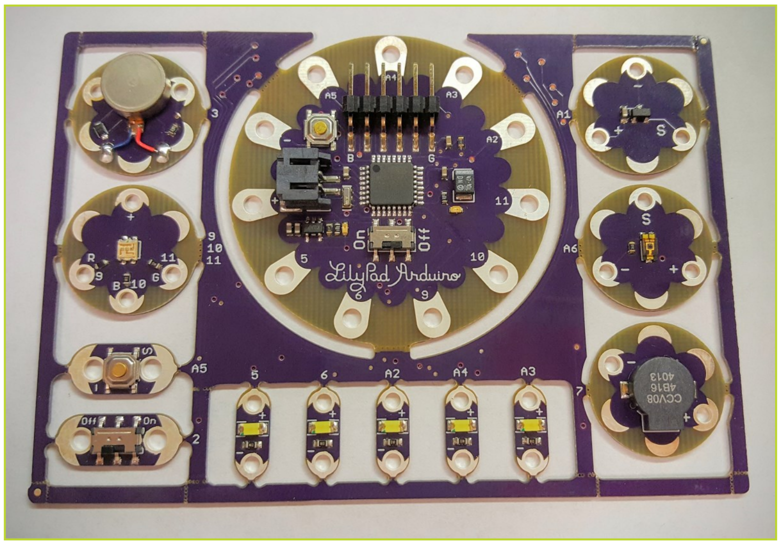
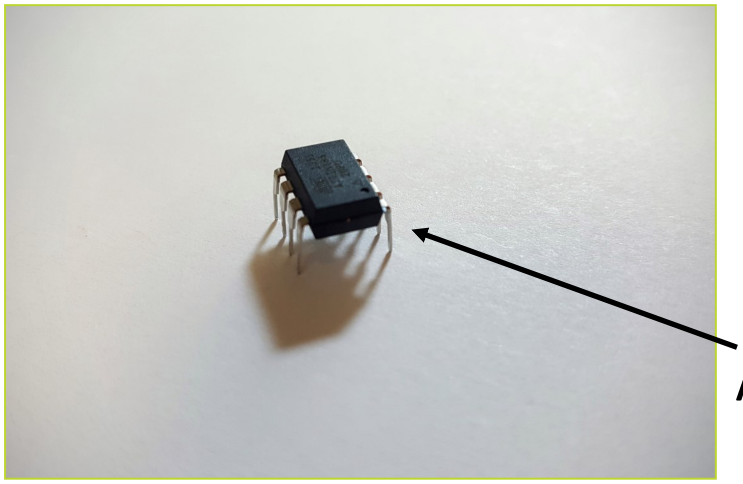
Programming in Arduino

CYBER RESILIENT ENERGY DELIVERY CONSORTIUM

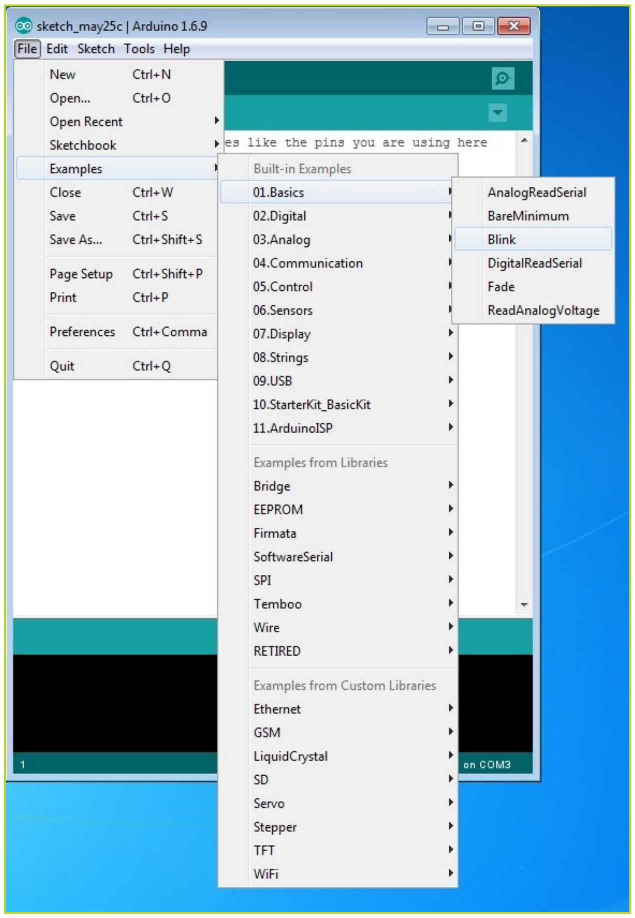
Arduino is a fun and powerful tool you can use to program a multitude of exciting projects, ranging from exciting light projects to personalizing clothing and other accessories with beautiful blinking lights! Not only is Arduino powerful, it is easy to learn with a little practice and curiosity!


LilyPad ProtoSnap Development Board


ATTiny85

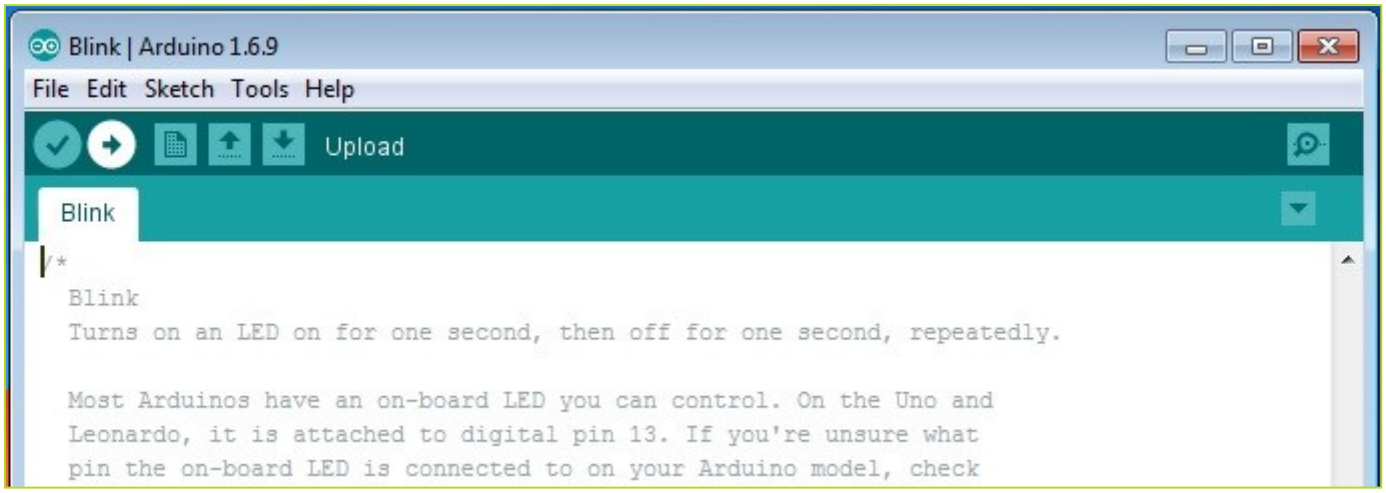
The Cyber Resilient Energy Delivery Consortium (CREDC) Education team continues the work of the TCIPG Education project. The team develops interactive lessons and activities designed to link researchers, educators, consumers, and students. The materials illustrate challenges, trade-offs, and decisions required for secure and economical power delivery. The project seeks to involve families learning together while creating interest in STEM disciplines and careers. The project website offers a variety of hands-on and virtual energy related activities and challenges. credc.mste.illinois.edu

Example—Blink!



In Arduino, go to File → Examples → Basics → Blink. Open it!

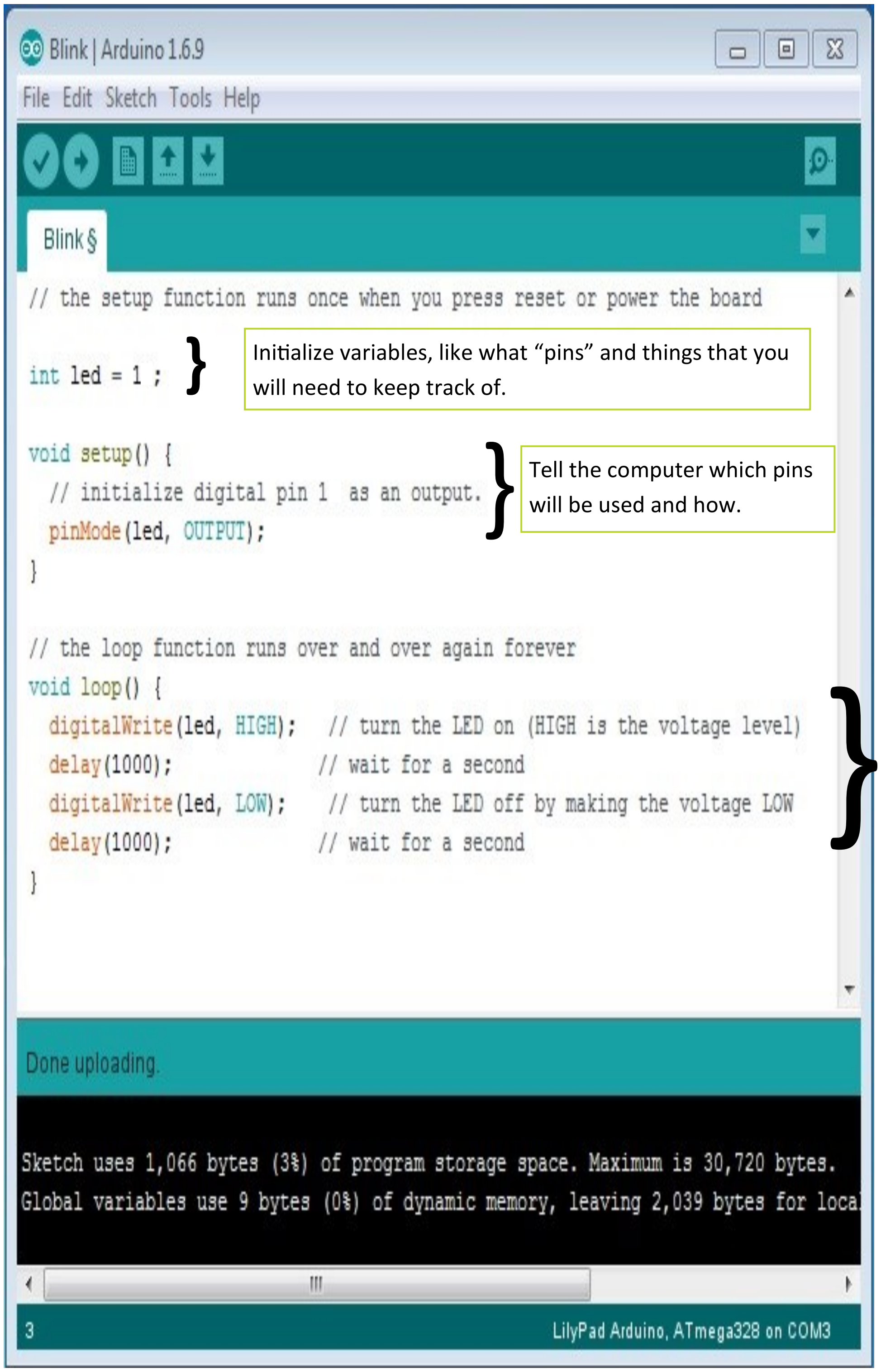
You will now see code for a simple program you can run! We'll talk about it more on the next page, but for now, click the "upload" arrow in the top left corner of the IDE and watch the program work! How would you describe what is happening to a friend? Could you do the same thing with a light switch in your home? If you wanted a friend to execute this code like your computer, what instructions would you give them?



```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  Most Arduinos have an on-board LED you can control. On the Uno and
  Leonardo, it is attached to digital pin 13. If you're unsure what
  pin the on-board LED is connected to on your Arduino model, check
```

Sooo… How does Blink work?

There are three basic chunks to every Arduino program, in this order: Variable Declaration section, Setup Section, and Loop Section. They are all very important to the creation of your program! Note that anytime you see "//" everything that follows is a comment, and won't be read by the computer.


Blink | Arduino 1.6.9
File Edit Sketch Tools Help
Blink §
Done uploading.
3
LilyPad Arduino, ATmega328 on COM3

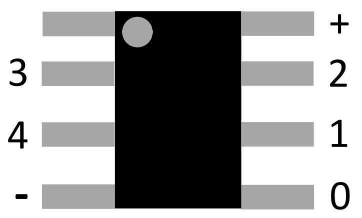
```
// the setup function runs once when you press reset or power the board

int led = 1 ;

void setup() {
  // initialize digital pin 1  as an output.
  pinMode(led, OUTPUT);
}

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

```
Sketch uses 1,066 bytes (3%) of program storage space. Maximum is 30,720 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2,039 bytes for loca
```

Initialize variables, like what "pins" and things that you will need to keep track of.

Tell the computer which pins will be used and how.

The actual instructions. The comments tell us exactly what is happening. The light turns on, wait, the light turns off, wait, repeat. This goes on as long as there is a charge.



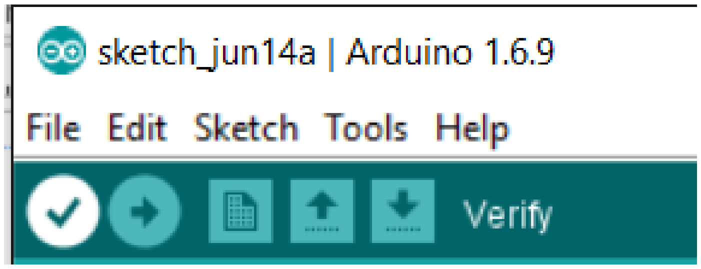
Now You Try!

Write some code that uses two LEDs and contains a different pattern depending on the LED. You will need to use two legs on the ATTiny85. Fill in the empty lines below.

```
int ledPin0 = 0; //First LED assigned to Pin 0
___________________    //Write code to assign another LED to pin 1

void setup() {
  // put your setup code here, to run once:
  pinMode(ledPin0, OUTPUT); //Sets pin 0 to output
  _________________________   //Write code to set pin 1 to output

}

void loop() {
  // put your main code here, to run repeatedly:
digitalWrite(LedPin0, HIGH); //Turns on LED
delay(1000);                 //Stays on for 1 second
_____________________________     //write code to turn off the LED
_____________________________    //and wait a second.


//Write your own code for your second LED below this line









}
```

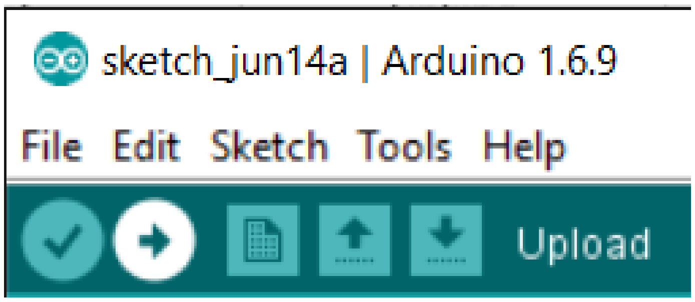
Compiling Your Code

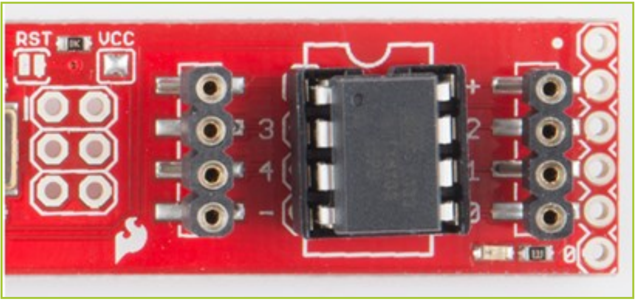
Now that you have written some of your own code, you need to compile and upload it to the ATtiny85.



In the top left corner of the Arduino IDE, you'll see the checkmark and arrow buttons. The checkmark, known as "Verify", is kind of like "spell check" for your code. You can use this to check if there are any mistakes in your code that will prevent it from working.

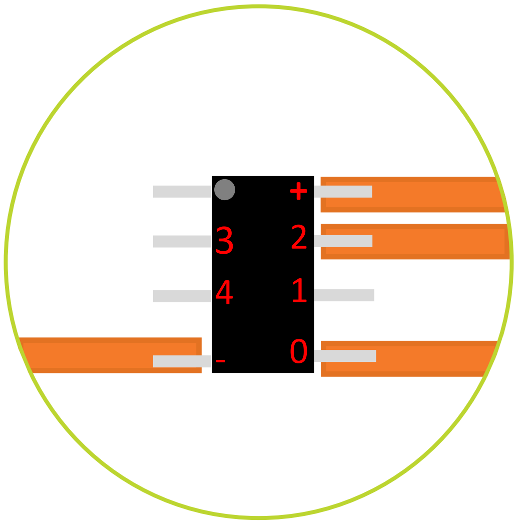
The Arrow button, known as "Upload", will compile the code and upload it to the device you have plugged into your machine. This is how we will use the ATtiny85! You first need to upload the code onto the microcontroller in order to run your program.

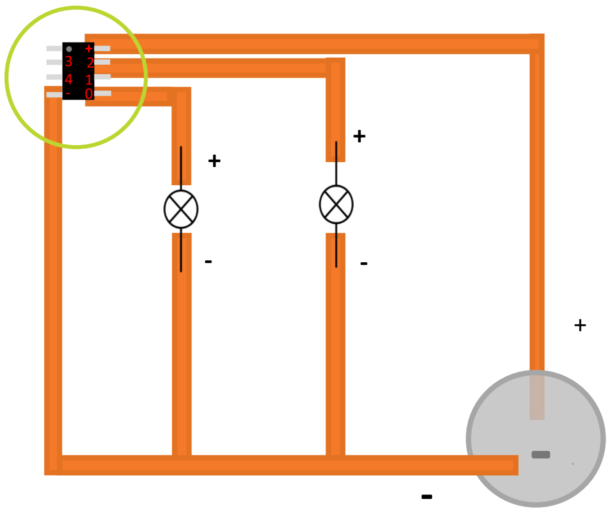


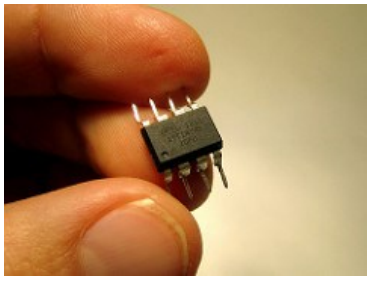


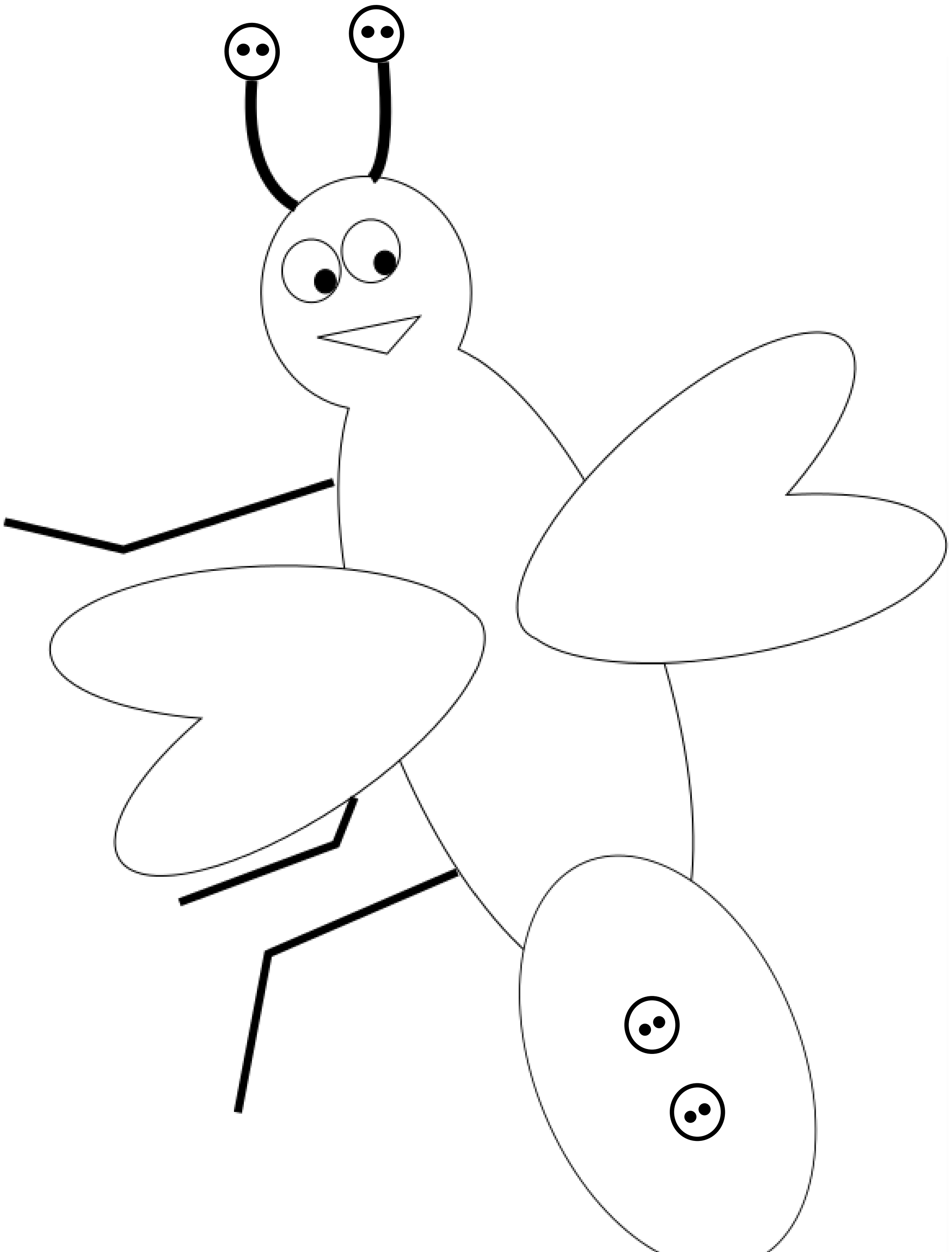
Plug the ATtiny85 into the AVR programmer as shown in the picture. The small "dot" in the top left corner of the ATtiny85 needs to be on the same side as the "notch" in the white outline. Plug the AVR into any USB port on your computer and you are ready to upload your code!

This material is based upon work supported by the Department of Energy under Award Number DE-OE0000780.



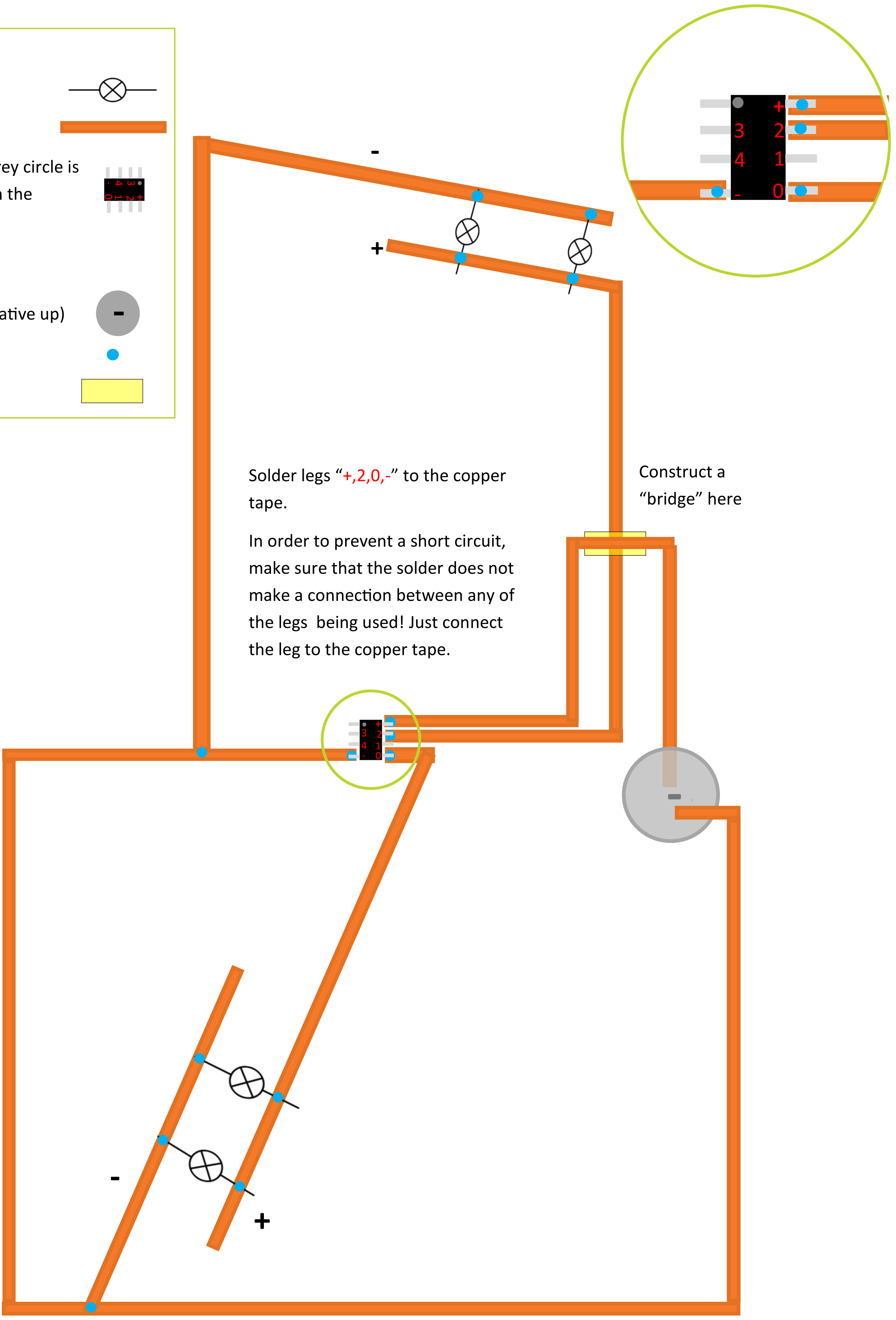



(Fold the legs of the ATtiny85 like this)



Key

LED	⊗
Tape Here	▬
ATTiny85 (grey circle is the notch on the corner)	▪
Battery (negative up)	●
Solder Here	●
Tape Bridge	▭



Solder legs “+,2,0,-” to the copper tape.

In order to prevent a short circuit, make sure that the solder does not make a connection between any of the legs being used! Just connect the leg to the copper tape.