Altium Schematic Tutorial

Week 1

Table of Contents

Introduction

This first tutorial of two will run you through the basics of getting Altium setup for everyday use and starting a schematic for ENCE461.

1. Schematic Tutorial – Setting up Altium and simple schematic diagram entry.
2. PCB Tutorial – Creating and laying out a PCB from the schematic.

These will walk you through the design of a SAM4S based microcontroller design, which will form the basis of your Wacky Racer circuit boards. The second tutorial begins later in the course.

Useful Schematic Editor Hotkeys

Function	**Hotkey**	**Context**
Pan the view	Right click drag	
Zoom the view	Middle click drag	
Place Menu	P	
Edit Component Properties	TAB	While placing components.
Rotate component	SPACE	While placing/dragging components
Change wire corner direction	SPACE	While placing wire
Flip along the x-axis	X	While placing/dragging
Flip along the y-axis	Y	While placing/dragging

The **"Panels"** button in the bottom right is where most information windows and properties are accessed. So if you cannot see a particular sub-window, check Panels and enable.

"Tab" key is used to access properties of highlighted/selected items. A paused symbol will appear in the middle of the screen. Once you have finished with the "Properties" window, click the pause symbol to return and continue editing your schematic or pcb.

Altium Setup – First Time Only

Libraries and Templates

Altium is a complex piece of software with many configuration options. We provide a collection of useful default settings and templates to make life easier. We also provide a library of the components we stock in the ECE component store that are available for you to use. These files can all be found in:

- `V:\AltiumDesigner`

You should copy the following directories to your P: drive into an appropriate Altium directory:

- `Libraries\ECE Components`
- `Libraries\ENCE461`
- `Templates`

NOTE: It is very important you copy all of these files to your personal drive instead of using them from the network drive. This is because when Altium opens some of these files, it "locks" them which blocks everyone else from being able to use them at the same time.

Altium Preferences

Now that you have copied over all of the libraries and templates, open up Altium.

Navigate to the ⚙ icon near top right of main window. This is the "Preferences" window where "System" & other Altium options can be set. Here we will setup Altium to use the appropriate templates. Remember to click *Apply* before leaving each configuration window.

Data Management

Under the *Data Management* → *Templates* section set *Local Templates Folder* to your templates folder e.g. `P:\Altium\Templates`. Right click the ECE A3 entry and *Set as default.*

Under the *Data Management* → *File based Libraries* section, click *Install* and add the `ECE_ComponentsAD23_x.IntLib and ENCE461AD23.IntLib` files,

e.g.

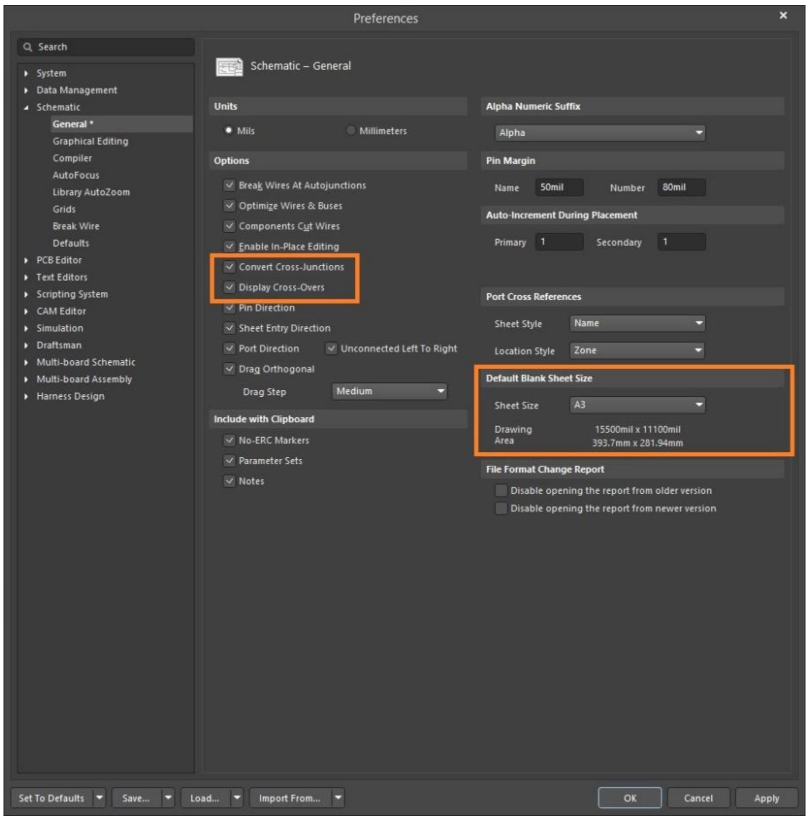
`P:\Altium\Libraries\ECE_ComponentsAD23_x.IntLib and`

`P:\Altium\Libraries\ENCE461AD23.IntLib`

Finally - from the list of installed file based libraries, disable the pre-installed libraries *Miscellaneous Connectors.IntLib* and/or *Simulation Generic Components.IntLib* by clicking each associated *Activated* button to remove the tick icon.

Schematic

Under the *Schematic → General* section, tick *Convert Cross-Junctions* and *Display Cross-overs*. Also set the default blank sheet size to A3.



Schematic – General

Units
● Mils ○ Millimeters

Options
✓ Break Wires At Autojunctions
✓ Optimize Wires & Buses
✓ Components Cut Wires
✓ Enable In-Place Editing
✓ Convert Cross-Junctions
✓ Display Cross-Overs
✓ Pin Direction
✓ Sheet Entry Direction
✓ Port Direction ✓ Unconnected Left To Right
✓ Drag Orthogonal
Drag Step: Medium

Include with Clipboard
✓ No-ERC Markers
✓ Parameter Sets
✓ Notes

Alpha Numeric Suffix
Alpha

Pin Margin
Name: 50mil Number: 80mil

Auto-Increment During Placement
Primary: 1 Secondary: 1

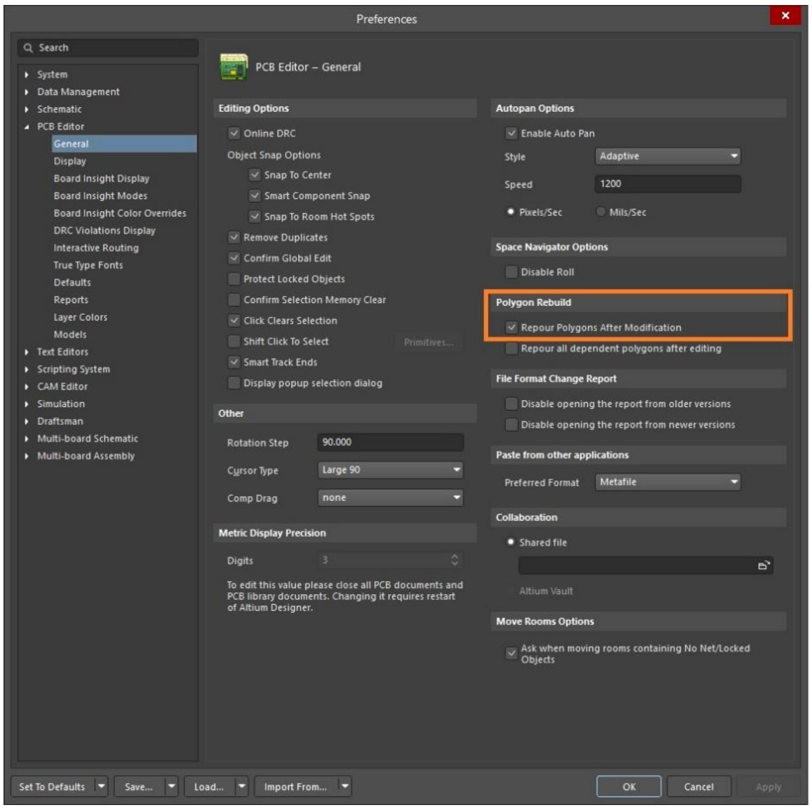
Port Cross References
Sheet Style: Name
Location Style: Zone

Default Blank Sheet Size
Sheet Size: A3
Drawing Area: 15500mil x 11100mil, 393.7mm x 281.94mm

File Format Change Report
☐ Disable opening the report from older version
☐ Disable opening the report from newer version

PCB Editor

Under *PCB Editor → General* tick the *Repour Polygons after Modification*



PCB Editor – General

Editing Options
✓ Online DRC
Object Snap Options
✓ Snap To Center
✓ Smart Component Snap
✓ Snap To Room Hot Spots
✓ Remove Duplicates
✓ Confirm Global Edit
☐ Protect Locked Objects
☐ Confirm Selection Memory Clear
✓ Click Clears Selection
☐ Shift Click To Select
✓ Smart Track Ends
☐ Display popup selection dialog

Other
Rotation Step: 90.000
Cursor Type: Large 90
Comp Drag: none

Metric Display Precision
Digits: 3
To edit this value please close all PCB documents and PCB library documents. Changing it requires restart of Altium Designer.

Autopan Options
✓ Enable Auto Pan
Style: Adaptive
Speed: 1200
● Pixels/Sec ○ Mils/Sec

Space Navigator Options
☐ Disable Roll

Polygon Rebuild
✓ Repour Polygons After Modification
☐ Repour all dependent polygons after editing

File Format Change Report
☐ Disable opening the report from older versions
☐ Disable opening the report from newer versions

Paste from other applications
Preferred Format: Metafile

Collaboration
● Shared file
○ Altium Vault

Move Rooms Options
✓ Ask when moving rooms containing No Net/Locked Objects

Press the save button at bottom of the window & store your preferences to a file. You can then reload your global preferences on any other PC running Altium.

Starting a New Project

Select *File → New → Project…*

NOTE: Have you created this project using Altium on your own computer? –> you MAY need to disable the constraints manager so that the design rules are visible. UNCHECK the "Constraint management" checkbox! (see explanation at end of this doc)

Project Type = PCB <Default>. Give it an appropriate name, and select a directory to place it in. Altium will create a new directory at the location specified, and then create the project inside that directory.

You should now have a completely empty project.

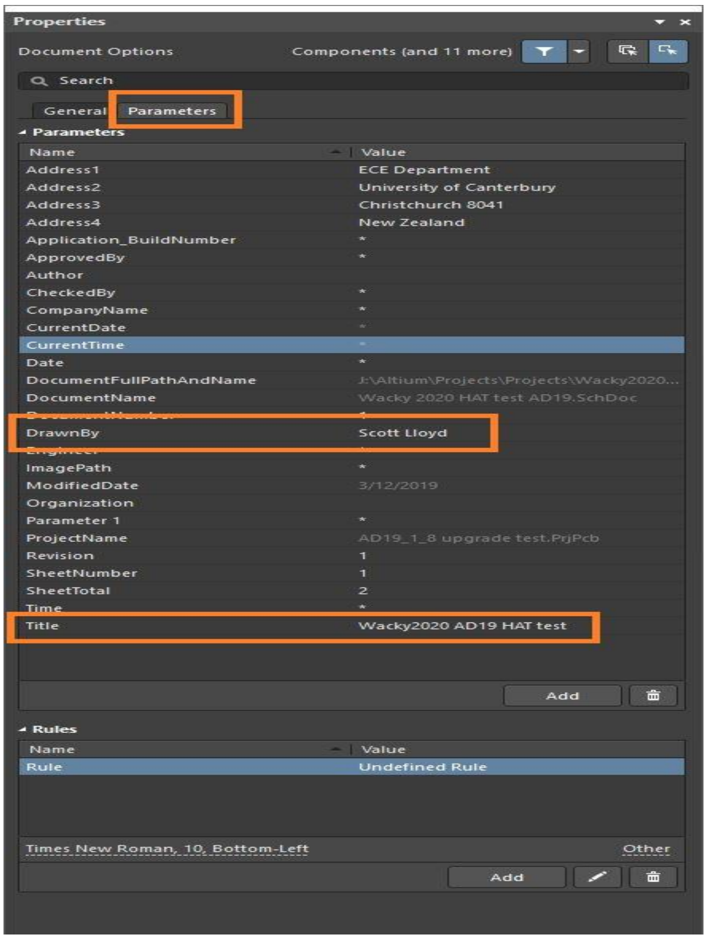
Workspace Layout

Altium has a number of very useful windows which are typically docked as side panes. Usually the *Projects* pane will be visible on the left side of the screen. If it is not, on the bottom right of Altium, select *Panels → Projects.* Also make sure the *Components* pane is also visible on the right side of the screen by selecting *Panels → Components*. You can undock these panels & float them simply by dragging them.

Adding a Sheet

Altium organises your schematic in sheets. These are individual documents which together describe the components and the logical connections between them. Add a new sheet by clicking *File → New → Schematic.* If the templates were properly set up you should now see an A3 sheet with a title block in the bottom right side. Save the new sheet and give it a name, e.g. *MCU.SchDoc.*

Click on schematic sheet, then Click *Panels → Properties → Parameters (or F11)*. Fill out the *Title* and *DrawnBy* fields. This is important to help the TAs and technicians identify the schematics.



Parameters

Name	Value
Address1	ECE Department
Address2	University of Canterbury
Address3	Christchurch 8041
Address4	New Zealand
Application_BuildNumber	*
ApprovedBy	*
Author	
CheckedBy	*
CompanyName	*
CurrentDate	*
CurrentTime	*
Date	*
DocumentFullPathAndName	J:\Altium\Projects\Projects\Wacky2020...
DocumentName	Wacky 2020 HAT test AD19.SchDoc
DocumentNumber	1
DrawnBy	Scott Lloyd
Engineer	*
ImagePath	*
ModifiedDate	3/12/2019
Organization	
Parameter 1	*
ProjectName	AD19_1_8 upgrade test.PrjPcb
Revision	1
SheetNumber	1
SheetTotal	2
Time	*
Title	Wacky2020 AD19 HAT test

Rules

Name	Value
Rule	Undefined Rule

Times New Roman, 10, Bottom-Left

Again *File → Save*. The *Project* panel should now show your named project with a titled schematic in its source documents.

Multiple Sheets in a single Project

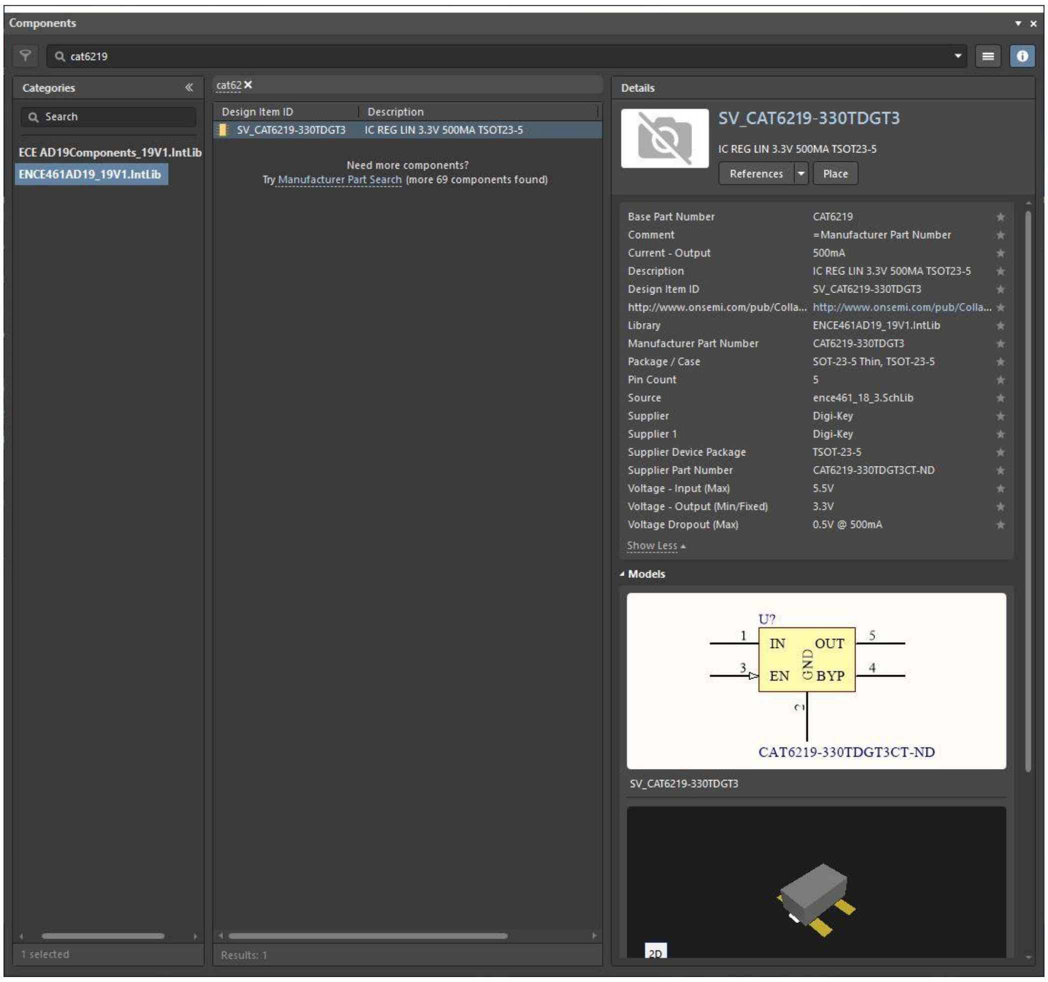
It is possible to add multiple schematic sheets to one project. You may choose to do this to make the schematics easier to read. An example would be creating a sheet for all the voltage regulators + power circuits AND another sheet that contains the microprocessor and peripherals. This is optional & two sheets is suitable for this project. Under *Panels->Projects* the single or multi *.SchDoc files will reside as source documents within your project tree.

Placing Components

Altium schematics are built up from components, and the wires that connect them. To place a component do the following steps:

1. In the *Components* panel, set the first drop down box to the ENCE461AD23_x library. You can scroll both the upper and lower section of this panel.
2. You can also undock/widen the *Components* panel to see the *Details* section & click the "i" icon box in the top right corner of the components panel. Click *Models* drop down to display the schematic symbol & PCB footprint.
3. In the upper text box (filter), start typing in "CAT6219". This will filter the components shown until only the 3.3V regulator is visible.
4. Double click the entry (or press the *Place* button in the *Components / Details* panel.
5. Click on the sheet to place the component.

At any point while placing an object of any type, you can push 'TAB' to edit its properties.



Design Item ID	Description
SV_CAT6219-330TDGT3	IC REG LIN 3.3V 500MA TSOT23-5

Need more components?
Try Manufacturer Part Search (more 69 components found)

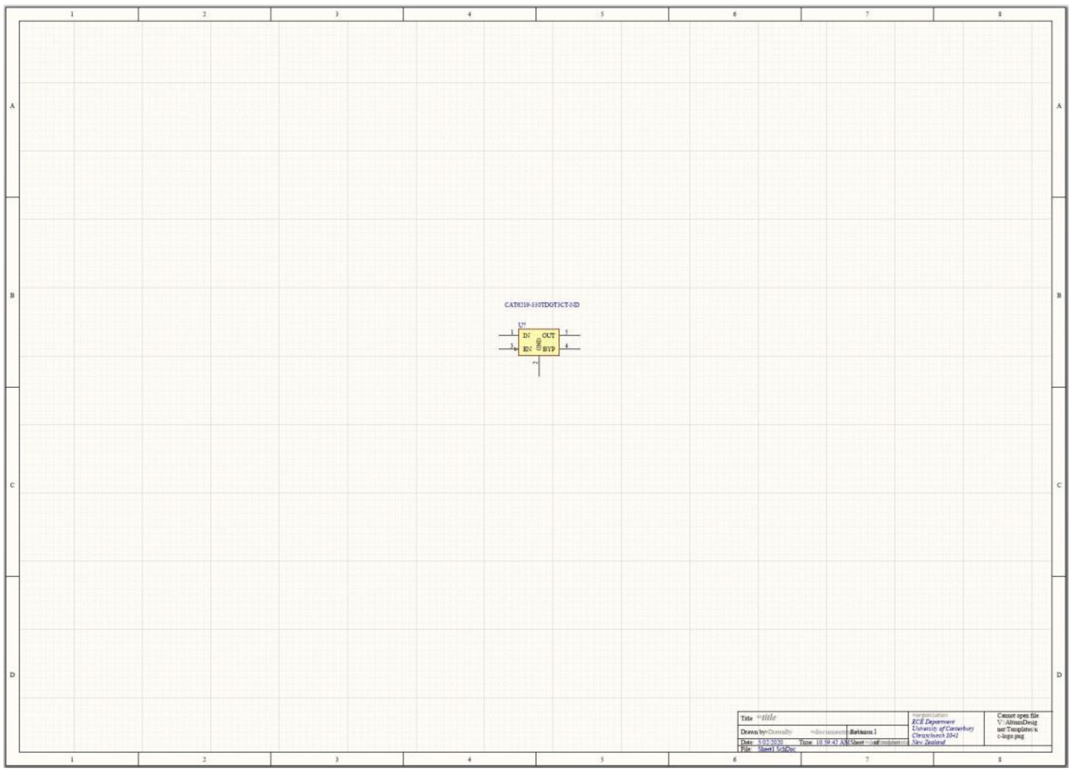
Results: 1

SV_CAT6219-330TDGT3
IC REG LIN 3.3V 500MA TSOT23-5

Base Part Number	CAT6219
Comment	=Manufacturer Part Number
Current - Output	500mA
Description	IC REG LIN 3.3V 500MA TSOT23-5
Design Item ID	SV_CAT6219-330TDGT3
http://www.onsemi.com/pub/Colla...	http://www.onsemi.com/pub/Colla...
Library	ENCE461AD19_19V1.IntLib
Manufacturer Part Number	CAT6219-330TDGT3
Package / Case	SOT-23-5 Thin, TSOT-23-5
Pin Count	5
Source	ence461_18_3.SchLib
Supplier	Digi-Key
Supplier 1	Digi-Key
Supplier Device Package	TSOT-23-5
Supplier Part Number	CAT6219-330TDGT3CT-ND
Voltage - Input (Max)	5.5V
Voltage - Output (Min/Fixed)	3.3V
Voltage Dropout (Max)	0.5V @ 500mA

Show Less

You should now have something that looks like:



Searching for Components

In the previous step, we used the filter string "CAT6219". Altium will display any component within the selected library that matches the string in the box. There is no need for an asterisk "*" to be matched as a wildcard.

If you cannot find a component, select & search the other installed library by clicking & changing the active library name at the top of the *Components* panel.

If you undock the *Components* panel, you can widen the floating panel & click the "i" icon in the top right corner to display component details to the right of the Library list. Otherwise just scroll down when you select components from the library to see additional details when the panel is docked.

Components for Design

For our design, we will need the following components (we will add the smaller passives later):

1. A SAM4S Microcontroller (ATSAM4S8), this component has two symbols and you will need both. Place Part1 and then Part2 symbol will appear automatically (or you can select Part A or B from the Components panel).
2. A USB port (Mini or Micro, pick whatever you have a cable for)
3. A 3.3V Voltage Regulator (CAT6219)
4. Two push switches (EVQPNF05M)
5. Two LEDs (0603 size, colour of your choice)
6. A programming connector (10 pin IDC connector "JA_HDR_IDC_5X2_VERT")

Note that when choosing your component, you need to be aware of what footprint it has. There are four kinds of LEDs in the ECE library. The 150060's are 0603[1] (imperial 60x30 mil[2]) sized surface

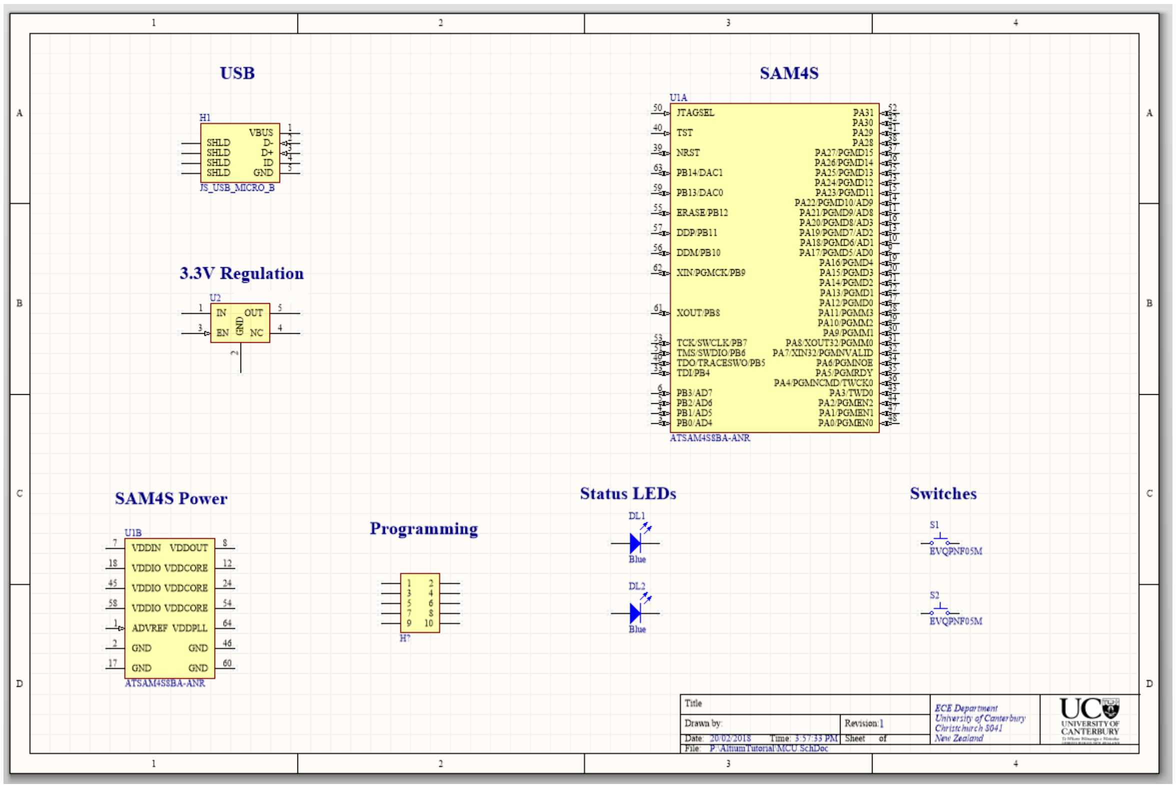
[1] https://en.wikipedia.org/wiki/Surface-mount_technology#Packages

[2] 1 mil = one thousandth of an inch. Most components in the library are given in imperial sizes but a few are metric.

mount (SMD) LEDs. The 150141's are 3528 (metric 3.5x2.8mm) SMD LEDs which are quite large. The remaining two are through-hole components which will not be used for this design.

When placing the components, you can push 'SPACE' to rotate them 90 degrees or 'x' and 'y' to flip them along that axis.

When laying the components out on the schematic, it is advisable to keep distinct blocks separate to make it easier to decipher. You can also add text or graphical lines from the *Place* menu (shortcut 'P'). Don't forget to push 'TAB' to set the text as you are placing it. You should now have something like this:

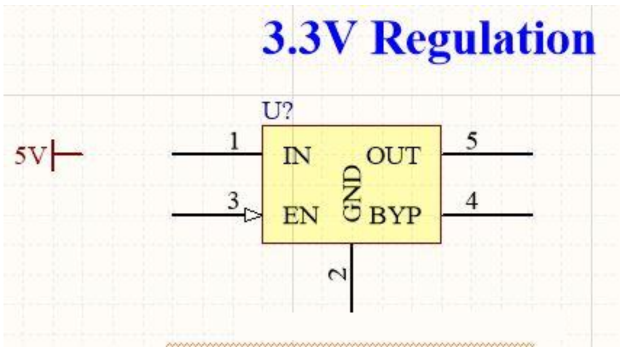


3.3V Regulator

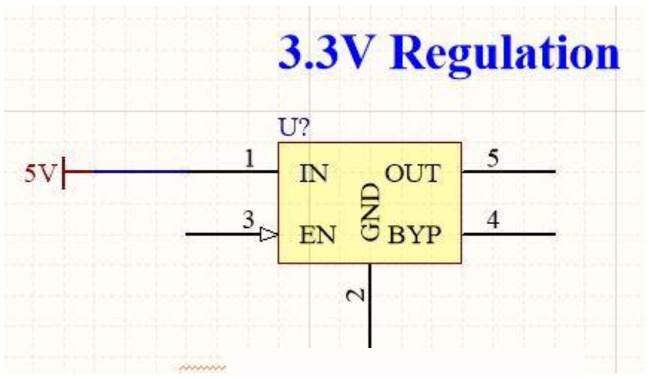
We will now add some of the connections for the 3.3V regulator:

- First ensure the *Properties* panel is active / visible.

1. Place ('P') → Power Port ('O')
2. Press 'TAB'
3. Set the style to "Bar"
4. Change the Name to "5V"
5. Place it next to the regulators IN pin.



Now we will connect it to the IN pin by using Place ‘P’ → Wire ‘W’.



These pins will now be connected to the same “net”. They should be connected with copper during the PCB layout and an error will be generated if they are not.

The CAT6219 is a fixed output, 500 mA, low-dropout regulator manufactured by ON Semiconductor. Its datasheet[3] provides instructions about how the rest of the connections should be made and about what kind of decoupling capacitors[4] it requires.

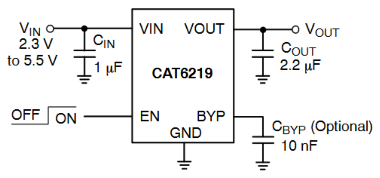


Figure 1. Typical Application Circuit

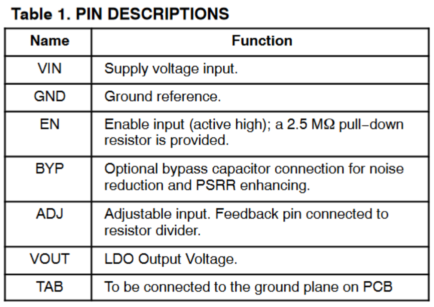


Table 1. PIN DESCRIPTIONS

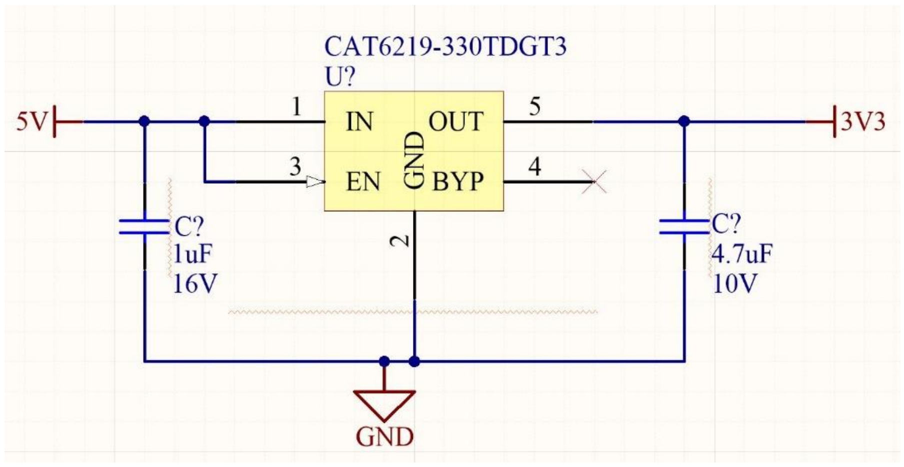
Name	Function
VIN	Supply voltage input.
GND	Ground reference.
EN	Enable input (active high); a 2.5 MΩ pull–down resistor is provided.
BYP	Optional bypass capacitor connection for noise reduction and PSRR enhancing.
ADJ	Adjustable input. Feedback pin connected to resistor divider.
VOUT	LDO Output Voltage.
TAB	To be connected to the ground plane on PCB

Pin Functions table from the CAT6219 datasheet, page 3.

The enable (EN) pin can be connected directly to the 5 V bus. The 1µF capacitor can be found by searching for “1uf” in the ECE library. It is important to make sure the 1uF & 4.7uF capacitors have an appropriate voltage rating, usually twice what is expected is a safe bet.

[3] https://www.onsemi.com/pub/Collateral/CAT6219-D.PDF
[4] https://en.wikipedia.org/wiki/Decoupling_capacitor (you will learn about these later in your lectures)

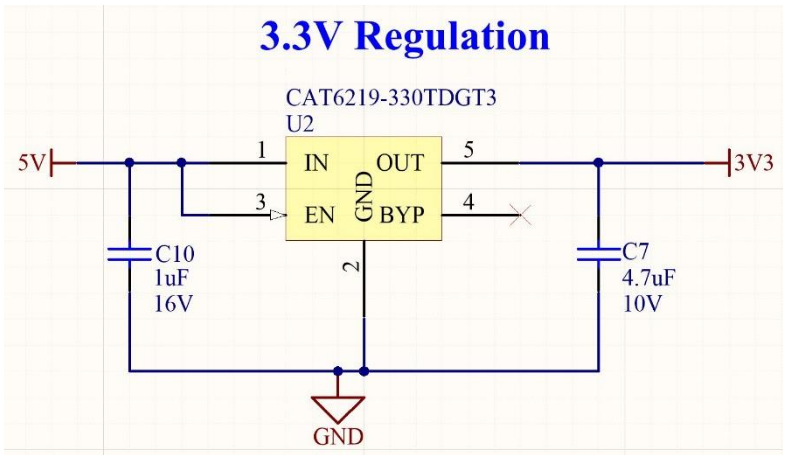
Make sure you add the power ports for the 3.3V bus and ground. A common convention in schematics is to use the names “3V3” and “GND” for these. You can also drag the comment out of the way of the GND pin. You should now have something that looks like:



NOTE: You select the symbols for the Power & Gnd ports. You MUST be consistent throughout your design. Above is a commonly used GND symbol.

Annotate

The red squiggles that can exist near placed components or objects, indicate that Altium has detected an issue. In this case it is simply that we have not given each component a unique designator. Do this now by using *Tools →Annotation→Annotate Schematics Quietly…* or *Force Annotate All Schematics...*



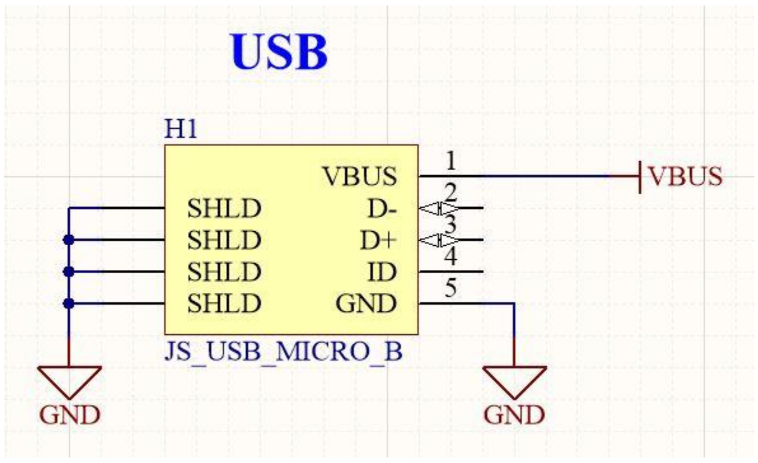
Note: Your designator numbering may be different. Altium likes to label them from left to right, top to bottom.

USB Connector

The USB connector has six connections.

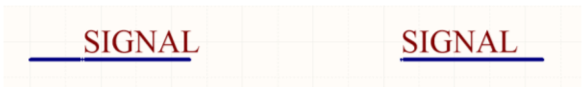
1. VBUS – A 5V power input with initial current supply of 100 mA which can be increased to 500 mA later. This must not have more than 10 µF capacitance to ground (from the USB specification).
2. D- – The negative differential signalling wire used by USB.
3. D+ – The positive differential signalling wire used by USB.
4. ID – Used for USB On-The-Go specification. Leave unconnected.
5. GND – Ground connection. Connect to system ground.
6. SHLD – Shield connection. Short to ground.

Add the connections as specified except for D+ and D-.

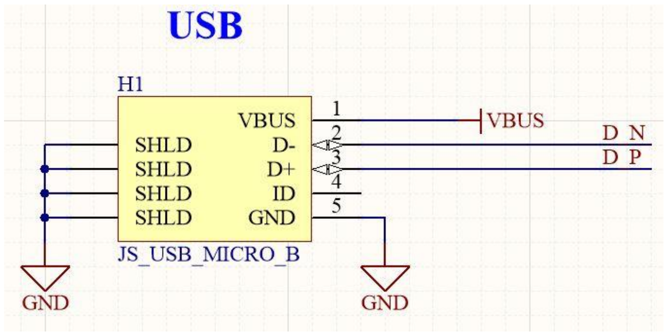


We will now connect the D+ and D- pins using net labels. A net label is used to force each wire to a certain net. This can be used to connect wires across the schematic without making a mess of things. For example, having two unconnected wires such as:

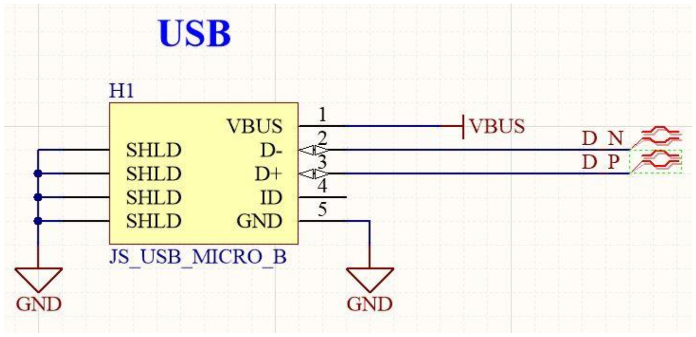
These can be set to the same net by using labels (Place 'P' → Net Label 'N'), do not forget to push 'TAB' to change the label net before you place it (see next steps):



Add a pair of Net labels (D_N and D_P) to the USB connector like so:

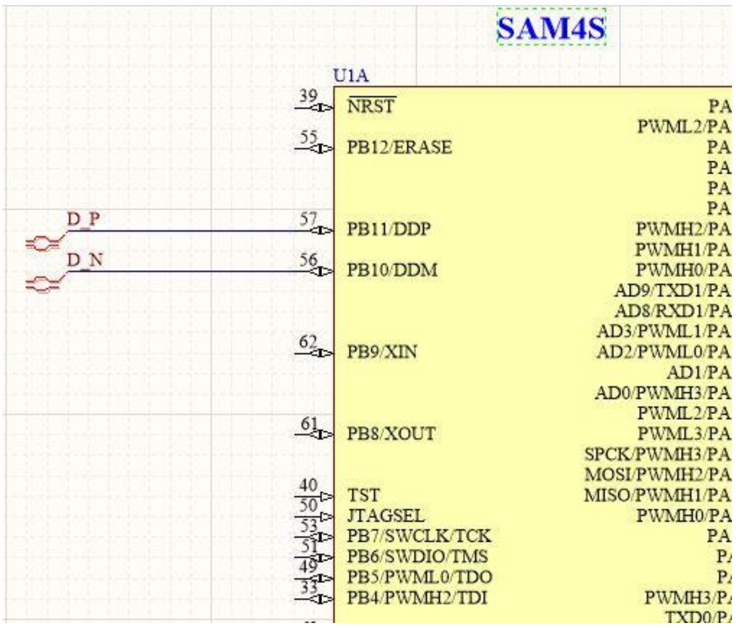


As these signals constitute a differential pair, we will tell Altium using its directive (Place → Directives → Differential Pair):

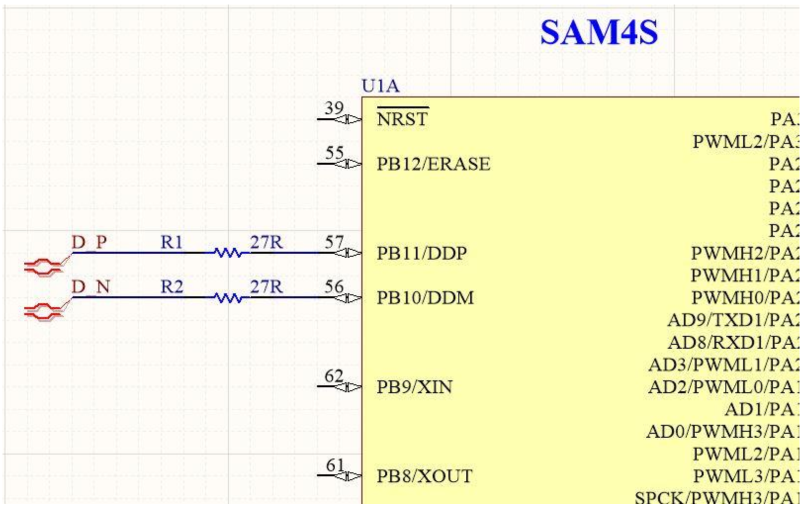


Note that it is important to use the "#NAME"_P (or) N syntax for your labels, as this is how Altium detects them as a pair. I.e. The D_N and D_P names tell Altium that it is those two nets in particular that form the differential pair.

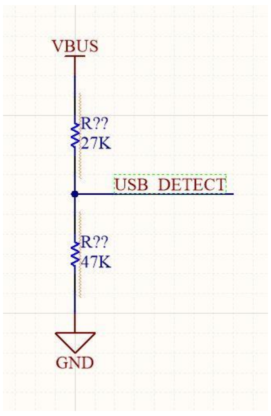
We will now connect these to the microcontroller – (you can copy & paste the net labels already placed at the USB port):



For improved signal quality, the USB specification requires series termination resistors. Some microcontrollers have these built in, but the SAM4S datasheet specifies a 27Ω on each signal. We will add two 27Ω resistors as close to the microcontroller as possible.



The final part of the USB connection is that we want to detect when it is connected so we can turn on the USB module in software. This takes the form of a resistor divider from the VBUS connection straight into a PIO pin. Add the divider using a 27 kΩ and a 47 kΩ to drop the 5 V to 3.2 V and connect it to any free PIO pin (i.e. PA5):



Keep in mind that R1 and R2 above provide a direct connection from USB5V to GND. Make sure you make the resistors large enough to limit the current to a sensible value (~5µA), otherwise you will waste all the USB power just checking it is present!

Programming Connector

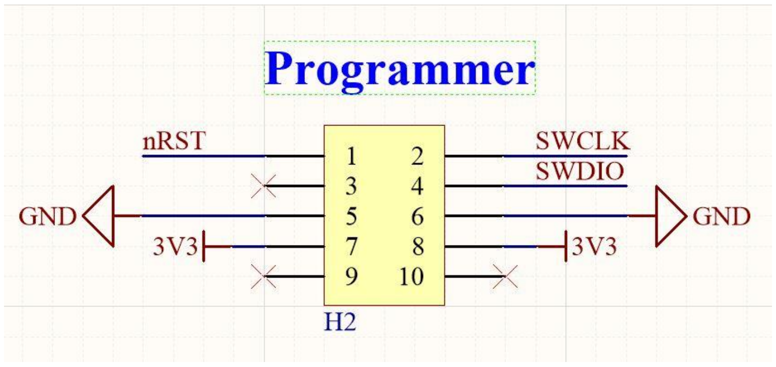
For this project. We will be using small ST-LINK V2 USB programmers. These are small USB sticks that have an exposed 2x5 pin IDC connector that contains a Serial Wire Debugging (SWD) connection which can be used to program most ARM microcontrollers. This pinout for this connector is:

Function	Pin		Function
nRST	1	2	**SWCLK**
SWIM	3	4	**SWDIO**
GND	5	6	**GND**
3V3	7	8	**3V3**
5V	9	10	5V

You need to connect the 3.3 V, GND, nRST, SWCLK, and SWDIO pins in order to program your controller as follows:

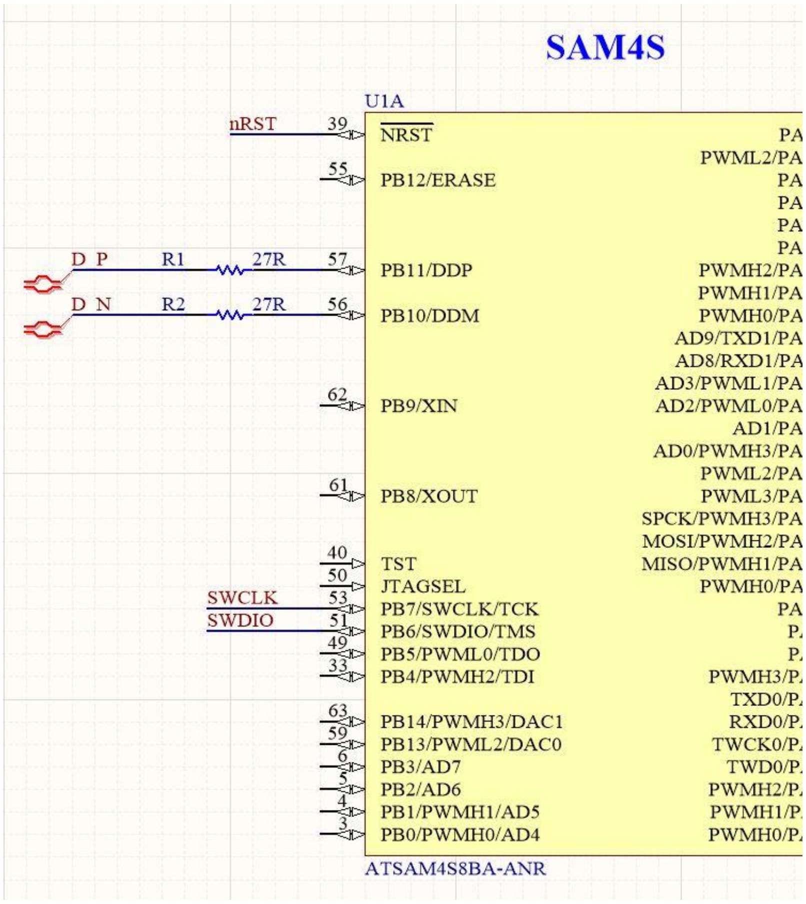
Signal	SAM4S Connection	SAM4S Pin
nRST	NRST	39
SWCLK	SWCLK/PB7	53
SWDIO	SWDIO/PB6	51

Using component JA_HDR_IDC_5x2_Vert add the following:



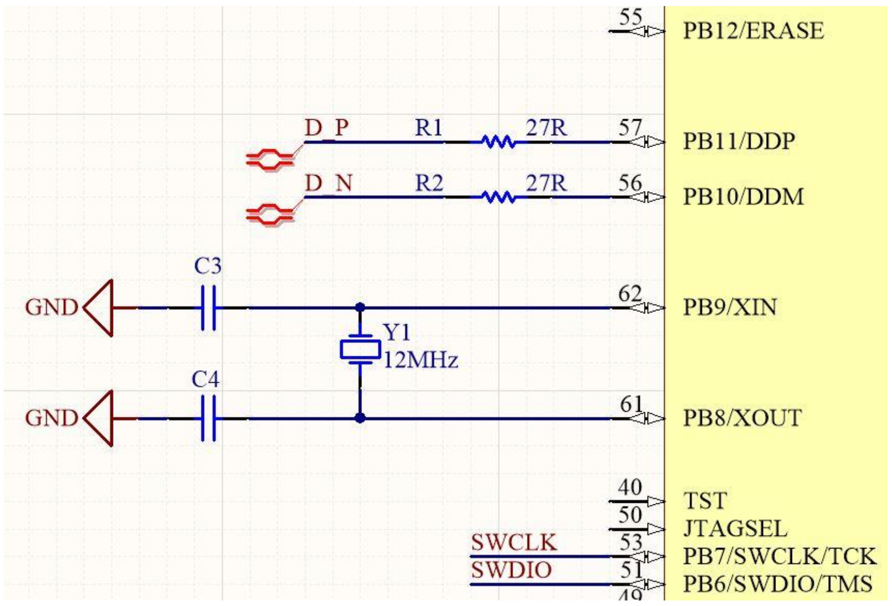
The red crosses are “Generic No ERC” which tells Altium to ignore this connection when running the electrical rules check. In other words, it instructs Altium that we intentionally left this pin unconnected and not to complain about it later. These can be found under the *Place → Directives* menu.

The connections on the SAM4S should now look like:



Crystal Oscillator

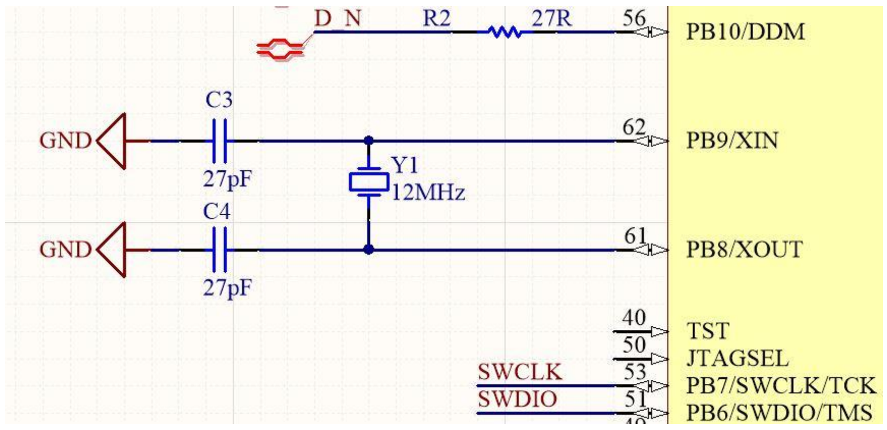
The SAM4S has a 16 MHz internal RC oscillator which is used to generate a clock signal when the micro first turns on. While this is enough for most simple applications, anything requiring accurate timing (like USB communication) will not have the necessary precision. We will now add a 12 MHz external crystal oscillator which will be set as the primary clock source during boot. This will provide a clock signal to the main clock, all of the peripheral clocks, and importantly, will be multiplied up to the 48 MHz required for the USB connection. The connections for a crystal oscillator look like:



There are two options in the ECE library for the crystal, we suggest using the smaller of the two (XT_CRYSTAL_12MHz_18pF) to make your PCB layout easier. In this circuit, C2 and C3 are used to tune the frequency of the 12 MHz crystal. This is because the crystal itself has some capacitance, C_L, and the circuit board has some stray capacitance, C_{Stray}, which will change the oscillator's frequency. We can counteract this by putting the appropriately sized load capacitors in parallel with the crystal connections. The equation to calculate this is:

$$C_2 = C_3 = 2 \times (C_L - C_{Stray})$$

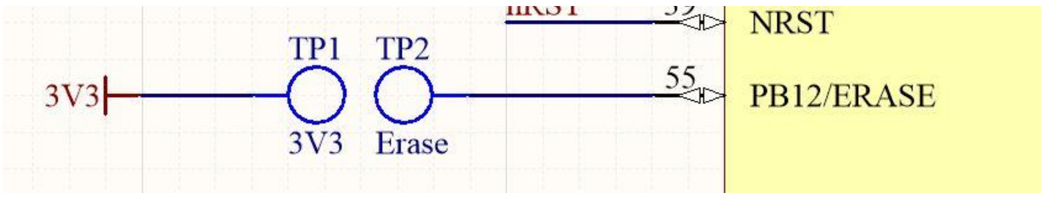
For the 12 MHz crystal we stock, C_L = 18 pF from the datasheet, and a typical stray capacitance is between 2 – 5 pF. From this we calculate a load capacitance in the range of 26 – 32 pF. We stock a 27 pF so this is a suitable choice. The clock input should now look like:



Erase Pin

Sometimes when programming, we can download code to the micro that causes it to boot improperly. Once this has been done it can become impossible to reflash the chip. This is essentially "bricking" the micro. The SAM4S has a way around this by providing an ERASE pin (PB12/Pin 55). If the ERASE pin is high when the chip is reset, the entire flash memory of the chip is wiped clean allowing you to reprogram it. To make this accessible while prototyping we can place two exposed pads on the board, one connected to the ERASE pin and one to 3.3 V. Simply bridge the two and you can program your chip again!

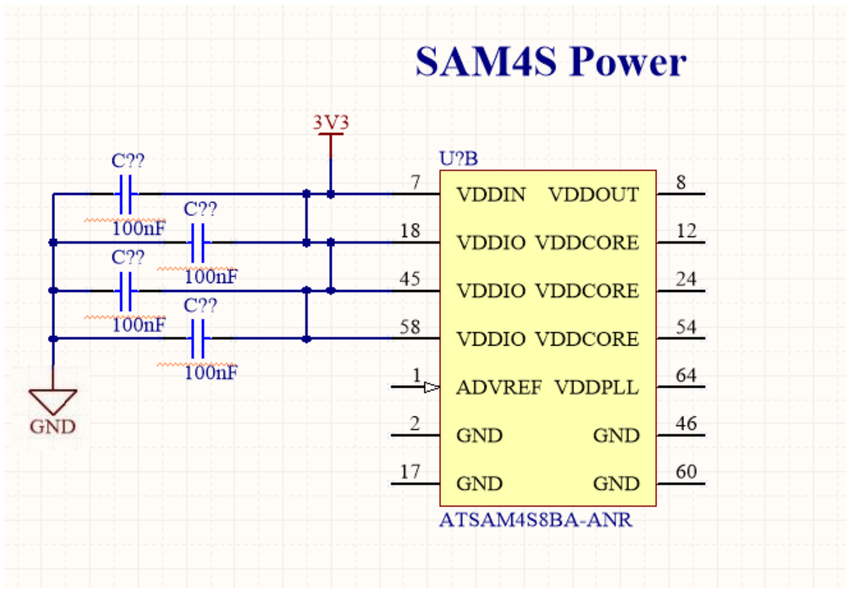
For these pads we will use test points. Place two TP_1206 components next to the ERASE pin and connect them as:



You should double click on the test points and give them an appropriate comment, “3V3” and “Erase” as above. Then click the “eye” symbol to the right of the COMMENT entry to make it visible on the schematic. These will also be made visible on the PCB silkscreen to make it easier to find them.

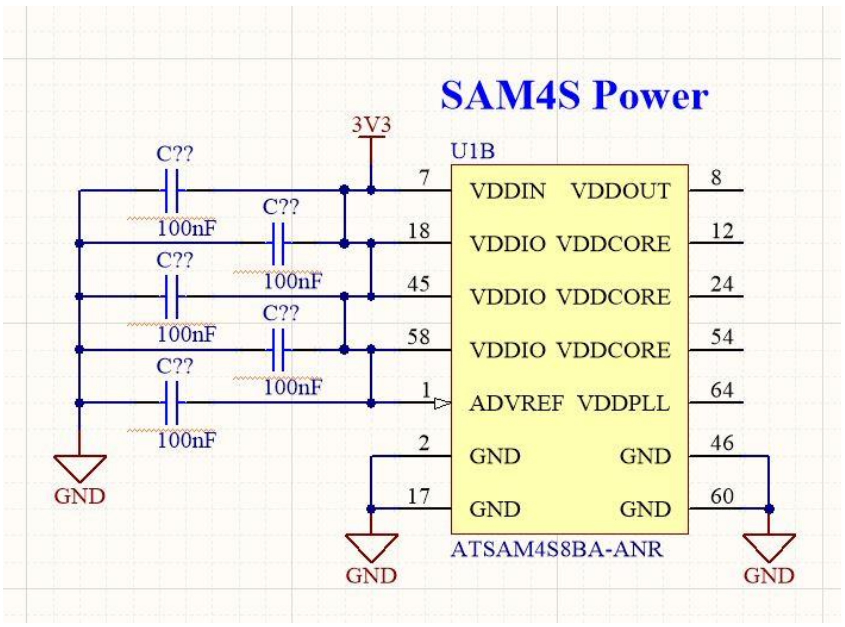
SAM4S Power

Now we shall add some power connections to the microcontroller[5]. Each of the VDDIN, and VDDIO pins should be connected to 3.3 V and have a 100nF decoupling capacitor to ground. This should look something like:

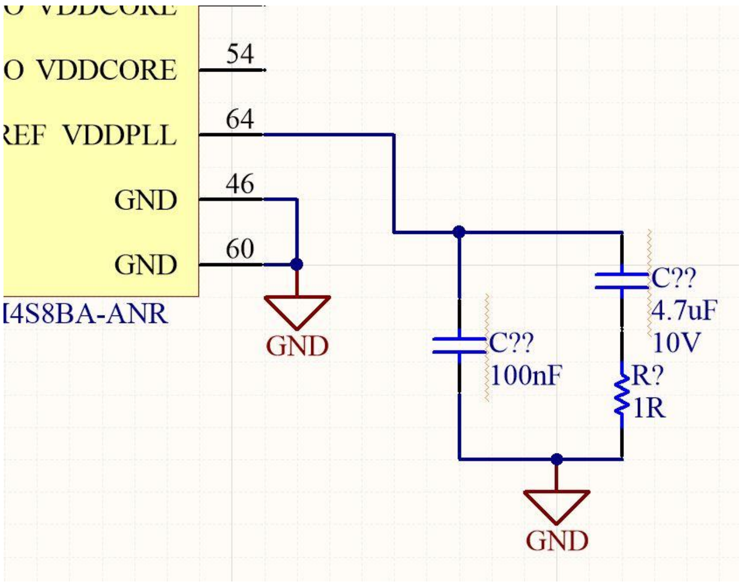


Connect the analogue reference pin (ADVREF) in the same manner. Next we will add all of the ground connections:

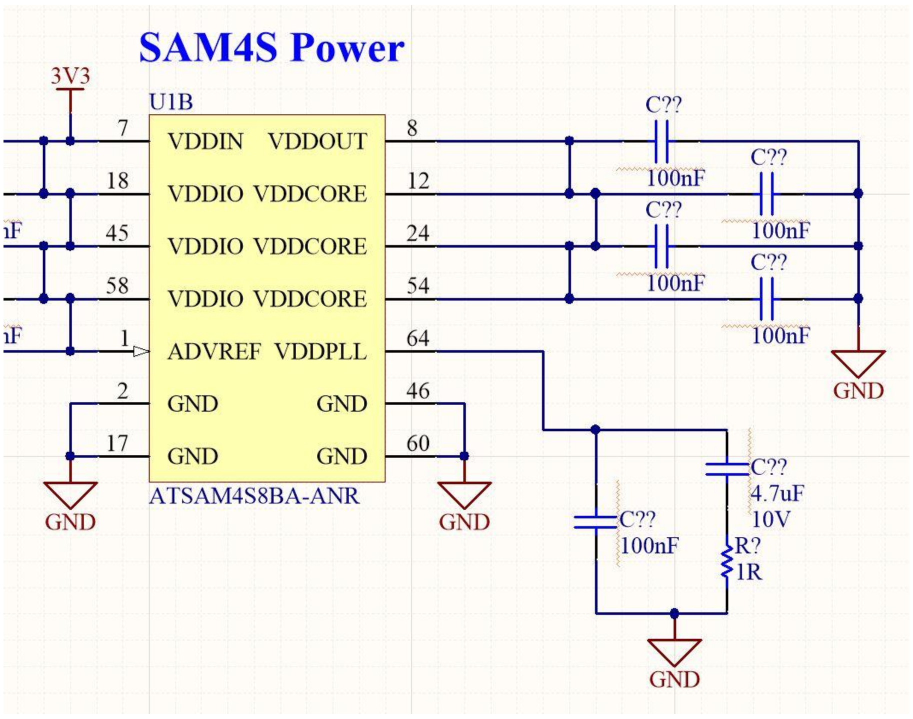
[5] http://ww1.microchip.com/downloads/en/AppNotes/Atmel-42155-SAM4S-Schematic-Checklist_ApplicationNote_AT03463.pdf



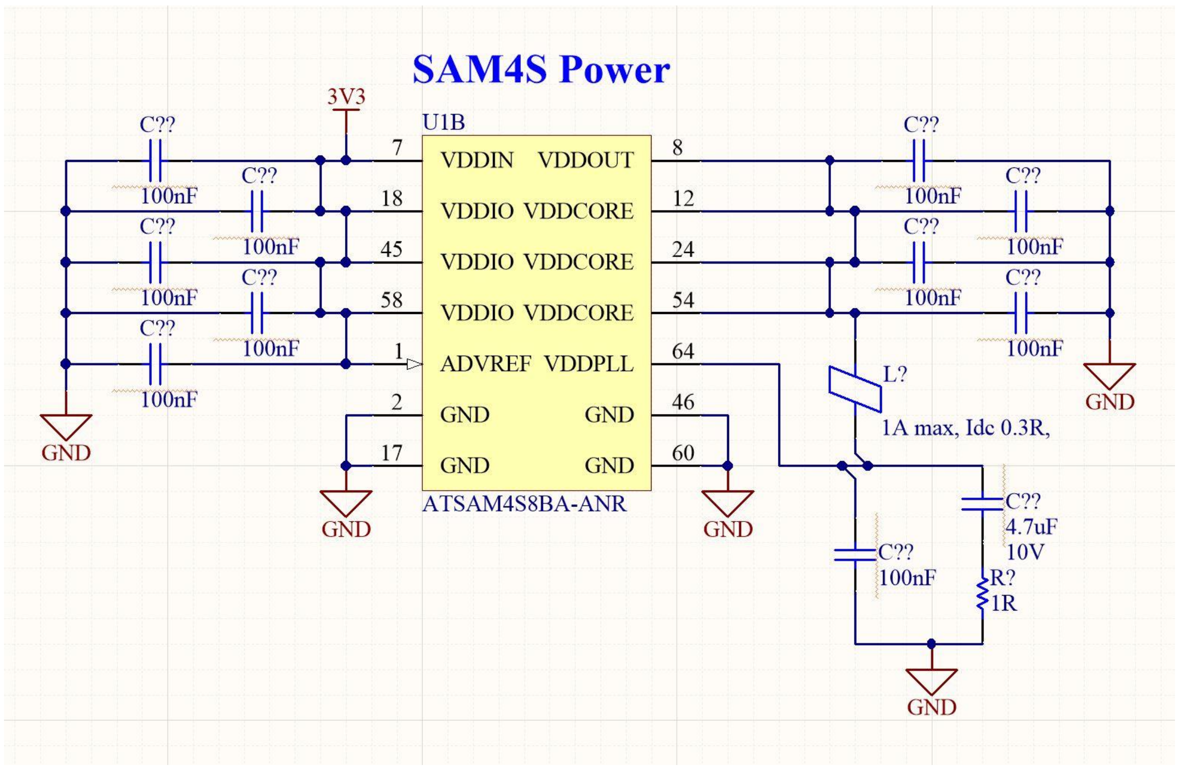
Next we will add the PLL connections. This consists of two parallel connections to ground, one is a 100 nF capacitor and the other is a 1Ω resistor in series with a 4.7 µF capacitor. Make sure to leave some room above for more decoupling capacitors:



Now we will add the VDDOUT and VDDCORE outputs. These do not need to be connected to 3V3 as VDDOUT is actually the output of a 1.2V internal regulator:



Finally we will connect the PLL and CORE nets with a ferrite bead:



Debugging LEDs and Switches

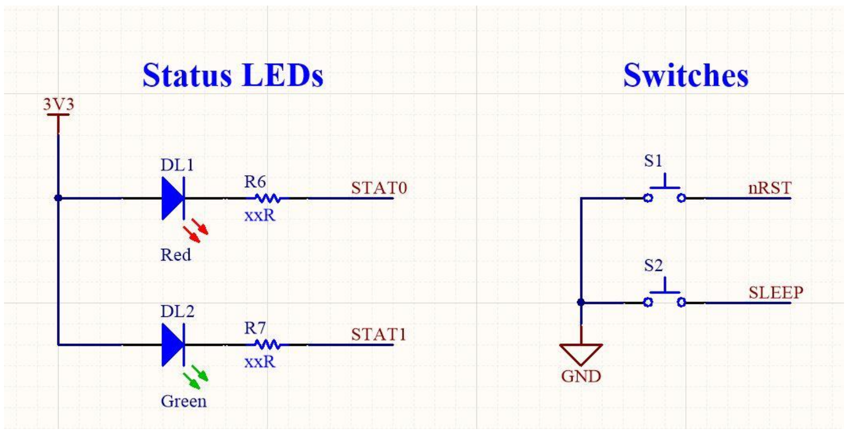
Finally let's connect the remaining LEDs and switches to the microcontroller. There are two ways to connect the LED to the PIO pins. The pin can either source current through the LED to ground, or sink current through the LED from the power supply. One thing to note is that when the micro boots up but before your code is executed, the pins will be in their default state. For the SAM4S this is typically input with the pull-up resistor enabled (some pins have other defaults, i.e. the JTAG connections). So if the LED is connected to ground, there will be a current path through the ~100 kΩ pull-up resistor and the LED. This will cause a very dim light to show. For this reason we will sink current from the LED through the PIO pin. Add the appropriate current limiting resistors for the LEDs and connect them to two of the PIO pins. An LED current of approx 500uA is enough to illuminate our stock 0603 sized light emitting diodes (the luminous intensity (mcd) for the green device is higher but at 500uA they all are at a suitable intensity)

Guide for calculating led resistors when LED current = 500uA

Stock 0603 LEDs	**V(Led) measured @ 500uA**
Red 0603:	**1.76V**
Green 0603:	**2.35V**
Blue 0603:	**2.52V**

We will now connect S1 as a reset button. The NRST pin has a permanent internal pull-up resistor so all that needs to be added is the switch that will short it to ground. Connect the other switch to a GPIO in the same manner as the GPIO pin has an optional pull-up resistor.

The connections should look like:



Finished Schematic

Now compile your design *Project->Validate PCB Project.* Altium will do the following during compilation:

- Builds a netlist for each schematic sheet.
- Creates sheet-to-sheet connectivity.
- Analyzes the complete design to check for drafting and electrical errors.

Whenever you make changes, Re-*Validate PCB Project.*

Any potential problems will be listed in the *Messages* panel. Be aware:

Ensure you have placed *no erc* directives on pin 40 & 50 of the SAM4S

Ignore warning messages regarding no driving source on nets D_N, D_P and any other nets that may have a passive component between a driving/power source & an input. This may also occur around the buck converter IC which you will soon add to convert your battery voltage to a stable 5V (note: the Buck IC is component ADP2302ARDZ-5.0 it has a fixed 5 volt output and does not require a voltage divider in the feedback loop – refer to the datasheet for details on fixed voltage versions of the Buck IC and where to connect the FB pin).

BUCK CONVERTER NOTES:

Based on the datasheet it would seem that this buck device will not perform at the fairly low battery voltage we use. But in reality it performs well beyond the minimum input voltage specified. This is a reliable device that has a very low undervoltage lockout which is one reason we recommend it for this project.

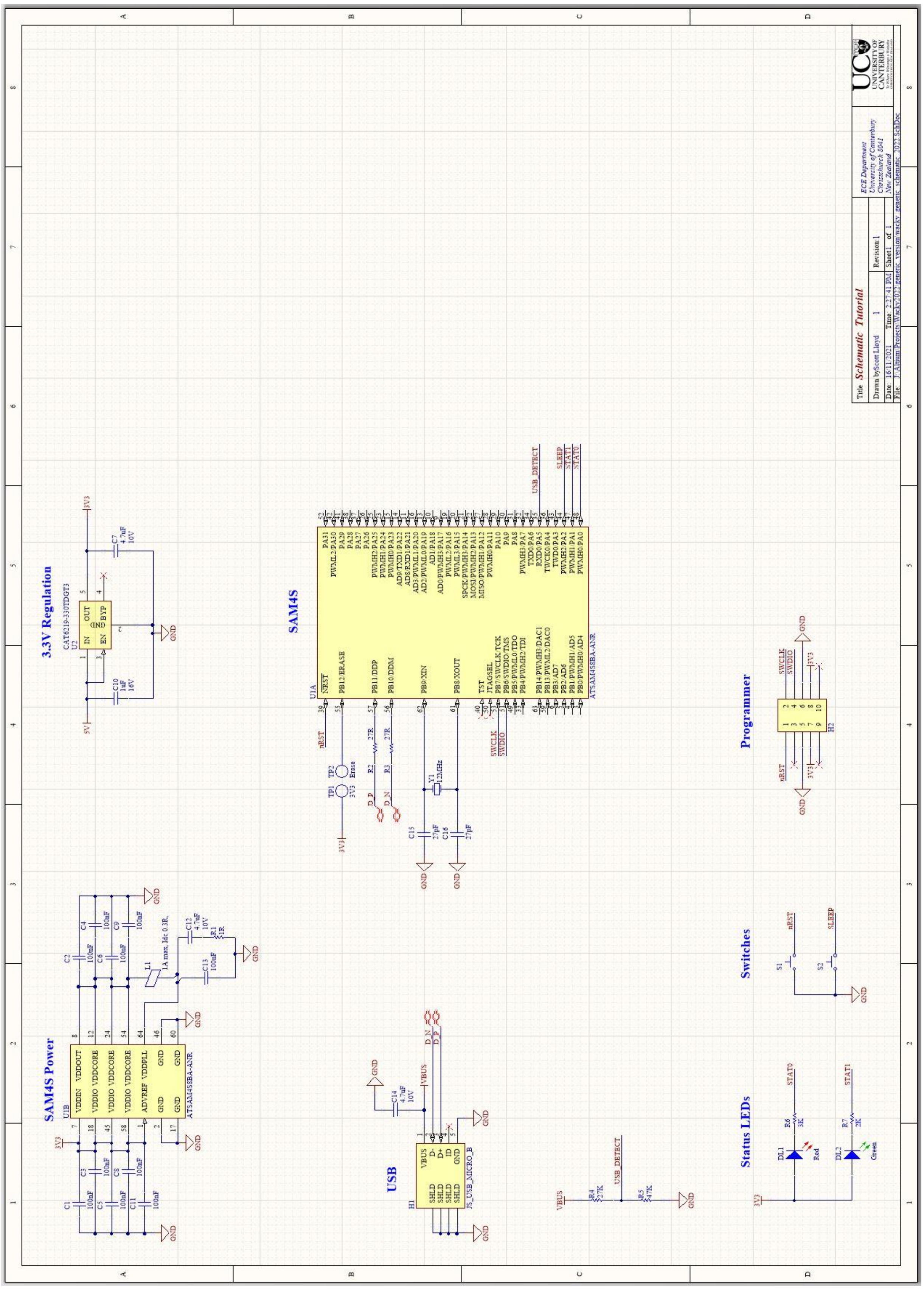
The NiMH battery packs will maintain 6V for most of their capacity – this will then drop off quickly when reaching “end of capacity”. It is up to your design to monitor this drop off at around 5V, thus detecting “battery flat” state & potentially shutting down your system to protect the battery.

For almost all of the battery capacity, the buck regulator will output 5V. Near the very end of battery life this output will reduce to about 4.3V BUT the buck regulator will continue to switch (it will not shut off until Vin is approx. 3V).
Even if you do not manage the battery voltage shutoff correctly, the ADP2302-50 will continue to run and not interfere with the operation of your design.

Note: You can optionally use a resistor divider on the enable pin to set a Vstartup=4V, refer to the datasheet -> *Application Info* -> *Programming the precision enable*. BUT because the Buck Regulator is run directly from a NiMH battery there can be power stability issues as the battery terminal voltage transitions from a loaded to unloaded state when the Buck Converter is (re)enabled and disabled. This step change in terminal voltage also shifts the resistor divider voltage on the converters enable pin – which can cause the Buck regulator to oscillate between the off and on state when the enable pin voltage shifts and causes this. It can be more reliable to simply connect the enable pin to the Buck Vin pin and keep your 5V regulator enabled at all times.

Save your work.

Hopefully you’re design should look something like:



SAM4S Power
3.3V Regulation
SAM4S
USB
Status LEDs
Switches
Programmer
ATSAM4S8BA-ANR
CAT6219-330TDGT3
JS_USB_MICRO_B
Title Schematic Tutorial
Drawn by Scott Lloyd
Date: 16/11/2021
Revision 1
ECE Department
University of Canterbury
Christchurch 8041
New Zealand
UNIVERSITY OF CANTERBURY

Next Steps

A list of things that should be added to you design:

1. Test Points! Add test points to every signal of importance. This includes: 3.3 V, 5 V, Ground (add lots of these for convenience), reset, USB, and the programming signals. Component *TP_1206* is a convenient sized test point to connect scope probes.
2. Add a battery connector, bulk capacitor at power input and appropriate 5 V regulator.
3. Adding some GPIO connections to a breakout header / strip pin connector.
4. Add a fuse between the battery and the rest of the circuit.
5. Add reverse polarity protection.
6. Add 0 Ω resistors to link power to different circuit sections. Useful when doing initial tests on circuit sections after PCB assembly. These links also can be removed when debugging to isolate sub circuits.
7. Add any required peripheral devices to your design (e.g. extra voltage regulators, radio, h-bridge, accelerometer, buzzer, custom extras)

Checklist (tick off prior to inspection)

The full design detail is covered in your lectures and the guide / instructions documents. Be sure to take note.

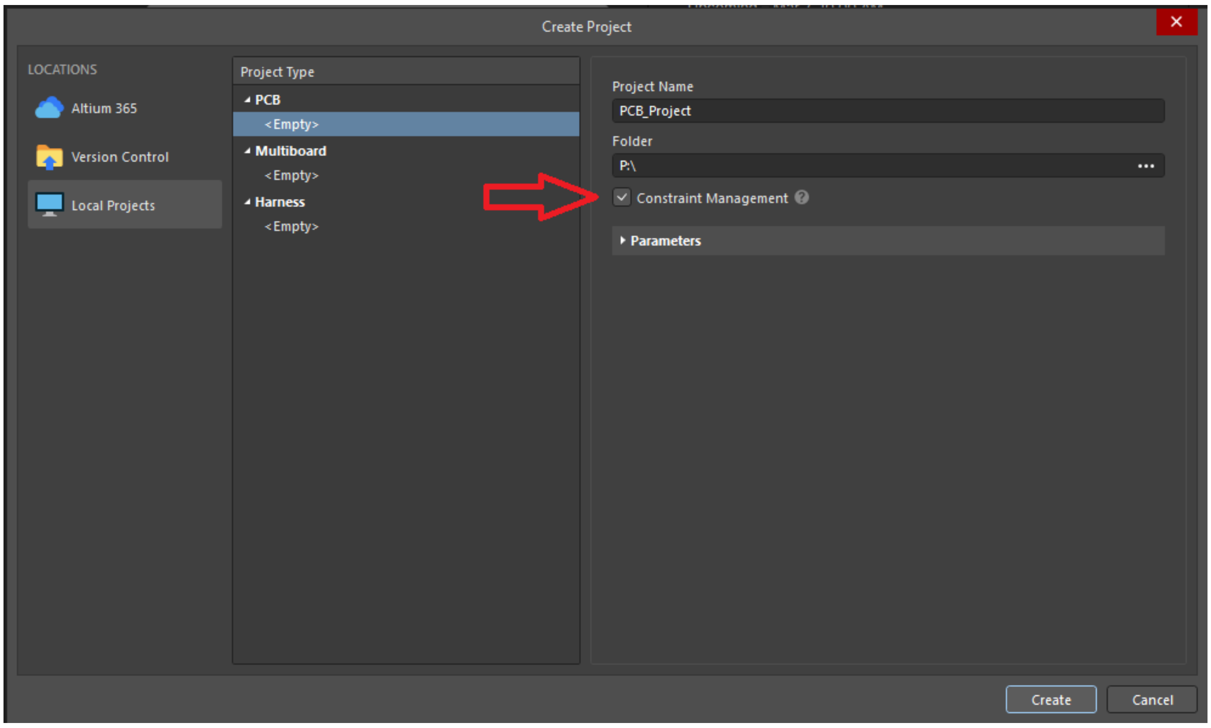
- ☐ Check your I/O connections: Have you connected SPI, I2C, PWM, A2D etc to the correct pins on the microcontroller?
- ☐ Have you used appropriate connectors for power device connections
- ☐ Check your power supplies & protection circuits
- ☐ Can you isolate circuit sections whilst hardware debugging (see step 6 above)

PTO for instructions on enabling design rules with some versions of Altium as referenced earlier in this tutorial

How to enable Design Rules menu (disable Constraint Management)

The lab PCs run stable Ver23.11.1 of Altium Designer. Since then Altium has made changes & some versions of Altium (Ver 24.x) automatically enable a new feature called *Constraints Manager* when you create a PCB project. This in turn disables the Design Rules feature which we use in the PCB tutorial.
Design Rules are re-enabled by DISABLING the Constraint Management during *PCB project* creation which is explained at the start of this tutorial (see below image - red arrow). Uncheck the tick-box before creating the project. Once this is done, the DESIGN RULES menu will be active in the PCB design window as described in the PCB tutorial.

Any groups using a different version of Altium on a student license may simply need to create their new Project ensuring the constraint management option is unchecked prior. IF you created a project and did not uncheck this – just create a new corrected project and shift your existing files across into this new project.
All should be fine if you created your project using the ElecEng Lab PCs during tutorials or if you are using Altium Designer V23.11.1 or earlier.



Project Name
PCB_Project
Folder
P:\
✓ Constraint Management