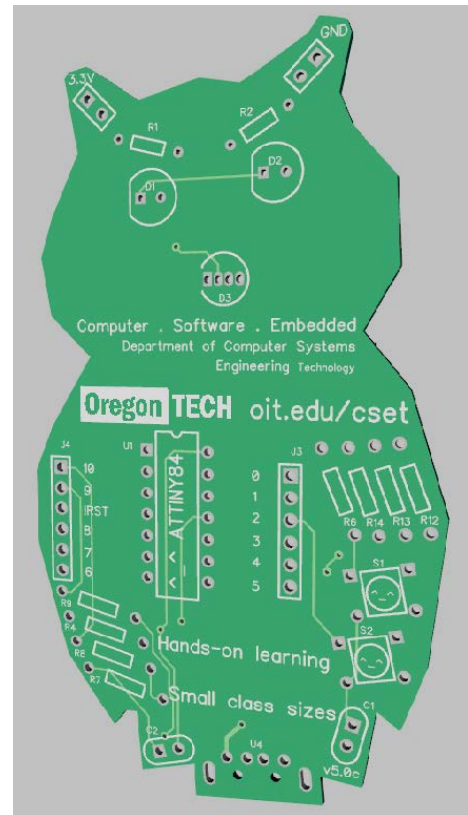
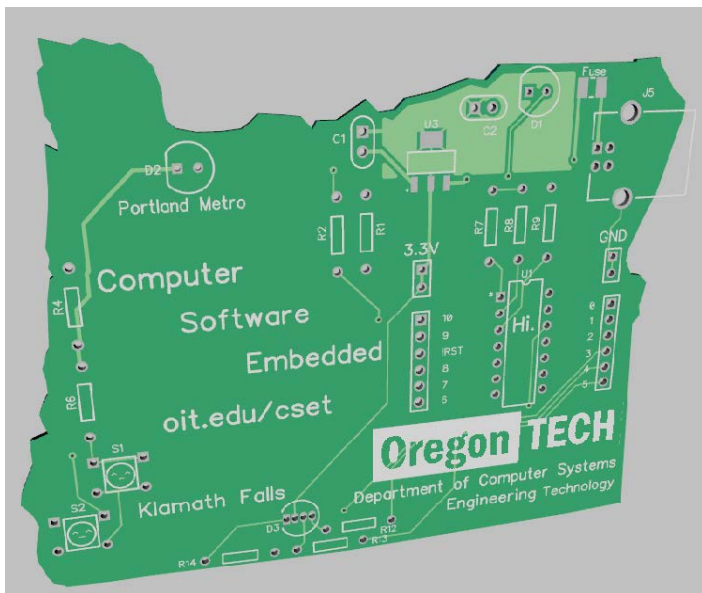


Owlboard v4.0 / Oregon board v1.0

Users Guide / Assembly Guide / Tutorials



Note



The USB bootloader uses V-USB, available from:
<https://www.obdev.at/products/vusb/index.html>

Owlboard Jr/ Oregon shape board utilizes the V-USB/ Micronucleus bootloader. GPL license agreement is available here: https://github.com/kevin-oit/owletBoard_attiny84/blob/master/License.txt

Safety and Liability

If you are participating in the soldering portion of the workshop, you must do the following:

- 1) Listen to and follow instructions from faculty and student assistants.
- 2) Wear personal protective equipment such as safety glasses.
- 3) Use the fume extractor to blow smoke away from you.
- 4) Be aware that the soldering iron is hot, and you should be careful when using it.
- 5) Be aware that the flux and solder fumes, while not hazardous can cause respiratory distress.
- 6) Wash your hands after the workshop and after handling your board.

Be aware that you are putting together a board which may be plugged into your computer. While the faculty and student assistants involved will try and make sure that the board is properly assembled, note that improperly assembled boards or improperly handled boards may cause damage to your computer.

Oregon Institute of Technology and affiliated parties are not responsible for any damage to your personal computer.

Owlboard Jr. v4.0 and Oregon board v3.0 Features

The owlboard Jr. and Oregon board are community outreach boards designed at Oregon Tech. They are open sourced boards using the VUSB bootloader. The software and hardware are available at the link below.

https://github.com/kevin-oit/owletBoard_attiny84

Features of the Owlboard Jr v4.0 / Oregon board v3.0

- ATTINY84 microprocessor in dip package
- One 3mm LED
- One 5mm RGB LED
- Two push buttons
- Integrated USB bootloader (V-USB)
- Built-in USB connector on Owlboard Jr.
- USB-B connector on Oregon board v1.0
- 350 mA poly fuse

- 3.3 V System voltage

Components

Microprocessor – ATTINY84



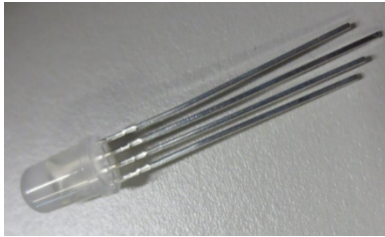
The microprocessor is considered the ‘brain’ of the system, and is what controls all the other components in the system. The microprocessor has been pre-programmed with a bootloader. When you move on to the coding portion of this workshop, the microprocessor is what you will program through the USB connector. The microprocessor is a 14-pin chip in DIP package. Note the notch in the package, which must match the notch on the circuit board. The dot or triangle indicates pin 1, and it must be inserted in the correct orientation. Verify the orientation before soldering.

Regular LED



The LED is a light emitting diode. You can choose between the white, red, yellow, blue, and green LEDs. The LED is a **polarised** component, meaning it must be inserted in the correct orientation to illuminate. The board is marked with a flat side, and the led also contains a flat side. This is an example of a **through-hole** component, meaning the component goes through both sides of the board.

RGB LED



The RGB LED is a light emitting diode like the one above. However, all three colours are integrated into one package. Note the orientation. The flat side on the RGB LED must be inserted in the correct orientation on the board.

USB Connector



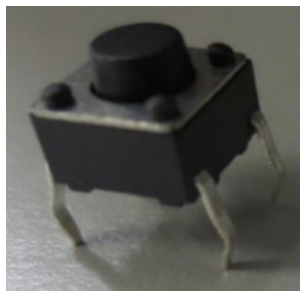
Type B for Oregon board



Type A for Owlboard Jr.

There are two USB connectors that you may see. USB type B is used on the Oregon shape board, and USB type A connector is used on the Owlboard jr. You will use this connector to interface with your computer.

Button



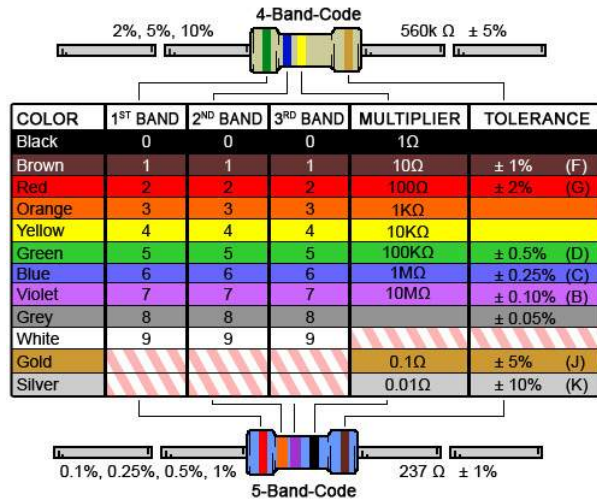
The button is a 6x6x6mm button. When pressed, the button makes a connection. When released, there is no connection.

Resistor



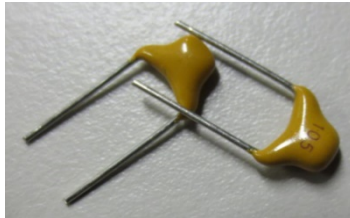
There are various resistors used in this design. There are 330 ohm resistors, 1.5k ohm resistors, and 68 ohm resistors in this design. Resistors are used to limit the flow of current. The colour codes on the resistor indicate the value of the resistor. Resistors are not polarised and can be inserted in any orientation.

You can use the table below to determine the resistor value, and where to insert it in the board.



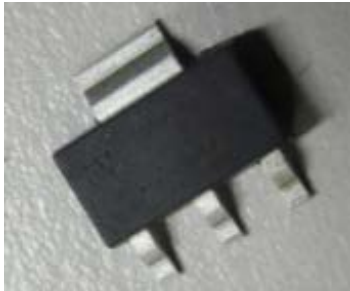
<https://www.digikey.com/en/resources/conversion-calculators/conversion-calculator-resistor-color-code-4-band>

Capacitor



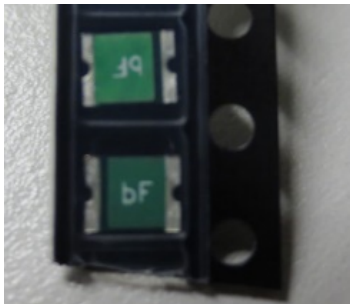
There are two capacitors required in this design. They are labeled 105, which mean they are 1 uF (microfarad) capacitors. These capacitors are not polarised, which means you can insert them in any direction and they will still function.

Linear regulator



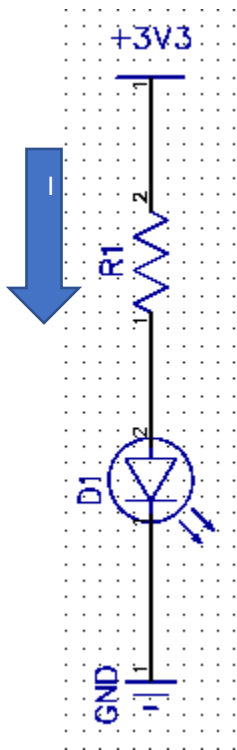
The board receives 5 V from the USB connector and regulates the voltage down to 3.3 V, which is needed for USB communication. This is a **surface mount** component, which means it sits only on one side of the board.

Fuse



The fuse will blow once there is too much current flow. This is a PTC fuse, so once it blows it will recover if left alone for some time. This is a surface mount component, which means it sits only on one side of the board.

Ohms law



Voltage is potential energy at a point. It is measured in volts, and the symbol is **V**. When we say the system operates at 3.3 V, we mean that the voltage of the system is 3.3 volts.

Current is the flow of energy from an area of high potential energy to low potential energy (High voltage to low voltage). Conventional current flows from areas of high voltage to low voltage. It is measured in Amps and the symbol is **A**.

Resistance controls how much current flows through the circuit. Current flows from a high potential or high voltage, to low voltage. High resistance means less current flows. Low resistance means more current flows. Resistance is measured in **Ohms**, and the symbol is given as **Ω** .

Ohm's law is given by **$V = IR$** , where V represents voltage, I represents current, and R represents resistance. This relationship represents the relationship between current, resistance, and voltage.

Problem

If we wish to have 10 mA of current flow to the LED and the LED has a forward voltage drop of 2.0 V, what should our resistor be?

Solution

Based on the diagram, the source voltage is 3.3 V. We must calculate the voltage drop across the resistor, R1 by using Ohm's Law.

$$V = I * R$$

Note that V is not just 3.3 V. V is the difference between the two nodes of the resistor.

$$V = 3.3 \text{ V} - 2.0 \text{ V} = 1.2 \text{ V}$$

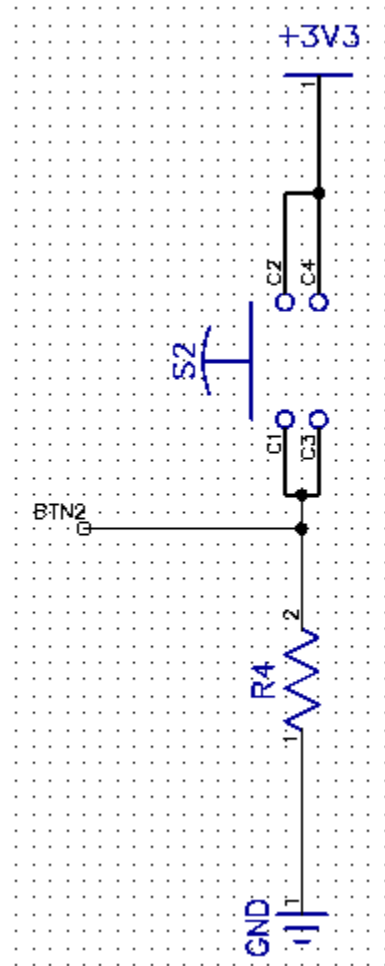
Applying Ohms Law and dividing,

$$1.2 \text{ V} = (10 \text{ mA}) * R$$

$$R = \frac{1.2 \text{ V}}{10 \text{ mA}} = 120 \Omega$$

The resistor should be 120 Ohms.

Button configuration



The buttons on the Oregon board and Owlboard Jr. are setup as 'active high' buttons. BTN2 links to a pin on the microcontroller.

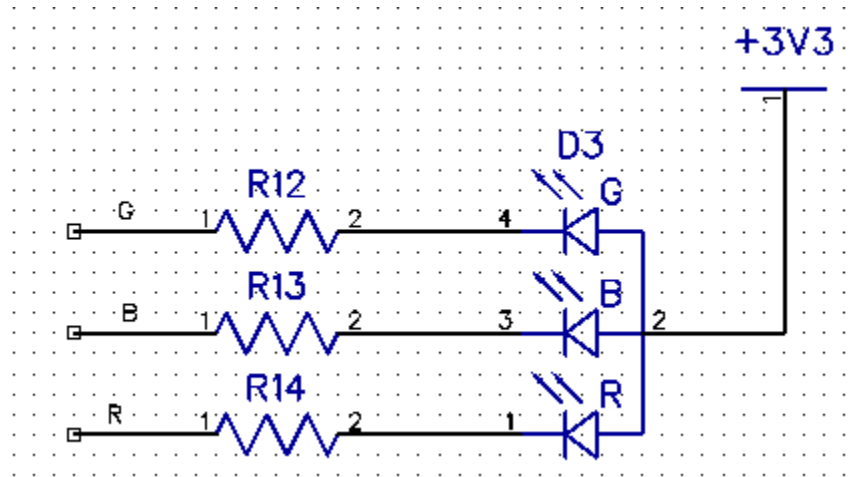
Active high means that when the button is not pressed, the microcontroller input sees the button as logic level '0'.

When the button is not pressed, the BTN2 input is tied to ground through a resistor. The BTN2 on the Owlboard Jr. sees a logic '0'.

When the button is pressed, the contacts C1 and C2 are linked, and the button is tied to 3.3 V. The BTN2 on the Owlboard Jr. sees a logic '1'.

LED active high or low

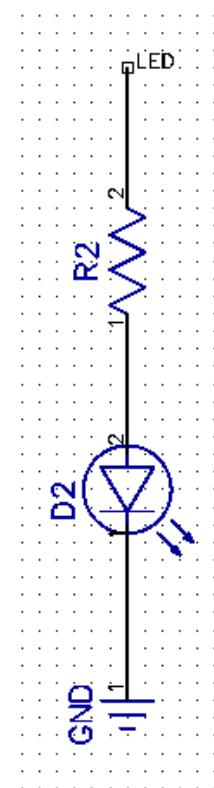
RGB LED



The RGB LED is setup as an active low signal. When you wish for one of the RGB LEDs to turn on, you write a '0' to the colour you want to turn on. This is because the circuit is setup as active low (common anode).

Recall that when you write '0' to the pin, it sets the voltage to 0.0 V, which means current will flow from the 3.3 V through the LED and resistor, into the pin.

Unicolour LED

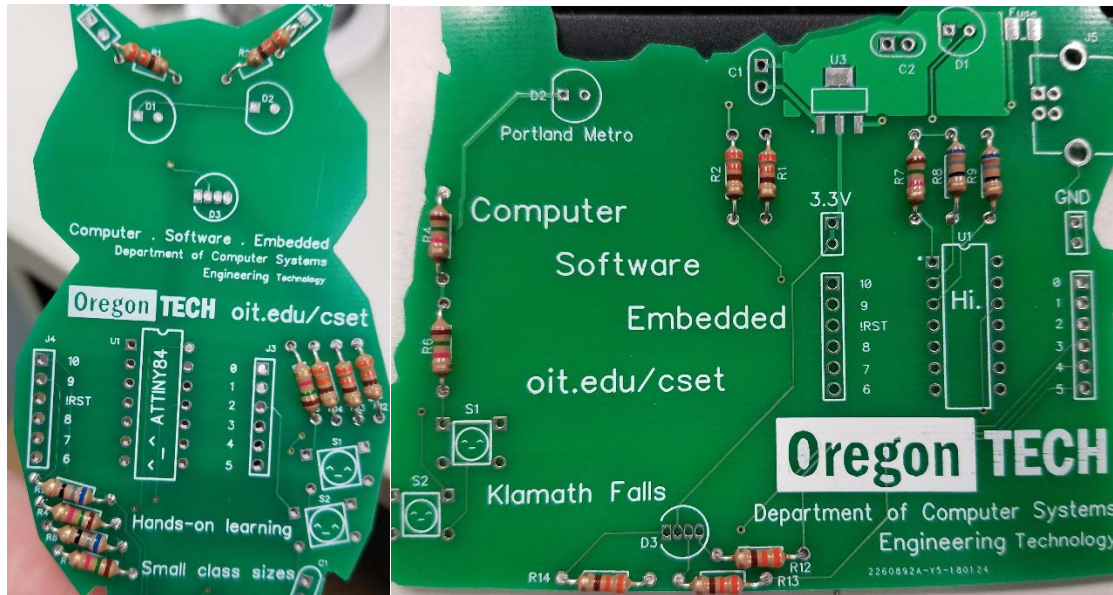


The other LED on the board will either be the 'Portland Metro' LED on the Oregon board and the right eye on the Owlboard Jr. This LED is setup as shown in the figure on the left.

In order to turn this LED on, write a '1' to pin 0. Recall writing a '1' to the pin sets the voltage to 3.3 V, which means current will flow through the resistor through the LED to ground, illuminating it.

Assembly Instructions

Place resistors



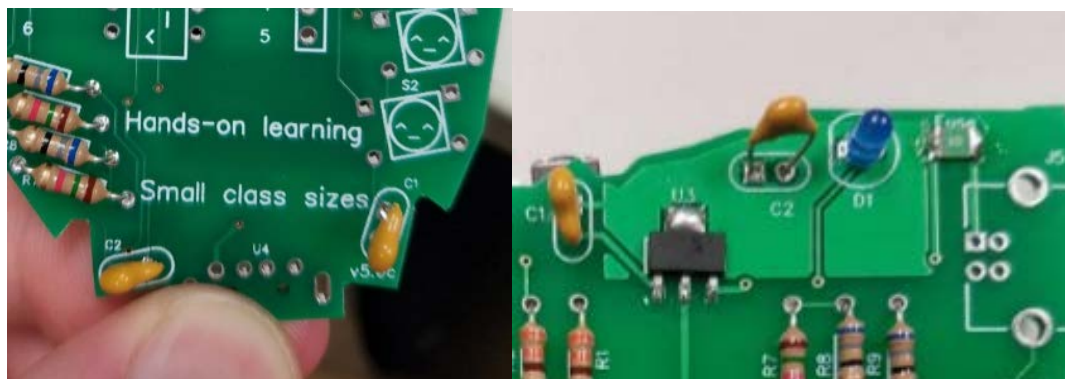
Referencing the schematic, resistors R1, R2, R12, R13, and R14 should all be 330 ohms. This is indicated by the orange-orange-brown-gold stripes on the resistor. Insert all these resistors into the boards and bend the leads. Polarity does not matter.

R8 and R9 should be 68 ohm resistors, with colour code blue-grey-black-gold. Insert these into the board and bend the lead so that the resistors will not move.

The last two resistors are R4, R6, and R7. These can be 1.5k or 10k resistors, and have colour code brown-green-red-gold. Insert these into the board and bend the lead so that the resistors will not move.

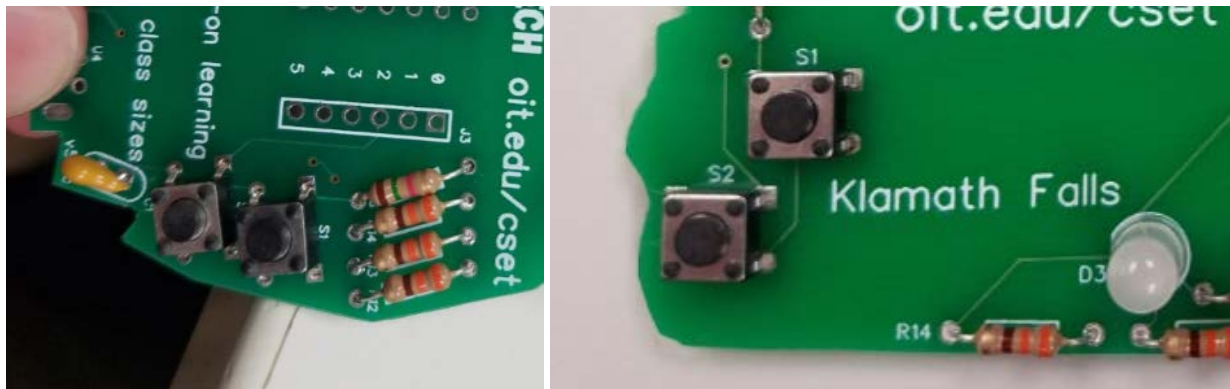
Place capacitors

Grab two yellow capacitors marked '105' and insert them into C1 and C2 references. Bend the leads so the capacitors do not fall out, and make sure they sit flush as possible to the board, like C1. C2 shown in the picture below is too far out.



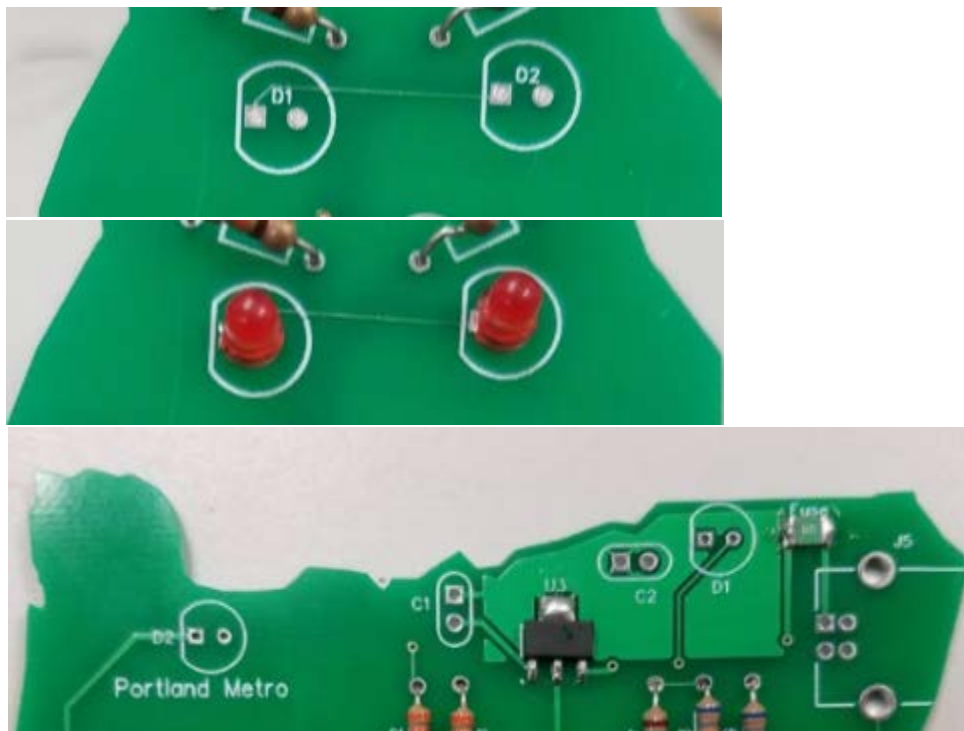
Place buttons

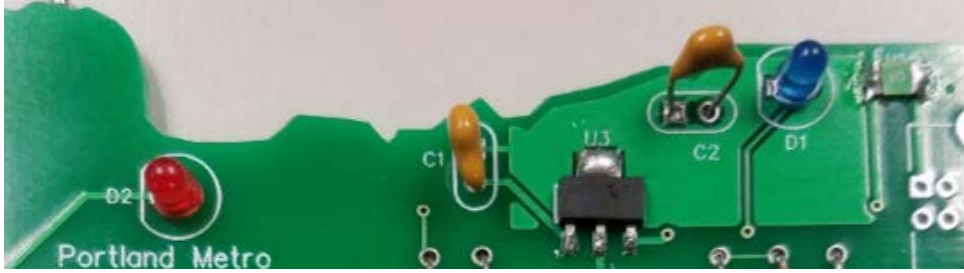
Grab two buttons and place them into the S1 and S2 positions, noting that the leads come out to one side of the button. Bend the leads below to make sure the button does not fall out.



Place LEDs

Now, grab the light emitting diodes of your choice (colour varies). Before placing them, notice that there is a flat side. Make sure the flat side on the LED lines up with the flat side as indicated on the PCB (D1 and D2). Once you are sure it is orientated correctly, place them into D1 and D2. Remember to bend the leads to make sure they don't move.



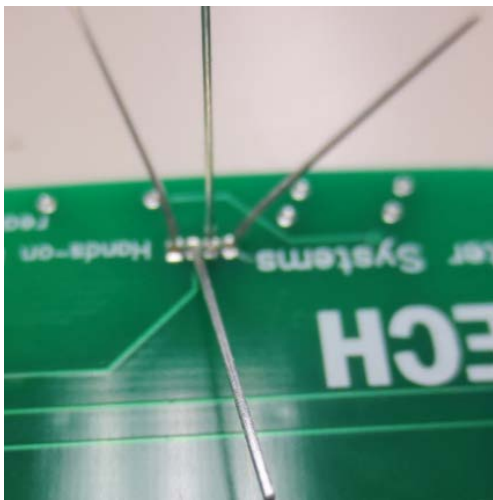


You should also now place the RGB LED into the D3 position. Make sure that the flat side of the LED faces the flat side of the board, as indicated by the line on D3. Remember to bend the leads to make sure the LED does not move.

Note that you need to be more careful with the RGB LED leads, as if you bend them close to each other, they will become shorted.

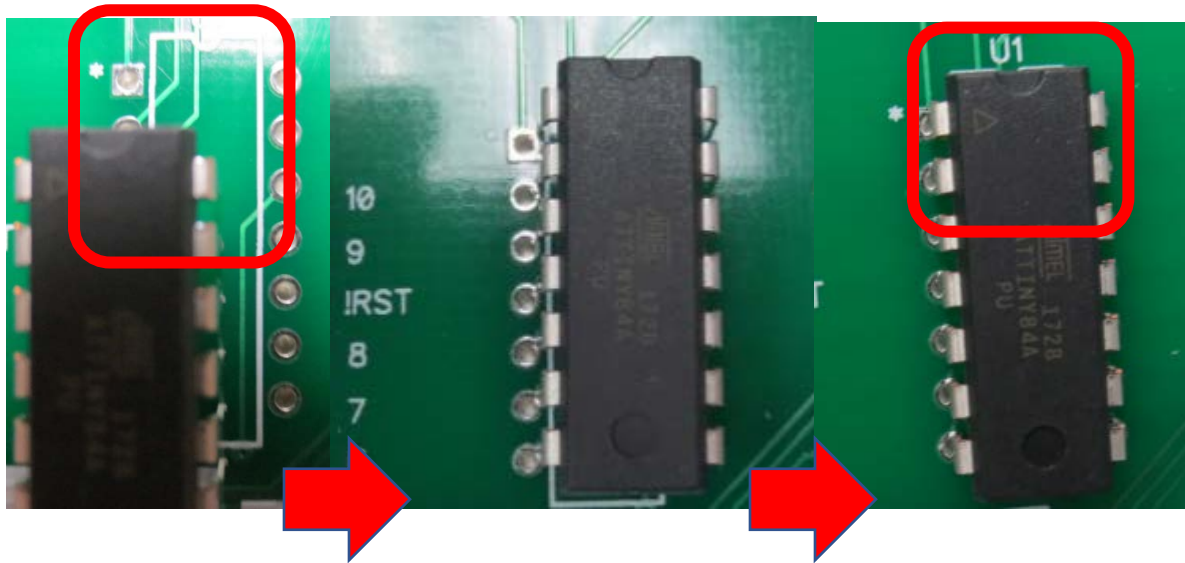


On the back of the board, you should bend leads so that the components do not fall out of the board.

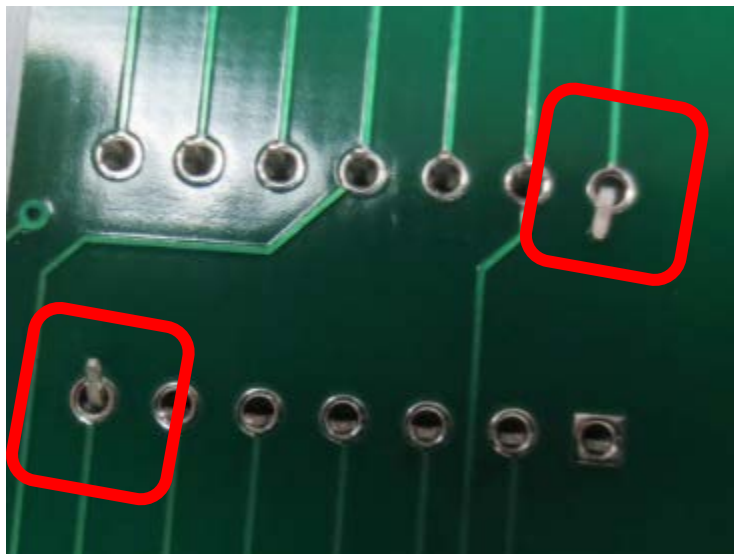


Attach microprocessor

Recall that the direction of the microcontroller is important. Match the notch of the microcontroller to the notch of the silkscreen printed on the board, and insert the microcontroller into the board being careful to not bend any pins.



Insert the microcontroller with the notch on the chip matching the notch on the board.



Bend two of the leads in place so the microcontroller does not move.

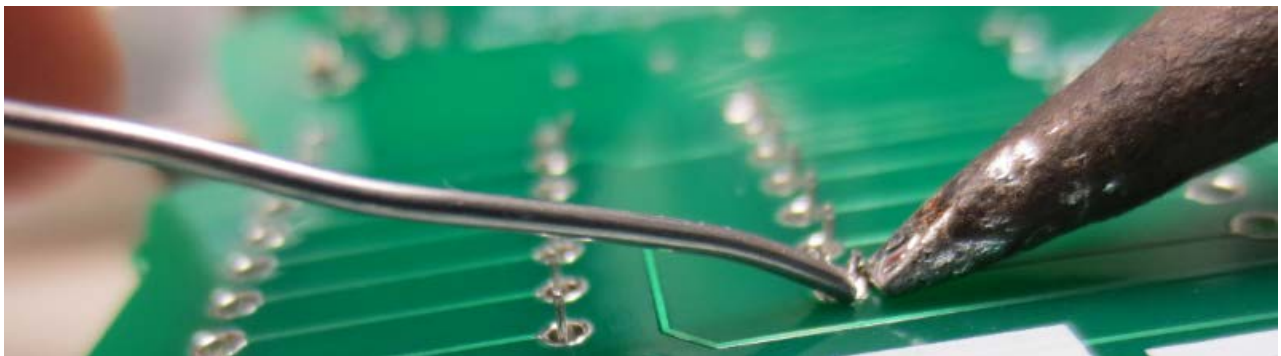
Start soldering

Now that all of your parts are in place, it is time to start soldering. Remember!

- The soldering iron is not a toy. Serious injury can result from misuse.
- Exercise caution as the soldering iron is very hot.
- Do not breathe in the solder and flux fumes. Use the fume extractor at all times.
- The surfaces remain hot for a while after the soldering iron and solder is applied.
- The solder and flux may contain hazardous substances. Wash your hands before leaving.

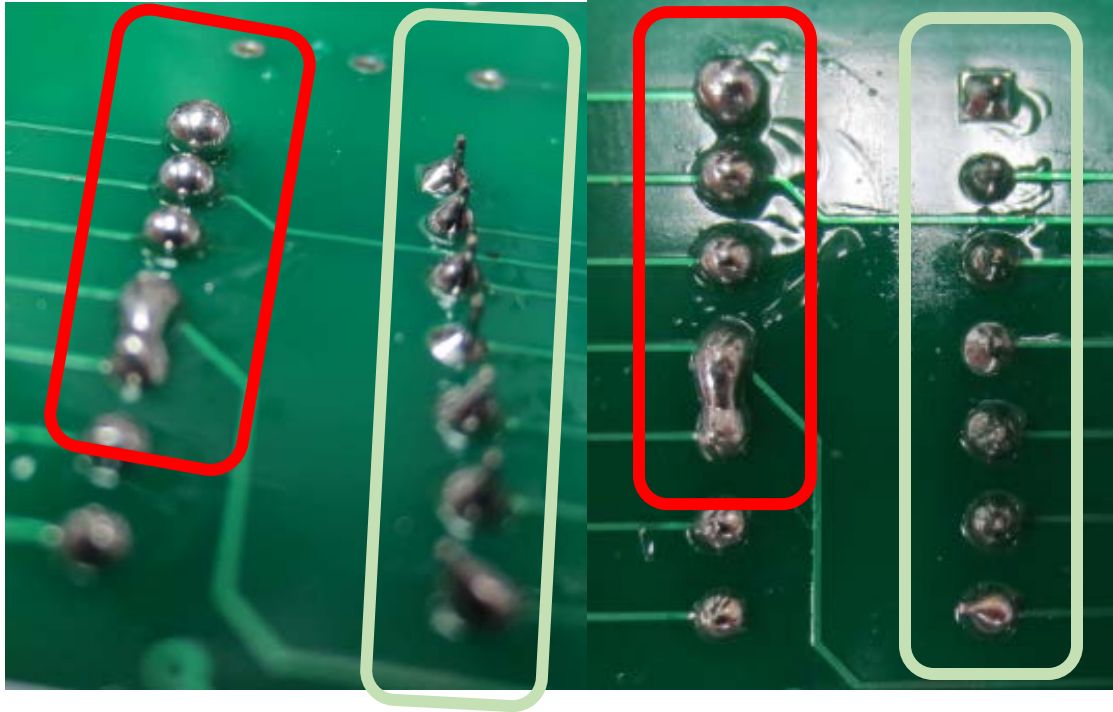
Soldering instructions and assistance are available through the instructor, teaching assistant, or YouTube video link included.

For through-hole components, apply heat to the pad and solder and wait for the solder to wick into the hole. The shape should be reminiscent of a Hershey's kiss. Do not hold the soldering iron on components for too long (Max. 4-5 seconds) or you may damage the components.



Some components may have long leads. After soldering, trim the leads.





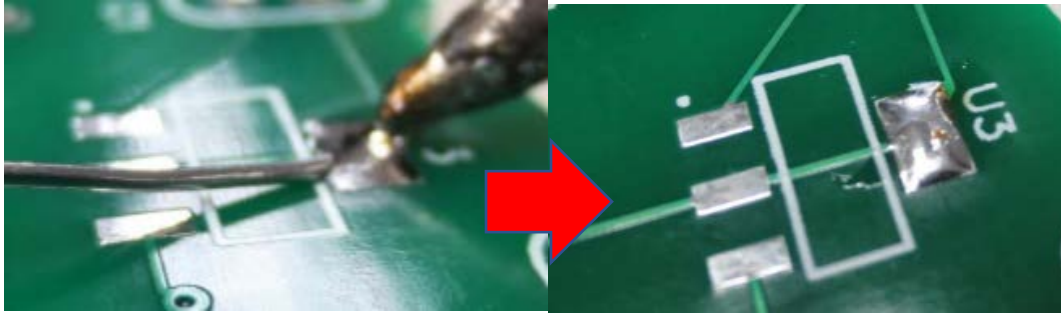
BAD

Red squares indicate bad solder joints. The first three are overfilled, whereas the last bottom two are shorted.

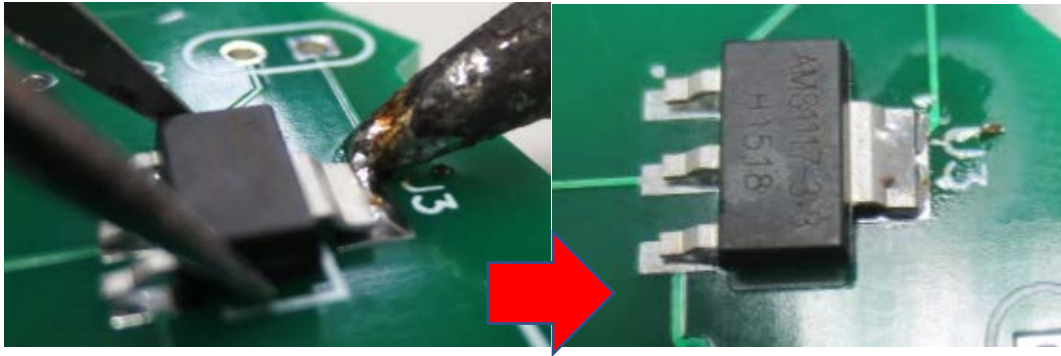
GOOD

Green square indicates good solder joint, tends to be in a Hershey's kiss shape.

Attach linear regulator
Load the pad with solder.



Grab the linear regulator as shown with tweezers and heat the big pad until the component sits flush with board. Remove iron and hold tweezers in place until component will not move (1-3 seconds)



Solder remaining pads.



Attach fuse

Put solder down onto the pad as shown.



Place the component using tweezers and heat the pad with solder. After component sits flush, remove soldering iron, holding component with tweezers for 1-3 seconds until the component does not move.

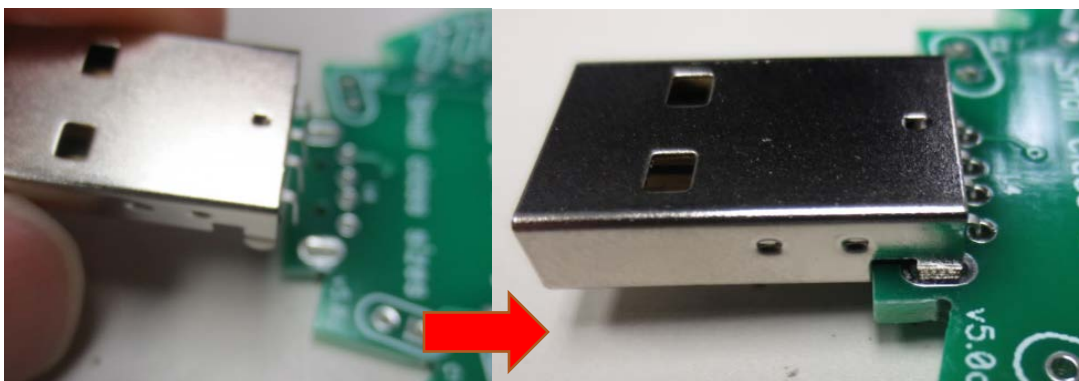
Feed solder into the other pad. The finished result is shown below.



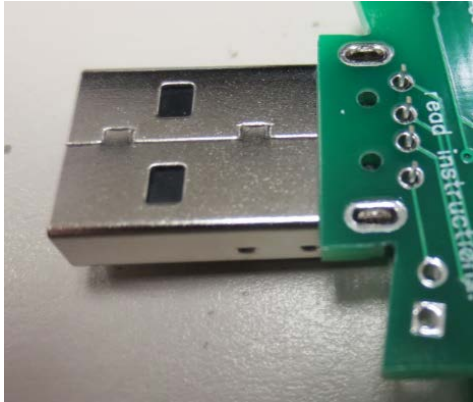
Attach USB connector (Owlboard Jr.)

If you are building Owlboard Jr, use instructions below. Otherwise skip to ***Oregonshape board attach USB connector*** section.

Place the USB connector into the board as shown.



Make sure all pins and holes shown are filled with solder.



Attach USB connector (Oregon shape board)

Place the USB connector into the board as shown.

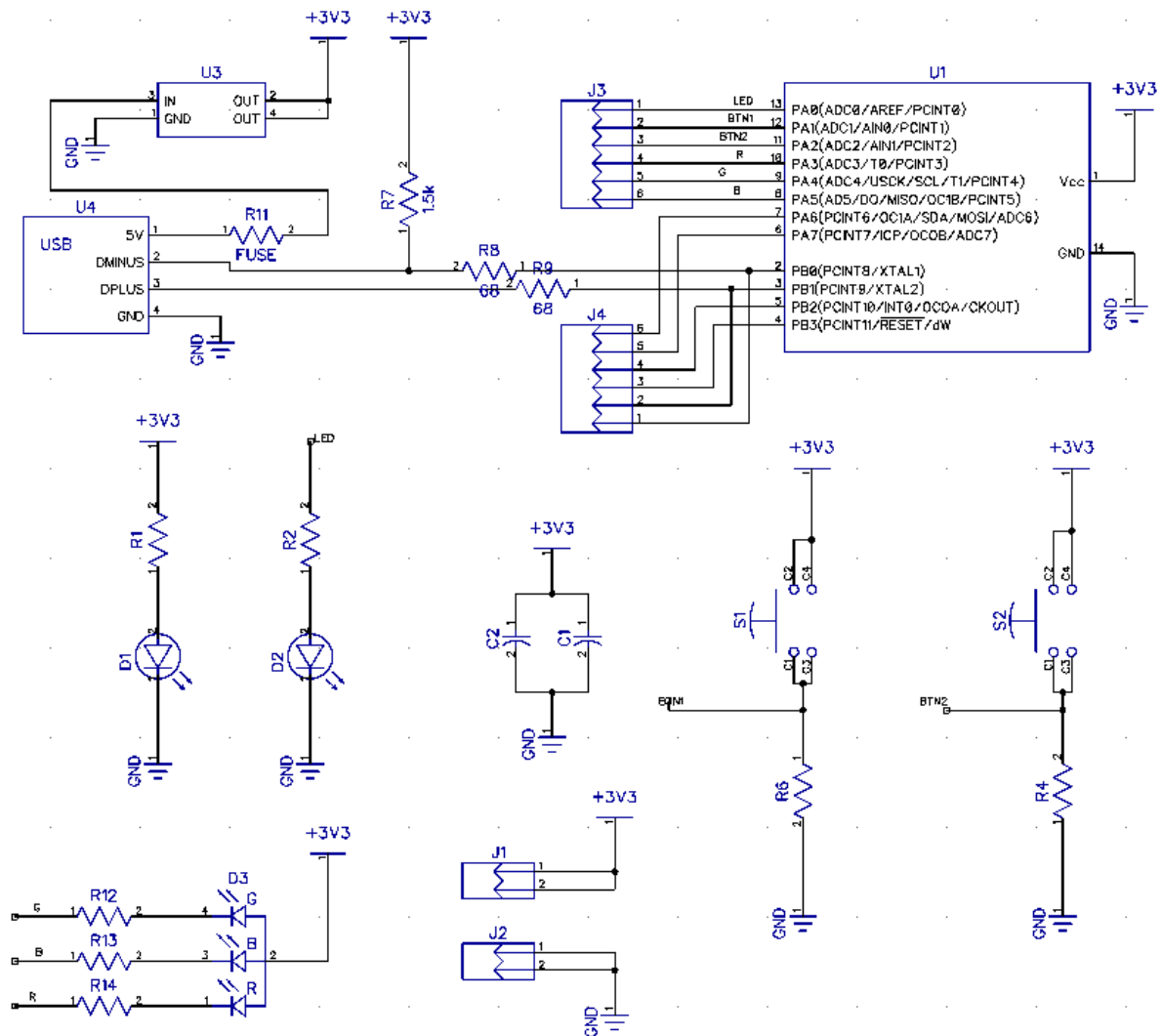


Make sure all pins and holes shown are filled with solder, and that the pins do not short.

Congratulations.

You have built your Owlboard Jr. or Oregon shaped board. Ask your instructor or teaching assistant to look over your board before plugging it into a computer.

Schematic



Bill of Materials

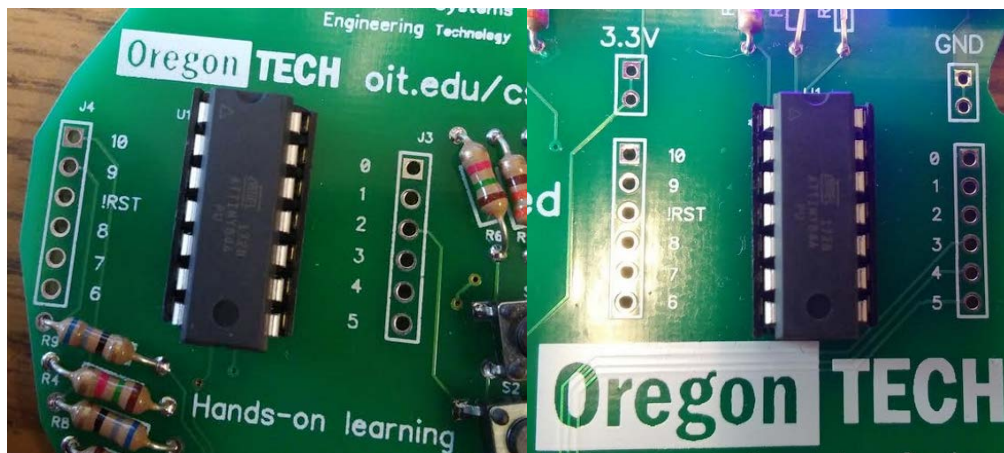
Component	Quantity	Reference Designator
RGB LED	1	D3
Regular LED (Always use red for Portland Metro LED)	2	D1, D2
Linear regulator AMS1117-3.3	1	U3
Ceramic capacitor 1.0 UF	2	C1,C2
330 Ohm resistor	5	R1, R2, R12,R13,R14
68 Ohm resistor	2	R8, R9
1.5K or 10K resistors	3	R4, R6, R7
USB connector type A (Owlboard Jr.) or type B (Oregon board)	1	J5/U4
Button	2	S1, S2
Fuse	1	Fuse
Male 0.1" Headers (1x6)	2	J3, J4
Programming header and power headers optional	--	--

Board setup – Buttons, LEDs, and pin locations

When writing code for the Owlboard Jr. or Oregon shaped board, please use the pin numbers below in Arduino IDE.

Pin number reference

Arduino Pin Number	Owlboard Jr.	Oregon shaped board	Logic
0	Right Eye LED	Portland Metro LED	'1' = ON; '0' = OFF
1	Switch 1	Switch 1	PUSHED = '1'; ELSE = '0'
2	Switch 2	Switch 2	PUSHED = '1'; ELSE = '0'
3	Red RGB	Red RGB	'0' = ON; '1' = OFF
4	Blue RGB	Blue RGB	'0' = ON; '1' = OFF
5	Green RGB	Green RGB	'0' = ON; '1' = OFF
6	Pinout to header		
7	Pinout to header		
8	Pinout to header		
9	Pinout to header		
10	Pinout to header		



For example, if you wish to turn the Portland Metro or Right Eye LED on, you would need to:

- 1) Set pin 0 to an output

```
pinMode(0, OUTPUT);
```

- 2) Write in the loop logic '1'.

```
digitalWrite(0,HIGH);
```

If you wish to configure the button S1, you would need to:

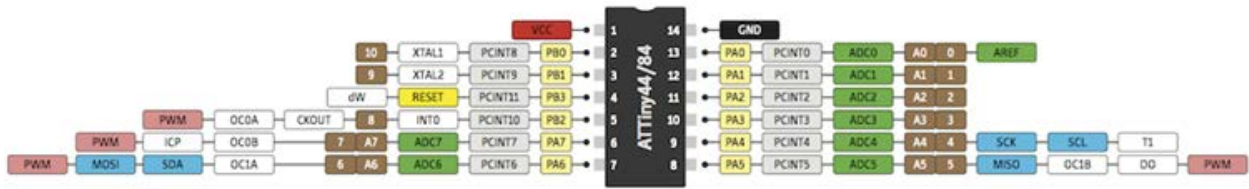
- 1) Set switch 1 to input

```
pinMode(1, INPUT);
```

- 2) Read the button

```
digitalRead(1);
```

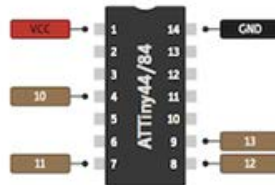
Pinout of ATTINY84



LEGEND

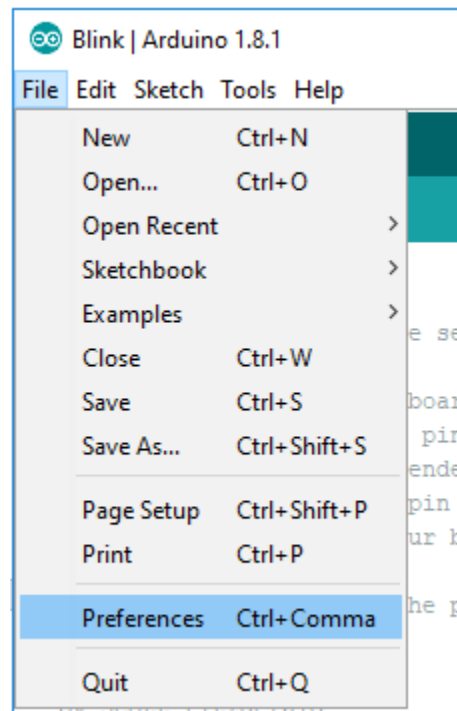
GND
POWER
CONTROL
PORT PIN
ATMEGA328 PIN FUNC
DIGITAL PIN
ANALOG-RELATED PIN
PWM PIN
SERIAL PIN
ARDUINO PIN

Using Arduino as ICSP Programmer for ATTiny44/84

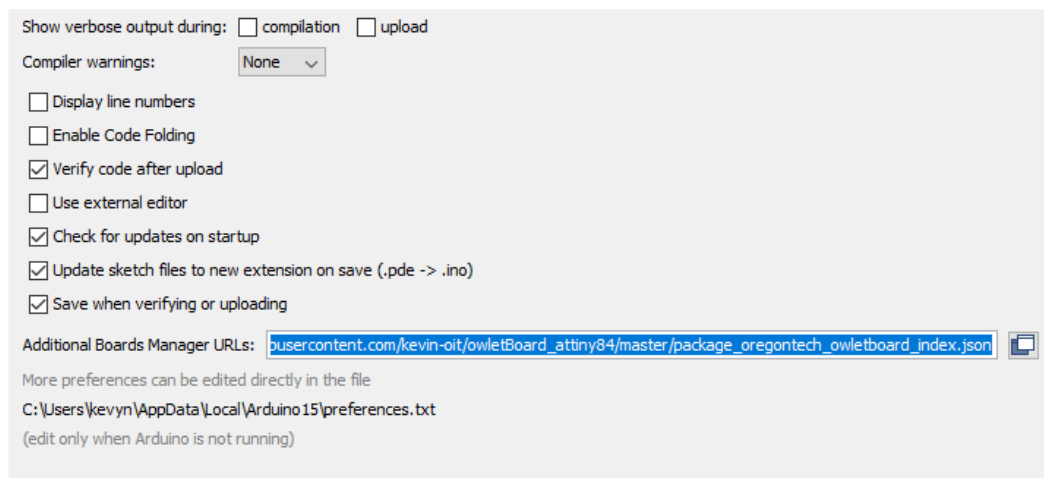


Installation

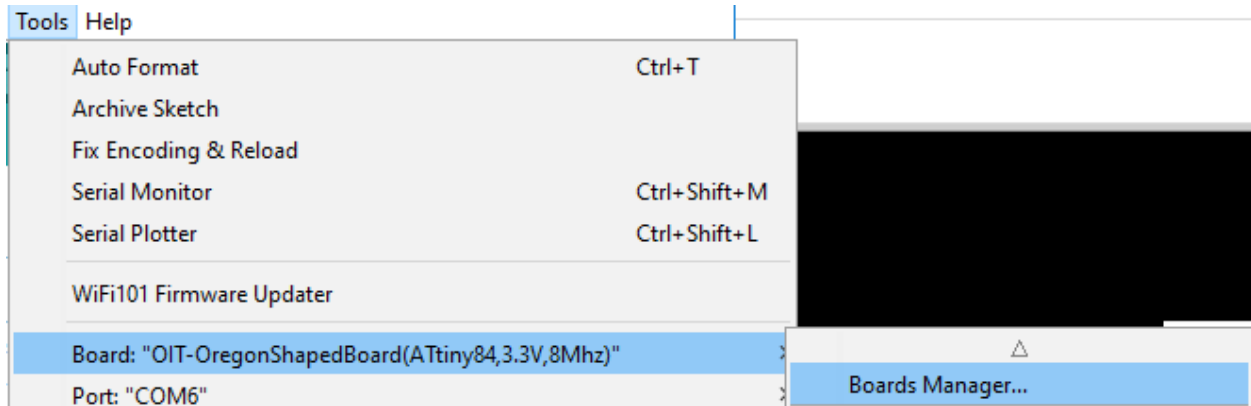
1. Install Arduino IDE from this link: <https://www.arduino.cc/en/Main/Software>
2. After Arduino IDE is installed, launch Arduino IDE.
3. Go to File > Preferences.



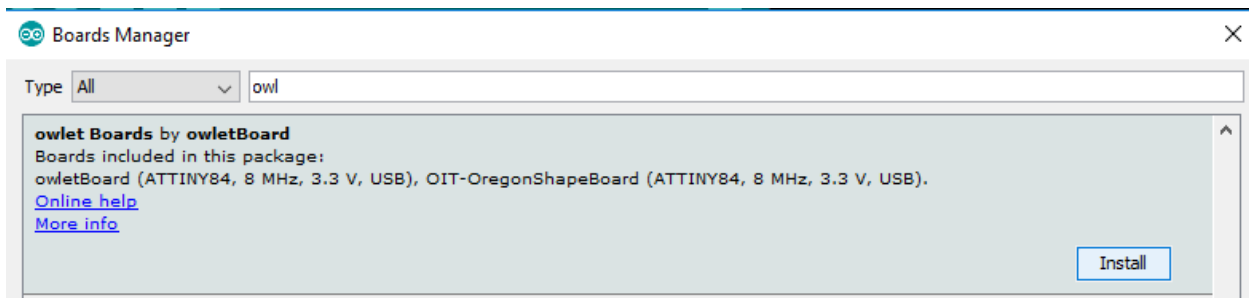
4. You should see the following screen.



- Under Additional Boards Manager URL paste this: https://raw.githubusercontent.com/kevin-oit/owletBoard_attiny84/master/package_oregontech_owletboard_index.json
- Click OK.
- Go to Tools > Boards > Board Manager

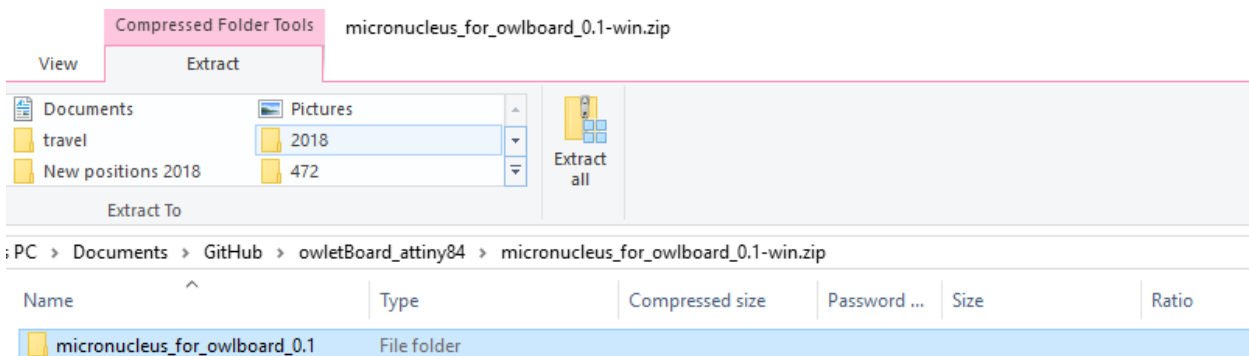


- Type owl into the menu bar and you should see owl Boards by owlBoard. Click install.

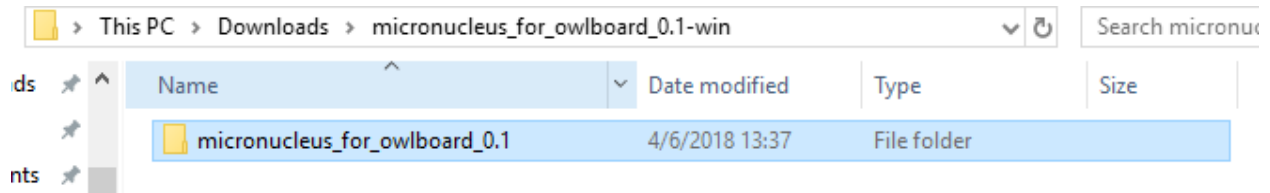


- Download this file: https://github.com/kevin-oit/owletBoard_attiny84/blob/master/micronucleus_for_owlboard_0.1-win.zip

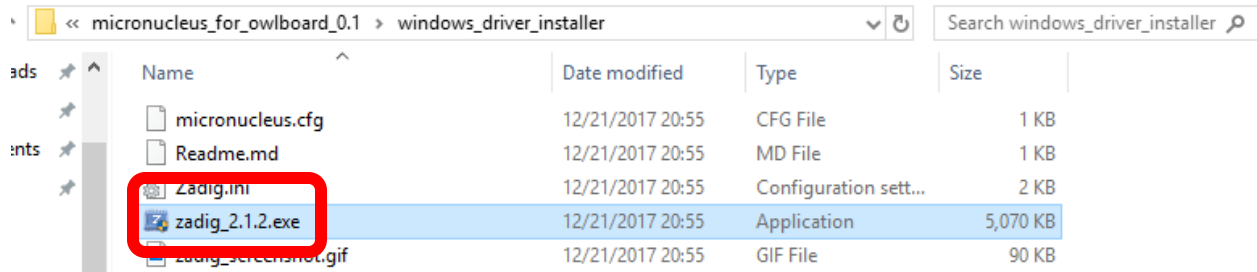
10. Unzip the file.



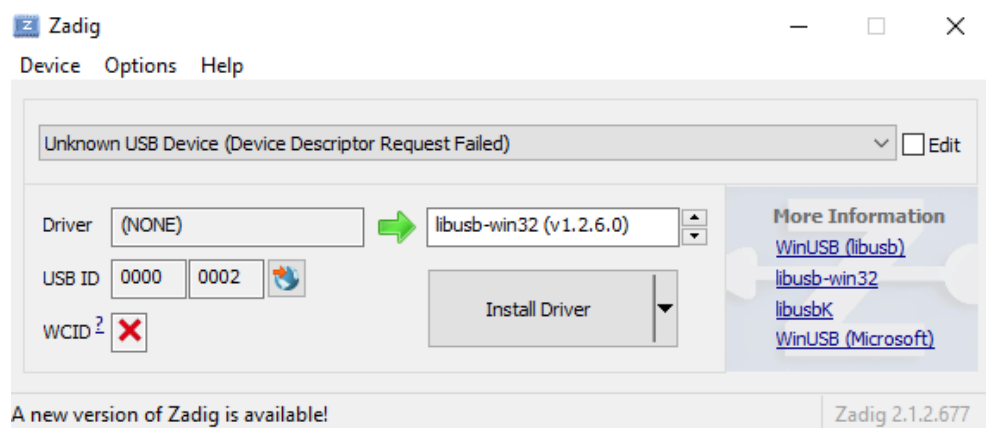
- When the files are extracted, a new window will open up.



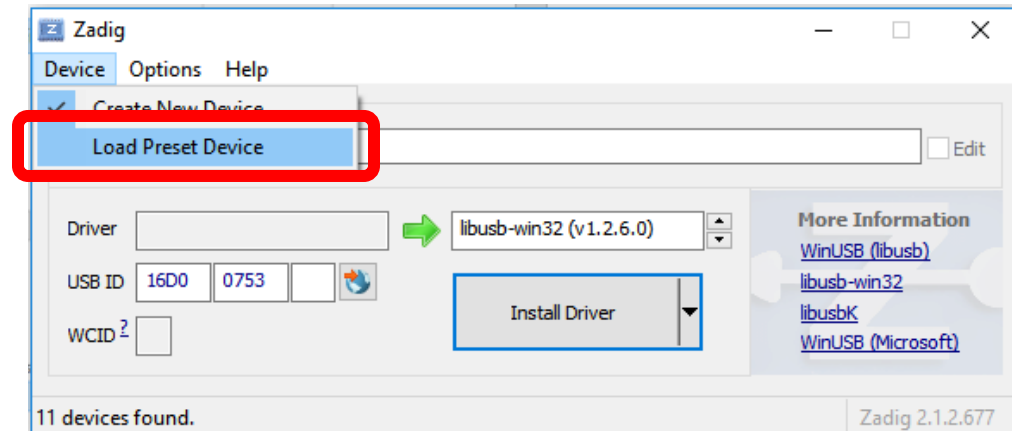
12. Navigate to the windows_driver_installer folder.



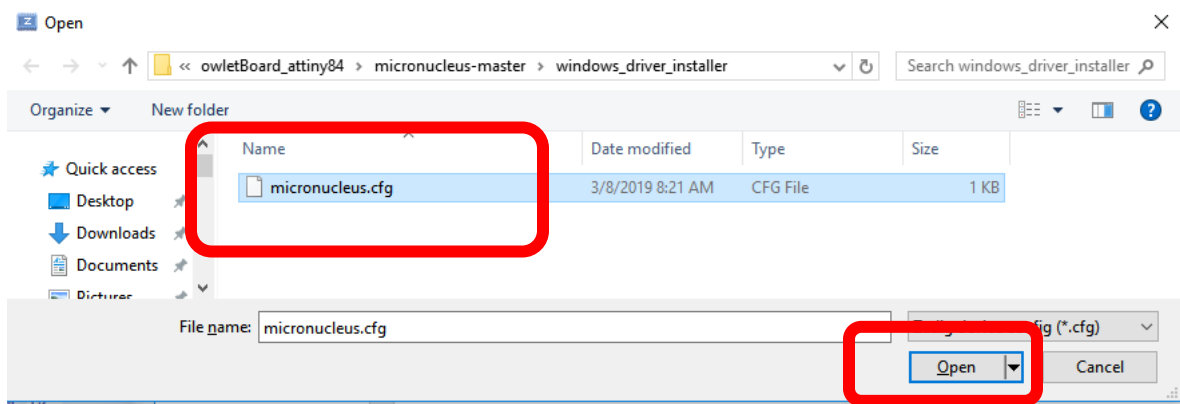
13. Double-click on zadig_2.1.2.exe. The window below should launch.



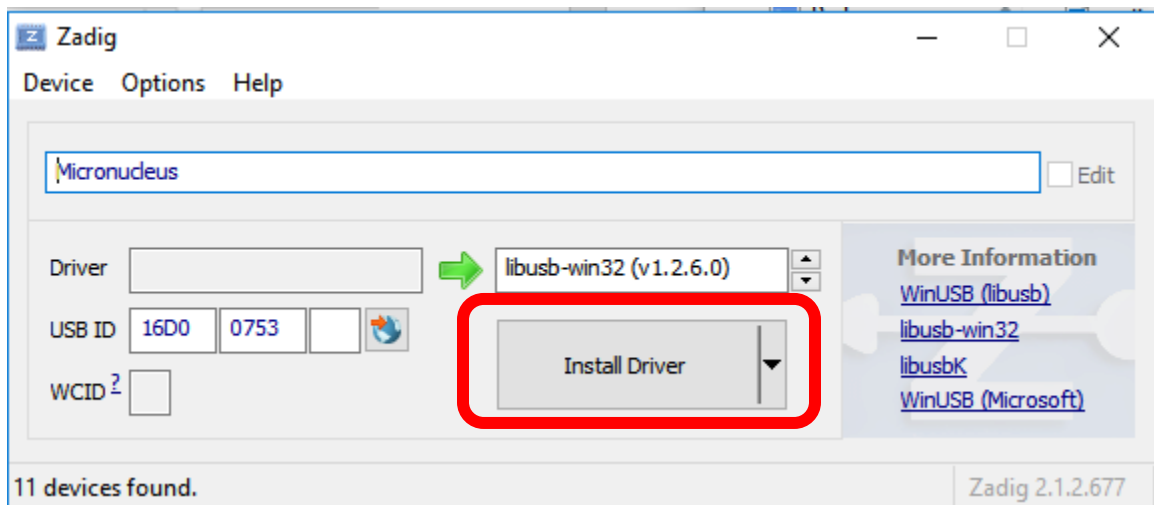
14. Click on Device > Load Preset Device



15. Select the micronucleus.cfg file.



16. Click Install Driver

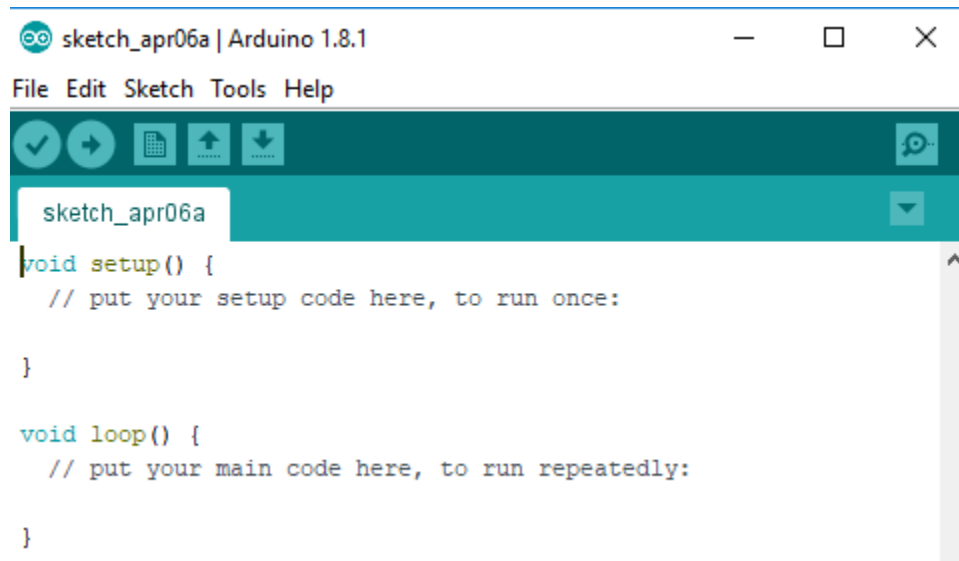


Congratulations.

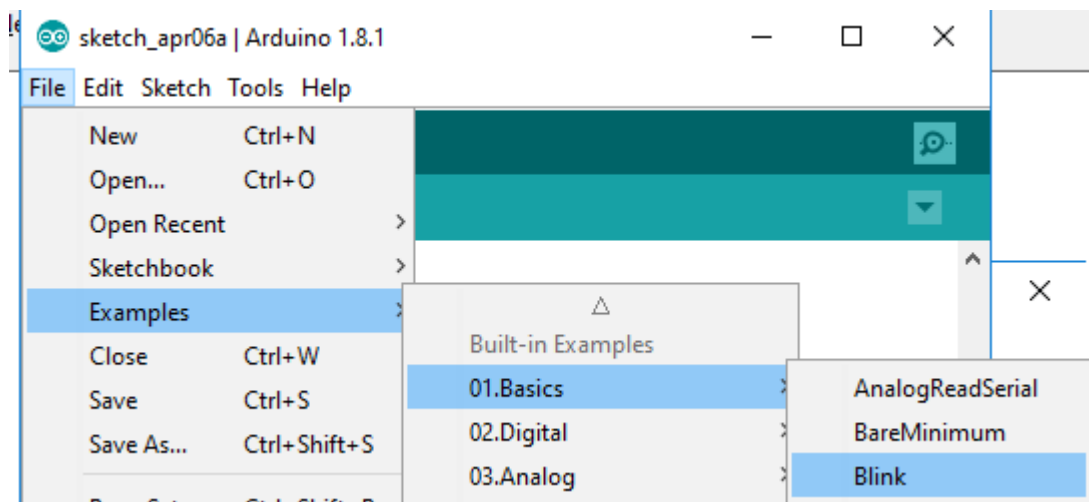
You have now installed all the software required to use the Owlboard Jr. or the Oregon shape board.

Tutorial 1: Uploading code to the board and blinking LED example

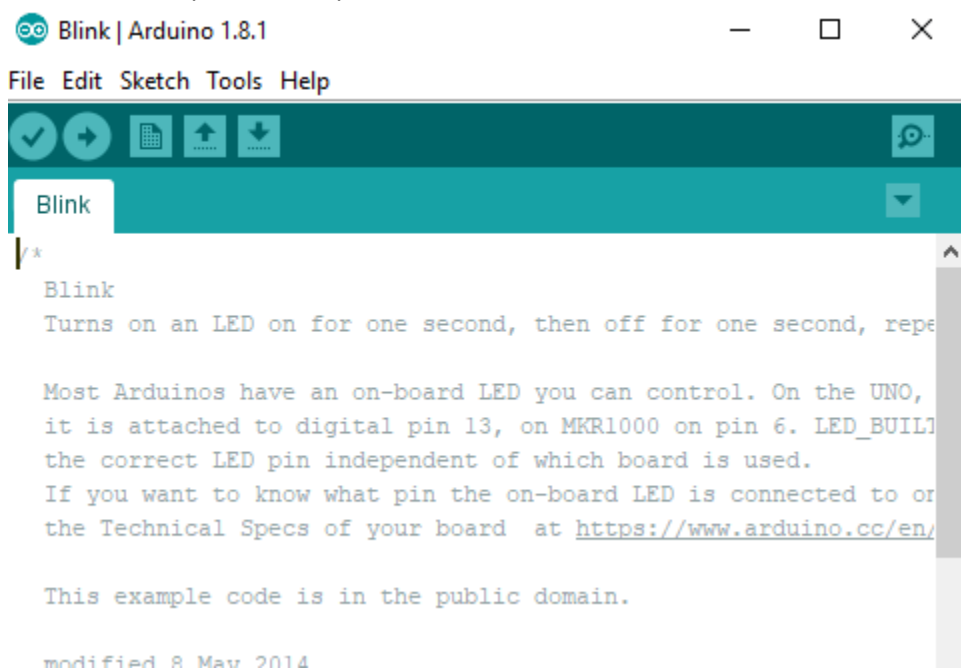
1. Make sure you have installed the driver, Arduino IDE, and installed the board file as listed in the previous section.
2. Launch Arduino IDE



3. We'll need something to program to the board, so navigate to the basic example.



- The basic example should open.



The screenshot shows the Arduino IDE interface with the 'Blink' example open. The title bar reads 'Blink | Arduino 1.8.1'. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below the menu bar is a toolbar with icons for checkmark, run, upload, and download. The main editor area shows the following text:

```
/*  
Blink  
Turns on an LED on for one second, then off for one second, repeats.  
  
Most Arduinos have an on-board LED you can control. On the UNO,  
it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN  
is the correct LED pin independent of which board is used.  
If you want to know what pin the on-board LED is connected to on  
your board, see the Technical Specs of your board at https://www.arduino.cc/en/  
  
This example code is in the public domain.  
  
modified 8 May 2014
```

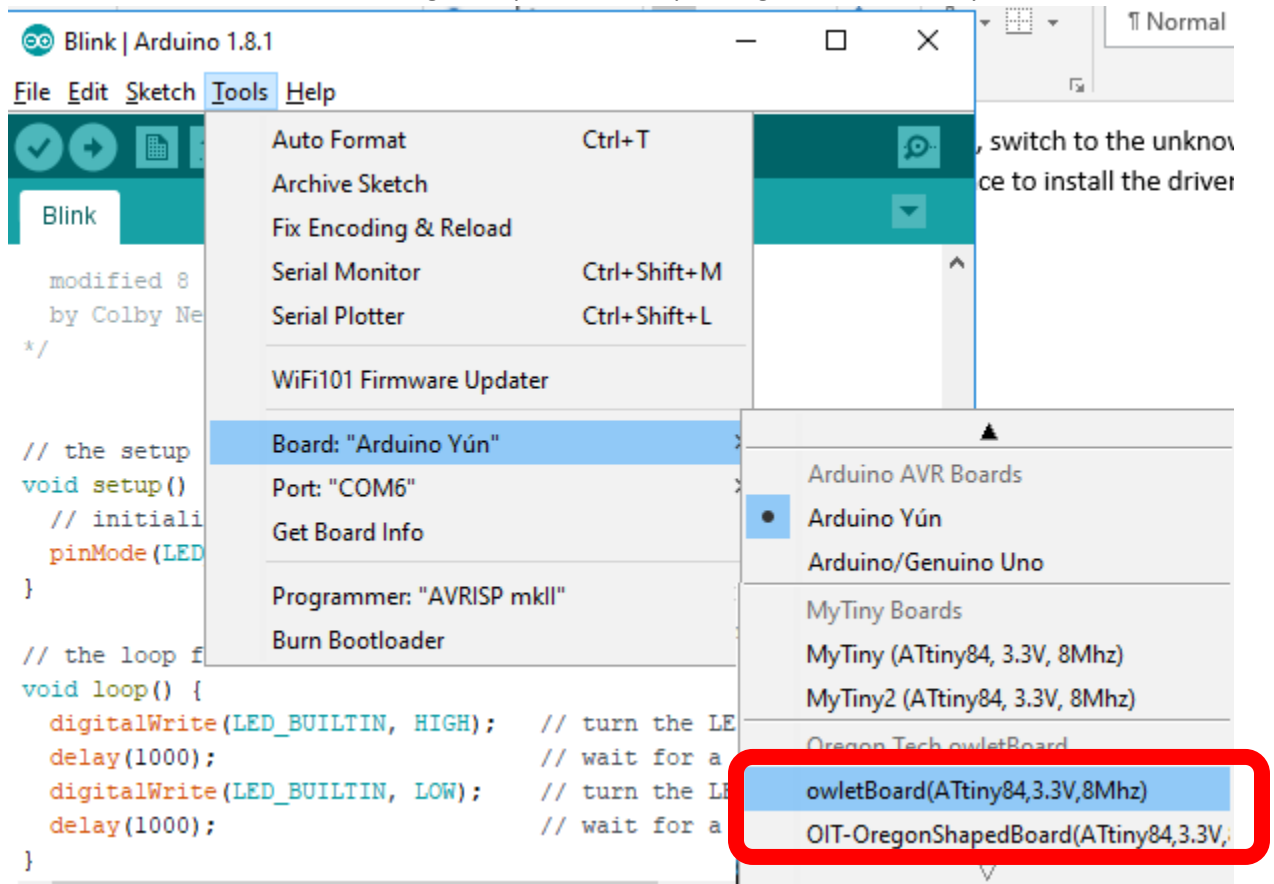
- The basic example needs to be modified, because one of the LEDs on the Owlboard Jr. or Oregon board is set to pin 0. Add the line of code shown below.



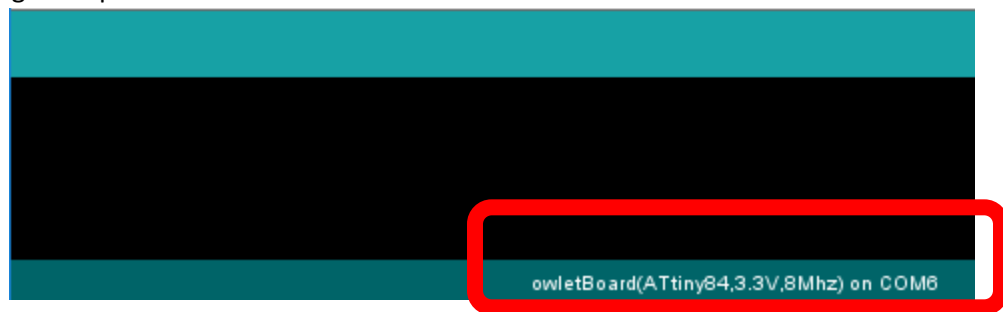
The screenshot shows the Arduino IDE interface with the modified 'Blink' example open. The title bar reads 'Blink | Arduino 1.8.1'. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below the menu bar is a toolbar with icons for checkmark, run, upload, and download. The main editor area shows the following text:

```
modified 8 Sep 2016  
by Colby Newman  
*/  
#define LED_BUILTIN 0  
  
// the setup function runs once when you press reset or power the  
void setup() {  
  // initialize digital pin LED_BUILTIN as an output.  
  pinMode(LED_BUILTIN, OUTPUT);  
}  
  
// the loop function runs over and over again forever  
void loop() {  
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is t
```

6. Click on the owletBoard or OIT-OregonShapedBoard, depending on which board you have.



7. Make sure that the bottom lower right corner indicates that the correct board, owletBoard or OIT-OregonShapedBoard is selected.



- Click the arrow button. This will compile and upload your code to the board. Note that it will ask you to plug in the device ('Running Digispark Uploader... Plug in device now...'). You must unplug and replug the device within 60 seconds or the system will timeout. If the timeout occurs, press the arrow button again, remembering plug in the board when asked.

```
Blink | Arduino 1.8.1
File Edit Sketch Tools Help
Blink
modified 8 Sep 2016
by Colby Newman
*/
#define LED_BUILTIN 13

// the setup function runs once when you press reset or power the
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the positive lead)
  delay(1000); // wait for a second
  digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the pin LOW
  delay(1000); // wait for a second
}

Uploading...
Global variables use 9 bytes of dynamic memory.
Running Digispark Uploader...
Plug in device now... (will timeout in 60 seconds)
22 owletBoard(ATTiny84,3.3V,8Mhz) on COM6
```

9. If it is successful you should see the following in the box below.



The image shows a terminal window with a dark background and light-colored text. The text is as follows:

```
Done uploading.  
> Starting the user app ...  
running: 100% complete  
>> Micronucleus done. Thank you!
```

The terminal window has a teal header bar at the top with the text "Done uploading." and a teal footer bar at the bottom with the text "22" on the left and "owletBoard(ATtiny84,3.3V,8Mhz) on COM6" on the right. A red rounded rectangle highlights the main terminal area containing the execution output.

10. On the Owlboard Jr, the right eye should now blink. On the Oregon shape board, the Portland Metro LED should now blink.