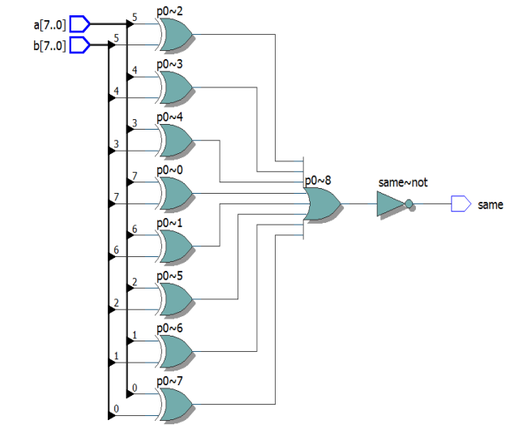
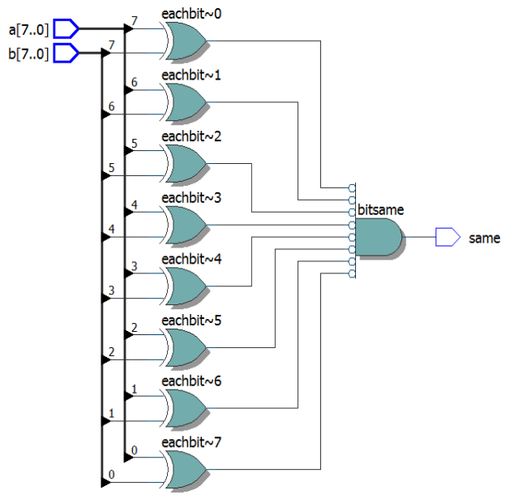
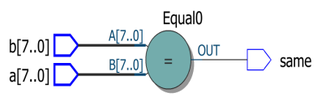
```vhdl
library ieee;
    use ieee.std_logic_1164.all;

entity equal is
    generic(n : in integer := 8);
    port(
        a, b : in std_logic_vector(n-1 downto 0);
        same : out std_logic);
end equal;

architecture rtl1 of equal is
begin
    p0 : process(a,b)
        variable same_so_far : std_logic;
begin
    same_so_far := '1';
    for i in a'range loop --with range attribute, code is
                          flexi to change width
        if(a(i) /= b(i))then
            same_so_far := '0';
            exit;
        end if;
    end loop;
    same <= same_so_far;
end process;
end rtl1;

architecture rtl2 of equal is
    signal eachbit : std_logic_vector(a'length-1 downto 0);
begin
    xnor_gen : for i in a'range generate
        eachbit(i) <= not (a(i) xor b(i));
    end generate;

    p0:process(eachbit)
        variable bitsame : std_logic;
   begin
    bitsame := eachbit(0);
    for i in 1 to a'length-1 loop
        bitsame := bitsame and eachbit(i);
    end loop;
    same <= bitsame;
end process;
end rtl2;

architecture rtl3 of equal is
begin
    same <= '1' when a = b else '0';
end rtl3;
```








```vhdl
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;

entity tbequal is
end entity;

architecture beh of tbequal is

    component equal
    generic(n : in integer := 8);
    port(
        a, b : in std_logic_vector(n-1 downto 0);
        same : out std_logic);
    end component;

constant w : integer := 5; --signal width
signal a, b : std_logic_vector(w-1 downto 0);
signal y, y1, y2, y3 : std_logic;

constant period : time := 50ns;
constant strobe : time := period - 5ns;

begin
    p0 : process
    begin
        for j in 0 to 2**w-1 loop
            a <= conv_std_logic_vector(j,w);
            for k in 0 to 2**w-1 loop
                b <= conv_std_logic_vector(k,w);
                wait for period;
            end loop;
        end loop;
        wait;
    end process;

    y <= '1' when a = b else '0';
    check : process
        variable err_cnt : integer := 0;
    begin
        wait for strobe;
        for j in 1 to 2**(w*2) loop
            if(y /= y1) or (y /= y2) or (y /= y3) then
                assert false report "not compare" severity warning;
                err_cnt := err_cnt +1;
            end if;
            wait for period;
        end loop;
        assert (err_cnt = 0) report "test failed" severity error;
        assert (err_cnt = 0) report "test passed" severity note;
        wait;
    end process;

    equal1 : equal
        generic map(n => w)
        port map(a => a, b => b, same => y1);
    equal2 : equal
        generic map(n => w)
        port map(a => a, b => b, same => y2);
    equal3 : equal
        generic map(n => w)
        port map(a => a, b => b, same => y3);
end beh;

configuration cfg_tbeqal of tbequal is
    for beh
       for equal1 : equal
             use entity work.equal(rtl1);
       end for;
       for equal2 : equal
             use entity work.equal(rtl2);
       end for;
       for equal3 : equal
             use entity work.equal(rtl3);
```

```
            end for;
        end for;
    end cfg_tbeqal;
```