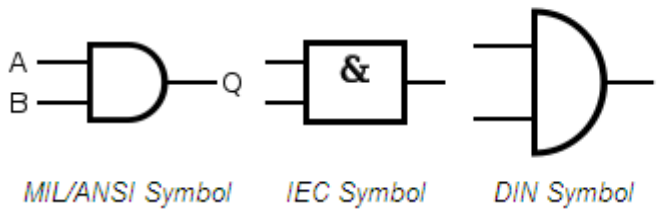
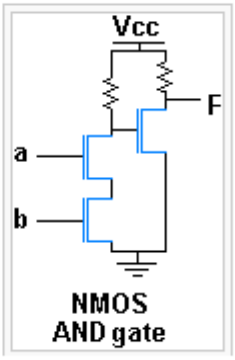
AND GATE





INPUT		OUTPUT
A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

VHDL CODES

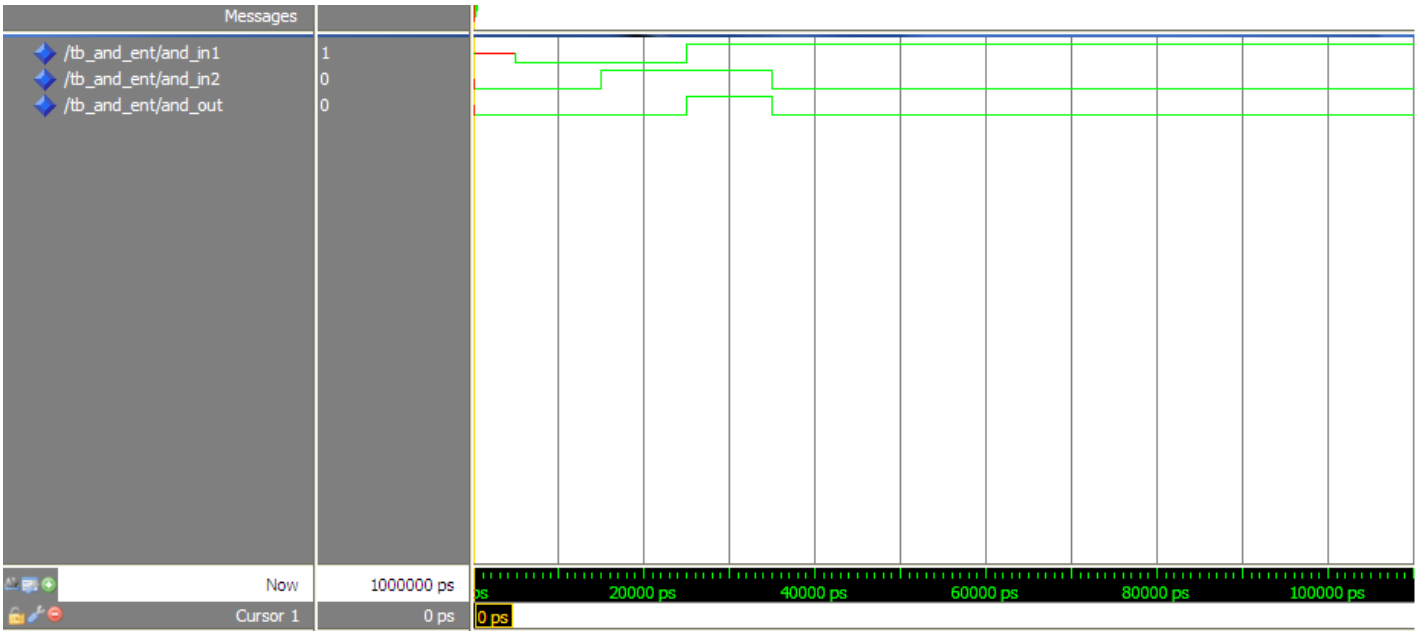
```
================================================================
and_Gate.vhdl
================================================================
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Hendra Kesuma

entity and_ent is
    Port ( and_in1 : in std_logic;
           and_in2 : in std_logic;
           and_out : out std_logic);
end and_ent;

architecture and_arch of and_ent is
begin
  process(and_in1,and_in2)
    begin
    if((and_in1='1') and (and_in2='1')) then
       and_out<='1';
    else
       and_out<='0';
    end if;
  end process;
end and_arch;
```

TESTBENCH CODE

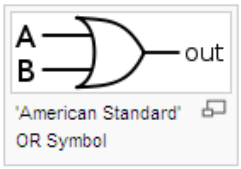
```
================================================================
tb_and_ent.vhd
================================================================
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

-- Hendra Kesuma
```

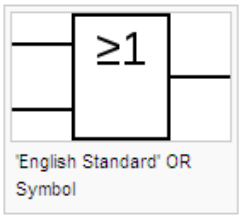
```
ENTITY tb_and_ent IS
END tb_and_ent;

ARCHITECTURE tb_and_arch OF tb_and_ent IS
        COMPONENT and_ent
        PORT(
                and_in1 : IN std_logic;
                and_in2 : IN std_logic;
                and_out : OUT std_logic
                );
        END COMPONENT;
    FOR uut:and_ent use entity work.and_ent(and_arch);
                SIGNAL and_in1 :  std_logic;
            SIGNAL and_in2 :  std_logic;
            SIGNAL and_out :  std_logic;
BEGIN
    and_in1<='0' after 5 ns, '1' after 25 ns;
    and_in2<='0' after 0 ns, '1' after 15 ns, '0' after 35 ns;
        uut:and_ent port map(and_in1, and_in2, and_out);
end tb_and_arch;
```

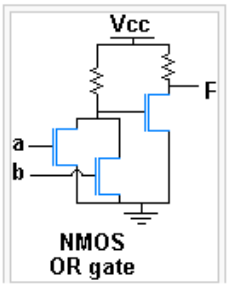
SIMULATION

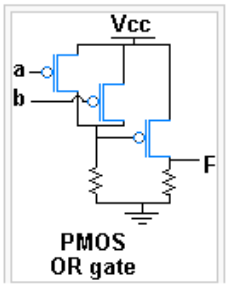


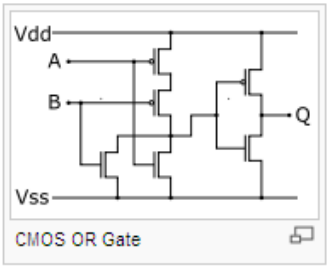
OR GATE











INPUT		OUTPUT
A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

VHDL CODE

==========

or_Gate.vhdl

==========

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Hendra Kesuma

entity or_ent is
    Port ( or_in1 : in std_logic;
           or_in2 : in std_logic;
           or_out : out std_logic);
end or_ent;

architecture or_arch of or_ent is
begin
  process(or_in1,or_in2)
    begin
    if((or_in1='0') and (or_in2='0')) then
       or_out<='0';
    else
       or_out<='1';
    end if;
  end process;
end or_arch;
```

TESTBENCH CODE

==========

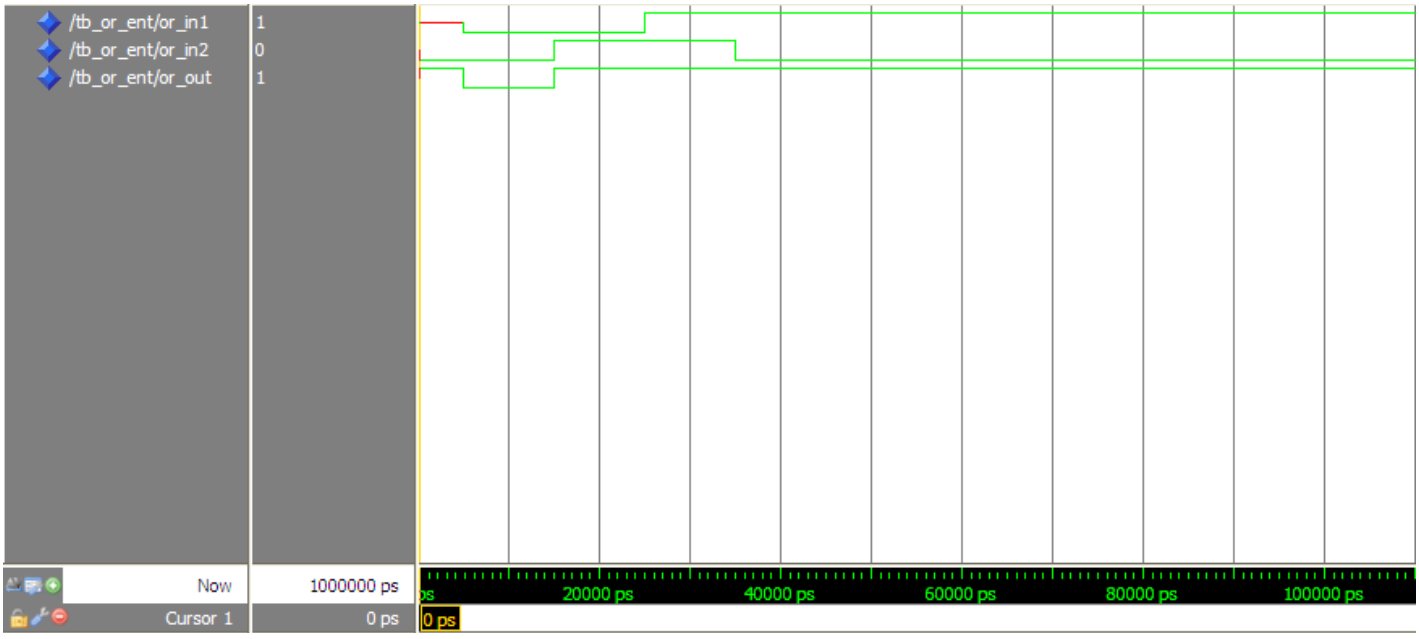
tb_or_ent.vhd

==========

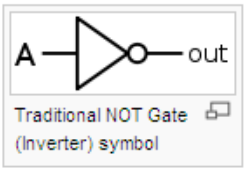
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

-- Hendra Kesuma
```

```
ENTITY tb_or_ent IS
END tb_or_ent;
ARCHITECTURE tb_or_arch OF tb_or_ent IS
        COMPONENT or_ent
        PORT(
                or_in1 : IN std_logic;
                or_in2 : IN std_logic;
                or_out : OUT std_logic
                );
        END COMPONENT;
    FOR uut:or_ent use entity work.or_ent(or_arch);
                SIGNAL or_in1 :  std_logic;
            SIGNAL or_in2 :  std_logic;
            SIGNAL or_out :  std_logic;
BEGIN
    or_in1<='0' after 5 ns, '1' after 25 ns;
    or_in2<='0' after 0 ns, '1' after 15 ns, '0' after 35 ns;
        uut:or_ent port map(or_in1, or_in2, or_out);
end tb_or_arch;
```

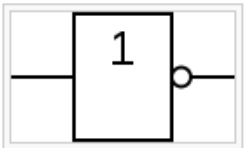
SIMULATION



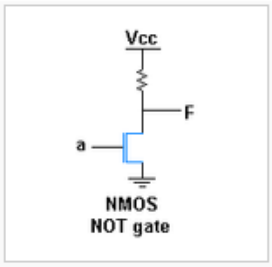
NOT GATE



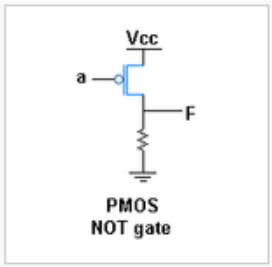
Traditional NOT Gate (Inverter) symbol



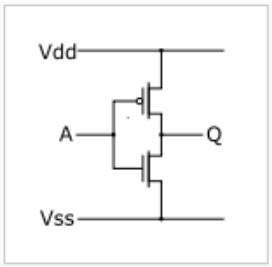
International Electrotechnical Commission NOT Gate



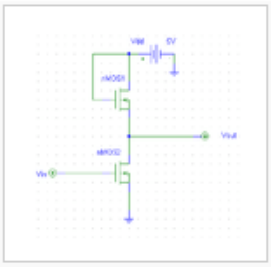
NMOS Inverter



PMOS Inverter



Static CMOS Inverter


Schematic of a *Saturated-Load* Digital Inverter

INPUT A	OUTPUT NOT A
0	1
1	0

VHDL CODE

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

-- Hendra Kesuma

entity not_ent is
    Port ( not_in : in std_logic;
           not_out : out std_logic);
end not_ent;

architecture not_arch of not_ent is
begin
  process(not_in)
    begin
    if(not_in='1') then
       not_out<='0';
    else
       not_out<='1';
    end if;
  end process;
end not_arch;
```

TESTBENCH CODE

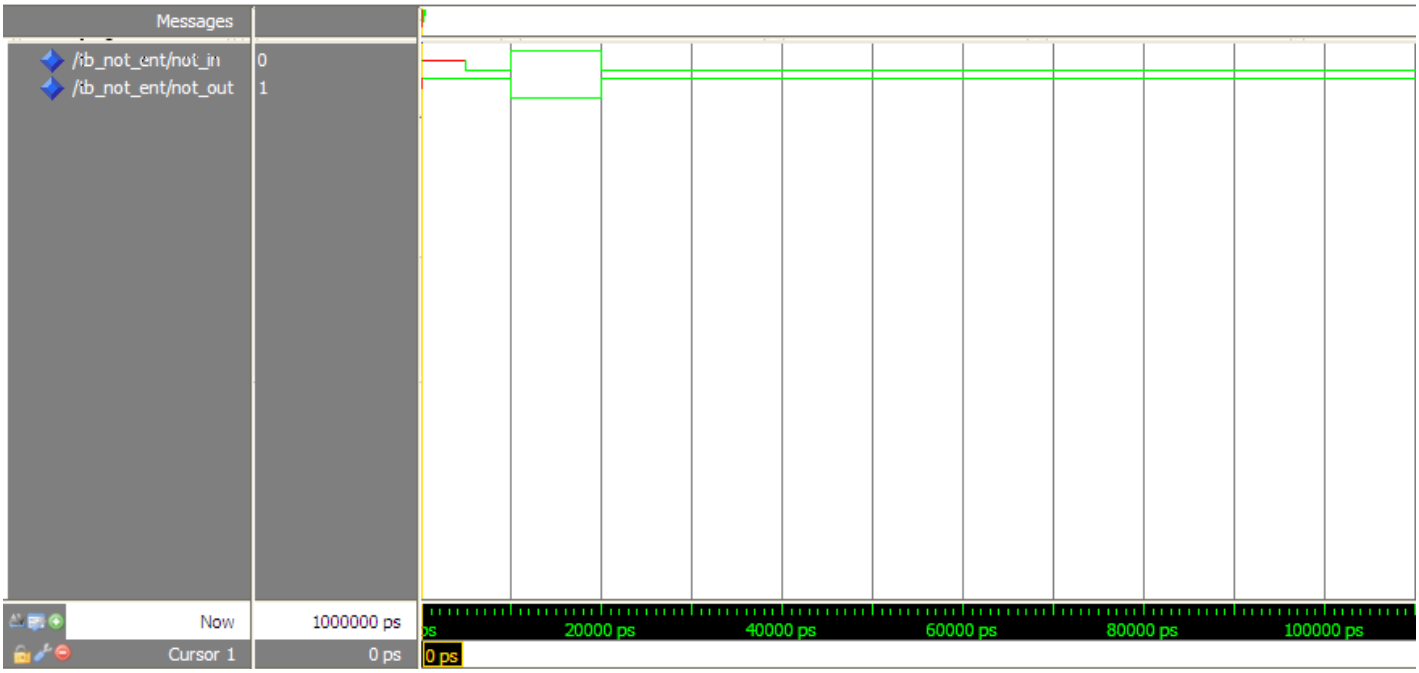
==========

tb_not_ent.vhd

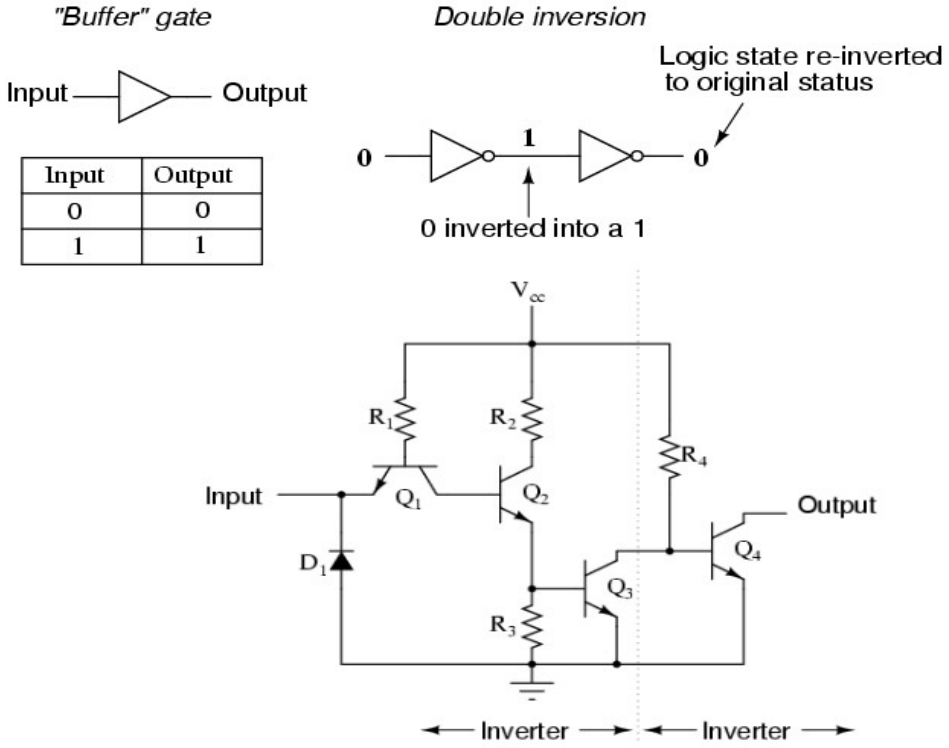
==========

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

-- Hendra Kesuma

ENTITY tb_not_ent IS
END tb_not_ent;

ARCHITECTURE tb_not_arch OF tb_not_ent IS
        COMPONENT not_ent
        PORT(
                not_in : IN std_logic;
                not_out : OUT std_logic
                );
        END COMPONENT;
    FOR uut:not_ent use entity work.not_ent(not_arch);
                SIGNAL not_in :  std_logic;
            SIGNAL not_out :  std_logic;
BEGIN
    not_in<='0' after 5 ns, '1' after 10 ns, '0' after 20 ns;
    uut:not_ent port map(not_in, not_out);
end tb_not_arch;
```

SIMULATION



Driver(BUFFER)



VHDL CODE

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

-- Hendra Kesuma

entity driver_ent is
    Port ( driver_in : in std_logic;
           driver_out : out std_logic);
end driver_ent;

architecture driver_arch of driver_ent is
begin
     driver_out <= driver_in after 5 ns;
end driver_arch;
```

TESTBENCH CODE

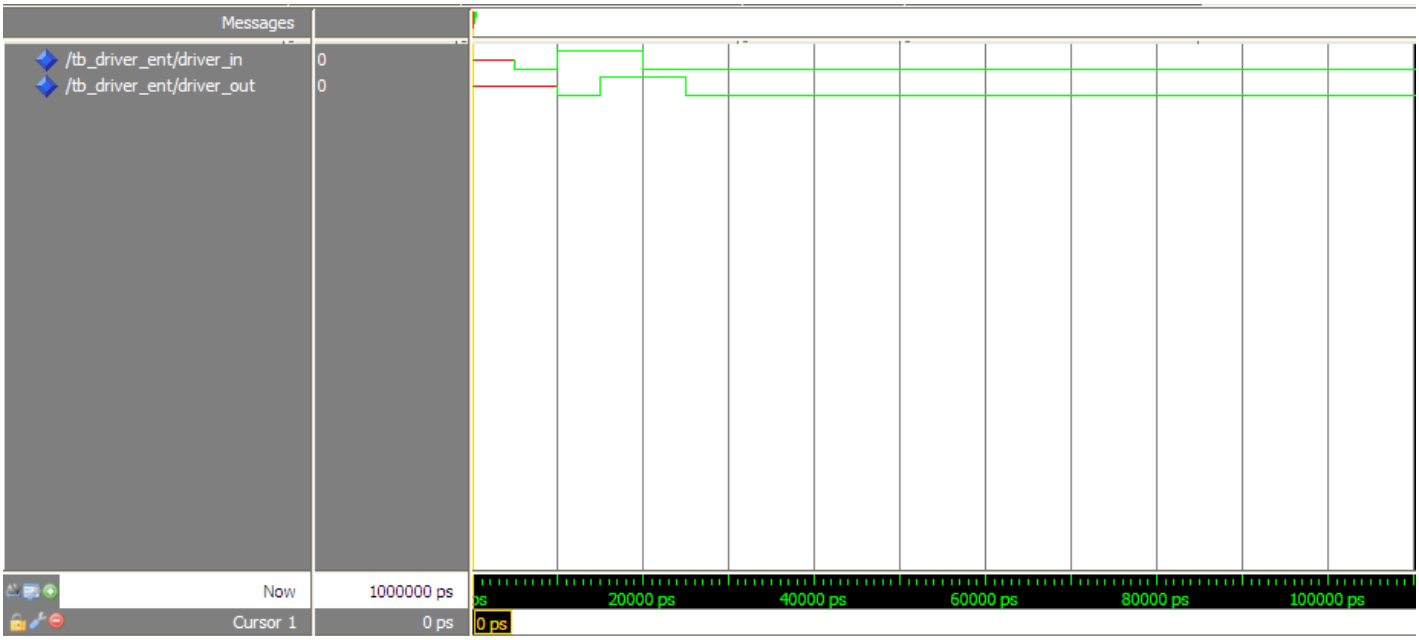
==========

tb_driver_ent.vhd

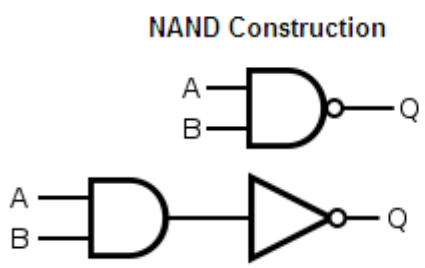
==========

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

-- Hendra Kesuma

ENTITY tb_driver_ent IS
END tb_driver_ent;

ARCHITECTURE tb_driver_arch OF tb_driver_ent IS
        COMPONENT driver_ent
        PORT(
                driver_in : IN std_logic;
                driver_out : OUT std_logic
                );
        END COMPONENT;
    FOR uut:driver_ent use entity work.driver_ent(driver_arch);
                SIGNAL driver_in :  std_logic;
            SIGNAL driver_out :  std_logic;
BEGIN
    driver_in<='0' after 5 ns, '1' after 10 ns, '0' after 20 ns;
    uut:driver_ent port map(driver_in, driver_out);
END tb_driver_arch;
```

SIMULATION



NAND GATE



Input A	Input B	Output Q
0	0	1
0	1	1
1	0	1
1	1	0

VHDL CODE (using ***if-else-then*** statement)

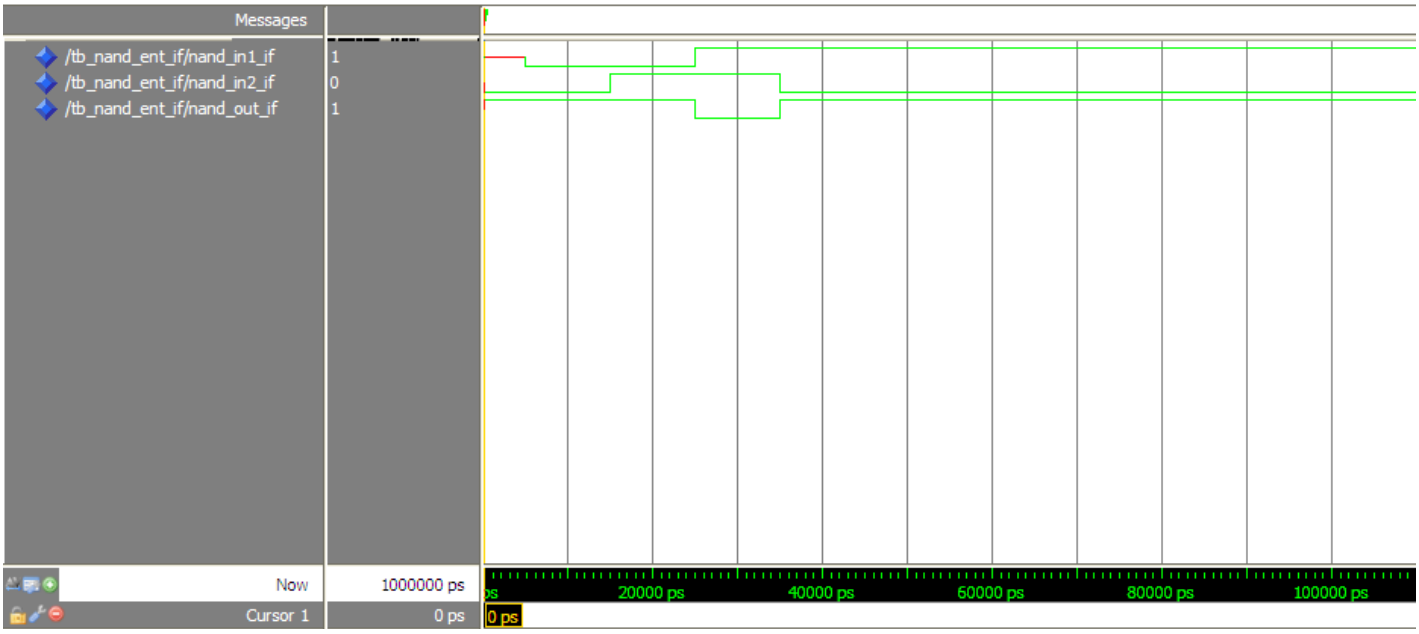
```
==========================================================================
nand_Gate_if.vhdl
==========================================================================
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Hendra Kesuma

entity nand_ent_if is
    Port ( nand_in1_if : in std_logic;
           nand_in2_if : in std_logic;
           nand_out_if : out std_logic);
end nand_ent_if;

architecture nand_arch_if of nand_ent_if is
begin
  process(nand_in1_if,nand_in2_if)
    begin
    if((nand_in1_if='1') and (nand_in2_if='1')) then
       nand_out_if<='0';
    else
       nand_out_if<='1';
    end if;
  end process;
end nand_arch_if;
```

TESTBENCH CODE

```
==========================================================================
tb_nand_ent_if.vhd
==========================================================================
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

-- Hendra Kesuma

ENTITY tb_nand_ent_if IS
END tb_nand_ent_if;
```

```
ARCHITECTURE tb_nand_arch_if OF tb_nand_ent_if IS

        COMPONENT nand_ent_if
        PORT(
                nand_in1_if : IN std_logic;
                nand_in2_if : IN std_logic;
                nand_out_if : OUT std_logic
                );
        END COMPONENT;
    FOR uut:nand_ent_if use entity work.nand_ent_if(nand_arch_if);
                SIGNAL nand_in1_if :  std_logic;
            SIGNAL nand_in2_if :  std_logic;
            SIGNAL nand_out_if :  std_logic;

BEGIN
    nand_in1_if<='0' after 5 ns, '1' after 25 ns;
    nand_in2_if<='0' after 0 ns, '1' after 15 ns, '0' after 35 ns;
        uut:nand_ent_if port map(nand_in1_if, nand_in2_if, nand_out_if);
end tb_nand_arch_if;
```

SIMULATION



VHDL CODE (using ***case-when*** statement)

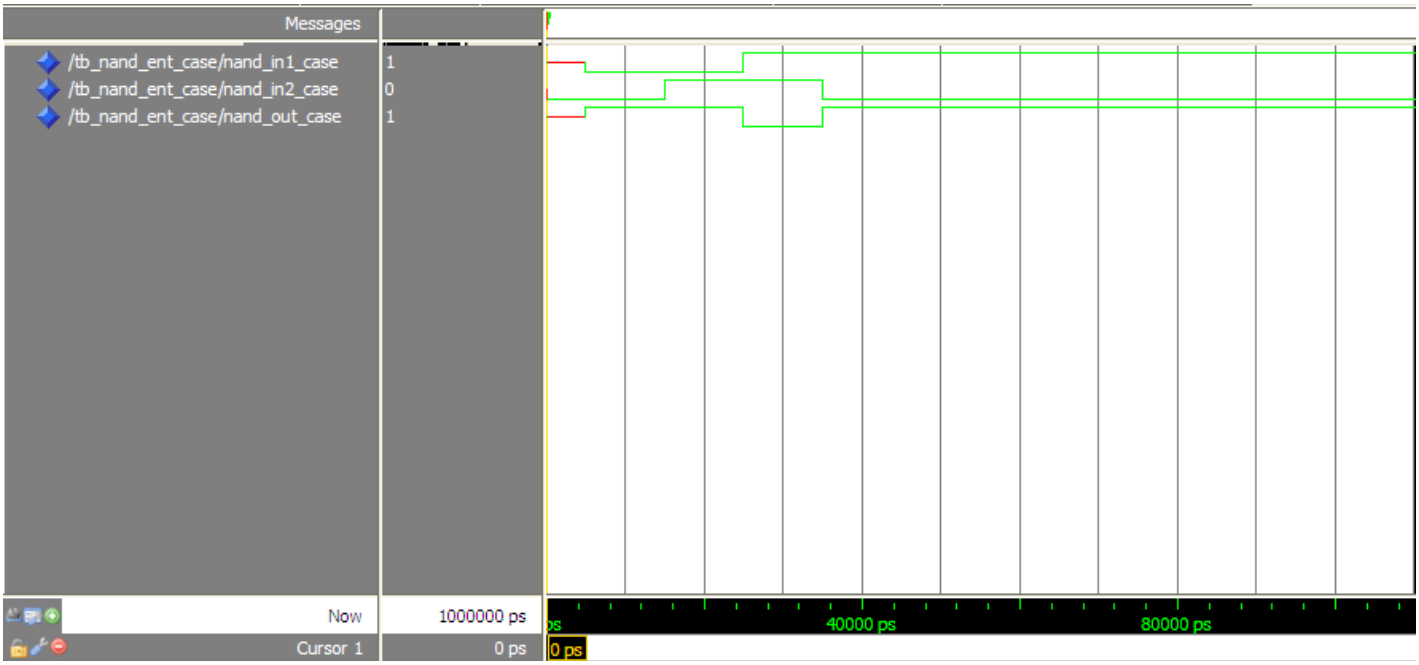
```
==========================================================================
nand_Gate_case.vhdl
==========================================================================
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Hendra Kesuma
```

```
entity nand_ent_case is
    Port ( nand_in1_case : in std_logic;
           nand_in2_case : in std_logic;
           nand_out_case : out std_logic);
end nand_ent_case;
architecture nand_arch_case of nand_ent_case is
begin
  process(nand_in1_case, nand_in2_case)
    variable Temp_variable : std_logic_vector (1 downto 0);
    begin
      Temp_variable := (nand_in1_case & nand_in2_case);
          case Temp_variable is
            when "00"  => nand_out_case <= '1';
            when "10"  => nand_out_case <= '1';
            when "01"  => nand_out_case <= '1';
            when "11"  => nand_out_case <= '0';
            when others => nand_out_case <= 'U';
          end case;
  end process;
end nand_arch_case;
```

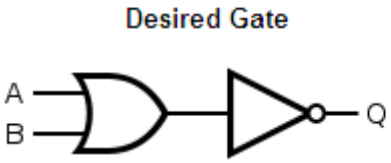
TESTBENCH CODE

```
================================================================
tb_nand_ent_case.vhd
================================================================
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

-- Hendra Kesuma

ENTITY tb_nand_ent_case IS
END tb_nand_ent_case;

ARCHITECTURE tb_nand_arch_case OF tb_nand_ent_case IS

        COMPONENT nand_ent_case
        PORT(
                nand_in1_case : IN std_logic;
                nand_in2_case : IN std_logic;
                nand_out_case : OUT std_logic
                );
        END COMPONENT;
    FOR uut:nand_ent_case use entity work.nand_ent_case(nand_arch_case);
                SIGNAL nand_in1_case :  std_logic;
            SIGNAL nand_in2_case :  std_logic;
            SIGNAL nand_out_case :  std_logic;

BEGIN
    nand_in1_case<='0' after 5 ns, '1' after 25 ns;
    nand_in2_case<='0' after 0 ns, '1' after 15 ns, '0' after 35 ns;
        uut:nand_ent_case port map(nand_in1_case, nand_in2_case, nand_out_case);
end tb_nand_arch_case;
```

SIMULATION



NOR GATE



Input A	Input B	Output Q
0	0	1
0	1	0
1	0	0
1	1	0

VHDL CODE (using ***if-else-then*** statement)

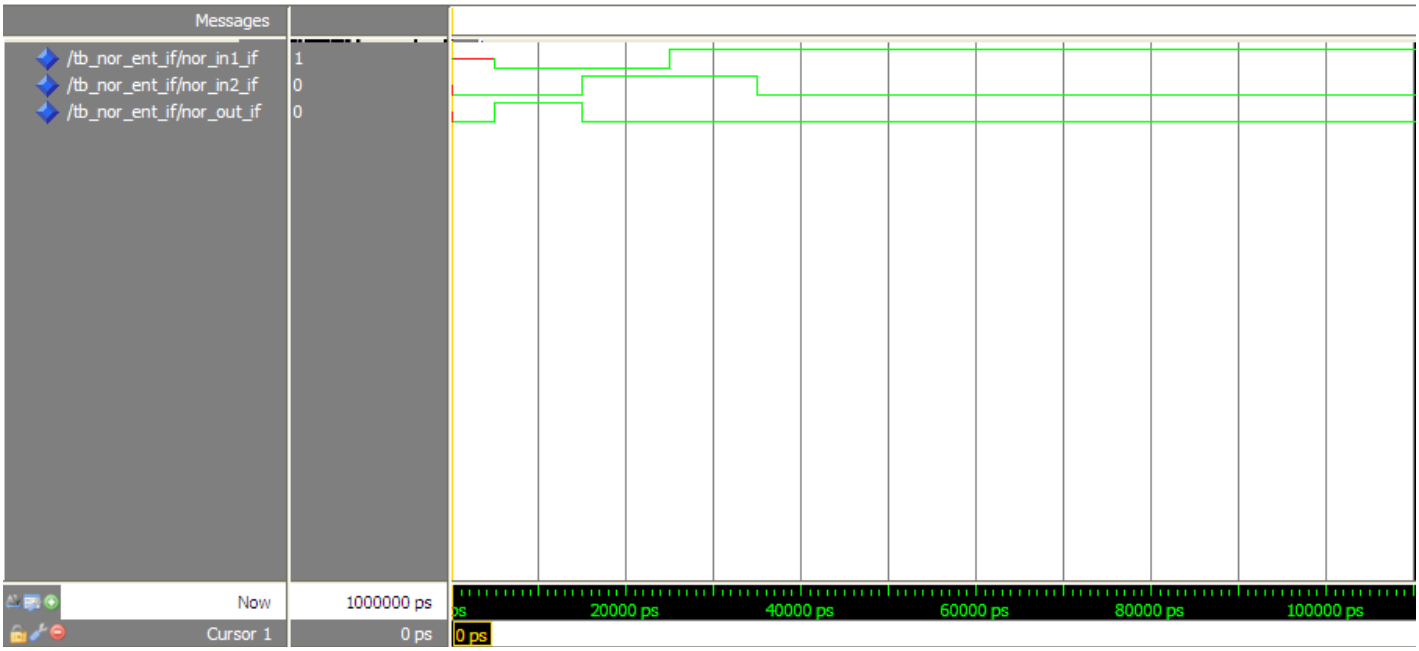
```
================================================================
nor_Gate_if.vhdl
================================================================
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Hendra Kesuma

entity nor_ent_if is
    Port ( nor_in1_if : in std_logic;
           nor_in2_if : in std_logic;
           nor_out_if : out std_logic);
end nor_ent_if;
```

```
architecture nor_arch_if of nor_ent_if is
begin
  process(nor_in1_if,nor_in2_if)
    begin
    if((nor_in1_if='0') and (nor_in2_if='0')) then
       nor_out_if<='1';
    else
       nor_out_if<='0';
    end if;
  end process;
end nor_arch_if;
```

TESTBENCH CODE

```
================================================================
tb_nor_ent_if.vhd
================================================================
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

-- Hendra Kesuma

ENTITY tb_nor_ent_if IS
END tb_nor_ent_if;

ARCHITECTURE tb_nor_arch_if OF tb_nor_ent_if IS
        COMPONENT nor_ent_if
        PORT(
                nor_in1_if : IN std_logic;
                nor_in2_if : IN std_logic;
                nor_out_if : OUT std_logic
                );
        END COMPONENT;
    FOR uut:nor_ent_if use entity work.nor_ent_if(nor_arch_if);
                SIGNAL nor_in1_if :  std_logic;
                SIGNAL nor_in2_if :  std_logic;
                SIGNAL nor_out_if :  std_logic;

BEGIN
    nor_in1_if<='0' after 5 ns, '1' after 25 ns;
    nor_in2_if<='0' after 0 ns, '1' after 15 ns, '0' after 35 ns;
        uut:nor_ent_if port map(nor_in1_if, nor_in2_if, nor_out_if);
end tb_nor_arch_if;
```

SIMULATION



VHDL CODE (using ***case-when*** statement)

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

-- Hendra Kesuma

entity nor_ent_case is
    Port ( nor_in1_case : in std_logic;
           nor_in2_case : in std_logic;
           nor_out_case : out std_logic);
end nor_ent_case;

architecture nor_arch_case of nor_ent_case is
begin
  process(nor_in1_case, nor_in2_case)
    variable Temp_variable : std_logic_vector (1 downto 0);
    begin
      Temp_variable := (nor_in1_case & nor_in2_case);
           case Temp_variable is
             when "00"  => nor_out_case <= '1';
             when "10"  => nor_out_case <= '0';
             when "01"  => nor_out_case <= '0';
             when "11"  => nor_out_case <= '0';
             when others => nor_out_case <= 'U';
           end case;
  end process;
end nor_arch_case;
```

TESTBENCH CODE

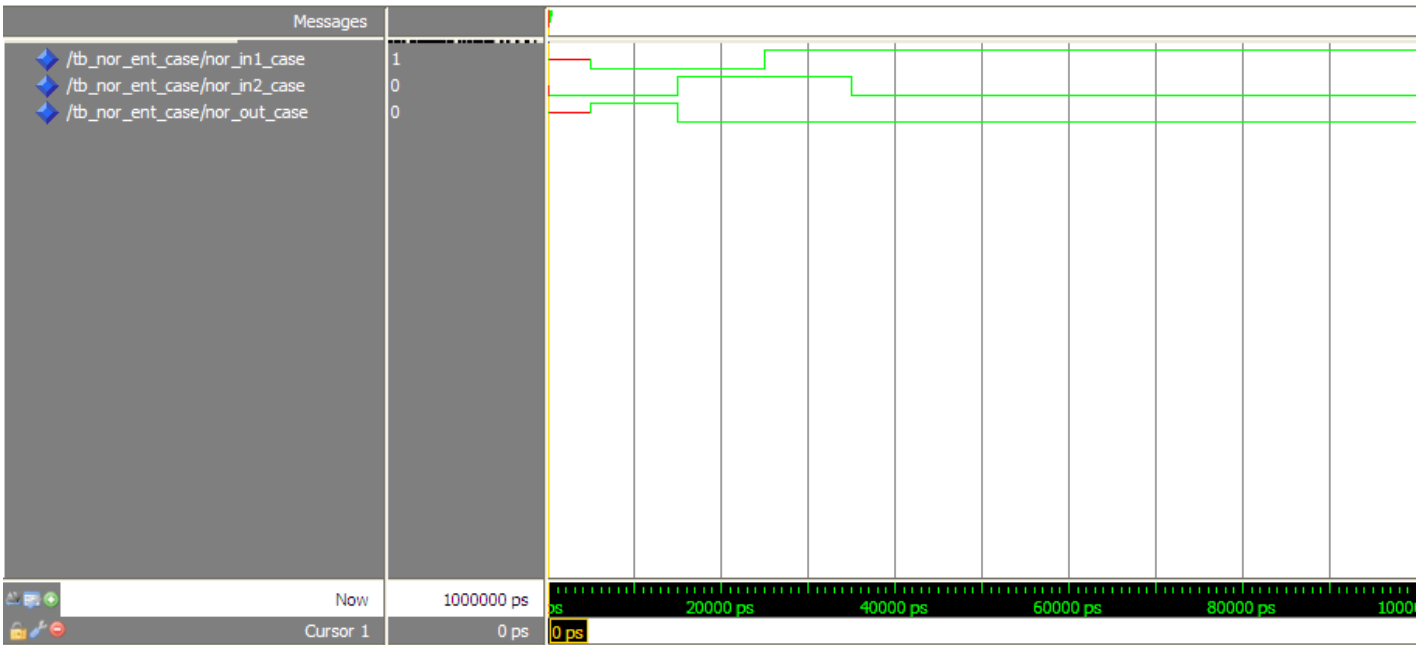
==========

tb_nor_ent_case.vhd

==========

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

-- Hendra Kesuma

ENTITY tb_nor_ent_case IS
END tb_nor_ent_case;

ARCHITECTURE tb_nor_arch_case OF tb_nor_ent_case IS

        COMPONENT nor_ent_case
        PORT(
                nor_in1_case : IN std_logic;
                nor_in2_case : IN std_logic;
                nor_out_case : OUT std_logic
                );
        END COMPONENT;
    FOR uut:nor_ent_case use entity work.nor_ent_case(nor_arch_case);
                SIGNAL nor_in1_case :  std_logic;
            SIGNAL nor_in2_case :  std_logic;
            SIGNAL nor_out_case :  std_logic;

BEGIN
    nor_in1_case<='0' after 5 ns, '1' after 25 ns;
    nor_in2_case<='0' after 0 ns, '1' after 15 ns, '0' after 35 ns;
        uut:nor_ent_case port map(nor_in1_case, nor_in2_case, nor_out_case);
end tb_nor_arch_case;
```

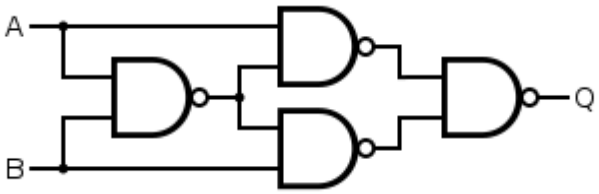
SIMULATION



XOR GATE

Desired Gate

NAND Construction



Input A	Input B	Output Q
0	0	0
0	1	1
1	0	1
1	1	0

VHDL CODE (using ***if-else-then*** statement)

==========

xor_Gate_if.vhdl

==========

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Hendra Kesuma

entity xor_ent_if is
    Port ( xor_in1_if : in std_logic;
           xor_in2_if : in std_logic;
           xor_out_if : out std_logic);
end xor_ent_if;

architecture xor_arch_if of xor_ent_if is
begin
  process(xor_in1_if,xor_in2_if)
    begin
    if((xor_in1_if='0') and (xor_in2_if='0')) then
       xor_out_if<='0';
    elsif((xor_in1_if='1') and (xor_in2_if='1')) then
       xor_out_if<='0';
    else
       xor_out_if<='1';
    end if;
  end process;
end xor_arch_if;
```

TESTBENCH CODE

==========

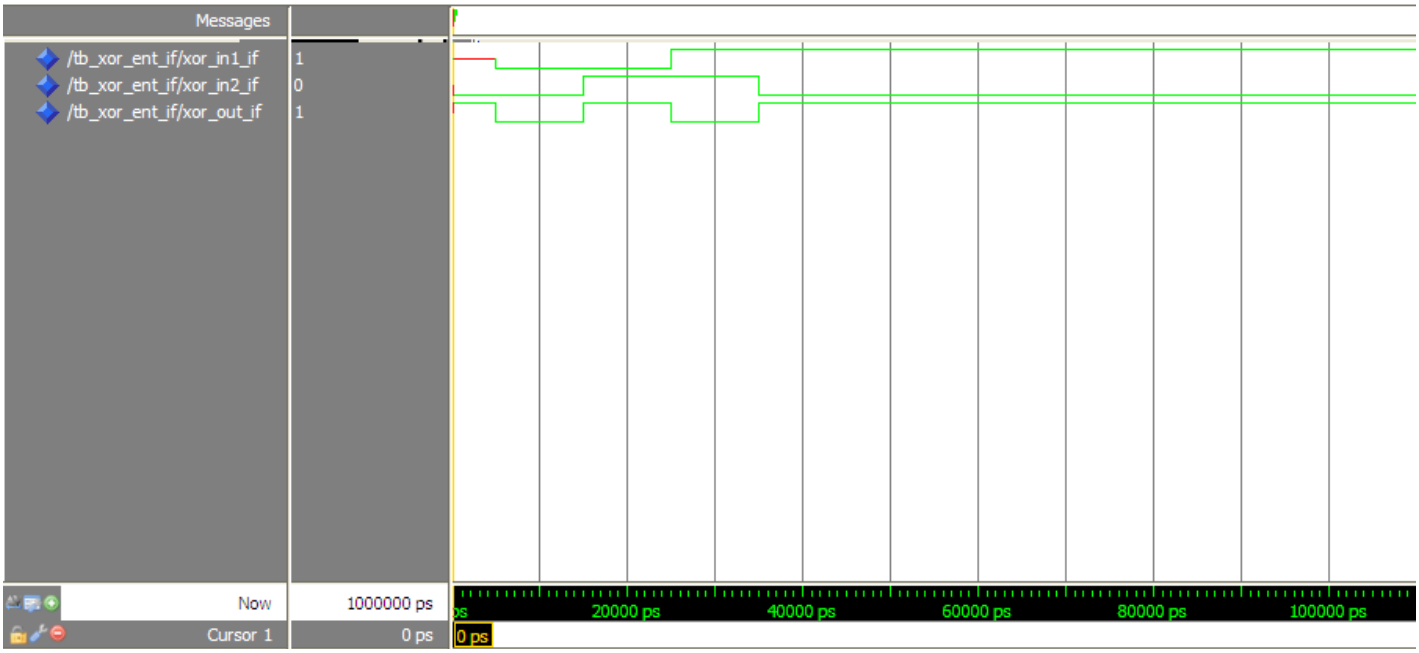
tb_xor_ent_if.vhd

==========

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

-- Hendra Kesuma

ENTITY tb_xor_ent_if IS
END tb_xor_ent_if;
```

```
ARCHITECTURE tb_xor_arch_if OF tb_xor_ent_if IS

        COMPONENT xor_ent_if
        PORT(
                xor_in1_if : IN std_logic;
                xor_in2_if : IN std_logic;
                xor_out_if : OUT std_logic
                );
        END COMPONENT;
    FOR uut:xor_ent_if use entity work.xor_ent_if(xor_arch_if);
                SIGNAL xor_in1_if :  std_logic;
            SIGNAL xor_in2_if :  std_logic;
            SIGNAL xor_out_if :  std_logic;

BEGIN
    xor_in1_if<='0' after 5 ns, '1' after 25 ns;
    xor_in2_if<='0' after 0 ns, '1' after 15 ns, '0' after 35 ns;
        uut:xor_ent_if port map(xor_in1_if, xor_in2_if, xor_out_if);
end tb_xor_arch_if;
```

SIMULATION



VHDL CODE (using ***case-when*** statement)

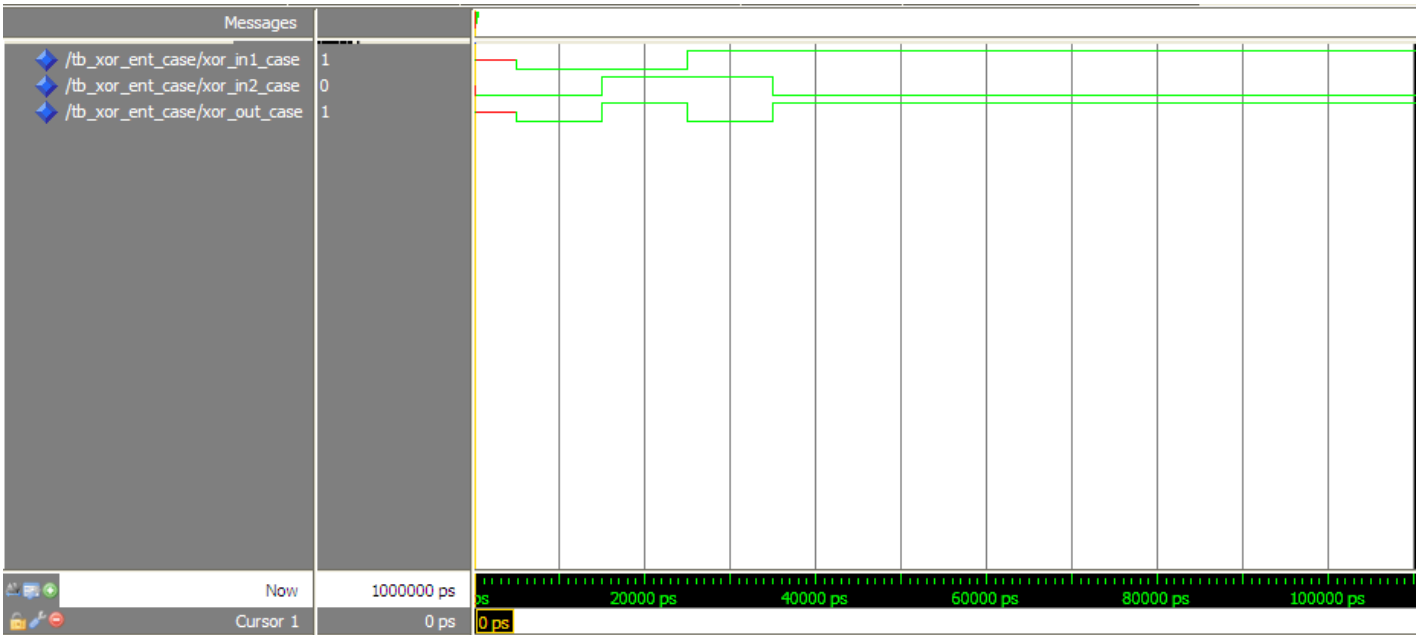
```
==================================================================
xor_Gate_case.vhdl
==================================================================
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Hendra Kesuma
```

```
entity xor_ent_case is
    Port ( xor_in1_case : in std_logic;
           xor_in2_case : in std_logic;
           xor_out_case : out std_logic);
end xor_ent_case;

architecture xor_arch_case of xor_ent_case is
begin
  process(xor_in1_case, xor_in2_case)
    variable Temp_variable : std_logic_vector (1 downto 0);
    begin
      Temp_variable := (xor_in1_case & xor_in2_case);
          case Temp_variable is
            when "00"  => xor_out_case <= '0';
            when "10"  => xor_out_case <= '1';
            when "01"  => xor_out_case <= '1';
            when "11"  => xor_out_case <= '0';
            when others => xor_out_case <= 'U';
          end case;
  end process;
end xor_arch_case;
```

TESTBENCH CODE

```
================================================================
tb_xor_ent_case.vhd
================================================================
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

-- Hendra Kesuma

ENTITY tb_xor_ent_case IS
END tb_xor_ent_case;

ARCHITECTURE tb_xor_arch_case OF tb_xor_ent_case IS

        COMPONENT xor_ent_case
        PORT(
                xor_in1_case : IN std_logic;
                xor_in2_case : IN std_logic;
                xor_out_case : OUT std_logic
                );
        END COMPONENT;
    FOR uut:xor_ent_case use entity work.xor_ent_case(xor_arch_case);
                SIGNAL xor_in1_case :  std_logic;
           SIGNAL xor_in2_case :  std_logic;
           SIGNAL xor_out_case :  std_logic;

BEGIN
    xor_in1_case<='0' after 5 ns, '1' after 25 ns;
    xor_in2_case<='0' after 0 ns, '1' after 15 ns, '0' after 35 ns;
        uut:xor_ent_case port map(xor_in1_case, xor_in2_case, xor_out_case);
end tb_xor_arch_case;
```

SIMULATION


Messages
/tb_xor_ent_case/xor_in1_case
/tb_xor_ent_case/xor_in2_case
/tb_xor_ent_case/xor_out_case
1
0
1
Now
1000000 ps
Cursor 1
0 ps
20000 ps
40000 ps
60000 ps
80000 ps
100000 ps
0 ps