;
          end
endmodule
```