```vhdl
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity orTB is
end entity;

architecture behavior of orTB is
  constant TIME_DELAY : time := 20 ns;
  constant NUM_VALS : integer := 4;


  type A_array is array(0 to (NUM_VALS - 1)) of std_logic_vector(15 downto 0);
  type B_array is array(0 to (NUM_VALS - 1)) of std_logic_vector(15 downto 0);
  type C_array is array(0 to (NUM_VALS - 1)) of std_logic_vector(15 downto 0);
  type mode_array is array(0 to (NUM_VALS - 1)) of std_logic_vector(2 downto 0);
  type Zero_array is array(0 to (NUM_VALS - 1)) of std_logic;
  type OE_array is array(0 to (NUM_VALS - 1)) of std_logic;
  type Cout_array is array(0 to (NUM_VALS - 1)) of std_logic;

  -- positive and negative
  -- negative and positive
  -- two full zeros
  -- result is odd

  constant A_vals : A_array := (B"0000_0000_0001_0000",
                                B"1111_1111_1111_1111",
                                B"0000_0000_0000_0000",
                                B"0000_0000_0000_0010");


  constant B_vals : B_array := (B"1111_1111_1111_0100",
                                B"0000_0000_0000_1000",
                                B"0000_0000_0000_0000",
                                B"0000_0000_0000_0001");



  constant mode_vals : mode_array := (B"101",
                                      B"101",
                                      B"101",
                                      B"101");

  constant Zero_vals : Zero_array := ('0','0','1','0');

  constant OE_vals : OE_array := ('1','1','1','1');

  constant Cout_vals : Cout_array := ('0','1','0','1');

  constant C_vals : C_array := (B"1111_1111_1111_0100",
                                B"1111_1111_1111_1111",
                                B"0000_0000_0000_0000",
                                B"0000_0000_0000_0011");



  signal A_sig : std_logic_vector(15 downto 0);
  signal B_sig : std_logic_vector(15 downto 0);
  signal C_sig : std_logic_vector(15 downto 0);
  signal mode_sig : std_logic_vector(2 downto 0);
  signal Zero_sig : std_logic;
  signal OE_sig : std_logic;
  signal Cout_sig : std_logic;

begin

  DUT : entity work.ALU(behavioral)
    port map(A => A_sig,
             B => B_sig,
```

```
70                C => C_sig,
71                Mode => mode_sig,
72                Zero => Zero_sig,
73                OE => OE_sig,
74                Cout => Cout_sig);
75
76    stimulus : process
77    begin
78      for i in 0 to (NUM_VALS - 1) loop
79        A_sig <= A_vals(i);
80        B_sig <= B_vals(i);
81        --C_sig <= C_vals(i);
82        mode_sig <= mode_vals(i);
83        OE_sig <= OE_vals(i);
84        wait for TIME_DELAY;
85      end loop;
86      wait;
87    end process stimulus;
88
89    monitor : process
90      variable i : integer := 0;
91    begin
92      wait for TIME_DELAY/4;
93      while (i < NUM_VALS) loop
94        assert C_sig = C_vals(i)
95          report "C value is incorrect."
96          severity error;
97
98        assert Zero_sig = Zero_vals(i)
99          report "Zero value is incorrect."
100         severity error;
101
102       wait for TIME_DELAY/2;
103
104       assert Cout_sig = Cout_vals(i)
105         report "Cout value is incorrect."
106         severity error;
107
108       i := i + 1;
109       wait for TIME_DELAY/2;
110     end loop;
111     wait;
112   end process monitor;
113
114 end behavior;
115
```