Table of Contents

Getting Started

This document is meant as a jumping off point for any desired use of the Owls. It is to be used in conjunction with code examples (on Github) and hardware hookup guides. [Ideally this document will eventually be transitioned to a flowchart or other visually intuitive form.] In this document you can find detailed descriptions of all parts of the Owls' functionality, including some projects that are not yet ready to be brought into classrooms.

All of the example code mentioned in this document is stored in the owl_dev repository of the WMSI github account. Github is an online version control repository and code sharing resource, and it is worth learning how to use for any beginning programmer. To get started learning the ins and outs of this awesome software tool, check out the Hello World guide below. If you can't possibly cram learning another thing into your overbooked brain and schedule have no fear, these examples are easy to copy and paste off this web pages linked from this document.

Resources:

- Github Hello World Tutorial

Battery Charging

All of the Owls manufactured so far have been intended to receive power from Lipo or Li-ion batteries through a 2 pin JST connector. This battery type was chosen because of its small size and because they're so easily rechargeable. However, this feature comes with a few caveats. These batteries fare best under constant charge-discharge cycles and have been known to catch fire when charged incorrectly. For this reason it is extremely important to follow these guidelines for recharging Owl batteries

- Use the Feather's built-in charging circuit for any battery under 500mAh. Until we come up with a better method, this is the only circuit we have with current control.
- Batteries 500mAh and up can be charged with Sparkfun's basic Lipo Charger.
- The lipo_charger.brd PCB is meant to be assembled with the Sparkfun Basic Lipo Charger and a 850 mAh battery.

Activities

Start here if you know what activity you want to plan and need resources for pulling the pieces together.

Radio Direction Finding (RDF)

As can be seen in the HAB blog post, radio direction finding techniques were instrumental in retrieving lost tech. This scenario is what originally inspired development of the

Owl kits, as a way to simulate this situation without actually losing lots of expensive tech in the field. Unfortunately, after long hours of testing we were still unable to reliable use RDF methods with either the LoRa or RFM radio boards.

A HAM radio interface (as was used to find the balloon) allows the user to listen to the demodulated signal, making it relatively easy to pick out peaks and troughs. These microprocessor-based boards take the process one step further, by fully interpreting the signal and reporting the decrypted message. Then they report the Received SIgnal Strength Indicator (RSSI) which is a notoriously inconsistent gauge of signal strength. For instance, many users of these boards report their RSSI dropping to -105 dB (about as low as it can go) but then continuing to receive signals successfully for a while longer.

After weeks of testing with several external antenna interfaces it was determined that the RSSI value just wasn't precise enough for radio direction finding tasks. At this point the development of RDF activities for the Owls was put on hold in favor of game design and other uses. The only Owls manufactured with RDF activities specifically in mind were the two Owls given to the Profile school. Therefore the Profile Owl Quickstart Guide might be a good place to start getting ideas for this application. Note that the Owls featured in the Quickstart Guide are slightly different from those being manufactured now: most significantly the "antenna mode" switch is tiny and usually requires needlenose pliers to move, and the power/ ground header pins on either side of the board have been switched (to ground on the left, power on the right).

Resources:

- Using an External Antenna with an Owl unit
- Emails with David Stillman on RSSI readings and radio board range.
- Profile Owl Quickstart Guide

Game Design Unit

Due to their versatility in accepting sensors, GPS, and displays of various types our Owls are uniquely well suited to game design curricula. Check out the Woodland blog posts (part 1 and part 2) for the pilot study of this particular application. Code for the GPS section ("seeking" owl) of the example game can be found in woodland_gps_demo.ino. This program releases a compass bearing whenever the Owl approaches a hard-coded waypoint. The bearing is typically released within about 11m (4th decimal place of lat/ long) of the waypoint due to accuracy limitations of the GPS.

One shortcoming of this example game is that all of the waypoints and bearings must be hardcoded into the program ahead of time, usually after being plotted in Google Earth or a similar application. Ideally this demo game will eventually be improved to the point that the hidden unit can broadcast its location from anywhere within range of the seekers and the seeking Owls will calculate bearings automatically.

Resources:

- Woodland_gps_demo.ino:

Sensor Networks

Although the Owls haven't been specifically tested for this application, they come supported with software and hardware options for setting up a network of sensors. Each unit is capable of taking sensor readings, sending them to other nodes in the network, and relaying data from other nodes. The original premise for this system was that a "base unit", located somewhere easily accessible, could collect and store sensor data from more remote nodes. For example, the base unit might be stationed at Falling Waters Trailhead, with sensing units placed up the side of Mt. Lafayette.

In order to get such a system to work, developers will need to increase the storage capability of the base unit and develop a protocol for collecting data from the base unit. Originally it will probably be best to prototype this system by manually collecting data from the base unit, but eventually the base unit should be able to upload the collected data from all the nodes when it senses an available uplink.

Resources:

- Sensor_read.ino: Takes readings from up to several sensors (limited by analog inputs) and sends them at regularly timed intervals. This same code can be used for transmitters and receivers; the change in mode is designated by the mode select switch.
- See the Sensors section of this document for more information about adding sensors to the Owls.

Built-in Features

The Owls come with a variety of features meant to make them easily adaptable for a whole range of activities. Our intent is the Owl "base model" (of either the Arduino or Feather variety) should work for any of the intended applications with little additional work. This means that soldering is rarely necessary; instead you can use the built-in features and add extra hardware with jumper wires.

Radio Communication

Perhaps the most useful feature of the Owls is their ability to send and receive radio messages. The two versions of the Owl (Feather- and Arduino-based, or Owl and Owl Lite) use different radio protocols and cannot communicate with each other, even though both operate at the same unlicensed frequency of 915 MHz. This frequency falls within the ISM radio band and therefore doesn't require a special license (like a HAM license) to operate. The Arduino-powered Owl is equipped with an RFM69HCW module which claims a 500m range in open air. The Feather comes packaged with a RFM95 LoRa radio module which claims to operate at 1.2mi line of sight. The two board types are programmed via the RFM69 and RH_RF95 libraries, respectively.

Key differences between the two protocols include:

- RFM69 uses a network ID and node IDs while RH_RF95 does not

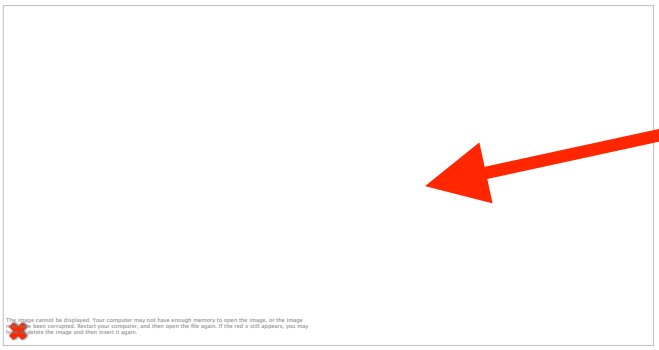
    - This means that an Arduino-based Owl will only receive messages addressed to its node, unless radio.promiscuous() is called in setup
- RFM69 has built-in handling of acknowledgment (ACK) messages. To do this on the LoRa you have to manually code it in

Resources:

- Feather hookup guide with example code
- RFM69HCW hookup guide with example code
- Simple transmit and receive for Arduino
- Simple transmit and receive on the Feather

Internal Speaker

The Owl boards come equipped with a 12mm 2.048 kHz speaker. By default, the speaker must be enabled with a button press in order for it to make noise. (This setup is intended to maximize available pins on the Owls by not allocating one pin for the speaker and a second for the button.) The button press closes a circuit running from DIO pin 2, through the speaker, to ground. In order to bypass this you can solder a second wire to the ground side of the speaker and connect this to another DIO pin, which becomes the SPEAKER_EN (speaker enable) output. This use can be seen in the ticking_timer.ino example on Github.



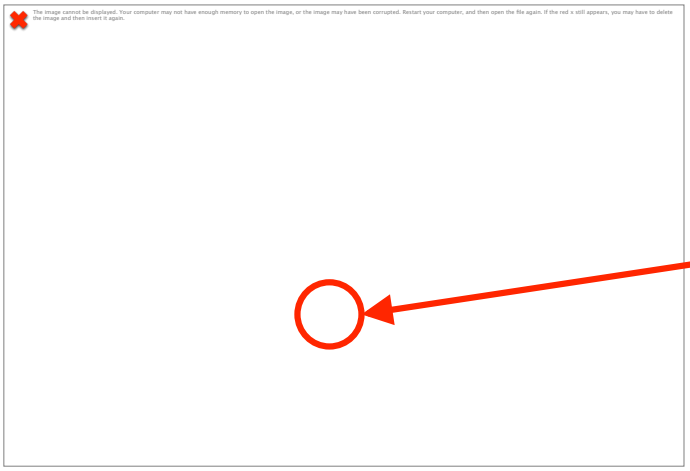
Note the blue speaker enable wire running between the speaker

Mode Switch

Each Owl is equipped with an SPDT slide switch on the right side of the microprocessor (opposite the RFM board for Arduino-base models). In many of the example sketches, the position of this switch is read on startup at which point the Owl boots into either transmit or

receive mode. This distinction was initiated when the Owls were primarily being used for RDF development and is less important for other applications, such as setting up multi-node sensor networks.

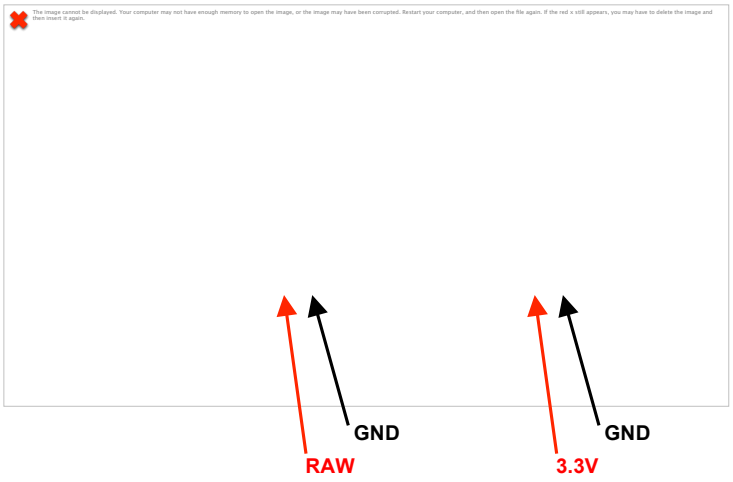
As can be seen in several of the Github example sketches, the mode switch is most often called in void setup() with pinMode(MODESELECT, INPUT); and then in void radioSetup() with digitalRead(MODESELECT). From then on the sketch proceeds in either Transmit or Receive mode and ignores the other mode's instructions entirely. This functionality is meant to make Owl sketches completely interchangeable between units. Even in a sensor network application, you could define one node as the master receiver and all others as sensor nodes, then make this decision on startup with the flick of a switch. For some code-heavy applications (e.g. game design units with different displays on different units) it may be necessary to write different sketches for different units, as can be seen in the Woodland game example code.



Header Pins

In order to allow for future prototyping, the Owls were assembled to leave as many of the microprocessor's pins available as possible. The Owls were also manufactured with power and ground rails on either side of the board. In each set of double header pins the column on the left is positive and the column on the right is ground. Though not immediately apparent, the power rail differs in voltage between the two sets of pins. In the left set, power is connected directly to the battery's positive terminal. This can be useful in powering devices that can handle voltage greater than 3.3V. For displays (like the Serial Enabled LCD and 7 Segment Serial Display) the extra voltage will make the readout a little brighter, and could actually make the difference of being able to read the display in daylight.

Marc bucchieri 4/7/2017 9:16 PM
Comment [1]: make power-left ground-right convention universal to avoid confusion



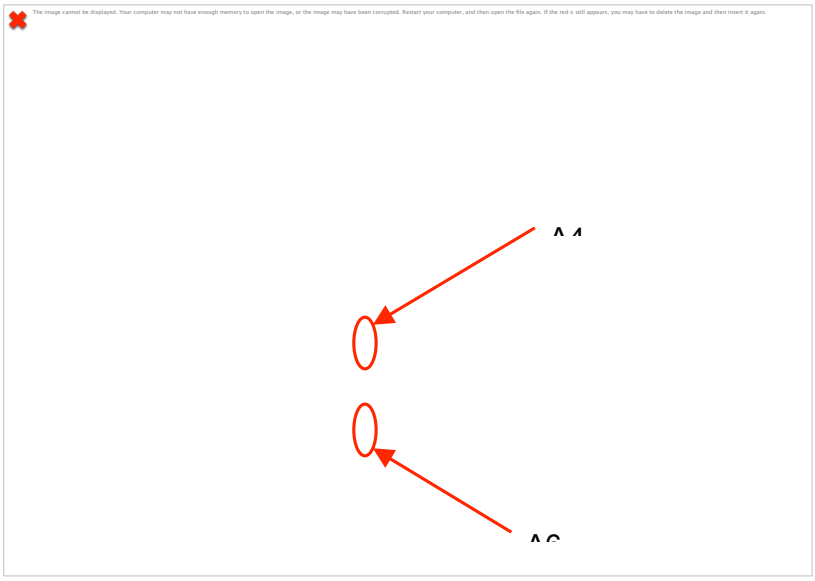
Battery Voltage Checker (Feather only)

https://learn.adafruit.com/adafruit-feather-m0-radio-with-lora-radio-module/pinouts?view=all#power-management

Sensors

Our Owls can interface with just about any sensor capable of running off 3.3V. This includes resistor-based sensors (e.g. photoresistors, thermistors) when wired up in a voltage divider. In this guide we provide support for interfacing with light sensors, temperature sensors, and accelerometers. All other sensors can be set up by following guides specific to your microcontroller.

The Arduino Pro Mini (Owl Lite) has four analog inputs (A4-A7) that are not originally connected to header pins since they don't line up with the edge of the board. If theses input are required extra wires can be soldered to the top of the board.



Resources:

- Sensor_read.ino - this sketch provides support for all three sensors addressed in this guide, with functionality for activating or deactivating each one.
    - Arduino version
    - Feather version

Light

Sparkfun link: https://www.sparkfun.com/products/9088
Note that a photoresistor needs to be set up in a voltage divider for proper use. The center node of the voltage divider can then be read with analogRead().

Temperature

Sparkfun link: https://www.sparkfun.com/products/10988
This temperature sensor is well suited to Owl applications because it requires no extra libraries and does not need to be set up in a voltage divider. To obtain a temperature reading connect the center pin to an analog input and use analogRead().

Accelerometer

Sparkfun link: https://www.sparkfun.com/products/12786

Note that this accelerometer model requires three analog inputs (e.g. A0, A1, A2) in order to read acceleration on all three axes. On the Arduino-based Owls this will severely limit the number of other sensors available without soldering wires to A4-A7.

Hardware Add-Ons

GPS

Sparkfun link: https://www.sparkfun.com/products/13740

This GPS module comes at a very affordable price and supports reasonable good uplink times and accuracy. It has not yet been successfully used in a differential GPS (DGPS) setting but maybe that's for you to figure out :). All of our Owls come with a 3-pin JST GPS connector that fits the end connector of this module.

Interfacing with the GPS requires installation of the TinyGPS library, which uses software serial. Note that any software serial interface requires you to define Tx and Rx pins, which can start to conflict with other parts of your program if you have a lot of digital inputs/ outputs.

Resources:

- Serial interface with TinyGPS from Sparkfun (includes helpful example code)
- Easy example
- Woodland_gps_demo.ino for a more in-depth example of releasing information at GPS waypoints and roughly calculating distances.

External Antenna

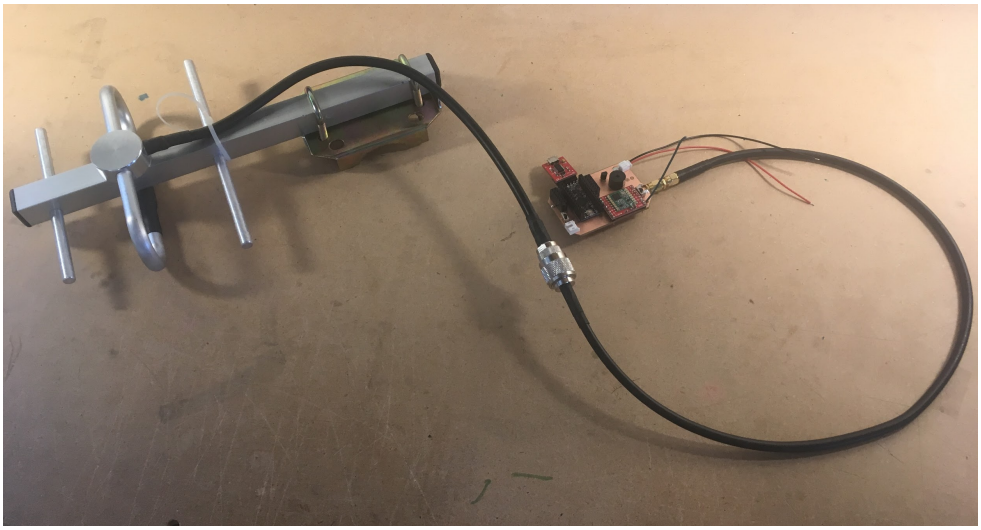
By default, the Owl uses a built-in quarter wave antenna to send and receive messages. It also supports an external antenna connected via the RP-SMA adapter on the top left corner of the board. This adapter supports manufactured antennas (like this Sparkfun Duck Antenna) and DIY antennas built with a short length of RP-SMA cable (such as you could get by hacking apart a cable like this).

The main advantage of adding an external antenna is added range and durability. The ¼ wave antenna is usually curled up to fit in the Owl enclosure (unless you make your own extra tall enclosure) and is somewhat less effective in this configuration than when it is left entirely straight. External antennas are also advantageous in that they can be directional, allowing for Radio Direction Finding activities or other direction-specific applications.

¼ wave antenna (included on all

Antenna select switch

RP-SMA



Owl connected to a commercially manufactured Yagi antenna, connected via an N-male to SMA male adapter cable and RP-SMA to SMA adapter.

External Displays

These can be very useful for game design units, or any other activity where you need accurate visual feedback from the Owl. Note that any Serial-enabled external display will require its own SoftwareSerial object, and each SoftwareSerial object requires its own Rx and Tx pin. If you notice softwareSerial devices acting strangely make sure each pin is only assigned once. You may need to create separate files for Rx and Tx modes, instead of using the mode select switch.

Serial Enabled LCD

Sparkfun link: https://www.sparkfun.com/products/9066

This particular LCD is very well suited to our applications because of the ease-of-use provided by its Serial communication interface. Instead of the usual 10 pin input required by LCDs, this model can be written to with a single output pin, programmed using the SoftwareSerial library. This interface can be buggy sometimes when trying to write to a high volume of text to the screen, so it's best to limit output as much as possible. This means don't have your loop() update the LCD on every iteration, but wait until you actually have something different to display. Using the Serial Enabled LCD can be easily learned through the Sparkfun Quickstart Guide or by perusing any one of our LCD-enabled examples, including woodland_game_demo.ino.

Note that the Feather does not support software serial; instead, use the regular Serial connection with DIO pins 0 (RX) and 1 (TX).

```
Serial1.begin(9600);
Serial1.print… etc.
```

Resources:

- lcd-hellow_world.ino

7 Segment Serial Display

Sparkfun link: https://www.sparkfun.com/products/11442

This display admittedly has fewer practical applications than the LCD, but it does look really cool as a countdown timer. Most notably it was used as the hidden objective for the Woodland school game design unit. Like the Serial Enabled LCD it gets written to via a SoftwareSerial connection on the Arduino, or Serial1 on the Feather.

Production

This section will lead you through the step-by-step process for fabricating a single Owl unit. As the Owls are still a long way from mass production, this section is intended as a set of

guidelines to the way units have been produced in the past. Once you make a unit or two you may gain some insight as to how they can be built more easily/ efficiently- feel free to edit this guide accordingly.

Owl Circuit

At the core of the Owl circuit is a microprocessor (Adafruit Feather M0 or Arduino Pro Mini) and a radio board. The two types of units based on these microcontrollers were termed the Owl (Feather) and Owl Lite (Arduino). The only functional differences between the two models are transmission range and pricing. However, the two radio boards use different communication protocols which require different libraries when programming. For this reason, the Github example code is divided into Arduino and Feather folders.

Surrounding the microprocessor-radio core are a collection of circuit components. These can be grouped as follow:

- The speaker circuit includes a 2.048 speaker, a transistor, and a 220 ohm resistor connected to DIO pin 3. This configuration allows for maximum volume from the speaker by supplying direct battery voltage through the transistor.
- The mode switch switches between power and ground and is connected to DIO 4. This input can be used for any purpose but is typically used to set the unit into transmit or receive mode at startup. In some prototypes there may be a resistor between the mode switch and DIO 4. This allows the pin to be used for other purposes if desired- in this case the switch and resistor would act as either a pullup or pulldown resistor.
- 3 pin JST connector for GPS, wired to DIO 5
- 2 pin JST connector for battery, wired to RAW voltage input (on the Arduino Pro Mini) and left set of header pins. When completely assembled, the positive terminal of the battery is wired through a rocker switch.
- External antenna switch is located next to the radio board and used to switch between the built-in ¼ wave antenna and the SMA adapter.

Resources:

- The Fritzing folder of the Github repository contains Fritzing files showing the breadboard circuit that was originally used to prototype the Owls. Unfortunately there is no easy way to download only these files from the repository, so if you want to view them you'll be better off downloading the whole repository from its home page.

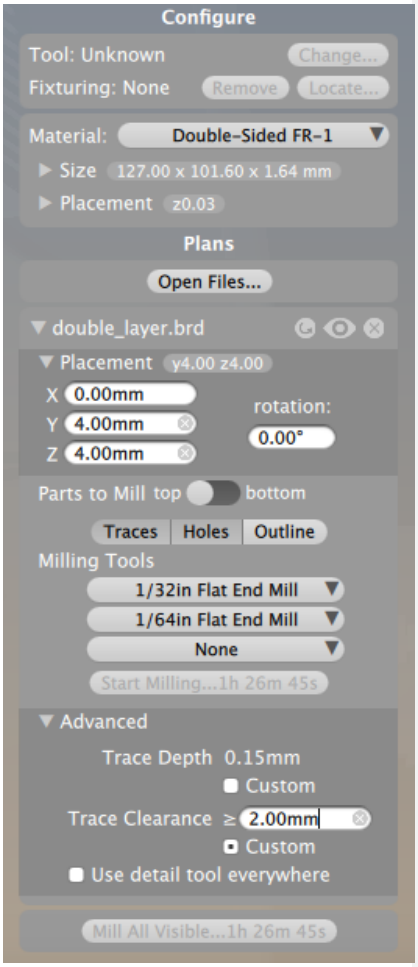
PCB Manufacture

The Owl PCBs are laid out in Eagle and milled on the Othermill Pro. For extra help with using either of these tools, check out the Othermill Training Guide. The Eagle board and schematic files can all be found under “Owl files” in the Eagle folder* of the owl_dev github repository. Specific examples are listed in the Resources below. Note that Github does not directly support downloading a single folder from a repository so if you want to use these Eagle files you should download the enter owl_dev repository.

*The Eagle folder also contains a folder called “Eagle-DRC-Files”. These are design rules files from Othermill to

In general, the PCBs are milled as double-sided boards with most traces laid out on the bottom side. This makes soldering more straightforward, as traces on the bottom connect easily to the solder points for components placed on top of the board. The top of the board is taken up largely by a ground plane, with several vias for grounding traces on the bottom side. These vias should be threaded through with wire and soldered on both sides, as shown in Othermachine’s Double-Sided Boards Tutorial, about a quarter of the way down the page.

When setting up your double-sided PCB in Otherplan there’s a few things you should know about the settings to help it mill as smoothly as possible. Most of these details are covered in the Othermill tutorial and double-sided boards tutorial, so if you’re familiar with these feel free to skim this section. If the Othermill does not already have an alignment bracket attached to the spillboard you’ll want to attach and locate the alignment bracket. Set the material to Double-Sided FR 1, and set the Z offset (from the Placement section, just under Material) to .04mm to account for tape under the board. Make sure your board size dimensions are correct for the double-sided FR 1 (as shown on right). Load one of the files below, and make sure your tools are properly selected in the Milling Tools section. This job requires a 1/32” Flat End Mill and a 1/64” Flat End Mill. Once the tools are selected, check the top and bottom of the board to make sure no toolpaths show up in red. (This usually means that something is too small for the machine to mill.)



You’re almost ready to start milling, but there are a couple other crucial Otherplan settings to check before you press start. Since you’re using the alignment bracket, set the Placement of your board to an offset of 4mm in the Y and Z axes. This will keep the drill bit from colliding with the bracket when you mill the Outline. When you mill the top side (first), you’ll want to

turn off Outline (as shown on right) as this will be cut later with all of the bottom features. You can also set the trace clearance to a custom, larger spacing if desired. This will have little to no effect on the top of the board, but on the bottom it will make soldering easier by providing more space around the pads you solder. The board pictured in this guide

Resources:
(all .brd files can be found in owl_dev>Eagle>Owl files)

- Double_layer.brd - the basic double-sided Owl Lite board.

***** - via to ground plane (top)
b1 - solder points for all-purpose button
b2 - solder points for speaker button
sp - solder point for speaker enable wire. The other end of this wire should be grounded to enable the speaker
r - solder points for rocker switch
s - SPDT slide switch
a - ¼ wave antenna solder point

- Lipo_charger.brd - includes attachment points for a Sparkfun Lipo Charger Basic. Without a charger on the board the only way to recharge an Owl's battery is by removing it and plugging it into an external charger. ****Important to Note** that these Lipo chargers output a constant 500mA current, which is too high for most Lipo and Li-ion batteries under 500 mAh. When assembling an Owl with a built in charger, **always** give it at least a 500 mAh battery. This board layout also include two screw holes for securing the PCB inside an enclosure.

***** - via to ground plane (top)
b1 - solder points for all-purpose button
b2 - solder points for speaker button
sp - solder point for speaker enable wire. The other end of this wire may be grounded to enable the speaker
r - solder points for rocker switch
s - SPDT slide switch
a - ¼ wave antenna solder point
Ω - 220Ω resistor
Li - solder points for the

- Mode_switch_resistor.brd - includes solder points for a resistor between DIO 4 and the mode switch. As discussed in Owl Circuit this allows for DIO 4 to be put to other uses for applications where a mode switch is not required. In this case, the switch and resistor would act as a pullup or pulldown resistor depending on the switch position.

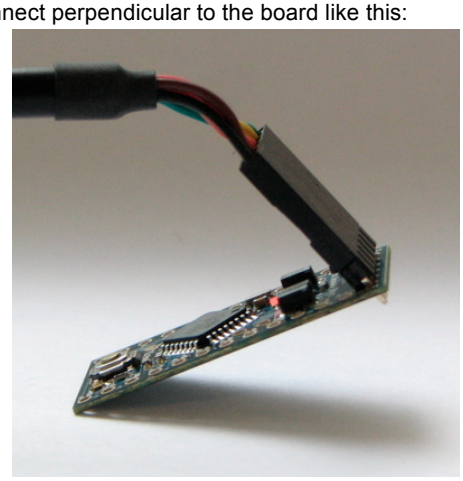
Soldering

Once you've successfully milled out an Owl PCB it's time to solder on the components. The best guide to doing this is by looking at an already soldered Owl board. It can also be helpful to look at the Owl Fritzing Diagram and Eagle schematic for the layout you are using. If you haven't already gotten to the point of cloning the entire owl_dev repository now would be a good time to do that. Once you have the repository stored on your computer, navigate to the Eagle>Owl Files folder and open the board/ schematic files best suited to your Owl application. Currently the most up-to-date versions are lipo_charger and double_layer, but the others may be worth looking at to gain some insight on the evolution of these boards. The annotated board layouts above are a great reference for soldering components with an eye for their functionality.

Basic Soldering Procedure
(may be adapted once you get a feel for your soldering preferences)

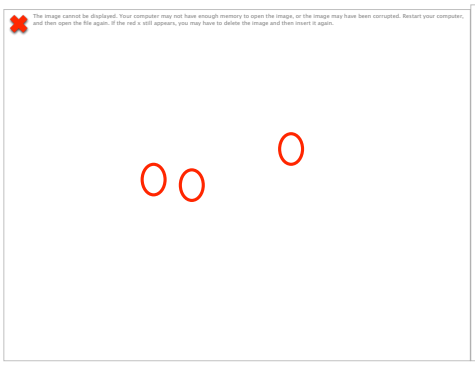
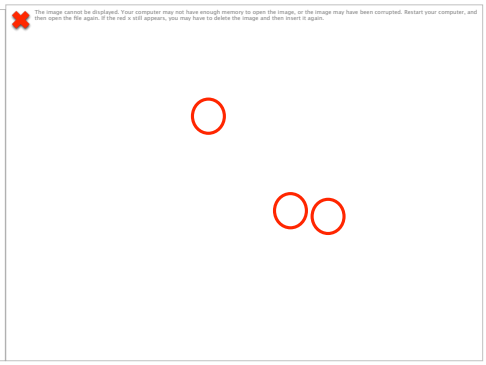
1. Gather all of the components you will be soldering onto the Owl. For an Owl Lite the list of components is as follows:
    - Arduino Pro Mini board
    - Stackable headers for the Pro Mini. Note that these may need to be cut down slightly on the side in order to fit next to each other on the board.

- Set of 6 right angle header pins for soldering the FTDI connection point onto the Pro Mini
    i. This could also use regular male header pins, in which case the FTDI would connect perpendicular to the board like this:

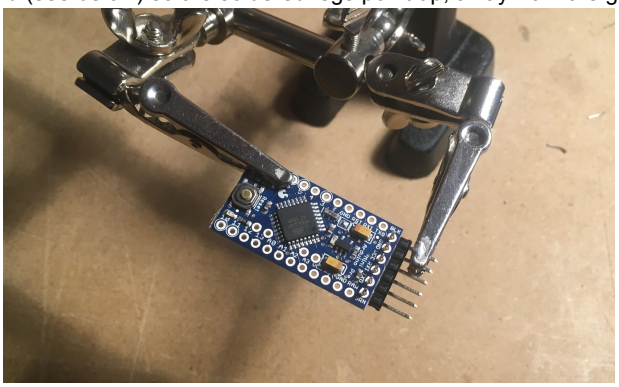


- RFM69 radio breakout board
- Two sets of 8x1 male header pins for mounting the radio board
- Two sets of 6x2 female header pins. These can be trimmed off a longer set of header pins by using wire cutters or the paper cutter
- 220 Ω resistor for the speaker circuit
- PNP Transistor for the speaker circuit. (This Sparkfun transistor works well.) For an overview of PNP vs. NPN transistors check out this guide
- 3 pin right angle JST connector for the GPS module
- 2 pin right angle JST connector for the battery
- Two SPDT slide switches, for the mode select and antenna switch
- Button speaker
- RP SMA external antenna port
- If you're using lipo_charger.brd you'll need these as well:
    i. Sparkfun Lipo Charger Basic
    ii. Set of 2x1 male header pins
    iii. Screw, nut, and plastic washer (can be 3D printed)
- If you're using mode_switch_resistor.brd add one resistor between DIO 4 and the mode switch

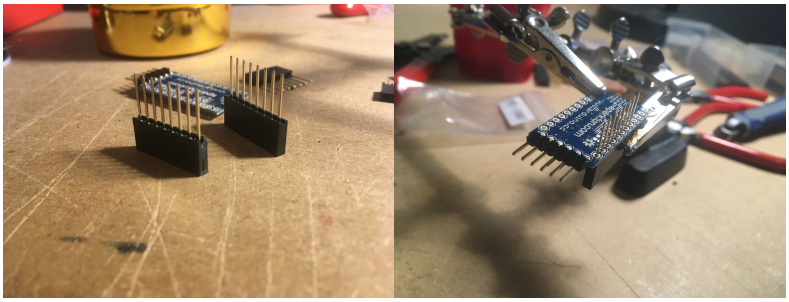
2. Solder all the vias as shown in this guide, about a quarter of the way down under *Design Considerations*. The procedure for this is to poke a small section of bare wire (like a resistor leg) through each via, and then bend it on either side to keep it in place. At this point you can add a small drop of solder to either side of the wire, effectively connecting the two sides of the board. The vias are used to get traces on the bottom of the board to connect to the ground plane on top of the board.

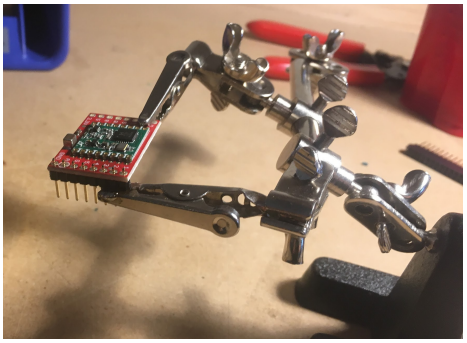
3. Solder right-angle headers onto the Arduino Pro Mini for the FTDI connection point. At this point it's important to make sure that the FTDI connection doesn't short out on the ground plane on top of the board. Mount the horizontal part of the headers on the bottom of the board (see below) so the soldered legs point up, away from the ground plane.

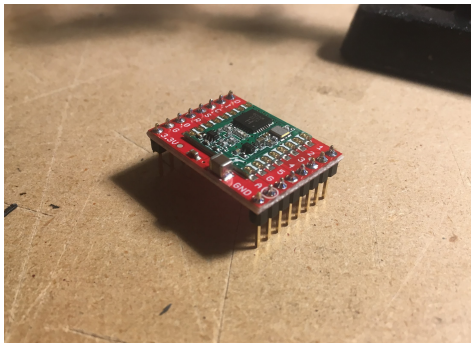


4. Solder header pins to the microcontroller and radio board. The microcontroller uses stackable header pins, which preserve access to pins on the board. These pins sometimes do not fit well next to each other on the Arduino board. To mitigate this you can shave away one side of the header pins' plastic casing with a box cutter.

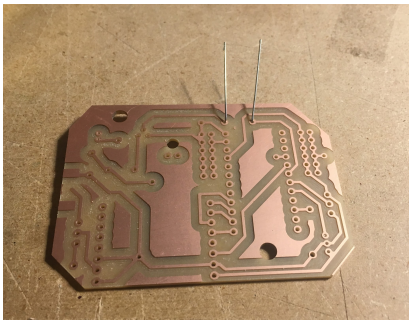


5. Solder the RFM69 breakout board with male header pins.



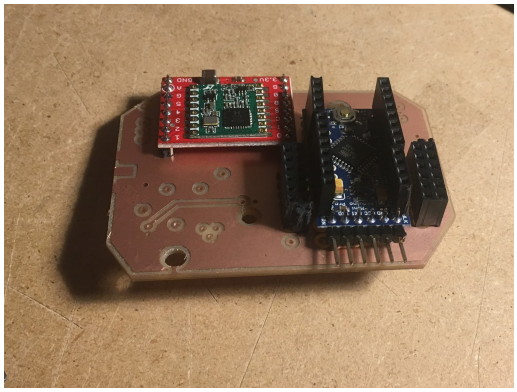


6. Solder the resistor that goes under the microcontroller. *Pro tip: this can also be done as the first step, and then the leftover resistor legs can be used to solder the vias.

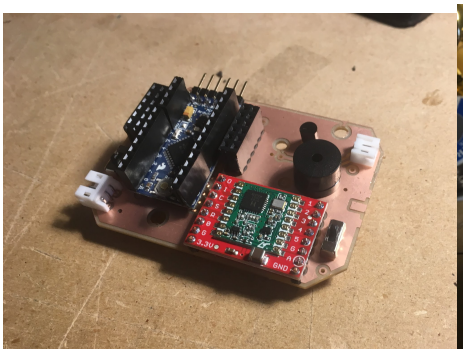


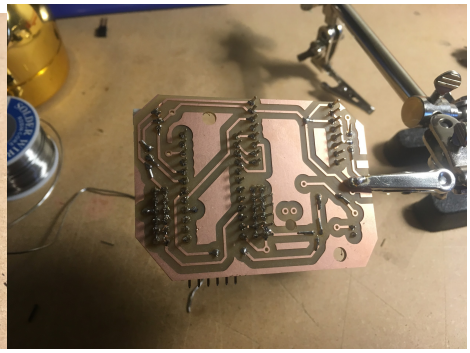
7. Solder the sets of 2x6 female headers located on either side of the Arduino board. Then trim the excess pins that stick out underneath the board.

8. Solder the microcontroller and radio board onto the PCB.

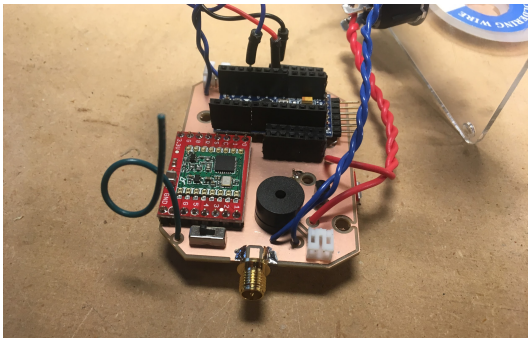


At this point, order doesn't really matter for the rest of the components. I prefer to place as many components as I can, bend the legs under the board, and then solder them all in one go. Depending on your experience level, it may be easier to start with components closer to the center of the board, then work your way out to the edges.

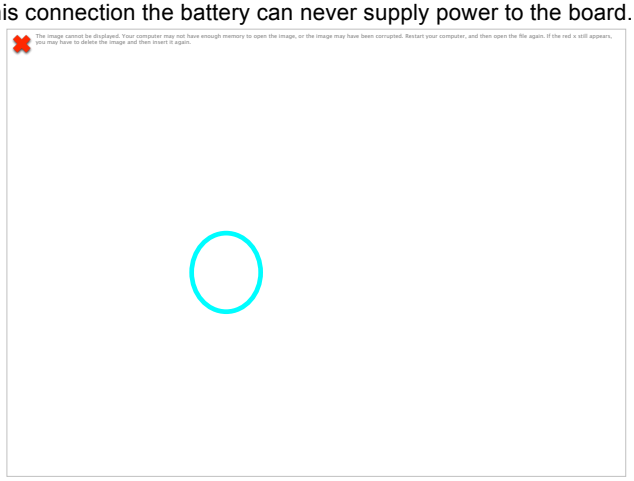




9. The ¼ wave antenna may be soldered on at any point in this process. Measure out a 78mm length of solid core wire, strip a small amount of insulation off one end, and solder it into the hole in the top left corner of the board. You will most likely want to put at least one coil in the wire to help it fit inside the enclosure.



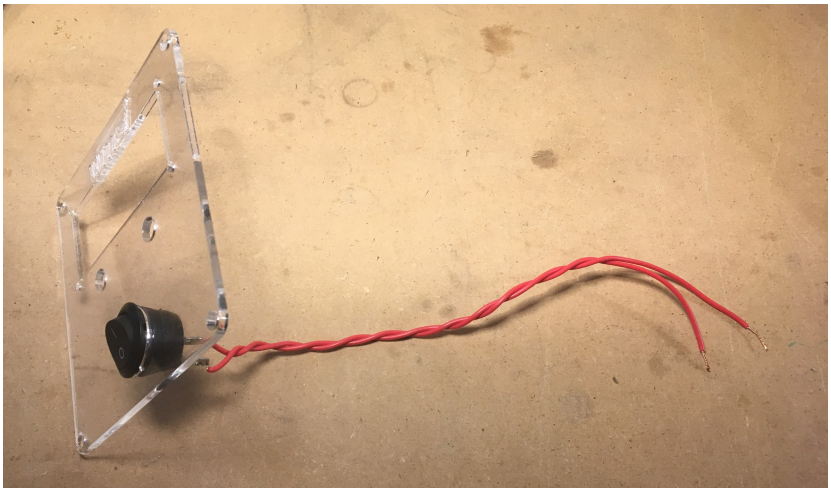
10. Soldering the speaker onto the board can be tricky, since the polarity of both the speaker and the circuit is at first a bit confusing. The polarity of the speaker can be read by flipping it over, where a small '+' and '-' are printed onto its circuit board (light green on dark green). You want to solder it with the negative leg on the left, or outside of the two pads. Another way to think about this is that positive voltage is flowing to the speaker from the transistor, so you want the positive leg closer to the transistor.
11. Solder the ground pin of the 2 pin battery JST connector to the top of the board. This is very important to do (and easy to forget, as one of the only top solder points) because without this connection the battery can never supply power to the board.



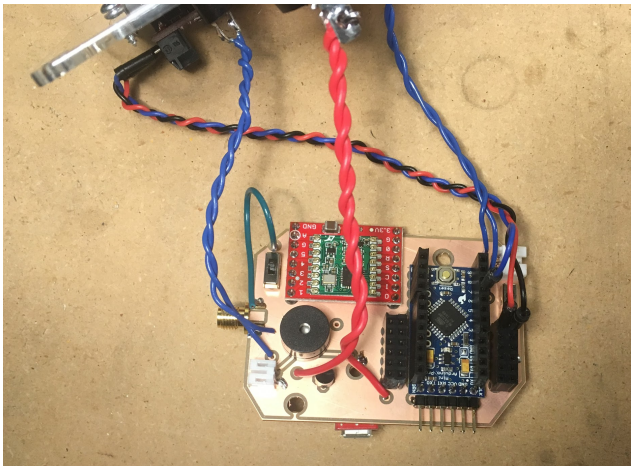
12. If your PCB includes a Lipo charger, solder two male header pins onto the lipo charger so that the black spacer part of the header pins is on the top of the charger. Then flip the charger, so its bottom side is against the bottom of the PCB
13. Before fully moving your Owl into an enclosure there remains the task of adding two buttons and the rocker switch. For these components more flexible wire is better, so it makes sense to use stranded wire instead of solid core.
    - Solder wires to both terminals of the component. I like to do this by feeding the wire through the terminal from the inside, then wrapping it around the outside.

    - Once you have both wires soldered onto the button twist them together. At this point the button is ready to solder to the PCB.

- Don’t solder the rocker switch to the PCB until you have the top of the enclosure ready to go. Then you can snap the rocker switch into the top and leave the wires trailing through. These wires can then be soldered to the two empty pads left near the bottom of the board.



Below is a picture of a completed Owl circuit based on the lipo_charger.brd file. This unit also has an LCD connected with jumper wires. This unit does not yet have a battery or GPS plugged in but is otherwise fully functional. Note that this board will need a 500 mAh or higher battery to match the Sparkfun Lipo charger.



Enclosure

Currently the Owl enclosure is made up off a Home Depot Junction box with a laser cut acrylic top. This system has worked so far because the junction box is reasonably sized and weatherproof, but allows for easy drilling of access holes as needed. The laser cut top is easily adapted to whatever components your particular Owl use case requires. Past designs have included tops with LCDs, 7 Segment Displays, and indicator LEDs.

Moving forward it would be great if WMSI could entirely manufacture the Owl enclosure, both for the cost benefit and in order to make the product more completely our own. This could be done with a single 3D-printed shell (like the junction box) and laser-cut top piece, or by laser cutting all the pieces with finger lock joints around the sides. However, these enclosures also have the disadvantage of being less weatherproof and taking some time to develop and test. For more on this and other future steps for the Owls check out the Next Steps document.

Resources:

- All of the Owl Onshape files are saved under the Owl label on Marc Bucchieri's account, with most relevant pieces included as parts in the Owl_top file.
- Check out WMSI's Laser Cutter training if you need a refresher on how to operate the machine
- The WMSI Laser Cutter dedicated computer has several useful owl top files saved on it. The most recent is simply called owl_top.svg and fits the 3D printed enclosure found in Marc's Onshape account