Concurrent Statements - Signal Assignment

Signal assignment

We have seen the simple signal assignment statement

sig_a <= input_a AND input_b;

VHDL provides both a concurrent and a sequential signal assignment statement. The two statements can have the same syntax, but they differ in how they execute.

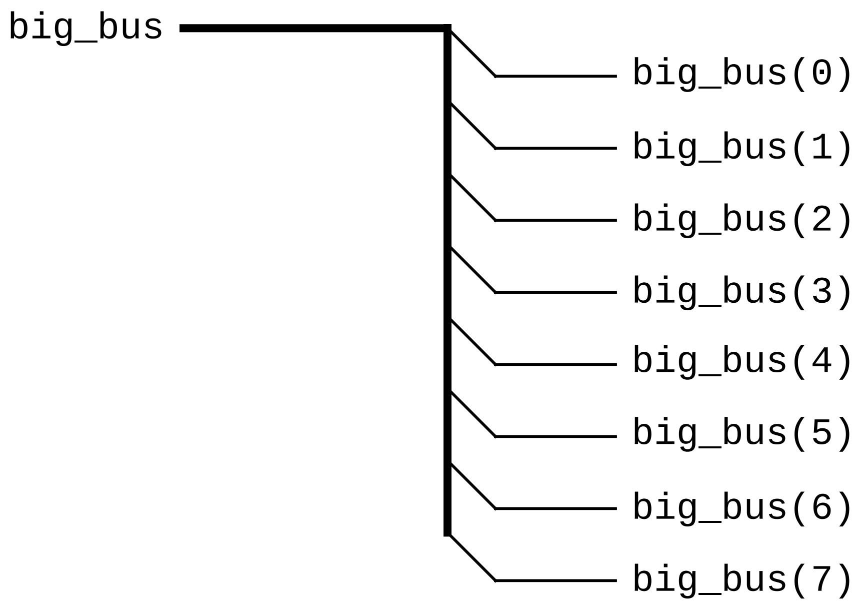
Signal Assignment with Busses

A bus is a collection of wires related in some way by function or clock domain. Examples would be an address bus or data bus.

In VHDL we refer to busses as a vector. For example:

```
--8-bit bus consisting of 8 wires carrying signals of
-- type std_logic
--all these wires may be referred to by the name big_bus
SIGNAL big_bus : STD_LOGIC_VECTOR(7 DOWNTO 0);
```

This creates:



When we define a bus as above, the width of the bus is defined by "7 DOWNTO 0". The position of the MSB is to the left of the DOWNTO keyword. The LSB bit is to the right of DOWNTO.

The usual convention is to use DOWNTO. We will use this convention. UPTO is seldom used.

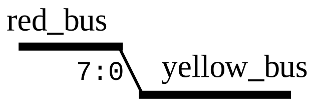
Signal Assignment with Busses (cont.)

Individual bits of a bus may be referred to like this:

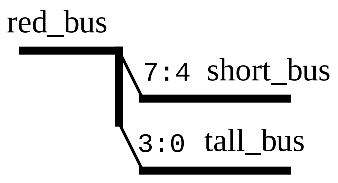
```
SIGNAL one_bit : STD_LOGIC;
SIGNAL big_bus : STD_LOGIC_VECTOR(7 DOWNTO 0);
BEGIN
--wire called one_bit is connected to bit 6 of bus big_bus
one_bit <= big_bus(6); -- bus ripping example
```

Consider the following declarations and how they can be used.

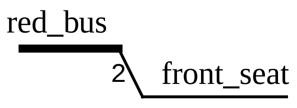
```
SIGNAL back_seat, front_seat: STD_LOGIC;
SIGNAL red_bus, yellow_bus, shift_bus
                            : STD_LOGIC_VECTOR(7 DOWNTO 0 );
SIGNAL short_bus, tall_bus, : STD_LOGIC_VECTOR(3 DOWNTO 0);
```



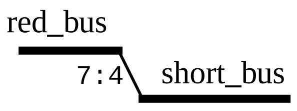
```
red_bus <= yellow_bus; -- connecting same size busses
```



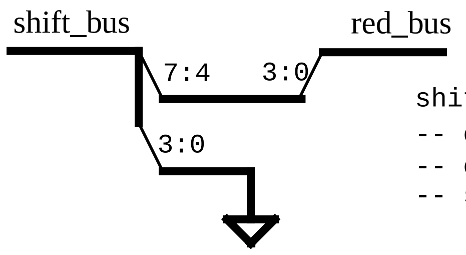
```
red_bus <= short_bus & tall_bus; -- bus concatenation
--“&” is the concatenation operator
-- MSB’s of red_bus come from left most signal
```



```
front_seat <= red_bus(2); -- bus ripping
```



```
short_bus <= red_bus(7 DOWNTO 4); -- bus to bus ripping
```



```
shift_bus <= red_bus(3 DOWNTO 0) & “0000”;
-- one bus created from ripping of one bus and
-- concatenation of signals connected to ground
-- shift bus is red_bus multiplied by 16
```

Bit Vector Usage

As we have seen the in the following examples VHDL has a convenient way to represent busses. A bit string literal allows us to specify the value of a bit vector. For example, the number 227_{10} could be represented as:

Binary format:	B”11111010”	B”1111_1010”
Hexadecimal format:	X”FA”	
Octal format:	O”372”	

The binary format may include underscores to increase readability. The underscores do not effect the value.

Values of bit string literals are inclosed in double quotes. For example: “1101”

Values of bit literals are inclosed in single quotes. For example: ‘Z’