The Challenge

With ever-increasing device switching speeds comes the challenge of maintaining the integrity of the signal, and meeting the signal's timing requirements. The signal integrity can be managed through controlled impedance routing, which is achieved through careful design of both the PCB stackup and the routing widths to be used on each layer.

The timing requirements are met by matching the routed lengths of the signal paths. For a set of 2-pin signal paths, each running from an output pin to a single input pin, calculating and comparing the lengths is a straightforward process. This is not the case for many typical design solutions though, where there may be a series termination component in the signal path, or there are more than 2 pins in the signal, which could then be routed using a Balanced T or a Fly-By routing topology, as shown in the image below.
Four DDR2 RAM chips routed using a Balanced T topology.

The Solution

The designer’s job is to translate their design requirements, such as the maximum route length allowed to meet the timing budget, into a set of design rules, such as a Length rule to ensure that the timing it met, and a Matched Length rule to detect potential timing mismatches.

Now the designer sees the signals in terms of their function - here is an address signal and it must be routed from this connector to each memory device, to achieve that I'll route using a fly-by topology with a termination resistor at the end. I might also require a series terminator at the source. Even though address A0 passes through a termination resistor, to the designer, that signal is still A0 on the other side of that resistor.

But the PCB editor sees each signal simply as a set of connected pins (commonly referred to as a net) — Net A0 goes from this connector pin to this memory component pin, then to this memory component pin, and so on. As soon as a series termination resistor is added, that address line becomes 2 discrete nets. This makes it difficult for the designer to specify key design requirements, such as Length and Matched Length design rules.

In Altium Designer 15 and later, this can be managed by a feature called xSignals. This feature enables the correct treatment of a high-speed signal path as just that - a path for a signal to travel between a source and destination, through termination components, as well as branches.
An xSignal is essentially a designer-defined signal path between 2 nodes - these can be 2 nodes within the same net, or they can be 2 nodes in associated nets separated by a component. The xSignal can then be used to scope relevant design rules such as Length and Matched Length, which will then be obeyed during design tasks, such as interactive length tuning.

Below is a summary of the xSignal features.

**Creating a New xSignal**

An xSignal is a designer-defined signal path between 2 nodes - they can be 2 nodes within the same net, or they can be 2 nodes in different nets.

xSignals are defined using one of the following methods:

1. Use the xSignals Wizard. This will be the most common approach to creating xSignals, and is covered in detail in the xSignal Wizard page.

   ![Checkmark]

   - The initial release of the xSignals Wizard included a Custom Component mode, use this to quickly detect and create xSignals between chosen components. Over time the Wizard will be extended to support specific high-speed design technologies, such as DDR3/4 and USB3. When you open the xSignal Wizard page, use the version selector to check out what later versions of Altium Designer support.

2. Create a single xSignal based on selected pads. Select the required start pad and end pad (these pads can be in different nets if there is a series termination component). Pads can be directly selected in the workspace, or the PCB panel can be used in Nets mode to locate and select the pads. Once the pads are selected, either right click on a selected pads in the workspace and run the Create xSignal from Selected Pins command, or right click on a pad in the PCB panel and run the Create xSignal command. The new xSignal will be listed in the xSignal mode of the PCB panel.

   - When you are defining an xSignal based on selected pins (footprint pads), select only the start pad and the end pad before running the Create xSignal from Selected Pins command.

3. Select the source component, then right-click on the selected component and choose the xSignal » Create xSignals between Components command from the context menu. The Create xSignals Between Components dialog will open, with the chosen source component selected. The dialog is described below.

4. Select one or more series components in the workspace, then right-click on one of the selected components and choose the xSignal » Create xSignals from Connected Nets command from the context menu. The Create xSignals From Connected Nets dialog will open. The chosen source component, and the nets connected to that component, will be selected. The dialog is described below.
5. There may also be situations where you wish to create an xSignal within an existing xSignal, in this situation the **xSignal** mode of the PCB panel can be used. Ensure that the **Select** option is enabled at the top of the panel, locate the current xSignal, select the required pads in the **xSignal Primitives** section of the panel, then right-click on one of the selected pads in the workspace and use the method described in point 3 of this list to complete the process.

Now that the xSignal is defined, it can be used to scope design rules.

Select the 2 pads in the **Nets** mode of the panel, then right-click on one of the selected pads and choose **Create xSignal**. Note that the pads are in different nets.

### Create xSignals Between Components Dialog

If you have a large number of xSignals to define, it is more efficient to use the **Create xSignals Between Components** dialog. Accessed via the **Design » xSignals » Create xSignals** command, the dialog presents **Source Components** and their **Source Nets** on the left, and **Destination Components** on the right, and allows you to create 1 or many xSignals in a single operation. The approach is to:
1. Select the Source Component.
2. Select the Destination Component(s).
3. Select the Source Net(s) of interest.
4. Click the Analyze button. The software will identify all possible xSignals, between the chosen components, that include the chosen nets, and list them in the xSignals region. All potential xSignals will be selected. Note that the analysis algorithm follows the current topology of the chosen nets and this will influence the proposed xSignals - more on this below.
5. Select the required class at the bottom of the dialog, or type in a name to create a new class. If no class is chosen, they are still created and can later be added to any xSignal class in the Classes dialog (Design » Classes).
6. Click OK to create the xSignals.

The dialog will close, returning you to the workspace. The new xSignals will be listed in the xSignals mode of the PCB panel.

Use the Filters above each list to quickly locate the components or nets of interest - wildcards are supported.
Create xSignals From Connected Nets Dialog

If you are creating xSignals that include series termination components, a good approach is to use the Create xSignals from Connected Nets command. The command is available whenever a component is selected, either via the Design » xSignals sub-menu, or the right-click xSignals sub-menu.

This command is designed to build xSignals outward from a selected series termination component, such as a resistor or capacitor. It supports both one or more discrete components, and one or more multi-instance pack-style components, such as resistor networks. After running this command, the Create xSignals from Connected Nets dialog will open.
The Role of the Net Topology

When you define an xSignal, it is between 2 nodes, or pads. However, when you select that xSignal in the xSignal mode of the PCB panel, it will actually follow the path of the connection lines that runs between those 2 pads - indicating that this is the path that the software assumes the xSignal will be routed. The reason it does this is because it is obeying the topology defined for that net. Net topology is defined by the applicable Routing Topology design rule, the default topology is Shortest.

The simple animation shows a CPU connected to 4 DDR3 memory chips, which is going to be routed using a fly-by routing strategy. The DRAM_A2 xSignal class contains 4 xSignals, first the class is selected, then each xSignal is selected in turn. You can see how the xSignal path follows the topology of the net, which is currently set to the default, Shortest.
Because the net topology is currently set to shortest, the xSignals are not following the required path from the processor to the memory chips.

If you plan on using the *Create xSignals Between Components* dialog, you will need to configure the topology of the net(s) to ensure the xSignal analysis algorithm understands the intended path of the routed xSignal.

**xSignal Creation Commands**

Apart from the **Design » xSignals » Create xSignals** command, there are other xSignal creation commands appear in the **xSignal** sub-menu when certain conditions are met.

Below is a summary of the commands and when they are available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Create xSignal from selected pins</strong></td>
<td>Immediately creates a single xSignal. This command is available when there are 2 or more pads selected in the workspace, and is the same command presented when you right-click on one of multiple selected pads.</td>
</tr>
<tr>
<td><strong>Create xSignals between components</strong></td>
<td>This command is available when components are selected in the workspace. When it is run the <strong>Create xSignals Between Components</strong> dialog opens with the component(s) pre-selected. Ensure that the correct Source and Designation components are selected, then complete the Analysis/Creation process.</td>
</tr>
<tr>
<td><strong>Create xSignals from connected nets</strong></td>
<td>Use this command when there are one or more series termination components to create xSignals for. Select the termination component(s), then run the command to open the <strong>Create xSignals from Connected Nets</strong> dialog, ready to complete the process of creating a set of xSignals.</td>
</tr>
</tbody>
</table>
### Command and Description

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create xSignals</td>
<td>Opens the <a href="#">Create xSignals Between Components</a> dialog. This command is always available.</td>
</tr>
</tbody>
</table>

### Defining the Branch Point in a Balanced T Pattern

One of the challenges of a Balanced T routing strategy is how to equalize the length of the trunks and the branches beyond the T points? The available nodes in the net are only at the pads, so it is not possible to define separate xSignals for the trunk, and from the branch point to the end of each branch. The branch points are indicated by the red dots in the image below.

One way to solve this problem is to add a single pin component to the net. Create a component with a single pad that is the size of the vias being used in the design. If the branch point component pad is single-layer, then it can also be used in combination with a blind or buried via, by placing it on the via's start or end layer, giving complete flexibility as to how the routing is created. If you only want to include the branch point component on the PCB, set the branch point component's Type to Mechanical to exclude it from the BOM and prevent any synchronization issues with the schematic. If you plan on including the branch point component on the schematic, the component Type can be set to Standard (no BOM).

Balanced T routing can require matched lengths between intermediate branch points.

Because the branch point is a node in the net, you can now define xSignals for just the trunk, for each major branch, and for each minor branch, if needed. These can then be used to scope matched length design rules, giving the designer complete control over how finely the length matching is to be performed.

### Working in the xSignal Panel

The PCB panel includes an [xSignal](#) mode, select it from the drop-down at the top of the panel to detail all [xSignal Classes](#), [xSignals](#), and their [Primitives](#). The [xSignal](#) mode of the panel is used to examine and manage existing xSignals.

Right-click in the relevant section of the panel to perform xSignal related editing tasks, including:

- **Add, Delete or Edit** the members in an xSignal class - appropriate classes of xSignals are
essential for correct design rule creation. How the xSignals are clustered into classes will depend on the what signal lengths must be matched, which will be determined by the chosen routing topology. The Add and Delete commands are available on the right-click menu, double-click on an existing class to edit it.

- Change the color of an xSignal (changes the color of all connection lines in the xSignal). This can be performed on a class, or selected xSignals. The use of color can simplify working with a crowded PCB. As well as changing the color of the ratsnest, the net color can also be used as an overlay on the routing, which greatly helps identify related nets during routing.

Displaying and Hiding the xSignal Lines

When you click to select an xSignal in the panel, the workspace display will dim, select and zoom, based on the settings of these options at the top of the panel. If the xSignal is already routed, a thin line will be shown, following the path that the software uses to calculate the xSignal length.

Deleting an xSignal

Select the xSignal in the panel and click the Delete button below the list of xSignals. Alternatively, right-click and select Delete from the context menu, or press Delete on the keyboard.

xSignal Query Keywords

The PCB editor includes a powerful and sophisticated filtering engine. This engine is used to identify objects when: searching for objects in the workspace; applying rules during interactive and automatic design tasks; and for checking rule compliance. The designer tells the filtering engine what objects they are interested in by writing a query, using query keywords recognized by the filtering engine.

The following xSignal type query keywords have been added for use in design rules and workspace filters:

Membership Check Type Keywords

- **InxSignal** - Is the object in the specified xSignal, an example might be InxSignal('DRAM_A0_PP1')
- **InxSignalClass** - Is the object in the specified xSignal class, an example might be InxSignalClass('PCIE')
- **IsxSignal** - Is the object an xSignal with the specified name, an example might be IsxSignal('DRAM_A0_PP1')

Attribute Check Type Keywords

- **InAnyxSignal** - Is the object in any xSignal, an example would be InAnyxSignal

Design Rule Support for xSignals

Design rules are how the designer translates their requirements into a set of instructions that the PCB editor can understand and obey. Rules can be checked during object placement, referred to as Online DRC, or as a post process, referred to as Batch DRC. Refer to the Design Rules page to learn more about design rules.
**Matched Length Rule**

The Matched Length rule is used to ensure that the length of the specified nets is within the specified range. This rule is essential in a high-speed design, where the challenge is not just about how long it takes the signals to arrive (which is determined by their overall length), but how important it is that the specified signals arrive at the same. Depending on the signal switching speeds, the function of the signal, and the materials used in the board, the allowed difference could be as much as 500mils, or as little as 1mil.

The image below shows an example of the Matched Length design rule configured to target all xSignals in the xSignal class PCIE. These signals have a tight tolerance of 0.025mm (~1mil).

Note that the Matched Length design rule now includes a Check Between xSignals constraint mode.

The image below shows the PCIE_TX xSignal class selected in the panel, and those xSignals selected in the workspace.
As well as the PCIE class, there are also classes defined for the TX and RX pairs. Note that 1 of the TX xSignals fails the applicable matched length rule.

## Length Rule

The Length rule is used to ensure that the overall routed length is within the specified range. This rule is typically used to ensure that the target nets are no longer than the specified length, for example to ensure that the circuit timing requirements will be met. The length rule now respects the xSignal type queries listed above.

## Accurate Length Calculations

A key requirement of defining high speed design rules is accurate calculation of the route lengths. The traditional approach to calculate signal length is to add up the centerline length of all segments used in a route, as well as the vertical distance due to the height of the vias, which was originally determined by the board thickness.

This approach is not adequate for a high speed design though, for a number of reasons, including:

- Stacked and overlapping objects - an algorithm that simply adds the centerline length of all objects in a net does not cater for stacked or overlapped objects.
- Wandering route path within an object - there is often routing objects completely within a pad or via, which can falsely add to the length, as shown in the image on the left below. The image on the right shows the correct way to calculate the length when a fill object is part of the routing.
• Via length - blind and buried vias do not traverse all layers of the board, so the board thickness is not sufficiently accurate to determine the vertical length. The actual via height must be used, taking into consideration the copper and insulation thicknesses that the via passes through.

The PCB editor's new length calculator returns the most accurate route length possible.

The length calculation is accurately calculated along the centerline of the shortest path, as shown in these 2 images.

<table>
<thead>
<tr>
<th>11 Primitives (1 Highlighted)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type / Name</td>
</tr>
<tr>
<td>Pad HDR1.9</td>
</tr>
<tr>
<td>Pad HDR1-10</td>
</tr>
<tr>
<td>Pad U1-135</td>
</tr>
<tr>
<td>Pad U5-3</td>
</tr>
<tr>
<td>Track Width=0.2mm (177mm,207.2mm)(180mm,207.2mm)</td>
</tr>
<tr>
<td>Track Width=0.2mm (177mm,210.2mm)(180mm,210.2mm)</td>
</tr>
<tr>
<td>Track Width=0.2mm (177mm,207.2mm)(177mm,210.2mm)</td>
</tr>
<tr>
<td>Track Width=0.2mm (180mm,210.2mm)(186mm,210.2mm)</td>
</tr>
<tr>
<td>Via (177mm,207.2mm) Signal Layer 2 to Bottom Layer</td>
</tr>
<tr>
<td>Via (177mm,210.2mm) Top Layer to Bottom Layer</td>
</tr>
<tr>
<td>Via (180mm,210.2mm) Top Layer to Signal Layer 1</td>
</tr>
</tbody>
</table>

Accurate lengths, based on the layers traversed and the stackup dimensions, are calculated for vias.

**Net-related Terminology**

In the PCB editor, the following terminology is used:

• **Net** - a collection of components pins (nodes) that are connected to each other. The arrangement of how those nodes connect to each other is referred to as the topology, the default topology is shortest.

• **From-To** - Conceptually, a From-To runs between 2 nodes in a net. The From-Tos can be
created to follow the topology, or arrangement of nodes, in that net. For example, the net topology could be from R1-1, to U1-5, to U3-2, to R5-2. This net could have 3 From-Tos: R1-1 to U1-5; U1-5 to U3-2; and U3-2 to R5-2. If the topology is changed, so will the possible From-Tos. From-Tos are created in the From-To mode of the PCB panel, by either clicking the Generate button to create them based on a topology, or by selecting 2 pads in a net and clicking the Add From To button.

- **xSignal** - A user-defined set of nodes, typically a sub-set of a net (from this node to that node), or a combination of 2 nets that include a series component, such as a termination resistor.

---

---

**Source URL:**
https://www.altium.com/documentation/display/ADES/((Defining+High+Speed+Signal+Paths+with+xSignals))_AD