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

entity oeTB is
end entity;

architecture behavior of oeTB is
  constant TIME_DELAY : time := 20 ns;
  constant NUM_VALS : integer := 2;


  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
  -- one that results in a zero
  -- result is odd

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


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



  constant mode_vals : mode_array := (B"000",
                                      B"000");

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

  constant OE_vals : OE_array := ('1','0');

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

  constant C_vals : C_array := (B"0000_0000_0000_0001",
                                "ZZZZZZZZZZZZZZZZ");



  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,
             C => C_sig,
             Mode => mode_sig,
             Zero => Zero_sig,
             OE => OE_sig,
             Cout => Cout_sig);

  stimulus : process
```

```
70      begin
71        for i in 0 to (NUM_VALS - 1) loop
72          A_sig <= A_vals(i);
73          B_sig <= B_vals(i);
74          --C_sig <= C_vals(i);
75          mode_sig <= mode_vals(i);
76          OE_sig <= OE_vals(i);
77          wait for TIME_DELAY;
78        end loop;
79
80  --      C_sig <= B"0000_0000_0000_0000";
81  --      wait for TIME_DELAY;
82  --      A_sig <= A_vals(0);
83  --      B_sig <= B_vals(0);
84  --      mode_sig <= mode_vals(0);
85  --      OE_sig <= '1';
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
112 --      assert C_sig = B"0000_0000_0000_0000";
113 --          report "Cout value is incorrect."
114 --          severity error;
115 --      wait for TIME_DELAY/2;
116
117 --      assert C_sig = C_vals(0);
118 --          report "Cout Value is incorrect."
119 --          severity error;
120
121       wait;
122     end process monitor;
123
124 end behavior;
125
```