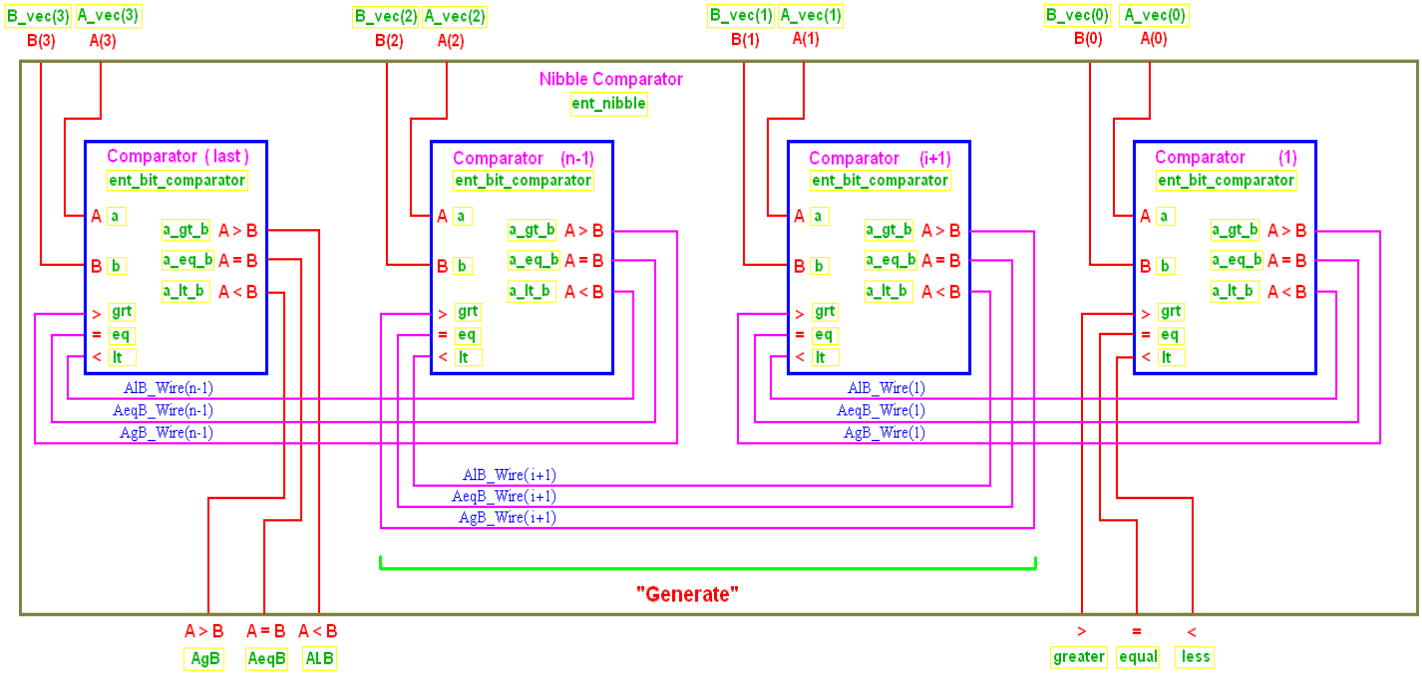
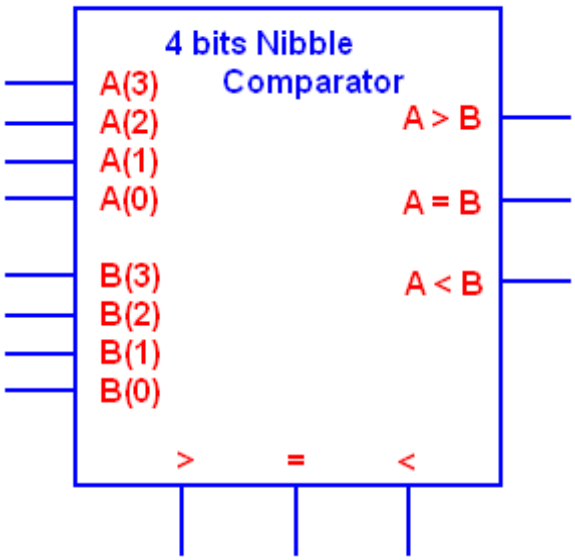
Nibble comparator

A comparator which compares two 4-bit data inputs is called **nibble comparator**. The nibble comparator can be built by cascading four **single bit comparators**. A comparator bit only uses the outputs of a less significant bit if its own data bits are equal. With this scheme, the comparator outputs of the **nibble comparator** are generated faster by not having to depend on all **single bit comparators**.



cascading four single bit comparators



4 bits nibble comparator

VHDL CODE

```
================================================================================
Nibble_Comparator.vhdl
================================================================================
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Hendra Kesuma

entity ent_nibble is
    Port ( A_vec   :    in std_logic_vector (3 downto 0);
           B_vec   :    in std_logic_vector (3 downto 0);
             AgB   :    out std_logic;
            AeqB   :    out std_logic;
              AlB   :    out std_logic;
          greater   :    in std_logic;
            equal   :    in std_logic;
        l      ess   :    in std_logic);
end ent_nibble;

architecture arch_nibble of ent_nibble 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;

constant n: integer := 4 ;
signal AgB_Wire   : std_logic_vector(1 to n-1);
signal AeqB_Wire  :std_logic_vector(1 to n-1);
signal AlB_Wire   :   std_logic_vector(1 to n-1);

begin
comparator_first: ent_bit_comparator
  port map (a       =>  A_vec(0),
            b       =>  B_vec(0),
            grt     =>  greater,
            eq      =>  equal,
             lt     => less,
          a_gt_b  => AgB_Wire(1),
         a_eq_b  => AeqB_Wire(1),
          a_lt_b  => AlB_Wire(1));
```

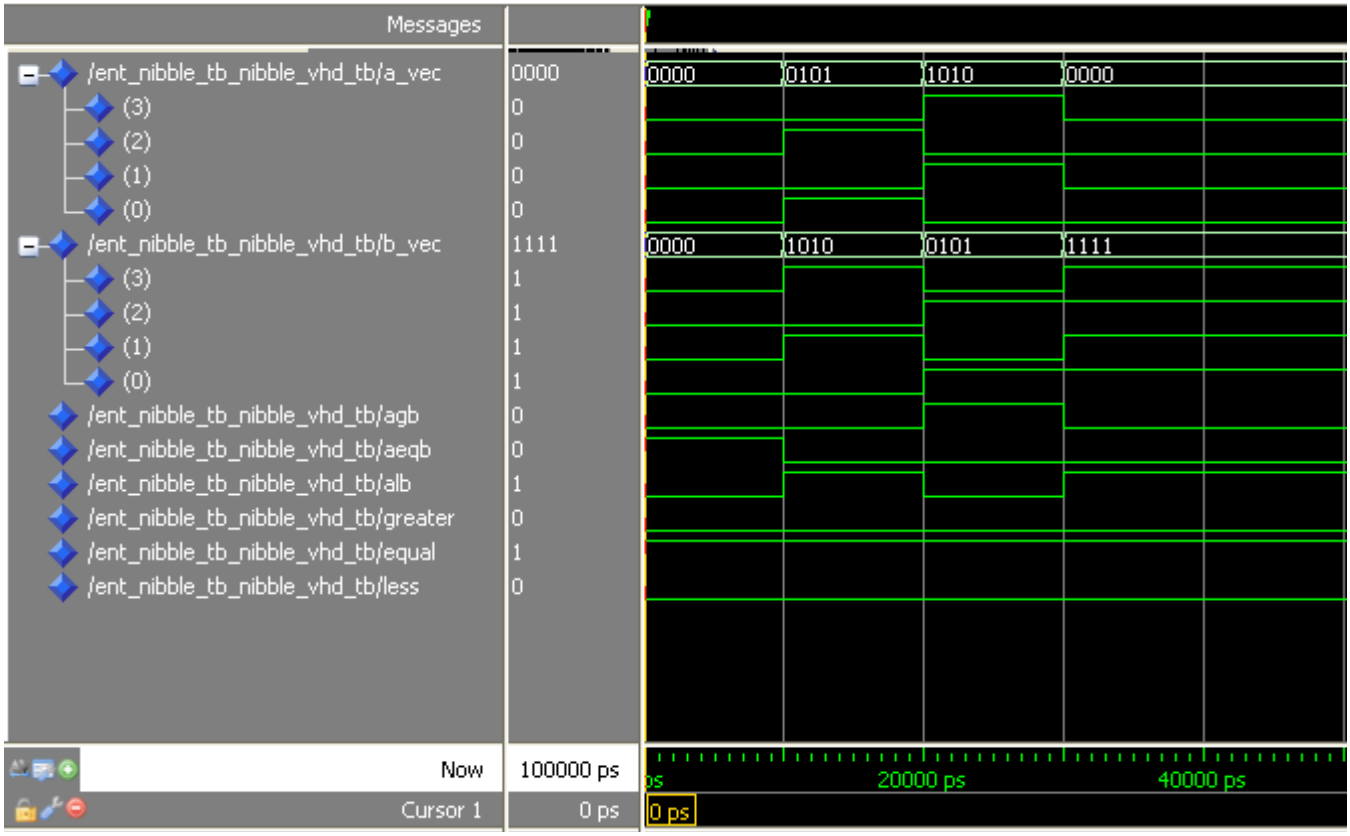
```
comparator_gen: for i in 1 to n-2 GENERATE

comparator_i: ent_bit_comparator
   port map ( a   =>  A_vec(i),
                b  =>  B_vec(i),
              grt  =>  AgB_Wire(i),
              eq   =>  AeqB_Wire(i),
               lt  =>  AlB_Wire(i),
          a_gt_b  =>  AgB_Wire(i+1),
          a_eq_b  =>  AeqB_Wire(i+1),
           a_lt_b  =>  AlB_Wire(i+1));

end GENERATE;

comparator_last: ent_bit_comparator
  port map (a    =>  A_vec(3),
               b    =>  B_vec(3),
             grt    =>  AgB_Wire(n-1),
            eq      =>  AeqB_Wire(n-1),
             lt     =>  AlB_Wire(n-1),
          a_gt_b  =>  AgB,
          a_eq_b  =>  AeqB,
           a_lt_b  =>  AlB);

end arch_nibble;
```

TESTBENCH CODE

```
================================================================================
tb_nibble.vhd
================================================================================
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
use work.comp_bit.ALL; -- Include package

-- Hendra Kesuma

ENTITY ent_nibble_tb_nibble_vhd_tb IS
END ent_nibble_tb_nibble_vhd_tb;

ARCHITECTURE behavior OF ent_nibble_tb_nibble_vhd_tb IS

        COMPONENT ent_nibble
        PORT(
                A_vec         : IN std_logic_vector(3 downto 0);
                B_vec         : IN std_logic_vector(3 downto 0);
                greater       : IN std_logic;
                equal         : IN std_logic;
                less          : IN std_logic;
```

```
                AgB             : OUT std_logic;
                AeqB            : OUT std_logic;
                AlB             : OUT std_logic
                );

        END COMPONENT;

        SIGNAL A_vec        :  std_logic_vector(3 downto 0);
        SIGNAL B_vec        :  std_logic_vector(3 downto 0);
        SIGNAL AgB          :  std_logic;
        SIGNAL AeqB         :  std_logic;
        SIGNAL AlB          :  std_logic;
        SIGNAL greater      :  std_logic;
        SIGNAL equal        :  std_logic;
        SIGNAL less         :  std_logic;

BEGIN

        uut: ent_nibble PORT MAP(
                A_vec           => A_vec,
                B_vec           => B_vec,
                AgB             => AgB,
                AeqB            => AeqB,
                AlB             => AlB,
                greater         => greater,
                equal           => equal,
                less            => less
        );

-- *** Test Bench - User Defined Section ***
   tb : PROCESS
   BEGIN
     greater    <= '0' after 0 ns;
     equal      <= '1' after 0 ns;
     less       <= '0' after 0 ns;

     ApplyData_Procedure(A_vec(0), "0100", 10 ns);
     ApplyData_Procedure(A_vec(1), "0010", 10 ns);
     ApplyData_Procedure(A_vec(2), "0100", 10 ns);
     ApplyData_Procedure(A_vec(3), "0010", 10 ns);

     ApplyData_Procedure(B_vec(0), "0011", 10 ns);
     ApplyData_Procedure(B_vec(1), "0101", 10 ns);
     ApplyData_Procedure(B_vec(2), "0011", 10 ns);
     ApplyData_Procedure(B_vec(3), "0101", 10 ns);

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

SIMULATION


Messages
/ent_nibble_tb_nibble_vhd_tb/a_vec
(3)
(2)
(1)
(0)
/ent_nibble_tb_nibble_vhd_tb/b_vec
(3)
(2)
(1)
(0)
/ent_nibble_tb_nibble_vhd_tb/agb
/ent_nibble_tb_nibble_vhd_tb/aeqb
/ent_nibble_tb_nibble_vhd_tb/alb
/ent_nibble_tb_nibble_vhd_tb/greater
/ent_nibble_tb_nibble_vhd_tb/equal
/ent_nibble_tb_nibble_vhd_tb/less
0000
0 0 0 0
1111
1 1 1 1
0 0 1 0 1
Now
100000 ps
Cursor 1
0 ps
0000
0101
1010
0000
0000
1010
0101
1111
20000 ps
40000 ps
0 ps