And:

Design Code:

```
 -- Entity declaration
entity AND_gate is
   Port ( A, B : in  std_logic;
          Y    : out std_logic);
end AND_gate;

-- Architecture definition
architecture Behavioral of AND_gate is
begin
   Y <= A and B;  -- AND gate behavior
end Behavioral;
```

Simulation Code:

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

entity AND_gate_tb is
end AND_gate_tb;

architecture Behavioral of AND_gate_tb is
   component AND_gate
      Port ( A, B : in  std_logic;
             Y    : out std_logic);
   end component;

   signal A_tb, B_tb, Y_tb : std_logic;
begin
   -- Instantiate the AND_gate component
   AND_gate_inst : AND_gate
      port map (A => A_tb, B => B_tb, Y => Y_tb);

   -- Stimulus process
   stimulus_process: process
```

```
    begin
        A_tb <= '0';
        B_tb <= '0';
        wait for 10 ns;

        A_tb <= '0';
        B_tb <= '1';
        wait for 10 ns;

        A_tb <= '1';
        B_tb <= '0';
        wait for 10 ns;

        A_tb <= '1';
        B_tb <= '1';
        wait for 10 ns;

        wait;
    end process;
end Behavioral;
```

Or:

Design Code:

```
-- Entity declaration
entity OR_gate is
    Port ( A, B : in  std_logic;
           Y    : out std_logic);
end OR_gate;

-- Architecture definition
architecture Behavioral of OR_gate is
begin
    Y <= A or B;  -- OR gate behavior
end Behavioral;
```

Simulation Code:

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

entity OR_gate_tb is
end OR_gate_tb;

architecture Behavioral of OR_gate_tb is
    component OR_gate
        Port ( A, B : in  std_logic;
               Y    : out std_logic);
    end component;

    signal A_tb, B_tb, Y_tb : std_logic;
begin
    -- Instantiate the OR_gate component
    OR_gate_inst : OR_gate
        port map (A => A_tb, B => B_tb, Y => Y_tb);

    -- Stimulus process
    stimulus_process: process
    begin
        A_tb <= '0';
        B_tb <= '0';
        wait for 10 ns;

        A_tb <= '0';
        B_tb <= '1';
        wait for 10 ns;

        A_tb <= '1';
        B_tb <= '0';
        wait for 10 ns;

        A_tb <= '1';
```

```
        B_tb <= '1';
        wait for 10 ns;

        wait;
    end process;
end Behavioral;
```

Not:

Design Code:

```
-- Entity declaration
entity NOT_gate is
    Port ( A : in  std_logic;
           Y : out std_logic);
end NOT_gate;

-- Architecture definition
architecture Behavioral of NOT_gate is
begin
    Y <= not A;  -- NOT gate behavior
end Behavioral;
```

Simulation Code:

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

entity NOT_gate_tb is
end NOT_gate_tb;

architecture Behavioral of NOT_gate_tb is
    component NOT_gate
        Port ( A : in  std_logic;
               Y : out std_logic);
    end component;

    signal A_tb, Y_tb : std_logic;
begin
    -- Instantiate the NOT_gate component
```

```
    NOT_gate_inst : NOT_gate
        port map (A => A_tb, Y => Y_tb);

    -- Stimulus process
    stimulus_process: process
    begin
        A_tb <= '0';
        wait for 10 ns;

        A_tb <= '1';
        wait for 10 ns;

        wait;
    end process;
end Behavioral;
```

XOR:

Design Code:

```
-- Entity declaration
entity XOR_gate is
    Port ( A, B : in  std_logic;
           Y    : out std_logic);
end XOR_gate;

-- Architecture definition
architecture Behavioral of XOR_gate is
begin
    Y <= A xor B;  -- XOR gate behavior
end Behavioral;
```

Simulation Code:

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

entity XOR_gate_tb is
end XOR_gate_tb;

architecture Behavioral of XOR_gate_tb is
```

```
    component XOR_gate
        Port ( A, B : in  std_logic;
               Y    : out std_logic);
    end component;

    signal A_tb, B_tb, Y_tb : std_logic;
begin
    -- Instantiate the XOR_gate component
    XOR_gate_inst : XOR_gate
        port map (A => A_tb, B => B_tb, Y => Y_tb);

    -- Stimulus process
    stimulus_process: process
    begin
        A_tb <= '0';
        B_tb <= '0';
        wait for 10 ns;

        A_tb <= '0';
        B_tb <= '1';
        wait for 10 ns;

        A_tb <= '1';
        B_tb <= '0';
        wait for 10 ns;

        A_tb <= '1';
        B_tb <= '1';
        wait for 10 ns;

        wait;
    end process;
end Behavioral;
```

XNOR:

Design Code: -- Entity declaration

```
entity XNOR_gate is
   Port ( A, B : in  std_logic;
          Y    : out std_logic);
end XNOR_gate;

-- Architecture definition
architecture Behavioral of XNOR_gate is
begin
   Y <= not (A xor B);  -- XNOR gate behavior
end Behavioral;
```

Simulation Code: library IEEE;

```
use IEEE.STD_LOGIC_1164.ALL;

entity XNOR_gate_tb is
end XNOR_gate_tb;

architecture Behavioral of XNOR_gate_tb is
   component XNOR_gate
      Port ( A, B : in  std_logic;
             Y    : out std_logic);
   end component;

   signal A_tb, B_tb, Y_tb : std_logic;
begin
   -- Instantiate the XNOR_gate component
   XNOR_gate_inst : XNOR_gate
      port map (A => A_tb, B => B_tb, Y => Y_tb);

   -- Stimulus process
   stimulus_process: process
   begin
      A_tb <= '0';
      B_tb <= '0';
```

```
        wait for 10 ns;

        A_tb <= '0';
        B_tb <= '1';
        wait for 10 ns;

        A_tb <= '1';
        B_tb <= '0';
        wait for 10 ns;

        A_tb <= '1';
        B_tb <= '1';
        wait for 10 ns;

        wait;
    end process;
end Behavioral;
```

NAND:

Design Code:

```
-- Entity declaration
entity NAND_gate is
    Port ( A, B : in  std_logic;
           Y    : out std_logic);
end NAND_gate;

-- Architecture definition
architecture Behavioral of NAND_gate is
begin
    Y <= not (A and B);  -- NAND gate behavior
end Behavioral;
```

Simulation Code:

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

entity NAND_gate_tb is
end NAND_gate_tb;
```

```
architecture Behavioral of NAND_gate_tb is
    component NAND_gate
        Port ( A, B : in  std_logic;
               Y    : out std_logic);
    end component;

    signal A_tb, B_tb, Y_tb : std_logic;
begin
    -- Instantiate the NAND_gate component
    NAND_gate_inst : NAND_gate
        port map (A => A_tb, B => B_tb, Y => Y_tb);

    -- Stimulus process
    stimulus_process: process
    begin
        A_tb <= '0';
        B_tb <= '0';
        wait for 10 ns;

        A_tb <= '0';
        B_tb <= '1';
        wait for 10 ns;

        A_tb <= '1';
        B_tb <= '0';
        wait for 10 ns;

        A_tb <= '1';
        B_tb <= '1';
        wait for 10 ns;

        wait;
    end process;
```

```
end Behavioral;
```

NOR:

Design Code: -- Entity declaration

```
entity NOR_gate is
   Port ( A, B : in  std_logic;
          Y    : out std_logic);
end NOR_gate;

-- Architecture definition
architecture Behavioral of NOR_gate is
begin
   Y <= not (A or B);  -- NOR gate behavior
end Behavioral;
```

Simulation Code: library IEEE;

```
use IEEE.STD_LOGIC_1164.ALL;

entity NOR_gate_tb is
end NOR_gate_tb;

architecture Behavioral of NOR_gate_tb is
   component NOR_gate
      Port ( A, B : in  std_logic;
             Y    : out std_logic);
   end component;

   signal A_tb, B_tb, Y_tb : std_logic;
begin
   -- Instantiate the NOR_gate component
   NOR_gate_inst : NOR_gate
      port map (A => A_tb, B => B_tb, Y => Y_tb);

   -- Stimulus process
   stimulus_process: process
   begin
```

```
        A_tb <= '0';
        B_tb <= '0';
        wait for 10 ns;

        A_tb <= '0';
        B_tb <= '1';
        wait for 10 ns;

        A_tb <= '1';
        B_tb <= '0';
        wait for 10 ns;

        A_tb <= '1';
        B_tb <= '1';
        wait for 10 ns;

        wait;
    end process;
end Behavioral;
```