VHDL Test Benches

And ModelSim Simulation

Announcements

- Homework #3 due Wednesday
- Exam #1 on Monday 9/23!
- Don't forget the Fram Lecture is tomorrow!
- Starfish is now open
  - Lecture and lab notifications

RIT | Division of Academic Affairs
Eugene H. Fram Chair in Applied Critical Thinking

2019 Fram Signature Lecture

"POWERFUL STUFF: An Entrepreneurial Mindset Built Upon Critical Thinking" with Doug Melton of KEEN

Date: Tuesday, September 17, 2019
Time: 3:30 pm – 4:45 pm
Place: Ingle Auditorium
(Reception immediately following in Fireside Lounge)

Access Services will be providing interpreters

Proudly cosponsored by:

RIT | College of Engineering Technology

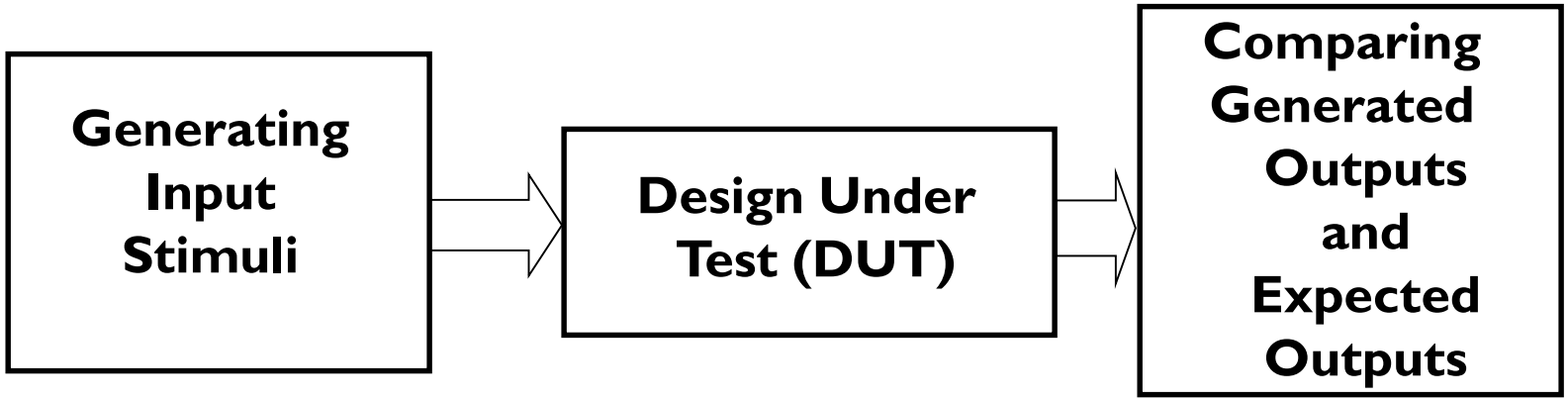
RIT | Kate Gleason College of Engineering

What is a Test Bench

- A Test Bench is VHDL code that provides the stimulus to simulate another VHDL module.
- Used in place of waveforms for toggling the inputs of the module.
- The test bench can also check the functional correctness of the module's outputs.

Main Functions of a Test Bench

- Generate stimulus for testing the hardware block.
- Apply the stimulus.
- Compare the generated outputs against the expected outputs.



Characteristics of Test Benches

- The unit/design under test, UUT or DUT, is instantiated as a component
- The entity has no ports (for now)
- The stimulus signals and their events are generated within the architecture of the test bench
- Reporting features can be used to monitor the expected outputs
  - This week we will visually inspect outputs

Example

- Testing of a 2-to-1 mux
- First define the mux

```
ENTITY  Mux_2to1 IS
   PORT (A, B, Sel  : IN      STD_LOGIC;
                Y        : OUT    STD_LOGIC);
END Mux_2to1;

ARCHITECTURE functional OF Mux_2to1 IS
BEGIN …
```

Example testbench

```
ENTITY  Mux_2to1_tb IS
END Mux_2to1_tb;                                    -- the test bench entity has no ports

ARCHITECTURE test OF Mux_2to1_tb IS
    SIGNAL          A_tb, B_tb          : STD_LOGIC;    -- setup internal Test Signals
    SIGNAL          Sel_tb              : STD_LOGIC;    -- give descriptive names to make
    SIGNAL          Y_tb                : STD_LOGIC;    -- apparent they are test signals

    COMPONENT Mux_2to1                                  -- this is the VHDL module being
      PORT (A, B, Sel          : IN      STD_LOGIC;     -- simulated. Must match entity.
            Y                  : OUT     STD_LOGIC);
      END COMPONENT;

    BEGIN

          UUT : Mux_2to1                                -- instantiate the design to test
            port map ( A    => A_tb,
                       B    => B_tb,
                       Sel  => Sel_tb,
                       Y    => Y_tb);
```

This is where you connect your VHDL component to the stimulus
Signals so that the inputs can be driven and the outputs
Can be monitored

Example

```
STIM : PROCESS                                  -- create process to generate stimulus
   BEGIN
     A_tb <= '0'; B_tb <= '0'; Sel_tb <= '0'; wait for 10ns;     -- we can use wait
     A_tb <= '0'; B_tb <= '1'; Sel_tb <= '0'; wait for 10ns ;    -- statements to control
     A_tb<= '1'; B_tb <= '0'; Sel_tb <= '0'; wait for 10ns ;     -- the speed of the stim

                                   :
                                   :
                                   :
     A_tb <= '1'; B_tb <= '1'; Sel_tb <= '1'; wait for 10ns;       -- end with a wait…
     wait;
 END PROCESS;


END test;
```

VHDL Constructs in TBs

- 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
  - Mostly time related
    - Wait
      - A <= '0'; wait for 20ns; A<= '1'; wait for 20ns; A<= '0';
    - After
      - A <= '0', '1' after 20ns, '0' after 40ns;

These are the same

Wait Statements

- A wait statement suspends and resumes execution of the process
  - When the process is suspended, the signal assignments are updated
- Not synthesizable
- Options
  - Wait on *sensitivity_list;*
    - Ex: wait on A_tb, B_tb, Sel_tb;
  - Wait until *boolean_expression;*
    - Ex: wait until interrupt = '1';
  - Wait for *time_expression;*
    - Ex: wait for 10 ns;
  - Wait;
    - Ex: wait
    - Usually the last statement in process. Why?

Stimulus Process

- This example is not efficient and would soon be unmanageable with a high number of inputs

```
A_tb <= '0'; B_tb <= '0'; Sel_tb <= '0‘; wait for 10ns;
A_tb <= '0'; B_tb <= '1'; Sel_tb <= '0‘; wait for 10ns;
A_tb<= '1'; B_tb <= '0';  Sel_tb <= '0‘; wait for 10ns;
```

- Once a signal is assigned a value, it does not have to be reassigned until the value changes
  - How would you rewrite above?

Stimulus Process

- Still too tedious for large design
- Use a Loop to do an exhaustive test of combinatorial logic
  - For n inputs the loop will go from 0 to 2^n-1
  - Why?

Using a Loop

```
ENTITY  Test_Mux IS
END Test_Mux;

ARCHITECTURE  Test_Mux_arch OF Test_Mux IS
    SIGNAL          inputs                  : STD_LOGIC_VECTOR(2 DOWNTO 0);
    SIGNAL          Y_tb                    : STD_LOGIC;

    COMPONENT Mux_2to1                                      -- declare any used components
      PORT (A, B, Sel           : IN        STD_LOGIC;     -- must match entity exactly
            Y                   : OUT       STD_LOGIC);
      END COMPONENT;

    BEGIN
      UUT : Mux_2to1                                        -- instantiate the design to test
        port map (  A   => inputs(2),
                    B   => inputs(1),
                    Sel => inputs(0),
                    Y   => Y_tb);
      STIM : PROCESS                            -- create process to generate stimulus
       BEGIN
         FOR i IN 0 TO 7 LOOP
           inputs <= STD_LOGIC_VECTOR(to_unsigned(i, 3));
          WAIT FOR 10 ns;
        END LOOP;
        wait;
       END PROCESS;
```

Converting integer to STD_LOGIC_VECTOR

- An integer can be converted to a STD_LOGIC_VECTOR of any length
- Requires numeric_std library
  - Use ieee.numeric_std.all
- Syntax
  - Vector <= STD_LOGIC_VECTOR(to_unsigned(int,n))
    - Vector is a STD_LOGIC_VECTOR of length n
    - Int is an integer in the range 0 to 2^n-1

VHDL type conversion

http://www.bitweenie.com/listings/vhdl-type-conversion/