# **Multiple-Bit Wire Naming Conventions**

You can connect multiple-bit wires in your design using any one of the following naming conventions:

- <u>Using Vector Expressions in Multiple-Bit Wire Names</u> on page 110
- Using Vector Expressions in Multiple Signals on page 110
- Evaluating Vector Expressions in Multiple-Bit Wire Names on page 111
- Using Prefix Repeat Operators in Multiple-Bit Wire Names on page 111
- Using Suffix Repeat Operators in Multiple-Bit Wire Names on page 112

#### Using Vector Expressions in Multiple-Bit Wire Names

A multiple-bit wire name can be a bundle, a bus, or a combination of the two. You can improve readability in your designs by shortening multiple-bit wire names.

To indicate that multiple bits of one wire carry similar information, give each bit the same base name. Then add a suffix to each bit name to distinguish the signals.

For example, to show four bits of a wire with a common base name (DATA), but with individual suffixes (<0>, <1>, <2>, and <3>), assign the following names: DATA<0>, DATA<1>, DATA<2>, and DATA<3>.

### **Using Vector Expressions in Multiple Signals**

To name a wire that contains multiple signals with the same base name, specify the base name followed by a vector expression. The vector expression can be

 A list of the individual bit numbers separated by commas and enclosed with angle brackets.

For example, DATA<2,1,0> represents DATA<2>, DATA<1>, and DATA<0>.

A range of numbers containing a lower and upper bound and, optionally, an increment value. The numbers are separated with colons and enclosed with angle brackets.

For example, DATA<1:7:2> represents

baseName <lowerBound : upperBound : [incrValue] >

Use a positive integer for *incrValue*. If you do not specify an increment value, the editor uses 1.

The ordering of the bits in a bus is important when you are connecting the bus to a pin that has a width greater than 1.

#### **Evaluating Vector Expressions in Multiple-Bit Wire Names**

The system evaluates vector expressions in multiple-bit wire names as follows:

- The vector expression DATA<0:3:2> names a 2-bit bus containing DATA<0> and DATA<2>.
- The vector expression DATA<1:3:2> names a 2-bit bus containing DATA<1> and DATA<3>.
- The vector expressions DATA<0:3> and DATA<0:3:1> are equivalent names that generate a 4-bit bus containing DATA<0>, DATA<1>, DATA<2>, and DATA<3>.

If the lower bound is larger than the upper bound, the editor generates the bit numbers in descending order, as follows:

■ The vector expression DATA<2:0> generates a 3-bit bus that contains DATA<2>, DATA<1>, and DATA<0>.

#### Using Prefix Repeat Operators in Multiple-Bit Wire Names

A multiple-bit wire name can be a bundle, a bus, or a combination of the two. You can improve readability in your designs by shortening multiple-bit wire names.

You can repeat a single signal name, a group of signal names, or a vector term any number of times in the wire name by placing a prefix repeat operator <\*n> in front of the name, where n is a positive integer that defines the number of times to repeat each bit in the vector term.

■ Use the prefix repeat operator <\*n> to repeat a single-signal name. The following equivalent wire names both name the same four-bit wire:

```
<*2>A,B,C
A,A,B,C
```

Use the prefix repeat operator < n > and parentheses to repeat a group of signal names. The following two wire names are equivalent:

```
<*2>(A,B),C
A,B,A,B,C
```

■ Use combinations of the prefix repeat operator <\*n> and parentheses to nest parenthetical expressions to any required depth. The editor expands nested expressions from the innermost expression outward. For example, a name with the expression <\*2>(A, <\*2>(X, Y)), B expands to A, X, Y, X, Y, A, X, Y, X, Y, B.

#### Using Suffix Repeat Operators in Multiple-Bit Wire Names

A multiple-bit wire name can be a bundle, a bus, or a combination of the two. You can improve readability in your designs by shortening multiple-bit wire names using vector expressions.

A suffix repeat operator is a number, < n>, after a vector term, where *n* is a positive integer that defines the number of times to repeat each bit in the vector term.

Use the suffix repeat operator < \*n> to repeat each bit in a group of bit names before expanding the vector term. For example, the following three names all describe the same six-bit wire:

```
A<0:2*2>
A<0*2,1*2,2*2>
A<0,0,1,1,2,2>
```

■ Use the suffix repeat operator < \*n> and parentheses to repeat the sequence of bit names. In this case, the vector term is expanded before the bits are repeated. The following names all describe the same six-bit wire:

```
A<(0:2)*2>
A<0:2,0:2>
A<0,1,2,0,1,2>
```

## **Pin Connections and Pin Naming Conventions**

A pin name consists of a string of printable characters. You must observe the <u>rules for</u> <u>reserved characters</u>.

You can name pins in a design

- To declare which nets in the schematic are connected to a higher level of the design hierarchy
- To declare which nets span across sheets in a multisheet schematic

This section describes the following pin connections and naming conventions:

- <u>Hierarchical Pin Names</u> on page 113
- Offsheet Pin Names on page 113
- Bus Pin Names on page 113
- <u>Pin-to-Pin Connections</u> on page 114
- <u>Tapping Pins</u> on page 114

#### Hierarchical Pin Names

If you are designing a multisheet schematic, your sheets contain hierarchical pins and offsheet pins.

Hierarchical pins are pins that also appear on the symbol of the design. The hierarchical pins from each sheet become the hierarchical pins of the multisheet.



#### **Offsheet Pin Names**

Offsheet pins connect signals across the sheets of a multisheet schematic only. Use the same name for the offsheet pins on each sheet.

If you want a signal to appear on a multiple sheet that is also exported by a hierarchical pin, place a hierarchial pin for the first usage and offsheet pins on other sheets. Do not place more than one hierarchical pin with the same name in a multisheet design.



#### **Bus Pin Names**

Use a base name with a vector expression. For example, IN<16:10:2> expands to these four pins: IN<16>, IN<14>, IN<12>, and IN<10>.

#### **Pin-to-Pin Connections**

A direct connection between two instance pins without an intermediate wire is called a pin-topin connection. There are four ways to make pin-to-pin connections:



### **Tapping Pins**

You cannot draw a tap directly from an instance pin.

To tap a multibit connection to an instance pin, you must attach a wire named with a bus name to the instance pin. You can then draw taps from the named bus.

If a schematic pin has a bus name, you can draw a tap directly from the pin using a <u>vector</u> <u>expression</u> to name a wire that intersects with the pin.



The following figure shows how to tap pins in a design.

In the example above, the wire named <7:4> is a tap of the schematic pin named INPUT<7:0> and connects a four-bit bus called INPUT<7:4> to the input of the inverter (an iterated instance).

The wire named <3:0> is also a tap of the INPUT<7:0> pin and connects a four-bit bus called INPUT<3:0> to the input of the NAND2 gate (also an iterated instance).

If you do not apply a name to a net that is attached to a pin, the editor gives the net the same name as the pin. For example, the wire that connects the output of the NAND2 gate to the OUTPUT<0:3> pin is called OUTPUT<0:3>.

## Patchcord Connections and Patchcord Naming Conventions

The *schPatchExpr* connection expression (value) for patchcords has the following form:

src\_vector\_expression = dst\_vector\_expression

