Practice Exercises

Topic #04 - a) Basic synthesizable data-types

Exercise #1- Legal or Illegal Assignments?

& means append!

1) Look at these signals... Which are the legal assignments? Why?

```
SIGNAL a: STD_LOGIC;
SIGNAL b: BIT;
SIGNAL x: byte;
SIGNAL y: STD_LOGIC_VECTOR (7 DOWNTO 0);
SIGNAL v: BIT_VECTOR (3 DOWNTO 0);
SIGNAL z: STD_LOGIC_VECTOR (7 DOWNTO 0);
```

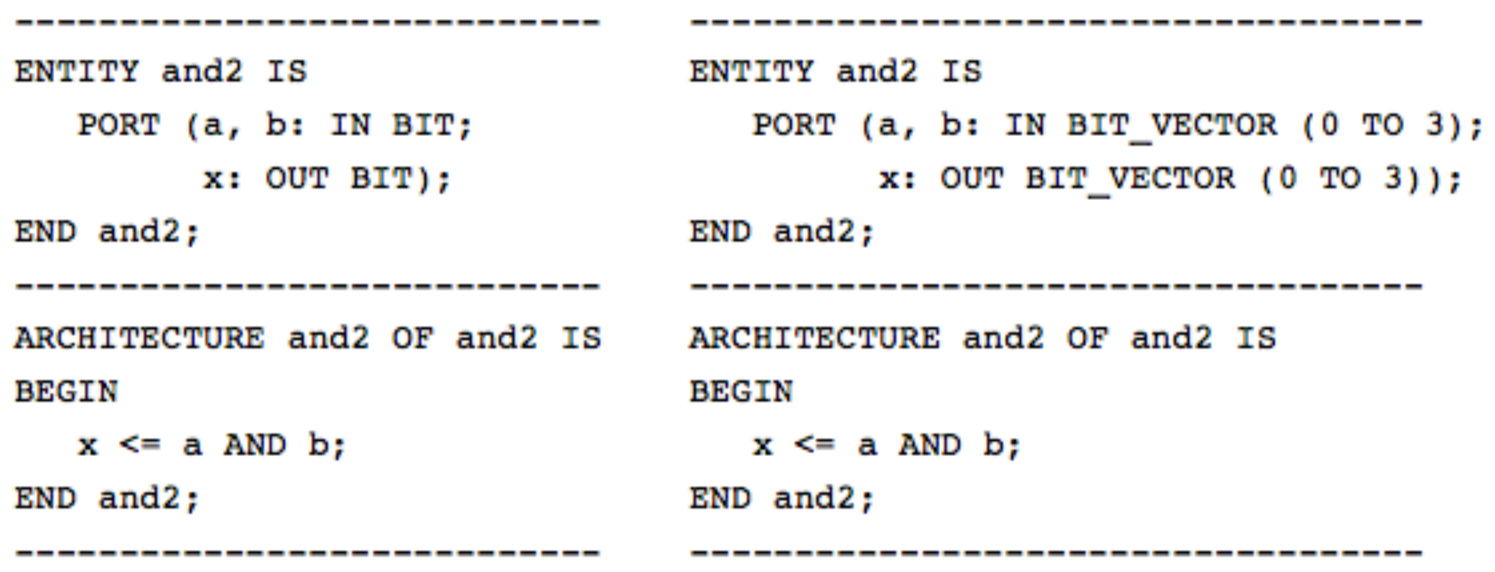
Not sure about some of these? Use Active HDL to check.

```
z <= "11111" & "000";
x(2) <= a;
b <= a;
y(5 TO 7) <= z(6 DOWNTO 0);
y(0) <= x(0);
y <= ('1','1','1','1','1','1','0','Z');
x <= "11111110";
z <= y;
b <= v(3);
y(2 DOWNTO 0) <= z(6 DOWNTO 4);
x <= y;
z(7) <= x(5);
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

entity test is
end entity;

architecture myarch of test is
signal a : std_logic;
signal b : bit;
--byte is not a standard VHDL data-type
--signal x : byte;
signal y : std_logic_vector(7 downto 0);
signal v : bit_vector(3 downto 0);
signal x : std_logic_vector(7 downto 0);
signal z : std_logic_vector (7 downto 0);

begin
  z <= "11111" & "000";  -- z will have 8 bits
  x(2) <= a; -- assigning a single std_logic
  --b <= a; --illegal cannot combine bit and std_logic
  --y(5 to 7) <= z(6 downto 0); --illegal size mismatch
  y(0) <= x(0); -- setting a single std_logic element
  y <= ('1','1','1','1','1','1','0','Z');
  x <= "11111110";
  z <= y; -- same size and same data types
  y(2 downto 0)<= z(6 downto 4); --valid size is same
  x <= y;
  z(7) <= x(5); --fine, since we use just one element
end architecture;
```

You don't have to do this, but here is the code to test these assignments.

Exercise #2- What is the difference between these two implementations



```
------------------------------
ENTITY and2 IS
   PORT (a, b: IN BIT;
         x: OUT BIT);
END and2;
------------------------------
ARCHITECTURE and2 OF and2 IS
BEGIN
   x <= a AND b;
END and2;
------------------------------
```

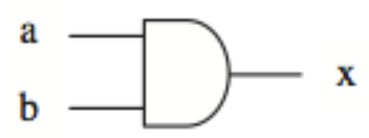
```
-------------------------------------
ENTITY and2 IS
   PORT (a, b: IN BIT_VECTOR (0 TO 3);
         x: OUT BIT_VECTOR (0 TO 3));
END and2;
-------------------------------------
ARCHITECTURE and2 OF and2 IS
BEGIN
   x <= a AND b;
END and2;
-------------------------------------
```

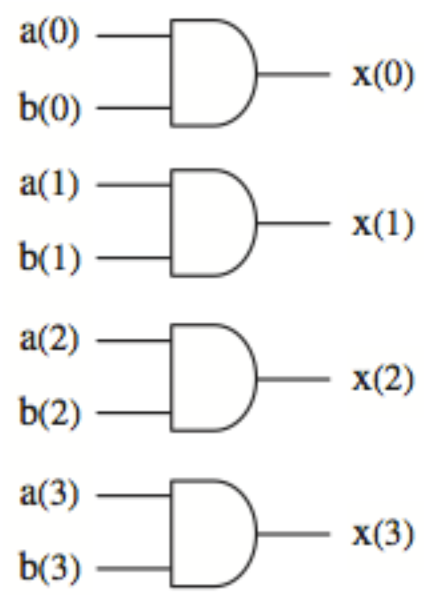
Draw the inferred circuit from each code snippet.

Solution #2- What is the difference between these two implementations

```
-----------------------------
ENTITY and2 IS
   PORT (a, b: IN BIT;
         x: OUT BIT);
END and2;
-----------------------------
ARCHITECTURE and2 OF and2 IS
BEGIN
   x <= a AND b;
END and2;
-----------------------------
```

```
------------------------------------
ENTITY and2 IS
   PORT (a, b: IN BIT_VECTOR (0 TO 3);
         x: OUT BIT_VECTOR (0 TO 3));
END and2;
------------------------------------
ARCHITECTURE and2 OF and2 IS
BEGIN
   x <= a AND b;
END and2;
------------------------------------
```




a(0)
b(0)
x(0)
a(1)
b(1)
x(1)
a(2)
b(2)
x(2)
a(3)
b(3)
x(3)