Creating a net expression works in a similar way.

Multiple-Bit Wire Connections

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

■ Tapping Multiple Bits of a Bundle on page 103

■ Tapping Multiple Bits of a Bus on page 105

■ Tapping Wire Intersections by Name on page 107

■ Designating Tap Size and Bit Order on page 107
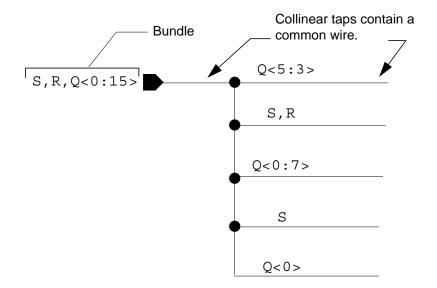
■ System-Generated Net Names on page 108

Tapping Multiple Bits of a Bundle

Tapping extracts specified bits from a named net and propagates the bits along the design. Tapping a subset of signals from a bus or a bundle is called bus tapping.

To create a bus tap, connect a wire that carries the subset of signals to the bus or bundle that contains the signals. You can use one of the many different configurations of wire-to-wire or pin-to-wire connections to create the bus tap. You can also use collinear taps.

Then add the name of the bus tap that specifies the tapped signals. The name depends on whether you tap from a bus or from a bundle.

You tap bits of a bus or bundle when you connect multiple-bit wires to a bus or bundle and name the wires accordingly. For example, in the following illustration, all multiple-bit wires use signal S, signal R, or bus Q.

Collinear taps contain a
Bundle   common wire.

```
                                    Q<5:3>
  S,R,Q<0:15>


                          S,R



                       Q<0:7>



                          S



                       Q<0>
```

A bundle is a wire or pin name consisting of simple names separated by a comma; for example, S,R. A bus is a single name with a vector expression; for example, Q<0:15>.
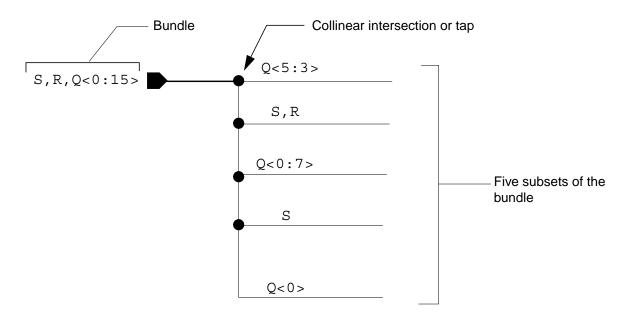
If you want to tap signals from a bundle (that is, a multibit wire whose signals do not have the same base name), you must specify the name of the signals in the wire name to indicate numbered bits.

In the following example, bus Q<0:7> taps eight bits from bus Q<0:15>, which is part of a bundle. In this case, the tap name contains a base name with a vector expression. Wire S,R taps two signals with different names from the bundle. In this case, the wire name is a list of names separated by commas.
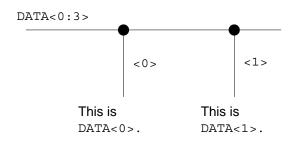
```
  S,R,Q<0:15>


       Q<0:7>          S,R           Q<0>
```

The following example shows the same multiple bits of a bundle with a pin and a collinear
intersection or tap.

Bundle   Collinear intersection or tap

`S,R,Q<0:15>`   `Q<5:3>`

`S,R`

`Q<0:7>`   Five subsets of the
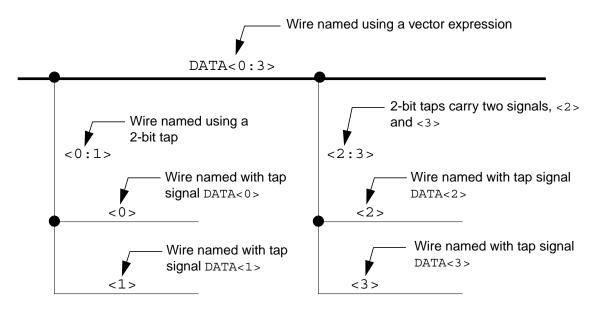bundle

`S`

`Q<0>`

Tapping Multiple Bits of a Bus

A wire named with a <u>vector expression</u> can intersect another wire named with a vector
expression, provided that one of them also intersects a wire named with the full bus name.
This procedure lets you cascade taps without repeating the base name. Any wires or pins of
instances connected to tap wires inherit the name of the tapped signal. This process is called
"cascading" bus taps. If a tap wire carries more than one bit from the bus, the editor can tap
the wire.

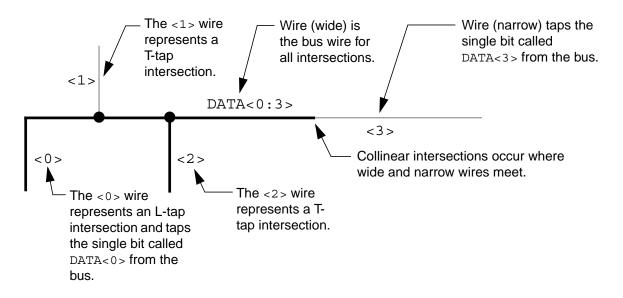For example, to tap signal `DATA<0>` from bus `DATA<0:3>`, specify `<0>` or `0`.

`DATA<0:3>`

`<0>`   `<1>`

This is   This is
`DATA<0>`.   `DATA<1>`.

You can create a geometric configuration when a tap wire carries more than one bit from the bus. The single-bit taps `<0>`, `<1>`, `<2>`, and `<3>` are drawn from the two-bit taps `<0:1>`, `<2:3>` rather than directly from the `DATA<0:3>` bus.

Wire named using a vector expression

`DATA<0:3>`

Wire named using a
2-bit tap

`<0:1>`

2-bit taps carry two signals, `<2>`
and `<3>`

`<2:3>`

Wire named with tap
signal `DATA<0>`

`<0>`

Wire named with tap signal
`DATA<2>`

`<2>`

Wire named with tap
signal `DATA<1>`

`<1>`

Wire named with tap signal
`DATA<3>`

`<3>`

As shown in the above example, if you intersect a bus wire named `DATA<0:7>` with a tap wire named `<0:3>`, you can intersect the tap wire with a second tap wire named `<0:1>`. The name of the first tap wire becomes `DATA<0:3>`. The name of the second tap wire becomes `DATA<0:1>`.

You can also intersect the second tap wire with other tap wires to individually tap bit `<0>`, bit `<1>`, and so on. In general, you can intersect any bus with a tap wire, even if the bus itself is a tap of a larger bus. Taps can cascade down any number of steps.

Tapping Wire Intersections by Name

The following illustration summarizes how to specify bus tap intersections.

The `<1>` wire represents a T-tap intersection.

Wire (wide) is the bus wire for all intersections.

Wire (narrow) taps the single bit called `DATA<3>` from the bus.

`<1>`

`DATA<0:3>`

`<3>`

`<0>`

`<2>`

Collinear intersections occur where wide and narrow wires meet.

The `<0>` wire represents an L-tap intersection and taps the single bit called `DATA<0>` from the bus.

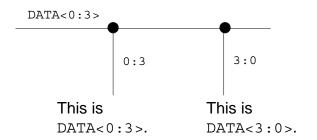The `<2>` wire represents a T-tap intersection.

Any other wires or pins of instances connected to tap wires inherit the name of the tapped signal. This process is called "cascading" bus taps. Any narrow or wide wire connected to the wire named `<3>` is connected to the `DATA<3>` bit of the bus. Any wire connected to the wire named `<2>` is connected to the `DATA<2>` bit of the bus, and so on.

Designating Tap Size and Bit Order

A tap wire can contain only the signals present in the bus, but the total number of bits in a tap wire can exceed the total number of bits in the bus. To make this happen, use one of the repeat operators in the tap expression. For example, you can intersect a 4-bit bus wire named `A<0:3>` with an 8-bit tap wire named `<0:1*2, (0:1)*2>`. The resulting name of the tap wire is `A<0,0,1,1,0,1,0,1>`.

The editor orders bits in a bus just as they appear in the bus name. Similarly, the editor orders bits in a tap as they appear in the tap name.

For example, you can tap the four bits of a bus named DATA<0:3> in order from DATA<0> to DATA<3>. However, a tap wire named DATA<3:0> that intersects with bus DATA<0:3> taps the bits in reverse order, as shown in the following figure.

```
DATA<0:3>
─────────────●───────────────●───────────────
             │0:3            │3:0

         This is         This is
         DATA<0:3>.      DATA<3:0>.
```

System-Generated Net Names

The following items explain how the system determines a name for unnamed wire segments.

■ The editor places all unnamed intersecting wires in the same unnamed net.

■ To determine the name of the unnamed net, the editor chooses a name from among the named nets that intersect with the unnamed net. Named nets are those nets whose names are derived from wire names, schematic pins, or tap names.

■ The chosen name for the unnamed net is the one with the largest number of unique members. For example, a net named A,B,C has three unique members, but a net named A,A has only one unique member.

■ At all wire-wire and wire-pin intersections, the editor verifies that all names present at the intersection are members of the name that contains the most unique members.

For example, in the following illustration

■ The editor places wire segments 3, 5, and 7 in the same net because these segments are intersecting and unnamed

■ This unnamed net is then named A,B,C because it takes the names of all the intersecting named nets. The name A,B,C is the one with the most unique member names; that is, three.

■   The editor then verifies that all intersecting named nets (`B,C,C,B` and `A,A,B,B`) are
    taps of `A,B,C`.

```
  A,B,C                          A,A,B,B
─────────────  1 ──●── 2  ─────────────────

                       │
                       3


   C,B                              B,C
─────────────  4 ──●── 5  ──────────●── 6  ─────────

                                    │
                                    7

                                    │
```