Test Bench in VHDL

BY
GAURAV VERMA
ASST. PROF.
ECE DEPT.
NIEC

VHDL : Test Benches

- **Test Benches**

  - We need to stimulate our designs in order to test their functionality

  - Stimulus in a real system is from an external source, not from our design

  - We need a method to test our designs that is not part of the design itself

  - This is called a "Test Bench"

  - Test Benches are VHDL entity/architectures with the following:

    - We instantiate the design to be tested using components

    - We call these instantiations "Unit Under Test" (UUT) or "Device Under Test".

    - The entity has no ports

    - We create a stimulus generator within the architecture

    - We can use reporting features to monitor the expected outputs

VHDL : Test Benches

- **Test Benches**

  - Test Benches are for Verification, not for Synthesis!!!

    - this allows us to use constructs that we ordinarily wouldn't put in a design because they are not synthesizable

- **Let's test this MUX**

```
entity  Mux_2to1 is

        port (A, B, Sel         : in        STD_LOGIC;
              Y                 : out       STD_LOGIC);

entity Mux_2to1;
```

VHDL : Test Benches

```
entity  Test_Mux is
end entity Test_Mux;                                        -- the test bench entity has no ports

architecture Test_Mux_arch of Test_Mux is

        signal      In1_TB, In2_TB      : STD_LOGIC;        -- setup internal Test Signals
        signal      Sel_TB              : STD_LOGIC;        -- give descriptive names to make
        signal      Out_TB              : STD_LOGIC;        --  apparent they are test signals

        component Mux_2to1                                  -- declare any used components
            port (A, B, Sel     : in    STD_LOGIC;
              Y                 : out   STD_LOGIC);
        end component;

        begin

          UUT : Mux_2to1                                    -- instantiate the design to test
            port map ( A    => In1_TB,
                       B    => In2_TB,
                       Sel => Sel_TB,
                       Y    => Out_TB);
```

VHDL : Test Benches

```
        STIM : process                                          -- create process to generate stimulus
          begin
            In1_TB <= '0'; In2_TB <= '0'; Sel_TB <= '0' wait for 10ns       -- we can use wait
            In1_TB <= '0'; In2_TB <= '1'; Sel_TB <= '0' wait for 10ns       -- statements to control
            In1_TB <= '1'; In2_TB <= '0'; Sel_TB <= '0' wait for 10ns       -- the speed of the stim

                                        :
                                        :
                                        :
            In1_TB <= '1'; In2_TB <= '1'; Sel_TB <= '1' wait for 10ns       -- end with a wait…

        end process STIM;


end architecture Test_Mux_2to1;
```

VHDL : Test Benches

- **Test Bench Reporting**

  - There are reporting features that allow us to monitor the output of a design

  - We can compare the output against "Golden" data and report if there are differences

  - This is powerful when we evaluate our designs across power, temp, process…..

- **Assert**

  - the keyword "assert" will check a Boolean expression

  - if the Boolean expression is FALSE, it will print a string following the "report" keyword

  - Severity levels are also reported with possible values {ERROR, WARNING, NOTE, FAILURE}

```
ex)   | A<='0'; B<='0'; wait for 10ns;
      | assert (Z='1') report "Failed test 00" severity ERROR;
```

  - The message comes out at the simulator console.

VHDL : Test Benches

- **Report**

  - the keyword "report" will always print a string

  - this is good for outputting the process of a test

  - Severity levels are also reported

ex)

```
report "Beginning the MUX test" severity NOTE;

A<='0'; B<='0'; wait for 10ns;
assert (Z='1') report "Failed test 00" severity ERROR;
```