File: C:/my_designs/ID_Stage/ID_Stage/src/ID_Stage_tb.vhd

```
1    ------------------------------------------------------------------------
     -----
2    --
3    -- Title       : ID_Stage_tb
4    -- Design      : ID_Stage
5    -- Author      : Aaron Lin and Hang Chen
6    -- Company     : Stony Brook University
7    --
8    ------------------------------------------------------------------------
     -----
9    --
10   -- File        : c:\my_designs\ID_Stage\ID_Stage\src\ID_Stage_tb.vhd
11   -- Generated   : Sun May  3 13:29:31 2020
12   -- From        : interface description file
13   -- By          : Itf2Vhdl ver. 1.22
14   --
15   ------------------------------------------------------------------------
     -----
16   --
17   -- Description : Testbench for the ID stage. Tests if registers are
     written to when I_W_EN is low, if registers are written to when
18   --  I_W_EN is high, repeated again to see if multiple writes work, and if
     reads on the same register at the same time work.
19   --  Also, checks if the I_W_EN signal is asserted for the correct set of
     instructions.
20   --
21   ------------------------------------------------------------------------
     -----
22   library work;
23   use work.all;
24   library IEEE;
25   use IEEE.std_logic_1164.all;
26   use IEEE.NUMERIC_STD.all;
27
28   entity ID_Stage_tb is
29   end ID_Stage_tb;
30
31   --}} End of automatically maintained section
32
33   architecture ID_Stage_tb of ID_Stage_tb is
34   signal CLK, I_W_EN, O_W_EN : std_logic;
35   signal I_DataIn, O_regA_Data, O_regB_Data, O_regC_Data : std_logic_vector(
     127 downto 0);
36   signal O_regA, O_regB, O_regC, O_regD, I_regD : std_logic_vector(4 downto
     0);
37   signal O_ALUop : std_logic_vector(19 downto 0);
38   signal Instruction : std_logic_vector(24 downto 0);
39   constant period : time := 20ns;
40   begin
41
42        uut : entity ID_Stage
43           port map (CLK => CLK, I_W_EN => I_W_EN, O_W_EN => O_W_EN, O_ALUop
     => O_ALUop, O_regD => O_regD, I_regD => I_regD,
44           I_DataIn => I_DataIn, Instruction => Instruction, O_regA => O_regA
     , O_regB => O_regB, O_regC => O_regC,
45           O_regA_Data => O_regA_Data, O_regB_Data => O_regB_Data,
     O_regC_Data => O_regC_Data);
```

```
46
47
48          clk_process : process
49          begin
50              CLK <= '0';
51              wait for period/2;
52              CLK <= '1';
53              wait for period/2;
54          end process;
55
56          tb : process
57          variable I_regA : std_logic_vector(4 downto 0);
58          variable I_regB : std_logic_vector(4 downto 0);
59          variable I_regC : std_logic_vector(4 downto 0);
60          variable regD : std_logic_vector(4 downto 0);
61          begin
62              wait for period/2;
63
64              --Checks if registers are written to even though I_W_EN is low
65              I_W_EN <= '0';
66              I_regA := std_logic_vector(to_unsigned(0, I_regA'length));
67              I_regB := std_logic_vector(to_unsigned(1, I_regB'length));
68              I_regC := std_logic_vector(to_unsigned(2, I_regC'length));
69              I_regD <= std_logic_vector(to_unsigned(0, I_regD'length));
70              Instruction(24 downto 20) <= "0XXXX";
71              Instruction(19 downto 15) <= I_regC;
72              Instruction(14 downto 10) <= I_regB;
73              Instruction(9 downto 5) <= I_regA;
74              Instruction(4 downto 0) <= (others => '1');
75              I_DataIn <= std_logic_vector(to_unsigned(100, I_DataIn'length));
76              wait for period;
77
78              I_regD <= std_logic_vector(to_unsigned(1, I_regD'length));
79              Instruction(4 downto 0) <= (4 => '1', others => '0');
80              wait for period;
81
82              I_regD <= std_logic_vector(to_unsigned(2, I_regD'length));
83              Instruction(4 downto 0) <= (3 => '1', others => '0');
84              wait for period;
85
86              --Checks if registerA is written to when I_W_EN is high
87              I_W_EN <= '1';
88              I_regD <= std_logic_vector(to_unsigned(0, I_regD'length));
89              Instruction(4 downto 0) <= (2 => '1', others => '0');
90              wait for period;
91
92              --Checks if registerB is written to when I_W_EN is high
93              I_regD <= std_logic_vector(to_unsigned(1, I_regD'length));
94              Instruction(4 downto 0) <= (1 => '1', others => '0');
95              wait for period;
96
97              --Checks if registerC is written to when I_W_EN is high
98              I_regD <= std_logic_vector(to_unsigned(2, I_regD'length));
99              Instruction(4 downto 0) <= (others => '0');
100             wait for period;
101
102             --Checks if registers are written again even when I_W_EN is low
```

```
103           I_W_EN <= '0';
104           I_regD <= std_logic_vector(to_unsigned(0, I_regD'length));
105           Instruction(4 downto 0) <= (4 downto 3 => '1', others => '0');
106           I_DataIn <= std_logic_vector(to_unsigned(1, I_DataIn'length));
107           wait for period;
108
109           I_regD <= std_logic_vector(to_unsigned(1, I_regD'length));
110           Instruction(4 downto 0) <= (3 downto 2 => '1', others => '0');
111           I_DataIn <= std_logic_vector(to_unsigned(2, I_DataIn'length));
112           wait for period;
113
114           I_regD <= std_logic_vector(to_unsigned(2, I_regD'length));
115           Instruction(4 downto 0) <= (2 downto 1 => '1', others => '0');
116           I_DataIn <= std_logic_vector(to_unsigned(3, I_DataIn'length));
117           wait for period;
118
119           --Checks if registers are written to again when I_W_EN is high
120           I_W_EN <= '1';
121           I_DataIn <= std_logic_vector(to_unsigned(1, I_DataIn'length));
122           wait for period;
123
124           I_regD <= std_logic_vector(to_unsigned(1, I_regD'length));
125           Instruction(4 downto 0) <= (1 downto 0 => '1', others => '0');
126           I_DataIn <= std_logic_vector(to_unsigned(3, I_DataIn'length));
127           wait for period;
128
129           I_regD <= std_logic_vector(to_unsigned(0, I_regD'length));
130           Instruction(4 downto 0) <= (2 downto 0 => '1', others => '0');
131           I_DataIn <= std_logic_vector(to_unsigned(2, I_DataIn'length));
132           wait for period;
133
134           I_regD <= std_logic_vector(to_unsigned(100, I_regD'length));
135           Instruction(4 downto 0) <= (4 => '1', 2 => '1', others => '0');
136           I_DataIn <= std_logic_vector(to_unsigned(16, I_DataIn'length));
137           wait for period;
138
139           --Read the same registers at the same time
140           I_regA := std_logic_vector(to_unsigned(100, I_regA'length));
141           I_regB := std_logic_vector(to_unsigned(100, I_regB'length));
142           I_regC := std_logic_vector(to_unsigned(100, I_regC'length));
143           Instruction(19 downto 15) <= I_regC;
144           Instruction(14 downto 10) <= I_regB;
145           Instruction(9 downto 5) <= I_regA;
146           wait for period;
147
148           --Ldi
149           Instruction(24 downto 23) <= "0X";
150           Instruction(18 downto 15) <= "XXXX";
151           wait for period;
152
153           --R4
154           Instruction(24 downto 23) <= "10";
155           Instruction(18 downto 15) <= "XXXX";
156           wait for period;
157
158           --R3
159           --BCW
```

```
160           Instruction(24 downto 23) <= "11";
161           Instruction(18 downto 15) <= "0101";
162           wait for period;
163
164           --CLZ
165           Instruction(24 downto 23) <= "11";
166           Instruction(18 downto 15) <= "0110";
167           wait for period;
168
169           --ABSDB
170           Instruction(24 downto 23) <= "11";
171           Instruction(18 downto 15) <= "0111";
172           wait for period;
173
174           --MPYU
175           Instruction(24 downto 23) <= "11";
176           Instruction(18 downto 15) <= "1000";
177           wait for period;
178
179           --MSGN
180           Instruction(24 downto 23) <= "11";
181           Instruction(18 downto 15) <= "1001";
182           wait for period;
183
184           --MPYU
185           Instruction(24 downto 23) <= "11";
186           Instruction(18 downto 15) <= "1010";
187           wait for period;
188
189           --POPCNTW
190           Instruction(24 downto 23) <= "11";
191           Instruction(18 downto 15) <= "1100";
192           wait for period;
193
194           --ROT
195           Instruction(24 downto 23) <= "11";
196           Instruction(18 downto 15) <= "1101";
197           wait for period;
198
199           --ROTW
200           Instruction(24 downto 23) <= "11";
201           Instruction(18 downto 15) <= "1110";
202           wait for period;
203
204           --SHLHI
205           Instruction(24 downto 23) <= "11";
206           Instruction(18 downto 15) <= "1111";
207           wait for period;
208
209           --Instructions with no I_W_EN
210           Instruction(24 downto 23) <= "11";
211           Instruction(18 downto 15) <= "0000";
212           wait for period;
213
214           Instruction(24 downto 23) <= "11";
215           Instruction(18 downto 15) <= "0001";
216           wait for period;
```

```
217
218          Instruction(24 downto 23) <= "11";
219          Instruction(18 downto 15) <= "0010";
220          wait for period;
221
222          Instruction(24 downto 23) <= "11";
223          Instruction(18 downto 15) <= "0011";
224          wait for period;
225
226          Instruction(24 downto 23) <= "11";
227          Instruction(18 downto 15) <= "0100";
228          wait for period;
229
230          Instruction(24 downto 23) <= "11";
231          Instruction(18 downto 15) <= "1011";
232          wait for period;
233
234          wait;
235      end process;
236
237  end ID_Stage_tb;
238
```