Prof. K. J. Hintz

Department of Electrical and Computer Engineering
George Mason University
Testbench Example

- Configuration
- Clock
- Test Bench
- Behavioral to Structural
Counter Test Bench Entity

```vhdl
entity Counter_Wrapper_TB is
description of the entity (e.g., inputs, outputs, signals, etc.)
end Counter_Wrapper_TB;
```

No port clause since encapsulating counter to test it.
First Counter Architecture

architecture Count_1 of Counter_Wrapper_TB is
  signal Clock : bit := ‘0’ ;
begin
Local Clock

Clock_P_1 : process
begin
    Clock <= not Clock after 50 ns;
    wait on Clock ;
end process Clock_P_1 ;
Counter Process

Inc_Cntr: process
    variable Count_Now : integer := 0 ;
begin
    --wait for trailing edge
    wait until Clock = '0' ;
Increment Counter

--increment mod 8

if Count_Now = 7 then
    Count_now := 0;
else
    Count_now := Count_Now + 1;
end if;

end process Inc_Cntr;

end Count_1;
architecture Count_2 of Counter_Wrapper_TB is

signal Clock : bit := ‘0’;

begin

-- no change
Clock_P_2 : process ( Clock )
begin
    Clock <= not Clock after 50 ns;
    -- removed ... wait on Clock ;
end process Clock_P_2 ;
Inc_Cntr: process
variable Count_Now : integer := 0 ;
begin
--wait for trailing edge
wait until Clock = '0' ;
-- no change
Increment Counter

--increment mod 8

if Count_Now = 7 then Count_now := 0 ;
else
    Count_now := Count_Now + 1 ;
end if ;

-- no change
end process Inc_Cntr ;

end Count_2 ;

-- no change
3rd Counter

- Same As Others Except for Clock Generator
- Concurrent Clock Instead of Process
- Wrapper Slides Not Repeated Here
Concurrent Local Clock

Clock_P_3 :
-- removed process ( Clock )
-- removed begin
    Clock <= not Clock after 50 ns;
-- removed ... wait on Clock ;
-- removed end process Clock_P_2;
Specific Configuration of Counter

```
configuration Conf_KJH_1 of
Counter_Wrapper_TB is
  --no ambiguity here
  --only one entity per library
  for Count_1
  --ambiguity so need to
  --specify architecture
```
End Configuration

end for ;
end Conf_KJH_1 ;

--architecture is now bound
--to entity
Encapsulation

Conf_KJH_1

CounterWRAPPER_TB

Count_1
2nd Specific Configuration of Counter

configuration Conf_KJH_2 of Counter_Wrapper_TB is
--no ambiguity here
--only one entity per library
for Count_2
--ambiguity so need to
--specify architecture
End Configuration

end for;
end Conf_KJH_2;

--different architecture is
--now bound to same entity
--mutually exclusive bindings
Encapsulation

Conf_KJH_2

Counter_Wrapper_TB

Count_2
Structural Decomposition of CounterWrapper_TB

CounterWrapper_TB

Count_3

CounterWrapper_TB

Clock

Counter

Verify
SD Counter Clock Component

architecture Count_SD of
  Counter_Wrapper_TB is
component Clock
  generic (PW : time );
  port ( Clock : out bit );
end component Clock ;
component Counter_Mod_8
    port (   
        Clock : in bit ;
        DataOut : out bit_vector (2 downto 0 ) );
end component Counter_Mod_8 ;
component Verify
  port (  
    DataIn : in bit_vector (2 downto 0 )) ;
end component Verify ;
signal Clock_Tic : bit := ‘0’;
signal Count_Data : bit_vector
     (2 downto 0 ) := ( others => ‘0’ );
--initializes all values to ‘0’
Clock Instantiation

begin
Synch : Clock

generic map ( PW => 50 ns )
--no ; required cause port next

port map ( Clock_Tic ) ;
Counter_Instance : Counter_Mod_8

port map ( Clock_Tic , Count_Data );
Verify Instantiation

Verify_Instance : Verify

port map ( DataIn => Count_Data ) ;

end Count_SD ;
Block Diagram of SD

Count_SD
- Clock out

Counter_Mod_8
- Clock in
  - DataOut (2)
  - DataOut (1)
  - DataOut (0)

Verify
- DataIn (2)
- DataIn (1)
- DataIn (0)
Approach

- Turn `Counter_Mod_8` into an entity and put in a library after it has been verified.
- `Count_SD` is a good start to a testbench where the counter drives a signal generator which creates the sequential signals to excite the device under test.
- `Verify` is a component which needs to be expanded with assert/report statements to verify performance.
End of Lecture

- Configuration
- Test Bench
- Structural Decomposition