VLSI LABORATORY [EC39004]
LAB REPORT - 2

Kundrapu N R Sai Akash [20EC10043]
Nagisetti Rithihas [20EC10049]

OBJECTIVE:

1) Implement barrel shifter that performs logical shift right or rotate right operations based on the control input. (3-bit select line,structural)

   if i/p is I3 I2 I1 I0

Cntrl	Operation	Output
0	logical shift right	0 I3 I2 I1
1	rotate right	I0 I3 I2 I1

2) Implement barrel shifter that performs logical shift left or logical shift right based on the control input. (3-bit select line,structural)

   if i/p is I3 I2 I1 I0

Cntrl	Operation	Output
0	logical shift left	I2 I1 I0 0
1	logical shift right	0 I3 I2 I1

CODES:

1) *Structural Code for Part 1.*

```
module mux(y,a,b,s);

    input a,b,s;
    output y;
    wire[2:0] t;

    not G1(t[0],s);
    and G2(t[1],t[0],a);
    and G3(t[2],s,b);
    or G4(y,t[1],t[2]);

endmodule

module submux(out,in0,in1,sel);

    input[7:0] in0,in1;
    input sel;
    output[7:0] out;

    mux M0(out[0],in0[0],in1[0],sel);
    mux M1(out[1],in0[1],in1[1],sel);
    mux M2(out[2],in0[2],in1[2],sel);
    mux M3(out[3],in0[3],in1[3],sel);
    mux M4(out[4],in0[4],in1[4],sel);
    mux M5(out[5],in0[5],in1[5],sel);
    mux M6(out[6],in0[6],in1[6],sel);
    mux M7(out[7],in0[7],in1[7],sel);

endmodule

module A2P1(out,in,sel,mode);

    input[7:0] in;
    input[2:0] sel;
    input mode;
    output[7:0] out;
```

```
    wire[7:0] t1,t2;
    wire[6:0] t;

    and M0(t[0],in[0],mode);
    submux S1(t1,in,{t[0],in[7:1]},sel[0]);
    and M1(t[1],t1[0],mode);
    and M2(t[2],t1[1],mode);
    submux S2(t2,t1,{t[2],t[1],t1[7:2]},sel[1]);
    and M3(t[3],t2[0],mode);
    and M4(t[4],t2[1],mode);
    and M5(t[5],t2[2],mode);
    and M6(t[6],t2[3],mode);
    submux S3(out,t2,{t[6],t[5],t[4],t[3],t2[7:4]},sel[2]);

endmodule
```

2) *Structural Code for Part 2.*

```
module mux(y,a,b,s);

    input a,b,s;
    output y;
    wire[2:0] t;

    not G1(t[0],s);
    and G2(t[1],t[0],a);
    and G3(t[2],s,b);
    or G4(y,t[1],t[2]);

endmodule

module submux(out,in0,in1,sel);

    input[7:0] in0,in1;
    input sel;
    output[7:0] out;

    mux M0(out[0],in0[0],in1[0],sel);
    mux M1(out[1],in0[1],in1[1],sel);
```

```
    mux M2(out[2],in0[2],in1[2],sel);
    mux M3(out[3],in0[3],in1[3],sel);
    mux M4(out[4],in0[4],in1[4],sel);
    mux M5(out[5],in0[5],in1[5],sel);
    mux M6(out[6],in0[6],in1[6],sel);
    mux M7(out[7],in0[7],in1[7],sel);

endmodule

module LS(out,in,sel);

    input[7:0] in;
    input[2:0] sel;
    output[7:0] out;
    wire[7:0] t1,t2;

    submux S1(t1,in,{in[6:0],1'b0},sel[0]);
    submux S2(t2,t1,{t1[5:0],2'b00},sel[1]);
    submux S3(out,t2,{t2[3:0],4'b0000},sel[2]);

endmodule

module RS(out,in,sel);

    input[7:0] in;
    input[2:0] sel;
    output[7:0] out;
    wire[7:0] t1,t2;

    submux S4(t1,in,{1'b0,in[7:1]},sel[0]);
    submux S5(t2,t1,{2'b00,t1[7:2]},sel[1]);
    submux S6(out,t2,{4'b0000,t2[7:4]},sel[2]);

endmodule

module A2P2(out,in,sel,mode);

    input[7:0] in;
    input[2:0] sel;
    input mode;
```

```
    output[7:0] out;
    wire[7:0] t1,t2;

    LS P1(t1,in,sel);
    RS P2(t2,in,sel);
    submux P3(out,t1,t2,mode);

endmodule
```

3) Testbench.

```
module A2P2_tb;

    reg[7:0] in;
    reg[2:0] sel;
    reg mode;
    wire[7:0] out;
    A2P1 dut(out,in,sel,mode);

        initial
    begin
        $dumpfile("A2P1.vcd");
        $dumpvars;
    end

    initial
    begin
        $monitor("Time=%5d, in=%b, sel=%b, out=%b, mode=%b",
$time,in,sel,out,mode);
        #2 sel=3'b001;in=8'b11000011;mode=1'b0;
        #2 sel=3'b101;in=8'b11000011;mode=1'b0;
        #2 sel=3'b001;in=8'b11000011;mode=1'b1;
        #2 sel=3'b101;in=8'b11000011;mode=1'b1;
        #2 $finish;
    end

endmodule
```

RUNTIME OUTPUTS:

For part1,

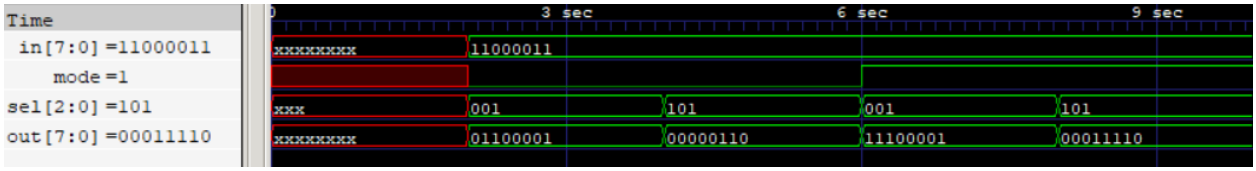
```
PS C:\IIT KGP\SEM 6\Verilog\Assignment_2> iverilog -o A2P1 A2P1.v A2P2_tb.v
PS C:\IIT KGP\SEM 6\Verilog\Assignment_2> vvp A2P1
VCD info: dumpfile A2P2.vcd opened for output.
Time=    0, in=xxxxxxxx, sel=xxx, out=xxxxxxxx, mode=x
Time=    2, in=11000011, sel=001, out=01100001, mode=0
Time=    4, in=11000011, sel=101, out=00000110, mode=0
Time=    6, in=11000011, sel=001, out=11100001, mode=1
Time=    8, in=11000011, sel=101, out=00011110, mode=1
```

For part2,

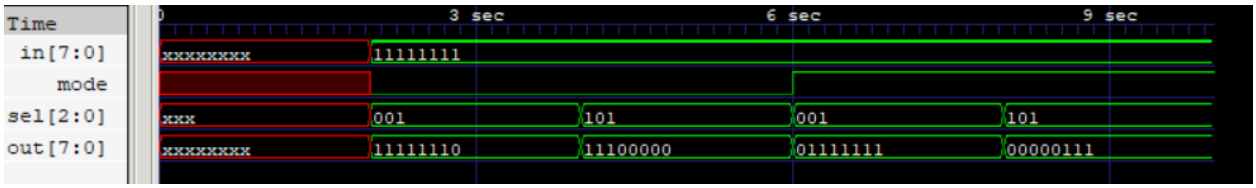
```
PS C:\IIT KGP\SEM 6\Verilog\Assignment_2> iverilog -o A2P2 A2P2.v A2P2_tb.v
PS C:\IIT KGP\SEM 6\Verilog\Assignment_2> vvp A2P2
VCD info: dumpfile A2P2.vcd opened for output.
Time=    0, in=xxxxxxxx, sel=xxx, out=xxxxxxxx, mode=x
Time=    2, in=11111111, sel=001, out=11111110, mode=0
Time=    4, in=11111111, sel=101, out=11100000, mode=0
Time=    6, in=11111111, sel=001, out=01111111, mode=1
Time=    8, in=11111111, sel=101, out=00000111, mode=1
```

GTK Waveform:

For part1,



For part2,



Discussion:

- In the second class of this lab, we learnt about the datatypes present in verilog, along with some operators, primitive gates.
- In this experiment, we need to write a structural code for barrel shifter which is a digital circuit that can shift a binary word by a specified number of bits in a single clock cycle.
- It is commonly used in computer processors and other digital logic circuits to perform bitwise operations such as shift, rotate, and masking. The name "barrel shifter" refers to the way the bits of the input word are "rolled" through the shifting stages, similar to the way a barrel is rolled.
- In this experiment, we implemented the barrel shifter with 3 select line input which specifies how many bits to shift or rotate.
- In the first part, we used the mode pin to either right shift or right rotate, this can be achieved using and gates to alternate between modes, and give this input to the mux.
- To make to code simpler, we defined submux, which is a set of 8muxes which has same select input. With this, we can avoid the brute force method of writing all the muxes every time.
- In the second part, we used the mode pin to either right shift or left shift. For this, we created a module for right shift and left shift separately then used another submux module to give the correct output based on the mode which is given as select input to this submux.