File: C:/my_designs/ALU/ALU/src/R4_0_tb.vhd

```
1    ------------------------------------------------------------------------
     -----
2    --
3    -- Title       : R4_0_tb
4    -- Design      : ALU
5    -- Author      : Aaron Lin and Hang Chen
6    -- Company     : Stony Brook University
7    --
8    ------------------------------------------------------------------------
     -----
9    --
10   -- File        : C:\my_designs\ALU\ALU\src\R4_0_tb.vhd
11   -- Generated   : Tue Apr 21 12:52:38 2020
12   -- From        : interface description file
13   -- By          : Itf2Vhdl ver. 1.22
14   --
15   ------------------------------------------------------------------------
     -----
16   --
17   -- Description : Testbench for the R4 000 Instruction. Checks if overflows
     and underflows are saturated properly, and checks if the normal case
18   --      is calculated correctly. Each of the 4 32 bit fields are tested.
19   --
20   ------------------------------------------------------------------------
     -----
21   library work;
22   use work.all;
23   library IEEE;
24   use IEEE.std_logic_1164.all;
25   use IEEE.NUMERIC_STD.all;
26
27   entity R4_0_tb is
28   end R4_0_tb;
29
30   --}} End of automatically maintained section
31
32   architecture R4_0_tb of R4_0_tb is
33   signal ALUop : std_logic_vector(19 downto 0);
34   signal Res, regA_Data, regB_Data, regC_Data : std_logic_vector(127 downto
     0);
35   constant period : time := 20ns;
36   begin
37
38       uut : entity ALU
39           port map (ALUop => ALUop, Res => Res, regA_Data => regA_Data,
     regB_Data => regB_Data, regC_Data => regC_Data);
40
41       tb : process
42       begin
43           wait for period/2;
44
45           ALUop <= "10000XXXXXXXXXXXXXXX";
46
47           --Left most 32 bits
48           --Testing (-1 * -1) + max positive
49           regB_Data <= (31 downto 0 => '1', others => '0');
50           regC_Data <= (31 downto 0 => '1', others => '0');
```

```
51            regA_Data <= (30 downto 0 => '1', others => '0');
52            wait for period;
53
54            --Testing (1 * -1) + max negative
55            regB_Data <= (31 downto 0 => '1', others => '0');
56            regC_Data <= (0 => '1', others => '0');
57            regA_Data <= (31 => '1', others => '0');
58            wait for period;
59
60            --Testing (3 * 3) + 0
61            regB_Data <= (1 downto 0 => '1', others => '0');
62            regC_Data <= (1 downto 0 => '1', others => '0');
63            regA_Data <= (others => '0');
64            wait for period;
65
66            --Testing (-3 * 3) + 0
67            regB_Data <= (1 downto 0 => '1', others => '0');
68            regC_Data <= (31 downto 2 => '1', 1 => '0', 0 => '1', others =>
    '0');
69            regA_Data <= (others => '0');
70            wait for period;
71
72            --Next 32 bits
73            --Testing (-1 * -1) + max positive
74            regB_Data <= (63 downto 32 => '1', others => '0');
75            regC_Data <= (63 downto 32 => '1', others => '0');
76            regA_Data <= (62 downto 32 => '1', others => '0');
77            wait for period;
78
79            --Testing (1 * -1) + max negative
80            regB_Data <= (63 downto 32 => '1', others => '0');
81            regC_Data <= (32 => '1', others => '0');
82            regA_Data <= (63 => '1', others => '0');
83            wait for period;
84
85            --Testing (3 * 3) + 0
86            regB_Data <= (33 downto 32 => '1', others => '0');
87            regC_Data <= (33 downto 32 => '1', others => '0');
88            regA_Data <= (others => '0');
89            wait for period;
90
91            --Testing (-3 * 3) + 0
92            regB_Data <= (33 downto 32 => '1', others => '0');
93            regC_Data <= (63 downto 34 => '1', 33 => '0', 32 => '1', others =>
    '0');
94            regA_Data <= (others => '0');
95            wait for period;
96
97            --Next 32 bits
98            --Testing (-1 * -1) + max positive
99            regB_Data <= (95 downto 64 => '1', others => '0');
100           regC_Data <= (95 downto 64 => '1', others => '0');
101           regA_Data <= (94 downto 64 => '1', others => '0');
102           wait for period;
103
104           --Testing (1 * -1) + max negative
105           regB_Data <= (95 downto 64 => '1', others => '0');
```

```
106          regC_Data <= (64 => '1', others => '0');
107          regA_Data <= (95 => '1', others => '0');
108          wait for period;
109
110          --Testing (3 * 3) + 0
111          regB_Data <= (65 downto 64 => '1', others => '0');
112          regC_Data <= (65 downto 64 => '1', others => '0');
113          regA_Data <= (others => '0');
114          wait for period;
115
116          --Testing (-3 * 3) + 0
117          regB_Data <= (65 downto 64 => '1', others => '0');
118          regC_Data <= (95 downto 66 => '1', 65 => '0', 64 => '1', others =>
     '0');
119          regA_Data <= (others => '0');
120          wait for period;
121
122          --Right most 32 bits
123          --Testing (-1 * -1) + max positive
124          regB_Data <= (127 downto 96 => '1', others => '0');
125          regC_Data <= (127 downto 96 => '1', others => '0');
126          regA_Data <= (126 downto 96 => '1', others => '0');
127          wait for period;
128
129          --Testing (1 * -1) + max negative
130          regB_Data <= (127 downto 96 => '1', others => '0');
131          regC_Data <= (96 => '1', others => '0');
132          regA_Data <= (127 => '1', others => '0');
133          wait for period;
134
135          --Testing (3 * 3) + 0
136          regB_Data <= (97 downto 96 => '1', others => '0');
137          regC_Data <= (97 downto 96 => '1', others => '0');
138          regA_Data <= (others => '0');
139          wait for period;
140
141          --Testing (-3 * 3) + 0
142          regB_Data <= (97 downto 96 => '1', others => '0');
143          regC_Data <= (127 downto 98 => '1', 97 => '0', 96 => '1', others
     => '0');
144          regA_Data <= (others => '0');
145          wait for period;
146
147          wait;
148      end process;
149
150  end R4_0_tb;
151
```