

Lab 5 Report

VHDL Source:

```

10  LIBRARY ieee;
11  USE ieee.std_logic_1164.all;
12  USE ieee.std_logic_arith.all;
13
14  ENTITY LargerFloat IS
15      GENERIC (width : POSITIVE := 8); -- Number of floating point number bits
16      PORT (A, B : IN std_logic_vector(width-1 DOWNT0 0);
17            Larger, Smaller : OUT std_logic_vector(width - 1 DOWNT0 0));
18  END ENTITY LargerFloat;
19
20  --
21  ARCHITECTURE Behavior OF LargerFloat IS
22  BEGIN
23      PROCESS(A,B)
24      BEGIN
25          for i in (width-1) DOWNT0 0 loop
26              if A(i)>B(i) then
27                  Smaller <= B;
28                  Larger <= A;
29                  EXIT;
30              elsif A(i)<B(i) then
31                  Smaller <= A;
32                  Larger <= B;
33                  EXIT;
34              elsif i=0 and A(0)=B(0) then
35                  Smaller <=A;
36                  Larger <=B;
37                  EXIT;
38              else
39                  next;
40              end if;
41          END LOOP;
42      END PROCESS;
43  END ARCHITECTURE Behavior;
44

```

VHDL code of the finding the larger of two floating point numbers

```

10  LIBRARY ieee;
11  USE ieee.std_logic_1164.all;
12  use ieee.numeric_std.all;
13
14
15  ENTITY AddSubtract IS
16      GENERIC (width : POSITIVE := 8); -- Number of fixed point number bits
17      PORT (Left, Right : IN std_logic_vector(width - 1 DOWNT0 0); -- Inputs
18            Sum : OUT std_logic_vector(width DOWNT0 0); -- Output
19            AddSub : IN std_logic); -- Control to choose Add or Subtract
20  END ENTITY AddSubtract;
21
22  --
23  ARCHITECTURE Behavior OF AddSubtract IS
24  BEGIN
25      PROCESS (Left, Right, AddSub)
26      Begin
27          if AddSub='1' then
28              Sum<=std_logic_vector(unsigned('0' & Left)-(unsigned('0' & Right)));
29          else
30              Sum<=std_logic_vector(unsigned('0' & Right)+(unsigned('0' & Left)));
31          end if;
32      End Process;
33  END ARCHITECTURE Behavior;
34
35

```

VHDL code of the fixed point adder and subtractor

```

10  LIBRARY ieee;
11  USE ieee.std_logic_1164.all;
12  use ieee.numeric_std.all;
13
14  ENTITY BarrelShifter IS
15      GENERIC (width : POSITIVE := 8; -- Number of input and output bits
16              ControlBits : POSITIVE := 4);
17      PORT (A : IN std_logic_vector(width -1 DOWNT0 0);
18            Y : OUT std_logic_vector(width -1 DOWNT0 0);
19            RightShifts : IN std_logic_vector(ControlBits -1 DOWNT0 0));
20  END ENTITY BarrelShifter;
21
22  --
23  ARCHITECTURE Behavior OF BarrelShifter IS
24  BEGIN
25      PROCESS(A, RightShifts)
26      BEGIN
27          if (RightShifts(ControlBits-1)='0') then
28              Y<=A srl to_integer(unsigned(RightShifts));
29          else
30              Y<=A sll (to_integer(unsigned(not RightShifts))+1);
31          end if;
32      END PROCESS;
33  END ARCHITECTURE Behavior;
34
35

```

VHDL code of the variable-bit shifter

```

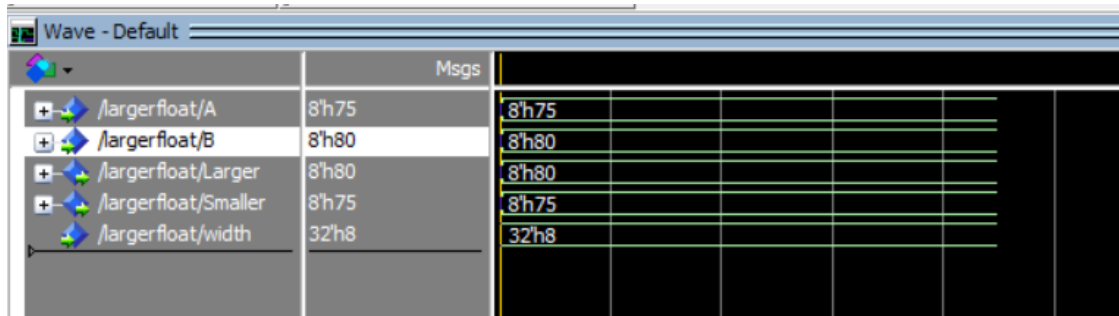
10  LIBRARY ieee;
11  USE ieee.std_logic_1164.all;
12  use ieee.numeric_std.all;
13
14  ENTITY NormalShiftCount IS
15      GENERIC (width : POSITIVE := 8; -- Number of mantissa bits
16              Shifts : POSITIVE := 3); -- Number of bits in shift count
17      PORT (A : IN std_logic_vector(width - 1 DOWNT0 0);
18           S : OUT std_logic_vector(shifts -1 DOWNT0 0);
19           NoOnes : OUT std_logic); -- NoOnes is asserted if the mantissa is all zeros
20  END ENTITY NormalShiftCount;
21
22  --
23  ARCHITECTURE Behavior OF NormalShiftCount IS
24  BEGIN
25      Process(A)
26      begin
27          for i in (width-1) DOWNT0 0 loop
28              if A(i)='1' then
29                  S<= std_logic_vector(to_signed((i - (width-2)), S'length));
30                  NoOnes<='0';
31                  EXIT;
32              end if;
33              if (A = (A'range => '0')) then
34                  NoOnes<='1';
35                  S<=(others=>'0');
36                  EXIT;
37              end if;
38          END loop;
39      end process;
40  END ARCHITECTURE Behavior;
41

```

VHDL code of finding the most significant '1'

Test and Results:

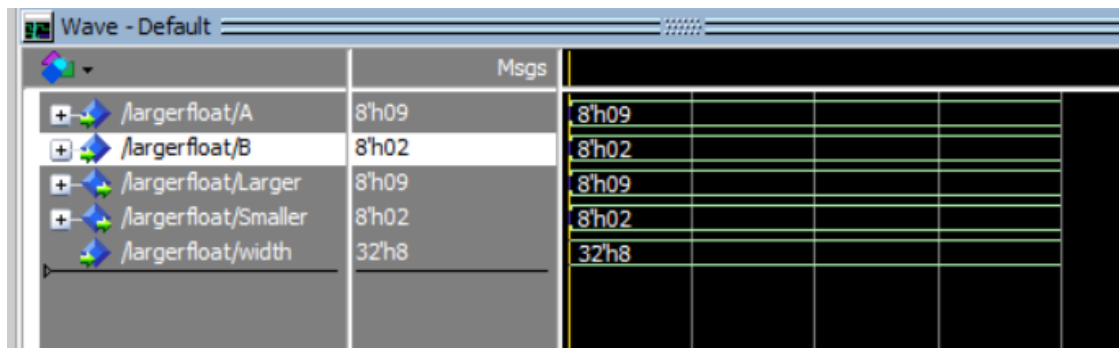
Finding the Larger of Two Floating Point Numbers



The screenshot shows a logic simulator window titled "Wave - Default". It displays a tree view on the left with variables: /largerfloat/A, /largerfloat/B, /largerfloat/Larger, /largerfloat/Smaller, and /largerfloat/width. The central "Msgs" column shows the values for these variables. The rightmost column shows a waveform with green horizontal lines representing signal levels over time.

Variable	Value
/largerfloat/A	8'h75
/largerfloat/B	8'h80
/largerfloat/Larger	8'h80
/largerfloat/Smaller	8'h75
/largerfloat/width	32'h8

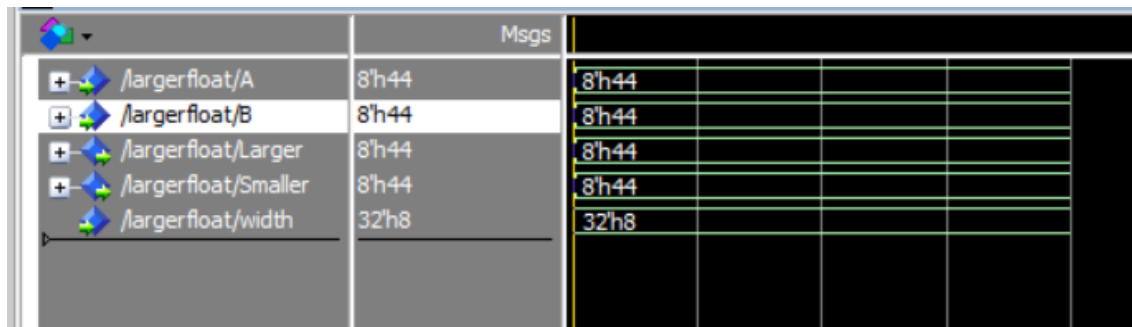
The variable A is set to 75 and the variable B is set to 80. The simulation determined that the larger value is 80 and the smaller value is 75, as seen in the figure above.



The screenshot shows a logic simulator window titled "Wave - Default". It displays a tree view on the left with variables: /largerfloat/A, /largerfloat/B, /largerfloat/Larger, /largerfloat/Smaller, and /largerfloat/width. The central "Msgs" column shows the values for these variables. The rightmost column shows a waveform with green horizontal lines representing signal levels over time.

Variable	Value
/largerfloat/A	8'h09
/largerfloat/B	8'h02
/largerfloat/Larger	8'h09
/largerfloat/Smaller	8'h02
/largerfloat/width	32'h8

The variable A is set to 9 and the variable B is set to 2. The simulation determined that the larger value is 9 and the smaller value is 2, as seen in the figure above.

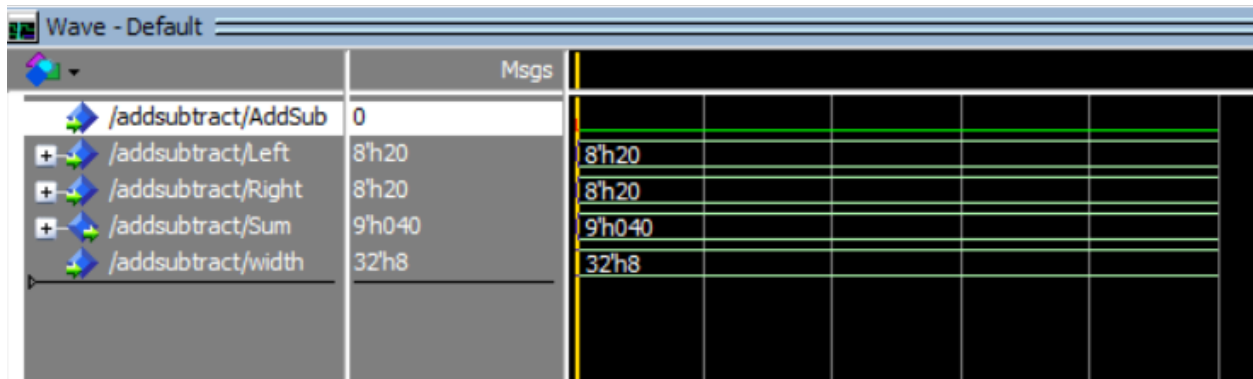


The screenshot shows a logic simulator window titled "Wave - Default". It displays a tree view on the left with variables: /largerfloat/A, /largerfloat/B, /largerfloat/Larger, /largerfloat/Smaller, and /largerfloat/width. The central "Msgs" column shows the values for these variables. The rightmost column shows a waveform with green horizontal lines representing signal levels over time.

Variable	Value
/largerfloat/A	8'h44
/largerfloat/B	8'h44
/largerfloat/Larger	8'h44
/largerfloat/Smaller	8'h44
/largerfloat/width	32'h8

Both A and B were set to 44. The simulation determined that the larger and smaller value were both 44.

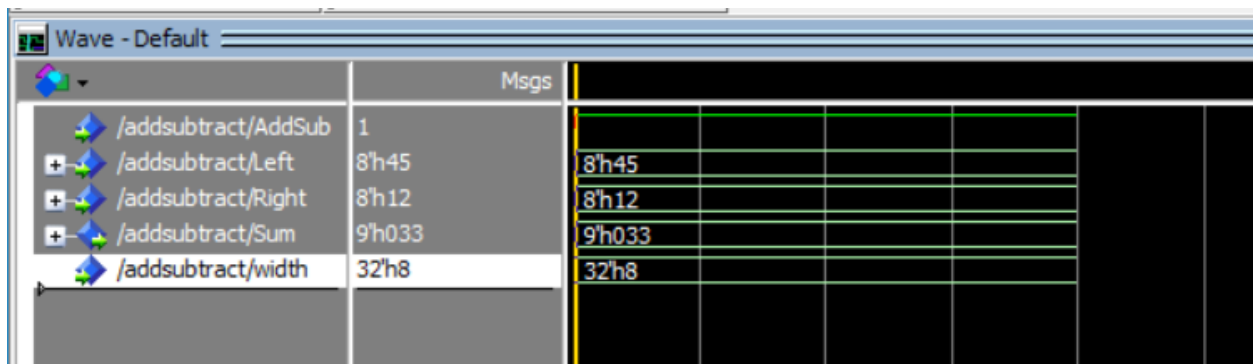
A Fixed Point Adder/Subtractor



The screenshot shows a logic simulator window titled "Wave - Default". It displays a configuration for a fixed-point adder/subtractor. The configuration table has two columns: a list of variables and their values, and a "Msgs" column. The variables and their values are: /addsubtract/AddSub (0), /addsubtract/Left (8'h20), /addsubtract/Right (8'h20), /addsubtract/Sum (9'h040), and /addsubtract/width (32'h8). The "Msgs" column shows a series of green horizontal lines, indicating that the simulation is running and the values are being updated.

	Msgs
/addsubtract/AddSub	0
/addsubtract/Left	8'h20
/addsubtract/Right	8'h20
/addsubtract/Sum	9'h040
/addsubtract/width	32'h8

The AddSum variable is asserted '0' which means to add the left input to the right input. With the left input set to 20 and the right input set to 20, the sum is 40, which matches the simulation.



The screenshot shows a logic simulator window titled "Wave - Default". It displays a configuration for a fixed-point adder/subtractor. The configuration table has two columns: a list of variables and their values, and a "Msgs" column. The variables and their values are: /addsubtract/AddSub (1), /addsubtract/Left (8'h45), /addsubtract/Right (8'h12), /addsubtract/Sum (9'h033), and /addsubtract/width (32'h8). The "Msgs" column shows a series of green horizontal lines, indicating that the simulation is running and the values are being updated.

	Msgs
/addsubtract/AddSub	1
/addsubtract/Left	8'h45
/addsubtract/Right	8'h12
/addsubtract/Sum	9'h033
/addsubtract/width	32'h8

The AddSum variable is asserted '1' which means to subtract the right input from the left input. With the left input set to 45 and the right input set to 33, the subtraction is 33, which matches the value assigned to the sum variable.

A Variable Bit-Shifter

Wave - Default			Msgs			
+ /barrelshifter/A	8'h03		8'h03			
+ /barrelshifter/RightShifts	4'h1		4'h1			
+ /barrelshifter/Y	8'h01		8'h01			
/barrelshifter/ControlBits	32'h4		32'h4			
/barrelshifter/width	32'h8		32'h8			

The input variable A was set to 3. The RightShifts variable was set to 1. This moves A one bit to the right, which results in Y being equal to 1.

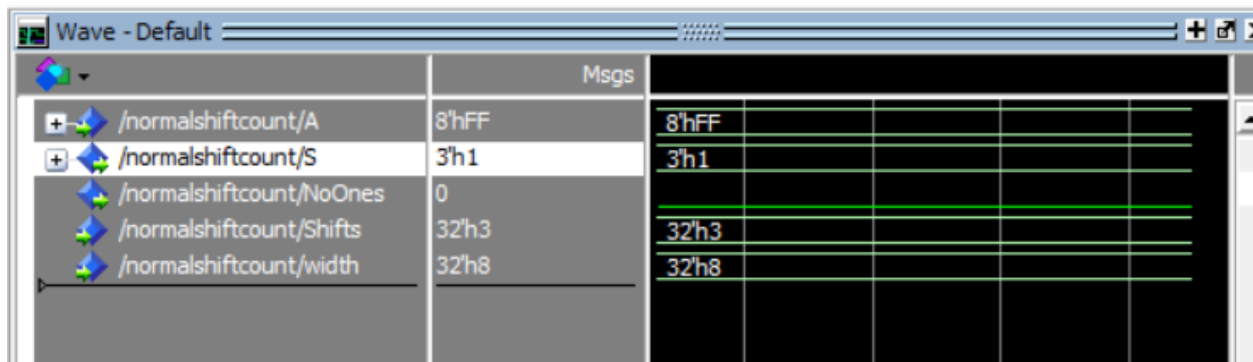
Wave - Default			Msgs			
+ /barrelshifter/A	8'h03		8'h03			
+ /barrelshifter/RightShifts	-2		-2			
+ /barrelshifter/Y	8'h0C		8'h0C			
/barrelshifter/width	32'h8		32'h8			
/barrelshifter/ControlBits	32'h4		32'h4			

The input variable A was set to 3. The RightShifts variable was set to -2. This moves A 2 bits to the left, which results in Y being equal to 12. 12 is C in hex and this is correctly displayed in the figure above.

			Msgs			
+ /barrelshifter/A	8'h03		8'h03			
+ /barrelshifter/RightShifts	0		0			
+ /barrelshifter/Y	8'h03		8'h03			
/barrelshifter/width	32'h8		32'h8			
/barrelshifter/ControlBits	32'h4		32'h4			

The input variable A was set to 3. Since the RightShifts variable was set to 0, A does not move any bits. This results in Y being equal to 3.

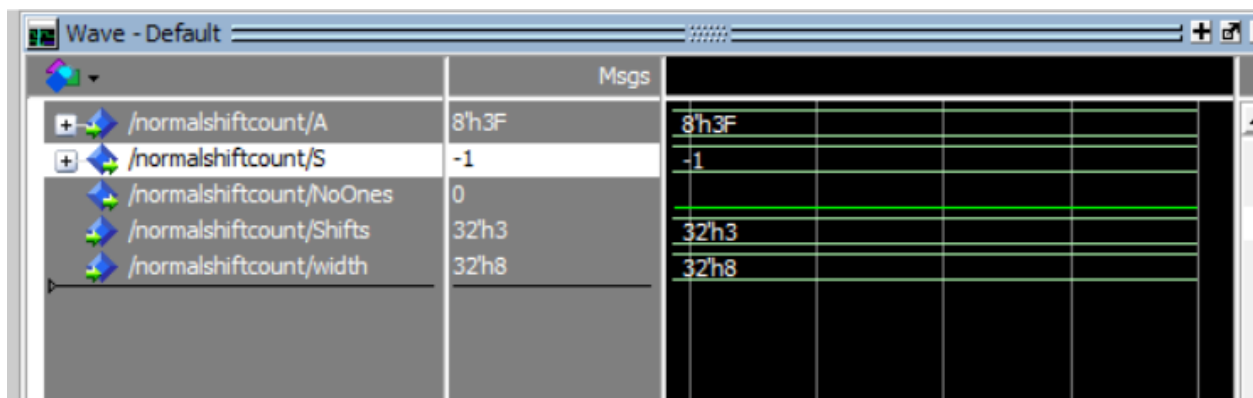
Finding the Most Significant '1'



The screenshot shows the 'Wave - Default' window with a list of variables on the left and a 'Msgs' table on the right. The variables are: /normalshiftcount/A (8'hFF), /normalshiftcount/S (3'h1), /normalshiftcount/NoOnes (0), /normalshiftcount/Shifts (32'h3), and /normalshiftcount/width (32'h8). The 'Msgs' table has 5 columns and 5 rows, with the first row containing the values 8'hFF, 3'h1, and three empty cells.

	Msgs
/normalshiftcount/A	8'hFF
/normalshiftcount/S	3'h1
/normalshiftcount/NoOnes	0
/normalshiftcount/Shifts	32'h3
/normalshiftcount/width	32'h8

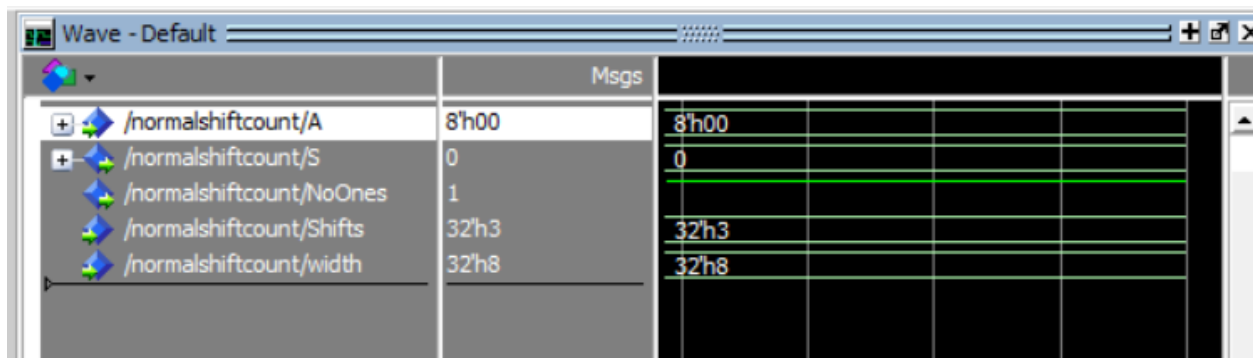
The input variable A is set to FF, which means that the most significant bit is the 8th one. As a result, S outputs 1, indicating how much should be shifted to the right.



The screenshot shows the 'Wave - Default' window with a list of variables on the left and a 'Msgs' table on the right. The variables are: /normalshiftcount/A (8'h3F), /normalshiftcount/S (-1), /normalshiftcount/NoOnes (0), /normalshiftcount/Shifts (32'h3), and /normalshiftcount/width (32'h8). The 'Msgs' table has 5 columns and 5 rows, with the first row containing the values 8'h3F, -1, and three empty cells.

	Msgs
/normalshiftcount/A	8'h3F
/normalshiftcount/S	-1
/normalshiftcount/NoOnes	0
/normalshiftcount/Shifts	32'h3
/normalshiftcount/width	32'h8

The input variable A is set to 3F, which means that the most significant bit is the 6th one. As a result, S outputs -1, as the amount that needs to be shifted to the right.

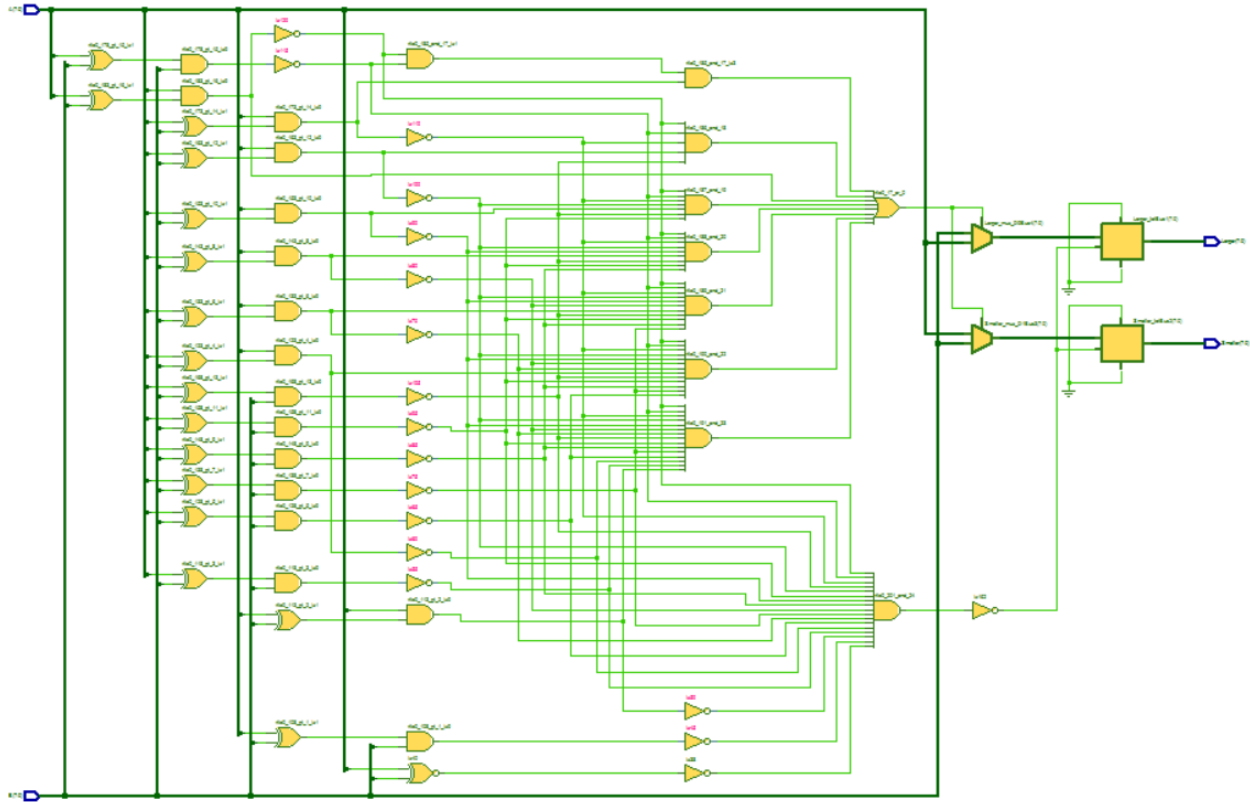


The screenshot shows the 'Wave - Default' window with a list of variables on the left and a 'Msgs' table on the right. The variables are: /normalshiftcount/A (8'h00), /normalshiftcount/S (0), /normalshiftcount/NoOnes (1), /normalshiftcount/Shifts (32'h3), and /normalshiftcount/width (32'h8). The 'Msgs' table has 5 columns and 5 rows, with the first row containing the values 8'h00, 0, and three empty cells.

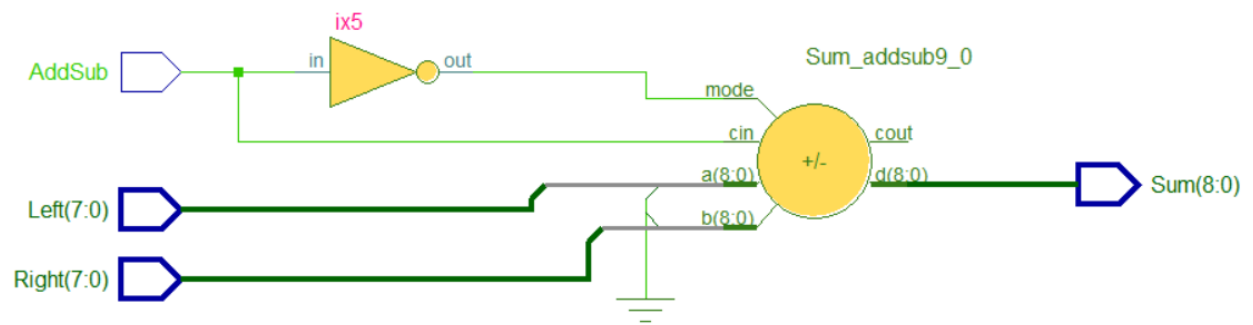
	Msgs
/normalshiftcount/A	8'h00
/normalshiftcount/S	0
/normalshiftcount/NoOnes	1
/normalshiftcount/Shifts	32'h3
/normalshiftcount/width	32'h8

The input variable is set to 0, which means that there are no '1's. As a result, S outputs 0, and the NoOnes variable asserts '1'.

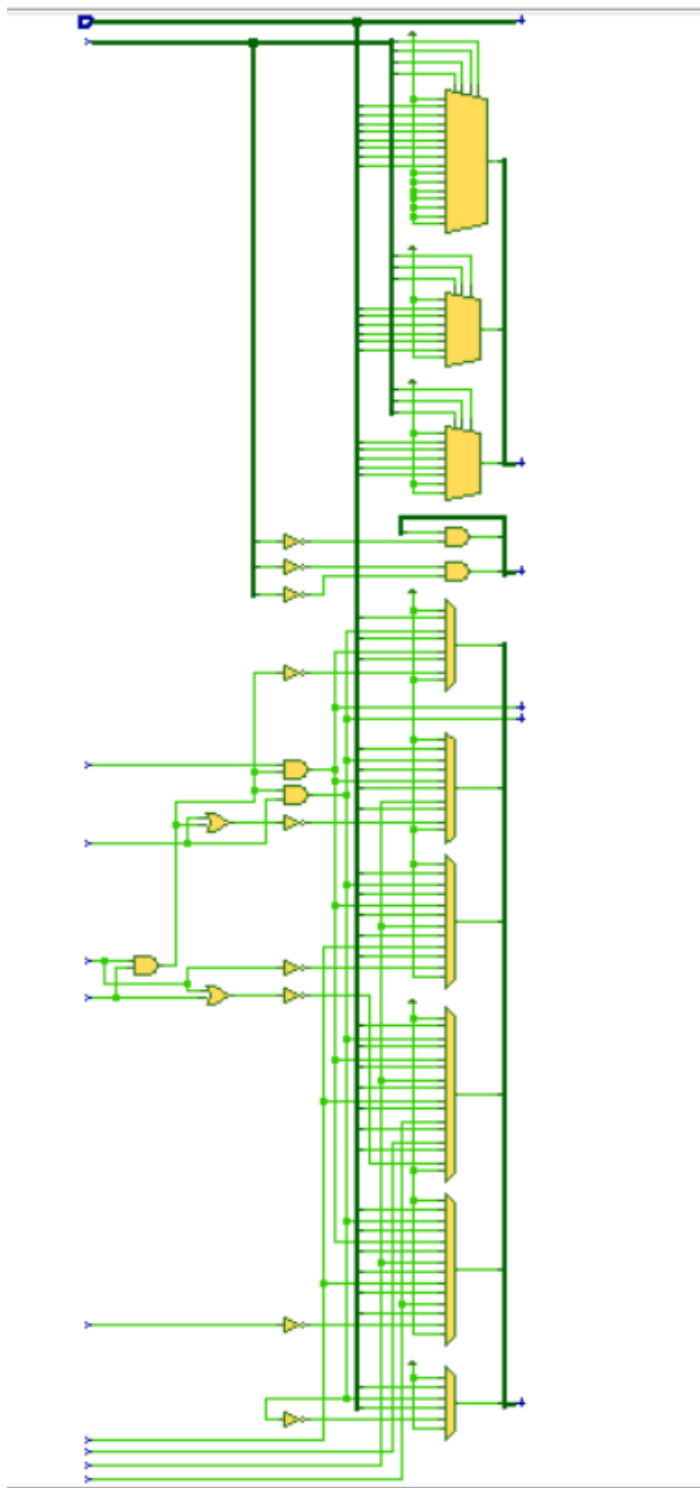
Schematics:



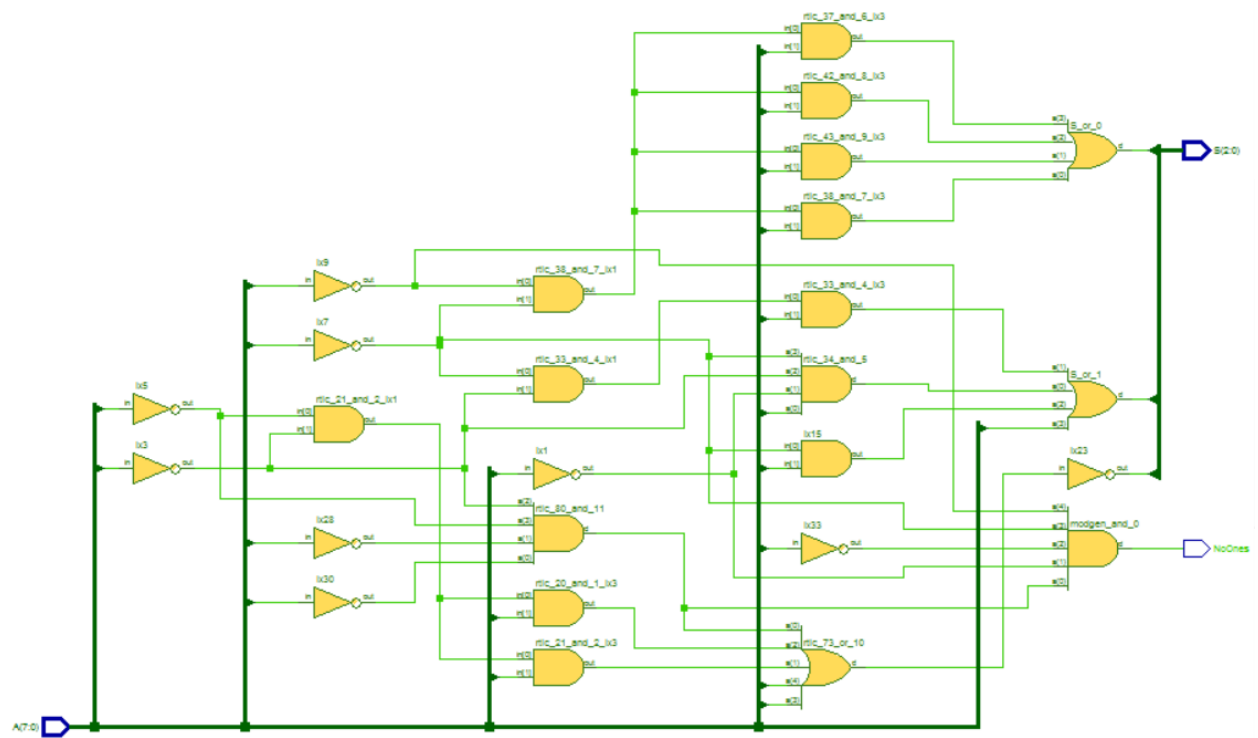
RTL view of finding the larger of two floating point numbers



RTL view of the fixed-point adder/subtractor



RTL view of the variable-bit shifter



RTL view of finding the most significant '1'