**PCB Design With FGPAs and BGA Packages by Kyler Callahan**

*This tutorial assumes that the reader has a basic knowledge of how to use Mentor Graphics PADs, and DxDesigner. If you have not already please go through the PADs evaluation guide located in: C:\PADS\_ES\_Evaluation\Document\PADS ES Suite Evaluation Guide.pdf*

**Introduction:**

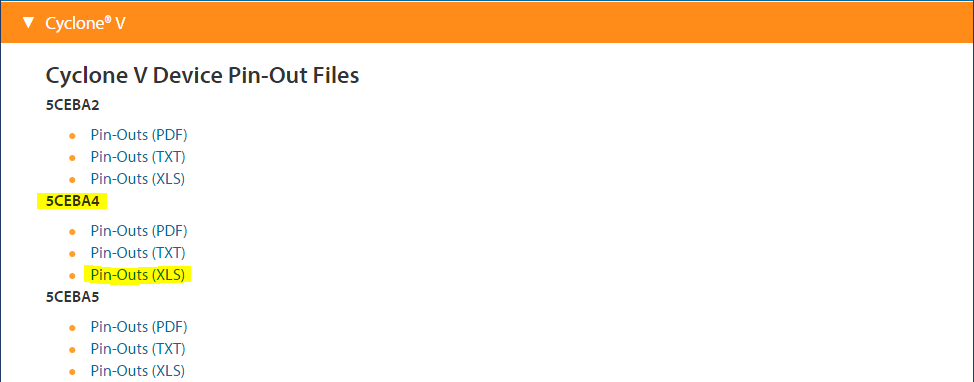
This tutorial will cover techniques of how to work with FPGAs and BGA packages using the *Mentor Graphics Suite*. The programs we will be focusing on using will be: PADS Layout, PADS Router, and DxDesigner. Topics covered will be: Acquiring necessary resources for using the specified chip, creating FPGA symbols in DxDesigner and PADS, basic break-out techniques, choosing decoupling capacitors, and further recommended reading.

This tutorial will be targeting the *5CEBA4U15C7N* Altera Cyclone V device. However many topics discussed in this tutorial will carry over to non-Altera devices and even non-FPGA devices.

**Acquiring necessary resources:**

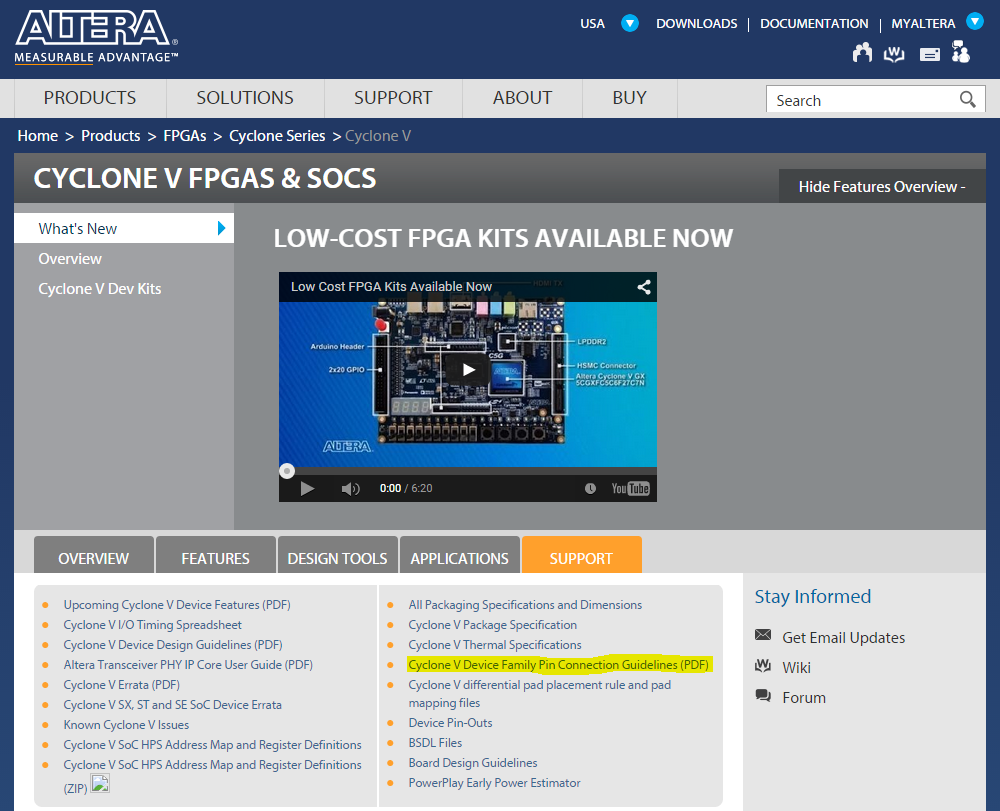
1) The first and foremost important things that need to be done before we do anything else, is finalizing a device and device package type. The requirements for doing this changes from project to project and will not be covered in this tutorial.

2) Once you have a *device* and *device package type* you need to find your pin-out files. Check with your device manufacturing website to find the pin-out file of your device. The pin-out may also be located in the device data sheet.



2a) If an XLS Pin-out is available it is recommended to use that as we will be using Excel to make our device symbols for DxDesigner.

3) Next acquire the *Device Connection Guidelines* (If available). This document will prove useful in determining how to hook up critical pins in the device.

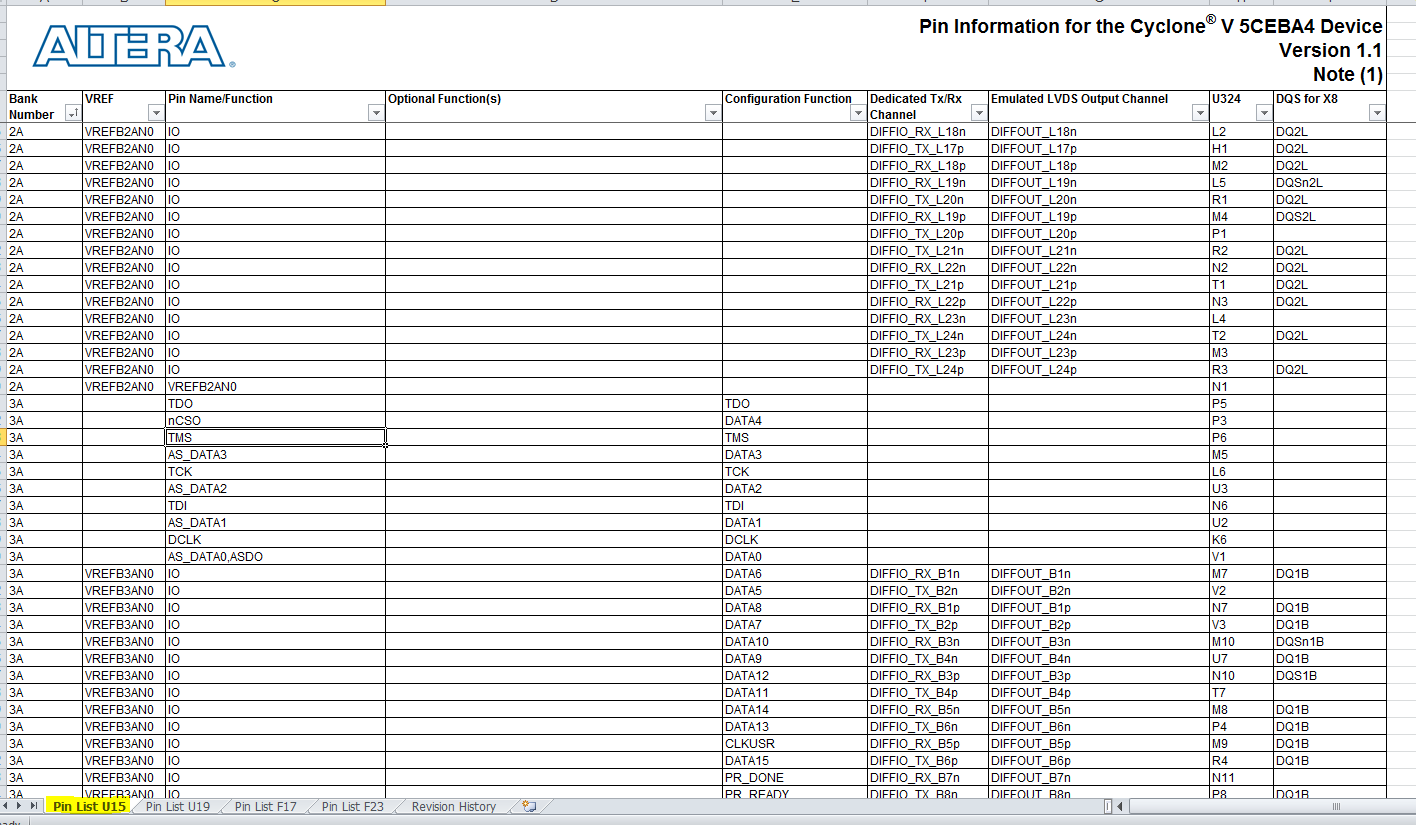


4) It is also recommended to look through other documentation provided by the device manufacturer to better familiarize yourself with how to hook up the device and whether there may be recommended designs or layouts.

*Note: You may skip step 4 if you are following along in this tutorial and trying to replicate the example as all of that work has been done for you.*

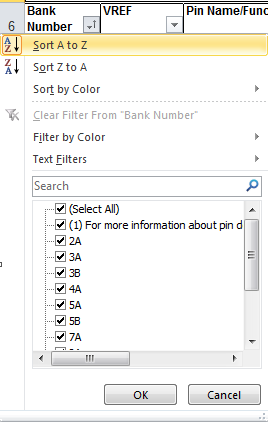
**Creating FPGA Symbols in DxDesigner and PADS:**

1) Open the pin-out file in Excel and browse to the package type that you have chosen. For our example we are using the *U15 (Stands for UltraFineBGA at .8mm pin pitch with a 15mmx15mm* package)



*The pin-out file should look something like this for Altera devices*

1a) We are going to break up the pin-outs into banks so we can create manageable symbols. Start by sorting the document by bank number.



2) Now that you have the pins sorted by bank number you will need to decide how to split up the banks. It helps if you have a single Excel file with separate sheets for each symbol you are going to make. For each sheet make note of at least:

* Pin Name/ Function
* Pin Number

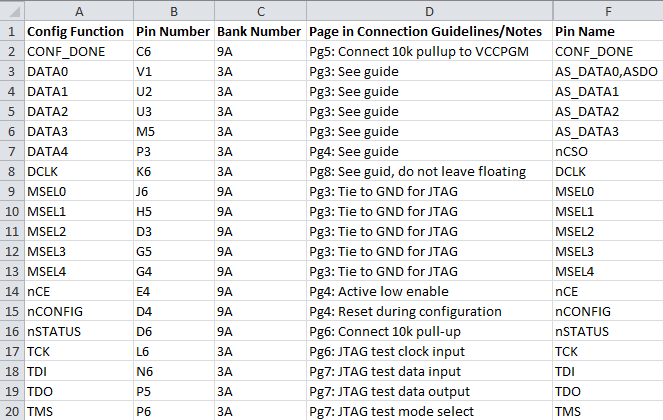
You may find it useful to also include

* Bank Number
* Page number in the Connection Guideline and any notes about connection guidelines

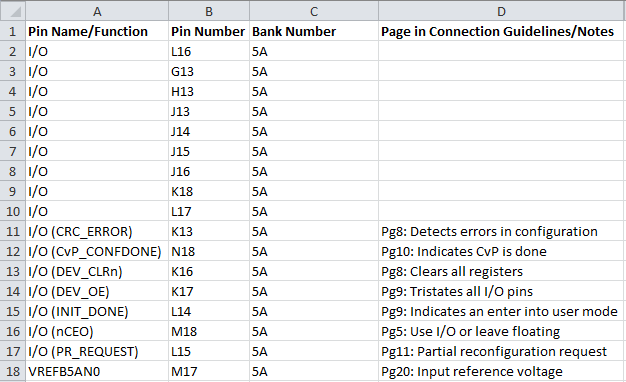
In this particular example we will have 18 different symbols:

C:\Users\kyler.callahan\Pictures\FPGA Tutorial\banksplits.PNG

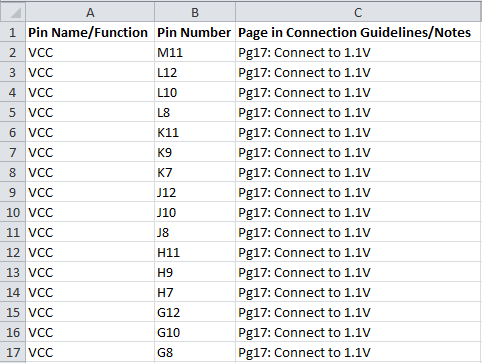
* Config Pins
  + These are pins used for configuring the device. They may come from other banks, if they do leave them off of that bank when you make it



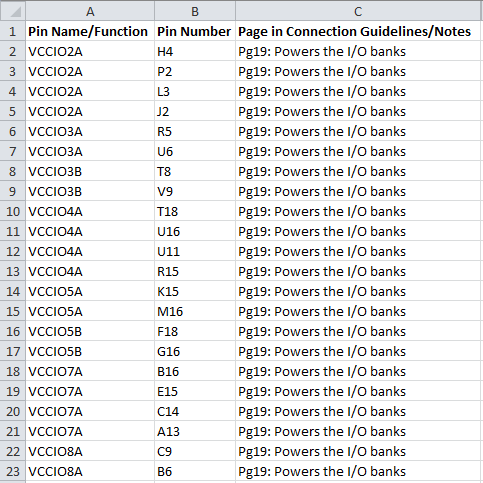
* Banks 2A, 3A, 3B, 4A, 5A, 5B, 7A, and 8A
  + These are all of the I/O pins with included VREF pins. If an I/O pin has an additional function it is indicated. Example below.



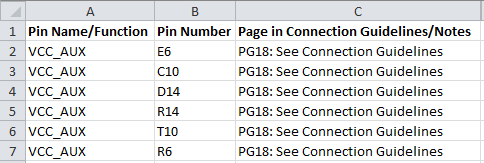
* VCC pins
  + These are the core voltages for the device



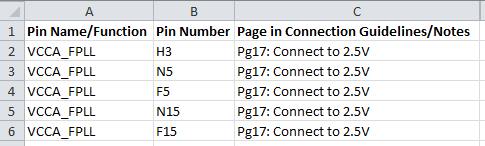
* VCCIO pins
  + These are the pins that determine the voltage that each bank is at.



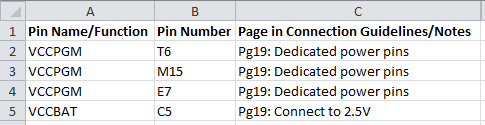
* VCC\_AUX
  + These pins are the AUX power pins



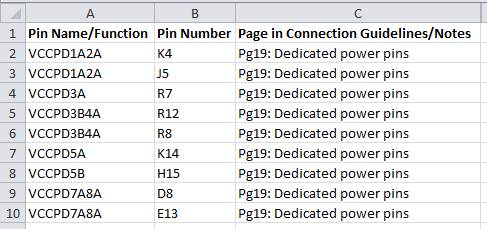
* VCCA\_FPLL
  + These are the power pins for the PLL



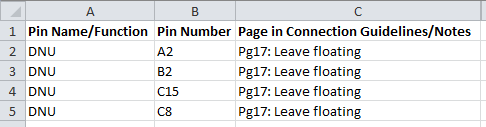
* VCCPGM
  + Pins related to the PGM function and extra VCC BAT pin



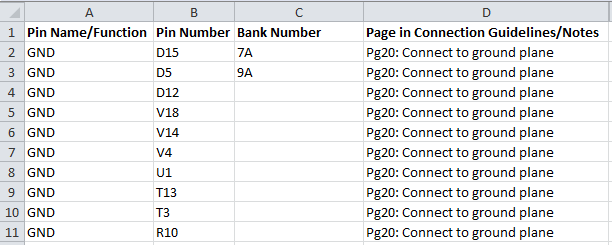
* VCCPD
  + Power pins related to the pre-driver voltages



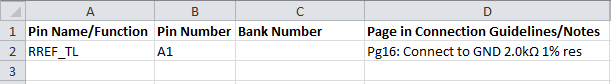
* DNU
  + Do not use pins. While these pins are not used you will still need a symbol for them



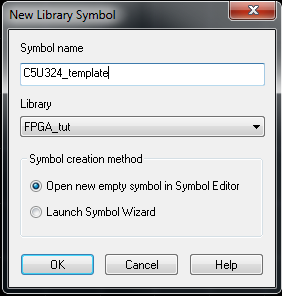
* GND
  + Ground pins. There are more pins than shown below, it’s just been edited for space reasons



* RREF\_TL
  + Special pin related to transceiver functions



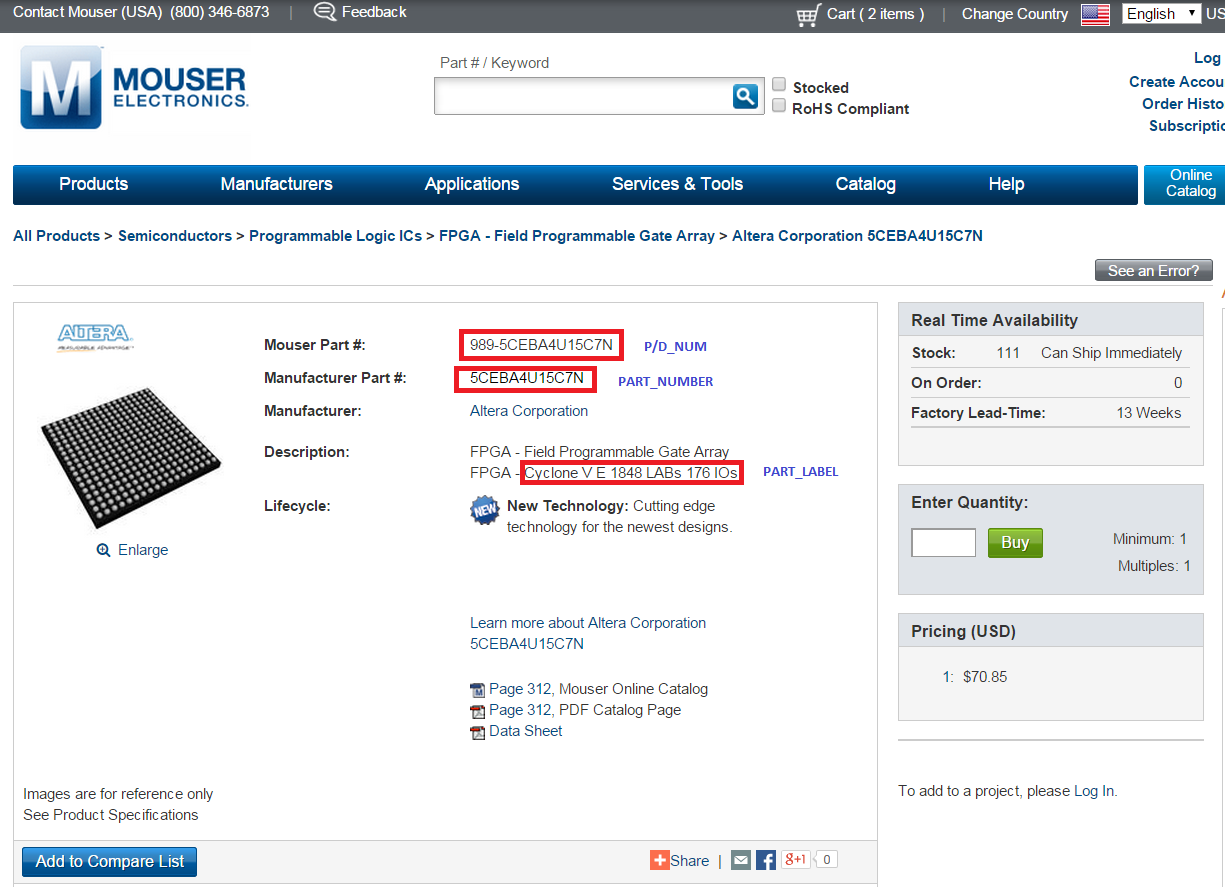
3) Next we create the symbols in DxDesigner. Open DxDesigner and give your symbol name something descriptive. This symbol will be our template for making the 18 other symbols. For this particular example we will be using the name C5U324\_template. The C5 lets us know it’s a Cyclone 5, U324 indicates the package type, and template shows that it is a template.

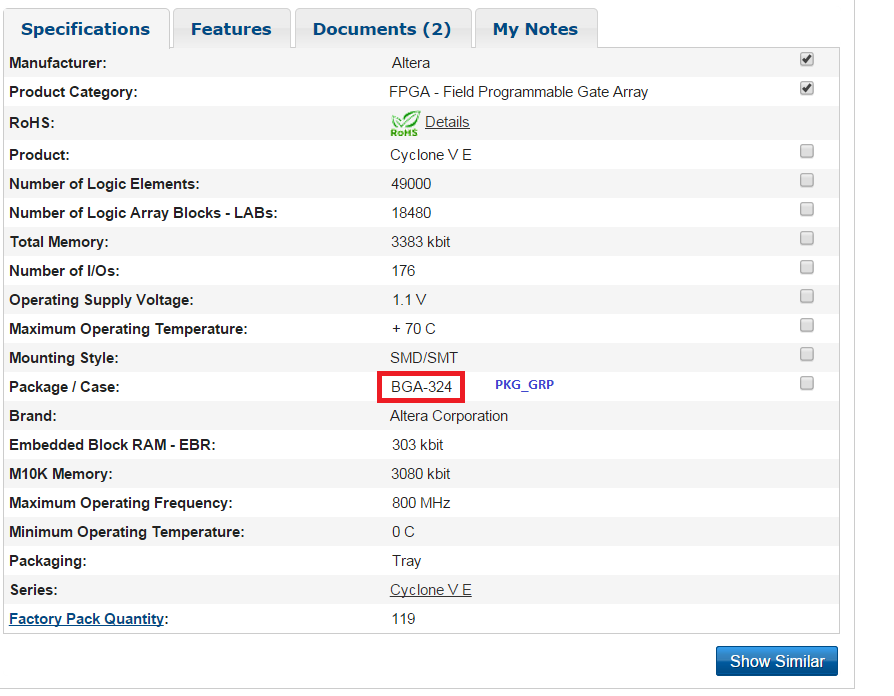


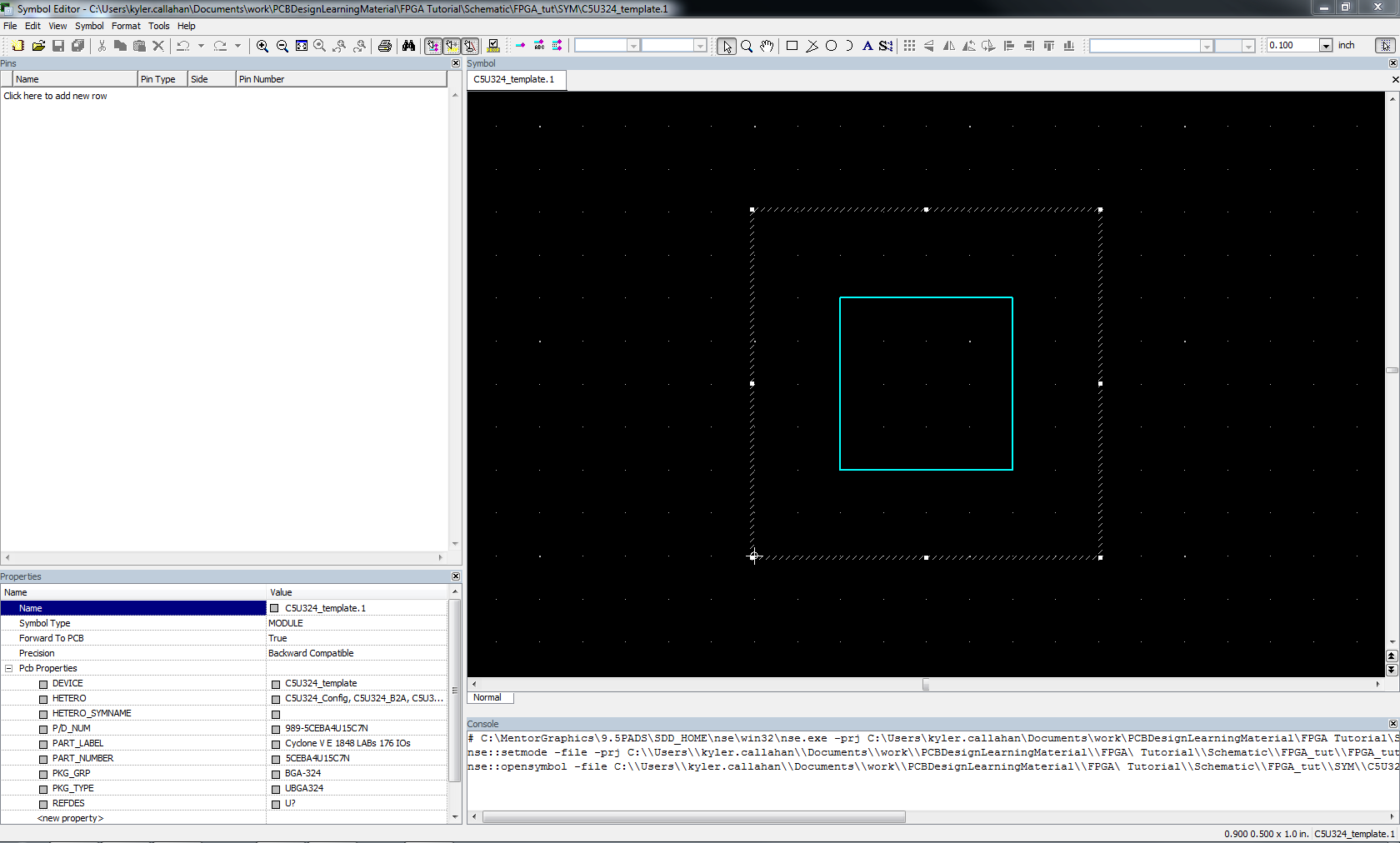
4) Add the following property fields to your PCB properties and fill out

* DEVICE
  + This should be something short but descriptive of the device. For this example we will use C5\_U324 for the same reasons as above.
* HETERO and HETERO\_SYMNAME
  + HETERO is a field of *all* the HETERO\_SYMNAMES separated by commas.
    - We are going to split our FPGA up into multiple symbols, this field is necessary for DxDesigner to associate all the separate symbols with each other.
    - This field can only accommodate up to 255 character so make your HETERO\_SYMNAMES short
    - The field for this example looks like this: C5U324\_Config, C5U324\_B2A, C5U324\_B3A, C5U324\_B3B, C5U324\_B4A, C5U324\_B5A, C5U324\_B5B, C5U324\_B7A, C5U324\_B8A, C5U324\_VCC, C5U324\_VCCIO, C5U324\_VCC\_AUX, C5U324\_VCCA\_FPLL, C5U324\_VCCPGM, C5U324\_VCCPD, C5U324\_DNU, C5U324\_GND, C5U324\_RREF\_TL
  + HETERO\_SYMNAME is the field that is the symbol name of each individual symbols. This field should match the Name
    - Leave this field blank for now. We will fill it out with each individual component. This is our template for now.
    - Make these names descriptive but short. For example all of the names in this example start with “C5U324\_” and have a descriptive name after the underscore such as “Config” or “B2A” to represent the Config symbol or the Bank 2A symbol.
* P/D\_NUM
  + This is the Part Distributer number. This is the number used by the distributer (IE: DigiKey or Mouser) in their catalog.
* PART\_LABEL
  + This is a quick part description
* PART\_NUMBER
  + The is the Part Manufacturer’s number
* PKG\_GRP
  + This is the group of packages this part belongs to. EG: BGA, 16-UFQFN, 100-QFP… etc
* PKG\_TYPE
  + This is the name of the footprint in PADS that this component will use
* REFDES
  + This is the reference designator as to what logic group this part belongs to. For more info on this visit this page <http://en.wikipedia.org/wiki/Reference_designator>

Much of this information can be found on the part’s page of the distributer’s website.





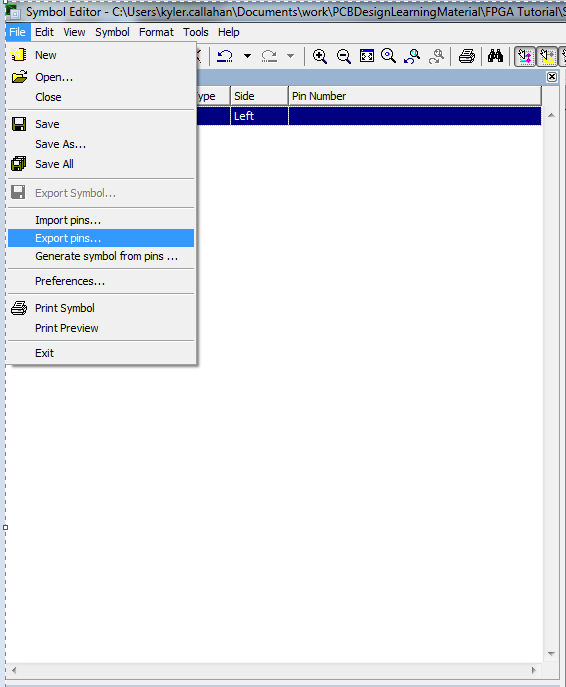


When you’re finished your project should look something like this

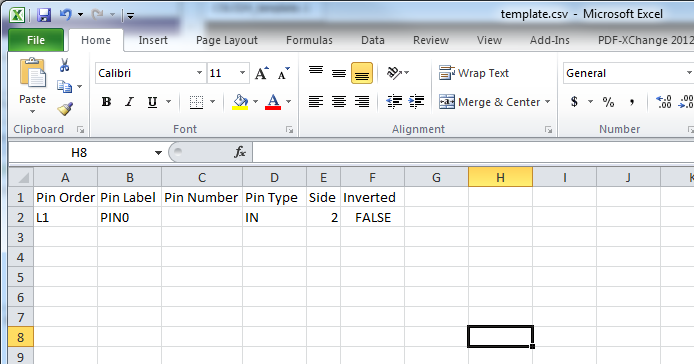
5) Create a CSV template that you will use for all of your symbols

5a) Place a single pin

5b) Export pins as CSV: File -> Export pins…

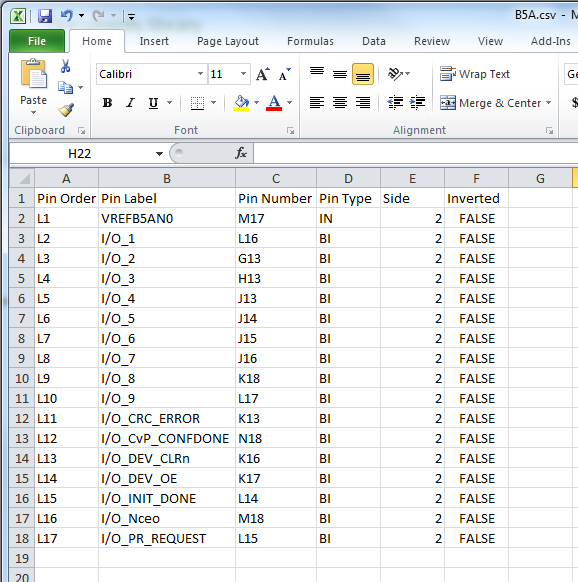


The file should look something like this

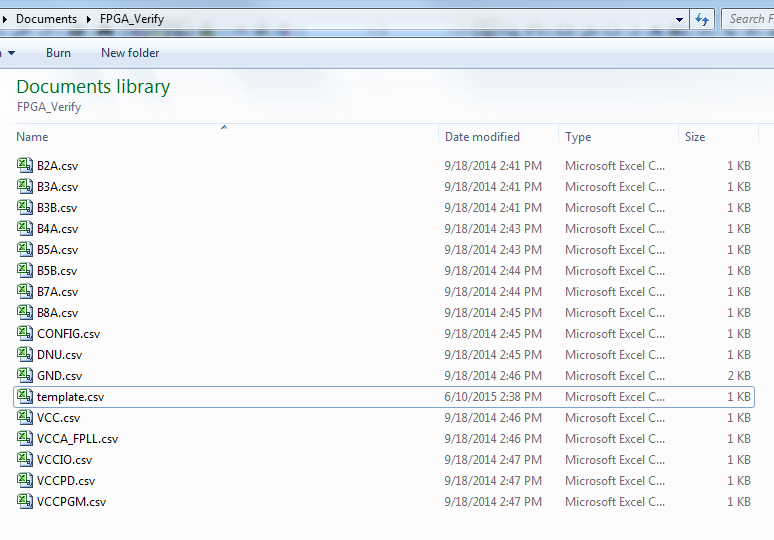


* Pin Order (Accepted inputs: T#,B#,L#,R#)
  + The order that the pins will be in in the *pins* field with respect to it’s orientation
* Pin Label
  + The label on the pin, make this descriptive. No two labels can match.
* Pin Number
  + The pin number. This is the number used by PADS to assign pins
* Pin Type (Accepted inputs: IN,OUT,BI,ANALOG,OCL,OEM,TRI,POWER,GROUND,TERMINAL)
  + This field is functionally insignificant to what we are doing right now but you may set the values if you wish, otherwise just leave them as “BI”
* Side (Accepted inputs: 0 = top, 1 = bottom, 2 = left, 3 = right)
  + This is the side that the pin is located on. It should match the corresponding Pin Order
* Inverted
  + Leave FALSE

6) Next edit the file to match the name and pin number from the sheets you made in your other Excel file. Save the file as something descriptive. It should look something like this when you’re done.

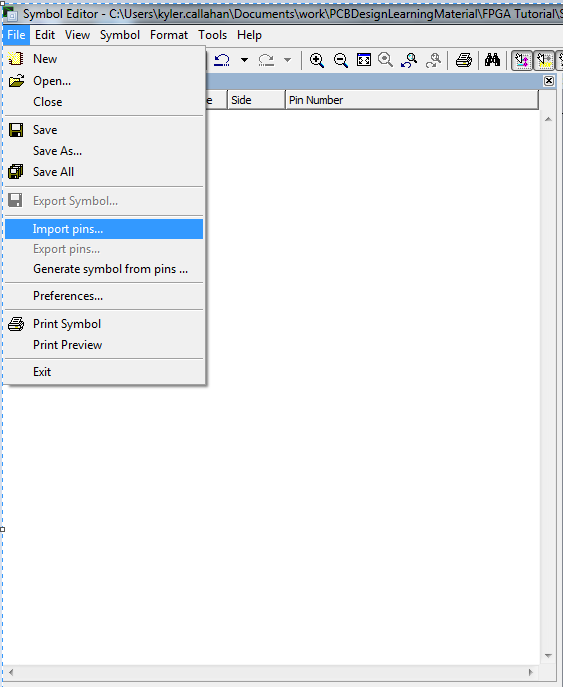


You may want to save all of your csv files to a single folder for ease of use and re-use.



**Be sure to triple check that each pin number matches its appropriate bank and pin name/function. Making a mistake here be very costly in the future.**

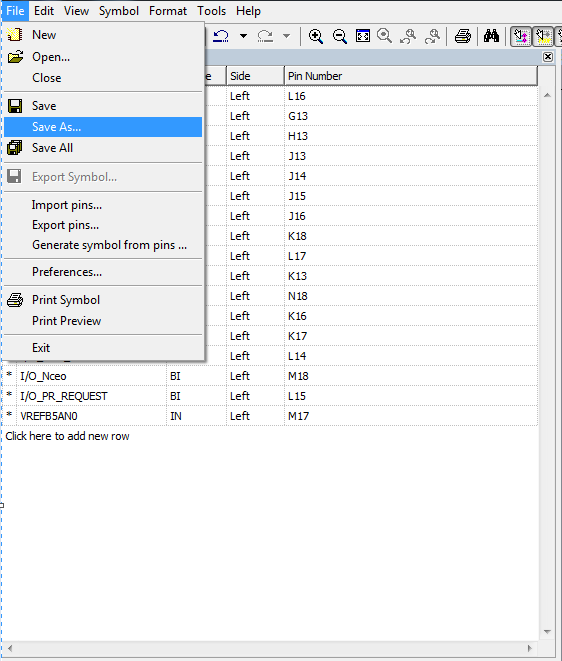
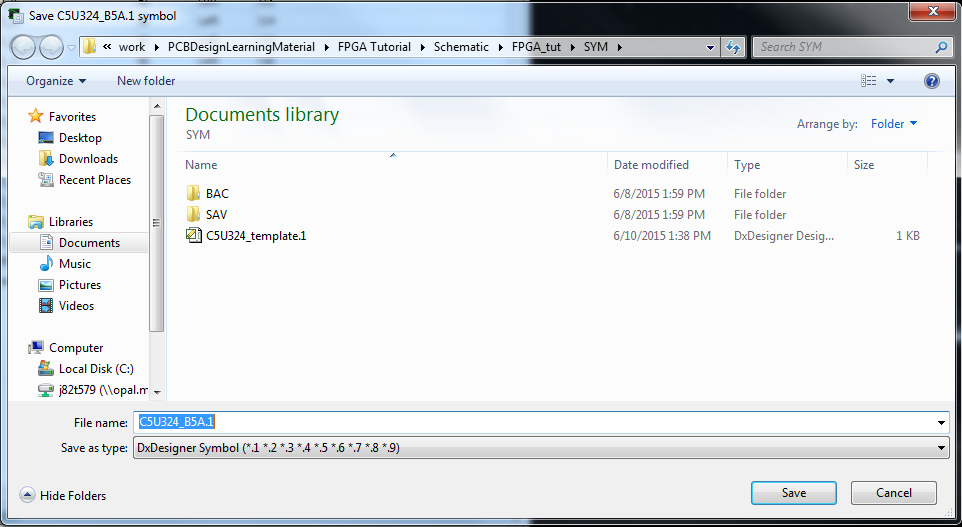
7) Import your CSV’s as pins. File -> Import Pins

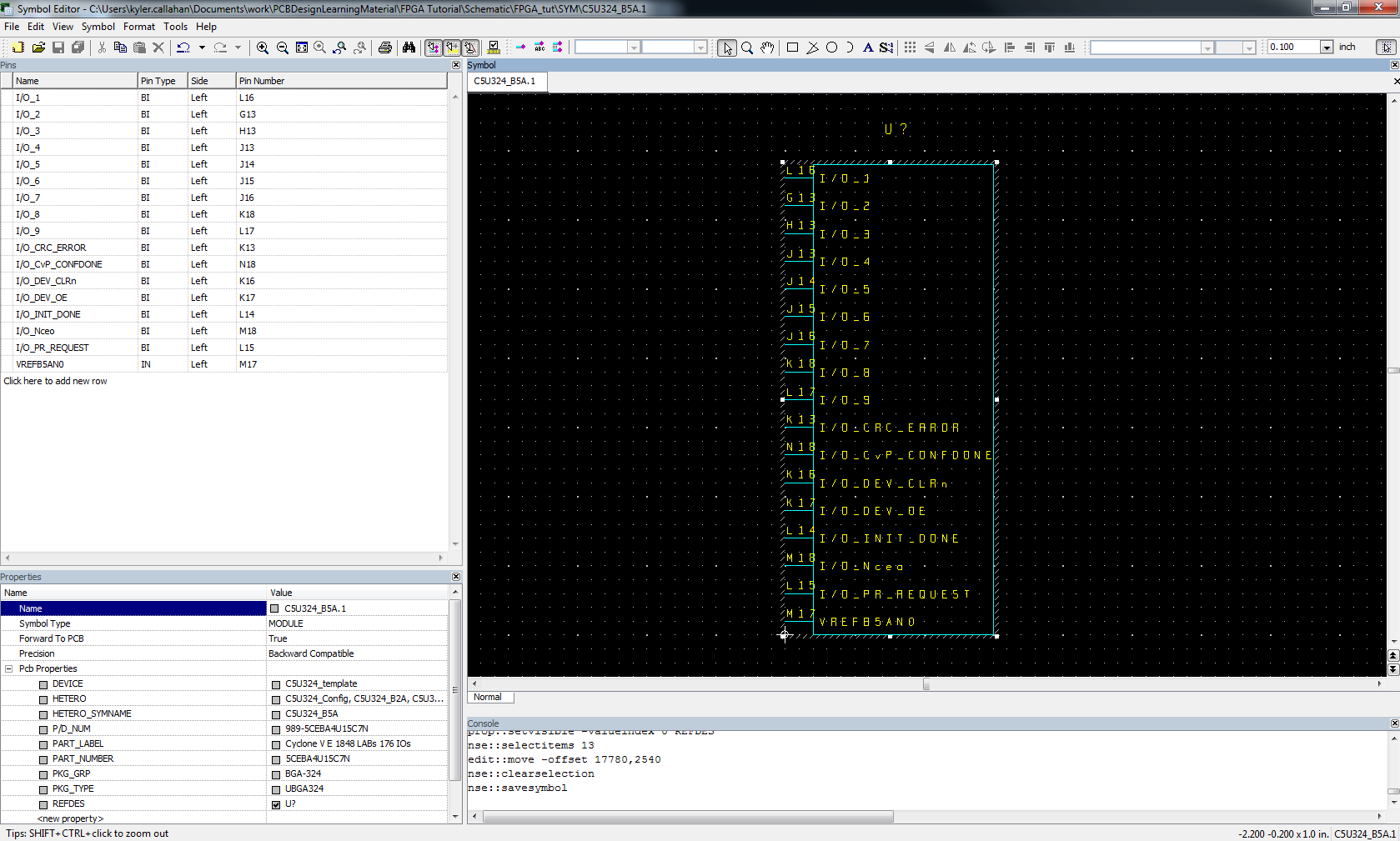


8) Edit Symbol Name, and HETERO\_SYMNAME to match the symbol this is associated with and place the pins. Then **SAVE AS A NEW PART!**

**Important note!**

**Do *not* press “save” or ctrl+s when editing from your template. DxDesigner will save the wrong name to the wrong place and mess up your template and whatever symbol you were trying to make. Make sure to press “Save as” instead.**

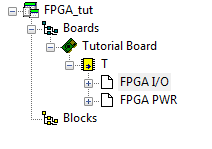


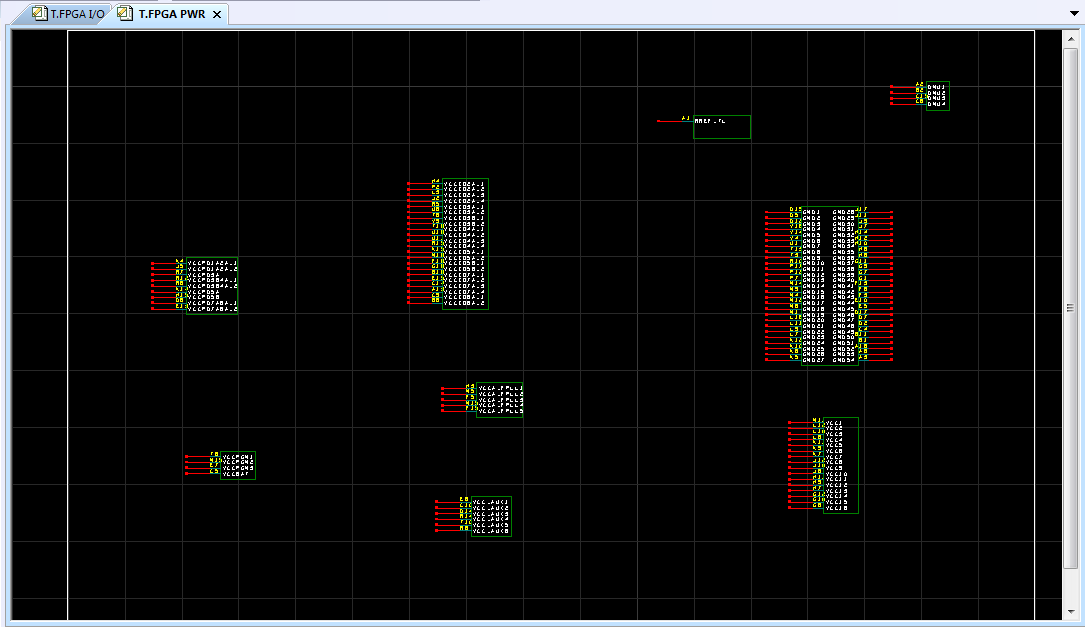
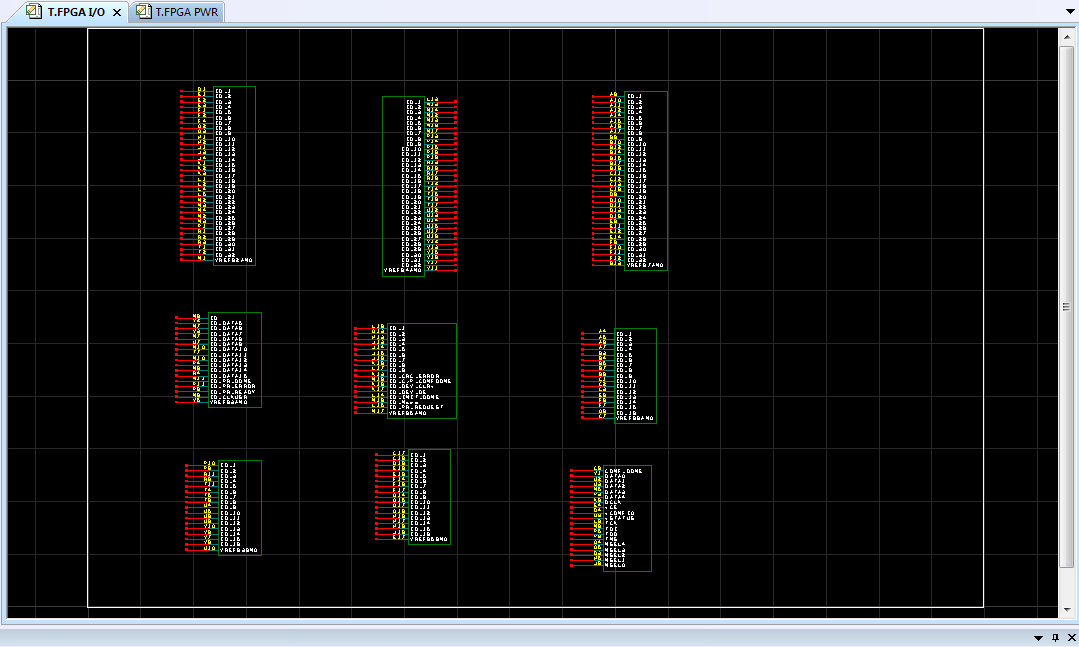


Your symbol should look something like that when you’re finished.

8) Reload your template file and repeat step 7 as needed.

9) After you have created all of your symbols you need to place them in a schematic in DxDesigner. If you have a lot of symbols it is recommended to split them across multiple sheets by function.



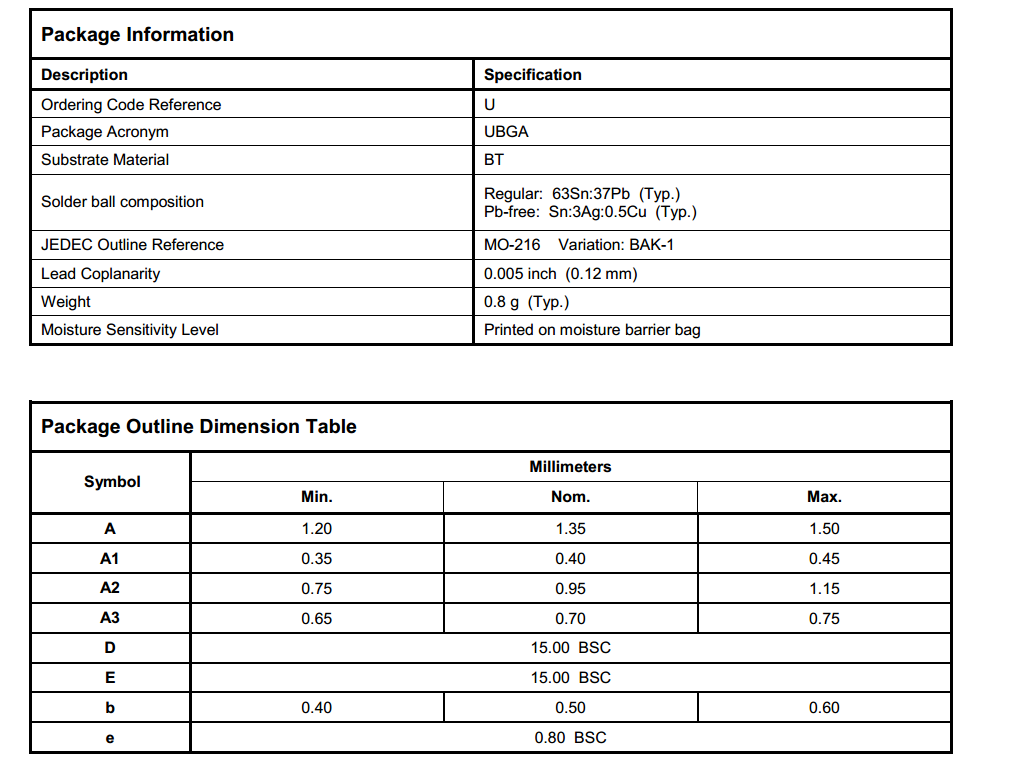


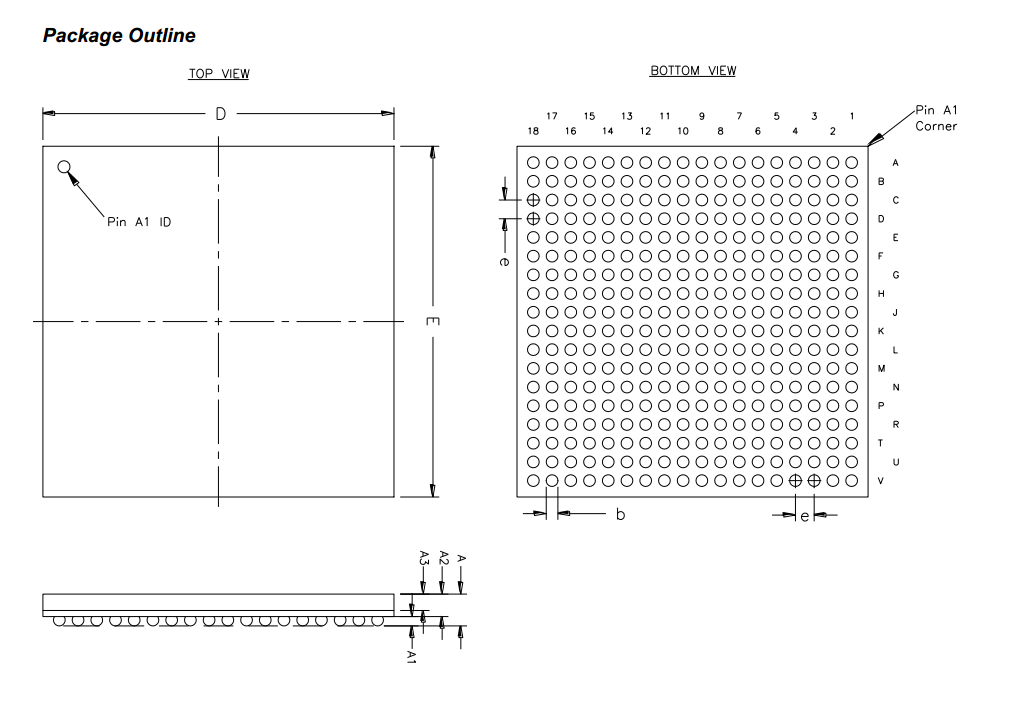
10) Next open PADS Layout and make a new library part: File -> Library -> Decal -> New

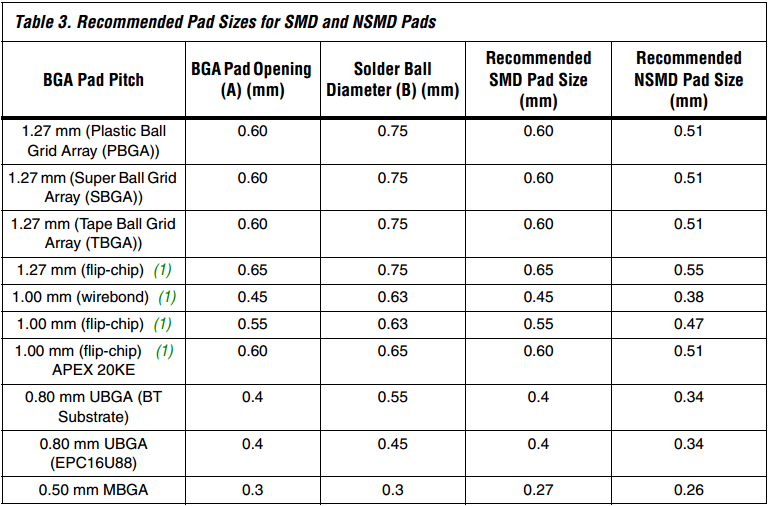
And launch the decal wizard



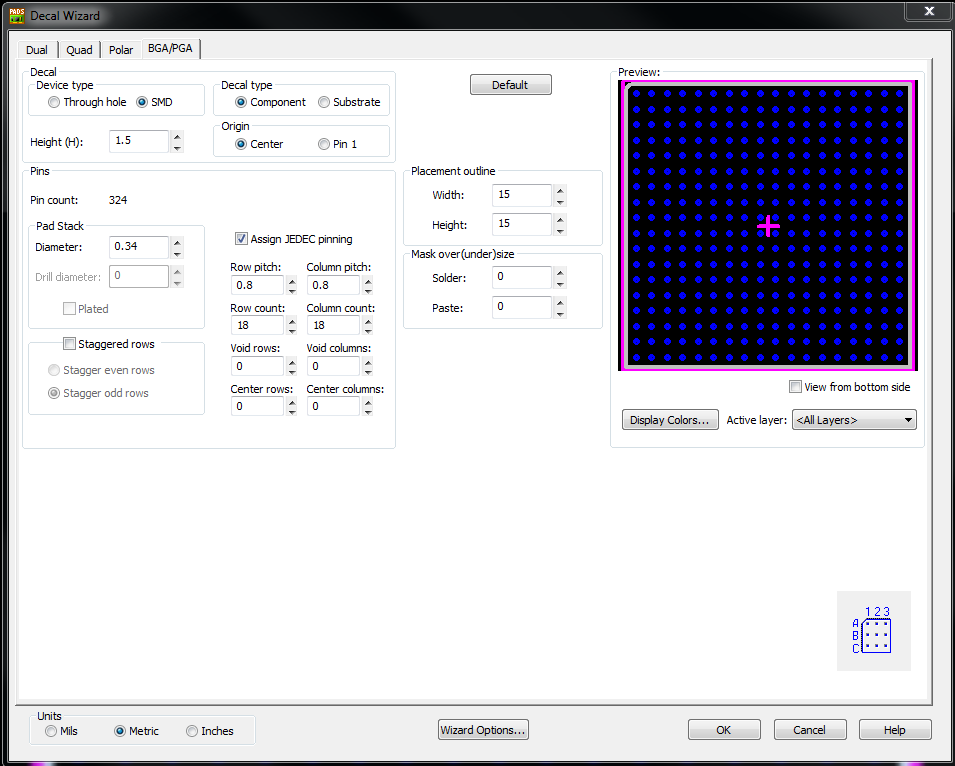
11) Go to the tab that matches your package type and fill out the necessary information. For this example we are making a BGA/PGA. If the info is not available in the data sheet there may be a separate document containing package information.



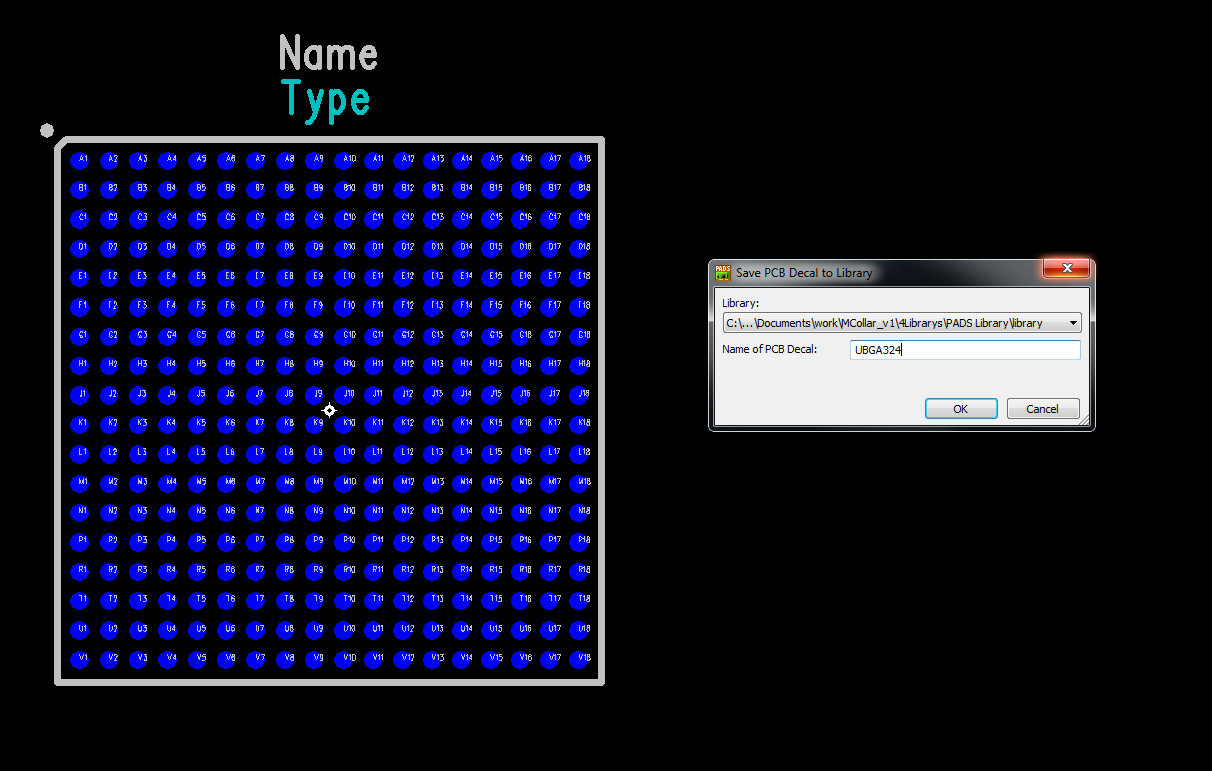




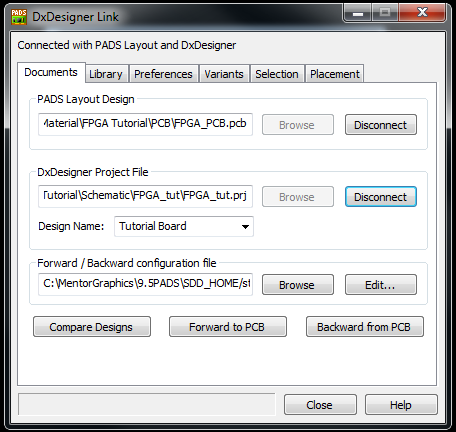
Using the table taken from *Designing With High-Density BGA Packages for Altera Devices* we can determine what the landing pad sizes are going to be. For this example we are using a 0.8 UBGA following the Recommended NSMD Pad Size. For further explanation please refer to the app note.



12) Make sure the pin numbering matches that of the data sheet, add a pin 1 indicator, and save the decal exactly what you put in the PKG\_TYPE field of the DxDesigner decal.

C:\Users\kyler.callahan\Pictures\FPGA Tutorial\pkg_type.PNG

13) Forward your design to your PADS project. If you did everything correctly then your part should show up!

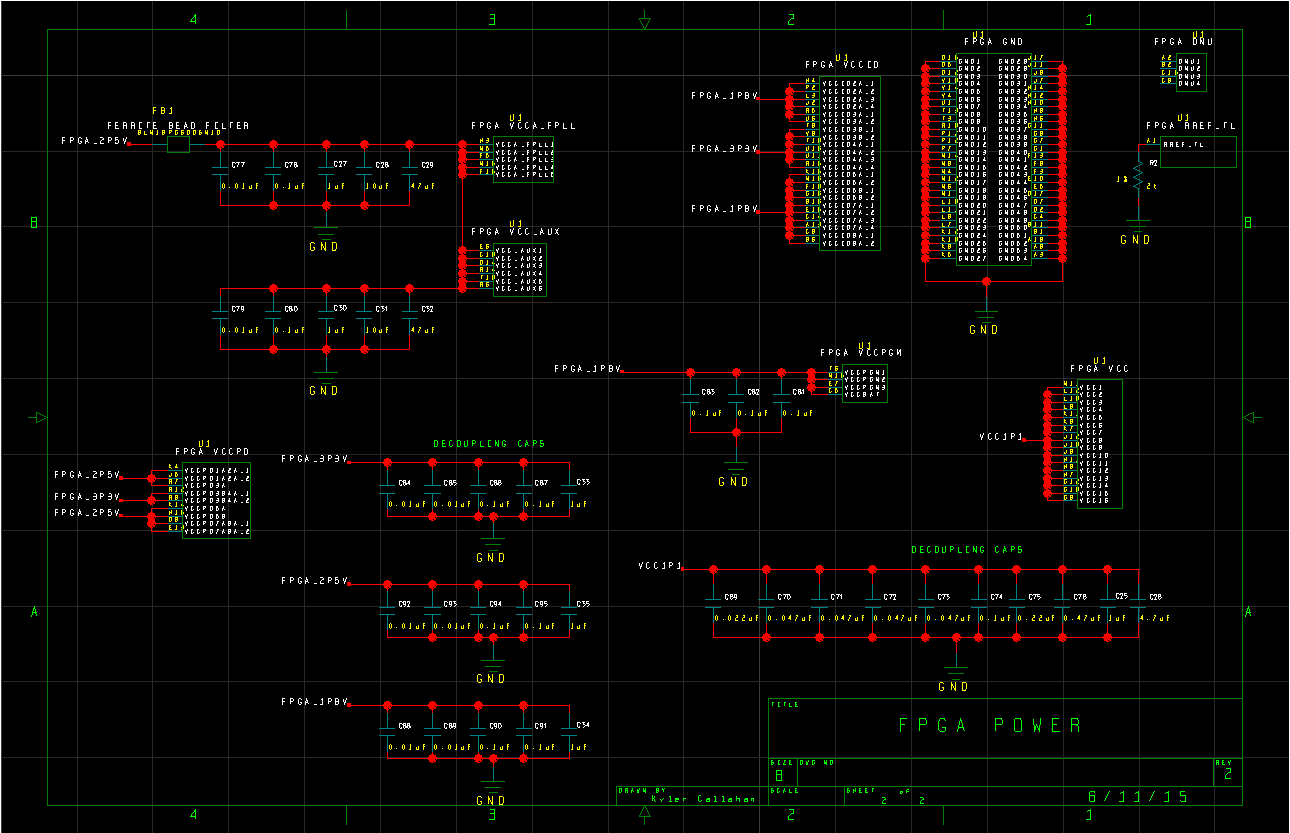


If you get an error forwarding your part common issues are forgetting to add one of the symbols in the hetero field. Check and make sure all parts are placed and that all your HETERO\_SYMNAMES matches those in the hetero field.

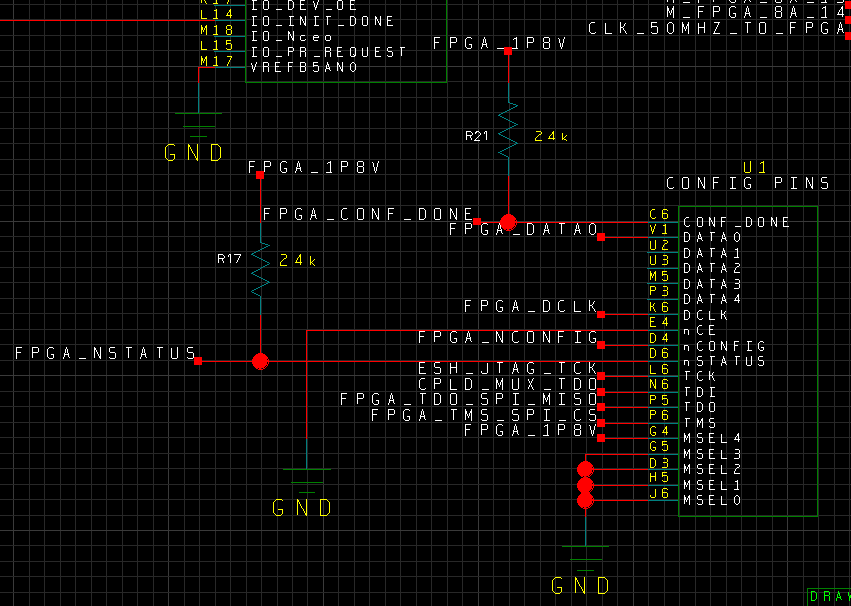
**Basic Break-Out Techniques In PADS**

This is just a quick guide on getting started routing BGA packages in PADS. Other techniques exist but will not be covered in this guide. For more information on different BGA package routing techniques please refer to the *further reading* section at the end of this tutorial. Additionally, this *PADS Specific* design flow is what I, the author, have found to be best use for quickly achieving high density routing. Other design flows exist and may change based upon your intended application. This design flow is designed to keep your schematic in sync with your layout while accommodating limited routing space, and assumes that the schematic creator and layout designer are working closely together (or the same person).

1) Your first step will be to hook up all your power circuitry according to your data sheet specification.

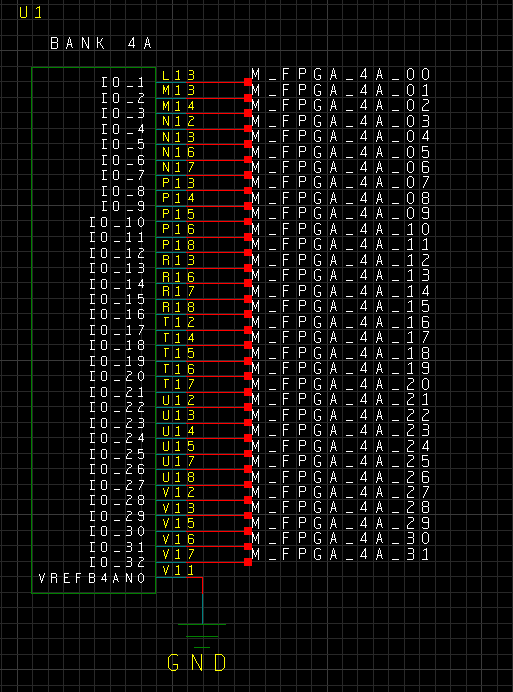


2) Your next step is to hook up all the needed config pins as recommended by the user’s manual.

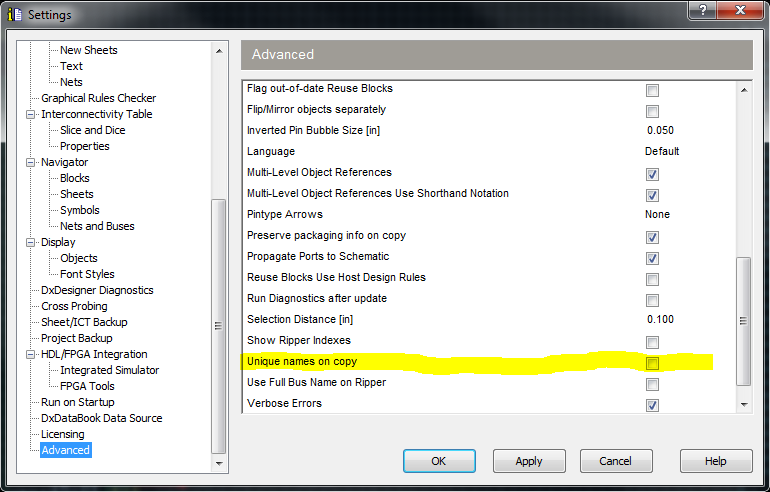


*Note: These steps are important for actual design work, but for the sake of the tutorial they may be skipped if you are trying to follow along.*

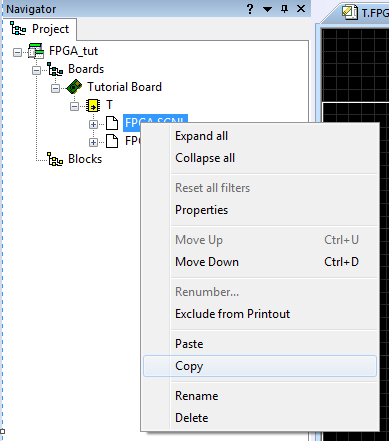
3) The next step is to hook up *all* (even if you don’t need all of them) of the I/O pins with a general signal name. Use a name that relates to the Bank that the pin belongs to. Example below. These names will be changed later but will be useful when breaking out the FPGA and hooking up general I/O signals.



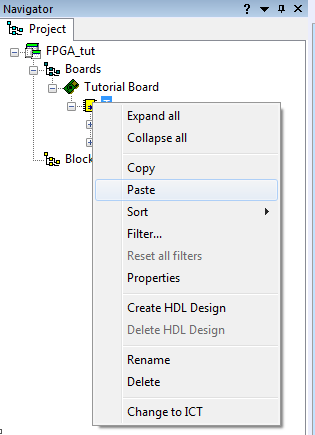
4) Make a copy of your FPGA, signal names and all. To do this make sure that “Unique names on copy” is disabled. In Setup -> Settings -> Advanced -> Unique names on copy. This setting will allow us to copy all of the names and symbols 1 to 1. The downside of this is there will be conflicts with the symbol reference designators that we will need to fix ourselves.



Make a copy of the sheet



Paste the sheet



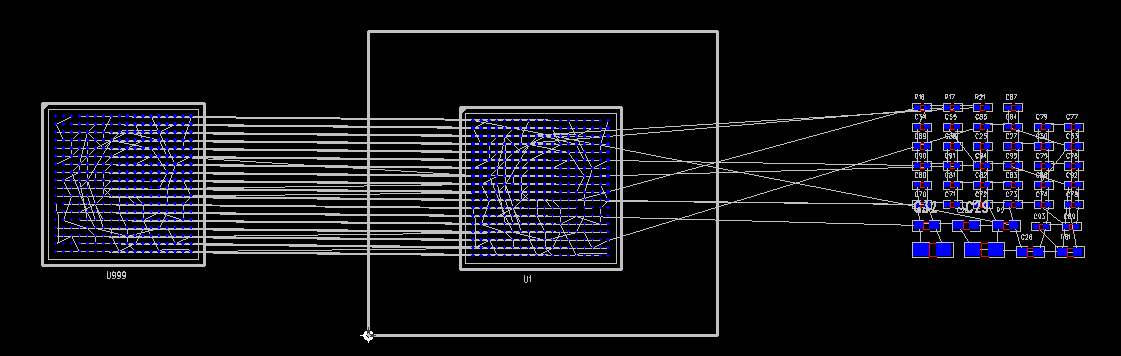
And it should give you an exact copy of that sheet

5) Change your copy FPGA reference designator to something abstract so you know it’s your copy device. For this example we will use U999. Make sure you change *all* the reference designators on your copy FPGA, otherwise it will not work properly.

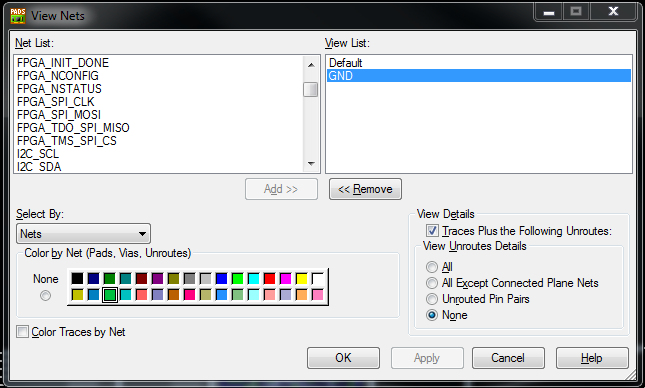
6) Delete the capacitors and resistors connected to U999 but leave the attached signals. We do not want extra unwanted components.

This copy FPGA will allow us to partially hook up our actual FPGA. PADs will not let you route from unconnected pins and it will unconnected signals that only go to a single pin. This is why we are creating a mirror part.

7) Forward your design to your PCB, place your FPGA, move U999 off screen and set your decoupling capacitors off to the side. They will be placed when we have broken out the FPGA.

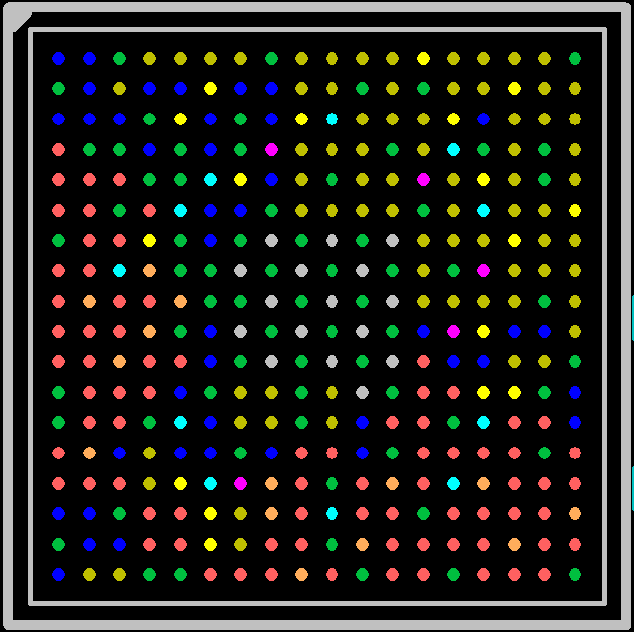


8) PADS allows for nets to be colored and the visibility of the un-routes to be modified. Press ctrl+alt+n and it should open up the *View Nets* dialog menu. Browse to GND on the left, select it and press add. Set *View Unroutes Details* to *None*. Select the color to be a shade of green and leave *Color Traces by Net* unselected.



What this is going to do is color all pins associated with the net *GND* green and make their unroutes invisible until you try and route the individual pin. This is useful for nets that have a large number of connections (such as power or ground) so you don’t have white lines all over your screen.

9) Repeat step 8 for the different voltages and the I/O pins on the FPGA. Color the I/O pins by the voltage of the bank.

 Yellow: 1.8V

Green: GND

Orange: 3.3V

Grey: 1.1V

Maroon: 2.5V

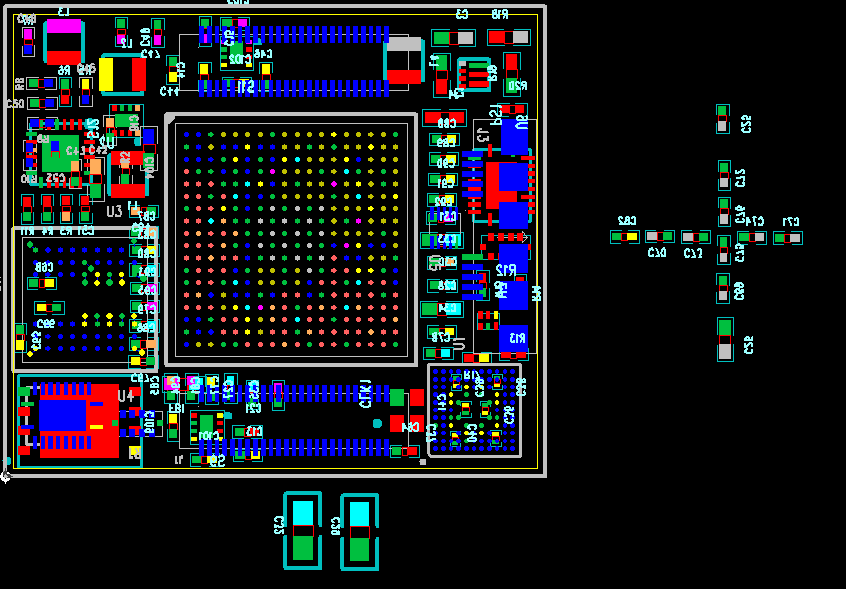
Teal: Filtered 2.5V

Gold: 1.8V I/O

Salmon: 3.3V I/O

Blue: Default/not colored

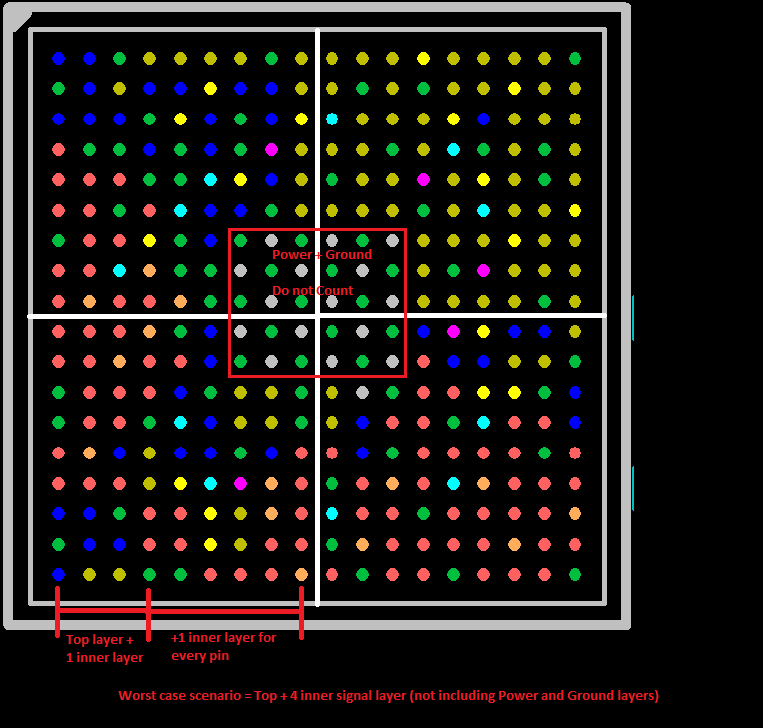
10) Finish your schematic in DxDesigner and place all your parts except for your decoupling capacitors. Don’t place any components underneath the FPGA until it is broken out.



*Note: It is important to have all your components placed before you route, but if you are following along you may skip step 10*

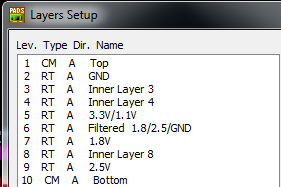
11) Next you need to determine how many layers you are going to need. Fabrication houses make boards in 2 layer sets. You can get an odd number of layers but it typically doesn’t cost anything more to have that extra layer to make your number of layers even. When determining number of *signal layers* needed a good rule of thumb is top + 1 signal layer for the first 4 pins and an additional signal layer after that.

Example:



So for this example we would need a minimum of 8 layers to accommodate this FPGA. Top + 4 signal layers + Power + Ground + Bottom. This is not accounting for multiple needed voltages for power either, so more realistically it would be 10 layers to use this package. You don’t count the bottom layer as a signal layer as we will be placing decoupling capacitors down there.

Define your layers in PADS. For this example we are using a 10 layer board. We have top and bottom, 3 inner signal layers 1 dedicated GND plane, and 4 split power and ground planes. We don’t need all of the I/O pins so we used 3 inner signal layers instead of 4.



When assigning layers try and make is so that a signal layer is next to either a power or ground layer. Do not have a signal layer in-between two other signal layers.

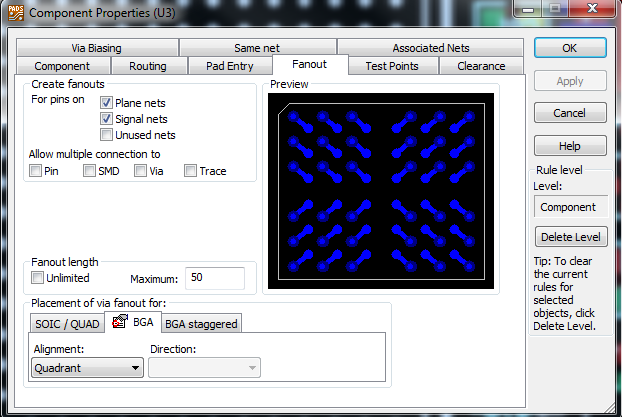
12) Now that you have your layers defined, open your design in PADS Router. We are going to break out the FPGA.

12a) Click on your component and open up the Properties window. Navigate to the *Fanout* tab.

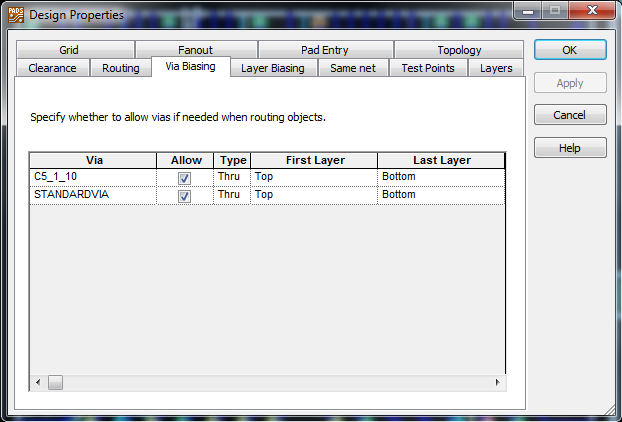
12b) Under the *Create fanouts* section select *Plane nets* and *signal nets.*

12c) Under *Fanout length* set the maximum to something small. If you have the length too long it will automatically attempt to connect the pins to their respective location. We do not want that at the moment.

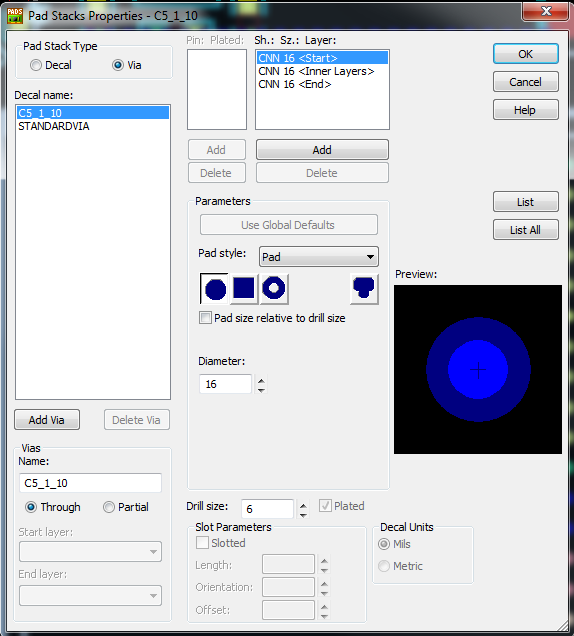
12d) In the *Placement of via fanout for:* select BGA. There are several options here but I have had the best results with *Quadrant*

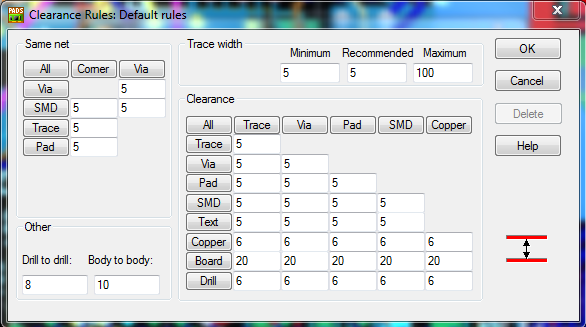


12c) Navigate to the *Via Biasing* tab and ensure that the via you wish to be placed underneath your BGA package is selected and at the top of the list.



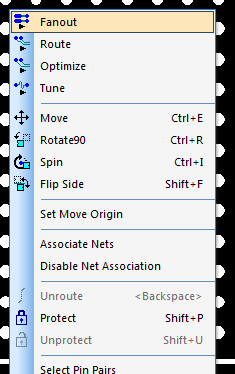
Side note about via sizes. Check with your board fabrication house what their requirements are and what will work underneath your package. For this particular example with 0.8mm spaced pins we will be using 6mil drill with a 5mil annular ring for a total diameter of 16 mils.

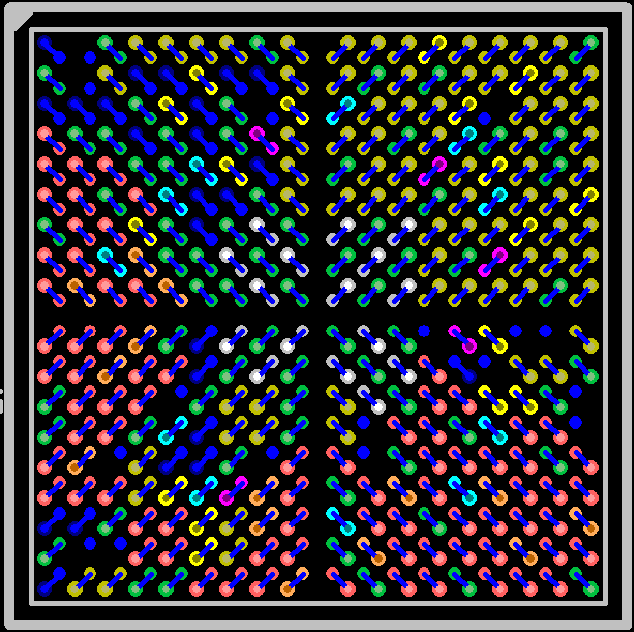




In addition to you your via sized you need to make sure that your clearances are set before you start routing. For this example we will be using 5 mil space/trace.

12d) Right click on your component and click on *fanout*. This should fan out your component

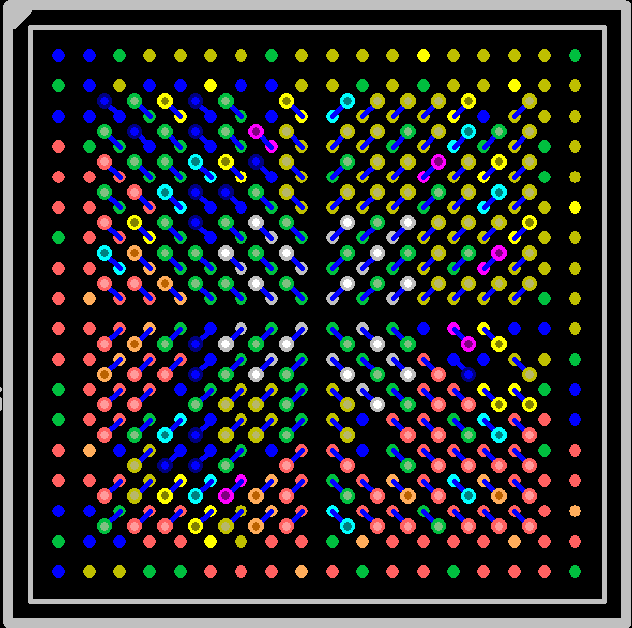




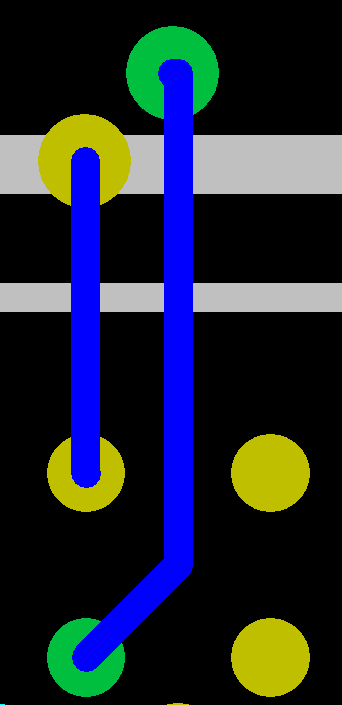
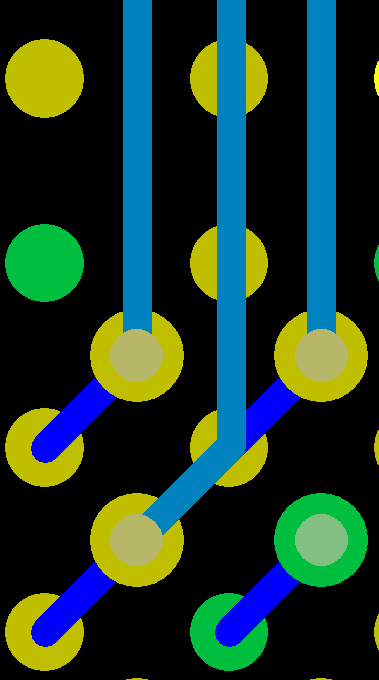
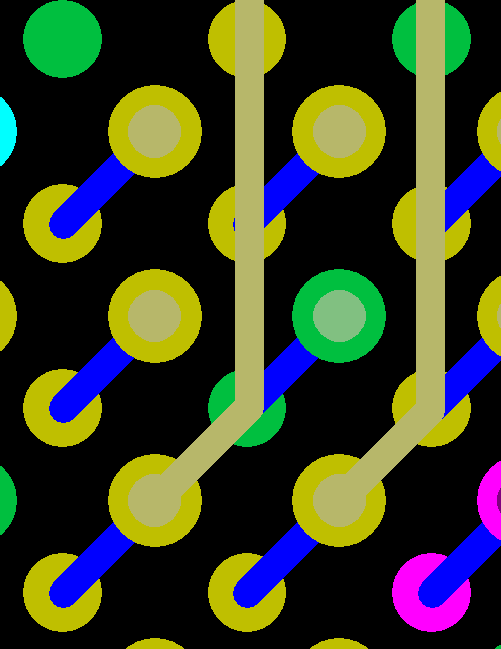
Notice how some pins are not connected. This is because we did not connect them in our schematic as they do not have an associated net.

13) At this point it is ok to place your decoupling capacitors as this is worst case scenario for your FPGA.

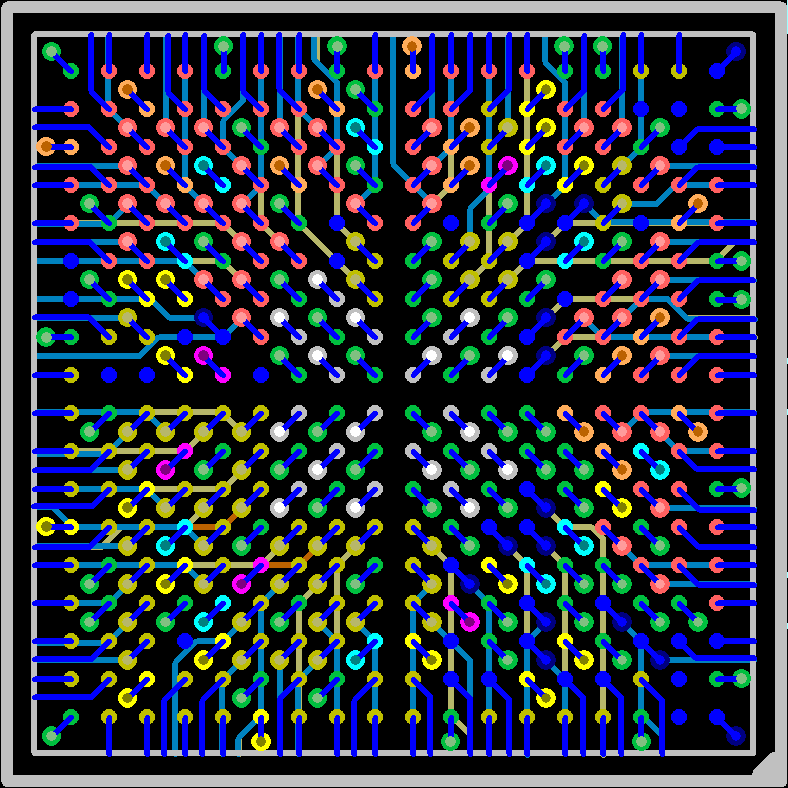
14) Delete all of the routes on the outer 2 pins. We will manually place these ourselves as their current positioning is sub optimal.



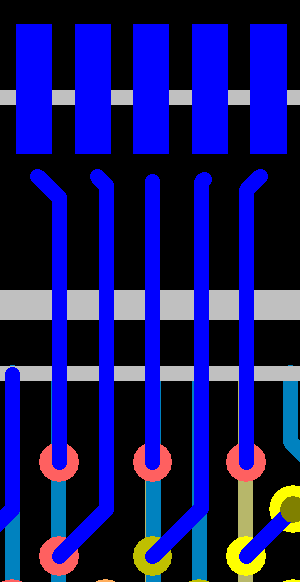
15) Route your traces to the edge of your package. This will allow us to just grab a trace that we need without worry how to break it out first. A basic break out pattern is shown below.



*First 2 rows on top layer Next 2 rows on an inner layer Subsequent rows on inner layers*

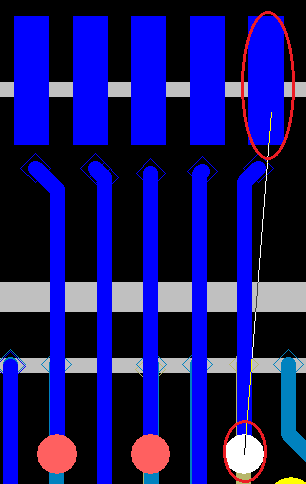


16) We can now begin routing. Since most of our FPGA is hooked up to nothing we need to hook up the pins properly. Go to one of your components that you need to hook up to the FPGA and route the *voltage appropriate* pins from the FPGA to the pins on the device. You won't be able to connect to the pins yet but this will give you an idea of the physical layout so you can hook up the pins after.



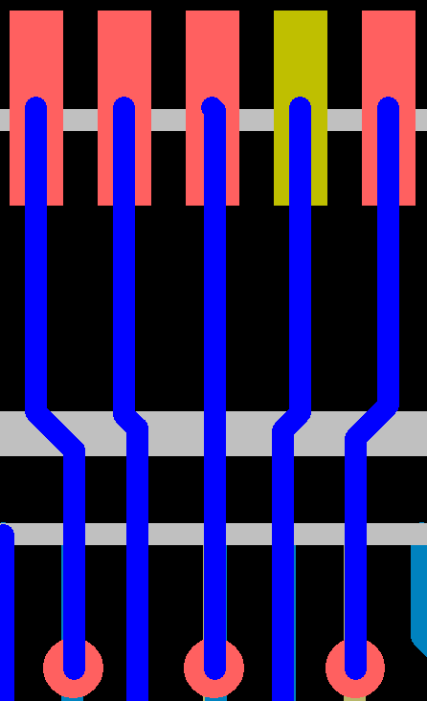
17) Open your design back up in Layout and open up the ECO toolbar and click on *add connection*



Click on the FPGA pin you want and then follow its trace to the pin you want to connect.

Ideally the non FPGA pin will not be connected anywhere else. If it is not just choose which name you think fits the pin best.

18) Open your design back up in Router and finish hooking up the pins.



19) The final thing to do is to update the pins *in the schematic* with the changes that you made in the PCB. To do this, navigate to the pins that you added connections to and make sure that they are hooked together in the schematic. You may change names if necessary, just make sure that the connections reflect the changes made in the layout. Push your updated schematic back to PADs to check that you made the proper changes.

20) Delete your copy FPGA (U999) an unused signals when you are all done routing.

It is recommended to use *Electroless Nickel Immersion Gold (ENIG)* for your PCB finish when working with BGA and fine-pitch pads. However please research the finishing process to ensure that it is the right finish for your design.

**A Brief introduction to Decoupling Capacitors**

Decoupling capacitors are important to a device for 2 reasons: They reduce unwanted frequencies in your power and can provide additional power for a brief amount of time if your device has a sudden increase in power consumption. Now ideally your device manufacturer has recommended capacitors for their device or a development board you can reference. For our example above we used a combination of both. Altera had a power calculator that would recommend capacitors based on power needs. We also had a reference design to work off of for our device that helped fill in the gaps from the power calculator.

If your device manufacturer does not have either of these things a good rule of thumb to use is a single 0.1uF X5R decoupling capacitor placed as close to every power pin as physically possible.

**Further Reading**

[Effective BGA Fanout Patterns](https://mentor1.adobeconnect.com/_a781163502/p24156661/): Presentation by Mentor Graphics. Requires Flash. Highly Recommended

[Via In Pad Guidelines](https://i.screamingcircuits.com/docs/Via_In_Pad_Guidelines.pdf): App Note by Screaming Circuits. Recommended

[Introducing Via-in-Pad Blind Via Technology to Any PCB Multilayer Fabricator](http://www.laservia.com/PDF/ipc97.pdf): Paper. Recommended

[Designing With High-Density BGA Packages for Altera Devices](https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/an/an114.pdf): App Note by Altera

[PCB Design Guidelines for 0.5mm Package-on-Package Applications Processor, Part I](http://www.ti.com/lit/an/sprabb3/sprabb3.pdf): App Note by TI

[PCB Design Guidelines for 0.4mm Package-on-Package (PoP) Packages, Part I](http://www.ti.com.cn/cn/lit/an/spraav1b/spraav1b.pdf): App Note by TI

[Choosing and Using Bypass Capacitors](https://www.intersil.com/content/dam/Intersil/documents/an13/an1325.pdf): App Note by Intersil

[PCB Surface Finishes](http://www.smta.org/chapters/files/umw_viasystems_surface_finishes.pdf): Presentation by Viasystems