Altium Designer 25 Tutorial

1. Introduction

This tutorial is targeting complex high-speed digital circuit board designs with Altium Designer release 2024. 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.

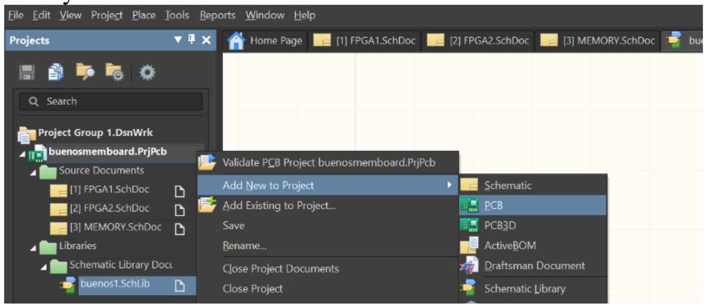
---

2. Basic PCB Design

2.1. Projects and Editor

In a **project folder** we have multiple files. One is the main project file .PRJPCB, then we have a layout file .PCBDOC, several schematic pages in separate .SCHDOC files, library files like .SCHLIB and .PCBLIB, a folder for manufacturing „ProjectOutputs". All files open in the main window design explorer, but the menus are dependent on which file type was open. A top tab can be used to select which of the open files to edit. All drawings allow CTRL+scroll for zooming.

New project: file>new>project, give name, enable CM (whether we want to use constraint manager) then "create". Create new files for the project on the projects panel, by right clicking the PRJPCB and then >add new to project > schematic, or PCB or library file...



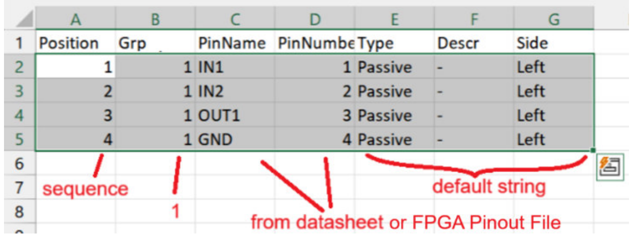
Altium also has several side **panels** for browsing and editing objects. They appear depending on file type, and whether we enabled them from the „Panels" button in the lower right corner. We can dock them to the left or right side of the screen, they can be set to stacked, static, auto hide. When they hide a tab label is shown. The projects panel shows the tree structure of our project, we can open files from there. Keep these panels always visible: Projects, PCB, Properties, List.

When **right clicking** in the editor on empty space, a big menu opens with lots of settings and actions, dependent on what file we are editing, what mode we are in.

Most companies have managed and released **shared libraries**, but one-person companies can keep library files edited within projects. We have to add existing company libraries to be accessible by the project. Either Project>Add Existing, or on the Components Panel> ≡ > Libraries Preferences> Installed> Install> select file type and browse.

2.2. SCH Library

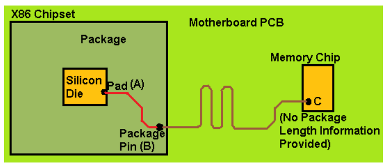
To make our own schematic symbols, we have to create/open a **SCHLIB file** on the projects panel by right clicking the PRJPCB and then >add new to project >Schematic Lib. The schlib panel opens, click Add, then enter name, select the new component it in the schlib panel. In heterogenous split components the top level is called component, the sub symbol is called a "part". They appear under the component name on the schlib panel. To add pins to the single part component or to the first part of a heterogenous component: select component on schlib panel, then Tools > Symbol Wizard, set number of pins, then fill out the table or import from Excel. We can prepare pin tables in Excel in a matching format like below, then select data (not header), then CTRL+C to copy, then in Altium upper/left cell CTRL+V to paste. For single-part-comp or first part we press Place> Symbol, for heterogenous sub-component-parts press Place> New Part. We can manually add more parts to a component by the button.

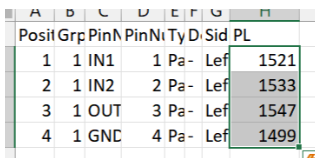


	A	B	C	D	E	F	G
1	Position	Grp	PinName	PinNumbe	Type	Descr	Side
2	1	1	IN1	1	Passive	-	Left
3	2	1	IN2	2	Passive	-	Left
4	3	1	OUT1	3	Passive	-	Left
5	4	1	GND	4	Passive	-	Left
6							
7	sequence					default string	
8		1	from datasheet or FPGA Pinout File				

On the properties panel we have to enter **parameters** like part number and value. We have to add company-specific properties, for example if we work at ACME-inc., then ACME_PN, ACME_DNP, MFR_PN… We also need to add a footprint using add> footprint> browse.

The signal length (**Package Length PL or PinDelay** A to B on diagram) within large BGA packages have to be entered for accurate length tuning later. When we are editing the schematic symbol, we have to do this separately for each sub-symbol (part). Prepare the PL data in excel (same unit as the tool, mils) for each subsymbol, sorted by pin number, select the length data only, CTRL+C to copy. In Altium select all pins in the subsymbol the drawing, but make sure the rectangle is not selected. Then open the SCHLIBlist panel, in the PinDesignator column sort, then in the Pin/PckLength column click the first item, CTRL+V to paste. Save.





A	B	C	D	E	F	G	H
Posit	Grp	PinN	PinNu	Ty	D	Sid	PL
1	1	IN1	1	Pa	-	Lef	1521
2	1	IN2	2	Pa	-	Lef	1533
3	1	OUT	3	Pa	-	Lef	1547
4	1	GND	4	Pa	-	Lef	1499

2.3. PCB Library

To make our own footprint symbols, we have to create/open a **PCBLIB file** on the Projects Panel> PRJPCB> rightclick >add new to project >PCB lib. The pcblib panel opens. To add a new footprint: Tools> IPC Compliant Footprint Wizard (or just the basic Footprint Wizard), fill in the parameters. Once done, we can edit it, by moving/deleting pins, Place>Pad, editing pads (click, then Properties panel edit, padstack, X/Y-size), drawing on silkscreen.

For large BGAs with **irregular** pattern, we should import from Allegro or Excel. In allegro open the reference design, export libraries, then open the footprint, reports> symbol pin report, then copy (pin number and X/Y coordinates) it into excel. We can also create this spreadsheet manually based on the datasheet in Excel. Organize the excel file to have the same columns as the Altium PCBLIBlist panel table has. In Altium, footprint library, create a package with the same number of pins using the wizard. Copy cells from Excel to Altium's PCBLIBlist XY coordinates columns. Make sure they were sorted the same way on both ends of the copy.

2.4. Schematics

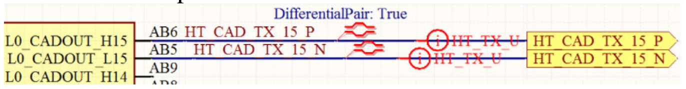
Every page is a separate SCHDOC file. Once we added enough SCHDOC files, in each we click Properties Panel> Sheet Size drop down to set a larger page size.

Multi-page schematics can be hierarchical with PORT (Place>Port) connections in the module and SHEET SYMBOL in the top level schematic. The sub-module symbol is created by: design>create sheet symbol from sheet. We can also design flat schematics with no top-level, then we have to use Place> Off Sheet.

We can place **components** from the „Components" panel, but first the SCHLIB file is selected on the top drop-down, then the component, doubleclick and move to desired location on schematic. Part rotation with SPACE key while moving it. Doubleclick on a part and we can see/edit properties, but don't edit here, rather in the released library. For example, have a separate library item for a 1k resistor than a 10k resistor.

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 DNP property in all components and NO means in place and Yes means DNP, or overwrite the part number with "DNP", then delete these rows from the BOM manually, or we can use variants. We could have a base design (add text next near all planned DNPs as "DNP" in red), and a production variant associated with the board part number. In this variant we mark the DNPs. First we annotate refdes, then we create variants in Project>Variants>Add Variant, Close. Then in the projects panel select the variant to edit by double clicking. In the schematic on a component (sub-part-A if split part) rightclick> Part Actions> Variants> In the column named after our variant click the […] button, select "NotFitted". All other edits should be done on the default "NoVariations" variant double-clicked in the Projects panel.

Once components are placed, we **wire** them with . We can place power symbols with . For non-GND nets we have to place this first, then select it, then on the Properties panel select Style=Circle, add a net name like P1V0_FPGAC, then we need to rotate it upside-down by moving and pressing TAB twice. We add net labels with Place> Net Label. We can draw buses with "signal harnesses". We can place parameter set directives with , then Add>NetClass or Rule… In older versions of Altium we had to place diffpair symbols in the schematic (Place> Directives> Diffpair), but in AD24 we should define diffpairs in the PCB CM or PCB-panel. Either way 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. If we use off-sheet or port then we don't need net labels.



Once done, we **annotate** refdes (designator in Altium) to all components with Tools > Annotation> Force Annotate All. Then export a BOM with Reports> Bill Of Materials, and if we used variants or variant-based DNPs then we select the correct variant from the drop-down. Error checking with project>compile document. We also export a PDF schematic with File> SmartPDF.

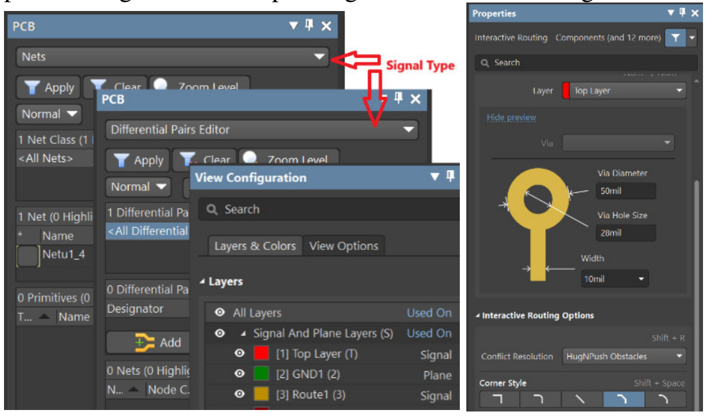
2.5. PCB Design

Start: Project> Add New> PCB, save. The old versions had a board wizard, that is now gone. Then 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 set up scale and layer (mechanical-1). Set up View> Grids> Set Up Global Snap Grid to be coarse/accurate, View> Board Planning Mode. Then Design>Redefine Board Shape, then redraw it, then View> 2D Layout Mode, then set the grids again to 10mils for placement. Set the origin Edit>Origin>Set on the board near the lower left corner.

Once the board is set up: we have to **import netlist** from schematic: design> import changes from [...prjpcb] validate, execute, close. This will place all parts next to the board.

Stackup Layers: Design> Layer Stack Manager. Add layers, define their thicknesses, signal/plane type order. Also define via types like BB, microvia depths. Layers are not categorized as class/subclass like in Allegro. We use the mechanical layers for all manufacturing comments and fab notes. To enable backdrilling, click the button, select backdrills, a lower tab appears, click it, then add as many BD layer pairs as we want to use with the [+Add] button, then editing the layers on the Properties panel. BD always starts from (Top or) Bottom, and ends one layer away from our routing MNC layer. On the via types tab, we define normal via sizes, and layer pairs for microvias or BB.

Panels: Enable/disable panels from the lower/right Panels button. During PCB design we use many panels, the most important one is the PCB panel, with a drop-down list on the top to select signal object type. In the PCB panel we can specify from a drop-down list what should happen when we select an object: either no effect („Normal"), or highlight („Dim"), or highlight and disable editing of other objects („Mask"). It is cancelled by pressing the „Clear" button in the bottom-right corner. The View Configuration panel is needed to show/hide layers. The Properties panel changes content depending on what we are doing.



Connection Lines: View> Connections> Show/Hide…

Layer colors and visibility can be changed: First enable the Panels>View Configuration panel at the lower/left panels button. Then we can change colors and temporarily enable/disable specific layer visibility as needed, by single clicking on the eye icon. In Altium everything that is on a layer (pads, vias, traces) are visible or invisible in the same time. The active layer being edited is selected on the bottom/lower labs under the PCB:

Object Coloring: Nets can be displayed either in layer color, or net color. On the PCB panel find and select the nets, then rightclick>Change Net Color. Enable Color Override for each net using the checkbox next to its name. Set solid coloring in Tools> Preferences> PCB editor> Board Insight Color Overrides.

The new Altium supports **object filtering**, like Allegro. Before moving or deleting objects, we can disable others on the Properties panel> Selection Filter, to avoid affecting them.

Measure distance: Reports> Measure Distance

Find items and zoom to them: Press "J", then select component or net, then type in the refdes or net name. To display info about an object: left click, the Properties panel will display relevant data.

Component placement is, really just moving from the auto placed (when imported changes from SCH) area to the board design. Before placement we make inner layers hidden, so we can select components instead of planes. Click on one comp, hold and move. If it doesn't work then select the small area enveloping the component then click/move. Rotate with SPACE, or move to bottom side with "L". Move refdes as if it was a component. We can also select a few comps on the sch, if cross select mode is enabled in tools, then in the layout move them. If the refdeses are too large, then we can select one, rightlcik> Find Similar Objects> Refdes = same, OK, then on the Properties panel we change the text height. How close we can pack them is set in a constraint: CM> AllR> Placem> CompClearance.

We can **place footprints** for floorplanning from the components panel. We can also place mounting holes and fiducials from it, although using schematic is preferred. MNT holes can also be placed as pads (Place> Pad then assign GND net).

Design rules: When the PRJPCB is created we have to decide whether we will be using the legacy design rules editor (DRE), or the constraints manager (CM). DRE opens from Design>Rules. CM opens from Design > Constraint Manager, and stored in constraints.xml file. We should set up at least some basic rules like clearance, width, solder mask, via style, plane connect/clr, diffpair rules before any fanout or routing. See chapters on CM/DRE.

Interactive routing starts with the icon. Set the grids to about 1mil for routing (View>Grids). Active layer is selected by clicking the bottom tabs. Routing mode can be controlled from: tools> Preferences> PCB> Interactive Routing in advance, or from the properties panel during editing, for example push/shove. We can just click on traces to slide/edit them; we don't have to select a specific mode first like in allegro.

Add **Vias**: press "+" click while routing. Ground stitching vias can to be placed from Place>via, then assign the net name on the Properties panel. The size is set in CM> Allr> Routing> Routing Via Style (All, not net class) or DRE> Route> RoutViaSt. Altium likes to create vias with soldermask opening, but modern complex boards are made with all VIPPO (state in fab drawing notes) and complete tenting (no opening) on both sides. Cheap boards without VIPPO need complete tenting on top (especially under BGAs) and a small opening on bottom (for outgassing). So, we can select a via, rightclick> find similar > ok, then on the Properties Panel > Solder Mask Expansion = manual = tented. Or set a rule before use: CM> Allr> Mask> SolderMExp create a new rule, Object Match = "IsVia", then click checkbox for tenting. To use microvias, we have to create them in the Layer Stack Manager, based on the layer-pairs form the vendor stackup.

Fanout: Set the constraints like width, clearance, SM expansion and routing via style, as shown above. Then set up fanout behavior with CM> Allr> Routing> Fanout Control. Then select Route> Fanout> Component, click component.

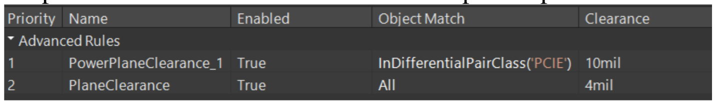
Draw Power **Shapes**: (polygon pour) on signal layers, for power delivery or VRM circuit power nets. Doubleclick to add net. Voids can be drawn with rightclick on the button and select Polygon Pour Cutout.

Keepout shapes: place> keepout> fill, then on the Properties panel select object type to keep out (via, pad, trace…). For example, via keepout on a routing layer.

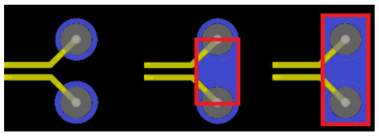
We can **specify areas** where different rules would apply locally. These areas are called „Rooms" and can be placed from the Design > Rooms > Place Rectangular Room menu. Then on the Properties panel they can be named and specify which layer to be used on. Then they can be referenced in the design rules or CM (based on room name), so the rule will only apply within the area covered by the Room. In the CM>Physical at the bottom of the table AllRooms/name we can enter new width/via constraints. This is good for example for BGA breakout neck down, or smaller via pads to implement "CLASS-3 with exception". It doesn't work for plane clearance, so use class-based clearance rules.

Power **plane layers**: Altium supports negative planes, so they need to be set in the stackup. On negative plane layers we place divider lines, enable the visibility of the layer on the ViewConf panel, then select the layer on the bottom tabs, then (Place >Line), then doubleclick an area and select net. There is also a thick outline layer to pull back from edge cuts.

Plane voids: All signal vias passing through planes will have an antipad. Altium automatically removes all non-functional pads (NFPs) on plane layers, but leaves them on signal layers. That also shrinks the antipads on the plane layers to AP= drill+2*CL clearance rule value, while keeping them larger AP= pad+2*CL on signal layer power shapes. Most signals are fine with tight APs around their via barrels, based only on etching clearance, but SERDES signals need enlarged pads for via-impedance control and backdrilling. BD requires a large BD-antipad AP= BD + 2*drillclearance. The drill clearance is usually 5…8mil on each side, depending on our fab. The BD tool size is usually 6…12 mil larger in diameter than the via drill size, also depending on the fab. So, we need to create two rules under CM> AllR> Plane> Plane Cl, one for all signals with 5mil clearance, and one for the net or DP classes that belong to 8Gig+ SERDES difffpairs to force a large AP. Clearance= (AP-viadrill)/2. Signal layers also need a route keepout circle the same size as the BD antipad on planes.



Priority	Name	Enabled	Object Match	Clearance
Advanced Rules				
1	PowerPlaneClearance_1	True	InDifferentialPairClass('PCIE')	10mil
2	PlaneClearance	True	All	4mil



Dual-**voids** are required for high-speed diffpairs on plane layers. Select GND layer on bottom tabs, then Place> Arc(center) on the pins then Place> Fill between. Either a small rectangle and 2 arcs, or a large rectangle to envelop them. If the clearance rule-driven void is large enough then we don't need extra arcs. We copy these on all planes. We can also copy this to the route keepouts, to help backdrilling and prevent signals passing between p/n vias.

DRC check: Tools> Design Rule Check. 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.

Thieving cannot be applied in Altium, but we could use polygon pours with crosshatched pattern or rely on our PCB fab.

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 meant for fab drawing, we place tables with Place> Drill Table and Place> Layer stack table, then we manually add text about technology statements, materials, surface finish, coupons, impedance and loss tables. We also create an assembly drawing on another mechanical layer, for verifying the build, by using a dimensioned DXF from our ME. Once all done we fab-out: File> Fabrication Outputs> Gerber Files, and also NC Drill. We send a 3D model to our ME, to verify system fit: File> Export> Step3D. We generate additional documents: File> Assembly> Pick & Place and Assembly drawing, and File> Smart PDF. Or all this in a more complicated way: Projects Panel> PRJPCB> rightclick> Add New> Output Job file.

We can export **data reports** from the PCB panel. Select the desired object type (xSignals, Components), then select all classes and all objects with CTRL+A, then rightclick>Report>Export.

2.6. Design Reuse

Design-reuse with a project is done using multi-channel hierarchical designs, with multiple instances of the same sch sheet symbol. Each will create a "room" in the PCB, then we manually place&route one channel, then copy the work over to the other channels: Design> Rooms> Copy Room Formats. Reusing designs from other projects can also be done: File> New> Reuse Block, it will have both a schematic (single page) and a PCB file, we edit both and save, then on the Project panel "save to server". In our new project schematics Place> Reuse Block, the Design Reuse panel opens, we find out block, place button> rightclick> Place As Sheet Symbol.

3. High-Speed Signal Objects

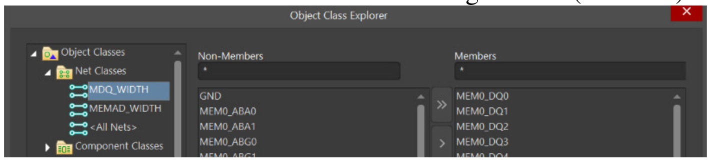
The connection objects define the signals, or groups of signals. They can be browsed on the PCB panel. Categories:

- ❖**Net**: created using net labels in schema.
- ❖**Bus**: create using bus symbols in schema, it uses indexed net names, like ABC1_[7..0] → ABC1_0. For serdes-links, it is better not to use buses, unless hierarchical.
- ❖**Diffpair** (DP): Create them in the CM Physical tab with rightclick > diffpair > Create Diffpairs From Nets, or in the PCB panel's "Differential Pair Rule Wizard" both objects and rules get created. Or graphically in the sch.
- ❖**Net Class** (NetC): A group object. In PCB using Design> Classes window (add new). Used for all same type (non-DP) signals on a bus, for length rules. Also used on trace width (impedance) rules for single-ended.
- ❖**Diffpair Class** (DPC): Similar to net class, create in the PCB Design>Classes editor. We need two rules for our DP class: a width and a diffpair rule (phase tol, gap, uncoupled). We need one for lane-to-lane matching.
- ❖**From-To**: Same as the Allegro PinPair, but obsolete.
- ❖**xSignal** (XS): Same as the Allegro PinPair, preferred. This is to control propagation delay from a pin to another pin. On a DDR4 fly-by address bus, on every address net we would have an xSignal from CPU-DRAM0, another

from CPU-DRAM1... They should be created with Design> xSignals> Create xSignals, or with a wizard. Xsignals also connect nets through res/cap.

- ❖**xSignal Class** (XSC): Similar to a matched group of PinPairs in Allegro. We create one XSC for all the xSignals of all DDR4 address nets between CPU-DRAM1. Then create a length matching constraint/rule for that XSC. Another XSC and a rule for CPU-DRAM2.
- ❖**XNET**: They can be created in the CM, XNETs are for two nets passing a series passive part.

Classes are used to group signals before adding them to a rule. We can have net classes, DP classes, xSignal classes. Classes are created in the „Object Class Explorer" at Design>Classes, by selecting a category, rightclick> Add Class, then add items to it. Diffpairs should be put into DP classes only, not net classes, to avoid double definition of width when using the CM (not DRE).


Object Class Explorer

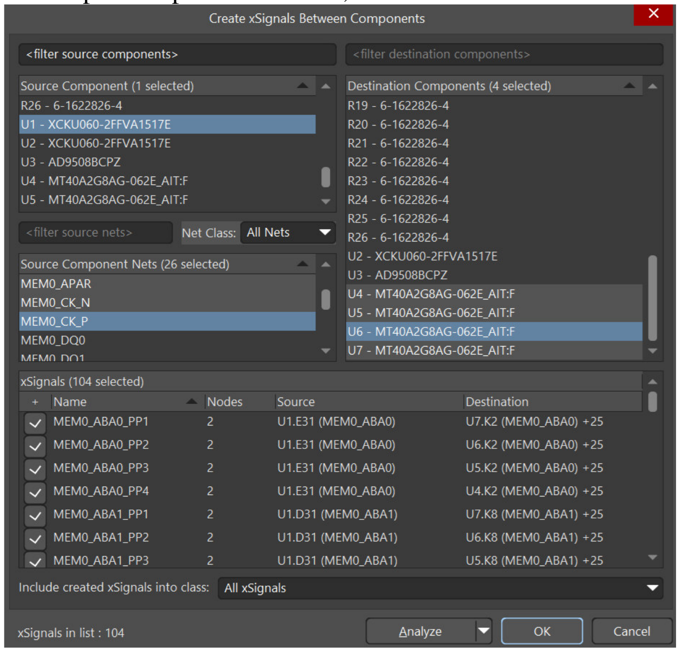
Object Classes
Net Classes
MDQ_WIDTH
MEMAD_WIDTH
<All Nets>
Component Classes

Non-Members: *
GND
MEM0_ABA0
MEM0_ABA1
MEM0_ABG0

Members: *
MEM0_DQ0
MEM0_DQ1
MEM0_DQ2
MEM0_DQ3

Trace **length measurement** still has bugs in AD25. It might fail to measure xSignal lengths accurately, which case we have to delete and reroute it. Either little trace segments overlap with others, or a segment is not counted at all.

Xsignals and xSignal classes, as well as their associated design rules could be created through the xSignal wizard: Design> xSignals> Run Xsignals Wizard. The other way is through Design> xSignals> Create xSignals (select source, load, nets, then analyze, OK, redo for the next lane or chip), which better shows exact parts. The XS names will end with "_PP1…N", where N is the Nth chip on the bus. A third way is in the CM>Electrical>Nets tab, click on any nets, then the Topology columns, then in the drop-down select "custom". In the Design> Classes> xSignal Classes we have to create XSCs so we can apply rules to them later in the CM. Use the search with "*" to get for example "MEM0_A*PP1" for address at DRAM1. We create XS for point to point buses also, so we can match them in XSCs.


Create xSignals Between Components

<filter source components>
<filter destination components>

Source Component (1 selected)
R26 - 6-1622826-4
U1 - XCKU060-2FFVA1517E
U2 - XCKU060-2FFVA1517E
U3 - AD9508BCPZ
U4 - MT40A2G8AG-062E_AIT:F
U5 - MT40A2G8AG-062E_AIT:F

Destination Components (4 selected)
R19 - 6-1622826-4
R20 - 6-1622826-4
R21 - 6-1622826-4
R22 - 6-1622826-4
R23 - 6-1622826-4
R24 - 6-1622826-4
R25 - 6-1622826-4
R26 - 6-1622826-4
U2 - XCKU060-2FFVA1517E
U3 - AD9508BCPZ
U4 - MT40A2G8AG-062E_AIT:F
U5 - MT40A2G8AG-062E_AIT:F
U6 - MT40A2G8AG-062E_AIT:F
U7 - MT40A2G8AG-062E_AIT:F

<filter source nets> Net Class: All Nets

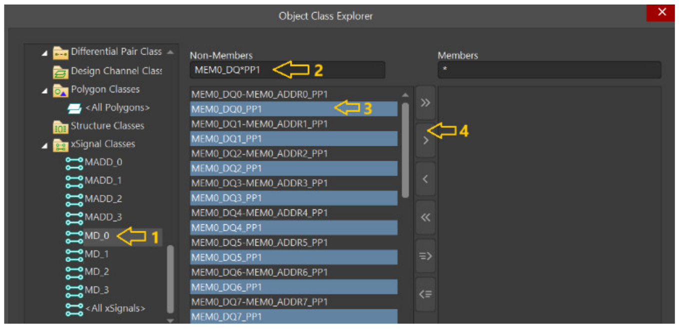
Source Component Nets (26 selected)
MEM0_APAR
MEM0_CK_N
MEM0_CK_P
MEM0_DQ0
MEM0_DQ1

xSignals (104 selected)

	Name	Nodes	Source	Destination
✓	MEM0_ABA0_PP1	2	U1.E31 (MEM0_ABA0)	U7.K2 (MEM0_ABA0) +25
✓	MEM0_ABA0_PP2	2	U1.E31 (MEM0_ABA0)	U6.K2 (MEM0_ABA0) +25
✓	MEM0_ABA0_PP3	2	U1.E31 (MEM0_ABA0)	U5.K2 (MEM0_ABA0) +25
✓	MEM0_ABA0_PP4	2	U1.E31 (MEM0_ABA0)	U4.K2 (MEM0_ABA0) +25
✓	MEM0_ABA1_PP1	2	U1.D31 (MEM0_ABA1)	U7.K8 (MEM0_ABA1) +25
✓	MEM0_ABA1_PP2	2	U1.D31 (MEM0_ABA1)	U6.K8 (MEM0_ABA1) +25
✓	MEM0_ABA1_PP3	2	U1.D31 (MEM0_ABA1)	U5.K8 (MEM0_ABA1) +25

Include created xSignals into class: All xSignals

xSignals in list : 104


Object Class Explorer

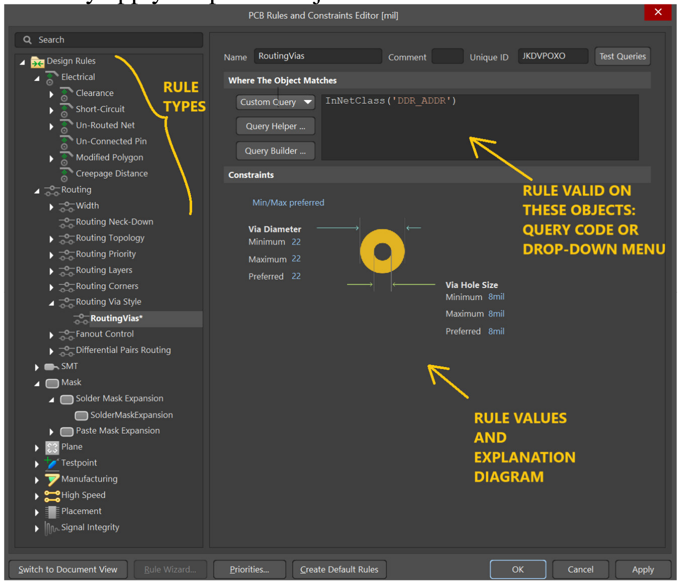
Differential Pair Class
Design Channel Class
Polygon Classes
<All Polygons>
Structure Classes
xSignal Classes
MADD_0
MADD_1
MADD_2
MADD_3
MD_0
MD_1
MD_2
MD_3
<All xSignals>

Non-Members: MEM0_DQ*PP1
MEM0_DQ0-MEM0_ADDR0_PP1
MEM0_DQ0_PP1
MEM0_DQ1-MEM0_ADDR1_PP1
MEM0_DQ1_PP1
MEM0_DQ2-MEM0_ADDR2_PP1
MEM0_DQ2_PP1
MEM0_DQ3-MEM0_ADDR3_PP1
MEM0_DQ3_PP1
MEM0_DQ4-MEM0_ADDR4_PP1
MEM0_DQ4_PP1
MEM0_DQ5-MEM0_ADDR5_PP1
MEM0_DQ5_PP1
MEM0_DQ6-MEM0_ADDR6_PP1
MEM0_DQ6_PP1
MEM0_DQ7-MEM0_ADDR7_PP1
MEM0_DQ7_PP1

Members: *

4. Design Rules

In Altium Designer in the traditional flow we normally specify PCB design rules in the **Design Rules editor** (DRE), in the Design> Rules... menu item, instead of using a constraint manager. For a PCB design, we have to set up various general design rules, PCB fabricator-driven rules, as well as high-speed design related trace length and impedance-driven width rules. In every category there is a default rule, and several user created ones that only apply to specific objects.


PCB Rules and Constraints Editor [mil]

Search

Design Rules
Electrical
Clearance
Short-Circuit
Un-Routed Net
Un-Connected Pin
Modified Polygon
Creepage Distance
Routing
Width
Routing Neck-Down
Routing Topology
Routing Priority
Routing Layers
Routing Corners
Routing Via Style
RoutingVias*
Fanout Control
Differential Pairs Routing
SMT
Mask
Solder Mask Expansion
SolderMaskExpansion
Paste Mask Expansion
Plane
Testpoint
Manufacturing
High Speed
Placement
Signal Integrity

Name: RoutingVias Comment: Unique ID: JKDVPOXO

Where The Object Matches
Custom Query
InNetClass('DDR_ADDR')

Constraints
Min/Max preferred

Via Diameter
Minimum 22
Maximum 22
Preferred 22

Via Hole Size
Minimum 8mil
Maximum 8mil
Preferred 8mil

When the **PRJPCB is created** we have to decide whether we will be using the legacy design rules in the design rules editor (DRE), or the constraints manager (CM).

The DRE has a **tree structure**. Each item in the left menu is a rule category. We can create several new rules for each category, each rule will be applied to one class (or complex formula), by rightclick> New Rule. Important categories:

- ❖Elecrical> clearance> clearance: mfr spacing.
- ❖Routing> width>width: trace width.
- ❖Routing> rout.via style: Set diameter, drill size, tenting.
- ❖Routing> Differential: diffp gap, phase-tol and uncoupled.
- ❖Mask: solder mask and paste mask expansion param. Create a rule for vias with scope="IsVia", and set tenting on top and bottom. The default rule for all other pads should be 2mil expansion and no tenting.
- ❖Plane: power plane/shape param, and via conn style.
- ❖High speed> Matched net lengths: Matching rules are for setup/hold timing on source-synchronous and clock-forwarding buses.

❖High speed> Length: max total trace length rules are for synchronous or asynchronous bus SU/H timing or for SERDES link loss budget (dB/inch).

There is always a **default** rule, and specific rules for defined objects (nets, classes). The default rules are based on fabricator minimum sizes (capability), the specific ones on calculations.

The typical **DRE flow flow**: 1. On PCB panel create objects like diffpairs, 2. In design>Classes create classes, 3. In Design Rules create rules that apply to classes.

Every board needs **basic rules** like via direct plane connect and plane clearance under DRE>Plane, component-to-comp clearance under DRE> Placement, default trace width under DRE> Routing> Width and Via Style (size), and DRE> Electrical> Clearance for copper trace gaps enforced by DRC and interactive push routing. Mask> SolderMask typically 2-3mils depending on fab, and Routing> Fanout Control for via patterns.

We can specify PCB design **rules in the schematics** level as well. We add one PCB_Layout directive (which can contain multiple rules) to a net. Instead of specifying the object to apply in the rule, the object is specified graphically by attaching the rule symbol to net or to multiple nets (copies of the same directive). We can place a rule directive from the Place> Directives> Parameter Set, then doubleclick on the symbol and doubleclick on the rule in the list, then click on the „Edit rule Values" button.

Trace width: Normally we put a group of nets based on characteristic impedance into a Net Class (for single-ended) or DP class. Then we set up a DRE> Routing> Width rule for every class separately, and also a default width rule for all other or non-impedance controlled traces (4mil). There are 3 values: for the default rule set min=pref<<max, for others min=pref=max. We should use width rules instead of impedance rules, from our fab vendor's impedance/width/space calculations (from the negotiated approved stackup document).

Spacing in PCB design has two aspects: manufacturability and controlling crosstalk levels. For the first aspect, we set up a DRE> Electrical> Clearance rule, which is normally the minimum spacing that our PCB manufacturer recommends. The Differential Pair rules also contain a field called „Gap", but this is a related to the differential impedance of the diffpair. For crosstalk control we use a more flexible spacing rule: DRE> High-Speed> Parallel Segment rule in the Design Rule Editor. This rule ignores short parallel segments, and only checks spacing if the two traces run too long in parallel. Basic spacing is enforced by a push/shove force in interactive routing. Sometimes we want to specify spacing (Parallel Segmenth rule) between differential pairs within a group, so we specify the rule's scope like „InDifferentialPairClass('classname')" for the first object and „IsDifferentialPair" for the second object.

For **diffpairs** the DRE> Routing> Differential Pair Routing rule has to be set up for every Differential-Pair Class, or to a list of classes. The rule specifies the trace spacing (Gap) between the positive and negative trace within a diffpair, and it is enforced in interactive editing when we use the Interactive Differential Pair routing option. Diffpair impedance is based on width+gap, we set the gap in the diffpair rule and the width in the width rule. We will also have to set up two „Matched Net Lengths" rules applied to a DP class, one to meet the phase tolerance requirements (within DP=On, others=Off), and another one for lane-to-lane matching (within DP=Off, others=On). If we have two multi-lane PCIe links, then each will be in a different DP class.

Matched length: With this rule we can control the trace lengths relative to each other in a class of signals, diffpairs or xSignals. It is used on source synchronous buses like DDR4 data bus, for diffpair phase tolerance matching or for diffpair lane-to-lane matching. We can find this at DRE> High Speed > Matched Net Lengths. An important parameter of the rule is the types of objects which between the rule will be applied. For differential phase tolerance rules, we enable only the „Check Nets Within Differential Pair" option, and leave the other two options disabled. For lane-to-lane matching or for single-ended buses we enable the opposite. Note that the „Tolerance" value in the rule is the maximum difference between the longest and shortest track.

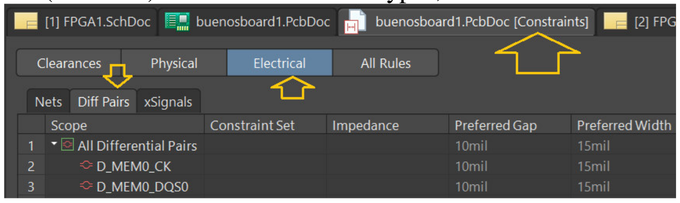
Min/Max trace length: With this rule, we can specify a length range for a class of signals (Net/DP/XS). We can find the DRE> High Speed > Length rule in the Design Rules Editor. It is useful for synchronous, asynchronous and loss budget driven buses.

5. Constraint Manager

Starting from 2024 Altium supports a spreadsheet-like constraints manager window, like Cadence Allegro and Mentor Expedition has. Open: Design > Constraint Manager. This is not available in the cheapest license option. This is preferred instead of using the DRE if we have hundreds of xSignals or diffpairs, on large server or router board designs. When a project starts we have to decide whether it will be a legacy design rules-based design or a CM based design, we cannot have both. Although we can still migrate an existing design from DRE to CM: Design > Migrate Project to Constraint Manager Flow. Migration only works if the system feature is enabled in: Tools> Preferences> System> General> Advanced> Constraint Manager. Project Migration Wizard = 1, then restart Altium.

The contents of CM are saved into an **XML file**, but only when we close Altium. There is a bug in AD25, so we cannot enter values to xSignal classes like tolerance or Target, but there is a solution: Once we set up all xSignals and XSCs, save, close Altium, then open Altium again, go to CM and enter these values, save, close Altium (to ensure it doesn't crash and lose all constraints, and re-open again). Now the constraints are usable in editing and DRC.

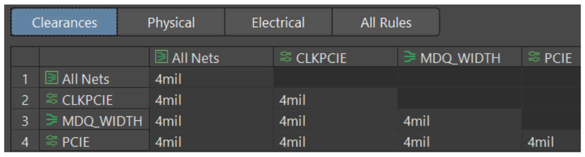
It has **3 tabs**: Clearances, Physical (width, diffpairs), Electrical (lengths). Under electrical, there are 3 sub tabs: Nets, Diffpairs, and xSignals. In any of these we can also create classes, by selecting multiple signals, rightclick>classes>… or create new objects like DP, XS, XSC… The Clearances tab allows us to add new rules, the Phys/Elec tabs list objects that we can enter numbers for. The last tab (AllRules) shows a tree of rule types, similar to the old DRE.



	Scope	Constraint Set	Impedance	Preferred Gap	Preferred Width
1	All Differential Pairs			10mil	15mil
2	D_MEM0_CK			10mil	15mil
3	D_MEM0_DQS0			10mil	15mil

We can use **Constraint sets**: Once we set up parameters for one object, we can rightclick > Save As Constraint Set, then we can reapply it to other objects: rightclick> Select Constraint Set.

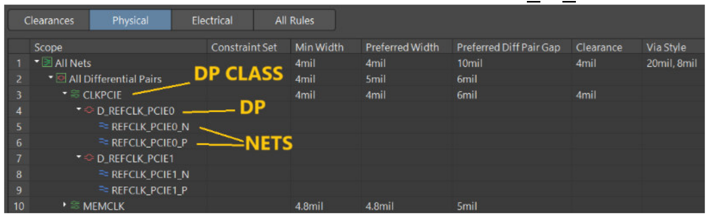
In the **Clearances tab** we can define a clearance between Net/DP classes in a matrix. We have to add existing classes to the matrix. When we click [+Add] then it creates both a column+row.



Clearances | Physical | Electrical | All Rules

		All Nets	CLKPCIE	MDQ_WIDTH	PCIE
1	All Nets	4mil			
2	CLKPCIE	4mil	4mil		
3	MDQ_WIDTH	4mil	4mil	4mil	
4	PCIE	4mil	4mil	4mil	4mil

In the **Physical tab** we can see most signal objects types in a tree hierarchy browser view, like Class/DP/nets, and specify trace width and via style/size to them. The PolygonConnect column should be set "DirectConnect" for "IsVia". We can auto create all diffpairs here: rightclick > diffpair > Create Diffpair From Nets. The editor will find them all from name suffixes _P/_N.

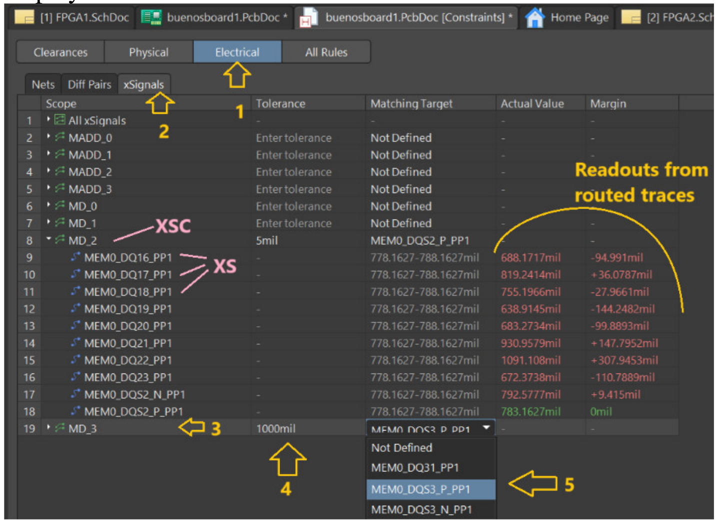


Clearances | Physical | Electrical | All Rules

	Scope	Constraint Set	Min Width	Preferred Width	Preferred Diff Pair Gap	Clearance	Via Style
1	All Nets		4mil	4mil	10mil	4mil	20mil, 8mil
2	All Differential Pairs		4mil	5mil	6mil		
3	CLKPCIE		4mil	4mil	6mil	4mil	
4	D_REFCLK_PCIE0						
5	REFCLK_PCIE0_N						
6	REFCLK_PCIE0_P						
7	D_REFCLK_PCIE1						
8	REFCLK_PCIE1_N						
9	REFCLK_PCIE1_P						
10	MEMCLK		4.8mil	4.8mil	5mil		

In the **Electrical tab** we can view/edit trace length related rules. We enter values into an existing table, instead of creating rules. The Electrical tab has three different sub-tabs: Nets, DP and xSignals. Each shows object types in a hierarchy/tree browser. On the Nets sub-tab, we can create XNETs, and we can enter numbers for max total length (for synchronous buses and loss-driven SERDES links) and max via stub length constraints (for 8G+ SERDES links). On the Diffpairs sub-tab we can define diffpairs, and we can enter numbers for diffpair width, gap, uncoupled length and phase tolerance of the diffpairs.

The **xSignals sub-tab** at CM> Electr> xSignals tab doesn't allow object creation, so we have to do that in the Design> xSignals> Create xSignals menu. When we first create XS/XSC objects, the CM> Electr> xSignals tab seems to be read-only, so we have to save, close Altium, reload Altium, and open CM, now we can enter values. XSC-based rules in Altium were made for trace length matching rules, so we have to enter tolerance in +/- 5mils or something, and click Target and open a drop-down menu to select which member of the XSC will serve as a target length for all other members. If we need max length rules on xSignals, like on a synchronous multi-master PCI bus, then we have to use the AllRules tab instead of the Electr> xSignals tab. The CM will display the length measurements for each object as "Actual Value", or the matching rule deviation as "Margin". The PCB panel also displays these measurements.



[1] FPGA1.SchDoc | buenosboard1.PcbDoc * | buenosboard1.PcbDoc [Constraints] * | Home Page | [2] FPGA2.Sch

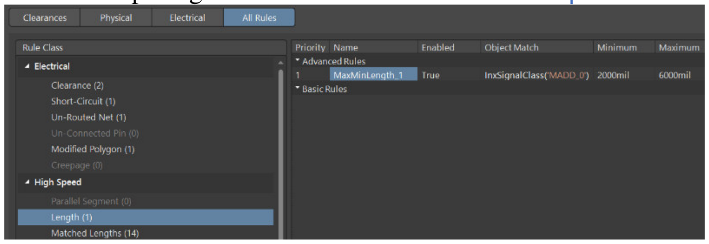
Clearances | Physical | Electrical | All Rules

Nets | Diff Pairs | xSignals

	Scope	Tolerance	Matching Target	Actual Value	Margin
1	All xSignals	-	-	-	-
2	MADD_0	Enter tolerance	Not Defined	-	-
3	MADD_1	Enter tolerance	Not Defined	-	-
4	MADD_2	Enter tolerance	Not Defined	-	-
5	MADD_3	Enter tolerance	Not Defined	-	-
6	MD_0	Enter tolerance	Not Defined	-	-
7	MD_1	Enter tolerance	Not Defined	-	-
8	MD_2	5mil	MEM0_DQS2_P_PP1	-	-
9	MEM0_DQ16_PP1	-	778.1627-788.1627mil	688.1717mil	-94.991mil
10	MEM0_DQ17_PP1	-	778.1627-788.1627mil	819.2414mil	+36.0787mil
11	MEM0_DQ18_PP1	-	778.1627-788.1627mil	755.1966mil	-27.9661mil
12	MEM0_DQ19_PP1	-	778.1627-788.1627mil	638.9145mil	-144.2482mil
13	MEM0_DQ20_PP1	-	778.1627-788.1627mil	683.2734mil	-99.8893mil
14	MEM0_DQ21_PP1	-	778.1627-788.1627mil	930.9579mil	+147.7952mil
15	MEM0_DQ22_PP1	-	778.1627-788.1627mil	1091.108mil	+307.9453mil
16	MEM0_DQ23_PP1	-	778.1627-788.1627mil	672.3738mil	-110.7889mil
17	MEM0_DQS2_N_PP1	-	778.1627-788.1627mil	792.5777mil	+9.415mil
18	MEM0_DQS2_P_PP1	-	778.1627-788.1627mil	783.1627mil	0mil
19	MD_3	1000mil	MEM0_DQS3_P_PP1	-	-

Not Defined
MEM0_DQ31_PP1
MEM0_DQS3_P_PP1
MEM0_DQS3_N_PP1

The **AllRules tab**: This shows a tree of many rule types, similar to the old DRE, mainly for basic rules. Check the DRE section for detailed explanations! Here we can create new rules by rightclick> Add Custom Rule, then enter ObjectMatch ([…]> Open Query Builder for classes or object types) and numbers. In the CM>AllRules > Mask> SolderMask category create a rule for vias with scope="IsVia", and set tenting on top and bottom. The default rule for all other pads should be 2mil expansion and no tenting. CM>AllRules > Routing> Routing Via Style sets the via pad and drill diameters. CM> AllRules > Routing> Fanout Contr sets pattern (auto, out, away, centered). The plane via connection style (DirectConn), and plane layer clearance have to be set in CM> AllRules> Plane> Power Plane Connect Style, and clearance. The plane clearance should be set to achieve a desired antipad size AP=BD+12mil= drill+2*CL. The CM> AllRules> HighSpeed> Parallel Segment is good for crosstalk control spacing that is only checked if it runs long (past limit). If we need max length rules on xSignals, like on a synchronous multi-master PCI bus, se set up a rule under CM> AllRules> HighSpeed> Length. A multi-lane differential SERDES link requires lane-to-lane matching, with the CM> AllRules > HighSp> Matched Length. Backdrilling is defined by max stub length and oversize (each side of hole) in CM> AllRules> HighSpeed> BackDrilling. BD also requires definitions in the Layer Stack Manager. Placement spacing: CM> AllRules > Placem> CompClearance.



Clearances | Physical | Electrical | All Rules

Rule Class
- Electrical
  - Clearance (2)
  - Short-Circuit (1)
  - Un-Routed Net (1)
  - Un-Connected Pin (0)
  - Modified Polygon (1)
  - Creepage (0)
- High Speed
  - Parallel Segment (0)
  - Length (1)
  - Matched Lengths (14)

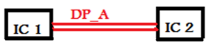
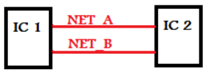
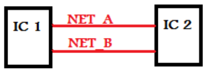
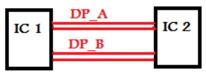
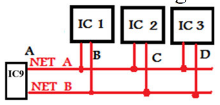
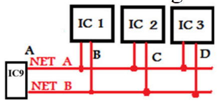
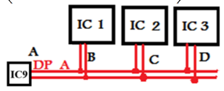
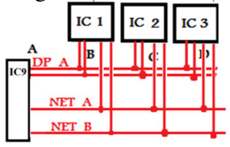
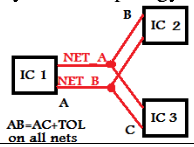
Priority	Name	Enabled	Object Match	Minimum	Maximum
Advanced Rules					
1	MaxMinLength_1	True	InxSignalClass('MADD_0')	2000mil	6000mil
Basic Rules					

For **diffpairs** in the DRE we specify 4 rules, a width, a diffpair rule, a matching rule for tolerance and a matching rule between lanes. In the CM we enter values for diffpairs on the CM> Electr> Diffp sub-tab, and on the CM> AllRules> HighSp> Matched Length sub-tab also for lane matching (rightclick> add custom rule, Object= InDifferentialPairClass('NAME'), then select "group- matched"). Altium25 seems to define width and gap rules in two places, so to avoid using the wrong values in routing, we should set the same at both: CM> Physical and at CM> Electr> Diffp. Most values are entered to the rows of the DPCs. If we have two multi-lane PCIe links, then each will be in a different DPC.

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<budget/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 timing analysis calculator spreadsheet, like this one: https://www.buenos.extra.hu/iromanyok/PCB_Timing_analysis.xls

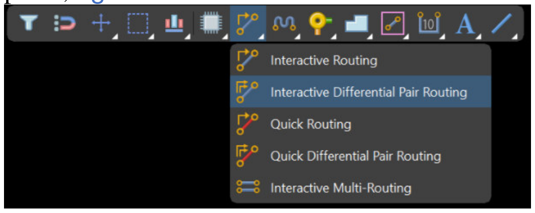
Typical high-speed objects and constraints:

Case	**Objects to create**	**Constraint numbers entered in CM**

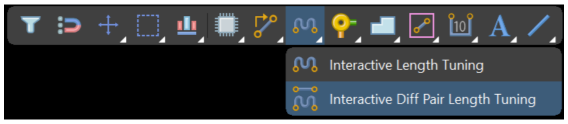
Differential Pair point-to-point signal 	•DP •DPC	• CM> Physical (DPC) • CM> Electr> Diffp (DPC) • CM> AllRules> HighSpeed> Parallel Segment. (DPC) • CM> AllRules> HighSpeed> BackDrilling (DPC, if >8Gbps)
Single-ended Sync/ Async point-to-point bus with min/max len. 	•NetC	• CM> Physical (NetC) • CM> AllRules> HighSpeed> Length (NetC) • CM> AllRules> HighSpeed> Parallel Segment. (NetC)
Single-en Source-sync point-to-point bus with matched lengths 	•NetC •XS •XSC	• CM> Physical (NetC) • CM> Electr>xSignals (XSC) • CM> AllRules> HighSpeed> Parallel Segment. (NetC)
Multi-lane Point-to-point diff SERDES bus 	•DP •DPC	• CM> Physical (DPC) • CM> Electr> Diffp (DPC) • CM> AllRules> HighSp> Matched Length (DPC) • CM> AllRules> HighSpeed> Parallel Segment. (DPC) • CM> AllRules> HighSpeed> BackDrilling (DPC, if >8Gbps)
Single-ended Sync/ Async multi-drop bus with min/max length.  IC 1 IC 2 IC 3 IC9 A NET_A NET_B B C D	•NetC •XS •XSC (one/chip)	• CM> Physical (NetC) • CM> AllRules> HighSpeed> Length (XSC) • CM> AllRules> HighSpeed> Parallel Segment. (NetC)
Single-ended Source-sync multi drop bus with matched lengths  IC 1 IC 2 IC 3 IC9 A NET_A NET_B B C D	•NetC •XS •XSC (one/chip)	• CM> Physical (NetC) • CM> Electr>xSignals (XSC) • CM> AllRules> HighSpeed> Parallel Segment. (NetC)
Diff multi drop bus (like DDR4 clock) 	•DP •DPC •XS •XSC (one/chip)	• CM> Physical (DPC) • CM> Electr> Diffp (DPC) • CM> Electr>xSignals (XSC) • CM> AllRules> HighSpeed> Parallel Segment. (DPC)
Mixed SE/Diff multi-drop bus with matched lengths, (DDR4 ACC)  IC 1 IC 2 IC 3 IC9 A DP_A B C D NET_A NET_B	•NetC (SE sign) •DP •DPC •XS •XSC (one/chip, SE+diff)	• CM> Physical (NetC for SE) • CM> Physical (DPC) • CM> Electr> Diffp (DPC) • CM> Electr>xSignals (XSC) • CM> AllRules> HighSpeed> Parallel Segment. (NetC+DPC)
Single-ended Source-sync tree topology  IC 1 IC 2 IC 3 NET_A NET_B A B C AB=AC+TOL on all nets	•NetC •XS •XSC (one/chip)	• CM> Physical (NetC) • CM> Electr>xSignals (XSC) • CM> AllRules> HighSpeed> Parallel Segment. (NetC)

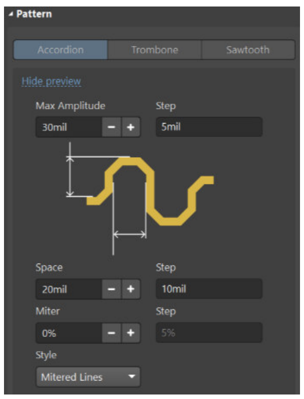
6. Interactive High-Sp Route

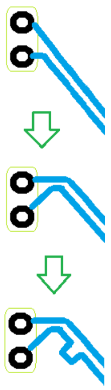
During interactive **routing**, the properties panel should be visible, that allows us to alter routing parameters. TAB pauses routing. We can manually route traces by right clicking the routing mode button, select interactive routing, then clicking a pad, then pulling the trace. For diffpairs we have to rightclick on the button, then select Interactive Differential Pair Routing option. First we route all traces with plenty of spacing, then we tune them later. Before routing starts, we should lock down large components by clicking on them, then Properties panel location clock the button. We can select objects in the schematic, then auto highlighted in the PCB by tools> Cross Select Mode=on. We might want to hide most connection lines (ratsnests), to see clearly: View> Connections> Hide All, then select a few nets on the PCB panel, rightclick> Show.

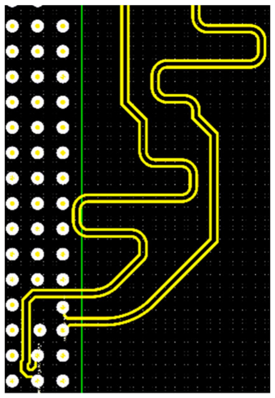

Interactive Routing
Interactive Differential Pair Routing
Quick Routing
Quick Differential Pair Routing
Interactive Multi-Routing

Length **tuning**: We can delete then re-route trace segments, auto-meander traces with the tuning button, or slide them (select, drag) to increase/decrease the signal length. The live tuning gauge only pops up during actual tuning or SE-routing, not diffpair-routing or sliding, so for those we have to rely on the PCB panel. While starting the tune, after pressing the tuning button, we can press TAB to open the Properties panel, where we can select the meander pattern and parameters. Any editing mode can be exited/ended with the ESC key. Single-ended traces are tuned by right clicking the tuning button, select Interactive Length Tuning, while diffpairs (lane to lane match) are tuned using the Interactive DiffPair Length Tuning option. During tuning, the Properties panel shows up with settings. Diffpairs need two tuning actions, a single-ended one for phase tolerance match (with a 5mil rule), and a differential one for lane-to-lane matching. The first thing for phase tolerance is to twist at the pads manually, and only after that we use meander tunes.

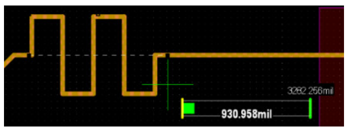

Interactive Length Tuning
Interactive Diff Pair Length Tuning


Pattern
Accordion | Trombone | Sawtooth
Hide preview
Max Amplitude 30mil Step 5mil
Space 20mil Step 10mil
Miter 0% Step 5%
Style Mitered Lines

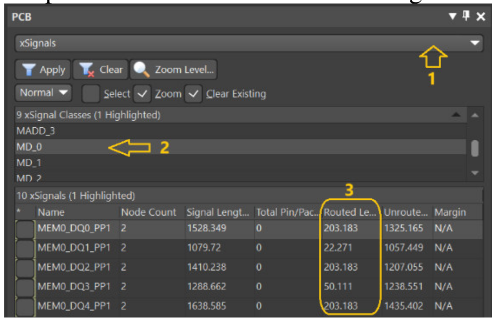




While we are manually tuning the traces, we can **measure** the trace lengths. The Altium Designer's on-screen Length Meter/ Gauge pops up during tuning, which shows the signal lengths of the currently edited trace in real time. The PCB panel also shows all Net/DP/XS/XSC



lengths as a table, but we have to select the drop-down menu for object type first, and we have to complete the last trace edit for it to update. The CM also shows XS lengths.


PCB

xSignals

Apply | Clear | Zoom Level...

Normal | Select | ✓ Zoom | ✓ Clear Existing

1

9 xSignal Classes (1 Highlighted)

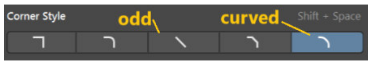
MADD_3
MD_0 2
MD_1
MD_2

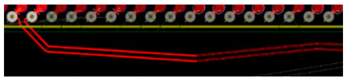
10 xSignals (1 Highlighted)

3

*	Name	Node Count	Signal Lengt...	Total Pin/Pac...	Routed Le...	Unroute...	Margin
	MEM0_DQ0_PP1	2	1528.349	0	203.183	1325.165	N/A
	MEM0_DQ1_PP1	2	1079.72	0	22.271	1057.449	N/A
	MEM0_DQ2_PP1	2	1410.238	0	203.183	1207.055	N/A
	MEM0_DQ3_PP1	2	1288.662	0	50.111	1238.551	N/A
	MEM0_DQ4_PP1	2	1638.585	0	203.183	1435.402	N/A

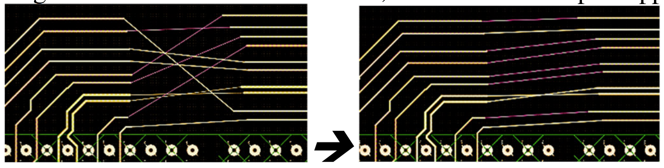
For long 8Gig+ SERDES signals, we use **<u>wavy routing</u>** and odd-angle routing, to mitigate fiber weave effect. In Altium this is achieved by pressing TAB while routing, and selecting the 3rd corner style on the Properties tab. SERDES links at 8Gbps+ should use the curved corners (5th style) when possible. Switch between curved for short and angled for long segments. Curved corners also enable snake fanout under offset-grid hex-BGAs.

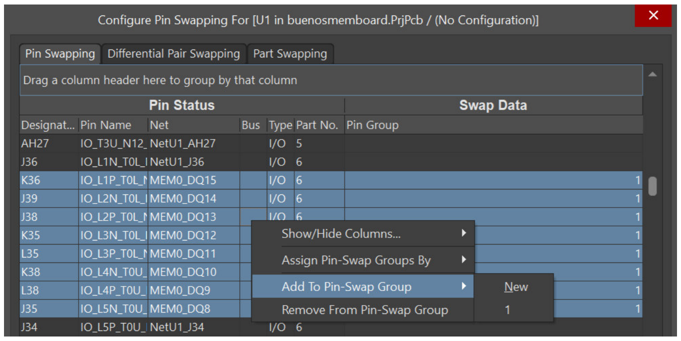




For low-cost and aerospace boards **<u>teardrops</u>** are used, that we enable: Tools> Teardrops.

<u>Pin swapping</u> 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 Altium, once we are done with the escape routes and some long distance routes from both ends, then we can set up swapping.




Configure Pin Swapping For [U1 in buenosmemboard.PrjPcb / (No Configuration)]

Pin Swapping | Differential Pair Swapping | Part Swapping

Drag a column header here to group by that column

Pin Status						Swap Data
Designat...	Pin Name	Net	Bus	Type	Part No.	Pin Group
AH27	IO_T3U_N12_	NetU1_AH27		I/O	5	
J36	IO_L1N_T0L_I	NetU1_J36		I/O	6	
K36	IO_L1P_T0L_I	MEM0_DQ15		I/O	6	1
J39	IO_L2N_T0L_I	MEM0_DQ14		I/O	6	1
J38	IO_L2P_T0L_I	MEM0_DQ13		I/O	6	1
K35	IO_L3N_T0L_I	MEM0_DQ12				1
L35	IO_L3P_T0L_I	MEM0_DQ11				1
K38	IO_L4N_T0U_	MEM0_DQ10				1
L38	IO_L4P_T0U_	MEM0_DQ9				1
J35	IO_L5N_T0U_	MEM0_DQ8				1
J34	IO_L5P_T0U_I	NetU1_J34		I/O	6	

Steps: Set the swap behavior to move net labels not pins: Project> Project Options> Options> Adding Rem Net Labels = on, while Change Sch Pins = off. In the design, select the component, then on the Properties panel> Swapping Options> Enable Pin Swapping. Next, on the same component Rightclick > Component Actions > Configure Pin/Part Swapping then upper tab PinSwapping or DiffPair swapping, then find/select multiple nets. Then on the selection list rightclick> Assign PS Group> New. A number appears in the PinGroup column. OK to close. Next is Tools> Pin/Part Swapping> Automatic Pin/Net Optimizer, then follow instructions. It swaps nets automatically. Finally, we run Design> Update Schematic, then Validate, Execute, Close. If it fails with an error, then we find the "Remove Pins From Nets" and "Add Pins to Nets" sections, take a screenshot, paste into Paint, as our manual swap list, then manually swap net labels or off-sheet connectors in the schematic, then save SCH, go back to PCB and Design> Import Changes.

<u>Backdrilling</u> control is achieved by creating a CM> AllRules > HighSpeed > Maximum Via Stub Length rule, define stub length and net class, board side, backdrill oversize (each side of the via barrel). 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. We have to avoid over drilling press-fit connector pins into their MBL region, by using a room or custom query ('IsPin') and a longer stub value (MBL minus depth). The start/stop layers are defined by adding a backdrill pair in the Design> Layer Stack Manager, upper right ☰ drop down menu select "backdrills", a new tab appears. On the backdrills tab add several BD layer pairs (from bottom or top to one layer away the routing MNC layer). We also need to take care of large antipads on planes, and route keepouts on signal layers, around backdrilled vias, as described in the section about voids. We get a separate NC drill file for each depth.

The **<u>Rules&Violations</u>** panel lists all the DRC violations. We can browse by category/rule/violation and click to highlight. We can also see them on the PCB panel in red, or in the CM.

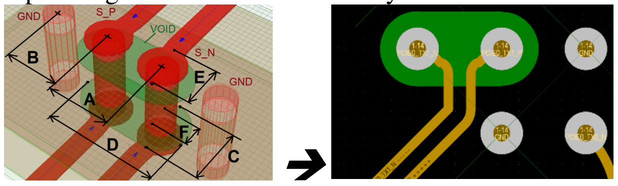
7. Signal Integrity

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, constraints, timing-based trace length calculations, trace impedance control, crosstalk control, ground returns, stackup design, materials, backdrilling, via impedance optimization, loss budgets and insertion loss control techniques.

Most **<u>SI simulations</u>** 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 Altium's built-in Power Analyzer by Keysight, that requires a downloadable add-on install with a separate license. Run: Tools> Visual Power Analyzer. It has an automated setup.

For differential SERDES links operating at 8Gbps/lane or above we need to ensure that the **<u>impedance of the via structures</u>** also comply to the impedance requirement. We do this by recreating the via structure in HFSS or Simbeor, optimize the structure dimensions (via-to-via spacing, via diameter, and manual void shape/size), then adjusting them in simulation, then replicating the structure in Altium layout to match the dimensions.

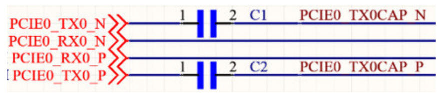


8. Typical Examples

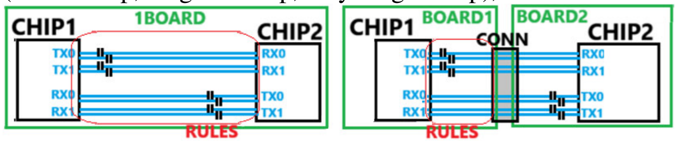
These examples are based on the CM-flow, not the DRE-flow.

8.1. PCIe Gen4 SERDES bus design

The lane-to-lane matching groups, or diffpair class (DPC) 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 of the cap in the schematic.



If we had a DC coupled Hyper Transport bus, then one DPC 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 DPCs, one class for the long segments including both TX and RX signals, and the other class 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 on RX signals on our board, and now we have 3 types of diffpairs (short to cap, long from cap, very long no-cap), so we need 3 DPCs.



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 25dB*0.7=17.5dB. 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 trace length 17.5dB/2=8.75". On our motherboard we have 3 DPCs, the longest one can have 8.75" max set in CM> AllRules> HighSpeed> Length, while the other 2 DPCs 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> Physical, then the diffpair classes in Design> Classes. Every separate refdes-to-refdes interface is a separate class. The impedance-driven width is in CM>Physical (set on diffpair class). The _P/_N phase tolerance matching is typically 5mils, that we set in CM> Electrical> Diffpairs on the diffpair class. We also set up a constraint for crosstalk control on all PCIe signals with CM> AllRules> HighSpeed> Parallel Segment. For any reference clocks, we only need diffpair rules, phase tolerance matching and DP class trace width rules.

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 between diffpairs, maybe within 5 mils. We apply this in CM> AllRules> HighSp> Matched Length, on each DPC separately.

During **routing**, we route all diffpairs using interactive DP 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 DP length tuning. We can monitor our progress on the PCB panel. Finally, we run the DRC, and check the rules and violations panel to see what is left.

We also need to set up **backdrilling**. In the Layer Stack Manager, we define all BD depths. Then we define the max stub length on the DPCs at: CM> AllRules> HighSpeed> BackDrilling. Any BD at the press-fit connector pins must be limited, as to not cut into the minimum barrel length of the connector.

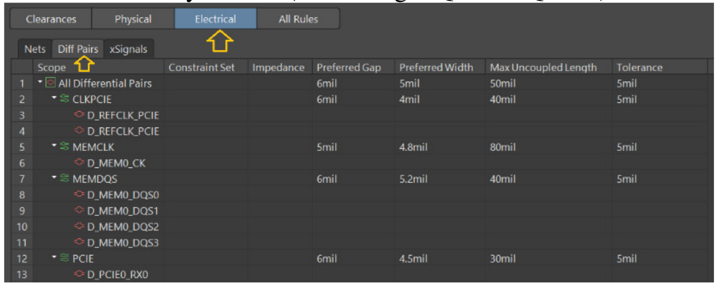
8.2. DDR4 Memory-Down design

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.

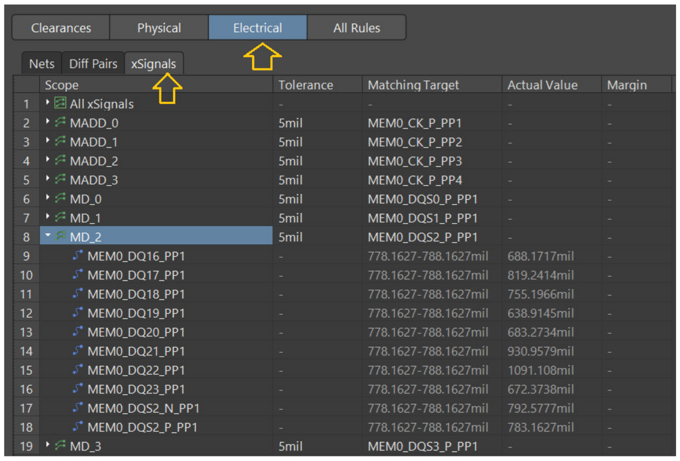
We have to create objects for the **address bus** signals: A net class for trace width rules, xSignals for each CPU-to-DRAMn component pair on every address bus signal, one XSC (match 5mil) for each CPU-to-DRAMn component pair (containing address and clock XS). Then we enter the width data for the net class in CM, and match-length data for each XSC. If we had 25 signals and 4 DRAM chips, then we will have 4*25=100 xSignals and 4 XSCs.

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

The data bus: Since we have DQS diffpairs and DQ SE signals in one matched group, we have to create xSignals for every signal, and XSCs (match 5mil) for every lane for matching, and use net classes only for impedance, otherwise we would have two different width rules on DQS. We will need one net class for DQ width and one DP class for DQS width. We will need as many XSCs as the number of data byte lanes (containing DQ and DQS XS).



	Scope	Constraint Set	Impedance	Preferred Gap	Preferred Width	Max Uncoupled Length	Tolerance
1	All Differential Pairs			6mil	5mil	50mil	5mil
2	CLKPCIE			6mil	4mil	40mil	5mil
3	D_REFCLK_PCIE						
4	D_REFCLK_PCIE						
5	MEMCLK			5mil	4.8mil	80mil	5mil
6	D_MEM0_CK						
7	MEMDQS			6mil	5.2mil	40mil	5mil
8	D_MEM0_DQS0						
9	D_MEM0_DQS1						
10	D_MEM0_DQS2						
11	D_MEM0_DQS3						
12	PCIE			6mil	4.5mil	30mil	5mil
13	D_PCIE0_RX0						



	Scope	Tolerance	Matching Target	Actual Value	Margin
1	All xSignals	-	-	-	-
2	MADD_0	5mil	MEM0_CK_P_PP1	-	-
3	MADD_1	5mil	MEM0_CK_P_PP2	-	-
4	MADD_2	5mil	MEM0_CK_P_PP3	-	-
5	MADD_3	5mil	MEM0_CK_P_PP4	-	-
6	MD_0	5mil	MEM0_DQS0_P_PP1	-	-
7	MD_1	5mil	MEM0_DQS1_P_PP1	-	-
8	MD_2	5mil	MEM0_DQS2_P_PP1	-	-
9	MEM0_DQ16_PP1	-	778.1627-788.1627mil	688.1717mil	-
10	MEM0_DQ17_PP1	-	778.1627-788.1627mil	819.2414mil	-
11	MEM0_DQ18_PP1	-	778.1627-788.1627mil	755.1966mil	-
12	MEM0_DQ19_PP1	-	778.1627-788.1627mil	638.9145mil	-
13	MEM0_DQ20_PP1	-	778.1627-788.1627mil	683.2734mil	-
14	MEM0_DQ21_PP1	-	778.1627-788.1627mil	930.9579mil	-
15	MEM0_DQ22_PP1	-	778.1627-788.1627mil	1091.108mil	-
16	MEM0_DQ23_PP1	-	778.1627-788.1627mil	672.3738mil	-
17	MEM0_DQS2_N_PP1	-	778.1627-788.1627mil	792.5777mil	-
18	MEM0_DQS2_P_PP1	-	778.1627-788.1627mil	783.1627mil	-
19	MD_3	5mil	MEM0_DQS3_P_PP1	-	-