Project Compiler Error Reference

Summary

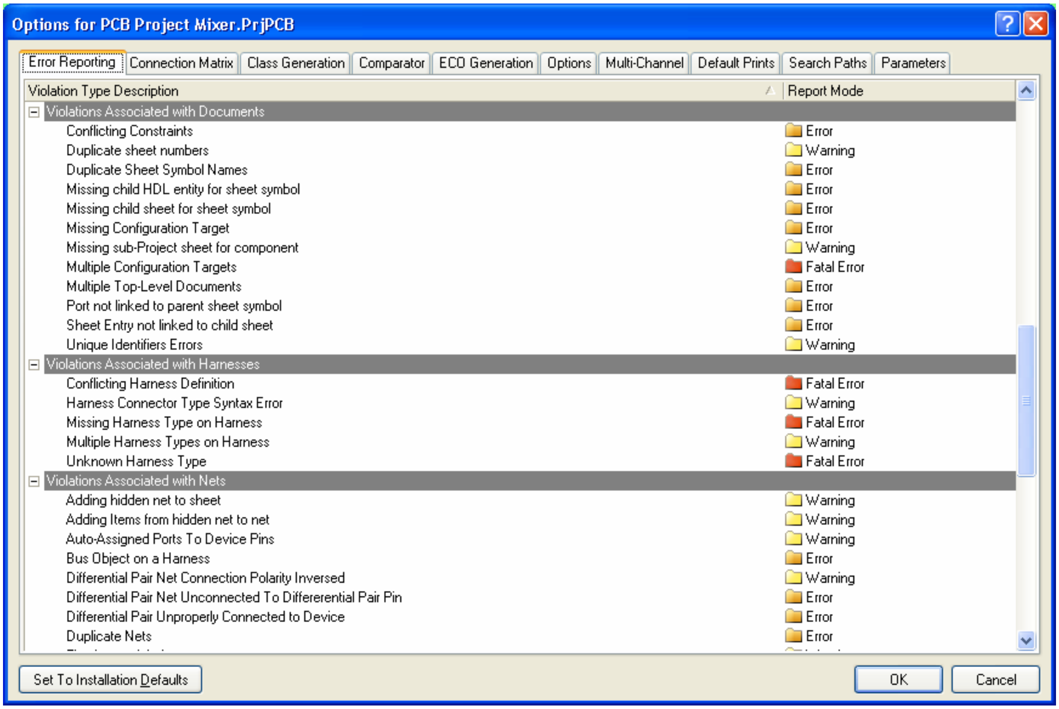
Technical Reference

TR0142 (v1.1) November 03, 2007

This comprehensive reference describes each of the possible electrical and drafting violations that can exist in source documents when compiling a project.

The process of compiling is integral to producing a valid netlist for a project. Connectivity awareness in your schematic diagram can be verified during compilation according to rules that are set up in your **Project Options** (**C » O**) in the **Error Reporting** and **Connection Matrix** tabs. You can customize reporting modes for violations in the **Error Reporting** and **Connection Matrix** tabs. Choose from:

- Fatal Error
- Error
- Warning
- No Report



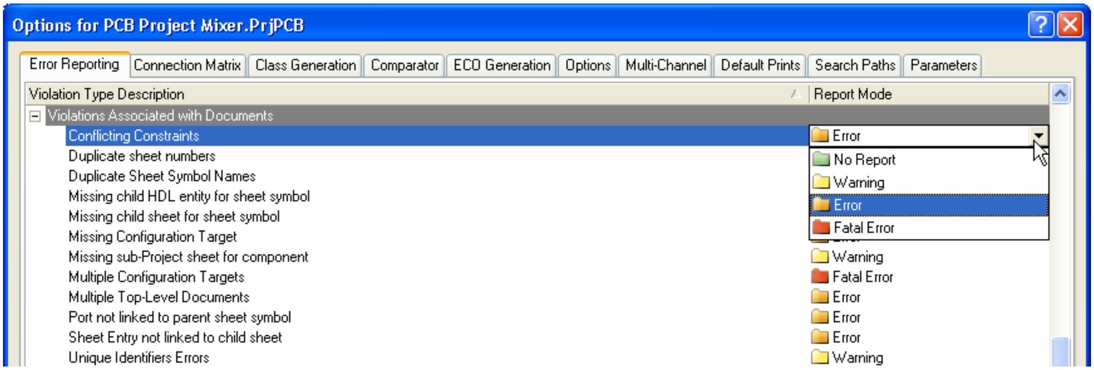
Options for PCB Project Mixer.PrjPCB

Error Reporting | Connection Matrix | Class Generation | Comparator | ECO Generation | Options | Multi-Channel | Default Prints | Search Paths | Parameters

Violation Type Description	Report Mode
Violations Associated with Documents	
Conflicting Constraints	Error
Duplicate sheet numbers	Warning
Duplicate Sheet Symbol Names	Error
Missing child HDL entity for sheet symbol	Error
Missing child sheet for sheet symbol	Error
Missing Configuration Target	Error
Missing sub-Project sheet for component	Warning
Multiple Configuration Targets	Fatal Error
Multiple Top-Level Documents	Error
Port not linked to parent sheet symbol	Error
Sheet Entry not linked to child sheet	Error
Unique Identifiers Errors	Warning
Violations Associated with Harnesses	
Conflicting Harness Definition	Fatal Error
Harness Connector Type Syntax Error	Warning
Missing Harness Type on Harness	Fatal Error
Multiple Harness Types on Harness	Warning
Unknown Harness Type	Fatal Error
Violations Associated with Nets	
Adding hidden net to sheet	Warning
Adding Items from hidden net to net	Warning
Auto-Assigned Ports To Device Pins	Warning
Bus Object on a Harness	Error
Differential Pair Net Connection Polarity Inversed	Warning
Differential Pair Net Unconnected To Differerential Pair Pin	Error
Differential Pair Unproperly Connected to Device	Error
Duplicate Nets	Error

Figure 1. The Error Reporting tab in Project Options

To change the **Report Mode** for a violation in the **Error Reporting** tab, click on the current Report Mode and select an alternative from the dropdown.

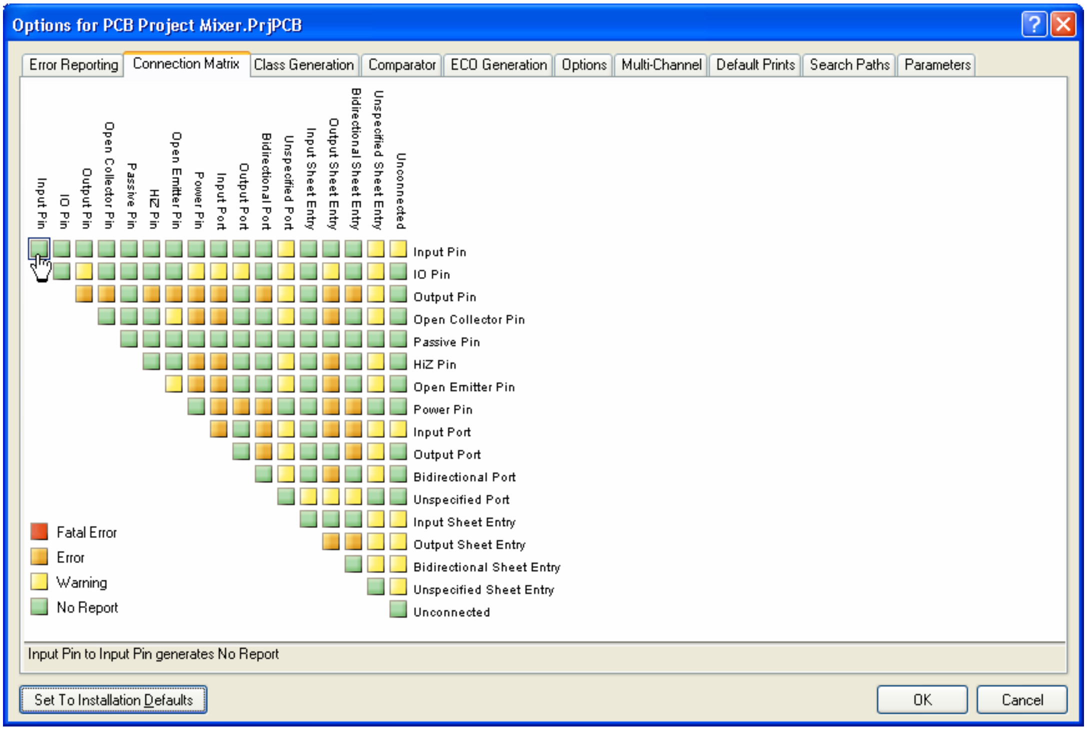


Options for PCB Project Mixer.PrjPCB

Violation Type Description	Report Mode
Violations Associated with Documents	
Conflicting Constraints	Error
Duplicate sheet numbers	
Duplicate Sheet Symbol Names	
Missing child HDL entity for sheet symbol	
Missing child sheet for sheet symbol	
Missing Configuration Target	
Missing sub-Project sheet for component	Warning
Multiple Configuration Targets	Fatal Error
Multiple Top-Level Documents	Error
Port not linked to parent sheet symbol	Error
Sheet Entry not linked to child sheet	Error
Unique Identifiers Errors	Warning

Figure 2.Selecting the Report Mode in the Error Reporting Tab

To change the **Report Mode** for a violation in the **Connection Matrix** tab, click on the existing Report Mode to cycle through the alternatives. When you click on the Report Mode, text is displayed at the bottom of the window to describe the connectivity violation and the Report Mode for the selection.



Options for PCB Project Mixer.PrjPCB

	Input Pin	IO Pin	Output Pin	Open Collector Pin	Passive Pin	HiZ Pin	Open Emitter Pin	Power Pin	Input Port	Output Port	Bidirectional Port	Unspecified Port	Input Sheet Entry	Output Sheet Entry	Bidirectional Sheet Entry	Unspecified Sheet Entry	Unconnected
Input Pin	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟨	🟩	🟩	🟩	🟨	🟨
IO Pin		🟩	🟨	🟩	🟩	🟩	🟩	🟨	🟨	🟨	🟩	🟨	🟩	🟨	🟩	🟨	🟩
Output Pin			🟧	🟧	🟩	🟧	🟧	🟧	🟧	🟩	🟧	🟨	🟩	🟧	🟧	🟨	🟩
Open Collector Pin				🟩	🟩	🟩	🟨	🟧	🟧	🟩	🟩	🟨	🟩	🟧	🟩	🟨	🟩
Passive Pin					🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
HiZ Pin						🟩	🟩	🟧	🟧	🟩	🟩	🟨	🟩	🟧	🟩	🟨	🟩
Open Emitter Pin							🟨	🟧	🟧	🟩	🟩	🟨	🟩	🟧	🟩	🟨	🟩
Power Pin								🟩	🟧	🟧	🟧	🟩	🟩	🟧	🟧	🟩	🟩
Input Port									🟧	🟩	🟧	🟨	🟩	🟧	🟧	🟨	🟨
Output Port										🟩	🟧	🟨	🟩	🟩	🟧	🟨	🟩
Bidirectional Port											🟩	🟨	🟩	🟧	🟩	🟨	🟩
Unspecified Port												🟩	🟨	🟨	🟨	🟩	🟩
Input Sheet Entry													🟩	🟩	🟩	🟨	🟨
Output Sheet Entry														🟧	🟧	🟨	🟨
Bidirectional Sheet Entry															🟩	🟨	🟨
Unspecified Sheet Entry																🟩	🟨
Unconnected																	🟩

🟥 Fatal Error
🟧 Error
🟨 Warning
🟩 No Report

Input Pin to Input Pin generates No Report

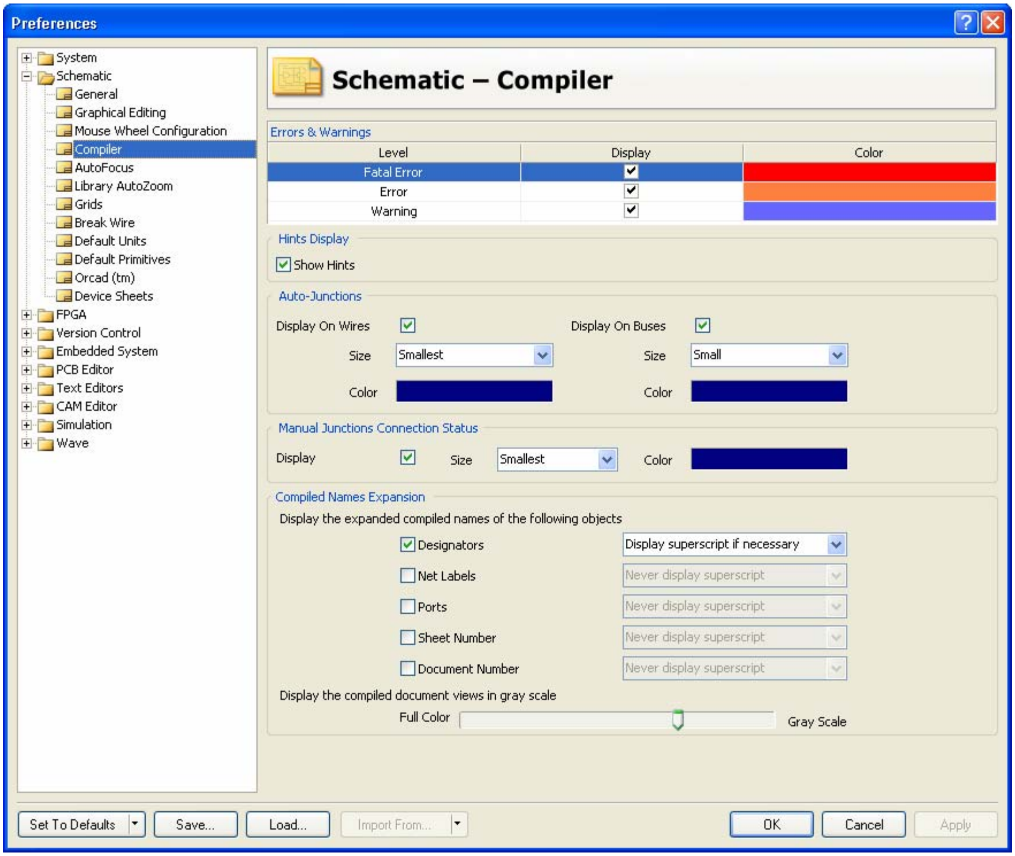
Figure 3.Selecting the Report Mode in the Connection Matrix Tab

Use the **Set to Installation Defaults** button on either the **Error Reporting** or **Connection Matrix** tabs to reset your Reporting Modes back to their default value on installation.

Note that the Violations listed in the **Error Reporting** tab correspond to the *compiler hints* you see in your Schematic Documents when a violation occurs.

To customize the visibility of *compiler hints* and the properties of the wavy underline for errors and warnings:

- Select the **DXP » Preferences** command which brings up the *Preferences* dialog
- Navigate to the **Compiler** tab under the **Schematic** folder
- Choose to display **Fatal Errors**, **Errors** and **Warnings** by enabling the **Display** checkbox
- If you choose to display the errors and/or warnings, a wavy underline will be displayed under your offending object in the color specified in your preferences. You can customize the color of the wavy underline by clicking on the respective **Color** field and selecting a new color in the **Choose Color** dialog
- Enable the **Show Hints** checkbox to display all hints in your schematic, including *compiler hints* which correspond to the violations listed in the **Error Reporting** tab. Run your mouse over the offending object (denoted by a colored wavy underline) to see the *compiler hints*. Note that *compiler hints* are only displayed if the **Display** field is also enabled for each corresponding error or warning.



Schematic – Compiler

Errors & Warnings

Level	Display	Color
Fatal Error	☑	🟥
Error	☑	🟧
Warning	☑	🟦

Hints Display

☑ Show Hints

Auto-Junctions

Display On Wires ☑ Size Smallest Color 🟦

Display On Buses ☑ Size Small Color 🟦

Manual Junctions Connection Status

Display ☑ Size Smallest Color 🟦

Compiled Names Expansion

Display the expanded compiled names of the following objects

☑ Designators — Display superscript if necessary
☐ Net Labels — Never display superscript
☐ Ports — Never display superscript
☐ Sheet Number — Never display superscript
☐ Document Number — Never display superscript

Display the compiled document views in gray scale

Full Color — Gray Scale

Figure 4.The Compiler tab in the Preferences dialog

The following sections provide the information for your compiled project including *compiler hints*, a description of the message displayed in the **Messages Panel**, the **Default Report Mode** and a recommendation for resolution for each violation.

Violations Associated with Buses

Arbiter loop in OpenBus document

This compiler hint appears when the IO and MEM ports of a processor in the OpenBus System are linked, through a single Arbiter component, to the same slave memory or I/O peripheral device, effectively forming a loop. The message is displayed in the **Messages** panel in the following format:

`Arbiter` *`ArbiterName`* `is in the loop`,

where

`ArbiterName` is the designator of the offending Arbiter component.

Default Report Mode

Fatal Error

Recommendation

It is nonsensical in a processor-based system to connect a slave device to both the Peripheral I/O and External Memory interfaces of a processor concurrently. Therefore ensure that the IO and MEM ports of a processor in the OpenBus System are linked to their intended (and different) slave devices using distinct interconnects – Interconnect and/or Arbiter components. Typically for a single processor system, an Interconnect component and one or more Arbiter components would be used on the MEM side of the processor, with the processor and any memory-based peripheral I/O devices linked to slave memory through the Arbiter(s). The IO side of the processor would involve use of an Interconnect component only, to which the bank of peripheral I/O devices – to be used by the processor – are linked.

Bus indices out of range

This compiler hint appears when the index of a constituent net connected to a bus lies outside the range specified by the net to which the bus is associated. The message is displayed in the **Messages** panel in the following format:

`Bus index out of range on NetPrefix Index = NetIndex`,

where

`NetPrefix` is the prefix of the constituent net connected to the bus (e.g. A for net A1, connected to a bus associated to net A[0..7])

`NetIndex` is the erroneous index of the constituent net (e.g. net A8 has an index of 8).

Default Report Mode

Warning

Recommendation

Use the *Compile Errors* dialog to quickly cross probe to the net label associated with the offending net and either amend the index of the net so that it lies within the correct range, or rename the net altogether. The latter would be typical if you have named the net by mistake and it is not a constituent of the net transported by the bus object.

Bus range syntax errors

This compiler hint appears when the syntax of the net to which the bus is associated is specified incorrectly. The message is displayed in the **Messages** panel in the following format:

```
Bus range syntax error NetName at Location,
```

where

`NetName` is the name of the parent net to which the offending bus object is associated

`Location` is the X, Y coordinates for the offending bus object's electrical hotspot.

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to quickly cross probe to the offending net identifier (e.g. net label, port, sheet entry, etc) whose bus syntax is defined incorrectly. The correct syntax should appear in one of the following formats:

```
NetName[LowerIndex..UpperIndex]
NetName[UpperIndex..LowerIndex]
```

For example, consider a bus that carries two constituent nets, `A0` and `A1`. The bus syntax in this case would be `A[0..1]` or `A[1..0]`. Examples of incorrect syntax would include:

```
A[0.1]
A[1-0]
A[0,1]
A[..1]
A[0..]
```

Cascaded Interconnects in OpenBus document

This compiler hint appears when the Master port of one Interconnect component is linked to the Slave port of another Interconnect component, thereby forming a cascade of Interconnect components in the OpenBus System. The message is displayed in the **Messages** panel in the following format:

```
Cascaded interconnects InterconnectName1 and InterconnectName2,
```

where

`InterconnectName1` is the designator of the first offending Interconnect component

`InterconnectName2` is the designator of the second offending Interconnect component

Default Report Mode

Fatal Error

Recommendation

Ensure that two Interconnect components are not linked together in your OpenBus System. Typically, there will be just two Interconnect components in a single processor system, one linked to the processor's IO port and the other linked to the processor's MEM port. If you have used more than one Interconnect component on either the slave memory or peripheral I/O sides of the system, simply delete the additional Interconnect component(s) and re-link the desired peripherals accordingly.

Forbidden OpenBus Link

This compiler hint appears when an OpenBus Link has been placed, invalidly, to effect connection of two Interconnect or Arbiter components in a cascaded fashion. Note that when placing links, you will automatically be prevented from linking two Master or two Slave ports together. The message is displayed in the **Messages** panel in the following format:

`Forbidden link between slave port` *`PortName1`* `(`*`ComponentName1`*`) and master port` *`PortName2`* `(`*`ComponentName2`*`)`,

where

`PortName1` is the name of the Slave port to which the offending link is connected (e.g. `s0`)

`ComponentName1` is the designator of the parent Interconnect or Arbiter component to which the Slave port is associated

`PortName2` is the name of the Master port to which the offending link is connected (e.g. `m0`)

`ComponentName2` is the designator of the parent Interconnect or Arbiter component to which the Master port is associated.

Default Report Mode

Fatal Error

Recommendation

Ensure that two Interconnect or Arbiter components are not linked together (cascaded) in your OpenBus System. Typically, there will be just two Interconnect components in a single processor system, one linked to the processor's IO port and the other linked to the processor's MEM port. If you have used more than one Interconnect component on either the slave memory or peripheral I/O sides of the system, simply delete the additional Interconnect component(s) and re-link the desired peripherals accordingly.

Similarly, Arbiter components in an OpenBus System containing a single processor will typically be used to provide shared access to physical memory devices. Depending on the intended use of the memory, the system may contain a single Arbiter – providing access to a single memory or bank of memories between the processor and one or more memory-based peripherals – or several Arbiters – if certain memory-based peripherals are to share access with the processor of one type of memory (e.g. EMAC RAM), and others are to share access with the processor of another type of memory (e.g. Video RAM).

Illegal bus definitions

This compiler hint appears if the object that is connected to the bus is not a netlabel, port, sheet entry, pin, cross sheet connector or a power object.

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to quickly cross probe to the offending object whose bus definition is defined incorrectly.

Illegal Bus Range Values

This compiler hint appears when at least one index in the syntax for a net associated with a bus is negative in value. The message is displayed in the **Messages** panel in the following format:

`Illegal bus range value BusLabel at Location`,

where

`BusLabel` is the defined bus labeling where the illegal value has been detected

`Location` is the X,Y coordinates for the offending bus object's electrical hotspot.

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to quickly cross probe to the offending net object (e.g. net label, port, sheet entry, etc) whose bus syntax is defined incorrectly. The correct syntax should appear in one of the following formats:

`NetName[LowerIndex..UpperIndex]`

`NetName[UpperIndex..LowerIndex]`

`LowerIndex` and `UpperIndex` can be zero or a positive integer, but can not be the same value.

Mismatched bus label ordering

This compiler hint appears when two net identifiers associated with the same bus slice define bus labels with ordering that is not in the same direction (ascending or descending). The message is displayed in the **Messages** panel in the following format:

`Mismatched bus ordering on NetName Low value first and High value first`,

where

`NetName` is the name of the parent net to which the mismatched bus ordering is associated.

Default Report Mode

Warning

Recommendation

Use the *Compile Errors* dialog to quickly trace the affected bus slice and identify the net identifiers (port, net label, sheet entry, etc) whose bus ordering is not consistent. Determine the correct ordering and amend the naming for the erroneous object.

Mismatched bus widths

This compiler hint appears when two net identifiers associated with the same bus slice define bus labels with differing widths. For example, a port, with name `A[0..7]` might be connected to a bus whose attached net label is defined as `A[0..15]`. The message is displayed in the **Messages** panel in the following format:

`Mismatched bus widths on bus section NetName (BusSize1 and BusSize2)`,

where

`NetName` is the name of the parent net to which the mismatched bus objects are associated

`BusSize1` is the width of the first of the offending bus objects

`BusSize2` is the width of the second of the offending bus objects.

Default Report Mode

Warning

Recommendation

Use the *Compile Errors* dialog to quickly trace the affected bus slice and identify the net identifiers (port, net label, sheet entry, etc), the bus label widths of which are not consistent. Determine the correct width and amend the naming for the erroneous object.

Mismatched Bus/Wire object on Wire/Bus

This compiler hint appears when a wire object is incorrectly connected to a bus, or a bus object is incorrectly connected to a wire. For example a sheet entry, A, might be connected to a bus, but the correct bus label syntax (e.g. `A[0..1]`) has not been entered for the sheet entry's name. In effect, the sheet entry is a single signal (or wire) object that is now erroneously connected to a bus. The message is displayed in the **Messages** panel in the following format:

`ObjectIdentifier at Location placed on an ObjectType`,

where

`ObjectIdentifier` represents the mismatched object, which can be either a bus or wire object (e.g. pin, port, power port, sheet entry, net label, off-sheet connector). The identifier will appear in one of the following two formats:

- for a bus - `Bus` *Object Name* (e.g. Bus Net Label GND_BUS[..])
- for a wire - `Wire` *Object Name* (e.g. Wire Port TXD)

`Location` is the X,Y coordinates for the object's electrical hotspot

`ObjectType` is the object on which the offending object has been placed - either a wire or a bus.

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to quickly cross probe to the offending object. To resolve the issue, consider the following:

- Is the connection correct - i.e. should a bus connecting to the object really be a wire and vice-versa.
- Is the object defined correctly. For a bus object, ensure that the object's name is specified using the correct bus syntax - in the form `Name[LowIndex..HighIndex]` or `Name[HighIndex..LowIndex]`. For example a byte-wide data output port might be specified as `DAT_OUT[7..0]`. For a wire object, ensure that the object's name defines a single signal and is not defined using bus syntax.

Mismatching Address/Data Widths of OpenBus Ports

This compiler hint appears when an OpenBus Link has been placed between the Master and Slave ports of two OpenBus components, and those ports have differing widths for their constituent address and/or data buses. The message is displayed in the **Messages** panel in the following format:

```
Address/data width of slave port PortName1 (ComponentName1) doesn't match
address/data width of master port PortName2 (ComponentName2),
```

where

`PortName1` is the name of the Slave port to which the link is connected (e.g. `s0`)

`ComponentName1` is the designator of the parent component to which the Slave port is associated

`PortName2` is the name of the Master port to which the link is connected (e.g. `m0`)

`ComponentName2` is the designator of the parent component to which the Master port is associated.

Default Report Mode

Fatal Error

Recommendation

This situation can arise for example if you connect a peripheral with an 8-bit data bus directly to the IO port of a processor, which has a 32-bit data bus. In such a system, delete the link between the processor and the peripheral, and place an Interconnect component in-between. The Interconnect will handle connection of the different data buses and also streamline the addressing – taking the 24-bit address line from the processor and mapping it to the n-bit address line used to drive the peripheral device.

On the memory side, a single physical memory device of any address width can be connected directly to the processor's MEM port. This port is 32-bits wide for both address and data – if memory is connected with a lower address width, the system will essentially just ignore the upper bits that aren't used.

Mixed generic and numeric bus labeling

This compiler hint appears when two net identifiers (port, net label, sheet entry, etc) connected to the same bus slice differ in their bus syntax - one defines a bus range in numeric format (e.g. `A[0..2]`), while the other defines the range in a generic format (e.g. `A[0..b]`). The message is displayed in the **Messages** panel in the following format:

```
Mismatched generic and numeric bus labeling on NetName Level value first and
Generic,
```

where

`NetName` is the name of the parent net to which the mismatched bus labeling is associated

`Level` depends on the numeric ordering for the net. If ascending (e.g. [0..2]) Level will appear as Low. If descending (e.g. [2..0]) Level will appear as High.

Default Report Mode

Warning

Recommendation

Use the *Compile Errors* dialog to cross probe to the offending objects. Determine which of the objects is erroneous in its bus label specification and change it accordingly.

The use of Generics in bus names is not supported. Ensure that numeric values are used instead.

Violations Associated with Components

Component Implementations with Invalid Pin Mappings

This compiler hint appears when compiling an Integrated Library Package (`*.LibPkg`) and the pin mapping between the schematic component and the linked model is found to be invalid. The message is displayed in the **Messages** panel in one of the following formats:

`ComponentName: Could not find port 'ModelPinNumber' on model 'ModelName' for pin 'ComponentPinNumber'` - PCB model related

`ComponentName: Could not map port 'ModelPinNumber' on model 'ModelName' to a pin` - simulation model related

where

`ComponentName` is the name of the component in the source schematic library

`ModelPinNumber` is the expected designator for the pin/pad that could not be found on the linked model

`ModelName` is the name of the model that is linked to the component

`ComponentPinNumber` is the designator of the pin on the source schematic component to which the erroneous pin of the model is mapped.

Default Report Mode

Error

Recommendation

If the PCB model related error message is displayed, double-click on the entry for the PCB model link to access the *PCB Model* dialog. Once there, click on the **Pin Map** button to access the *Model Map* dialog. In the Component Pin Designator column, find the pin number flagged by the message (`ComponentPinNumber`). The violation arises because the corresponding entry in the Model Pin Designator column points to a pad designator that does not exist in the PCB model. Amend the entry as required. Typically there will be one-to-one mapping, with the designators on both sides the same.

If the simulation model related message is displayed, double-click on the entry for the simulation model link to access the *Sim Model* dialog. Once there, click on the **Port Map** tab. This violation will arise when the model pin is not correctly mapped to a pin of the schematic component. This can happen when the entry for the model pin has been set to a pin that is already mapped, or to `Not Connected`. Amend the entry as required.

Components containing Duplicate Sub-Parts

This compiler hint appears when the same part of a multi-part component instance has been placed more than once in a schematic design. For example you have placed a `74HC32` component, with designator `U9`, but have inadvertently placed two instances of part one of this component, resulting in two instances of `U9A` in the design. The message is displayed in the **Messages** panel in the following format:

`Component ComponentName has duplicate sub-parts at Location1 and Location2,` where

`ComponentName` is the name of the offending component in terms of its designator and library reference

`Location1` is the X,Y coordinates for the first instance of the particular sub-part

`Location2` is the X,Y coordinates for the duplicate instance of the particular sub-part.

Default Report Mode

Error

Recommendation

Change the part number for the offending parts as required. This can be achieved in one of the following ways:

- Access the *Component Properties* dialog for the part and change the part number using the available arrow buttons found in the **Properties** region of the dialog.
- Use the **Increment Part Number** command - available from the main **Edit** menu or from the **Part Actions** sub-menu (when right-clicking over the part). The advantage of launching the command from the **Edit** menu is that you remain in increment mode, allowing you to cycle through the part numbers until you reach the desired one.

Components with Duplicate Pins

This compiler hint appears when two or more pins in a component have the same designator. The message is displayed in the **Messages** panel in the following format:

`Duplicate pins in component Pin Identifier1 and Pin Identifier2`,

where

`Identifier1` is the identifier for the first instance of the duplicated pin, represented by the part designator-pin designator pairing

`Identifier2` is the identifier for the second instance of the duplicated pin, represented by the part designator-pin designator pairing.

Default Report Mode

Warning

Recommendation

Change the designator of the offending pin(s) accordingly, so that each has a unique assignment. Pin designators can be edited from within the Schematic Editor for a component that has already been placed. If the component pins are not locked, you can simply double-click on the pin and edit its designator in the associated *Pin Properties* dialog. Otherwise edit the pin(s) using the *Component Pin Editor* dialog - accessed from the *Component Properties* dialog by clicking the **Edit Pins** button.

Typically, the duplication will reside in the library component, in which case you should edit the pin designator for that component in the source schematic library and then pass the change on to placed instances of the component, using the **Update From Libraries** (Schematic Editor) or **Update**

Schematics (Schematic Library Editor) commands. Both commands are available from the main **Tools** menus of these editors respectively.

Notes

Only one error instance will be listed in the **Messages** panel for each distinct component. A component may well have more than two pins with the same designator, but when investigating the error using the **Compile Errors** panel, only the first two duplicate pins (in alphabetical pin name order) will be listed.

Duplicate Part Designators

This compiler hint will appear when at least two parts across source schematic sheets in a design have the same designator associated to them. The message is displayed in the **Messages** panel in the following format:

`Duplicate Component Designators PartDesignator at Location1 and Location2`,

where

`PartDesignator` is the offending designator

`Location1` is the X,Y coordinates marking the center of the parent part for the first instance of the offending designator

`Location2` is the X,Y coordinates marking the center of the parent part for the second instance of the offending designator.

Default Report Mode

Error

Recommendation

Assign different and unique designators to the duplicates as required. This can be done manually, by editing each offending designator, or through use of the Annotate dialog (**Tools » Annotate**). A number of additional annotation-related commands are also available from the main **Tools** menu in the Schematic Editor.

Errors in Component Model Parameters

This compiler hint appears when compiling an Integrated Library Package (`*.LibPkg`) and there is a problem with one or more pin models defined for a linked signal integrity model. The message is displayed in the **Messages** panel in the following format:

`ComponentName: Pin models could not be found`

where

`ComponentName` is the name of the component in the source schematic library.

Default Report Mode

Error

Recommendation

This violation typically arises when the model name for the specified pin model is entered incorrectly, or that particular model does not exist in any of the signal integrity libraries (located in the `Program`

`Files\Altium Designer 6\Library\SignalIntegrity` folder of the installation). These include default model libraries (contained in the `\base\b_models` and `\base\fpga` sub-folders) and user defined libraries (contained in the `\user\u_models` sub-folder). User defined libraries include pin models defined manually, as well as those added by importing IBIS model files.

Double-click on the entry for the signal integrity model link to access the *Signal Integrity Model* dialog. Check that you are using the correct model(s) in the **Pin Models** region of the dialog and amend as necessary. If you are using pin models other than the defaults, ensure that the models reside in a library in the `Library\Signal Integrity\user\u_models` directory. Manually create them or import them if they do not, from within the *Signal Integrity Model* dialog.

Extra Pin Found in Component Display Mode

This compiler hint appears if an extra pin has been detected in one of the display modes for a part. The message is displayed in the **Messages** panel in the following format:

`Extra Pin Identifier in DisplayMode of part PartName`,

where

`Identifier` is used to identify the pin in question. When compiling a schematic library document, the identifier appears in the format PhysicalComponentName-PinDesignator (e.g. DIP14-15). When compiling the source schematic or project, the identifier appears in the format PartDesignator-PinDesignator (Inferred) (e.g. X1-1 (Inferred)).

`DisplayMode` is the specific graphical representation mode for the part in which the extra pin has been found. A part has a Normal mode and can have up to 255 defined Alternate modes

`PartName` is either the physical component name or the designator for the affected part, depending on whether you are compiling the schematic library document or source schematic sheet/project respectively.

Default Report Mode

Warning

Recommendation

This violation typically arises when an alternate graphical mode is defined for a component and either:

- An extra pin has been added to the display that is not specified in the Normal display mode or
- A pin has been specified with a different Designator and/or Name to a pin specified in the Normal display mode.

Not only must there be an identical number of pins between graphical display modes, the pins must be identical in both Designator and Name.

In the source schematic library, display the offending display mode for the component and delete the extra pin. This can be performed directly on the schematic sheet for a part that has been placed already, but you would typically tackle the problem from within the library, then push the change across (**Tools » Update Schematics**).

Mismatched Hidden Pin Connections

This compiler hint appears when there are two hidden pins in a component that have the same designator and are hidden but are connected to different nets. Typically this scenario is happens when placing multi-part components where a common hidden pin to more than one parts is connected to different nets. The message is displayed in the **Messages** panel in the following format:

```
Mismatched hidden pin connections in Pin ComponentPinNumber and Pin
ComponentPinNumber
```

where

`ComponentPinNumber` is the designator of the pin on the source schematic component of the offending pins.

Default Report Mode

Error

Recommendation

Reassign the offending pin(s) to the correct nets to remedy this message. This is done through the *Component Pin Editor* dialog.

Mismatched Pin Visibility

This compiler hint is related to the power pins (VCC and GND) of a multi-part component. Typically, these pins are associated to part 0, are automatically connected to the VCC and GND nets for the design and are hidden. If, for one of the component parts, you enable visibility of such a pin, it is no longer connected to the target power net and the error will be flagged. The message is displayed in the **Messages** panel in the following format:

```
Pin is visible in one sub-part and hidden in another sub-part
```

Default Report Mode

Error

Recommendation

Either disable display of the offending power pin(s) in the workspace or, if keeping the pins displayed, ensure that a VCC and/or GND power port object is attached to the pin(s) accordingly.

Missing Component Models

This compiler hint appears when compiling an Integrated Library Package (`*.LibPkg`) and a linked model for a component in the source schematic library could not be found. When the linked model is a footprint model, simulation model or PCB3D model, the message is displayed in the **Messages** panel in one of the following formats:

`ComponentName: Could not find 'ModelName'` - when the model search scope is `Any`.

`ComponentName: Could not find 'ModelName'in 'LibraryName'` - when the model search scope is `Library Name`.

`ComponentName: Could not find 'ModelName'in 'Path'` - when the model search scope is `Library Path`.

where

`ComponentName` is the name of the component in the source schematic library.

`ModelName` is the name of the footprint, PCB3D or simulation model that is linked to the source component and which could not be found.

`LibraryName` is the name of the library/model file specified to contain the linked model.

`Path` is the absolute path to a library/model file specified to contain the linked model.

When the linked model is a signal integrity model, the message is displayed in the **Messages** panel in the following format:

`ComponentName: Could not find 'GenericEntity'in 'Path'`

where

`ComponentName` is the name of the component in the source schematic library.

`Path` is the absolute path to a library/model

Default Report Mode

Warning

Recommendation

When the problem is a linked footprint, simulation or PCB3D model

This issue is typically caused by one of the following scenarios:

- The model name is incorrectly specified when defining the model link
- The linked model does not reside in the specified library/model file
- The library/model file containing the linked model has been moved or deleted.

The first port of call in resolving this violation is the associated setup dialog for the model type you are linking to - the *PCB Model* dialog, the *Sim Model* dialog, or the *PCB3D Model Libraries* dialog. In each case, check and ensure:

- The name of the model to which you are linking is correct and
- The correct option is used to locate the library/model file in which that model resides.

The format of the displayed error message depends on the search scope you have enabled when locating the model, and can be of great help when tracking down the problem with the model link:

- If the model could not be found in a specified path (search scope: `Library Path`), ensure that the library/model file you have specified actually exists at that location and also check the library/model file to see if the model with the specified name exists within.
- If the model could not be found in a specified library/model file (search scope: `Library Name`), ensure that the library/model file has been added to the Available Libraries list (Project Libraries, Installed Libraries, Project Search Paths). Also check to make sure the library/model file contains the model with the same name specified in the link.

- If the model could simply not be found (search scope: `Any`), ensure that a library/model file - containing a model with the same name as that specified in the link - has been added to the Available Libraries list.

When the problem is a linked signal integrity model

Typically caused when the type of signal integrity model (e.g., bit, diode, IC) is not specified, this is resolved in the associated setup dialog for signal integrity models. The easiest way to access this is through the *Component Properties* dialog where both the signal integrity model type and pin models can be specified. Check that you are using the correct model(s) in the **Models** region of the *Component Properties* dialog and amend if necessary. The **Add** and **Edit** buttons can be used to create a new model or modify the existing user models without the need for having to edit them directly. You can then launch the *Signal Integrity Model* dialog by double-clicking on the entry for **Signal Integrity** Type where the **Update Ibis File** button allows pins models to be imported from an Ibis model file.

Missing Pin Found in Component Display Mode

This compiler hint appears if a pin is found to be missing in one of the display modes for a part. The message is displayed in the **Messages** panel in the following format:

`Missing Pin Identifier in DisplayMode of part PartName`,

where

`Identifier` is used to identify the pin in question. The identifier appears in the format PartLibraryReference-Pin Designator (e.g. DIP14-8)

`DisplayMode` is the specific graphical representation mode for the part in which the missing pin has been found. A part has a Normal mode and can have up to 255 defined Alternate modes

`PartName` is the library reference for the affected part.

Default Report Mode

Warning

Recommendation

This violation typically arises when an alternate graphical mode is defined for a component, but not all pins specified in the Normal mode have been specified for the Alternate. Not only must there be an identical number of pins between graphical display modes, the pins must be identical in both Designator and Name.

In the source schematic library, copy the missing pins from an existing display mode into the offending display mode for the component. This can be performed directly on the schematic sheet for a part that has been placed already, but you would typically tackle the problem from within the library, then push the change across (**Tools » Update Schematics**).

Sheet Symbol with Duplicate Entries

This compiler hint appears if a sheet symbol contains two sheet entries possessing the same name. The message is displayed in the **Messages** panel in the following format:

`Sheet Symbol with duplicate entries Sheet Entry Identifier at Location1 and Location2`,

where

`Identifier` is used to represent the offending sheet entry. The identifier appears in the format `SheetSymbolName-SheetEntryName(SheetEntryIOType)`

`Location1` is the X,Y coordinates for the first instance of the particular sheet entry

`Location2` is the X,Y coordinates for the duplicate instance of the sheet entry.

Default Report Mode

Error

Recommendation

Change the name of the offending sheet entry object as required. If in-place editing is enabled on the **Schematic - General** page of the *Preferences* dialog (**DXP » Preferences**), simply edit the name in-situ. Alternatively, double-click on the offending sheet entry and edit the **Name** field in the corresponding *Sheet Entry* dialog.

Un-Designated Parts Requiring Annotation

This compiler hint appears when a component in the design is found to have a default designator (with a ? suffix) - either it has yet to be annotated or the designator has been reset. The message is displayed in the **Messages** panel in the following format:

`Un-Designated Part PartDesignator`,

where

`PartDesignator` is the default designator for the un-designated part (e.g. U?, D?, C?, etc).

Default Report Mode

Warning

Recommendation

Assign a unique designator to the offending component as required. This can be done manually, by editing the designator, or through use of the Annotate dialog (**Tools » Annotate**). A number of additional annotation-related commands are also available from the main **Tools** menu in the Schematic Editor.

Notes

Only one error instance will be listed in the Messages panel for each distinct designator type (C?, U?, S?, etc). Multiple errors may exist.

Unused Sub-Part in Component

This compiler hint appears when a part of a multi-part component instance has not been used within the design. For example, three out of four parts for an instance of a `74HC32` component may have been placed and wired and the fourth has not. The message is displayed in the **Messages** panel in the following format:

`Component Identifier has unused sub-part (PartNumber)`,

where

`Identifier` is the parent component, represented using the format Designator Library Reference (e.g. U11 74HC32)

PartNumber is an integer used to indicate which specific part is not being used (e.g. 1 represents part A, 2 represents part B, and so on).

Default Report Mode

Warning

Recommendation

Place the unused part and connect its inputs to ground. To ensure the same root designator, simply copy an existing part for that component's instance and, after pasting, increment its part number accordingly.

Violations Associated with Configuration Constraints

Constraint Board Not Found in Configuration

This compiler hint appears when there is a mismatch detected between the constraint record declaring a board instance (NanoBoard, Daughter Board, Peripheral Board) and the constraint record in the board-level constraint file targeting that instance. The message is displayed in the **Messages** panel in the following format:

`Unable to create board instance from constraint: ConstraintRecord`,

where

`ConstraintRecord` is the specific record relating to the offending board declaration.

Default Report Mode

Warning

Recommendation

Use the *Compile Errors* dialog to quickly cross probe to the constraint file containing the record that declares the board. This record is typically contained within a separate constraint file that declares not only the boards used, but also the connector mappings - how the Daughter Board and/or Peripheral Board(s) are plugged into the NanoBoard. This is referred to as the Mapping constraint file, and is generated for you when using the suto-configuration feature. The record, which is also displayed as part of the violation message, appears in the form:

```
Record=Constraint | TargetKind=PCBInstance | TargetId=IDEntry |
PCB=InstanceName
```

In the underlying board-level constraint file (for the NanoBoard, Daughter Board, or Peripheral Board), the following record is used to 'point to' the declared board instance:

```
Record=Constraint | TargetKind=PCB | TargetId=InstanceName
```

To resolve this violation, ensure that the `InstanceName` entry for both of these constraint file records is identical.

Constraint Configuration Has Duplicate Board Instance

This compiler hint appears when two constraint records declaring the same board instance (NanoBoard, Daughter Board, and Peripheral Board) have been detected within the associated constraint files of a configuration. The message is displayed in the **Messages** panel in the following format:

`Board instance with the same name found: ConstraintRecord`,

where

`ConstraintRecord` is the specific record relating to the offending board declaration.

Default Report Mode

Warning

Recommendation

Typically, a particular board (NanoBoard, Daughter Board, and Peripheral Board) will be declared within a dedicated mapping constraint file (this file is produced for you if you have used the auto-configuration feature). Use the *Compile Errors* dialog to quickly cross probe to a constraint file containing an entry for the duplicate board declaration. The record, which appears as part of the violation message, is in the form:

```
Record=Constraint | TargetKind=PCBInstance | TargetId=IDEntry |
PCB=InstanceName
```

Delete any duplicate instance from the file.

If the duplicate record exists in another constraint file, you will need to open each constraint file associated to the configuration, search for the offending record, and delete it. If separate constraint files have been used to declare boards for different assemblies (e.g. one file for an Altera assembly, one for a Xilinx assembly and another for peripheral boards), check in these files to begin with.

Constraint Connector Creation Failed in Configuration

This compiler hint appears when a specified connector mapping constraint is unable to be created for a defined configuration. The message is displayed in the **Messages** panel in the following format:

```
Unable to create board connector map from constraint: ConstraintRecord,
```

where

`ConstraintRecord` is the specific record relating to the offending connector mapping declaration.

Default Report Mode

Warning

Recommendation

Use the *Compile Errors* dialog to quickly cross probe to the constraint file containing the connector mapping record that is failing. The record, which appears as part of the violation message, is in the form:

```
Record=Constraint | TargetKind=ConnectorMap | TargetId=SourceBoardConnector
| ConnectTo=DestinationBoardConnector
```

The connector mapping constraints will typically be specified in a separate mapping constraint file. For example, the mapping between connectors on a daughter board and the NanoBoard, and the mapping between connectors on peripheral boards and the NanoBoard. Declaration of the connectors themselves is usually made within the board-level constraint file, where constraints targeting ports and/or connections are defined.

To resolve this violation, consider the following:

- The *`SourceBoardConnector`* entry should appear in the format *`SourceBoardId.ConnectorId`*. The *`SourceBoardId`* must be the value assigned to the `TargetId` field for the constraint record declaring the board instance. The *`ConnectorId`* must be the value assigned to the `TargetId` field for the constraint record declaring the connector on the source board. For example `DB30_04.HDR_T` defines the connector HDR_T on the source board, whose declared PCB instance has been assigned the Id DB30.04 (a Xilinx Spartan 3 Daughter

Board). Depending on the source board (Daughter Board or Peripheral Board) you will need to check the relevant constraint files to ensure the Ids are matching.

- The *DestinationBoardConnector* entry should appear in the format *TargetBoardId*.*ConnectorId*. The *TargetBoardId* must be the value assigned to the TargetId field for the constraint record declaring the board instance. The target board will be the NanoBoard. The *ConnectorId* must be the value assigned to the TargetId field for the constraint record declaring the connector on the target board. For example NB2DSK01_07.HDR_T defines the connector HDR_T on the NanoBoard, whose declared PCB instance has been assigned the Id NB2DSK01.07.

Constraint Port Without Pin in Configuration

This compiler hint relates to the constraint file dealing with port-to-connector mappings for a Peripheral Board. It appears when the record for a port specifies connection to an invalid pin (one that does not exist on the Peripheral Board's connector). The message is displayed in the **Messages** panel in the following format:

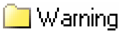
```
Configuration ConfigurationName has port in constraint file without any pin
mapped: PortName,
```

where

ConfigurationName is the name of the configuration to which the offending port's parent constraint file is associated.

PortName is the name of the offending port.

Default Report Mode



Warning

Recommendation

Use the *Compile Errors* dialog to quickly cross probe to the constraint file containing the problematic port definition. Locate the constraint record for the port - the PortName entry in the violation message corresponds to the assigned value for the record's TargetId field. Where the connection has been explicitly defined, the value entered for the record's ConnectTo field must be in the form:

ConnectorName-PinNumber,

where

ConnectorName must be the same name as that entered when declaring the connector. The record declaring the connector will be located previously within the same constraint file and appears in the form: Record=Constraint | TargetKind=Connector | TargetId=ConnectorName.

PinNumber must be a valid (existing) pin on the connector.

Change the ConnectorName prefix and/or the PinNumber, as required, to satisfy these conditions.

Violations Associated with Documents

Conflicting Constraints

This compiler hint appears when a configuration contains at least two constraints targeting the same top-level port in an FPGA design, and those constraints are mapping the port to different pins of the target physical device. The message is displayed in the **Messages** panel in the following format:

```
Port PortName FPGA_PINNUM Constraints Pin1 And Pin2 Conflict in
ConfigurationName,
```

where

`PortName` is the name of the top-level port in the design. In an associated constraint file, this entry is the value assigned to the TargetId field of the corresponding constraint record

`Pin1` is the value assigned to the FPGA_PINNUM field for the targeted port in a constraint file associated to the configuration

`Pin2` is the value assigned to the FPGA_PINNUM field for the same targeted port in an additional constraint file associated to the configuration

`ConfigurationName` is the name of the offending configuration.

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to cross probe quickly to the offending constraints. Decide which of the duplicate constraints are redundant to the design and delete them accordingly. The pin number assignments for ports will typically be kept within a single constraint file. If a duplicate constraint resides in another constraint file, it is highly likely to be an erroneous entry.

If the duplicate constraint entries reside in the main constrain file (for pin mappings) and you are unsure which constraint is the right one to keep, you can simply:

- Delete all the Port-related constraints in the file
- Run the **Design » Import Port Constraints from Project** command - to add a constraint for each port in the FPGA project.
- Run the Place and Route tools (Build stage in the associated Process Flow) and then import the pin assignments back into the constraint file.

Duplicate Sheet Symbol Names

This compiler hint appears when at least two sheet symbols on the same schematic sheet are detected to have the same designator. The message is displayed in the **Messages** panel in the following format:

```
Duplicate Sheet Symbol Names Sheet Symbol SheetSymbolDesignator at Location1
and Location2,
```

where

`SheetSymbolDesignator` is the offending designator

`Location1` is the X,Y coordinates marking the top-left corner of the parent sheet symbol for the first instance of the offending designator

`Location2` is the X,Y coordinates marking the top-left corner of the parent sheet symbol for the second instance of the offending designator.

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to quickly cross probe to the offending sheet symbols. Amend the designator of one of the sheet symbols as required, in order to keep the designators distinct and unique.

Notes

This violation type is checked only for source schematic documents that are part of an FPGA (`*.PrjFpg`) or Core (`*.PrjCor`) project.

Missing Child HDL Entity for Sheet Symbol

This compiler hint appears when the link between a sheet symbol and a target HDL sub document is invalid, specifically due to mismatched Entity Parameter Properties. This can occur when:

- A sheet symbol has been placed manually but the `VHDLEntity` parameter has been incorrectly specified
- You have changed the value of the `Entity` parameter in the target HDL sub document and you have not updated the `VHDLEntity` parameter on your sheet symbol.

The message is displayed in the **Messages** panel in the following format:

```
Missing HDL Entity: VHDLEntityValue in SymbolFileName in Symbol
SymbolDesignator
```

where

`VHDLEntityValue` is the parameter value specified for the `VHDLEntity` on the parent sheet symbol

`SymbolFileName` is the current entry for the parent sheet symbol's Filename field.

`SymbolDesignator` is the designator of the parent sheet symbol.

Default Report Mode

Error

Recommendation

Check the sheet symbol's `VHDLEntity` parameter and ensure that it matches the value in the HDL sub document.

Missing Child Sheet for Sheet Symbol

This compiler hint appears when the link between a sheet symbol and a target schematic, OpenBus System, or HDL sub-document is invalid. This can occur when:

- A sheet symbol has been placed manually but no sub-level document reference has been entered into the symbol's **Filename** field.
- The document reference in the symbol's **Filename** field has been entered incorrectly - effectively targeting a document that does not exist.
- The referenced target document has been removed from the project or deleted.

The message is displayed in the **Messages** panel in the following format:

`Missing child-sheet in SymbolFileName in Symbol SymbolDesignator`,

where

`SymbolFileName` is the current entry for the parent sheet symbol's Filename field.

`SymbolDesignator` is the designator of the parent sheet symbol.

Default Report Mode

Error

Recommendation

Check the entry in the sheet symbol's **Filename** field. If the required target document already exists, ensure that the document name (including extension) is entered correctly into the field. If the target document has been removed from the project and you have access to it, add it back in to the project. If the target document is a schematic or HDL file and it does not exist, simply right-click on the symbol and choose one of the following commands from the **Sheet Symbol Actions** sub-menu - depending on the type of target document required:

- Create Sheet From Symbol
- Create VHDL File From Symbol
- Create Verilog File From Symbol

If the target document is an OpenBus System document and it does not exist, you will need to create this document, adding it to your project.

Notes

This error is also generated when Device Sheet Symbols have been placed but the target Device Sheet cannot be found.

Missing Configuration Target

This compiler hint appears when a configuration does not target a physical device. The message is displayed in the **Messages** panel in the following format:

`Configuration ConfigurationName does not target any device`,

where

`ConfigurationName` is the name of the offending configuration.

Default Report Mode

Error

Recommendation

Access the *Configuration Manager* dialog for the FPGA project (**Project » Configuration Manager**). For the offending configuration, check which constraint files have been associated. Quite often, the issue arises from a constraint file not having been associated where it should be. In such a case, simply associate that constraint file to the configuration.

If the correct constraint files have been associated to the configuration, then the problem lies within the constraint files. Typically, the target device will be declared in only one of these files - the one containing the pin mappings from the ports of the FPGA project to the physical device pins. Open this constraint file and check for the following record:

```
Record=Constraint | TargetKind=Part | TargetId=
```

Add it if it does not exist, ensuring that the physical device name is also entered (e.g. `TargetId=XC2S300E-6PQ208C`).

Multiple Configuration Targets

This compiler hint appears when a configuration targets at least two different physical devices. The message is displayed in the **Messages** panel in the following format:

```
Configuration ConfigurationName targets conflicting devices: Device1 and
Device2,
```

where

`ConfigurationName` is the name of the offending configuration

Device1 is a device found to be targeted in a constraint file associated to the configuration

Device2 is a different device found to be targeted in an additional constraint file associated to the configuration.

Default Report Mode

Fatal Error

Recommendation

Access the *Configuration Manager* dialog for the FPGA project (**Project » Configuration Manager**). For the offending configuration, check which constraint files have been associated. Quite often, the issue arises from a constraint file having been associated where it should not be. In such a case, simply disassociate that constraint file from the configuration.

If the correct constraint files have been associated to the configuration, then the problem lies within the constraint files. Typically, the target device will be declared in only one of these files. Open each constraint file and check for the following record:

```
Record=Constraint | TargetKind=Part | TargetId=
```

If the record exists more than once and targets different devices, keep the entry for the correct device and delete all others.

Multiple Top-Level Documents

This compiler hint appears in hierarchical designs, where two or more schematic sheets are at the top-level of the structure. The message is displayed in the **Messages** panel in the following format:

`Multiple top level documents: SheetName has been used`,

where

`SheetName` is the name of the schematic document currently being used as the top-level sheet.

Default Report Mode

Error

Recommendation

This issue typically arises due to the sheet symbol on the true top sheet not targeting the intended sub-sheet correctly. To resolve this issue, first determine which schematic sheet is the intended sub-sheet. Check to see if a sheet symbol has been placed for the intended sub-sheet on the top-level schematic:

- If a sheet symbol does not exist, create it - either by manual placement or by using the **Create Sheet Symbol From Sheet Or HDL** command (available from the main **Design** menu).
- If the sheet symbol exists, check the symbol's **Filename** field and ensure that it references the sub-sheet.

Upon recompiling, the hierarchy will be resolved and the error will disappear from the **Messages** panel.

Port not Linked to Parent Sheet Symbol

This compiler hint appears when a port on a child sheet is found not to be matched with a sheet entry on the parent sheet symbol. All sheet entries in the parent sheet symbol must be synchronized (matched) to corresponding ports on the child sheet. The message is displayed in the **Messages** panel in the following format:

`Port PortName not matched to Sheet-Entry at Location`,

where

`PortName` is the name of the port on the child sheet

`Location` is the X,Y coordinates for the port's electrical hotspot.

Default Report Mode

Error

Recommendation

This issue can arise for a number of reasons:

- The corresponding sheet entry for the port does not exist
- The corresponding sheet entry for the port exists but with a different name
- The corresponding sheet entry for the port exists but with a different I/O Type.

Use the *Compile Errors* dialog to cross probe to the port in question, then **Ctrl**+double-click on the port to ascend to the parent sheet symbol. Right-click on the sheet symbol and choose **Sheet Symbol**

Actions » Synchronize Sheet Entries and Ports from the menu that appears. This will give you access to the *Synchronize Ports To Sheet Entries* dialog for that sheet symbol.

Use the dialog to match the port in question to the required sheet entry. If the sheet entry does not exist, you can create it directly from the dialog. Where the sheet entry exists but name and/or I/O Type differ you can determine, as part of the match, whether the name and I/O Type to be used comes from the port or the sheet entry.

For more detailed information, refer to the area for *Synchronizing Sheet Entries and Ports*, in the Sheet Symbol section of the Schematic Editor and Object Reference.

Notes

When the sheet entry and port exist but have different names and/or I/O Types, there will be a corresponding error message stating that the sheet entry is not matched to a port. Synchronizing the sheet entry with the port will clear both errors.

Sheet Entry not linked to Child Sheet

This compiler hint appears when a sheet entry is found not to be matched with a port on the child sheet referenced by the parent sheet symbol. All sheet entries in the parent sheet symbol must be synchronized (matched) to corresponding ports on the child sheet. The message is displayed in the **Messages** panel in the following format:

```
Sheet-Entry SheetEntryName not matched to Port at Location,
```

where

`SheetEntryName` is the name of the sheet entry associated with the parent sheet symbol

`Location` is the X,Y coordinates for the sheet entry's electrical hotspot.

Default Report Mode

Error

Recommendation

This issue can arise for a number of reasons:

- The corresponding port for the sheet entry does not exist
- The corresponding port for the sheet entry exists but with a different name
- The corresponding port for the sheet entry exists but with a different I/O Type.

Use the *Compile Errors* dialog to cross probe to the sheet entry in question. Right-click on the parent sheet symbol and choose **Sheet Symbol Actions » Synchronize Sheet Entries and Ports** from the menu that appears. This will give you access to the *Synchronize Ports To Sheet Entries* dialog for that sheet symbol.

Use the dialog to match the sheet entry in question to the required port. If the port does not exist, you can create it directly from the dialog. Where the port exists but name and/or I/O Type differ you can determine, as part of the match, whether the name and I/O Type to be used comes from the sheet entry or the port.

For more detailed information, refer to the area for *Synchronizing Sheet Entries and Ports*, in the Sheet Symbol section of the Schematic Editor and Object Reference.

Notes

When the sheet entry and port exist but have different names and/or I/O Types, there will be a corresponding error message stating that the port is not matched to a sheet entry. Synchronizing the sheet entry with the port will clear both errors.

Unique Identifiers Errors

This compiler hint will appear when at least two parts, two sheet symbols, or a combination of these objects - across source schematic sheets in a design - have the same Unique ID associated to them. The message is displayed in the **Messages** panel in the following format:

`Unique Identifiers Errors: found at Location1 and Location2`,

where

`Location1` is the X,Y coordinates for the first object found with a Unique ID error

`Location2` is the X,Y coordinates for the second object found with a Unique ID error.

Default Report Mode

Warning

Recommendation

Reset the Unique ID's for the offending objects as required. This can be done at the individual object level, by accessing the object's associated properties dialog and clicking the **Reset** button next to the field for the Unique ID. Alternatively, you can reset Unique IDs on a more global level using the *Reset Part/Sheet Symbol Unique IDs* dialog (**Tools » Convert » Reset Component Unique IDs**). The dialog allows you to reset IDs for the active document, all source schematics in the active project, or all open schematics (regardless of the project to which they belong).

Violations Associated with Harnesses

Conflicting Harness Definition

This compiler hint appears when there are conflicting Harness Entries for the same Harness Type either at a graphical or textual level (in the Harness Definition File). The message is displayed in the **Messages** panel in the following format:

```
Conflicting Harness Definition for HarnessType
```

where

`HarnessType` is the current conflicting Harness Type.

Default Report Mode

Fatal Error

Recommendation

Use the *Compile Errors* dialog to cross probe to the offending objects (Harness Definition Files and Harness Connectors) within the Signal Harness. Update the Harness Definition File to reflect the changes or remove the offending Harness Entry.

Harness Connector Type Syntax Error

This compiler hint appears when either the Harness Type has invalid characters (such as [ ] { } . :) or if the Harness Type is blank.

If the Harness Type has invalid characters, this message is displayed in the **Messages** panel in the following format:

```
Harness Connector Type HarnessType should not contain these characters []
{}. :
```

where

`HarnessType` is the current conflicting Harness Type.

If the Harness Type is blank, this message is displayed in the **Messages** panel in the following format:

```
Harness Connector Type cannot be Blank.
```

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to cross probe to the offending Harness Connector. Double-click on the Harness Connector object and specify a valid Harness Type. Ensure the Harness Connector Type does not contain invalid characters.

Missing Harness Type on Harness

This compiler hint appears when a Signal Harness connecting Sheet Entries or connecting a Sheet Entry to a Port is missing a Harness Type. The message is displayed in the **Messages** panel in the following format:

```
Missing Harness Type on Signal Harness
```

Default Report Mode

Fatal Error

Recommendation

Use the *Compile Errors* dialog to cross probe to the offending objects (Sheet Entries or Ports) within the Signal Harness. Specify a Harness Type across the Signal Harness for at least one of the objects.

Multiple Harness Types on Harness

This compiler hint appears when there are multiple Harness Types defined across a Signal Harness. The message is displayed in the **Messages** panel in the following format:

```
Multiple harness types on harness HarnessType1, HarnessType2
```

where

`HarnessType1` and `HarnessType2` are the multiple Harness Types specified across the Signal Harness.

Default Report Mode

Warning

Recommendation

Use the *Compile Errors* dialog to cross probe to the offending objects (Sheet Entries, Ports, Harness Connectors or Harness Entries) within the Signal Harness. Specify only one Harness Type across the Signal Harness.

Unknown Harness Type

This compiler hint appears when a Harness Type is found but a matching definition cannot be found in a Harness Definition file. This can happen for one of two reasons:

1. A Signal Harness is connected to an object (Sheet Entry, Port or Harness Entry) with an unknown Harness Type
2. When a Harness Entry in a Harness Definition file refers to an unknown Harness Type

The message is displayed in the **Messages** panel in the following format:

```
Unknown Harness Type HarnessType
```

where

`HarnessType` is the current unknown Harness Type.

Default Report Mode

Fatal Error

Recommendation

Use the *Compile Errors* dialog to cross probe to the offending object and then ensure that Sheet Entries, Ports and Harness Entries have a known Harness Type and Harness Definitions are accurate.

Violations Associated with Nets

Adding Hidden Net to Sheet

This compiler hint is related to multi-part components and appears when you have specified one or more pins to be hidden and connected to an existing net within the design - typically a power pin connected to VCC or GND for example. The message is displayed in the **Messages** panel in the following format:

```
Adding hidden net
```

Default Report Mode

Warning

Recommendation

The problem arises when the following properties for the offending pin(s) are evident (in the associated *Pin Properties* dialog):

- The **Hide** option is enabled
- The **Connect To** field contains the specific power net name
- The **Part Number** is that of the placed part (i.e. other than `0`).

To resolve this issue, you could enable the display of the pin(s) in the workspace (simply disable the **Hide** option). This option may prove to be less than desirable, especially if you have many hidden pins connected to power nets. Revealing these pins in the workspace can cause clutter as each pin would need to be wired to the appropriate power port object - preventing the design schematic(s) from being easily read.

A better solution is to clear the **Connect To** field and set the **Part Number** field to `0`. Leave the **Hide** option for the pin enabled. Repeat for each pin that has been connected to a power net in this way. Ideally, the power net connections should be assigned through use of part `0` in the source library component.

Notes

Only one instance of this violation type will be listed in the **Messages** panel. When investigating the error using the **Compile Errors** panel, a single entry will be listed reflecting the net which is being added. There may be multiple nets added - such as GND and VCC - but only one will be listed, determined by alphabetical order. If you clear the violation for a particular net, the next net (in order) will appear under this violation type.

Adding Items from Hidden Net to Net

This compiler hint is related to components and appears when you have specified one or more pins to be hidden and connected to an existing net within the design - typically a power pin connected to VCC or GND for example. The message is displayed in the **Messages** panel in the following format:

```
Adding items to hidden net NetName,
```

where

`NetName` is the name of the target net.

Default Report Mode

Warning

Recommendation

The problem arises when the following properties for the offending pin(s) are evident (in the associated *Pin Properties* dialog):

- The **Hide** option is enabled
- The **Connect To** field contains the specific power net name.

Resolution of this issue is on a per-component basis and also depends on whether a component contains multiple sub-parts.

For a non-multi-part component, enable the display of the pin(s) in the workspace (simply disable the **Hide** option). You will need to wire each pin to the appropriate power port for the net you wish to connect to.

The previous solution can also be applied to multi-part components, but a far better solution is to clear the **Connect To** field and set the **Part Number** field to `0`. Leave the **Hide** option for the pin enabled. Repeat for each pin that has been connected to a power net in this way. Ideally, the power net connections should be assigned through use of part `0` in the source library component.

Auto-Assigned Ports to Device Pins

This compiler hint occurs when there is a port on the schematic and there is no FPGA_PINNUM placement constraint specified for the design's configuration. The system automatically assigns a pin number to this port based on the device you have selected. The message is displayed in the **Messages** panel in the following format:

```
Port PortName will be automatically assigned to a Device Pin
```

where

`PortName` is the name of the offending port (e.g. LEDS[7..0])

Default Report Mode

Warning

Recommendation

Assign a placement constraint in the constraint file for that port by using FPGA_PINNUM with the appropriate device pin(s).

Bus Object on a Harness

This compiler hint appears when a Port, Sheet Entry or a Net Label within a Signal Harness has a label in the form `[X..Y]` indicating a bus object. The message is displayed in the **Messages** panel in the following format:

```
Bus Object Object Name [X..Y] at location placed on a harness
```

where

`Object` is the offending Port, Sheet Entry or Net Label and

`ObjectName` is the label of the offending object and

`Location` is the `X`,Y coordinates for the offending object.

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to cross probe to the offending object (Port, Sheet Entry, Net Label) and ensure the objects across the Signal Harnesses do not have labels using bus syntax.

Differential Pair Net Connection Polarity Inversed

This compiler hint appears when the polarity of a differential pair net is not the same as that of the differential pair pin to which it is connected. For example, the positive net is connected to the negative pin, or vice-versa. The message is displayed in the **Messages** panel in the following format:

```
Inversed connection on differential pair PairName: net NetName is connected
to pin PinDesignator (Polarity),
```

where

`PairName` is the name of the differential pair (e.g. `V_TX1`)

`NetName` is the name of the offending net (e.g. `V_TX1_P`)

`PinDesignator` is the designator of the device pin to which the offending net is connected (e.g. `E6`)

`Polarity` is the polarity of the pin (e.g. `negative`).

Default Report Mode

Warning

Recommendation

Use the *Compile Errors* dialog to quickly cross probe to the differential pair pin to which the connection has been made. Ensure that the net label attached to the connected wire is the same polarity as that of the pin. For example if the pin name is `IO_L02N_0` and the net label is `V_TX1_P`, change the net label to `V_TX1_N`.

If a pair of violation messages of this type appears - one for the positive net of the pair attached to the negative pin of the device and one for the negative net of the same pair attached to the positive pin - simply swap the attached net labels around.

Differential Pair Net Unconnected To Differential Pair Pin

This compiler hint appears when a differential pair net is not connected to a differential pair pin of a physical device. The message is displayed in the **Messages** panel in the following format:

`Net NetName of differential pair PairName is not connected to a differential pair pin`,

where

`NetName` is the name of the positive or negative polarity net in the pair (e.g. `V_RX1_N` or `V_RX1_P`)

`PairName` is the name of the differential pair (e.g. `V_RX1`).

Default Report Mode

Error

Recommendation

A true differential pair pin is hard-wired for a physical device (e.g. a Xilinx Virtex-II Pro FPGA device). Although such a pin may appear with the identifier `N` (for negative) or `P` (for positive) as part of its name, you cannot make a pin differential simply by changing its name.

Use the *Compile Errors* dialog to quickly cross probe to the pin to which the differential pair net is currently connected. Consider the following when resolving a violation of this type:

- If the net is correct but has simply been attached/wired to the wrong pin of the device, determine the correct pin and move the net label accordingly.
- If the net has been erroneously specified as a differential pair net, change the associated net label to the correct (non-differential) naming and remove the attached differential pair directive from the connecting wire.
- If the pin is, in fact, not to be used at all in the design, ensure that net label, wire and differential pair directive are removed and place a No ERC marker on it.

Differential Pair Unproperly Connected to Device

This compiler hint appears when a differential pair net is not properly connected to a differential pair pin of a physical device. The message is displayed in the **Messages** panel in the following format:

`Misconnected differential pair PairName: net NetName should be connected to pin PinDesignator`,

where

`PairName` is the name of the differential pair (e.g. `V_TX1`)

`NetName` is the name of the positive or negative polarity net in the pair (e.g. `V_TX1_N` or `V_TX1_P`)

`PinDesignator` is the designator of the device pin to which the offending net should be connecting (e.g. `E6`)

Default Report Mode

Error

Recommendation

This type of violation typically arises when the wire object for the differential pair net is not making electrical connection with the target pin of the device. Use the *Compile Errors* dialog to quickly cross probe to the device pin and ensure proper connection between the wire of the net and the pin itself.

Duplicate Nets

This compiler hint appears when two nets with the same name have been detected within the design. The message is displayed in the **Messages** panel in the following format:

`Duplicate Net Names Object NetName`,

where

`Object` is either `Wire` or `Bus Slice` or `Element[n]` (for a bus element)

NetName is the name of the affected net.

Default Report Mode

Error

Recommendation

This violation can arise when, for example:

- The design is hierarchical, with separate sheet symbols used to reference distinct child sheets, and sheet entries connecting to ports on those child sheets. The Net Identifier Scope is automatically (or manually) set to `Hierarchical (Sheet entry <-> port connections)`. The violation will occur if the same net label has been used on both child sheets. This is because net labels defined on each sub-sheet, even with the same name, remain local to those sub-sheets. The resolution in this case is to ensure unique net labeling is used between sheets.
- The design is flat and ports have been used within the design. The Net Identifier Scope is automatically (or manually) set to `Flat (Only ports global)`. The violation will occur if the same net label has been used between sheets. This is because net labels defined on each sheet, even with the same name, remain local to those sheets. The resolution in this case is to ensure unique net labeling is used between sheets.
- The net continuity between flattened schematic sheets is broken by the inadvertent use of ports or offsheet connectors with different names. Use the *Compile Errors* dialog to quickly cross probe to the duplicate net naming. Trace the net back to the incoming/outgoing port on each sheet and ensure the names for the ports are made the same.
- You may have the same net used in two different branches of a hierarchical design - i.e. different sheet symbols are used to reference different child sheets, but the same name is used for the top-level sheet entries and descendent ports, and the two symbols are connected by a physical wire or bus. The net continuity between these branches can be broken by the inadvertent use of sheet entries with different names or the omission of a physical bus/wire connecting the sheet entries. Ensure that the physical wire connecting the two sheet symbols is in place and wired correctly and that the sheet entries are named the same.

Floating Net Labels

This compiler hint appears when a net label has been detected to be floating - not attached to a wire or bus object - within the design. The message will also appear for a bus power port object that is not electrically connected to the rest of the circuit. The message is displayed in the **Messages** panel in the following format:

`Floating Net Label NetLabelName`,

where

`NetLabelName` is the name of the offending net label.

Default Report Mode

Warning

Recommendation

Ensure that the offending net label object is connected to the required wire or bus object. If the net label is redundant, simply delete it from the design.

Floating Power Objects

This compiler hint appears when a power port object has been detected to be floating - not electrically connected to a component - within the design. For example the power port may have been placed but is not yet wired up to the rest of the circuit. The message is displayed in the **Messages** panel in the following format:

`Floating Power Object NetName`,

where

`NetName` is the name of the net associated with the floating power port object.

Default Report Mode

Warning

Recommendation

Ensure that the offending power port object is connected into the circuit as required. If the power port is redundant, simply delete it from the design.

Notes

This message is related to the standard, single-signal power port objects. A floating bus power port object (available when the schematic is part of an FPGA project) will be flagged in the **Messages** panel as a Floating Net Label.

Global Power-Object Scope Changes

This compiler hint appears when a port-based object (port, offsheet connector, NanoBoard Port-Plugin component) has been connected to a power port object. The object can no longer exist on a global level - connected to a global power net - and is instead changed to be a local-level power net. The message for this error check is displayed in the **Messages** panel in the following format:

```
Global Power-Object NetName at Location1 has been reduced to local level by
presence of port at Location2,
```

where

`NetName` is the net to which the power port object is associated

Location1 is the X,Y coordinates for the power port object's electrical hotspot

Location2 is the X,Y coordinates for the port object's electrical hotspot.

Default Report Mode

Warning

Recommendation

This violation can typically arise when a power port object is incorrectly wired to a port rather than the intended pin or sheet entry. There may however be cases where you wish to force (and use) a scope change of this kind. Use the *Compile Errors* dialog to quickly cross probe to the offending objects. Assess whether the connection between port and power port is intended and, if not, remove the power port and wire the remaining port object to its intended destination as required.

Harness Object on a Bus

This compiler hint appears when an object such as a Port, Sheet Entry or Harness Entry has an associated Harness Type which represents a connection to a Signal Harness when it is placed on a bus. The message is displayed in the **Messages** panel in the following format:

Harness `Object ObjectName` at `location` placed on `bus`

where

`Object` is the offending Port, Sheet Entry or Harness Entry and

`ObjectName` is the label of the offending object and

`Location` is the X,Y coordinates for the offending object.

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to cross probe to the offending object (Port, Sheet Entry or Harness Entry). Use correct naming conventions for wires, buses and Signal Harnesses.

Harness Object on a Wire

This compiler hint appears when an object such as a Port, Sheet Entry or a Harness Entry has an associated Harness Type which represents a connection to a Signal Harness when it is placed on a wire. The message is displayed in the **Messages** panel in the following format:

Harness `Object ObjectName` at `location` placed on a wire

where

`Object` is the offending Port, Sheet Entry or Harness Entry and

`ObjectName` is the label of the offending object and

`Location` is the X,Y coordinates for the offending object.

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to cross probe to the offending object (Port, Sheet Entry or Harness Entry). Use correct naming conventions for wires, buses and Signal Harnesses.

Missing Negative Net in Differential Pair

This compiler hint appears when a negative polarity net has not been detected for a particular differential pair object within a design. The positive polarity net for the pair does exist. The message is displayed in the **Messages** panel in the following format:

Missing Negative Net for differential pair `PairName`, positive net `NetName`,

where

`PairName` is the name of the differential pair for which a positive polarity net has already been defined (e.g. `V_RX1`)

`NetName` is the name of the positive polarity net (e.g. `V_RX1_P`).

Default Report Mode

Error

Recommendation

This violation typically arises in the following situations:

- A differential pair directive has not been attached to the negative polarity wire of the signal pairing. The required net label (e.g. `V_RX1_N`) has been attached to the wire as required.
- An appropriately-named net label (e.g. `V_RX1_N`) has not been attached to the negative polarity wire of the signal pairing. The required differential pair directive has been attached to the wire as required.

To resolve this violation, simply locate the negative wire object for the pair and ensure that both the net label and differential pair directive are attached and that the name of the net label is specified as required. The net label for the negative wire will be essentially the same as that for the positive wire, but with an `_N` suffix instead of an `_P`.

Missing Positive Net in Differential Pair

This compiler hint appears when a positive polarity net has not been detected for a particular differential pair object within a design. The negative polarity net for the pair does exist. The message is displayed in the **Messages** panel in the following format:

Missing Positive Net for differential pair `PairName`, negative net `NetName`,

where

`PairName` is the name of the differential pair for which a negative polarity net has already been defined (e.g. `V_RX1`)

`NetName` is the name of the negative polarity net (e.g. `V_RX1_N`).

Default Report Mode

Error

Recommendation

This violation typically arises in the following situations:

- A differential pair directive has not been attached to the positive polarity wire of the signal pairing. The required net label (e.g. `V_RX1_P`) has been attached to the wire as required.
- An appropriately-named net label (e.g. `V_RX1_P`) has not been attached to the positive polarity wire of the signal pairing. The required differential pair directive has been attached to the wire as required.

To resolve this violation, simply locate the positive wire object for the pair and ensure that both the net label and differential pair directive are attached and that the name of the net label is specified as required. The net label for the positive wire will be essentially the same as that for the negative wire, but with a `_P` suffix instead of a `_N`.

Net Parameters with no Name

This compiler hint appears when a parameter set object is attached to a net object (wire or bus) and at least one of the defined parameters in the set has no name assigned to it. The message is displayed in the **Messages** panel in the following format:

`Invalid net-parameter name at Location`,

where

`Location` is the X,Y coordinates for the hotspot of the parameter set object associated to the net.

Default Report Mode

Warning

Recommendation

Access the *Parameters* dialog for the offending parameter set object and ensure that all parameters defined within have a name assigned to them. If the offending constituent parameter is not required, simply remove it from the set.

Net Parameters with no Value

This compiler hint appears when a parameter set object is attached to a net object (wire or bus) and at least one of the defined parameters in the set has no value assigned to it. The message is displayed in the **Messages** panel in the following format:

`Invalid net-parameter value at Location`,

where

`Location` is the X,Y coordinates for the hotspot of the parameter set object associated to the net.

Default Report Mode

Warning

Recommendation

Access the *Parameters* dialog for the offending parameter set object and ensure that all parameters defined within have a value assigned to them. If the offending constituent parameter is not required, simply remove it from the set.

Nets Containing Floating Input Pins

This compiler hint appears when an input pin for a placed part within the design has been detected to be floating - not electrically connected to any other part of the circuit. The message is displayed in the **Messages** panel in the following format:

`Net NetName contains floating input pins (PinList)`,

where

`NetName` is the name of the offending net

`PinList` is the comma-separated list of pins in that net which are floating.

Default Report Mode

Error

Recommendation

This violation can arise in a number of situations. Consider the following when resolving a violation of this type:

- If the pin is not to be used within the design, either tie it to the appropriate power line (e.g. `GND`), or place a No ERC directive on it.
- Ensure that the wiring to the pin is making electrical contact - i.e. the wire or bus connects to the pin's electrical hot spot.
- Use the **Navigator** panel to trace the connectivity of the parent net to which the offending pin is associated. Sometimes, a pin can be caused to 'float' when there is a break somewhere else in the net. For example, a pin might receive its signal from an input port on the sheet, which in turn is fed a signal from a linked sheet entry higher up in the design hierarchy. The input to this sheet entry may be disconnected. Fixing the connection to the sheet entry will resolve the floating input pin violation.

- Look for additional violation messages in the **Messages** panel that relate to the same parent net, especially those that mention unconnected objects - this can give an indication where the break in connectivity lies.

Nets Containing Multiple Similar Objects

This compiler hint is displayed when two or more objects of the same type (pin, port, sheet entry) and same electrical I/O specification, have been detected to be connected to each other in the same parent net. For example, an `Input Port` connected to an `Input Port`. The message is displayed in the **Messages** panel in the following format:

`Net NetName contains multiple ObjectType (ObjectList)`,

where

`NetName` is the name of the offending net

`ObjectType` is the type of object which has multiple instances found in the offending net. The entry will also reflect the object's electrical type

ObjectList is a comma-separated list of all instances of the object found in the offending net.

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to cross probe to the offending objects. Determine if the connected objects pose a real connectivity problem and, if so, change the I/O specification for one or both objects accordingly.

If an object is redundant, remove it from the design. If an object was intended to be wired into a different part of the design and has been connected to the wrong destination (therefore becoming part of the wrong net), ensure that it is disconnected from its current net and wired up to the correct one.

Notes

The actual report mode used for a violation of this type is dependant on the specific object types, their electrical I/O types, and the corresponding reporting level defined on the **Connection Matrix** tab of the *Options for Project* dialog (**Project » Project Options**).

The Report Mode setting on the **Error Reporting** tab of the *Options for Project* dialog has no affect on the Report Mode setting that appears for a violation of this type in the **Messages** panel. It is, however, used by the Online (real-time) error reporting, directly within the design workspace. Controls for the Online reporting are available on the **Schematic - Compiler** page of the *Preferences* dialog. Note that the Online error reporting for a specific object type-I/O type pairing will not be present if the individual check for that combination in the Connection Matrix has been set to `No Report`.

Nets with Multiple Names

This compiler hint appears when a net in the design has been detected to have multiple names associated with it. The message is displayed in the **Messages** panel in the following format:

`Nets Identifier has multiple names (NameList)`,

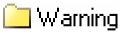
where

`Identifier` represents the type of connection and the name of the net. The connection can be one of the following:

- **Wire** - where the identifier will appear in the format `Wire NetName` (e.g. Wire DTSA)
- **Bus** - where the identifier will appear in the format `Bus Slice NetName` (e.g. Bus Slice A[0..7])
- **Bus Element** - where the identifier will appear in the format `Element[n]: NetPrefix` (e.g. Element[0]: A)

NameList is a comma-separated list of all names found associated with the offending net. These names can come from attached net labels, sheet entries, power ports and offsheet connectors.

Default Report Mode


Warning

Recommendation

This violation can be resolved by ensuring that the names of all net identifiers associated with a particular net are the same. However, in many cases it is beneficial to use different names for a particular net - for example when that net is present on different branches of a hierarchical design and different names better reflect the conducted signal in those branches. Similarly, you may want to describe the sheet entry of a particular sheet symbol using a different name to that of the net label attached to the incoming/outgoing wire or bus.

To freely use multiple names with nets in your design, and prevent related violation messages appearing in the **Messages** panel, simply set the **Report Mode** for this violation type to `No Report`, on the **Error Reporting** tab of the *Options for Project* dialog.

Nets with no Driving Source

This compiler hint appears when a net in the design has been detected to have no driving source. The message is displayed in the **Messages** panel in the following format:

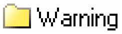
`Net NetName has no driving source (PinList)`,

where

`NetName` is the name of the offending net

`PinList` is the comma-separated list of pins in that net.

Default Report Mode


Warning

Recommendation

To resolve this violation, ensure that the net contains at least one pin with one of the following electrical types:

- IO
- Output
- Open Collector
- HiZ
- Emitter
- Power

Nets with only one Pin

This compiler hint appears when a net in the design has been detected to contain only one component pin. The message is displayed in the **Messages** panel in the following format:

`Net NetName has only one pin (Pin PinName)`,

where

`NetName` is the name of the parent net

`PinName` is the component designator-pin designator identifier.

Default Report Mode

No Report

Recommendation

By default, this message will not appear in the **Messages** panel. If you have enabled reporting for this violation type and want to resolve the issue, simply ensure that the offending net is wired to at least two component pins in the design.

Nets with Possible Connection Problems

This compiler hint appears when two objects connected to each other on the same net have mismatched electrical types that could lead to problematic connectivity. The pair of object types considered by this error check can be any combination of pins, ports, or sheet entries. For example an `Output Port` connected to a `Bidirectional Port`. The message is displayed in the **Messages** panel in the following format:

`NetName contains ObjectType1 and ObjectType2 objects (Reference)`,

where

`NetName` is the name of the parent net containing the possible conflicting objects

`ObjectType1` is the first offending object type

`ObjectType2` is the second offending object type

`Reference` is the identifier for one or both objects (e.g. `Port PortName`, `Pin PinName`, and `Sheet Entry` `SheetSymbolName-SheetEntryName`).

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to cross probe to the offending objects. Determine if the connected objects pose a real connectivity problem and, if so, change the I/O specification for one or both objects accordingly.

Notes

The actual report mode used for a violation of this type is dependant on the specific object types, their electrical I/O types, and the corresponding reporting level defined on the **Connection Matrix** tab of the *Options for Project* dialog (**Project » Project Options**).

The Report Mode setting on the **Error Reporting** tab of the *Options for Project* dialog has no affect on the Report Mode setting that appears for a violation of this type in the **Messages** panel. It is, however, used by the Online (real-time) error reporting, directly within the design workspace. Controls for the Online reporting are available on the **Schematic - Compiler** page of the *Preferences* dialog. Note that the Online error reporting for a specific object type-I/O type pairing will not be present if the individual check for that combination in the Connection Matrix has been set to `No Report`.

Same Net used in Multiple Differential Pairs

This compiler hint appears when the same polarity net (positive or negative) has been detected in multiple differential pair objects within the design. The message is displayed in the **Messages** panel in the following format:

`Net NetName is used in more than one differential pair objects: ObjectList`,

where

`NetName` is the name of the positive or negative polarity net being used in multiple differential pair objects (e.g. `V_RX1_N` or `V_RX1_P`)

`ObjectList` is a list of the differential pair objects in which the offending net is used. As the differential pair name is taken as the root of the net name, the names in this list will be identical (e.g. `V_RX1 V_RX1`).

Default Report Mode

Error

Recommendation

This violation typically arises when the same net label has been attached to multiple wire objects upon which differential pair directives have also been attached. Simply locate the offending net label objects (use the *Compile Errors* dialog to cross probe to the relevant area of the source schematic document) and amend the names as required.

The positive and negative wires in a pairing should have associated net labels with the same root name for the net, along with a `_P` and `_N` suffix respectively. For example, if two wires in a particular pair have the same net label, `V_RX1_N`, simply change the positive wire's net label to `V_RX1_P`.

Sheets Containing Duplicate Ports

This compiler hint appears when two ports with the same name are detected on the same schematic sheet in the design. The message is displayed in the **Messages** panel in the following format:

`Sheet contains duplicate ports Port Identifier at Location1 and Location2`,

where

`Identifier` is the name of the offending port

`Location1` is the X,Y coordinates for the first instance of the particular port

`Location2` is the X,Y coordinates for the second instance of the port.

Default Report Mode

Warning

Recommendation

Use the *Compile Errors* dialog to quickly cross probe to the duplicate port objects. Determine which port object is in error and either rename it or delete it from the design.

Signals with Multiple Drivers

This message appears when a pin (with I/O Type `Input` or `IO`) or a sheet entry (with I/O Type `Input` or `Bidirectional`) is connected to multiple driving pins or ports on the schematic sheet. The message is displayed in the **Messages** panel in the following format:

`Signal SignalName has multiple drivers`,

where

`SignalName` is the name of the affected signal.

Default Report Mode

Error

Recommendation

Use the *Compile Errors* dialog to cross probe to the affected signal pins/sheet entries. Trace the wiring from the pin/sheet entry to ensure it is connected as required, and correct if it is not.

Signals with no Driver

This compiler hint appears when a pin (with I/O Type `Input` or `IO`) or a sheet entry (with I/O Type `Input` or `Bidirectional`) is not connected to a driving pin or a port on the schematic sheet. The message is displayed in the **Messages** panel in the following format:

`Signal SignalName has no driver`,

where

`SignalName` is the name of the affected signal.

Default Report Mode

Error

Recommendation

This error can sometimes be caused by the offending pin/sheet entry not being wired to the intended net in the circuit. If this is the case, an additional error stating that the affected signal pin/sheet entry is unconnected will also appear in the **Messages** panel. Use the *Compile Errors* dialog to cross probe to the affected signal pin/sheet entry. Trace the wiring from the pin/sheet entry to ensure it is connected as required, and connect if it is not.

If the 'signal has no driver' message appears on its own, look for the component output pin to which the input pin/sheet entry is directly connected. Once identified, you can check/edit the output pin's electrical type from the *Component Pin Editor* dialog. Access this dialog from the associated *Component Properties* dialog for the part by clicking the **Edit Pins** button.

Notes

This violation type will only be displayed when compiling an FPGA project (`*.PrjFpg`), a single source schematic that is part of an FPGA project, or a free schematic document.

Signals with no Load

This compiler hint appears when a pin (with I/O Type `Output` or `IO`) or a sheet entry (with I/O Type `Output` or `Bidirectional`) is not connected to another part of the circuit (e.g. pin, port, sheet entry, power port, offsheet connector). The message is displayed in the **Messages** panel in the following format:

`Signal SignalName has no load`,

where

`SignalName` is the name of the affected signal.

Default Report Mode

Warning

Recommendation

If the offending pin or sheet entry is intended to be used within the design, ensure that it is connected to the relevant point in the circuit.

If the offending object is a component pin and you do not intend to use it within the design, simply place a No ERC directive on the pin.

If the offending object is a sheet entry that you do not intend to use, simply remove it from its parent sheet symbol.

Notes

This violation type will only be displayed when compiling an FPGA project (`*.PrjFpg`), a single source schematic that is part of an FPGA project, or a free schematic document.

Unconnected Objects in Net

This compiler hint appears when a pin, port or sheet entry object is not wired up to the rest of the circuit. The message is displayed in the **Messages** panel in the following format:

`Unconnected Object at Location`,

where

`Object` is the type and name of the offending object (pin, port or sheet entry)

`Location` is the X,Y coordinates of the object on the source schematic sheet.

Default Report Mode

Warning

Recommendation

Consider the following in order to resolve this error:

- If the pin/port/sheet entry is to be used, ensure that it is wired up to the rest of the circuit accordingly.
- If a port or sheet entry are redundant, remove them from the design.
- Tie any unused input pins to the appropriate power line.
- Place No ERC directives on unused input or output pins.

Notes

When the unconnected object is an input pin, an additional violation message will appear alerting you to the fact that the net to which the pin is associated contains floating input pins. The identifier for the pin will be listed in that message.

The actual report mode used for a violation of this type is dependant on the specific object type, its electrical I/O type, and the corresponding reporting level defined on the **Connection Matrix** tab of the *Options for Project* dialog (**Project » Project Options**).

The Report Mode setting on the **Error Reporting** tab of the *Options for Project* dialog has no affect on the Report Mode setting that appears for a violation of this type in the **Messages** panel. It is, however, used by the Online (real-time) error reporting, directly within the design workspace. Controls for the Online reporting are available on the **Schematic - Compiler** page of the *Preferences* dialog.

Unconnected Wires

This compiler hint appears when a wire object is detected to be floating - not electrically connected to any part of the circuit design. The message is displayed in the **Messages** panel in the following format:

`Unconnected line Location1 To Location2`,

where

`Location1` is the X,Y coordinates for the start point of the floating wire

`Location2` is the X,Y coordinates for the end point of the floating wire.

Default Report Mode

Warning

Recommendation

Ensure that the offending wire object is connected into the circuit as required. If the wire is redundant, simply delete it from the design.

Violations Associated with Others

Object not Completely within Sheet Boundaries

This compiler hint appears when a design object resides beyond the extents of the schematic sheet. The message is displayed in the **Messages** panel in the following format:

`Off sheet ObjectIdentifier at Location`,

where

`ObjectIdentifier` identifies the specific object that currently does not reside completely within the boundary defined by the sheet. The identifier is composed of the object's type and its name/designator (e.g. `Port PortName`)

`Location` is the X,Y coordinates for the object's electrical hotspot.

Default Report Mode

Warning

Recommendation

When placing or pasting objects onto a sheet, you are prevented from placing/pasting beyond the extents of the sheet's border. This issue typically arises when the size and orientation of the sheet is changed after object placement. Consider the following to resolve the problem:

- Change the sheet orientation.
- Choose a larger sheet size.
- Move the offending objects back within the sheet boundary.

The first two options are carried out from the **Sheet Options** page of the *Document Options* dialog (**Design » Document Options**). Changing sheet size is the simplest way to resolve the issue. Moving objects manually may require layout changes to the circuit to provide enough space to accommodate the offending objects.

Off-Grid Object

This compiler hint appears when an object is not aligned to the current Snap grid. The message is displayed in the **Messages** panel in the following format:

`Off grid ObjectIdentifier at Location`,

where

`ObjectIdentifier` identifies the specific object that is currently off-grid. The identifier is composed of the object's type and its name/designator (e.g. `Pin PinDesignator`)

`Location` is the X,Y coordinates for the object's electrical hotspot.

Default Report Mode

Warning

Recommendation

Ensure that the Snap grid is enabled on the **Sheet Options** tab of the *Document Options* dialog (**Design » Document Options**). The offending object can be moved back onto grid manually, or by using the **Edit » Align » Align To Grid** command.

Violations Associated with Parameters

Same Parameter Containing Different Types

This compiler hint appears when two parameters possessing the same name have been assigned to the same design object, but the parameters have differing types. The message is displayed in the **Messages** panel in the following format:

`Same parameter contains different types Object Types`,

where

`Object` is the particular object to which the offending parameter is associated

`Types` shows the different types for the parameter as a pairing (e.g. BOOLEAN and STRING).

Default Report Mode

Error

Recommendation

Typically, you would not have multiple parameters of the same name associated to an object. Review the parameters assigned to the object and remove the erroneous/redundant one.

Same Parameter Containing Different Values

This compiler hint appears when two parameters possessing the same name have been assigned to the same design object, but the parameters have differing values. The message is displayed in the **Messages** panel in the following format:

`Same parameter contains different values Object Values`,

where

`Object` is the particular object to which the offending parameter is associated

`Values` show the different values for the parameter as a pairing.

Default Report Mode

No Report

Recommendation

Typically, you would not have multiple parameters of the same name associated to an object. Review the parameters assigned to the object and remove the erroneous/redundant one.

Revision History

Date	Version No.	Revision
07-Nov-2006	1.0	New template release
03-Nov-2007	1.1	Updated for Altium Designer 6.8