VLSI LABORATORY [EC39004]
LAB REPORT - 3

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

OBJECTIVE:

1. Design 1x 8 demultiplexer using data flow modelling without using any gates or logical operators.

2. Design a 8-bit ALU using data flow.(8 bit inputs a,b;3 bit select line)

CODES:

1) *Data Flow Model for part1*

```
module generate_demux(out, in, sel);

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

    assign out[0] = (sel == 0) ? in : 0;
    assign out[1] = (sel == 1) ? in : 0;
    assign out[2] = (sel == 2) ? in : 0;
    assign out[3] = (sel == 3) ? in : 0;
    assign out[4] = (sel == 4) ? in : 0;
    assign out[5] = (sel == 5) ? in : 0;
    assign out[6] = (sel == 6) ? in : 0;
    assign out[7] = (sel == 7) ? in : 0;

endmodule
```

2) *Testbench for part1*

```
module A3P1_tb;

    reg in;
    reg[2:0] sel;
    wire[7:0] out;
    generate_demux D1 (out, in, sel);

    initial
    begin
        $dumpfile("A3P1");
        $dumpvars;
    end

    initial
    begin
```

```
        $monitor("Time = %3d, input = %b, select = %b ; output =
%b",$time,in,sel,out);
        #2 in = 1'b1 ; sel = 3'b000;
        #2 in = 1'b1 ; sel = 3'b001;
        #2 in = 1'b1 ; sel = 3'b010;
        #2 in = 1'b1 ; sel = 3'b011;
        #2 in = 1'b1 ; sel = 3'b100;
        #2 in = 1'b1 ; sel = 3'b101;
        #2 in = 1'b1 ; sel = 3'b110;
        #2 in = 1'b1 ; sel = 3'b111;
        #2 $finish;
    end

endmodule
```

3) *Data Flow Model for part2*

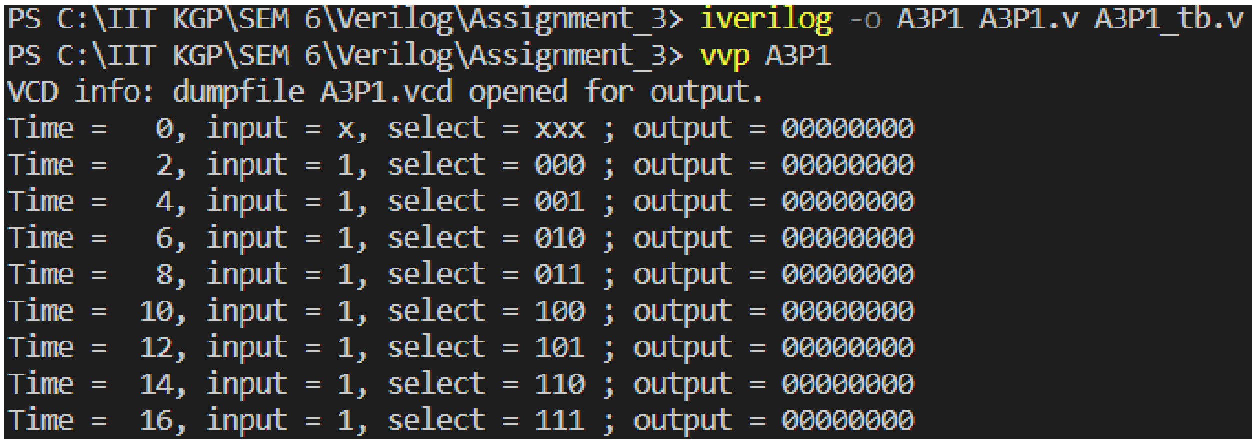
```
module ALU (c,out,a,b,sel);

    input[7:0] a,b;
    input[2:0] sel;
    output[7:0] out;
    output c;

    assign {c,out} = sel[2] ?
                ( sel[1] ? (sel[0] ? {1'b0,a[0],a[7:1]} :
{1'b0,a[6:0],a[7]} ) : (sel[0] ? {1'b0,a^b} : {1'b0,~a} )  ) :
                ( sel[1] ? (sel[0] ? {1'b0,a>>1} : {1'b0,a<<1} )   :
(sel[0] ? a-b : a+b ) ) ;


endmodule
```

4) *Testbench for part2*

```
module A3P2_tb;

    reg[7:0] a,b;
    reg[2:0] sel;
    wire[7:0] out;
    wire c;
    ALU A1 (c, out, a, b, sel);

    initial
    begin
        $dumpfile("A3P2");
        $dumpvars;
    end

    initial
    begin
        $monitor("Time = %3d, a = %b, b = %b ; sel = %d; out = %b; c =
%b",$time,a,b,sel,out,c);
        #2 a = 8'b11001010 ; b =8'b10010110 ; sel = 3'b000 ;
        #2 a = 8'b11001010 ; b =8'b10010110 ; sel = 3'b001 ;
        #2 a = 8'b11001010 ; b =8'b10010110 ; sel = 3'b010 ;
        #2 a = 8'b11001010 ; b =8'b10010110 ; sel = 3'b011 ;
        #2 a = 8'b11001010 ; b =8'b10010110 ; sel = 3'b100 ;
        #2 a = 8'b11001010 ; b =8'b10010110 ; sel = 3'b101 ;
        #2 a = 8'b11001010 ; b =8'b10010110 ; sel = 3'b110 ;
        #2 a = 8'b11001010 ; b =8'b10010110 ; sel = 3'b111 ;
        #2 $finish;
    end

endmodule
```

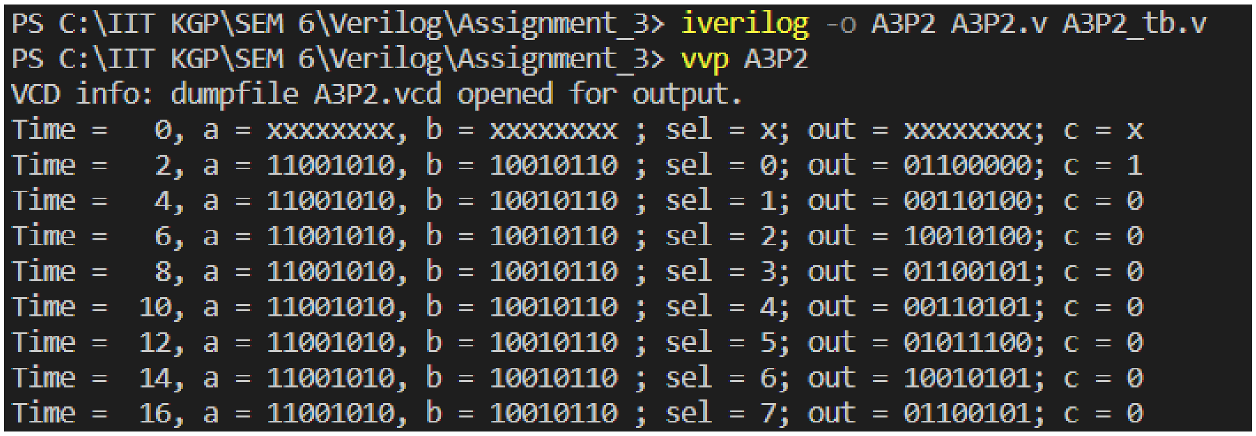
RUNTIME OUTPUTS:

For part1,



```
PS C:\IIT KGP\SEM 6\Verilog\Assignment_3> iverilog -o A3P1 A3P1.v A3P1_tb.v
PS C:\IIT KGP\SEM 6\Verilog\Assignment_3> vvp A3P1
VCD info: dumpfile A3P1.vcd opened for output.
Time =   0, input = x, select = xxx ; output = 00000000
Time =   2, input = 1, select = 000 ; output = 00000000
Time =   4, input = 1, select = 001 ; output = 00000000
Time =   6, input = 1, select = 010 ; output = 00000000
Time =   8, input = 1, select = 011 ; output = 00000000
Time =  10, input = 1, select = 100 ; output = 00000000
Time =  12, input = 1, select = 101 ; output = 00000000
Time =  14, input = 1, select = 110 ; output = 00000000
Time =  16, input = 1, select = 111 ; output = 00000000
```

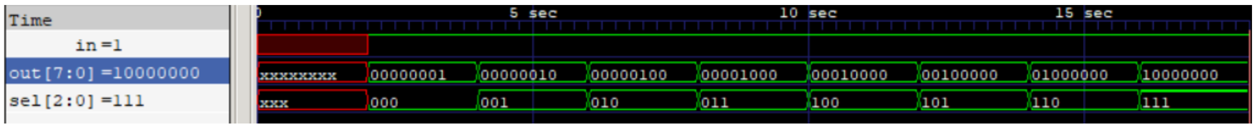
For part2,



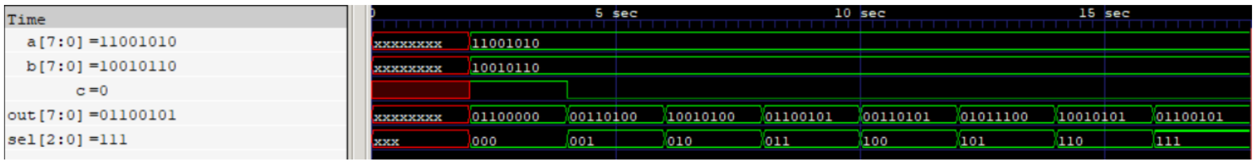
```
PS C:\IIT KGP\SEM 6\Verilog\Assignment_3> iverilog -o A3P2 A3P2.v A3P2_tb.v
PS C:\IIT KGP\SEM 6\Verilog\Assignment_3> vvp A3P2
VCD info: dumpfile A3P2.vcd opened for output.
Time =   0, a = xxxxxxxx, b = xxxxxxxx ; sel = x; out = xxxxxxxx; c = x
Time =   2, a = 11001010, b = 10010110 ; sel = 0; out = 01100000; c = 1
Time =   4, a = 11001010, b = 10010110 ; sel = 1; out = 00110100; c = 0
Time =   6, a = 11001010, b = 10010110 ; sel = 2; out = 10010100; c = 0
Time =   8, a = 11001010, b = 10010110 ; sel = 3; out = 01100101; c = 0
Time =  10, a = 11001010, b = 10010110 ; sel = 4; out = 00110101; c = 0
Time =  12, a = 11001010, b = 10010110 ; sel = 5; out = 01011100; c = 0
Time =  14, a = 11001010, b = 10010110 ; sel = 6; out = 10010101; c = 0
Time =  16, a = 11001010, b = 10010110 ; sel = 7; out = 01100101; c = 0
```

GTK Waveform:

For part1,



For part2,



Discussion:

- In the third class of this lab, we learnt about different types of modeling techniques present in verilog.
- In this experiment, we learnt about data flow modelling in depth, which is modelling the vlsi module behaviour using continuous assignment.
- We can do that using assign keyword, which marks a continuous assignment between a net type of LHS and an expression which may consist of both net and register type on RHS
- Data flow explains how a design processes the data rather than instantiation of individual gates.
- The difference between data flow and behavioral modelling is that, behavioral modelling we observe that it closely resembles a programming language whereas the data flow modelling being closer to a graphical(flow-chart) notation.
- In the first question, we are asked to generate a 1 to 8 demux without using gates and even logical operators, for this, we use the ternary operator to assign everything.
- Ternary operator is a short hand representation of if-else condition. We use this to design the demux in first part.
- In the second part we need to generate an 8-bit ALU for this, we use the ternary operator but this time, we use multiple ternary which is similar to nested if else.
- We used concatenation(for rotating) whenever needed to get the correct output, and we used bitwise operators whenever needed.