Team Design #2: 32-bit ALU

COVER SHEET

Honor Code:	I have neither give	en or received, n	or have I tolerated others' use
	of una	uthorized aid	
Name:	Joe Leveille	Signature:	Joseph Loveisse
Honor Code:	I have neither give	en or received, no	or have I tolerated others' use
	of unaut	horized aid	
Name:	_Jon Bayert	Signature:	Jonathan Bayert
1) Your VH	erpage stapled on the fro IDL code veform results: 6 pages		nction

```
library ieee;
    use ieee.std logic 1164.all;
    use ieee.std logic misc.or_reduce;
 3
 4
    ENTITY alu32 IS
 5
         PORT( A, B: in std logic vector (31 downto 0);
 6
             ALUOp: in std_logic_vector (3 downto 0);
 7
             RESULT: out std logic vector (31 downto 0);
8
             Z, V, C: out std logic );
9
    END alu32;
10
11
    architecture big alu of alu32 is
12
         signal intRes, carry: std logic vector(31 downto 0);
13
         signal set : std logic;
14
         component bit slice port(
15
             a,b,Less,Ainvert,carryIn
                                            : in std logic;
16
                                             : in std logic vector (2 downto 0);
17
             result, carryOut
                                             : out std logic
18
          ); end component;
19
          component MSB slice port(
20
             a,b,Less,Ainvert,carryIn
                                            : in std logic;
21
                                            : in std logic vector (2 downto 0);
22
            result, carryOut, set, overflow
                                            : out std logic
         ); end component;
23
24 begin
25
         LSB: bit slice PORT MAP (
26
             a=>A(0), b=>B(0), Less=>set, Ainvert=>ALUOp(3), carryIn=>ALUOp(2),
27
             Op=>ALUOp(2 downto 0),
28
             result=>intRes(0), carryOut=>carry(0));
29
        bitSliceSetup: for x in 1 to 30 generate
30
            BITx: bit slice PORT MAP (
31
                 a=>A(x), b=>B(x), Less=>'0', Ainvert=>ALUOp(3), carryIn=>carry(x-1),
32
                 Op=>ALUOp(2 downto 0),
33
                 result=>intRes(x), carryOut=>carry(x));
34
         end generate bitSliceSetup;
35
        MSB: MSB slice PORT MAP (
36
                 a=>A(31), b=>B(31), Less=>'0', Ainvert=>ALUOp(3), carryIn=>carry(30),
37
                 Op=>ALUOp(2 downto 0),
38
                 result=>intRes(31), carryOut=>C,set=>set, overflow=>V);
39
40
        --Handle zero flag
41
         Z <= not or reduce(intRes);</pre>
42
        --connect result output to signal
43
        result <= intRes;
44
45
    end big alu;
46
47
```

48

```
library ieee;
    use ieee.std logic 1164.all;
 3
 4
    entity bit slice is
5
     port(
6
        a,b,Less,Ainvert,carryIn : in std_logic;
 7
        Οp
                                   : in std logic vector (2 downto 0);
8
        result,carryOut
                                   : out std logic
9
     );
10
    end bit slice;
11
12
    architecture stdBitSlice of bit slice is
13
     signal aMux, bMux, andOut, orOut : std logic;
14
      signal AxorB, AandB, sum, CandXor : std logic;
15
   begin
16
    --Inverter muxes
17
      aMux <= a xor Ainvert;
18
     bMux \leq= b xor Op(2);
                                       -- Assuming Binvert is MSB of Op
19
20
     --Basic gates
21
     andOut <= aMux and bMux;
22
     orOut <= aMux or bMux;
23
24
     --Following is adder block
25
     AxorB <= aMux xor bMux;
26
     AandB <= aMux and bMux;
     CandXor <= AxorB and carryIn;
27
     sum <= AxorB xor carryIn ;
28
29
     carryOut <= AandB or CandXor;</pre>
30
31
      --Big mux
32
      result \leftarrow (not Op(1) and not Op(0) and andOut) or
33
                 (not Op(1) and Op(0) and orOut) or
34
                 (Op(1) and not Op(0) and sum) or
35
                 (Op(1) \text{ and } Op(0) \text{ and Less});
36
37
    end stdBitSlice;
38
```

```
library ieee;
    use ieee.std logic 1164.all;
 3
 4
    entity msb slice is
 5
     port(
6
        a,b,Less,Ainvert,carryIn
                                  : in std_logic;
 7
                                       : in std logic vector (2 downto 0);
8
        result, carryOut, set, overflow : out std logic
9
      );
10
    end msb slice;
11
12
    architecture MSBBitSlice of msb slice is
13
     signal aMux, bMux, andOut, orOut : std logic;
14
      signal AxorB, AandB, sum, CandXor : std logic;
15
   begin
16
    --Inverter muxes
17
      aMux <= a xor Ainvert;
18
     bMux \leq= b xor Op(2);
                                       -- Assuming Binvert is MSB of Op
19
20
     --Basic gates
21
     andOut <= aMux and bMux;
22
     orOut <= aMux or bMux;
23
24
     --Following is adder block
25
     AxorB <= aMux xor bMux;
26
     AandB <= aMux and bMux;
     CandXor <= AxorB and carryIn;
27
     sum <= AxorB xor carryIn ;
28
29
     carryOut <= AandB or CandXor;</pre>
30
31
      --Big mux
32
       result \leftarrow (not Op(1) and not Op(0) and andOut) or
33
                 (not Op(1) and Op(0) and orOut) or
                 (Op(1) and not Op(0) and sum) or
34
35
                 (Op(1) \text{ and } Op(0) \text{ and Less});
36
37
      --MSB additions
38
      set <= sum;
39
      overflow <= carryIn xor (AandB or CandXor);</pre>
40
41 end MSBBitSlice;
```

/alu32/A	0000001	0000FFFF	7FFFFFF	F000FFFF	A0A0A1A1
/alu32/B	FFFFFFF	FFFF0000	7FFFFFF	7000FFFF	7151B17B
u32/ALUOp	0				
32/RESULT	0000001	00000000	7FFFFFF	7000FFFF	2000A121
			<u> </u>	†	†
/alu32/V	†				
/alu32/C	1				
/ u.u.s.z./ c					

/alu32/A	0000001	0000FFFF	7FFFFFF	F000FFFF	A0A0A1A1
/alu32/B	FFFFFFF	FFFF0000	7FFFFFF	7000FFFF	7151B17B
32/ALUOp	1				
2/RESULT	FFFFFFF		7FFFFFF	F000FFFF	F1F1B1FB
					† †
/alu32/V	†				†
/alu32/C	1				
		100 ps 150 ps			

/alu32/A 1	65535	2147483647	-268369921	-1600085599
/alu32/B (-1	-65536	2147483647	1879113727	1901179259
32/ALUOp 2				
2/RESULT 0	-1	-2	1610743806	301093660
/alu32/Z	<u> </u>		†	
/alu32/V				
/alu32/C				
, , .				

/alu32/A		65535	2147483647	-268369921	1600085599
/alu32/B	-1	-65536	2147483647	1879113727	1901179259
32/ALUOp	6			•	
32/RESULT	2	131071	0	-2147483648	793702438
/alu32/Z					†
, , ,					

/alu32/A		65535	2147483647	-268369921	-1600085599
/alu32/B	(-1	-65536	2147483647	1879113727	1901179259
32/ALUOp (7				
2/RESULT	0		0	1	0
/alu32/Z	+				
/alu32/V լ	†				
, , ,					

	0000001	0000FFFF		7FFFFFF		F000FFFF	-	A0A0A1A1	+
/alu32/B	FFFFFFF	FFFF0000		7FFFFFF		7000FFFF		7151B17B	-
ı32/ALUOp	С								
32/RESULT	00000000			80000000		0FFF0000		0E0E4E04	
/alu32/Z [[]			·						
/alu32/V լ			•		<u> </u>	-			
/alu32/C _l			-	-		-	-	†	