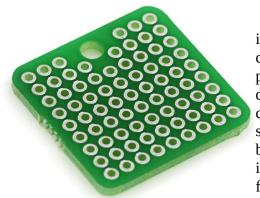
After a system has been developed, the circuit must be electrically and mechanically protected to survive a trip to the stratosphere. A breadboard may be a great tool for quick and easy prototyping, the electrical connections are however not very mechanically stable and in a high vibration environment, sensors and devices can be shaken loose. Additionally, breadboards represent a volume and mass that is unnecessary to a payload, and with a 3lb mass limit most Arizona Spacegrant Consortium associated balloon teams have to work with, teams would be wise to create a more permanent circuit without the use of breadboards for the final product for launching.



There are multiple methods for creating a strong and stable physical circuit, all with their own benefits and negatives. Most of these options follow a somewhat similar base material design, they have a bulk made of electrically insulating fiberglass and a top or top and bottom layer of copper used to create the actually circuit connections. Copper clad boards are panels of fiberglass with a solid layer of copper on one or both faces, and they are often utilized by either using a chemicals, blades, or drills the circuit pattern to make a design.



Perforated board or proto-boards are another option where instead of entire sides are covered in a copper layer, a grid of copper pads with drill holes fill the board ready for inserting parts and soldering. This means there is no copper removal or drilling steps required like copper clad boards. Perf-boards also come in various sizes and patterns like some that mimic the standard breadboard style, which can make transfer from breadboard to permanent circuit conceptually easier. On the left is a view of a small grid perf-board and below is a style following a breadboard pattern.



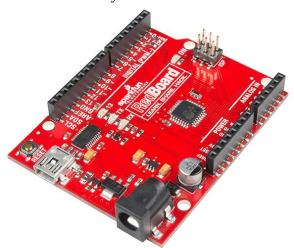
Copper clad boards and perf-boards are great because often times they can be bought in bulk and at brick and mortar stores well before a circuit design is well defined. Often times copper clad and perf-boards are available at locations like Fry's Electronics or many places where hobby electronics products are sold. Copper clad boards are great because the size and shape of all copper signals or traces can be fully controlled to accommodate parts with various hole size requirements. Take for example the DC barrel jack that is on an Arduino Uno, which uses solder pads an drill holes much larger than most holes common to perf-boards.

Perf-boards can be advantageous to copper clad boards because copper clad boards require a copper removal and drilling step that is already completed in perf-boards. Additionally all of the unused space in perf-boards have holes pre-drilled which represents mass that is cut away. With tight mass limitations like those in ballooning, this set of pre-drilled holes can represent a nice weight optimization.

The last most common option for circuit production and in my opinion the best option is printed circuit board (PCB) production using circuit design software like Eagle CAD to create a design, and having the product fabricated by companies that provide PCB manufacturing services like OSHPark.

Circuit design with Eagle can be used to define the pattern for etching copper clad boards, but outsourcing fabrication to a manufacturing service has the benefit of having a high degree of control over final design for aspects like weight optimization, and ease of final soldering assembly since no cross links between parts must be made because those would be defined in the circuit sent to and fabricated by the milling service. The downside of this is that every board can be more expensive per board than the cost of a blank copper clad or perf-board, and after fabrication the product must be shipped to the customer in the mail which represents more wait time. The benefit however, is once the product is in hand soldering takes less time and is less error prone than both copper clad and perf-board options, along with the fact that fabrication services also offer adding a solder mask, which is a layer of insulation over the top of the copper for electrically protecting the circuit. In my opinion, PCB fabrication with this method creates the highest quality end product with the least amount of hands on time necessary for creating the product compared to copper clad and perf-board options that take significantly longer times to solder and assemble the end product. However, Eagle layout and manufacturing requires more forethought to be successful, since it is harder to edit the circuit after it's made and shouldn't be ordered near the end of the development process because of the danger of shipping times taking longer than expected. In general, I'd advise to create designs in software like Eagle and fabricate using a service like OSHPark, but always have copper clad or perf-board in stock as backup in case product development takes so long that a PCB can't be ordered in time. Below are some examples of products made using this process. The rest of this section will be focused on development of a circuit design using Eagle for the purpose of fabrication by services like OSHPark.



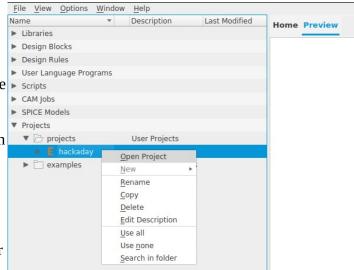


Eagle is an electronic design automation (EDA) application for schematic capture, PCB layout and routing, and more initially developed by CadSoft Computer GmbH and was acquired by Autodesk in 2016. Since this product is now under the many computer aided design (CAD) programs offered by Autodesk, it requires an Autodesk account to use. This brings an added benefit of the ability to import designs from Eagle into a 3D CAD modeling program called Fusion 360 which is also owned by Autodesk.

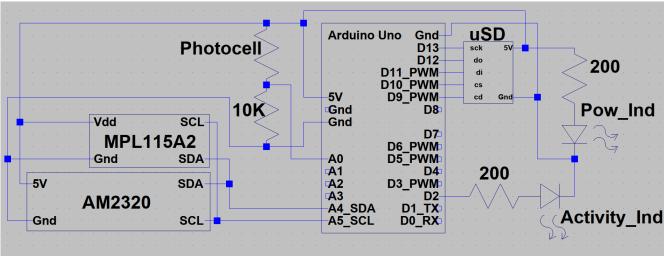
To get started, download Eagle from its product page and install or extract the downloaded archive and run the executable file. When the program opens, it will ask the user to sign in, at this point either sign in or create an account to sign in. This account can also be used for Fusion 360 for 3D CAD as well as any other Autodesk brand products. The first window that opens is called the control panel, create a project by right clicking the projects folder and create a new project like the project called "hackaday" in this screenshot. Make sure to check that the

Sign in	4
Email	
name@example.com	
NEXT	
NEW TO AUTODESK? CREATE ACCOUNT	

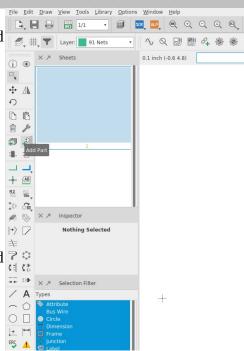
project is "opened", which can be done by right clicking the project after it's created. We will create a schematic and PCB design for our example payload as a shield that plugs into an Uno by stacking onto its headers. To start, create a project named "example payload". Recall from the integration section of this workshop that we had 5 devices connected to the Uno with some support parts like power and activity indicator leds and some resistors. Eagle has many part libraries available to for adding devices to a design, but it doesn't have every device that exists, so along with creating a schematic, we will eventually have to create our own device symbol and footprint for including



in a schematic and PCB design. Start the design by right clicking the newly created project and create a new schematic. Here's the original schematic for the payload we used in the integration section just for reference. We will need to create this same design in Eagle.



To the right is the schematic editor. To start, click the "add part" button highlighted in the screenshot to the right. A new window will open up that is a collection of all of the parts available, before choosing anything lets add more parts to our library so we have more options available to us. Click the "Open Library Manager" button to view the library manager, then click the "available" tab and it should show many optional part libraries. Click one of them and press CRTL-A on the keyboard to select all of them and click the "use" button to include all of them in your libraries. This just gives more options for premade parts to use in our design. After the library manager has finished adding all of the parts, close the window using the X in the top bar. Back in the parts view, type "arduino" into the search bar and press enter, one of the options should be a footprint in the shape of the Arduino Uno. We will use this to make a board that is meant to fit on top of the Uno. Click ok to and click anywhere in



the schematic window to place the Uno part symbol, and after placing the part press the ESC key to go back to the add more parts. Do the same for 2 3mm leds and 5 ¼Watt resistors. When searching for parts for the schematic, many options are available, and not all details are very important, for example specific color leds can be found, but this won't matter since after fabrication the designer will be the one to add the led and not the fabrication service unless assembly is requested, which costs extra. For our leds, our important parts are specifically "through hole" 3mm diameter leds. Same Attributes & goes for the resistors, we can use whatever resistors are available in the

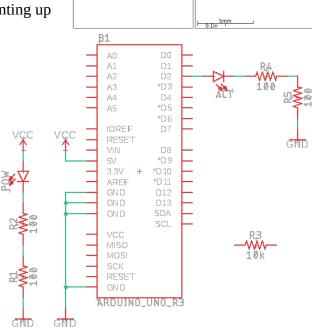
▶ SEWTAP SparkFun LilyPad Sew Taps SparkFun Aesthetics ▼ SparkFun-Aesthet Funnel IO (FIO) Logo - Silkscre ▼ SparkFun-Boards SparkFun Electron ARDUINO_MEGA_R3FUL ARDUINO UNO R3 Arduino R3 Footprint with SPI heade سيبي سيبلل ARDUINO_UNO_R3 ARDUINO_R3_NO_HOLES ► ARDUINO UNO R3 SH Arduino R3 Shield Footprin ► ARDUINO_UNO_R3_SH. Arduino R3 Shield Footprint with ICSP H ARDUINO UNO R3 (Version 1) ► SPARKFUN EDISON SparkFun Edisor Arduino R3 Footprint with SPI header ► RESONATOR Resonators (Generic) Arduino Uno R3 Compatible Footprint. Matches PCB size of the original board RESONATOR-8MHZSM. 8MHz Resonator SparkFun Products RESONATOR-16MHZS. 16MHz Resonato ▼ SparkFun-Connectors SparkFun Connectors SAMD21 Dev Breakout R3 Stackable Headers ▶ 6 PIN SERIAL TARGET 6-pin header connection for use with th. ► CONN 01 Single connection point. Often used as ► CONN_06 ► CONN_08 Multi connection point. Often used as G. ► CONN_10 ► I2C_STANDARD Multi connection point. Often used as G. SparkFun I2C Standard Pinout Header Pads ✓ Smds ✓ Description
Search 🐼 arduino Open Library Manager

© Cancel

Ø OK

parts selection as long as it's the correct watt rating and are through hole. Surface mount devices (SMD) are another option as opposed to through hole, and are extremely tiny, but we will focus exclusively on through hole for our example. Here are some pictures of the led and resistors I've chosen to show their symbol and footprints. After adding those parts, use the "add part" button to search for and add "VCC" and "GND" symbols for positive power and ground respectively. Eagle will keep in mind that anything connected to VCC should be connected together since everything needs to get power from somewhere, and same goes for GND. VCC looks like an arrow pointing up and GND looks like an upside down "T" for the variations I chose. Use the "name" R2 button to rename the leds to "pow" and "act" for power and activity indication, and use the "value" button

to change the 5 resistors to have 1 10KOhm resistor and 4 100Ohm resistors. Just as a reminder, we are using 2 100Ohm lets for each led because if we assume the led will absorb 2V, then the Uno 5V pin will supply (5V-2V) / 100Ohm = 30mA to the led, which is higher than our 20mA maximum we stated earlier. However 3V/200Ohm = 15mA, which is perfectly safe for the controller pins and the led. After the names and values have been changed, use the move button and green "Net" button to place the parts into appropriate positions and wire them together like in this screenshot. If you desire, make copies of the VCC and GND symbol to make the design visually neater. At this point, save the project



>VALUE

>NAME

>UALUE

to preserve all of the changes. If we click "add part" and search for photocell, AM2320, or MPL115A2 we will find that the parts are not in the built in libraries by default. To add the parts to our design, we will need to create a part library to define the symbols and footprints for the devices.

To add new parts that aren't already in the parts library, we'll need to create our own library. A library can be created by right clicking the project and creating a new library. I've previously created a library containing some parts for the example payload, these library files must be copied into the current project's folder. Right click the schematic file and click "show containing folder" to view the file in a file explorer window. Copy the library

▼ Projects ▼ 🗁 projects **User Projects** Close Project examplePayload.sch New Schematic workshop parts.lbr Board Rename E hackaday Library Copy ▶ □ examples Fx CAM Job Delete **Edit Description** ULP Use all Script Use none Text Search in folder Folder Project Name

eagle.epf

examplePayload.s#1

examplePayload.s##examplePayload.sch

workshop parts.l#1

workshop parts.l#2

workshop parts.l#3

workshop parts.l#4

workshop parts.l#6workshop parts.l#7

workshop parts.l#8

workshop parts.l#9

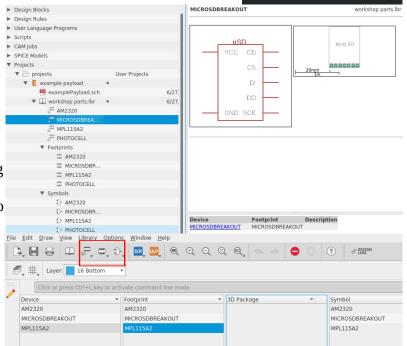
workshop parts.lbr

workshop parts.l#5

files to the project folder, and the library will show up in the project view in eagle. To the right is a screenshot of the project folder with the eagle project file, schematic file, and library files. After these files are added to the project folder, they can be viewed in Eagle like the "workshop parts" library seen in these screenshots. Right click the part library in Eagle and click the "use" checkbox, this will make these parts searchable in the schematic file. Opening the library will show all of the parts available, and clicking on a part will show the symbol and the footprint of the part. Right click one of the parts and click "open in library" to view and edit the part. The library used for the workshop will be missing one part from this screenshot, the photocell, so our next task is to complete this library by making the photocell.

If the library isn't already open, right click the library and click "open". This will open a new window showing all of the footprints and symbols available in the library and the devices that are made from those footprints and symbols. We want to create a new symbol and footprint for the photocell to combine into a device that can be used in our schematic. Start by clicking "add symbol" or clicking the symbol button circled in red in the screenshot to the right, and name the new symbol "PHOTOCELL".

With the symbol editor open, use the line tools to create the symbol and use the pin button to add pins that can be connected to other devices. Create the photocell symbol with pins that align to the top and bottom of the



symbol. The "R2/10k" button can be used to name the pins for when a device has dedicated power or SCL/SDA pins, but since photocells only have two pins and orientation doesn't matter, we won't bother naming its pins.

After finishing the symbol, save and click the footprint button in the top left next to the symbol button and create a new footprint named "PHOTOCELL". The symbol doesn't need to perfectly match our example symbol because the symbol is just an abstraction of the device for use in the schematic. The footprint however is the physical representation of the part and needs to closely match the actual part shape so that when the board is actually made the part can fit. With the footprint editor open, create the following footprint for the photocell. The grid button can be used to change the grid measurements from metric to inches if desired. The green circular "pad" button can be used to create pins for the device, which will create drill holes with metal on them for soldering the part to the final product. The white lines are for making silkscreen graphics on the final board, so it's not essential to be exactly the same as this footprint screenshot. The critical dimensions to copy from the example screenshots are the pad sizes distance from each other. The pads should have a 40 mil drill hole and 74 mil diameter, which is 0.04 inch drill hole and 0.074 inch diameter. The pads should be placed 0.1 inches apart which is 100 mil apart. The same name button can be used to name these pads, which we'd use if we had a more complex device but since this is just a photocell we'll skip

When the part footprint is finished, click the device button above the "layers" dropdown menu and create a new device called "PHOTOCELL". This device will combine the footprint and symbol into a single unit that we can use in our 💠 🗥 schematic and board layout. Eagle separates symbols from footprints because a device can have have many different physical configurations while having the same underlying conceptual behavior, for example photocells can come in many sizes, but the schematic symbol for them would remain the same. In the device editor window, click "add part" to add the photocell symbol, and click the "new" button on the bottom right to add the photocell footprint. Click the "connect" button to associate the symbol pins to the footprint pads. Again, since our device is just a simple photocell, it doesn't really matter which pin is connected to which pad, but more complex devices would require pin names and correct symbol to pad association. If this is done incorrectly for a more complex device, the physical device

that step.

diameter, pads should be name button la more

e withe la more

The late of the

1

+ /

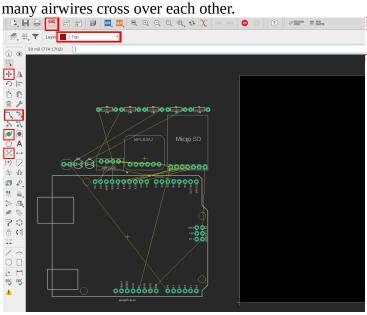
Ö

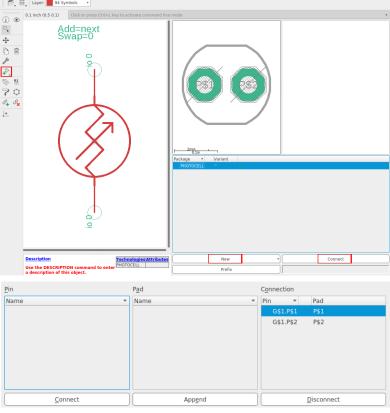
pinout will not match its footprint representation and will be wired incorrectly. With the symbol and footprint pins associated, the new part is complete and the library can be saved and closed.

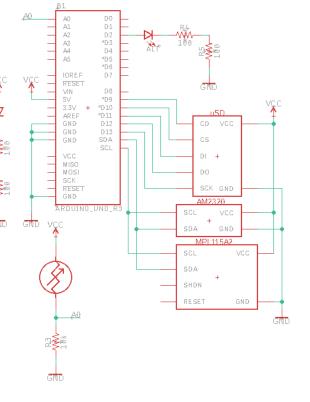
With the library editor is closed, the new part can now be seen in the library contents in the control panel window. Right click the library to make sure that the "use" checkbox is checked so it can be used by the schematic. Open the schematic file and click the "add part" button and search for the library parts to add them to the schematic. The search function can be fickle sometimes, so use the exact part names how they appear in the library by searching for "photocell", "mpl115a2", "am2320", and "microsdbreakout". After adding the parts to the schematic, use the green "net" button to wire the parts together finishing our schematic. The name and label buttons can be used to connect wires without visually connecting them, which can make a neater looking schematic. Here, the photocell connection to Arduino pin A0 is done using wire labels. When the schematic is finished and all parts

connected, save the design and click the gray and green "switch to board" button on the top left part of the window. This will generate a board file for laying out the physical form of the design.

In this board file, the black region surrounded by an orange border represents the physical board. The yellow lines are "airwires" that represent the connections made in the schematic. Move all of the parts into the black region while minimizing how

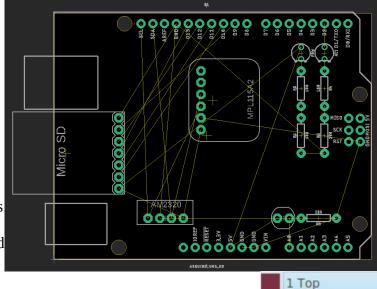






Use the move tool to place the Arduino footprint and the other parts into the black area with an orange border, this represents the region that will be fabricated. Use the move tool to place all of the sensors within the borders of the Uno outline, and rotate them to try to minimize how many airwires cross over each other. The cheap options for fabrication usually only involves two copper layers, one on top and on bottom of the PCB, so airwires crossing represents lots of connections that will end up crossing over each other and can be difficult to define the actually copper wire routes. Here's the board

layout with all of the parts moved to decent positions, note there's less airwire crossover. Notice that the SD card breakout outline sticks out of the Uno outline and orange border, this just means that the SD card outline will be cutoff in manufacturing, but this is ok, since the solder pads are the important part for making our electrical connections. Making the end of the SD breakout close to the Uno edge will also make inserting and removing the SD card easy. Always keep in mind how these sensors work, and what parts could get in the way underneath, for example nudging the SD card breakout upward could possibly make the pins touch the bulky USB port housing and cause an electrical short circuit.



16 Bottom

19 Unrouted

20 Dimension

21 tPlace

22 bPlace

23 tOrigins

24 bOrigins

25 tNames

26 bNames

17 Pads

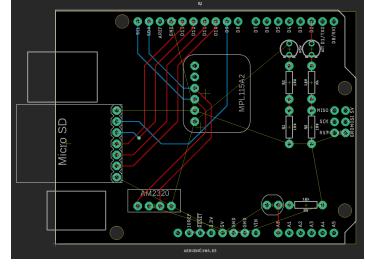
18 Vias

Will all parts placed and the border resized to the Uno outline, it's time to route our electrical connections. The blue "route" button will create a copper trace defining the actual connection between two points. The "layer" defines where the route will be placed, either topside or bottom of the PCB. From this perspective, top side would be the side we're looking at, and bottom would be the side facing the Uno itself when plugged in. The dimension layer defines the outline of the PCB to be fabricated, tPlace and bPlace are for printing silkscreen graphics on the top and bottom layers, and vias are use to allow a route to jump from one copper layer to another.

Using the route tool, create all of the electrical connections until there are no airwires left. Try to route signal wires first on the same layer, and if routes need to cross, route on the other side or use a via to make the route jump to another layer

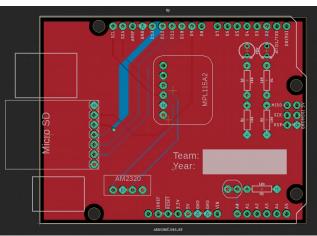
using the middle mouse button. The "ratsnest" can be used to redraw the airwires and copper layers and generally update the view of the board.

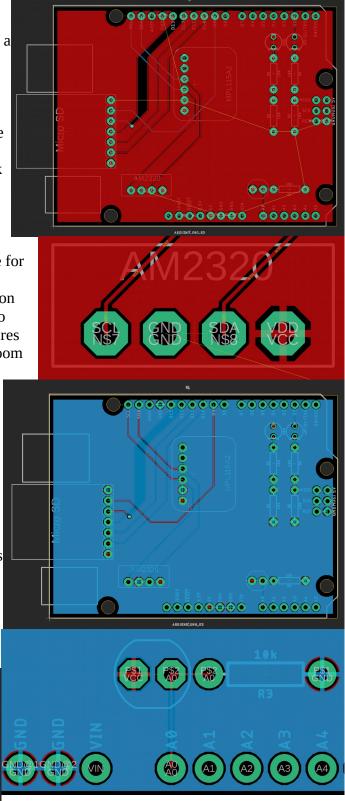
In the screenshot to the right, all of the signal wires are now routed, all that's left are the power pins. Notice that the airwires for the power pins have quite a bit of crossover especially near the SD card and barometer. We could create decent routes by just completely going around the other sides of the pads or by using vias to jump between the top and bottom layer a couple times, but we're going to use another tool to make it simpler on ourselves.



Just above the "ratsnest" button is a pentagon button for drawing polygons. Click the polygon button with the top layer active and draw a square following the orange border line defining the PCB dimensions. When the square is finished, a prompt will open asking what the name of the signal is. Since the top layer is red colored, name this polygon "VCC" because red is often associate with positive power, then click ok and click the ratsnest button to refresh the PCB view. The black areas of the board should now be mostly filled in with red, and zooming into any VCC pin should show this infill connecting to it. This is a copper pour, which simply lays out a huge plate that connects to any pin sharing its name. Do the same for the ground pins on the other side by selecting the bottom layer and drawing a square with the polygon tool around the PCB border and name it "GND" to connect it to all of the ground pins. The blue pictures below show the ground plane completed with a zoom on the photocell circuit showing both the power and ground planes connected to their appropriate pins. As a final touch, use the text and rectangle tool to add a space for signing the team name and launch year on the board using the "tPlace" layer. This will add the text and rectangle using silkscreen.

The board is now finished, and could be sent to a fabrication house, but we should verify its quality first. Fabrication houses like OSHPark don't have the ability to drill infinitely small holes and have limitations on how small features they can create. In fact all fabrication houses have small differences in how they produce parts, so it's important to check to make sure the part is within the abilities of the fab house.





The fabrication constraints are checked by clicking the "DRC" button on the bottom left toolbar or in the tools menu, which performs a "design rule check". The DRC window has many tabs for manually setting the design rules to conform to your fab house of choice, however many provide a file that can be imported to change all of those value manually. Download the OSHPark file for 2 layer boards on this page; docs.oshpark.com/design-tools/eagle/design-rules-files/. After downloading, open the DRC window in Eagle and use the "load" button to load the design rule file, click apply and then click the check button to check the design. If there are any issues with the design, a new window will appear pointing to the features in the design that could present problems. If a feature doesn't seem like it would actually be a problem, the rule violations can be approved to ignore them, but be very careful because imperfections in the design can render the final product faulty.

With the design verified, it's time to upload the board file to OSHPark to check how much it will cost. To do this, point a web browser to oshpark.com and upload the design ".brd" file. OSHPark will render how the board would look after fabrication and may signal any possible problems. The screenshots to the right are the renderings of the top and bottom of the board when I tried uploading. When uploading this file I received a warning saying that my design doesn't contain a bottom silkscreen layer. This is not an issue for me because I didn't add silkscreen to the bottom layer as opposed to the top side that has part labels and the area for signing the team name and launch year.

The price of the board is calculated when the design is uploaded. This design is quoted at \$28.15.
OSHPark charges for 2 layer boards at a rate of \$5 per square inch which come with 3 copies, or \$10 per square inch for 3 copies rushed. This design is 2.7in X 2.09in, which means that any Uno shaped shield will cost around this amount, and double for faster service. In PCB layout, cost optimization is done by simply minimizing the outer size of the board as much as humanly possible.

For extra fun, click the "sharing" at the top of the page in OSHPark to view boards that others have uploaded.

