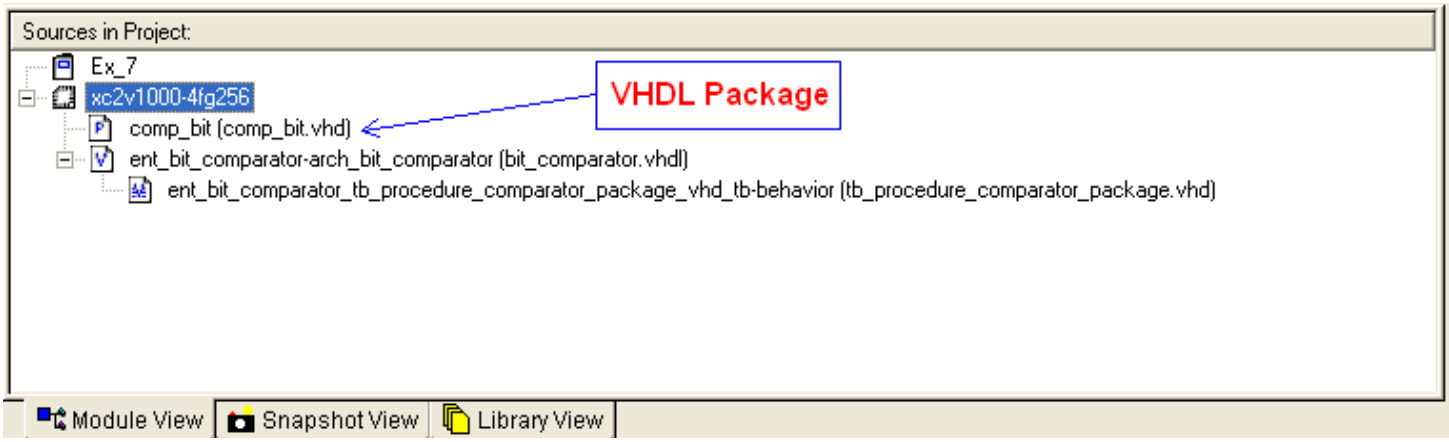
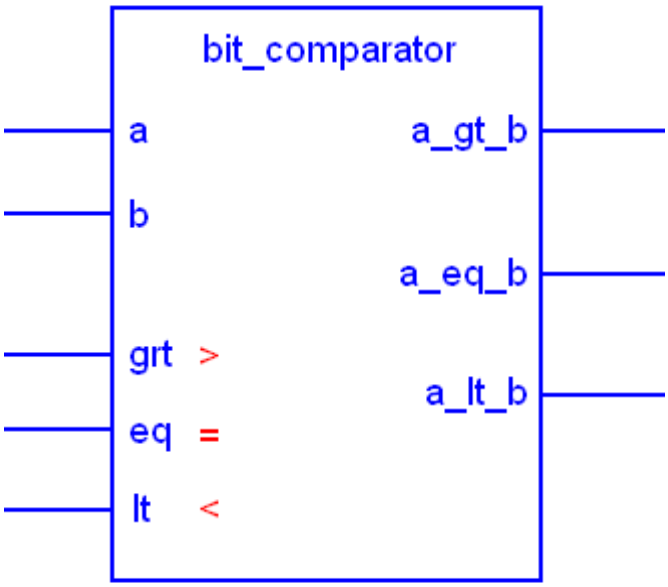
VHDL Package

A good design involves grouping certain components, some commonly used user defined types and subprograms. In **VHDL**, packages can be used for this grouping. In order for various designs to share several subprograms, a **PACKAGE** declaration must contain a declaration of the subprograms and the body of the subprograms must reside in a corresponding **PAKAGE BODY**. A package declaration defines the interface of a package that includes the declarations that are to be visible from outside the package. A package body contains declarations that are local to it as well as bodies of locally or externally used subprograms.



The PACKAGE in VHDL



A single bit comparator

VHDL PACKAGE CODE

```
================================================================================
bit_comparator.vhd
================================================================================
library IEEE;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

-- Hendra Kesuma
--------------------------------------------------------------------------------------------------------------------------
-- PACKAGE DECLARATION

package comp_bit is
  type my_vector is array  (0 to 3) of std_logic; -- Type declaration

  function fgl(x, y, gl: std_logic) RETURN std_logic;
  function feq(x, y, equ: std_logic) RETURN std_logic;

  PROCEDURE ApplyData_Procedure
          (signal   OutBit:              out   std_logic;
           constant GivenBit:            in    my_vector;
           constant GivenPeriod :        in    time);
end comp_bit;

--------------------------------------------------------------------------------------------------------------------------
-- PACKAGE BODY

package body comp_bit is
  -- Function 1
    FUNCTION fgl(x, y, gl: std_logic) RETURN std_logic IS
      BEGIN
    RETURN ((x and  gl) or ((not y) and gl) or (x and (not y)));
    END fgl;


  -- Function 2
    FUNCTION feq(x, y, equ: std_logic) RETURN std_logic IS
      BEGIN
    RETURN ((x and y and equ) or ((not x) and (not y) and equ));
    END feq;

   -- PROCEDURE for SIMULATION
    PROCEDURE ApplyData_Procedure
          (signal   OutBit:                     out   std_logic;
                    constant GivenBit:          in    my_vector;
                    constant GivenPeriod :      in    time) IS

    BEGIN
      OutBit <= transport GivenBit(0) after 0*GivenPeriod;
      OutBit <= transport GivenBit(1) after 1*GivenPeriod;
```

```
        OutBit <= transport GivenBit(2) after 2*GivenPeriod;
        OutBit <= transport GivenBit(3) after 3*GivenPeriod;

                                --   or  --
   --   for i in 0 to 3
   --        OutBit <= transport GivenBit(i) after i*GivenPeriod;
   --   end loop;

    END PROCEDURE ApplyData_Procedure;
end comp_bit;
```

VHDL CODE

==========

bit_comparator.vhdl

==========

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use work.comp_bit.ALL;  --Include the package

--Hendra Kesuma

entity ent_bit_comparator is
    Port ( a : in std_logic;
           b : in std_logic;
           grt : in std_logic;
           eq : in std_logic;
           lt : in std_logic;
           a_gt_b : out std_logic;
           a_eq_b : out std_logic;
           a_lt_b : out std_logic);
end ent_bit_comparator;

architecture arch_bit_comparator of ent_bit_comparator is
begin
process(a, b, grt, eq, lt)
  begin
    a_gt_b <= fgl(a, b, grt);
    a_eq_b <= feq(a, b, eq);
    a_lt_b <= fgl(b, a, lt);
  end process;
end arch_bit_comparator;
```

TESTBENCH CODE

tb_bit_comparator.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
use work.comp_bit.ALL;  -- Include the package
-- Hendra Kesuma

ENTITY ent_bit_comparator_tb_procedure_comparator_package_vhd_tb IS
END ent_bit_comparator_tb_procedure_comparator_package_vhd_tb;

ARCHITECTURE behavior OF ent_bit_comparator_tb_procedure_comparator_package_vhd_tb IS

	COMPONENT ent_bit_comparator
	PORT(
		a : IN std_logic;
		b : IN std_logic;
		grt : IN std_logic;
		eq : IN std_logic;
		lt : IN std_logic;
		a_gt_b : OUT std_logic;
		a_eq_b : OUT std_logic;
		a_lt_b : OUT std_logic
		);
	END COMPONENT;

	SIGNAL a :  std_logic;
	SIGNAL b :  std_logic;
	SIGNAL grt :  std_logic;
	SIGNAL eq :  std_logic;
	SIGNAL lt :  std_logic;
	SIGNAL a_gt_b :  std_logic;
	SIGNAL a_eq_b :  std_logic;
	SIGNAL a_lt_b :  std_logic;

BEGIN

	uut: ent_bit_comparator PORT MAP(
		a => a,
		b => b,
		grt => grt,
		eq => eq,
		lt => lt,
		a_gt_b => a_gt_b,
		a_eq_b => a_eq_b,
		a_lt_b => a_lt_b
	);
```

```
-- *** Test Bench - User Defined Section ***
        tb : PROCESS
             BEGIN
                grt <= '0' after 0 ns;
                eq  <= '1' after 0 ns;
                 lt  <= '0' after 0 ns;


                ApplyData_Procedure(a, "0110", 10 ns);
                ApplyData_Procedure(b, "0011", 10 ns);

                wait; -- will wait forever
            END PROCESS;
-- *** End Test Bench - User Defined Section ***
END;
```

SIMULATION