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

entity adderTB is
end entity;

architecture behavior of adderTB is
  constant TIME_DELAY : time := 20 ns;
  constant NUM_VALS : integer := 6;


  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;

  -- Expected input and output data.
  -- You need to correct and add more values here based on your design
  -- TODO: test the following cases (make 10 test cases total):
  -- Full Zeros
  -- a negative and a positive
  -- a negative and a negative
  -- a positive and negative that sums to zero
  -- overflows in the negative and positive direvtion

  -- a positive and a positive

  constant A_vals : A_array := (B"0000_0000_0000_0000",
                                B"1000_1000_0000_0000",
                                B"1111_0000_0000_0000",
                                B"1111_1111_1111_1111",
                                B"1000_0000_0000_0000",
                                B"0000_0000_0000_1000");



  constant B_vals : B_array := (B"0000_0000_0000_0000",
                                B"0000_0001_0001_0000",
                                B"1000_0101_0000_0000",
                                B"0000_0000_0000_0001",
                                B"1000_0000_0000_0001",
                                B"0000_0000_0000_0010");



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

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

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

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

  constant C_vals : C_array := (B"0000_0000_0000_0000",
                                B"1000_1001_0001_0000",
                                B"1111_0101_0000_0000",
                                B"0000_0000_0000_0000",
                                B"0000_0000_0000_0001",
                                B"0000_0000_0000_1010");

```

```
70
71      signal A_sig : std_logic_vector(15 downto 0);
72      signal B_sig : std_logic_vector(15 downto 0);
73      signal C_sig : std_logic_vector(15 downto 0);
74      signal mode_sig : std_logic_vector(2 downto 0);
75      signal Zero_sig : std_logic;
76      signal OE_sig : std_logic;
77      signal Cout_sig : std_logic;
78
79    begin
80
81      DUT : entity work.ALU(behavioral)
82        port map(A => A_sig,
83                 B => B_sig,
84                 C => C_sig,
85                 Mode => mode_sig,
86                 Zero => Zero_sig,
87                 OE => OE_sig,
88                 Cout => Cout_sig);
89
90      stimulus : process
91      begin
92        for i in 0 to (NUM_VALS - 1) loop
93          A_sig <= A_vals(i);
94          B_sig <= B_vals(i);
95          --C_sig <= C_vals(i);
96          mode_sig <= mode_vals(i);
97          OE_sig <= OE_vals(i);
98          wait for TIME_DELAY;
99        end loop;
100       wait;
101     end process stimulus;
102
103     monitor : process
104       variable i : integer := 0;
105     begin
106       wait for TIME_DELAY/4;
107       while (i < NUM_VALS) loop
108         assert C_sig = C_vals(i)
109           report "C value is incorrect."
110           severity error;
111
112         assert Zero_sig = Zero_vals(i)
113           report "Zero value is incorrect."
114           severity error;
115
116         wait for TIME_DELAY/2;
117
118         assert Cout_sig = Cout_vals(i)
119           report "Cout value is incorrect."
120           severity error;
121
122         i := i + 1;
123         wait for TIME_DELAY/2;
124       end loop;
125       wait;
126     end process monitor;
127
128   end behavior;
129
```