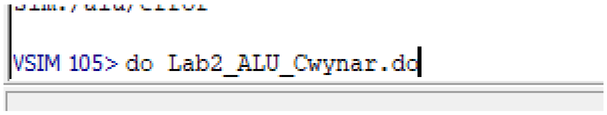
Jack Cwynar

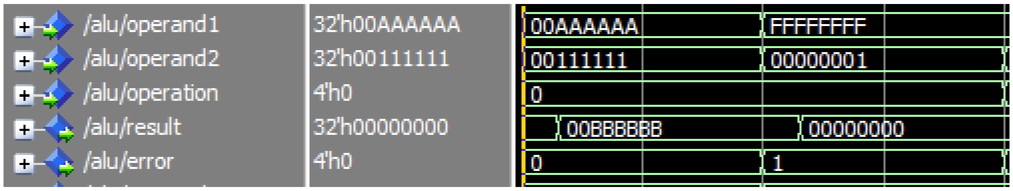
Lab 2 Testing/Simulation Screenshots

Starting Sim



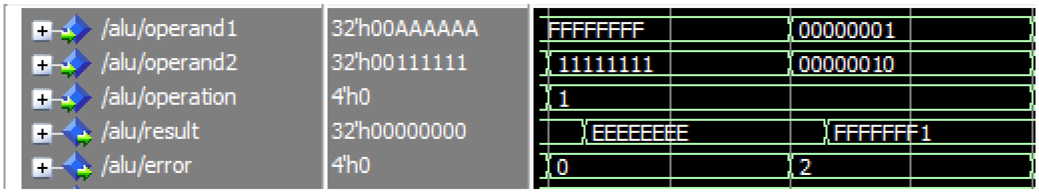
```
sim:/alu/error

VSIM 105> do Lab2_ALU_Cwynar.do
```

Unsigned Add Tests (normal, overflow)



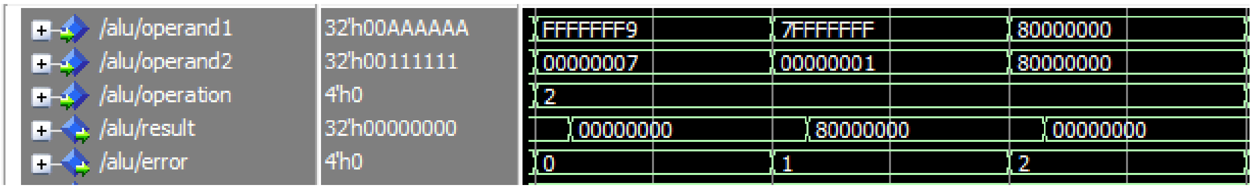
```
/alu/operand1   32'h00AAAAAA   00AAAAAA             FFFFFFFF
/alu/operand2   32'h00111111   00111111             00000001
/alu/operation  4'h0           0
/alu/result     32'h00000000      00BBBBBB             00000000
/alu/error      4'h0           0                    1
```

Unsigned Sub Tests (normal, underflow)



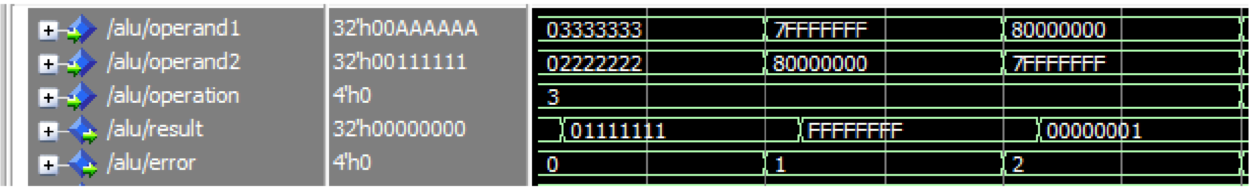
```
/alu/operand1   32'h00AAAAAA   FFFFFFFF             00000001
/alu/operand2   32'h00111111   11111111             00000010
/alu/operation  4'h0           1
/alu/result     32'h00000000      EEEEEEEE             FFFFFFF1
/alu/error      4'h0           0                    2
```

2’s Complement Add (normal, overflow, underflow)



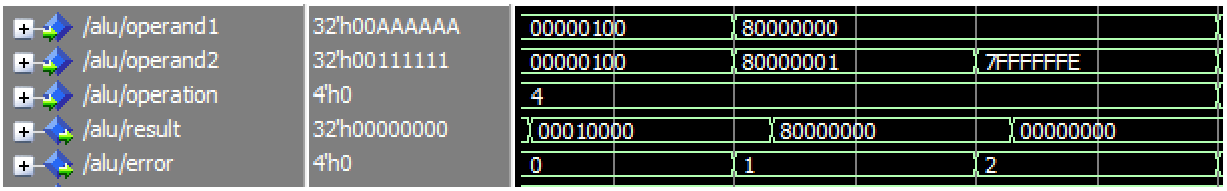
```
/alu/operand1   32'h00AAAAAA   FFFFFFF9             7FFFFFFF             80000000
/alu/operand2   32'h00111111   00000007             00000001             80000000
/alu/operation  4'h0           2
/alu/result     32'h00000000      00000000             80000000             00000000
/alu/error      4'h0           0                    1                    2
```

2’s Complement Subtract (normal, overflow, underflow)

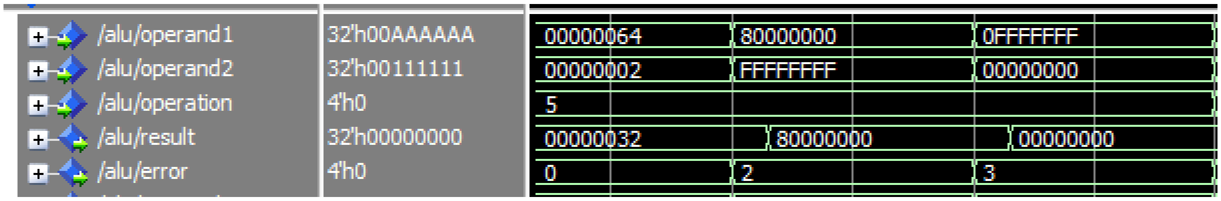


```
/alu/operand1   32'h00AAAAAA   03333333             7FFFFFFF             80000000
/alu/operand2   32'h00111111   02222222             80000000             7FFFFFFF
/alu/operation  4'h0           3
/alu/result     32'h00000000      01111111             FFFFFFFF             00000001
/alu/error      4'h0           0                    1                    2
```

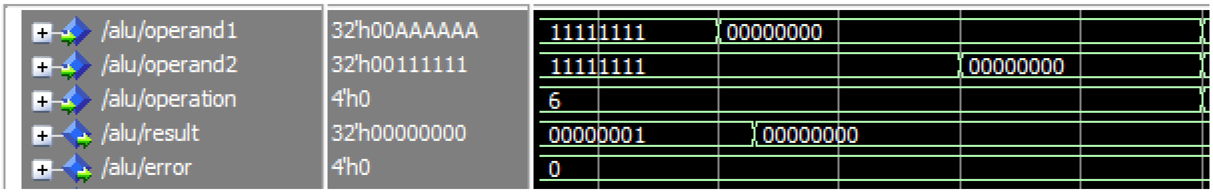
2's Complement Multiply (normal, overflow, underflow)


/alu/operand1 32'h00AAAAAA
/alu/operand2 32'h00111111
/alu/operation 4'h0
/alu/result 32'h00000000
/alu/error 4'h0

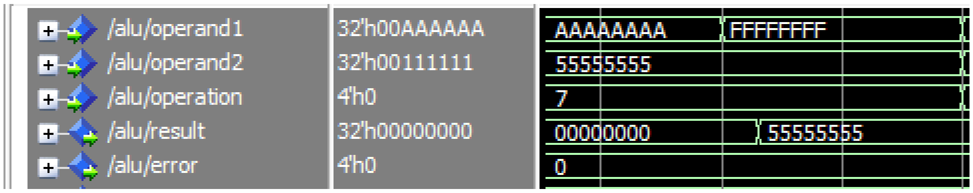
2's Complement Divide (normal, underflow, divide by zero)


/alu/operand1 32'h00AAAAAA
/alu/operand2 32'h00111111
/alu/operation 4'h0
/alu/result 32'h00000000
/alu/error 4'h0

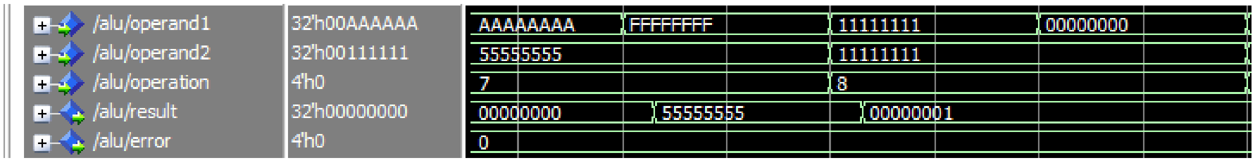
Logical AND (1 AND 1, 0 AND 1, 0 AND 0)


/alu/operand1 32'h00AAAAAA
/alu/operand2 32'h00111111
/alu/operation 4'h0
/alu/result 32'h00000000
/alu/error 4'h0

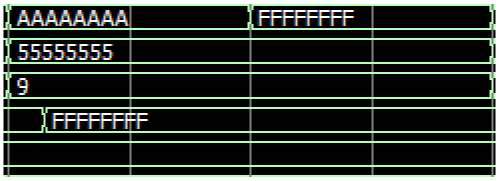
Bitwise AND


/alu/operand1 32'h00AAAAAA
/alu/operand2 32'h00111111
/alu/operation 4'h0
/alu/result 32'h00000000
/alu/error 4'h0

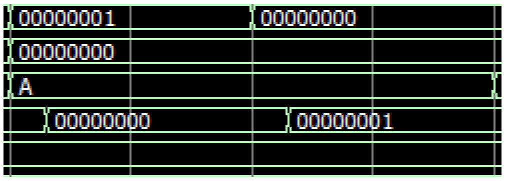
Logical OR (1 OR 1, 0 OR 1)


/alu/operand1 32'h00AAAAAA
/alu/operand2 32'h00111111
/alu/operation 4'h0
/alu/result 32'h00000000
/alu/error 4'h0

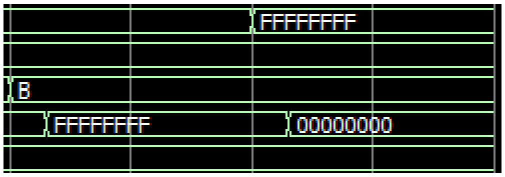
Bitwise OR



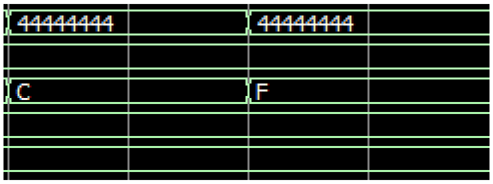
Logical NOT of operand1



Bitwise NOT of operand1



Checking “others” (result remains 0 after Bitwise NOT test)


44444444
44444444
C
F