CSE 331/503 COMPUTER ORGANIZATION:

-Serdil Anıl Ünlü

-1801042672

REPORT:

- I only did part2 in this assignment. Except for part2's mult operation, I can do all of them Alu. In order to do the operations on Alu, I used muxes first, using 2x1 , 2x1s to create an 8x1 and finally a 32x1 mux so that the Alu is ready. I shared the testbenches and test results of the codes I wrote below.

Half adder:

This module performs an addition operation on 2 1-bit inputs without a carry in.

This is test result:

```
# time =  0, a =0, b=0, sum=0, carry_out=0
# time = 20, a =1, b=0, sum=1, carry_out=0
# time = 40, a =0, b=1, sum=1, carry_out=0
# time = 60, a =1, b=1, sum=0, carry_out=1
```

This is half adder testbench

```
`define DELAY 20
module half_adder_testbench();
reg a, b;
wire sum, carry_out;

half_adder hatb (sum, carry_out, a, b);

initial begin
a = 1'b0; b = 1'b0;
#`DELAY;
a = 1'b1; b = 1'b0;
#`DELAY;
a = 1'b0; b = 1'b1;
#`DELAY;
a = 1'b1; b = 1'b1;
end


initial
begin
$monitor("time = %2d, a =%1b, b=%1b, sum=%1b, carry_out=%1b", $time, a, b, sum, carry_out);
end

endmodule
```

FULL ADDER:

This module performs an addition operation on 2 1-bit inputs with a carry in.

FULL ADDER TEST BENCH RESULT:

```
# time =  0, a =0, b=0, carry_in=0, sum=0, carry_out=0
# time = 20, a =0, b=0, carry_in=1, sum=1, carry_out=0
# time = 40, a =0, b=1, carry_in=0, sum=1, carry_out=0
# time = 60, a =0, b=1, carry_in=1, sum=0, carry_out=1
# time = 80, a =1, b=0, carry_in=0, sum=1, carry_out=0
# time = 100, a =1, b=0, carry_in=1, sum=0, carry_out=1
# time = 120, a =1, b=1, carry_in=0, sum=0, carry_out=1
# time = 140, a =1, b=1, carry_in=1, sum=1, carry_out=1
```

FULL ADDER TEST BENCH:

```
module full_adder(sum, carry_out, a, b, carry_in);
input a, b, carry_in;
output sum, carry_out;
wire temp_sum, first_carry_out, second_carry_out;

half_adder first_sum(temp_sum, first_carry_out, a, b);
half_adder second_sum(sum, second_carry_out, temp_sum, carry_in);

or final_carry_out(carry_out, second_carry_out, first_carry_out);

endmodule
```

32 BİT XOR TEST RESULT:

```
# time=                    0
# ,a=00000000000000000000000000001001
# ,b=00000000000000000000000000000011
# ,R=00000000000000000000000000001010
#
# time=                   20
# ,a=00000000000000000000000000000110
# ,b=00000000000000000000000000001000
# ,R=00000000000000000000000000001110
#
# time=                   40
# ,a=00000000000000000000000000000010
# ,b=00000000000000000000000000000101
# ,R=00000000000000000000000000000111
#
# time=                   60
# ,a=00000000000000000000000000000011
# ,b=00000000000000000000000000000111
# ,R=00000000000000000000000000000100
#
```

32 BİT XOR TEST BENCH:

```
`define DELAY 20
module _32bit_xor_testbench();
reg [31:0] a, b;
wire [31:0] R;
_32bit_xor fatb (R,a, b);

initial begin
a = 32'd9; b = 32'd3;
#`DELAY;
a = 32'd6; b = 32'd8;
#`DELAY;
a = 32'd2; b = 32'd5;
#`DELAY;
a = 32'd3; b = 32'd7;
end

initial
begin
$monitor("time=%d\n,a=%b\n,b=%b\n,R=%b\n", $time, a, b, R);
end

endmodule
```

32 Bit or test result:

```
# time=                    0
# ,a=00000000000000000000000000000110
# ,b=00000000000000000000000000000010
# ,R=00000000000000000000000000000110
# time=                   20
# ,a=00000000000000000000000000000110
# ,b=00000000000000000000000000001000
# ,R=00000000000000000000000000001110
# time=                   40
# ,a=00000000000000000000000000000010
# ,b=00000000000000000000000000000101
# ,R=00000000000000000000000000000111
# time=                   60
# ,a=00000000000000000000000000000011
# ,b=00000000000000000000000000000111
# ,R=00000000000000000000000000000111
```

32 Bit or test bench:

```
`define DELAY 20
module _32bit_or_testbench();
reg [31:0] a, b;
wire [31:0] R;
_32bit_or fatb (R,a, b);

initial begin
a = 32'd6; b = 32'd2;
#`DELAY;
a = 32'd6; b = 32'd8;
#`DELAY;
a = 32'd2; b = 32'd5;
#`DELAY;
a = 32'd3; b = 32'd7;
end


initial
begin
$monitor("time=%d\n,a=%b\n,b=%b\n,R=%b", $time, a, b, R);
end

endmodule
```

32 Bit and test result:

```
# time=00000000000000000000000000000000000000000000000000000000000000000000
# ,a=00000000000000000000000000001000
# ,b=00000000000000000000000000000010
# ,R=00000000000000000000000000000000
#
# time=00000000000000000000000000000000000000000000000000000000000000010100
# ,a=00000000000000000000000000000110
# ,b=00000000000000000000000000001000
# ,R=00000000000000000000000000000000
#
# time=00000000000000000000000000000000000000000000000000000000000000101000
# ,a=00000000000000000000000000000010
# ,b=00000000000000000000000000000101
# ,R=00000000000000000000000000000000
#
# time=00000000000000000000000000000000000000000000000000000000000000111100
# ,a=00000000000000000000000000000011
# ,b=00000000000000000000000000000111
# ,R=00000000000000000000000000000011
#
```

32 bit and test bench:

```
`define DELAY 20
module _32bit_and_testbench();
reg [31:0] a, b;
wire [31:0] R;
_32bit_and fatb (R,a, b);

initial begin
a = 32'd8; b = 32'd2;
#`DELAY;
a = 32'd6; b = 32'd8;
#`DELAY;
a = 32'd2; b = 32'd5;
#`DELAY;
a = 32'd3; b = 32'd7;
end


initial
begin
$monitor("time=%b\n,a=%b\n,b=%b\n,R=%b\n", $time, a, b, R);
end

endmodule
```

32 bit nor test result:

```
# time=00
# ,a=00000000000000000000000000001010
# ,b=00000000000000000000000000001000
# ,R=11111111111111111111111111110101
# ,carry_out=z
#
# time=10100
# ,a=00000000000000000000000000000110
# ,b=00000000000000000000000000001000
# ,R=11111111111111111111111111110001
# ,carry_out=z
#
# time=101000
# ,a=00000000000000000000000000000010
# ,b=00000000000000000000000000000101
# ,R=11111111111111111111111111111000
# ,carry_out=z
#
# time=111100
# ,a=00000000000000000000000000000011
# ,b=00000000000000000000000000000111
# ,R=11111111111111111111111111111000
```

32 bit nor test bench:

```
`define DELAY 20
module _32bit_nor_testbench();
reg [31:0] a, b;
wire [31:0] R;
wire carry_out;

_32bit_nor fatb (.R(R), .A(a), .B(b));

initial begin
a = 32'd10; b = 32'd8;
#`DELAY;
a = 32'd6; b = 32'd8;
#`DELAY;
a = 32'd2; b = 32'd5;
#`DELAY;
a = 32'd3; b = 32'd7;
end

initial
begin
$monitor("time=%2b\n,a=%b\n,b=%b\n,R=%b\n,carry_out=%b\n", $time, a, b, R,carry_out);
end

endmodule
```

32 bit slt test result:

```
# time=                    0
# ,slt=0
# ,a=          6
# ,b=          5
#
# time=                   20
# ,slt=1
# ,a=          6
# ,b=          8
#
# time=                   40
# ,slt=1
# ,a=          2
# ,b=          5
#
# time=                   60
# ,slt=1
# ,a=          3
# ,b=          7
#
```

32 bit slt test bench:

```
`define DELAY 20
module _32_slt_testbench();
reg [31:0] a, b;
wire [31:0] sum;
wire carry_out;

_32bit_slt fatb (slt,a, b);

initial begin
a = 32'd6; b = 32'd5;
#`DELAY;
a = 32'd6; b = 32'd8;
#`DELAY;
a = 32'd2; b = 32'd5;
#`DELAY;
a = 32'd3; b = 32'd7;
#`DELAY;

end

initial
begin
$monitor("time=%d\n,slt=%d\n,a=%d\n,b=%d\n", $time, slt,a, b);
end

endmodule
```

32 bit subtract test result:

```
# time=00
# ,a=00000000000000000000000000000110
# ,b=00000000000000000000000000000101
# ,sum=00000000000000000000000000000001
# ,carry_out=1
#
# time=10100
# ,a=00000000000000000000000000000110
# ,b=00000000000000000000000000001000
# ,sum=11111111111111111111111111111110
# ,carry_out=0
#
# time=101000
# ,a=00000000000000000000000000000010
# ,b=00000000000000000000000000000101
# ,sum=11111111111111111111111111111101
# ,carry_out=0
#
# time=111100
# ,a=00000000000000000000000000000011
# ,b=00000000000000000000000000000111
# ,sum=11111111111111111111111111111100
# ,carry_out=0
#
```

32 bit subtract test bench:

```
`define DELAY 20
module _32_sub_test();
reg [31:0] a, b;
wire [31:0] sum;
wire carry_out;

_32_sub fatb (sum, carry_out, a, b);

initial begin
a = 32'd6; b = 32'd5;
#`DELAY;
a = 32'd6; b = 32'd8;
#`DELAY;
a = 32'd2; b = 32'd5;
#`DELAY;
a = 32'd3; b = 32'd7;
#`DELAY;

end

initial
begin
$monitor("time=%2b\n,a=%b\n,b=%b\n,sum=%b\n,carry_out=%b\n", $time, a, b, sum, carry_out);
end

endmodule
```

ALU:

This module performs alu operations according to the 3-bit op code. 000 = add, 001 = xor, 010 = sub, 011 = mult, 100 = slt, 101 = nor, 110 = and, 111 = or. I couldn't do the mult part, so when op code is 100 (mult), result is 0.

ALU TEST BENCH:

```
`define DELAY 20
module Alu_testbench();
reg [31:0] A, B;
wire [31:0] R;
reg op0,op1,op2;
Alu fatb (R,op0,op1,op2,A,B);

initial begin
op0 = 1'b0;
op1 = 1'b0;
op2 = 1'b0;
A = 32'd2; B = 32'd6;
#`DELAY;
op0 = 1'b0;
op1 = 1'b0;
op2 = 1'b1;
#`DELAY;
op0 = 1'b0;
op1 = 1'b1;
op2 = 1'b0;
#`DELAY;
op0 = 1'b0;
op1 = 1'b1;
op2 = 1'b1;
#`DELAY;
op0 = 1'b1;
op1 = 1'b0;
op2 = 1'b0;

#`DELAY;
op0 = 1'b1;
op1 = 1'b0;
op2 = 1'b1;
#`DELAY;
op0 = 1'b1;
op1 = 1'b1;
op2 = 1'b0;
#`DELAY;
op0 = 1'b1;
op1 = 1'b1;
op2 = 1'b1;
end

initial
begin
$monitor("time=%d\n,A=%b\n,B=%b\n,op0=%b\n,op1=%b\n,op2=%b\n,R=%b\n", $time, A, B, op0,op1,op2,R);
end

endmodule
```

ALU TEST RESULT:

```
# time=                    0
# ,A=00000000000000000000000000000010
# ,B=00000000000000000000000000000110
# ,op0=0
# ,op1=0
# ,op2=0
# ,R=00000000000000000000000000001000
#
# time=                   20
# ,A=00000000000000000000000000000010
# ,B=00000000000000000000000000000110
# ,op0=0
# ,op1=0
# ,op2=1
# ,R=00000000000000000000000000000100
#
# time=                   40
# ,A=00000000000000000000000000000010
# ,B=00000000000000000000000000000110
# ,op0=0
# ,op1=1
# ,op2=0
# ,R=11111111111111111111111111111100
#
# time=                   60
# ,A=00000000000000000000000000000010
# ,B=00000000000000000000000000000110
# ,op0=0
# ,op1=1
# ,op2=1
# ,R=00000000000000000000000000000000
#

 time=                   80
 ,A=00000000000000000000000000000010
 ,B=00000000000000000000000000000110
 ,op0=1
 ,op1=0
 ,op2=0
 ,R=00000000000000000000000000000001

 time=                  100
 ,A=00000000000000000000000000000010
 ,B=00000000000000000000000000000110
 ,op0=1
 ,op1=0
 ,op2=1
 ,R=11111111111111111111111111111001

 time=                  120
 ,A=00000000000000000000000000000010
 ,B=00000000000000000000000000000110
 ,op0=1
 ,op1=1
 ,op2=0
 ,R=00000000000000000000000000000010

 time=                  140
 ,A=00000000000000000000000000000010
 ,B=00000000000000000000000000000110
 ,op0=1
 ,op1=1
 ,op2=1
 ,R=00000000000000000000000000000110
```

- I also tested the ALU with other numbers, the test result is as follows.

With the other numbers ALU test result:

```
# ,A=00000000000000000000000000000100
# ,B=00000000000000000000000000001001
# ,op0=0
# ,op1=0
# ,op2=0
# ,R=00000000000000000000000000001101
#
# time=                    20
# ,A=00000000000000000000000000000100
# ,B=00000000000000000000000000001001
# ,op0=0
# ,op1=0
# ,op2=1
# ,R=00000000000000000000000000001101
#
# time=                    40
# ,A=00000000000000000000000000000100
# ,B=00000000000000000000000000001001
# ,op0=0
# ,op1=1
# ,op2=0
# ,R=11111111111111111111111111111011
#
# time=                    60
# ,A=00000000000000000000000000000100
# ,B=00000000000000000000000000001001
# ,op0=0
# ,op1=1
# ,op2=1
# ,R=00000000000000000000000000000000
#
```

```
# ,A=00000000000000000000000000000100
# ,B=00000000000000000000000000001001
# ,op0=1
# ,op1=0
# ,op2=0
# ,R=00000000000000000000000000000001
#
# time=                   100
# ,A=00000000000000000000000000000100
# ,B=00000000000000000000000000001001
# ,op0=1
# ,op1=0
# ,op2=1
# ,R=11111111111111111111111111110010
#
# time=                   120
# ,A=00000000000000000000000000000100
# ,B=00000000000000000000000000001001
# ,op0=1
# ,op1=1
# ,op2=0
# ,R=00000000000000000000000000000000
#
# time=                   140
# ,A=00000000000000000000000000000100
# ,B=00000000000000000000000000001001
# ,op0=1
# ,op1=1
# ,op2=1
# ,R=00000000000000000000000000001101
#
```

With the other numbers Alu test result:

```
# time=                    0
# ,A=00000000000000000000000000001100
# ,B=00000000000000000000000000000111
# ,op0=0
# ,op1=0
# ,op2=0
# ,R=00000000000000000000000000010011
#
# time=                   20
# ,A=00000000000000000000000000001100
# ,B=00000000000000000000000000000111
# ,op0=0
# ,op1=0
# ,op2=1
# ,R=00000000000000000000000000001011
#
# time=                   40
# ,A=00000000000000000000000000001100
# ,B=00000000000000000000000000000111
# ,op0=0
# ,op1=1
# ,op2=0
# ,R=00000000000000000000000000000101
#
# time=                   60
# ,A=00000000000000000000000000001100
# ,B=00000000000000000000000000000111
# ,op0=0
# ,op1=1
# ,op2=1

# time=                   80
# ,A=00000000000000000000000000001100
# ,B=00000000000000000000000000000111
# ,op0=1
# ,op1=0
# ,op2=0
# ,R=00000000000000000000000000000000
#
# time=                  100
# ,A=00000000000000000000000000001100
# ,B=00000000000000000000000000000111
# ,op0=1
# ,op1=0
# ,op2=1
# ,R=11111111111111111111111111110000
#
# time=                  120
# ,A=00000000000000000000000000001100
# ,B=00000000000000000000000000000111
# ,op0=1
# ,op1=1
# ,op2=0
# ,R=00000000000000000000000000000100
#
# time=                  140
# ,A=00000000000000000000000000001100
# ,B=00000000000000000000000000000111
# ,op0=1
# ,op1=1
# ,op2=1
```