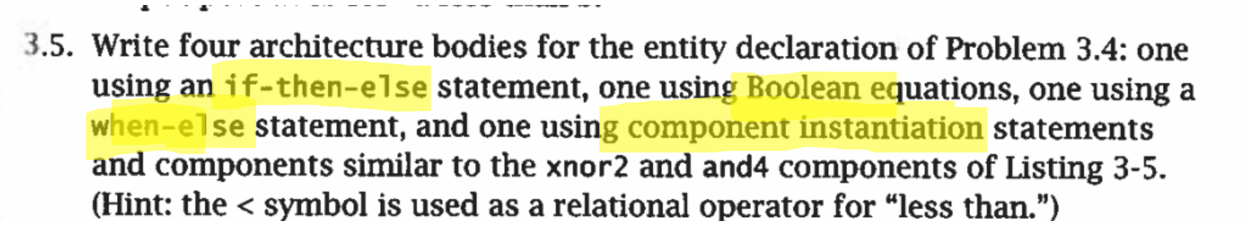
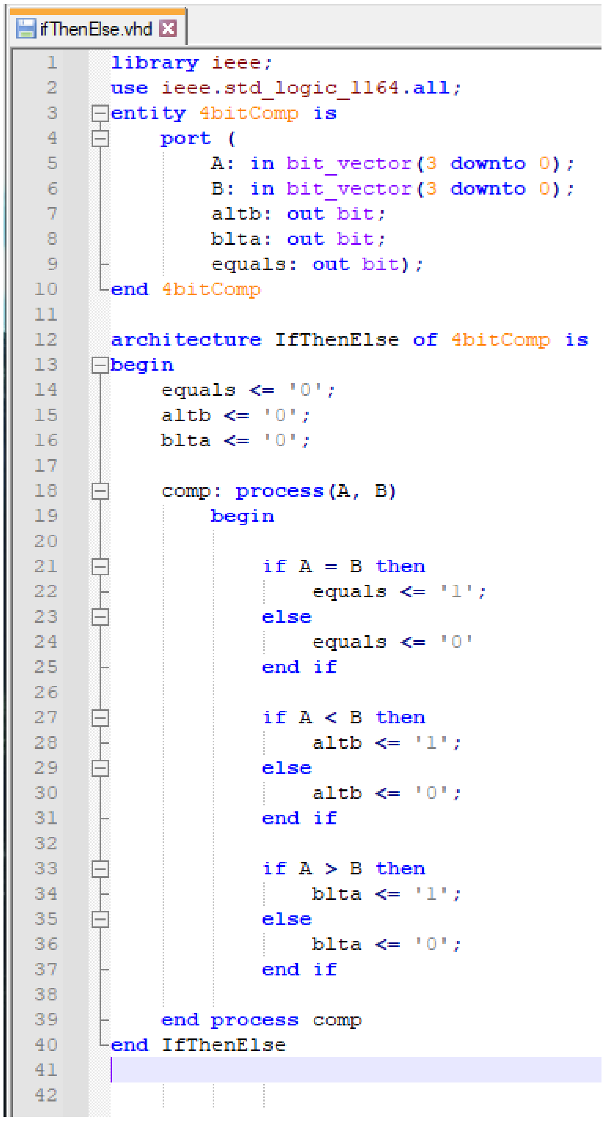
1. Do question 3.5 (chapter 3). This question can be seen in the attachment with the submission link for HW4. Please note that if your Boolean equation or gate-level design itself is not 100% correct then that will not make you lose marks. The purpose here is to familiarize you with the different ways you can represent a digital design in VHDL. When actually designing a circuit/system you will select the best way yourself. In this type of problem, that will be through if-then-else or when-else statement, but this problem asks for more just to show you other possibilities in VHDL.



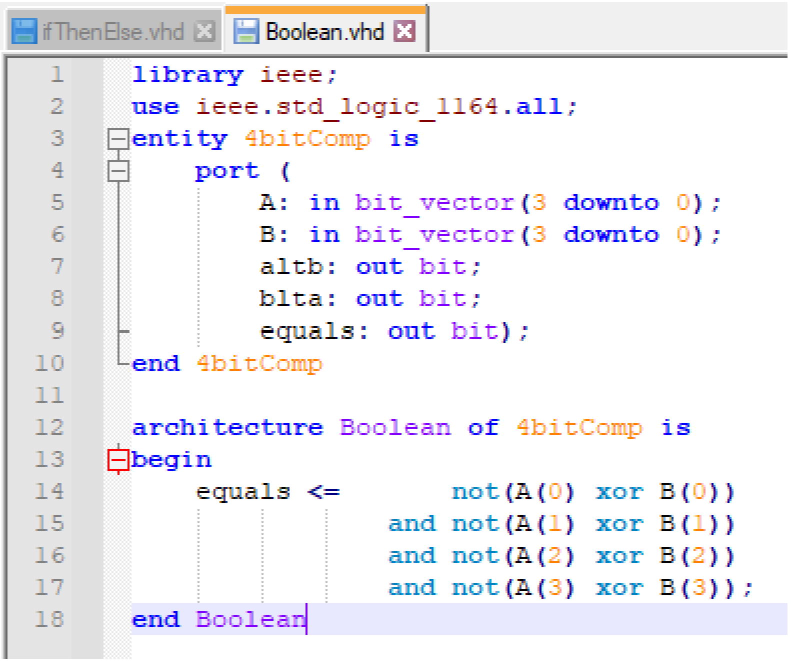
3.5. Write four architecture bodies for the entity declaration of Problem 3.4: one using an if-then-else statement, one using Boolean equations, one using a when-else statement, and one using component instantiation statements and components similar to the xnor2 and and4 components of Listing 3-5. (Hint: the < symbol is used as a relational operator for "less than.")

If-then-else



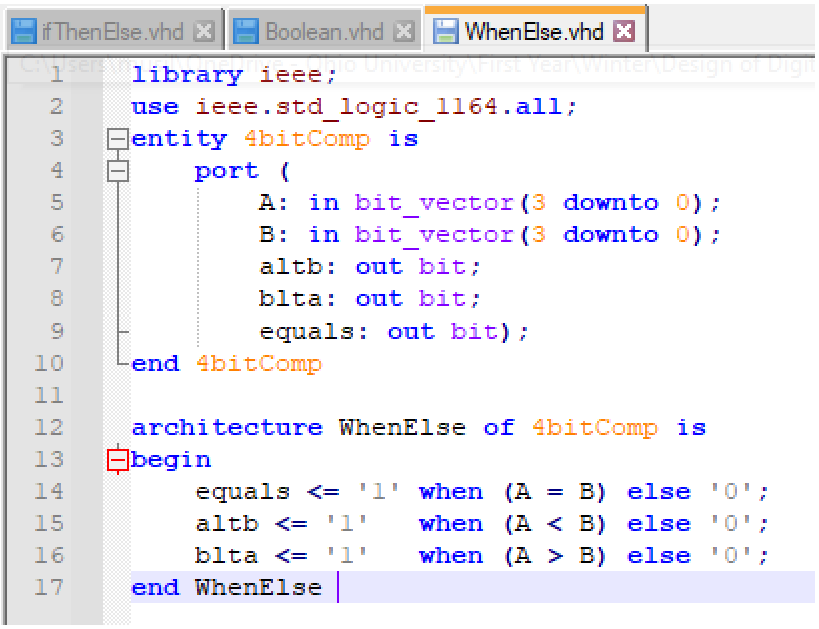
```
library ieee;
use ieee.std_logic_1164.all;
entity 4bitComp is
    port (
        A: in bit_vector(3 downto 0);
        B: in bit_vector(3 downto 0);
        altb: out bit;
        blta: out bit;
        equals: out bit);
end 4bitComp

architecture IfThenElse of 4bitComp is
begin
    equals <= '0';
    altb <= '0';
    blta <= '0';

    comp: process(A, B)
        begin

            if A = B then
                equals <= '1';
            else
                equals <= '0'
            end if

            if A < B then
                altb <= '1';
            else
                altb <= '0';
            end if

            if A > B then
                blta <= '1';
            else
                blta <= '0';
            end if

    end process comp
end IfThenElse
```

Boolean



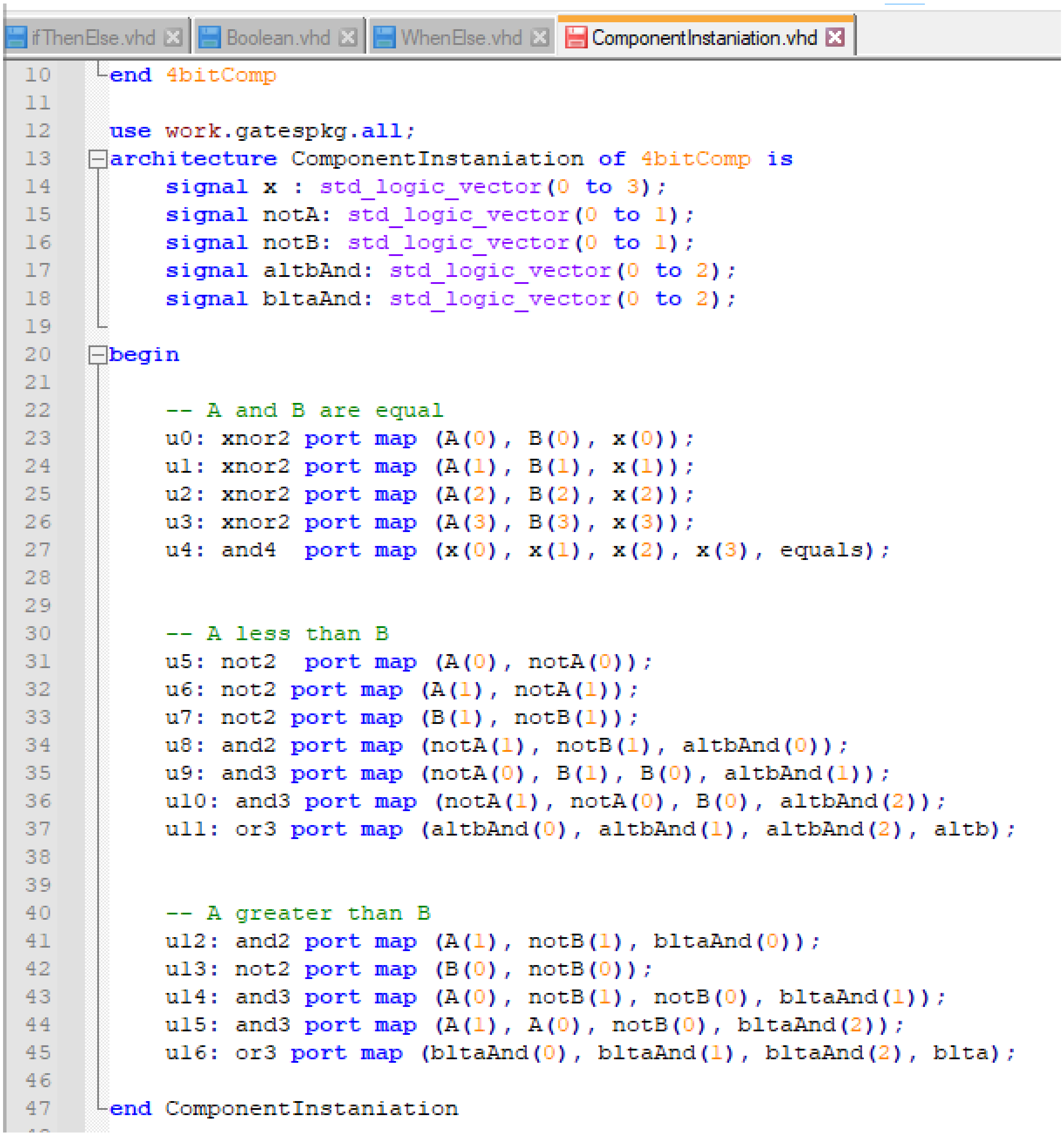
```
library ieee;
use ieee.std_logic_1164.all;
entity 4bitComp is
    port (
        A: in bit_vector(3 downto 0);
        B: in bit_vector(3 downto 0);
        altb: out bit;
        blta: out bit;
        equals: out bit);
end 4bitComp

architecture Boolean of 4bitComp is
begin
    equals <=       not(A(0) xor B(0))
                and not(A(1) xor B(1))
                and not(A(2) xor B(2))
                and not(A(3) xor B(3));
end Boolean
```

When-else



```
library ieee;
use ieee.std_logic_1164.all;
entity 4bitComp is
    port (
        A: in bit_vector(3 downto 0);
        B: in bit_vector(3 downto 0);
        altb: out bit;
        blta: out bit;
        equals: out bit);
end 4bitComp

architecture WhenElse of 4bitComp is
begin
    equals <= '1' when (A = B) else '0';
    altb <= '1'   when (A < B) else '0';
    blta <= '1'   when (A > B) else '0';
end WhenElse
```

Component Instantiation



```
end 4bitComp

use work.gatespkg.all;
architecture ComponentInstaniation of 4bitComp is
    signal x : std_logic_vector(0 to 3);
    signal notA: std_logic_vector(0 to 1);
    signal notB: std_logic_vector(0 to 1);
    signal altbAnd: std_logic_vector(0 to 2);
    signal bltaAnd: std_logic_vector(0 to 2);

begin

    -- A and B are equal
    u0: xnor2 port map (A(0), B(0), x(0));
    u1: xnor2 port map (A(1), B(1), x(1));
    u2: xnor2 port map (A(2), B(2), x(2));
    u3: xnor2 port map (A(3), B(3), x(3));
    u4: and4  port map (x(0), x(1), x(2), x(3), equals);


    -- A less than B
    u5: not2  port map (A(0), notA(0));
    u6: not2 port map (A(1), notA(1));
    u7: not2 port map (B(1), notB(1));
    u8: and2 port map (notA(1), notB(1), altbAnd(0));
    u9: and3 port map (notA(0), B(1), B(0), altbAnd(1));
    u10: and3 port map (notA(1), notA(0), B(0), altbAnd(2));
    u11: or3 port map (altbAnd(0), altbAnd(1), altbAnd(2), altb);


    -- A greater than B
    u12: and2 port map (A(1), notB(1), bltaAnd(0));
    u13: not2 port map (B(0), notB(0));
    u14: and3 port map (A(0), notB(1), notB(0), bltaAnd(1));
    u15: and3 port map (A(1), A(0), notB(0), bltaAnd(2));
    u16: or3 port map (bltaAnd(0), bltaAnd(1), bltaAnd(2), blta);

end ComponentInstaniation
```