

# Cadence Allegro X (24.1) Tutorial

Copyright (C) Istvan Nagy, 2025 [www.buenos.extra.hu](http://www.buenos.extra.hu) Free to share!

## 1. Introduction

This tutorial is targeting complex high-speed digital circuit board designs with Cadence Allegro X, using OrCAD or CHDL for schematics. There are different grades of high-speed designs from the microcontroller boards up to the server computer motherboard and data center line card designs, so the methodology here to be suitable for all, is based on the high-end and scalable down.

<b>Cadence Allegro X (24.1) Tutorial</b> _____	<b>1</b>	<b>3. High-Speed Signal Objects</b> _____	<b>6</b>
<b>1. Introduction</b> _____	<b>1</b>	<b>4. Constraint Manager</b> _____	<b>6</b>
<b>2. Basic PCB Design</b> _____	<b>1</b>	<b>5. Interactive High-Sp Route</b> _____	<b>8</b>
<b>2.1. Projects and Editors</b> _____	<b>1</b>	<b>6. Signal Integrity</b> _____	<b>9</b>
<b>2.2. Schematics with OrCAD Capture</b> _____	<b>1</b>	<b>7. Typical Examples</b> _____	<b>10</b>
<b>2.3. Schematics with Concept HDL</b> _____	<b>2</b>	<b>7.1. PCIe Gen4 SERDES bus design</b> _____	<b>10</b>
<b>2.4. SCH Library (OrCAD)</b> _____	<b>2</b>	<b>7.2. DDR4 Memory-Down design</b> _____	<b>10</b>
<b>2.5. PCB Layout Design (Allegro)</b> _____	<b>2</b>		
<b>2.6. PCB Library (Allegro footprints)</b> _____	<b>5</b>		
<b>2.7. Design Reuse (OrCAD)</b> _____	<b>5</b>		

## 2. Basic PCB Design

### 2.1. Projects and Editors

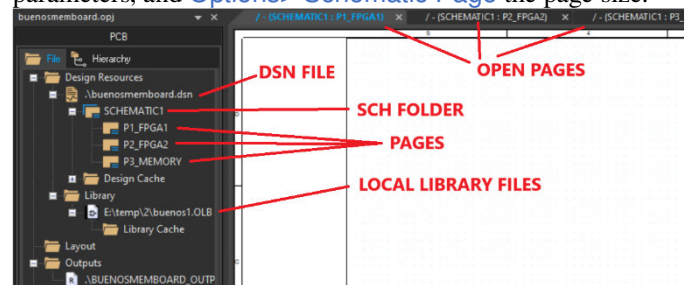
In the Cadence flow we open the schematics and the layout in **separate programs**, by separate teams, so there is not really an overall project environment. We can use OrCAD capture or ConceptHDL (CHDL) for schematic design, while the Allegro PCB editor program is for PCB layout. The Cadence Project Manager is only used for schematic editing in ConceptHDL.

Most companies have managed and released **shared libraries**, but one-person companies can keep library files edited within project folders. We have to add existing company libraries to be accessible by the project. The way to do it, and the file types are dependent on each program, OrCAD/CHDL/Allegro.

### 2.2. Schematics with OrCAD Capture

OrCAD has two modes, for component library handling, the basic and the **Capture CIS mode**. When we start the program we have to select license type for that mode. The file structure includes an .OPJ project file with settings, and one .DSN file for the drawing containing all pages. We can also have local .OLB library files for editing, but using the CIS company database parts is preferred. To create a new design: **File> NewProject> Schematic**. Open an existing: **File> Open...** the OPJ file. On the left side we have the project panel, with a hierarchical view. We

have upper tabs to select what we are viewing: one of the schematic pages, or a component properties page, or a BOM page... The **Options> Preferences** menu lets us set design parameters, and **Options> Schematic Page** the page size.



**Multi-page** schematics can be either hierarchical with






PORT connections in the module and SHEET SYMBOL in the top level schematic, or a flat design with off-page connectors . When we place a sheet symbol, we select what schematics it refers to. On the project panel we can see a tree hierarchy view. Within the DSN we have schematic folders, within those we have pages. On the schematic folder we can **rightclick> NewPage** to add more. For a hierarchical design we would have modules in separate schematic folders, or in separate DSN files in a library.

We **place parts** from the network CIS company library by **Place> Component** (the CIS/Component Explorer, used to be "Place> Database Part"). The **Place> Part** uses local libraries. Part rotation is with **rightclick> Rotate**, or press "R" while placing it. Doubleclick on a part opens a full-page property editor,

where we can see/edit/add properties. Usually we don't edit properties here, rather in the released library, except for the POPULATE property. For example, have a separate library item for a 1k resistor than a 10k resistor. We should make the POPULATE property visible, by **rightclick> Display** in the property editor. The rows/columns can swap with the Pivot button. We can search for placed parts in the search bar or in the browse spreadsheet (select DSN file, **Edit> Browse**). We can edit multiple net labels, off page names or part properties, if we select multiple in the drawing, then **Edit> Properties**, then copy from the prop editor to Excel, edit there and copy back to OrCAD.

**CIS library:** The **Options> CIS config** sets our access to the company CIS component database (spreadsheet in MS Access database) on the network, by selecting a DBC file. This contains a component/property table and library file (folders for .OLB, and footprint files) names for each part. Need a Microsoft ODBC driver for it to work. Library file access is set in the capture.ini file, edited in the CIS Admin tool.

A component can be **do not populate DNP**, to leave it out of the purchase BOM and pick&place file. Either to design for debugging (to swap a PU/PD on the prototype without trace cutting), add possible future features or product variants. We can do it in different ways depending on company; we could use a POPULATE property in certain components with a value "DNP" meaning not populated, or overwrite the part number with "DNP", then delete these rows from the Excel BOM, or we can use variants.

Once components are placed, we **wire** them with . We can place ground symbols with , and power rail symbols with . We add net labels (alias) with . We can draw buses with . Diffpair nets must be named with \_P and \_N suffixes. Nets and ports are local, Off-sheet-connectors and power symbols are global on all pages. Zoom works with CTRL+scroll.

Once done, we **annotate** refdes to all components by clicking on the DSN file in the project tree, then **Tools > Annotate**, then **Tools> Create Netlist**. Then export a BOM with **Tools> Bill Of Materials**, ensuring that the "property string" matches our company standard, and includes the POPULATE, company part number, MFR\_PN, description, footprint, value and anything needed. We find all DNPs in the exported spreadsheet and delete them, before uploading. We need to update the page numbers at the off-page symbols by **Tools> Cross Reference**. Error checking with **Tools> Part Manager** and **Tools> Design Rules Check**. We also export a PDF schematic with **File> Print to PDF**.

### 2.3. Schematics with Concept HDL

We start this with the Cadene „**Project Manager**” program. It has large buttons on it for open/create project, and for open schematics. **File> New> Design** creates a new project. If we open the schematics then the CHDL window opens. There is a complex file structure for CHDL projects with many files, the important sub-folders are the name/worklib/name/physical where the Allegro board file will be.

We can **place components** from **Place> Component**. There are OrCAD-like buttons for wiring, net names. Press "T" for text. The pages are listed on the left. One page is open at a time. No properties can be edited in the schematic, only in the library. We have to set part numbers **Tools> Options> Custom Variables**.

**Multi-page** schematics can be hierarchical with PORT connections in the module and SHEET SYMBOL in the top level

schematic. We can also design flat schematics with no top-level, then we have to use off-page connectors from the component library.

A component can be **do not populate (DNP)**, to leave it out of the purchase BOM and pick&place file. Either to design for debugging (to swap a PU/PD on the prototype without trace cutting), add possible future features or product variants. For DNP in CHDL we have to use variants. We have to create a variant associated with the board part number in **Variants> Create Variant**. Then **Variants> Edit Variant** to set up part numbers and such. Then we edit the design, mark intended DNPs with text comments, then save, create netlist, then **Variants> View Variant Schematic> select** the one with the part number. Then click on marked parts, **rightclick> Variants> Mark as do not install**. Then **Variants> View Variant Schematic> Base**.

Once done, we **annotate** refdes to all components and export netlist with **CHDL> File> Export> Physical**, and update all off-page numbers in **Project Manager> Tools> Crefer**. Then export a BOM with **CHDL> Tools> Packager> Bill Of Materials**. We also export a PDF schematic with **File> Publish PDF**.

### 2.4. SCH Library (OrCAD)

To make our own schematic symbols, we have to create/open a **.OLB file** using **File> New> Library**, that is added to the project tree on the project panel. In heterogenous split components the top level is called part, the sub-symbol is called a "section". In the project tree on the libraries/.OLB **rightclick> New Part From Spreadsheet**. We have to enter how many sections we will have. We can prepare a pin table in Excel in a matching format like below, then select data (not header), then CTRL+C to copy, then in OrCAD upper/left cell CTRL+V to paste.

	A	B	C	D	E	F	G	H	I
1	Pin Number	Pin Name	Type	Pin Visibility	shape	Pin Group	Position	section	
2	1	IN1	Passive	1 Line	1 Line	1 Left	A		
3	2	IN2	Passive	1 Line	1 Line	1 Left	A		
4	3	OUT1	Passive	1 Line	1 Line	3 Left	A		
5	4	GND	Passive	1 Line	1 Line	3 Left	A		
6	5	IN3	Passive	1 Line	1 Line	2 Left	B		
7	6	IN4	Passive	1 Line	1 Line	2 Left	B		
8	7	OUT2	Passive	1 Line	1 Line	3 Left	B		
9	8	GND	Passive	1 Line	1 Line	3 Left	B		
10									
11									
12									
13									

from datasheet or FPGA Pinout File      default      for pin swapping within groups      sub-part

On the project tree we can see our new part, we can doubleclick to edit for manual graphical modifications. In the properties window we have to enter **properties** like part number, footprint and value. We have to add company-specific properties, for example if we work at ACME-inc., then ACME\_PN, ACME\_DNP, MFR\_PN... In most cases this would have to be released in the company CIS library by a librarian.

### 2.5. PCB Layout Design (Allegro)

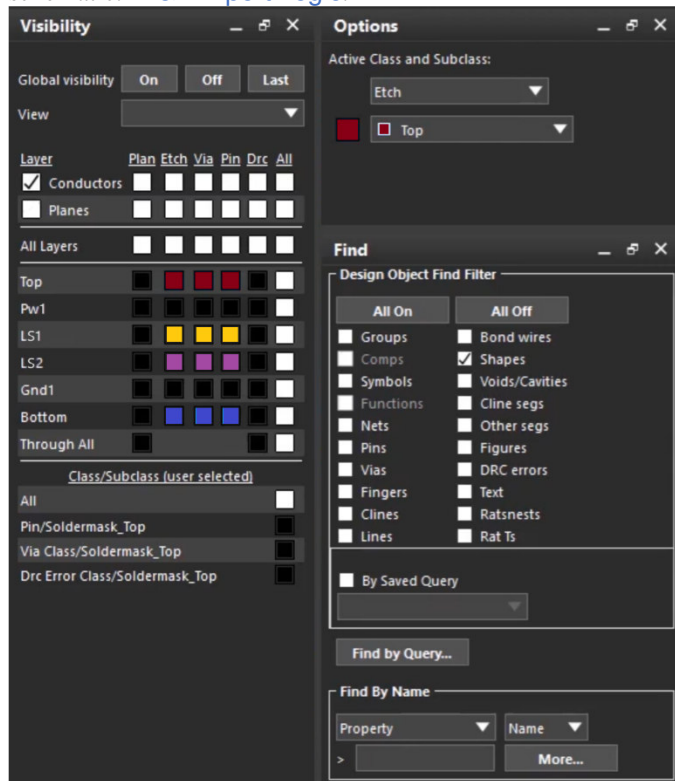
Allegro is also called "PCB Editor", it opens from the windows start menu. We will need the "high-speed option" license. Some **basic settings** for hole visibility are in **Setup> Design Parameters**, and grids in **Setup> Grids** (5mil for placement, 1mil for routing) Allegro has to see our (company) libraries/folders, that is set in **Setup> User Preferences> Paths> Library> padpath** for pads and **>PSM path** for footprints.

We create a **new layout** design with **File> New> Board (Wizard)**, and set a few basic settings. To start the PCB design, we

need to get a DXF file from our mechanical engineer, that contains the board outline, external connector outlines and mounting holes, and import it into a mechanical layer. [File> Import> DXF](#), then create and set up a layer conversion file with layer mapping to Board Geometry / new mech layer, then click incremental, and Import. In the [Setup> Subclasses](#) we can create more mech layers. We have to draw a closed line with [Add> Line](#) on the BoardGeometry/Outline layer. Set the origin [Edit>Setup> Change Drawing Origin](#), then click on the board at the corner.

**Stackup Layers:** [Setup> Cross Section](#). Add copper layers (rightclick), define their thicknesses, signal/plane type order. Selectable stackup materials are managed in [Setup> Materials](#). Layers are categorized in a 2-level hierarchy as class/subclass, the stackup copper layers are always Etch/layername. We use the mechanical layers for all manufacturing comments and fab notes, created in [Setup> Subclasses](#). The main layer classes are Etch (for copper traces/planes, per layer), Board geometry (outline, text, mechanical info, fab notes), Package Geometry (outline, pin number from footprint, solder mask, paste), Pin (per layer), via (per layer), Manufacturing (drill figure, legend), refdes (top/bottom), various keep-outs (per layer). We can have the same subclass in multiple classes, like silkscreen in Board-geo and Package-geo.

Once the board is set up: we have to **import netlist** from schematic: [File> Import Logic](#).








The allegro user interface has 3 standard **panels**, some buttons and top drop-down menus. The panels can auto hide and seen as a tab, or they can be fixed visible. The Options panel is for setting things for the current command. The Visibility panel is for enabling visibility for a matrix of object type vs layer. The Find panel is to control object filtering for click edits or type in net or component name to be highlighted. In Allegro everything that is on a layer (pads, vias, traces) can be visible or invisible independently, set as a matrix. The object filter on the Find panel is used to control what object type (package symbol, net, pad, via, text, line, cline, cline

segment, shape) we want to edit and what we don't at the moment, so when we are sliding traces the components will not move.



There is a free Allegro **viewer** software that anyone can download for free, that is useful for coworkers, layout review or lab debugging. It is lacking any editing functions, while it still has all the layer/color commands. Starting in 2021 the Allegro Free Viewer has the user interface of the OrCAD Presto PCB, not the Allegro's GUI. This new viewer has more data displayed on its Panels than the Allegro editor's panels: The visibility panel splits to 3 new sub-tabs, one for layers, one for nets/objects, one for general display stings. The layers sub-tab shows more than just the Etch subclasses. The eye icon crossed means not visible. The rectangles can change color here without a separate window. The options and find panels from Allegro are rearranged as the new Properties panel (similar to options, object filter and info) and a Search panel (to browse components or nets, can search, but without a "\*" key). For editing, the active layer is selected on the bottom of the screen on tabs. The rest of this document is not about the OrCAD Presto, but Allegro, although they have similar CM.



**Ratsnests:** [Display> Show/Blank Rats> All/Net/Comp...](#)


**Layer colors** can be changed in the color dialog with the color (old ) button. Layer visibility can be changed on the **Visibility** panel, by layer and object (class) type. Temporarily enable/disable specific layer visibility as needed, by single clicking on the color rectangles. There is a drop-down menu for pre-defined layer sets.

**Object Coloring:** Nets can be displayed either in layer color, or net color. Net coloring mode/command is selected with  (old ) and decoloring with  (old ). First we enable the coloring mode, then we click the object type filter on the find panel, then either click on an object in the drawing, or find by name on the Find panel (select type: Symbol or Net, then enter a partial name extended with "\*").

All Allegro edits are based on **object filtering**. Before moving, selecting, deleting objects, we can disable for example components on the **Find panel**, to avoid deleting/moving them.

The **info button**  (old ) is used to check what we are seeing on the screen, like component refdes or part number, or net name; select filter on the find tab, then click an object, and a popup will tell what it is. We can also use this to find/search objects, by pressing the info button, then on the Find panel find by name (select type: Symbol or Net, then enter a partial name extended with "\*").

**Measure** distance or object size: Use the  (old ) button.

**Component placement:** First we need to place all parts next to the board using [Place> QuickPlace](#), click Place, then Ok. After this we move component symbols from the auto placed area to the real board design. Click the move button , then the object filter (a "symbol" is a component =on, others =off), then left click/hold the comp and move it. Once we moved enough parts and want to do something else, [rightclick>Done](#). The rightclick menu has many options, but the most important are Done, Cancel, and Oops (the last click undo), that are also used for all command types. Rotate with [rightclick>Rotate](#), or move to bottom side with [rightclick> Mirror](#). Moving refdes, vias, traces works the same way, with the object filter set accordingly. When moving or editing, we can set the [rightlick> SnapPickTo](#) object types or off-grid. If the refdeses are too large, then we can [Edit> Text](#), set the new size on the Options or Properties panel, then click the refdes text. We can lock a component to prevent moving it, by typing "fix" in the bottom command window, enter, then on the Find panel

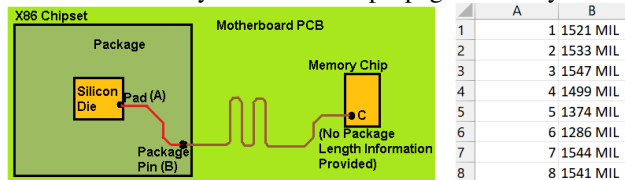


select “symbols”, then click on the part. The reverse is typing “unfix”, then click a part or [rightclick>UnfixAll](#). Component spacing constraint is set in [CM> Manuf> DesignforAssy> PkgToPkg Spacing](#).

We can **place footprints** for floorplanning from: [Place> Manually> advanced> Lib=on](#), then PList tab, dropdown = PackSym, select checkbox next to the symbol name, click on layout. We can also place mounting holes and fiducials from here, although using schematic is preferred. MNT holes can also be placed by copying a via, then editing it in the [Tools> Padstack> Modify Design Padstack](#), then assign GND net.

**Design Constraints:** We set all of them in the Constraint manager. We should set up at least some basic ones like clearance, trace width, via (browse), shape connect, and diffpair rules.

The signal length (**Package Length** (PL) or Pin Delay A to B on diagram) within large BGA packages have to be entered for accurate length tuning later. We do this in the Allegro layout, not in the schematic. We might have to enable it in [Setup> Constraints> Modes> Electrical> Pin Delay> include =on](#), and also [Zaxis> include=on](#). Prepare the PL data in excel (same unit as the tool, mils), sorted by pin number, in 2 columns (pin number, delay ”123 MIL”), and save it as a .CSV file. In Allegro [File> Import> Pin Delay](#), then click the component. The PL shows up in the CM but only at the relative propagation delay worksheet.



**Interactive routing** starts with the (old ) icon. Zoom in Allegro works by scrolling, or by click/release the middle button wheel and draw a zoom rectangle. Pan works with middle/when button press and hold while moving the cursor, or by arrows on keyboard. Set the grids to about 1mil for routing ([Setup>Grids](#)). Active layer is selected on the Options tab, but we also select a second layer, in case we want to place a via while routing, it will continue on that second layer. Routing mode can be controlled from the properties panel during editing, for example push/shove, width, layer, via padstack, angles, corners. We can select the (old ) button to slide/edit traces, or the button (old ) to tune the trace length. Allegro calls the routed traces either as net, or “cline”, while a part of the cline is the “cline segment”, these are all selectable on the Find panel. To delete anything, we have to select the (old ) delete command first, then Find filter, then click. To delete voids: [Shape> Manual> Delete](#).

Add **Vias**: doubleclick while routing. The via size and padstack is selected on the Options panel. We could also use GSSG via structures while routing by [Rightclick> ReturnParth](#), using a template drawn earlier manually then recorded by [Route> Struct> Create](#). Ground stitching vias can be placed by copy/paste a routed via, then reassign net on the Options panel, or [Place> Via Array](#).

**Vias and Component-Pads** in Allegro are handled through the padstack editor, separate padstack .PAD files are called in the footprints, but we can edit them in the layout using [Tools> Padstack> Modify Design Padstack](#), then on the Options panel select definition (all) or instance (just the selected one), click edit to open the padstack editor. The via tenting of soldermask (or expansion) is also set in the padstack editor. On modern designs

with VIPPO we do complete tenting on top/bottom (no object on SM layer). On cheaper non-VIPPO designs we need complete tenting on top-SM (no object, especially under BGAs) and a small opening on bottom-SM (for outgassing) in their padstack designs. We have to create vias before using them in the layout, in a separate program in [Start> Cadence> Pad Stack Editor](#), and saved into our library. Vias in Allegro PCB editor can display a layer span label like “1:8”, if enabled in [Setup> Design Parameters> Drill Labels](#). Here we can enable display of holes, and filled pads too. To use microvias, we have to create as many padstacks in the library as valid microvia layer pairs we have in the vendor stackup. Then in [CM>Physical](#) vias list has to include all of them (doubleclick on the cell to open via list selector).

**Fanout:** Set the constraints like width and vias, spacing first. On the part [Route> Create Fanout](#), set the options panel (via, fanout style), then click the component.

Draw Power **Shapes**: on any layer, for power delivery or VRM circuit power nets. Select with [Shape> Polygon](#), then on the Options panel select layer, parameters and net, then click to draw it. Voids can be drawn with similarly using [Shape> Manual Void> Polygon](#) or Rectangular, then set parameters on the Options panel, draw it, then [rightclick> Assign Net](#). A shape in Allegro can be dynamic or static-solid. Dynamic pours around objects. Some of the shape rules and behaviors (like thermal/direct via contact, and spacing/clearance oversize) are set in [Shape> Global Dynamic Parameters](#). Also here, we need to update all dynamic shapes often, especially before review and fab-out. To edit an existing one: [Shape> Select Shape or Void> click](#), then [Shape> Edit Boundary](#) or Merge, or [rightclick>Assign Net](#).

**Keepout** shapes: We place regular shapes on keepout layers, like via keepout class, having subclasses associated with every routing copper layer. Component keepout, route keepout.

We can **specify areas** where different rules would apply locally. These areas are called „Regions” and can be specified in the [CM> Physical> Region> All Layers](#), add constraint values (like neck down trace width, or smaller via pad for “CLASS-3 with exception”). Then we can use [Shape> Polygon](#) to place a shape on a Constraint Region class and whatever subclass (like layer 1), and assign a CM-created region on the Options panel.

Power **plane layers**: Allegro prefers positive planes, as set in the stackup. Although negative can also be used. On positive plane layers we place regular shapes, from the [Shape>](#) menu. We have to keep them off the board outline by 1mm.

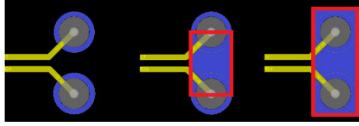
**Plane voids**: All signal vias and through pins passing through planes will have a circle antipad. The antipad size is defined in Allegro by a combination of parameters from multiple places: The padstack editor (antipad size and BD clearance), the clearance constraints ([CM>Spacing](#)), pad suppression, and the shape parameters (oversize parameter and drop-down void-mode selector “DRC vs padstack”, in [Shape> Global Dynamic Parameters](#)). Whenever we alter padstacks, voids or traces near shapes, we have to also update the backdrills (BD setup dialog) and the shapes (in global dynamic parameters). We have to use the measure tool to verify the actual AP size.

Allegro does not automatically remove all non-functional pads (NFPs), but we can “suppress pads” to reduce the Swiss-cheese effect. It is useful for signal layer rout-ability under finer pitch BGAs, and for via impedance for SERDES. It also creates smaller antipads. To suppress NFPs we have to enable the "Unused Pin Suppression" and "Unused Via suppression" in [Cross-section](#)

**Editor**, per layer, and for the specific vias that need it "Suppress unconnected internal pads" enabled in the **Padstack Editor**.

Backdrilling requires a large BD-antipad  $AP = BD + 2 * \text{drillclearance}$ . Allegro places a route keepout circle automatically on all affected layers, that DRC-checks any traces and pushes the planes. The drill clearance is usually 5...10mil on each side, depending on our fabricator. The BD tool size is usually 6...12 mil larger in diameter than the via drill size (BD oversize).

Dual-**voids** are required for high-speed diffpairs on plane layers. On Allegro's positive plane layers this means drawing **Shape> Manual Void> Polygon** or Rectangular. We could also draw a regular shape on the route keepout layer instead. Since we need



these on all ground planes, we can use **Edit> Zcopy**. We can also copy this to signal layer route keepouts, to prevent signals passing between p/n vias. To delete a void: **Shape> Manual> Delete**.

**DRC** check: **Tools>Update DRC**. The list of DRC violations should be worked down to zero by interactive layout editing, except a few items that are reviewed and accepted/waived. The **Tools> DRC Browser** lists all remaining ones. The CM also shows all the signals and their relations to constraints with actual length values with green/red coloring for pass/fail. Allegro also has DRC markers per layer that can be enabled for visibility on the Visibility tab. They look like a bow tie; we can use the info button on them if we selected DRC errors on the Find filter. Spacing is enforced during editing, if **Setup> Enable Online DRC** =on. What to include in DRC: **Setup> Constraints> Modes**. To find unconnected pins, **Tools> Quick Reports> Unconnected Pins**.

**Thieving** can be applied in Allegro, using **Manufacture> Thieving**. We could also use polygon pours between traces, or rely on our PCB fab to apply thieving for us.

Preparing for **manufacturing**: Every layer should have text outside of the outline about layer name, layer number, whether it is upside-down (mirrored), company info and design part numbers. On one mechanical layer (and Gerber layer) meant for fab drawing, we place tables for drilling, impedance and stackup, then we manually add text about technology statements, materials, surface finish, coupons, impedance and loss tables. The drill chart/table is created using **Manufacture> NC> Drill Legend**, on a MANUFACTURE / NCLEGEND-xx layer. We also create an assembly drawing on another mechanical (and a Gerber layer), for verifying the build, by using a dimensioned DXF from our ME. On mech layers we can place line or text. **Add>Text** or **Edit>Text**, **Add>Line**. Once all done we fab-out: **Manufacture> Artwork** to set up and generate Gerber file layers as combinations of Allegro Class/subclasses. We also generate drilling files with **Manufacture> NC> NC drill**, but enable the checkbox "include backdrill", [...] to create a file name, then press [drill]. We generate a pick and place file using **File> Export> Placement** (origin = body center), or **Tools> Reports> Custom**. We send a 3D drawing to our ME to check system fit: **File> Export> IDF** (EMN). PDF fab and assembly drawings to be exported from **File> Export> PDF**, and select which Gerber layers to be used (one Gerber layer per PDF sheet).

The **Gerber films** have to be set up: We have to create extra Gerber film layers, for example silkscreen, solder mask and paste, fab notes and assembly notes. In the colors dialog turn off all layers, then make only the layers that we want to add to a new

Gerber layer visible. Then **Manufacture> Artwork**, then on one of the existing films **rightclick> Add>** enter name. Redo the above for all needed layers. Then back to the artwork dialog, **rightclick> Select All**, then click [**create artwork**]. This will generate several files with the extension .ART, to be sent to the PCB fab vendor.

We can export **data reports** from the **Tools> Reports** menu. Select the desired report types and run it. We commonly export "Etch Length by Pin Pair", and "Component Pin" reports for hardware engineering analysis, DRC report or "Unconnected Pin" report for helping finish up the layout. These are HTML, but with a CTRL+C we can copy it and paste it in Excel.

## 2.6. PCB Library (Allegro footprints)

To make our own footprint symbols, we have to create a new **DRA file** in Allegro. Each pin in the footprint is a padstack, that has its own file. Within the padstack, if we want to use a thermal relief plane connection, then we define the pad diameter here, and the **Shape> Global dynamin param** describes the spoke size.

**Padstacks** have to be designed for pins, before using them in footprints, and also save them into the library. From the **Start menu> Cadence> Padstack Editor. File> New**, select SMD, via or through type. On different tabs we set different parameters. We also have to specify pad size and antipad size on every layer type separately. SMD will only be on etch/top. Backdrill enable, solder mask and paste mask is defined too. We can enable NFP removal (pad suppression). **File>Save**, in to a .PAD file.

To make the **footprint** in Allegro: **File> New> Package Symbol**. In that **Layout> Pins**, then on the Options panel we set how many pins (quantity X & Y), what padstack, then click the drawing and place all pins at once, **rightclick> Done**. Place an outline with **Add> Line**, on the Options tab select layer Package Geometry / Silkscreen-Top, then draw the body. Add pin1 marker dots, copper corner marks, numbers or anything that's needed. **Setup> Change Drawing Origin**, to the center or pin1. **Setup> Areas> Package Boundary**, draw it, it will be used with placement DRCs. Similarly, **Setup> Areas> Package Height**. Save it. It will save both the .DRA file and a .PSM file that we will use in the layout.

An old free program called **FPM** Allegro Footprint Maker could generate complete footprint and padstack files.

For large BGAs with **irregular** pattern, we should get an allegro reference board design file .BRD. Then export footprints with: **File> Export> Libraries** into a folder.

## 2.7. Design Reuse (OrCAD)

Design-reuse with a project is done using hierarchical designs, with single/multiple instances of the same sch **sheet symbol**. The hierarchical block (sheet) symbol parameters dialog has 3 drop-down menus. In the first one (Implementation Type) we select "schematic", in the second (Impl. Name) we select the SCH folder name for in-design hierarchy, or leave it blank and browse to a saved external reuse block in the third (Path) menu.

To edit the **block's schematic**: **rightclick> Descend Hierarchy**. The block should not contain global power symbols, except GND, rather use ports to deliver power nets and all signals. Once instantiated, the refdes and net names will be overridden as netname\_instancexx. Instance and occurrence property.

Reusing designs **from external projects** can also be done by creating a separate OrCAD project with a .DSN schematics, an


.OLB library file and a .BRD layout file (in the allegro subfolder), a generated .MDD board snippet file. All with the same filename. We will be referencing it in a hierarchical block properties. The properties will look like this: Implementation Path property = "C:\path\ filename.DSN", the PCB Footprint property = "filename", the Source Library = "filename.OLB". If we have multiple copies of a block, then we will see yellow "occurrence" property columns, one for each copy, and a white instance property column for the overall.

### 3. High-Speed Signal Objects

The connection objects define the signals, or groups of signals. They can be browsed in the CM. Categories:

- ❖ **Net**: created using net labels in schema. Net length does not include via length and package length.
- ❖ **Bus**: create using bus symbols in schema, it uses indexed net names, like ABC1\_[7:0] → ABC1\_0. For serdes-based interfaces, it is better not to use buses, unless we have a hierarchical schematic on a large line card design.
- ❖ **Diffpair** (DPr): Create them in the CM Electrical/ Diffpairs worksheet with [Objects> Create> Differential Pair> Auto Setup](#), then in the setup dialog w specify the \_P and \_N suffixes, and then press [Create].
- ❖ **Net Class** (NCI): A group object, used for trace width (impedance) related constraints in Allegro. In [CM> Physical> Net> All Layers](#) worksheet, select all needed nets, then [rightclick> Create> Class](#).
- ❖ **PinPair** (PPr): This is to control propagation delay from a pin and another pin on a multi-point net. On a DDR4 fly-by address bus, on every address net we would have several PinPairs from CPU-DRAM0, CPU-DRAM1... CPU-DRAM7. PinPairs should be created in [CM> Electrical> Net> Relative Prop](#), by selecting a net, [rightclick> Create> PinPair](#). They only appear in relative prop delay and min/max prop delay rules. PPr length does include via length and package length.
- ❖ **Matched group** (MGrp): We create one MGrp for all the PinPairs of all address nets between CPU-DRAM1. Then enter a length matching constraint number set for that MGrp. Another MGrp and a numbers entered for CPU-DRAM2. MGrp's should be created with the [CM> Electrical> Net> Relative Prop](#), by selecting one or more PPr's, [rightclick> Create> Matched Group](#).
- ❖ **XNET**: XNETs are for two nets passing a series passive part. First the components have to have .DML models assigned to them, in [Analyze> Model Assignment](#), then they are auto created and appear in CM.

### 4. Constraint Manager

**Open** the Allegro CM: [Setup > Constraints> Constraint Manager](#), or use the  button (old ), for design rules.

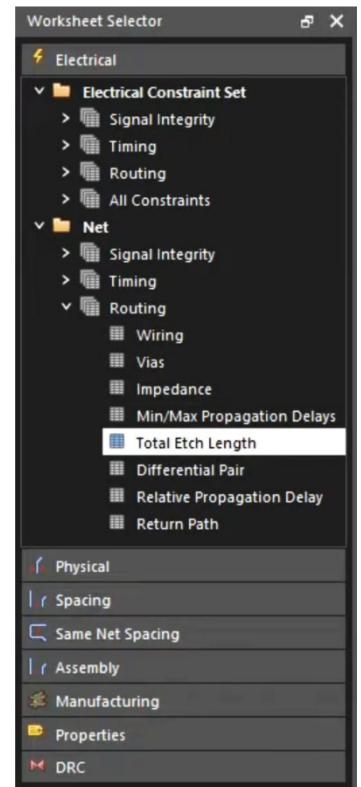
It has several types of **worksheets**: Electrical (lengths), Physical (width), Spacing, Same Net Spacing, Properties. Each type (domain) has a tree structure of categories (folder/ workbook/ worksheet).

We can create **Constraint sets**, that are sets of rule-number values, that can be applied to multiple objects (Net, DPr, PPr...) instead of typing those numbers in for every object separately.

ECSETs in the Electrical worksheets, PCSs on the physical trace width sheets, SCS in spacing. All CM categories have a tree structure of worksheets, half of the tree is for CSETs (one row is one CSET) and the other half called "Net" (one row is one object) for individual entry or CSET application (CSET name from a drop down). One parameter has one column. Some constraint domains also have a sub-tree for regions.

In the **Spacing Worksheet**

we can define a clearance between Net or net class and other objects by object type (pad, trace, via, shape), in a matrix. Constraints sets (SCS) can be created here too. Spacing in PCB design has two aspects: manufacturability and crosstalk levels. For the first aspect, we set up a [CM> Spacing> SCS> All Layers](#), which is normally the min spacing that our PCB manufacturer recommends. Basic spacing is enforced by a push/shove force in interactive routing. The [CM> Electrical> Routing> Differential Pair](#) rules also contain a field called „Gap“, but this is a related to the differential impedance of the diffpair. We can enter more spacing object type combinations, if we select a column header, [rightclick> Show More](#).



Type	S	Objects Name	Referenced Spacing CSet	Line To	Thru Pin To	SMD Pin To	Test Pin To	Thru Via To	BB Via To
				All	All	All	All	All	All
*	*	*	*	mil	mil	mil	mil	mil	mil
Dsn	*	buenosboard1	DEFAULT	4.00	4.00	4.00	4.00	4.00	4.00
NCIs	*	MDQ_WIDTH(32)	DEFAULT	4.00	4.00	4.00	4.00	4.00	4.00
NCIs	*	MEMAD_WIDTH(24)	DEFAULT	4.00	4.00	4.00	4.00	4.00	4.00
DPr	*	D MEMO_CK	DEFAULT	4.00	4.00	4.00	4.00	4.00	4.00

The **Max Parallel** constraint is used for crosstalk control spacing: [CM> Electrical> Net> Routing> Wiring> Parallel-column](#). This rule only checks spacing if the two traces run in parallel longer than specified. We have to enter a weird string into this field, containing up to 4 length/separation number pairs: "<len1>:<space1>;<len2>:<space2>". For example, "60 mil:4 mil; 200 mil:8 mil;16000 mil: 16 mil", which means up to 60mils parallel length we enforce 4mil space, 60...200 mil length we enforce 8mils, between 200mil and 16 inches we enforce 16mil. If we click on the field, we can set them up in a box/matrix window.

In the **Physical Worksheet** we can see most signal object types (DPr, Net, NCI, bus) in a tree hierarchy browser view, and specify trace width and a via padstack file to them. Net classes are created here by selecting multiple nets, [rightclick> Create> NetClass](#). This way we only have to apply constraints to their members in one row. The diffpair constraints reappear here in the physical worksheets, although we prefer to set them in the Electrical> Routing worksheet. We either enter the constraint numbers for each row in Phys> Net, or select a physical constraint set PCS. Normally we put a group of nets based on characteristic impedance into a Net Class. Then we set up a PCS for every class separately, and also a default width rule for all other or non-impedance controlled traces



(4mil). The PCS folder has 2 worksheets, “All Layers” and “By Layer”. The All Layers has entries like “4.5:5.2:4.0:5.2...”, but clicking the arrow opens a tree/table showing regular values per layer. We should use width rules instead of impedance rules, because we have to use the fab vendor’s impedance/width/space calculations (from the negotiated approved stackup document), instead of Allegro’s calculator.

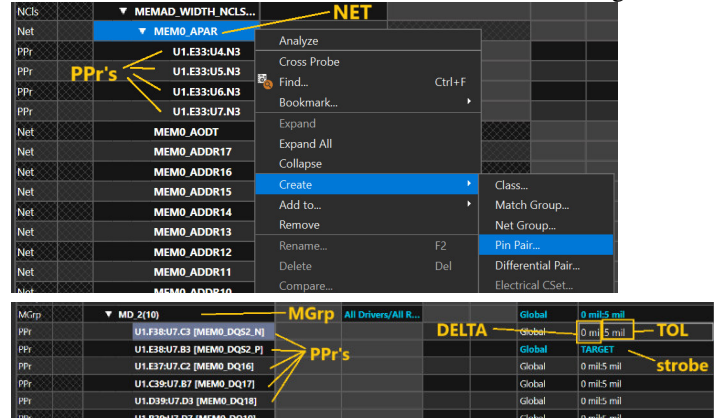
The **Electrical worksheet** is for high-speed design trace length constraint categories. Its constraint set is called **ECSETs**. Both the ECSET and the Net has the same sub categories. On the Net tab the most used categories are the “Differential Pair” (phase tolerance, trace/gap, uncoupled length), the “Total etch length”, the “Relative Propagation Delay” (for matching trace lengths in MGrp’s) and the “Min/Max Trace length”. The PPr/MGPr/DPr objects are created in CM by selecting multiple objects and **rightclick>Create>...**

The **Electr> Routing> Differential Pair** category is for phase tolerance length matching, diffpair gap and uncoupled length constraints (numbers), applied to diffpairs. We either create a new ECSET for each diffpair impedance type (like 85, 93 Ohm...), or we just enter the constraints (numbers) into the rows of specific diffpairs. We set static phase typically 5mils. If we use dynamic phase tolerance, then usually 5mil on 5” “max length”. The width is inherited from the physical worksheets. Saved GSSG via structures can be added to nets in **Electr> Routing> Vias> ViaStr**.

The **Electr> Routing> Relative Prop Delay** category is for trace length matching constraints (numbers), applied to matched groups (MGrp’s). MGrp’s are usually applied to PinPairs, but in some cases we can apply them to nets, XNETs, buses or diffpairs too. We can use this for single-ended buses, or lane-to-lane matching of diffpairs. The diffpair P-to-N phase tolerance matching is not here, it is on the diffpair worksheet. Typically, we enter a delta and a tolerance value as numbers, to the MGrp, then all members immediately inherit these numbers below. Tolerance is the max deviation, while delta means a fixed offset. For example, if we want SIG1 to be matched to SIG2 within 5mils, while SIG2 is already routed to 200 mils, but SIG1 needs to be 50 mils longer, then the correct range for SIG1 will be 245...255mils once routed and tuned. The constraint format to enter is “50 MIL:5 MIL”. One member of the MGrp can be the main signal (like strobe on a DDR4 byte lane), so for that one we can override the inherited value and type in text “TARGET”.

**PinPairs and Matched groups**, as well as their associated design rules could be created in the constraint manager, Rel Prop Delay worksheet. We can create PinPairs from nets (rightclick> Create), and then select those PPr’s and create a MGrp for them (rightclick> Create). To speed up PinPair creation, we create all useful pin pairs on one net, for example CPU-DRAM0, CPU-DRAM1..., by selecting the first net name **rightclick> Create> PinPair**, select the CPU refdes on the left, all DRAM refdes on the right, OK. This creates as many PPr’s as the number of memory chips on the net. For a DDR4 mem-down fly-by address bus we would have several PPr’s on a net, for the data bus we would have one PPr on a net. Next, we create a separate MGrp for each new PPr by clicking on the PPr then **rightclick> Create> MGrp**, give it a name, redo for all PPr’s on the net. Then on the net name again **rightclick> Create> Electrical CSET**, give it a name. Then click the ECSET column for the net and select the new ECSET’s name from the drop-down menu. Then select all other net names that we wanted to be in the same group, click the ECSET column and select the new ECSET from the drop-down menu. It creates the same

topology pin pairs for all selected nets. It will also put all of them into the right matched groups like the first net was. If we need multiple independent MGrps, like separate byte lanes on a memory interface, then we will use a separate ECSET for each lane. Basic nets do not include via length and package length, only PPr’s do, so we have to use PPr’s in MGrp’s, even on point-to-point nets. We can still use Nets for SERDES link lane-to-lane matching.

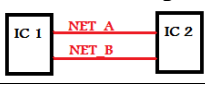
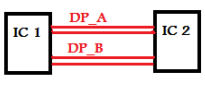
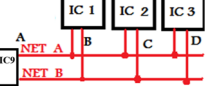
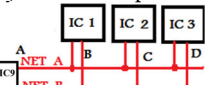


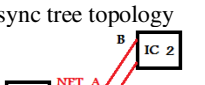


The **Electr> Routing> Min/Max Propagation Delay** category is for specifying min or max trace lengths for synchronous or asynchronous buses, so they can meet setup and hold timing. We can create ECSETs for them, and apply it to as many PinPairs, nets, buses, or diffpairs as needed. On a multi-drop PCIX bus we would need constraining from CPU to each target device, as PinPairs. We can specify max length in the MinMax Prop (supporting PinPairs and package length) or in the **CM> Electr> Routing> Total Etch Length** (nets only, no PL). For SERDES diffpairs we might need loss-budget-based max length constraints. At 200Gbps/lane the budget includes the PL too, while below that PL is excluded.

Maximum **Length constraint values** for SERDES buses come from insertion loss budget calculations. The dB/inch loss data, specific to a fabricator and material combination, comes from VNA measurements on Delta-L test boards, then the max trace length is calculated as  $L < \text{budget} / \text{dBpi}$ . The total budget comes from the relevant standards like IEEE802.3xx, or SFF8418. For synchronous, asynchronous, source-synch and clock forwarding buses the rule values are either obtained from the chip vendor’s datasheet or design guide document, or we calculate them using a pre-layout setup/hold timing analysis calculator spreadsheet, like: [https://www.buenos.extra.hu/romanyok/PCB\\_Timing\\_analysis.xls](https://www.buenos.extra.hu/romanyok/PCB_Timing_analysis.xls)


**Typical high-speed objects and constraints:**


Case	Objects	Constraints entered in CM
Differential Pair point-to-point signal 	<ul style="list-style-type: none"> <li>•DPr</li> <li>•ECSET</li> </ul>	<ul style="list-style-type: none"> <li>• CM&gt; El&gt; Rout&gt; Diffp (DPr)</li> <li>• CM&gt; El&gt; Rout&gt; Wir&gt; Paral. (DPr)</li> <li>If &gt;8Gbps: <ul style="list-style-type: none"> <li>• CM&gt; El&gt; Rout&gt; TotalE (DPr)</li> <li>• CM&gt; Prop&gt; Comp&gt; Pin&gt; Manu&gt; backdrill (comp pin)</li> <li>• CM&gt; Prop&gt; Net&gt; Gen&gt; Backdrill (Net)</li> </ul> </li> </ul>
Single-ended Sync/ Async point-to-point bus with min/max len. 	<ul style="list-style-type: none"> <li>•ECSET</li> </ul>	<ul style="list-style-type: none"> <li>• CM&gt; El&gt; Rout&gt; Wir&gt; Paral. (Net)</li> <li>• CM&gt; El&gt; Rout&gt; Minmax (Net)</li> </ul>

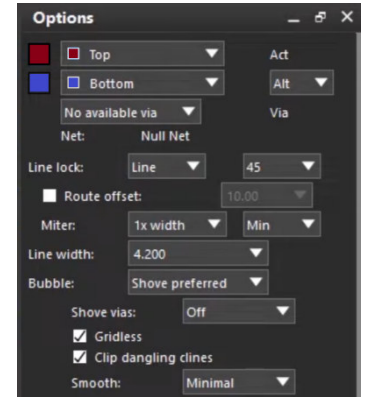
Single-en Source-sync point-to-point bus with matched lengths 	<ul style="list-style-type: none"> <li>•PPr</li> <li>•MGRP (1/lane)</li> <li>•ECSET (1/lane)</li> </ul>	<ul style="list-style-type: none"> <li>• CM&gt; El&gt; Rout&gt; Wir&gt; Paral. (Net)</li> <li>• CM&gt; El&gt; Rout&gt; Relat (MGrp of PPr)</li> </ul>
Multi-lane Point-to-point diff SERDES bus 	<ul style="list-style-type: none"> <li>•DPr</li> <li>•MGpr</li> <li>•ECSET (1/port)</li> </ul>	<ul style="list-style-type: none"> <li>• CM&gt; El&gt; Rout&gt; Diffp (DPr)</li> <li>• CM&gt; El&gt; Rout&gt; Wir&gt; Paral. (DPr)</li> <li>• CM&gt; El&gt; Rout&gt; Relat (MGrp of Nets)</li> <li>If &gt;8Gbps: <ul style="list-style-type: none"> <li>• CM&gt; El&gt; Rout&gt; TotalE (DPr)</li> <li>• CM&gt; Prop&gt; Comp&gt; Pin&gt; Manu&gt; backdrill (comp pin)</li> <li>• CM&gt; Prop&gt; Net&gt; Gen&gt; Backdrill (Net)</li> </ul> </li> </ul>
Single-ended Sync/ Async multi-drop bus 	<ul style="list-style-type: none"> <li>•PPr</li> <li>•ECSET (1/bus)</li> </ul>	<ul style="list-style-type: none"> <li>• CM&gt; El&gt; Rout&gt; Wir&gt; Paral. (Net)</li> <li>• CM&gt; El&gt; Rout&gt; Minmax (PPr)</li> </ul>
Single-ended Source-sync multi drop bus 	<ul style="list-style-type: none"> <li>•PPr</li> <li>•MGRP (1/chip)</li> <li>•ECSET (1/lane)</li> </ul>	<ul style="list-style-type: none"> <li>• CM&gt; El&gt; Rout&gt; Wir&gt; Paral. (Net)</li> <li>• CM&gt; El&gt; Rout&gt; Relat (MGrp of PPr)</li> </ul>
Diff multi drop bus (like RS485) 	<ul style="list-style-type: none"> <li>•DPr</li> <li>•PPr</li> <li>•MGRP (1/chip)</li> <li>•ECSET (1/bus)</li> </ul>	<ul style="list-style-type: none"> <li>• CM&gt; El&gt; Rout&gt; Diffp (DPr)</li> <li>• CM&gt; El&gt; Rout&gt; Wir&gt; Paral. (DPr)</li> <li>• CM&gt; El&gt; Rout&gt; Relat (MGrp of PPr)</li> </ul>
Mixed SE/Diff multi-drop matched lengths 	<ul style="list-style-type: none"> <li>•DPr</li> <li>•PPr</li> <li>•MGRP (1/chip)</li> <li>•ECSET (1/bus)</li> </ul>	<ul style="list-style-type: none"> <li>• CM&gt; El&gt; Rout&gt; Diffp (DPr)</li> <li>• CM&gt; El&gt; Rout&gt; Wir&gt; Paral. (Net)</li> <li>• CM&gt; El&gt; Rout&gt; Relat (MGrp of PPr)</li> </ul>
Single-ended Source-sync tree topology 	<ul style="list-style-type: none"> <li>•PPr</li> <li>•MGRP (one/chip)</li> <li>•ECSET (1/lane)</li> </ul>	<ul style="list-style-type: none"> <li>• CM&gt; El&gt; Rout&gt; Wir&gt; Paral. (Net)</li> <li>• CM&gt; El&gt; Rout&gt; Relat (MGrp of PPr)</li> </ul>

They also need net classes, for spacing and physical/width constraints sets (PCS and SCS), if impedance controlled.


## 5. Interactive High-Sp Route

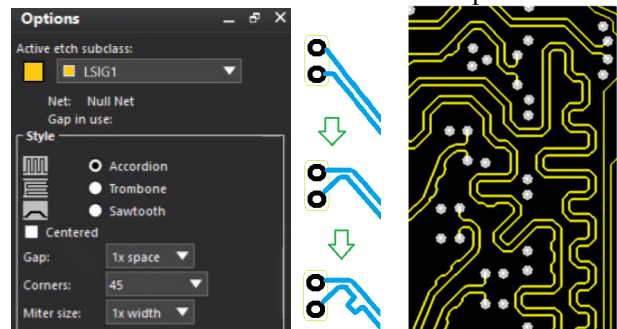
During interactive **routing**, the Options panel should be visible, that allows us to alter routing parameters like width, active layer, angle, corner style and vias. We can manually route traces by left clicking the routing mode button , then clicking a pad, then start pulling the trace. During routing the right click menu offers important commands, like cancel, done, oops (undo the last click), next (net), snake routing mode, switching between “single trace mode” (twist one leg of a diffpair) and default diff routing. For diffpairs it will automatically use diffpair routing, unless we click single trace mode. While routing we can doubleclick to place a via

and change to the alternate layer (shown on the options panel). We can slide traces by selecting the slide menu  first then clicking the trace. First we route all traces with plenty of spacing, then we tune them later. We might want to hide most connection lines (ratsnests), to see clearly: **Display> Hide Rats> All**, then enable a few on the PCB drawing after selecting the **Display> Show Rats> Component or net**.

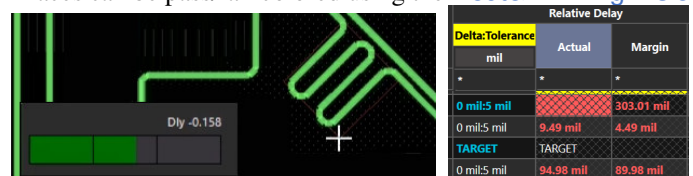


For several constraints (physical, spacing, diffp) there is a primary and a **neck-down** width/gap. During routing, especially while under a BGA, we can **rightclick> Neck Mode** to switch to/from the neck-defined width/gap. We could also use regions.

**Length tuning:** We can delete then re-route trace segments, auto-meander traces with the delay-tuning button , or slide them (select, drag) to increase/decrease the signal length. While starting the tune, after pressing the tuning button, the Options panel displays meander pattern and parameters. The tuning can be ended by the **rightclick> Done**, cancel or Next. Single-ended and diff traces are both tuned by the same tuning button. We can just add phase tolerance tune bumps by enabling **rightclick> Single Trace Mode**, or disabling it and tune the lane-to-lane matching. We can also do phase tolerance matching using **Route> Phase Tune** (instead of Route> Delay Tune). The first thing for phase tolerance is to twist at the pads, only after that we use meanders. **The Route> Auto-Interactive** tunes multiple selected signals.



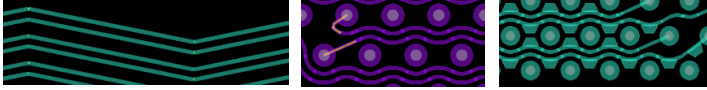
The Allegro on-screen Length Meter/ Gauge pops up during sliding/tuning the traces, which **measures the signal lengths** of the currently edited trace in real time, relative to all active constraints. The CM also shows all signal object type lengths as a table, but first we have to enable it in **CM>Analyze> Analysis Modes> Electr> RelativeProp =on**, all others we need also on, then **OnLineDRC =on, OK**). Then we can measure by selecting cells in CM and **rightclick> Analyze**. Must have a constraint entered. Package and via length is only included in PinPairs, not in nets. Traces can be pass/fail colored using the **Route> Timing Vision**.



For long 10Gig+ SERDES signals, we use **wavy routing** and odd-angle routing, to mitigate fiber weave effect. In Allegro angled

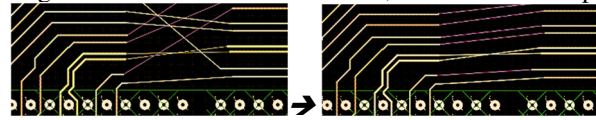


routing is done by changing the Line Lock angle to “off” instead of 45deg, on the Options panel before routing. Click the route button, then options, then click the trace. Wavy routing is achieved by enabling the [route offset] checkbox. We can also do it by [Route> UnsupportedP> Fiber Weave Effect> Add zigzag](#). Curved routing is enabled by selecting arc instead of line, on the Options panel Line Lock section. SERDES links should always use the curved corners. We can enable curvy snake fanout under offset-grid hex-BGAs, while routing we can right [click>Snake Mode](#) to enable it. Intel-style “tabbed routing” can be added to memory traces to lower their impedance in the escape route pin field, using: [Route> Unsupported Prot> Tabbed Routing> Generate](#).



For low-cost and aerospace boards **teardrops** are used, that we enable: [Route> Gloss> Param> Fillet> Dynamic=on> OK](#).

**Pin swapping** might be required on many designs, if a parallel bus seems un-routable due to connection-line (ratsnest) crossings. With hard chips like CPUs the datasheet might tell us which pins can be swapped with which (within groups), while with FPGAs we can likely swap most signals (maybe except the diffpairs that must be on diff capable pin pairs), if we also update the FPGA pinout file. In Allegro, once we are done with the escape routes and some long distance routes from both ends, then we can set up swapping.



Steps for swapping in Allegro: In the schematics symbols the pins have to be allocated pin groups. In the Allegro PCB editor, [Place> Swap> Pins](#), then select 2 pins to swap, swap group members will be highlighted. Repeat until all look good, [rightclick> Done](#). Then either we back annotate the data into the schematic, or generate a swap report ([Tools> Quick Reports> Pin Swap Report](#)) and manually update the schematic (by editing net labels or off-page) and export new netlist from SCH-to-PCB.

If we have used bus symbols, then implementing swaps in schematic might be harder. To help it, we can create separate net **swapping pages**, where 2 offpage symbols are shorted to the same wire, the left side goes to one component, the right to the other, with similar signal names but with a prefix. The bus offpage (OUT[1:2]) on the device page splits to separate net offpage symbols (OUT1) on the swap page. Then swaps can be implemented by editing the signal names on the off-page (=select multiple, rightclick> edit properties). Then copy the table with CTRL+insert, paste it in Excel, edit there, copy, then back paste with shift+insert. We will have something like this: OUT1>---<COUT2 and below it OUT2>---<COUT1.

**Backdrilling** is set up in 2 places together: First the stub constraint is created in [CM> Properties> Net> General Properties> Backdrill](#), by typing in the value for the SERDES nets that need it. The maximum value means any stub longer than that will be backdrilled. The constraint is set slightly longer than what we write in the fab notes. Up to 64Gbps we don't backdrill from bottom to L(N-2) routes, that results in a 2-layer deep stub (6...16mil), so we set the constraint slightly longer than that. In the second step we open the [Manufacture> NC> Backdrill Setup](#), and we create all backdrill layer pairs automatically, set up BD oversize (diameter) and other parameters, then press the Backdrill

button then OK. The BD antipad size is set in the footprint's padstack design. BD always starts from (Top or) Bottom, and ends one layer away from our routing MNC layer. We should enable the “drill labels and backdrill holes” in [setup> User Parameters](#). This will put a B40-28-27 style text on it, meaning layer 40 to 28, with layer 27 being MNC. Once the design is done, we have to go back and press the backdrill button again, to update (depths, drills, voids and keepouts). Then we manually review it: Enable one routing layer to be visible, then also enable a layer manufacturing/ncbackdrill-x-y, then check if every high-speed diffpair ends with a BD symbol. A separate NC drill file will be generated for each BD depth. Note that the [CM> Electrical> Routing> Wiring> Stub Length](#) is not for via stubs, but for daisy chain traces. For press-fit connectors we have to set a property to prevent drilling into the minimum barrel length area, in [CM> Properties> Component> Pin> Manufacturing> backdrill column](#). Find reffdes, select multiple pins, then exclude= “exclude top”, and enter MBL value from connector datasheet. We also get route keepouts/antipads on plane and signal layers, around backdrilled vias, as described in the section about voids.

The **CM** lists all the constraints and related DRC violations with details (e.g., deviation from preferred length). We have to select several cells, [rightclick> analyze](#). The numbers in the actual fields appear green if they meet the constraints, or red if there is a failing object that does not meet it. We can browse by category. If we fix a violation then it disappears from the list. We can also see DRC violations in the DRC report, and in the drawing as markers.

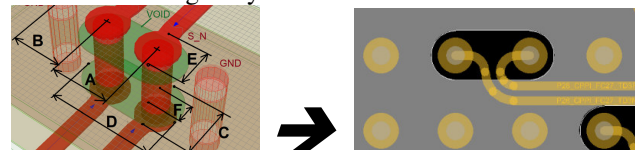
## 6. Signal Integrity

To ensure good signal integrity, we utilize high-speed design techniques, as explained in the book titled “Complex Digital Hardware Design”. It also provides guidance about architecture, debugging, simulations, measurements, constraints, timing-based trace length calculations, trace impedance control, crosstalk control, ground returns, stackup design, materials, backdrilling, via impedance optimization, loss budget calculations and insertion loss control techniques.



Most **SI simulations** should be done in proper external tools, for example pre-layout and decoupling in Keysight ADS, or post layout in Hyperlynx, HFSS or Simbeor. Power plane DC voltage drop can be simulated with the Allegro's built-in (Sigrity) IR Drop analysis, that requires a separate license. Might need to [File> Change Editor, Logic> Identify DC Nets](#), then run: [Analyze> Workflow Manager> IR Drop Workflow](#), mode=VRM/S, Options, [setup VRMs/sinks, Start, IRD Table](#).

For differential SERDES links operating at 8Gbps/lane or above we need to ensure that the **impedance of the via structures** also comply to the target impedance requirement, for example 93 Ohm diff for backplane Ethernet. We do this by recreating the via structure in HFSS or Simbeor, optimize the dimensions (via-to-via spacing, via diameter, and void shape/size), then adjusting them until the TDR response is flat enough, then replicating the structure in Allegro layout to match the dimensions.



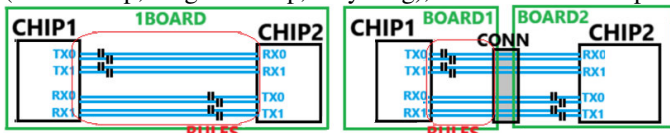
## 7. Typical Examples

### 7.1. PCIe Gen4 SERDES bus design

The lane-to-lane matching groups setup depends on the **architecture**. If we have an AC-cap then it creates a short segment between the chip and the cap, and a long segment between the cap and the other chip. We should add a net name on both sides in the schematic.



If we had a DC coupled Hyper Transport bus, then one MGrp would be enough. If our PCIe link is between two chips on the same board, then we have AC-caps on both RX and TX signals on our board. So, we would need 2 MGrp's, one group for the long segments including both TX and RX signals, and the other group is for the short segments. If we are designing a PCIe link that passes through a connector (a motherboard, add-in card or backplane system), then our constraints will only be created for the segments that exist on one board. This case we likely have an AC cap only on TX or RX on our board, and now we need 3 types of diffpairs (short to cap, long from cap, very long), so we need 3 MGrp's.



We also need to calculate an **insertion loss budget** at the Gen4 16Gbps speed. We have a budget of 25dB@8GHz. If our PCIe link is on one board, then we have the whole 25dB available, but if it goes through a connector, then we only have a portion, budgeted between 2 boards. Let's assume we have 70% available for the motherboard, that means  $25\text{dB} * 0.7 = 17.5\text{dB}$ . We have to obtain a fabricator and material related dB/inch loss data from our SI team or fab vendor, let's say we got 2dB/inch@8GHz. With 17.5dB budget and 2dB/inch we can have our max total etch length  $17.5\text{dB} / 2 = 8.75"$ . On our motherboard we have 3 groups (enter values for several DPr's), the longest one can have 8.75" max set in the **CM> Electr> Routing> Total Etch Length** worksheet, while the other 2 groups have to share that 8.75", so one would have let's say 3" max and the other  $8.75 - 3 = 5.75"$  max.

The **diffpairs** need setting up. We create the diffpairs in the **CM> Electr> Routing> Differential Pair**, and 3 net classes. Every separate refdes-to-refdes interface is a separate net class. The impedance-driven width is in **CM> Physical** (PCS on nets). The  $\_P/\_N$  phase tolerance matching is typically 5mils, that we set in **CM> Electr> Routing> Differential Pair** in ECSET applied on the diffpairs. We also set up a constraint for crosstalk control on all PCIe signals with **CM> Electrical> Net> Routing> Wiring> Parallel-column**. For any reference clocks, we only need diffpair rules and phase tol matching in ECSET, and PCS trace width.

Embedded clock interfaces (like PCIe or HDMI) usually have De-Skew circuits built-in, so they only need loose **lane-to-lane** matching, maybe within 2 inches. Clock forwarding interfaces (e.g., Hyper Transport 1.0 or XGMII) do tight matching, maybe within 5 mils. We apply this in **CM> Electr> Routing> Min/Max Propagation Delay**, on each MGrp-of-DPr's separately.

During **routing**, we route all diffpairs using interactive routing loosely. Then we match the phase tolerance within each diffpair by twisting first then single ended tuning. After this we match them

lane to lane using interactive length tuning. We can monitor our progress on the popup length gauge or in the CM. Finally, we run analyze in CM and verify all objects green.

We also need to set up **backdrilling**. In the **Manufacture> NC> Backdrill Setup**, we define all BD depths. Then we define the max stub length related constraints on nets or net class in **CM> Properties> Net> General Properties> Backdrill**, and **CM> Properties> Component> Pin> Manufacturing> backdrill** column.

### 7.2. DDR4 Memory-Down design

The „Memory-Down” is the design technique where we design a complete DIMM memory **on to the motherboard**, so we don't need to use DIMM sockets, all the memory chips will be soldered down. The design rules come from CPU design guide documents.

We have to create objects for the **address bus** signals: A net class for trace width rules, PinPairs for each CPU-to-DRAMn component pair on every address bus signal, one MGRP+ECSET (match 5mil) for each CPU-to-DRAMn component pair (containing address and clock PPr). Then we enter the width data for the net class in **CM> Physical**, and match-length data for each MGRP. If we had 25 signals and 4 DRAM chips, then we will have  $4 * 25 = 100$  PinPairs and 4 MGRPs. The PPr/MGRPs/ECSETs creation is done at once, first we manually create all PPr's on one net, then MGrp's for all the new PPr's, then one ECSET for the net, then select all other nets, apply the same ECSET, the rest is auto created. Finally enter delta, tolerance and TARGET values.

The **clock** needs diffpairs created, and a net class for trace width/impedance and diffpair parameters. We have to create PinPairs for the clocks too, while we are creating them for address bus, and they will go into the MGrp of the address bus.

**The data bus**: Since we have DQS diffpairs and DQ SE signals in one matched group, we have to create PinPairs for every signal, and MGRP+ECSET (match 5mil) for every lane. We use net classes only for width/impedance. We will need one net class for DQ width and one for DQS width. We will need as many MGRPs and ECSETs as the number of data byte lanes (containing DQ and DQS XS). These are created the same way as the address bus groups.

Type	S	Name	Referenced Electrical Cst	Pin Delay		Uncoupled Length				Static Phase						
				Pin 1	Pin 2	Gather	Length Ignored	Max	Actual	Margin	Tolerance	Actual	Margin	Max	Min	
				mil	mil	Control	mil	mil	mil	mil	mil	mil	mil	mil	mil	mil
Net		MEMO_DQ00	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ01	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ02	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ03	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ04	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ05	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ06	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ07	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ08	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ09	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ10	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ11	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ12	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ13	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ14	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				
Net		MEMO_DQ15	DIFFPAIRSROUTING	Ignore			70.00			5 mil		100.00				

Type	S	Name	Referenced Electrical Cst	Pin Delay		Scope	Delta Tolerance
				Pin 1	Pin 2		
				mil	mil		mil
MGrp		MEMO_DQ23	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ24	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ25	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ26	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ27	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ28	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ29	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ30	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ31	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ32	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ33	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ34	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ35	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ36	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ37	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ38	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ39	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ40	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ41	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ42	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ43	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ44	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ45	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ46	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ47	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ48	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ49	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ50	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ51	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ52	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ53	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ54	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ55	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ56	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ57	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ58	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ59	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ60	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ61	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ62	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ63	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ64	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ65	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ66	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ67	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ68	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ69	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil
MGrp		MEMO_DQ70	DIFFPAIRSROUTING	All Drivers/All R...		Global	0 mns mil