

# ATtiny workshop

Computational Craft Week 7

Fall 2014

Many thanks to the [High Low Tech](#) group for all their work with the ATtinies.

# agenda

WHERE WE ARE GOING

## What's on for today:

PART 1: Programming ATtinies

PART 2: Midterm concept shareout

# computers

BIG >> SMALL



\$\$\$\$\$\$



\$\$\$

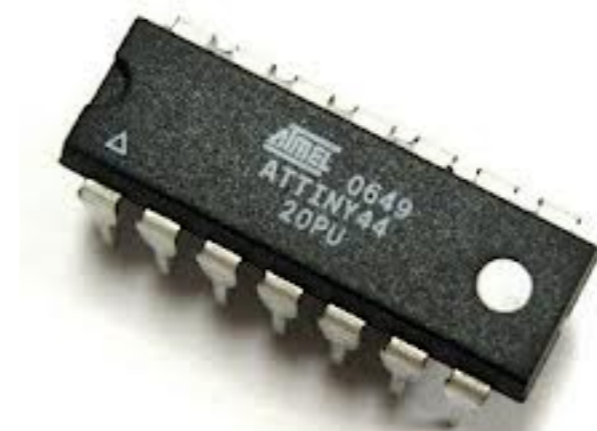


\$

# attiny

WHAT IS IT?

An **ATtiny** is a small, cheap (\$2-3) microcontroller that is amazing for running simple programs that you can program with the Arduino (vs. an expensive propriety ISP)





Arduino Uno  
32 KB Flash  
2 KB RAM



5 pins: 3 Analog in, 2 PWM

Smaller  
programs  
Can't store as  
much data



11 pins: 8 Analog in, 4 PWM

ATtiny 45  
4 KB Flash  
256 bytes RAM

ATtiny 85  
8 KB Flash  
512 bytes RAM

ATtiny 44  
4 KB Flash  
256 bytes RAM

ATtiny 84  
8 KB Flash  
512 bytes RAM

# attiny

WHAT CAN YOU DO WITH THEM?

Functions supported in Arduino IDE:



pinMode()  
digitalWrite()  
digitalRead()  
analogRead()  
analogWrite()  
shiftOut()

pulseIn()

millis()

micros()

delay()

delayMicroseconds()

SoftwareSerial (has been updated in Arduino 1.0)

# attiny

WHAT CAN YOU DO WITH THEM?



## Sound

//Halfway down. Doesn't support tone() library!

## Capacitance

## Servo

I2C Communication (Wire library) - a little unstable



**REMEMBER:** Libraries written for the 45/85 won't work for the 44/84.

# overview

WHAT WE ARE GOING TO DO

**Turn the Arduino into an “In System Programmer”**

//This means we can program the ATtiny with it

//There are other types of ISPs that are also great - if you want to work extensively with ATtinys, I suggest the USBtinyISP from Adafruit

**Program the ATtiny with the Arduino**

//We can write code in the Arduino IDE and upload it to the ATtiny through the Arduino.

//You can think of the Arduino as a translator.



# materials

## HARDWARE

Arduino Uno or Duemilanove (w/ an ATmega328, not an older board with an ATmega168)

ATtiny45

a 10 uF capacitor

breadboard

jumper wires

LED

3 volt battery

alligator clips

# materials

SOFTWARE

Arduino 1.0.5 (at least)

//This is important - previous versions were a tad buggy.

ATtiny core

//A library

step 1

# attiny core

LET'S TAKE CARE OF THIS FIRST

Follow this [link](#) or go to the link I posted on the blog.

//This is important - previous versions were a tad buggy.

# attiny core

LET'S TAKE CARE OF THIS FIRST

Follow this [link](#) or go to the link I posted on the blog.

//This is important - previous versions were a tad buggy.

Find the Arduino folder in your computer.

//All of your sketches live here. Usually in "Documents."

# attiny core

LET'S TAKE CARE OF THIS FIRST

Follow this [link](#) or go to the link I posted on the blog.

//This is important - previous versions were a tad buggy.

Find the Arduino folder in your computer.

//All of your sketches live here. Usually in "Documents."

Create a new folder and name it **hardware**.

//The lower case is important!

# attiny core

LET'S TAKE CARE OF THIS FIRST

Unzip the file you just downloaded.

# attiny core

LET'S TAKE CARE OF THIS FIRST

Unzip the file you just downloaded.

You should see a folder called **attiny** that contains **boards.txt** and a folder called **variants**.



# attiny core

LET'S TAKE CARE OF THIS FIRST

Unzip the file you just downloaded.

You should see a folder called **attiny** that contains **boards.txt** and a folder called **variants**.

Drag the **attiny** folder into the **hardware** folder you just created.

# attiny core

LET'S TAKE CARE OF THIS FIRST

Unzip the file you just downloaded.

You should see a folder called **attiny** that contains **boards.txt** and a folder called **variants**.

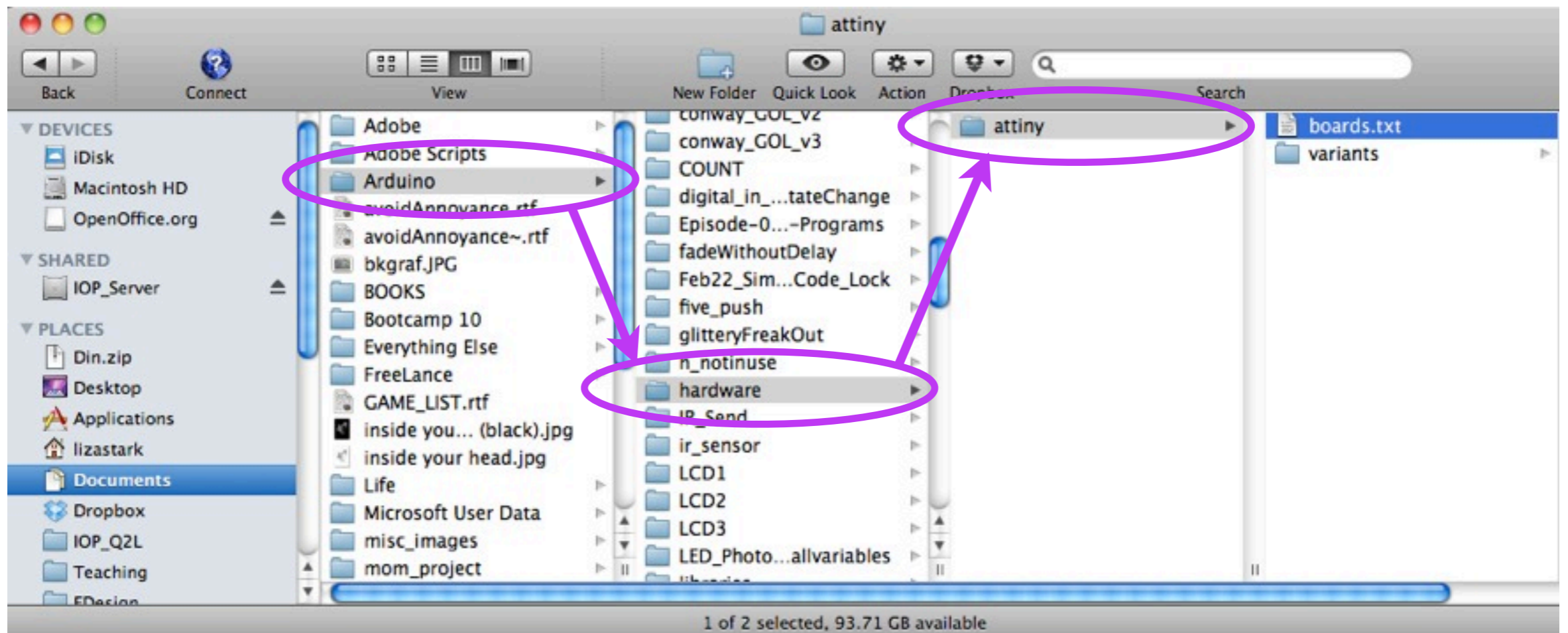
Drag the **attiny** folder into the **hardware** folder you just created.

If you had Arduino open, you must **restart** it.

# attiny core

LET'S TAKE CARE OF THIS FIRST

It should look like this:



step 2

# upload the sketch

MAKE THE ARDUINO AN ISP

**Open** Arduino and **connect** your Arduino to your computer.

# upload the sketch

MAKE THE ARDUINO AN ISP

Open Arduino and **connect** your Arduino to your computer.

Go into **Files > Examples > Arduino ISP**.

# upload the sketch

MAKE THE ARDUINO AN ISP

Open Arduino and **connect** your Arduino to your computer.

Go into **Files > Examples > Arduino ISP**.

Go into **Tools > Boards > (whichever board you are using)**

//You need to tell Arduino which board it is talking to.

//On another note, you should see a list of ATtiny boards to choose from. Raise your hand if you do not see them.

# upload the sketch

MAKE THE ARDUINO AN ISP

Open Arduino and **connect** your Arduino to your computer.

Go into **Files > Examples > Arduino ISP**.

Go into **Tools > Boards > (whichever board you are using)**

//You need to tell Arduino which board it is talking to.

//On another note, you should see a list of ATtiny boards to choose from. Raise your hand if you do not see them.

**Upload** the sketch to Arduino.



step 3

# the circuit

CONNECTING THE ATTINY TO THE ARDUINO

This is where you might make a **mistake** (if at all :)

//It can be difficult to see which wires go where.

Pay close attention to where you are putting your wires.

//We're gonna take this step by step.

First off, unplug your Arduino from your computer.

//We don't want a power supply right now.

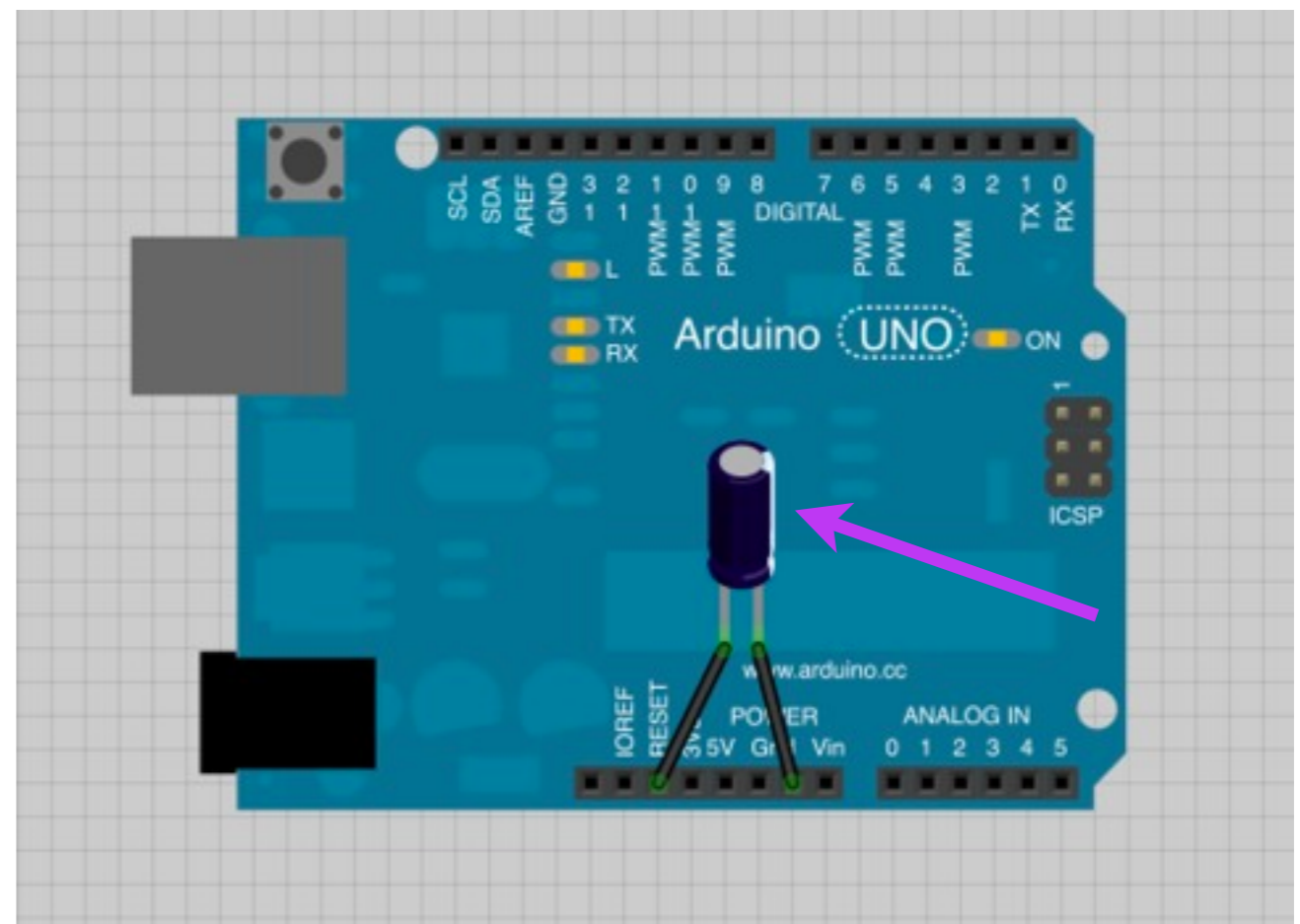
# breadboarding prep

CONNECTING THE ATTINY TO THE ARDUINO

Connect a 10  $\mu\text{F}$  electrolytic capacitor between GROUND and RESET.

The white strip should be on your RIGHT.

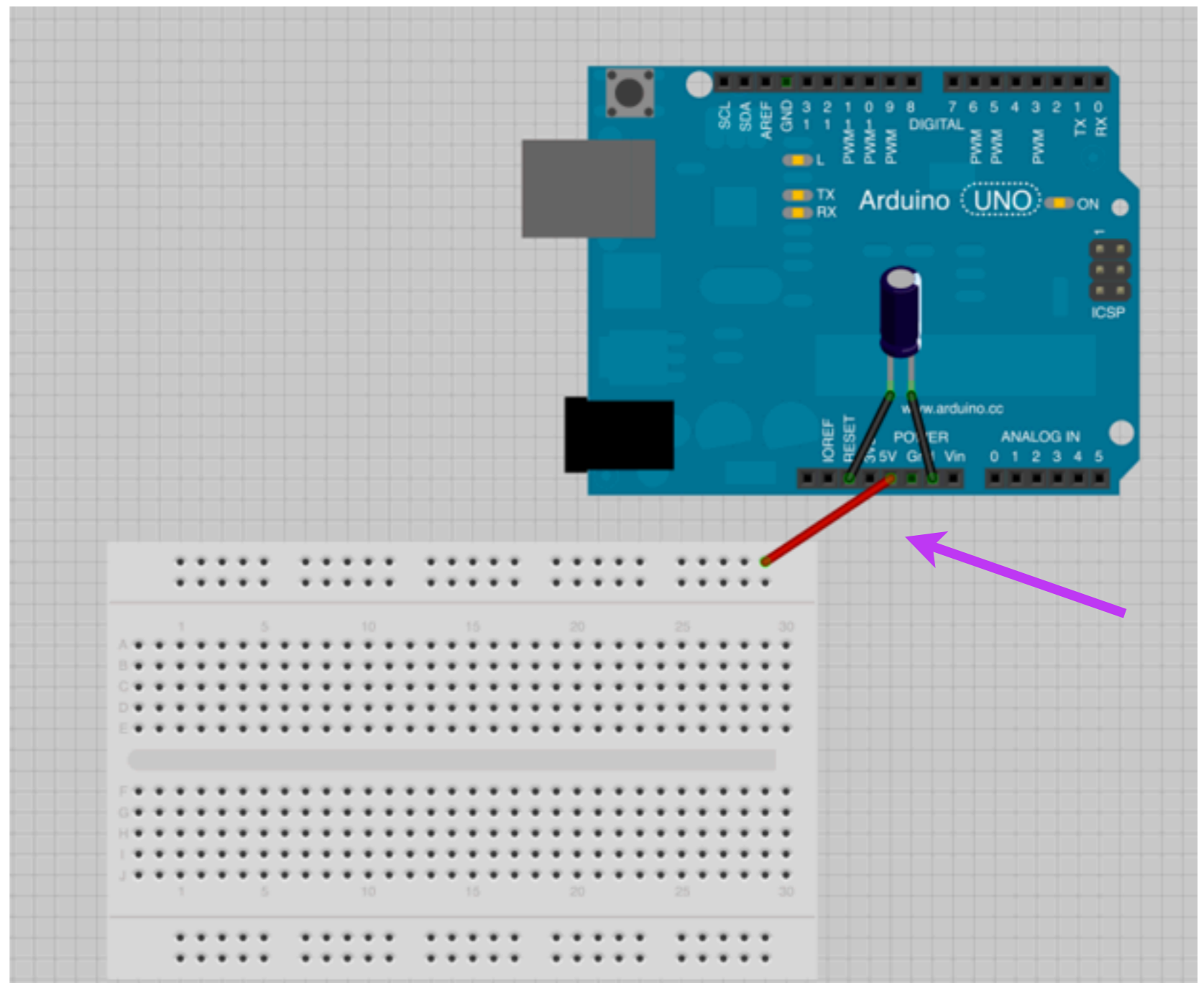
//Electricity can only flow one way through this capacitor, so this is important!



# breadboarding prep

CONNECTING THE ATTINY TO THE ARDUINO

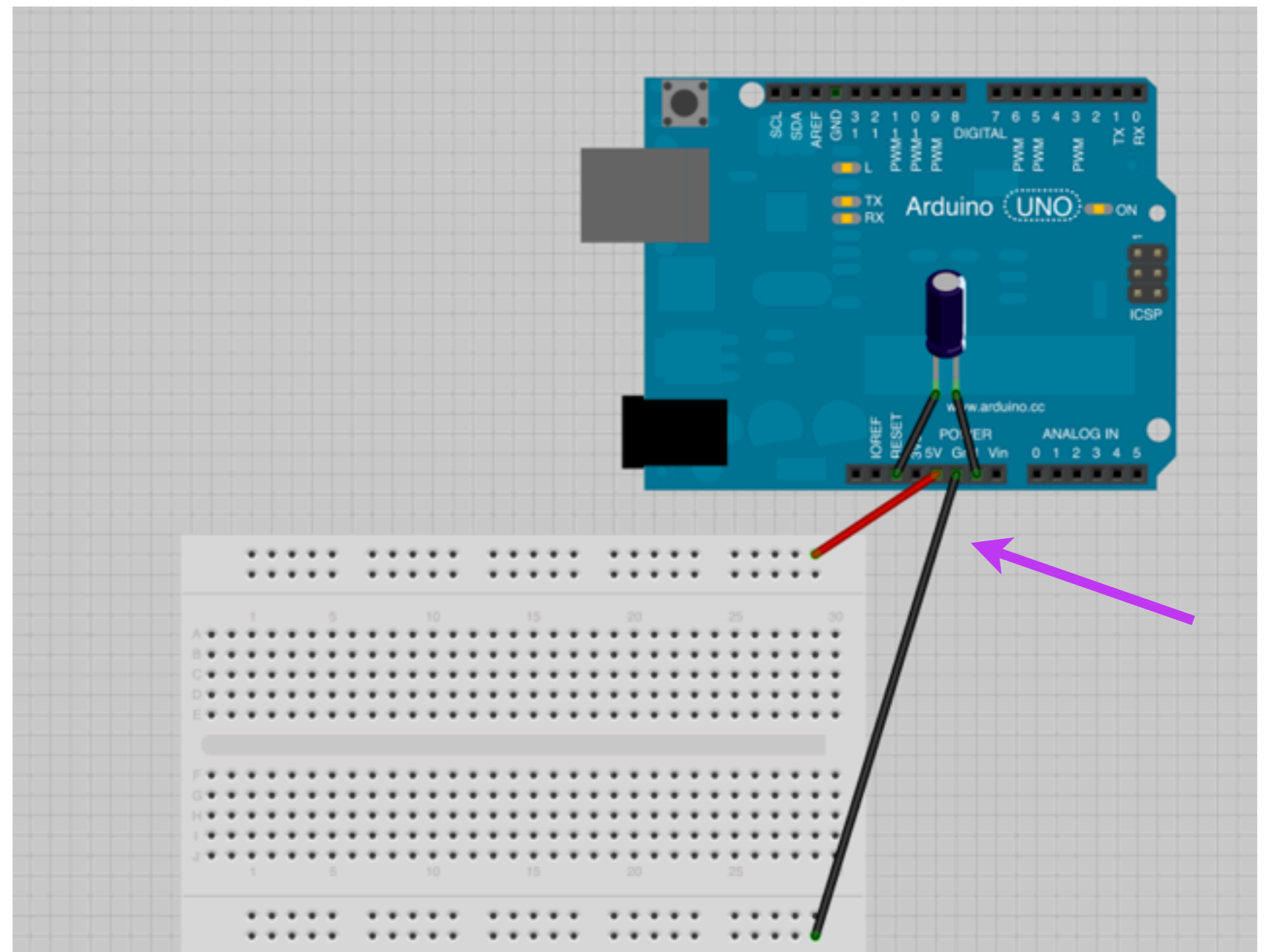
Connect one side of your breadboard to **Power**.



# breadboarding prep

CONNECTING THE ATTINY TO THE ARDUINO

And the other side to **Ground**.

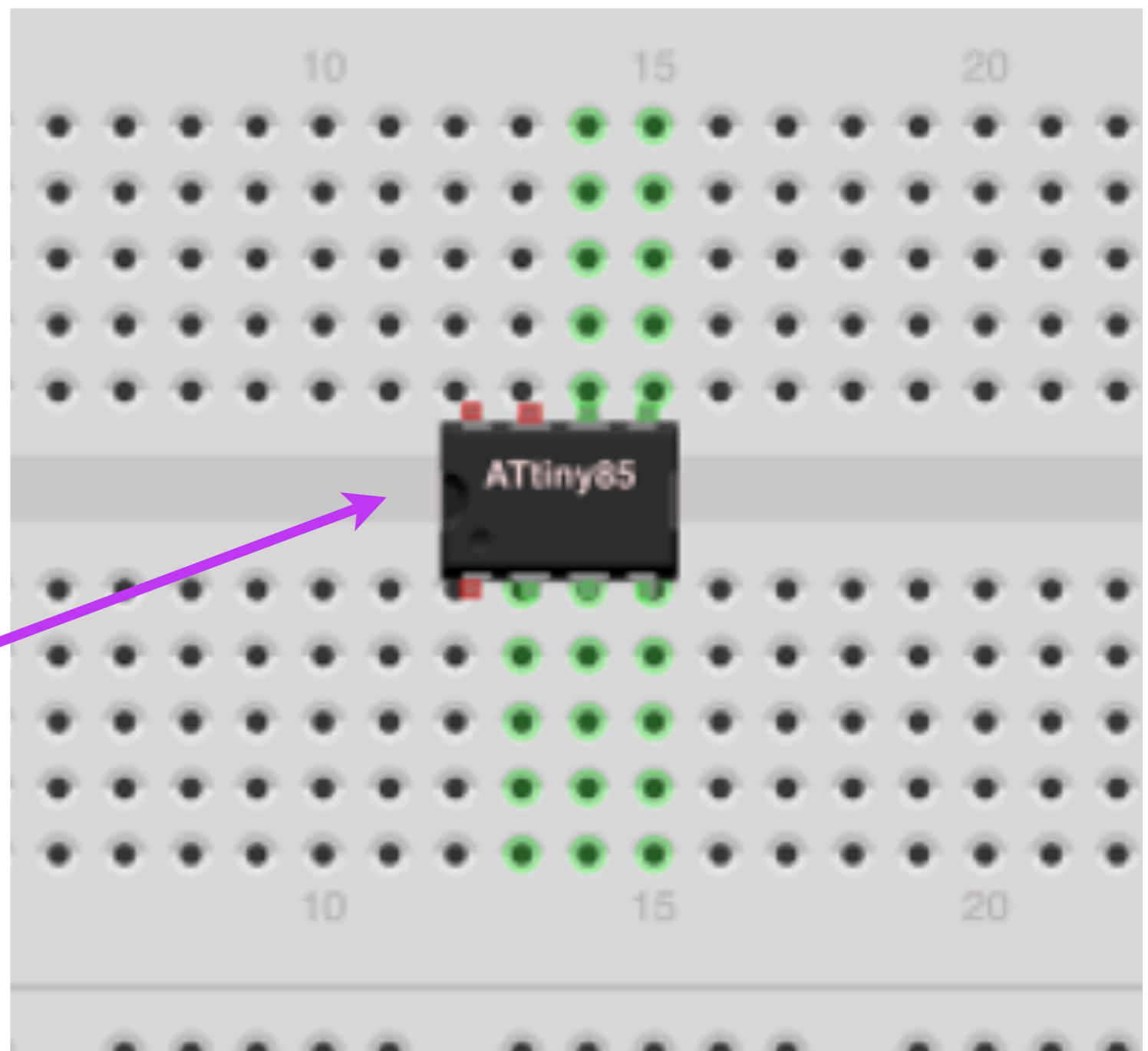


# breadboarding prep

CONNECTING THE ATTINY TO THE ARDUINO

Insert the ATtiny into your breadboard, **over** the bridge in the middle.

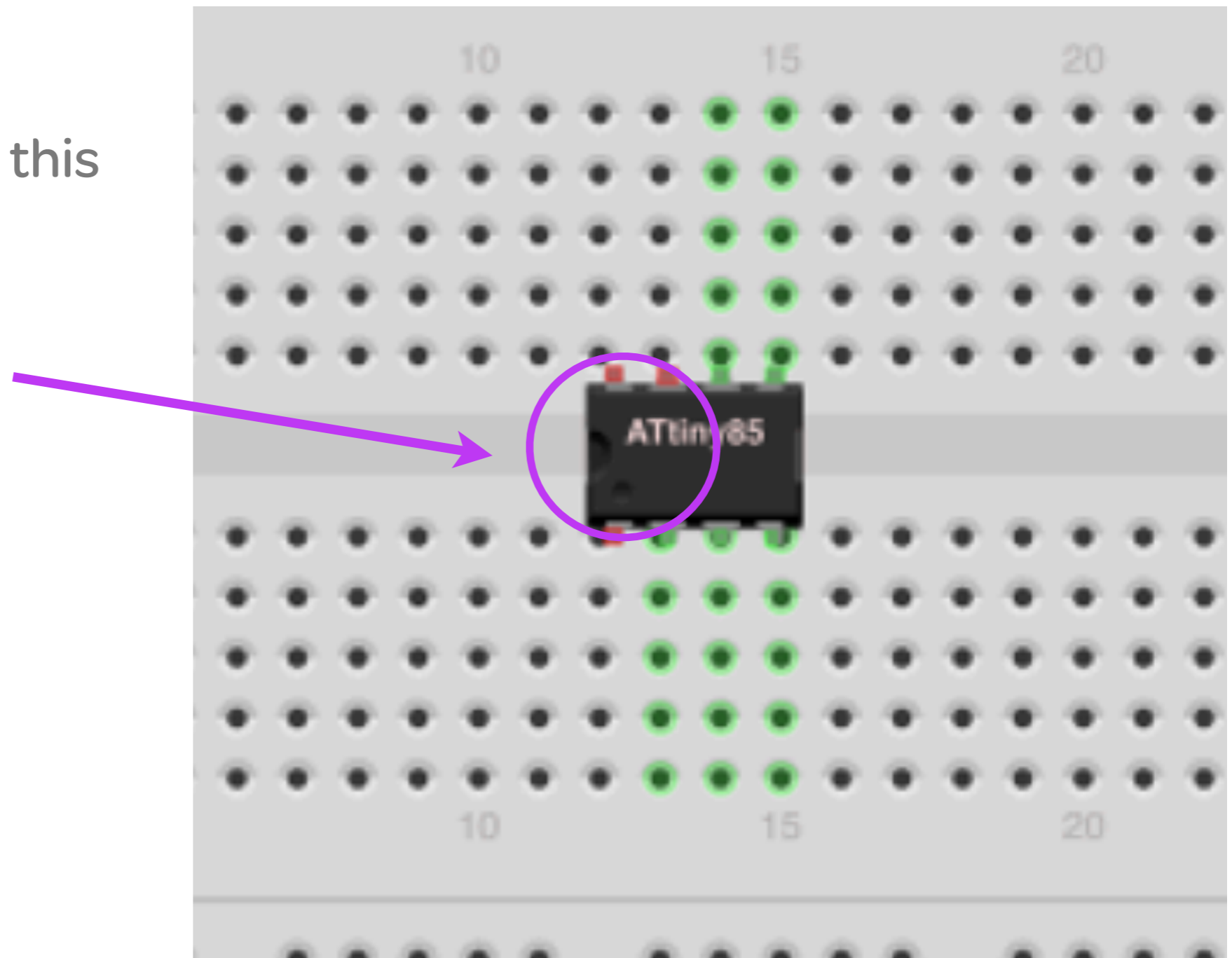
If you have an ATtiny44/84, it will have 14 pins and not 8.



# hookin it up

CONNECTING THE ATTINY TO THE ARDUINO

Notice the circle on this circuit?

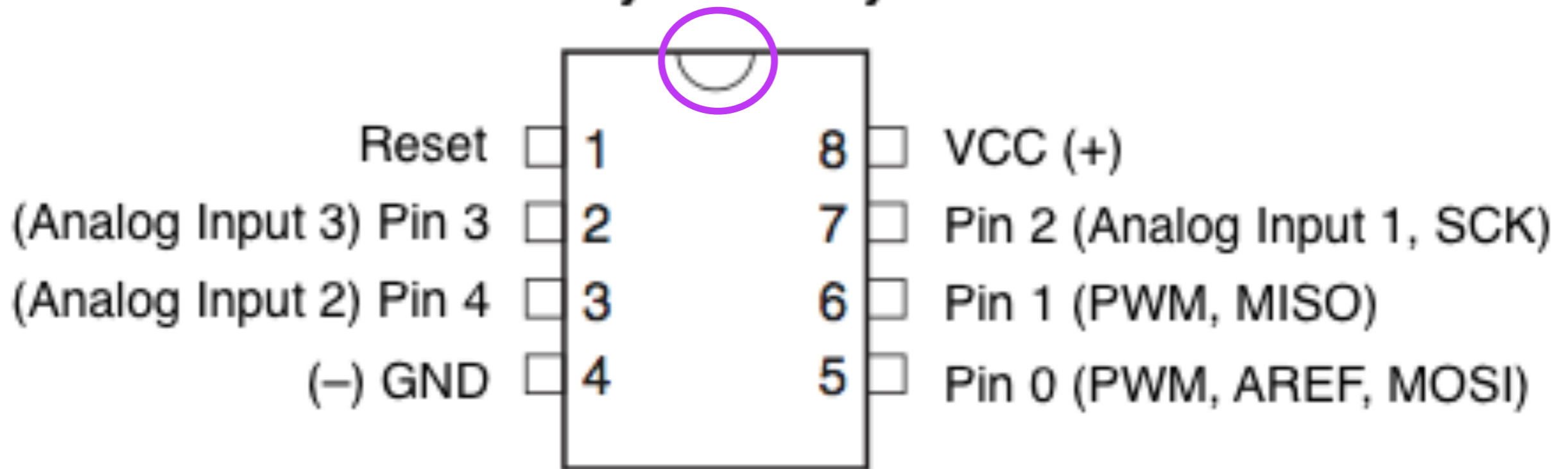


# hookin it up

CONNECTING THE ATTINY TO THE ARDUINO

This is how you know which side is which.

## ATTiny45 / ATTiny85

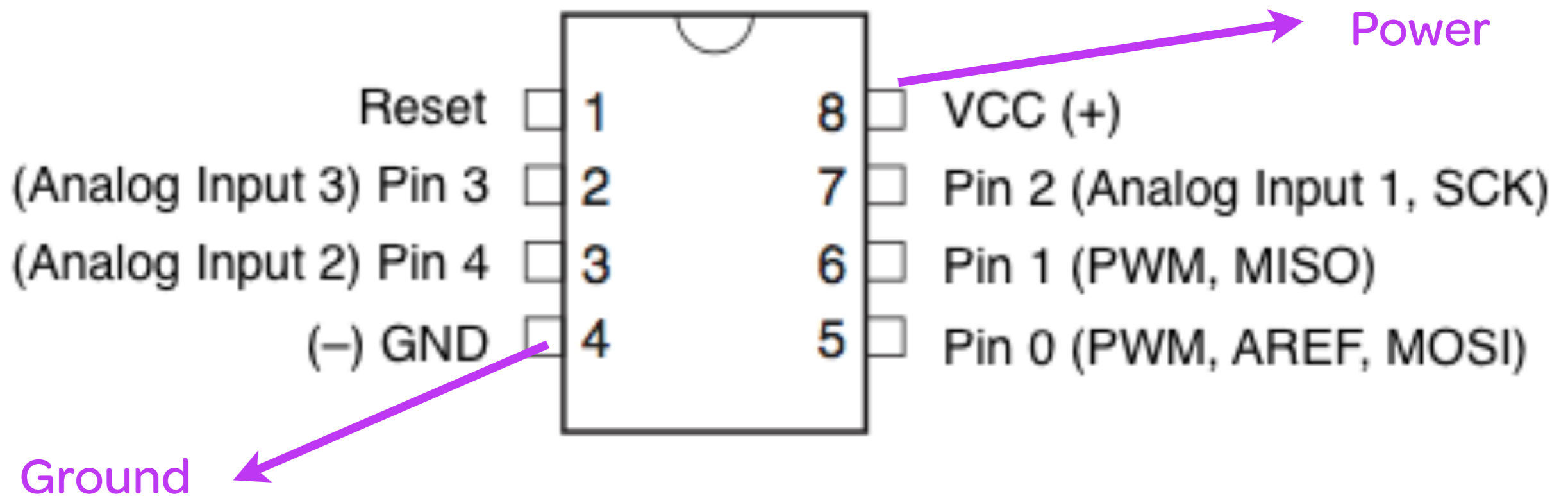


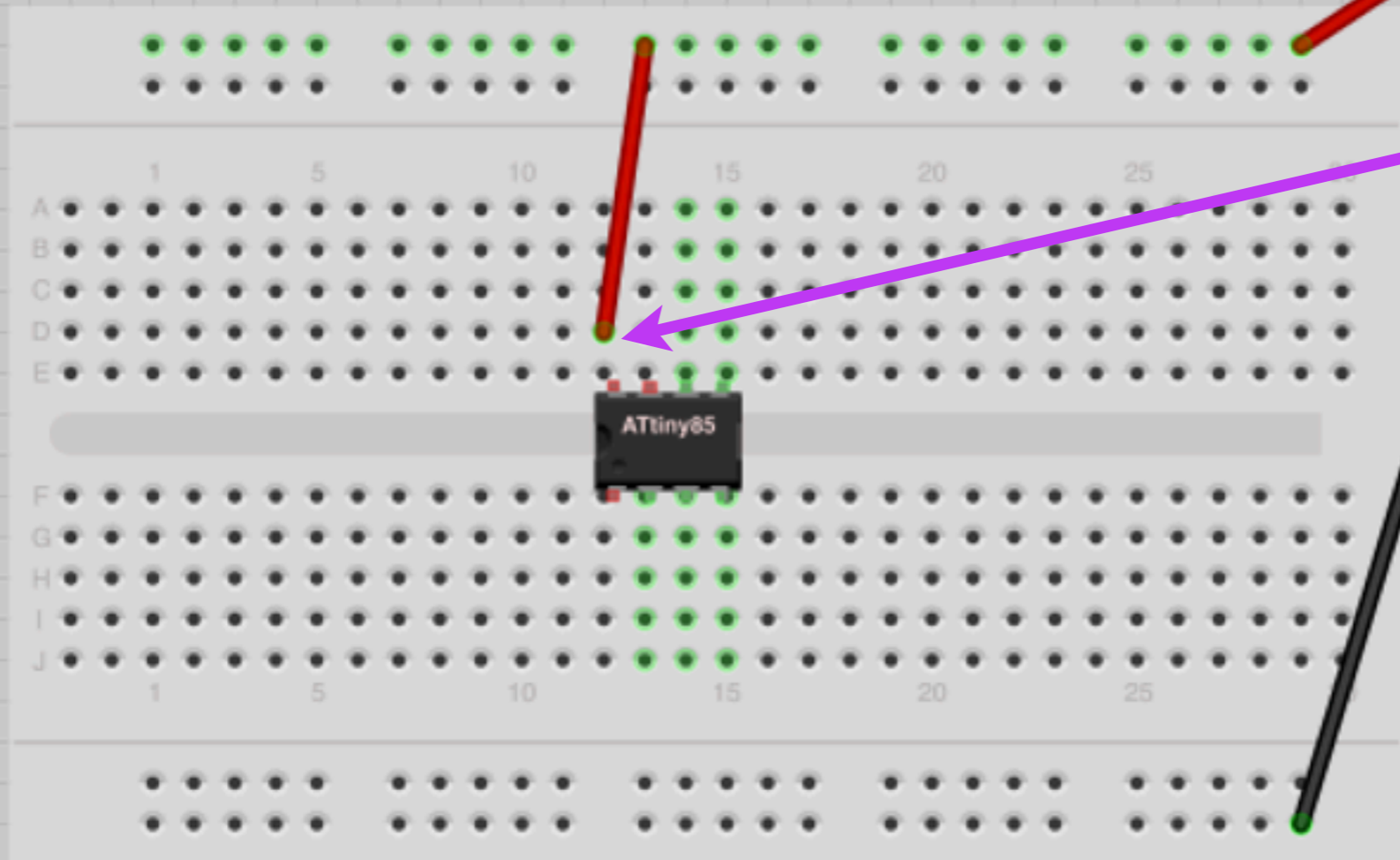


# hookin it up: attiny45/85

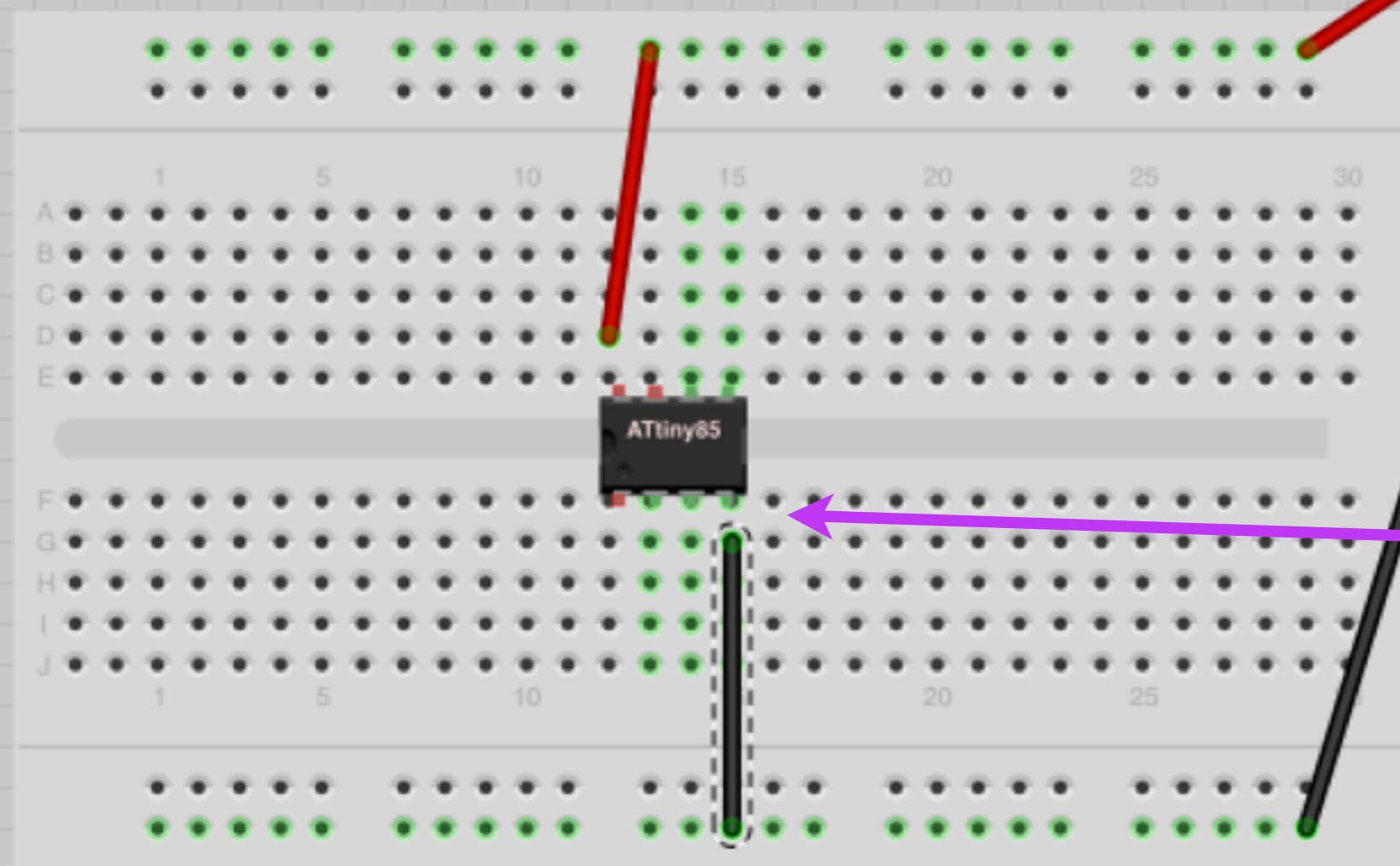
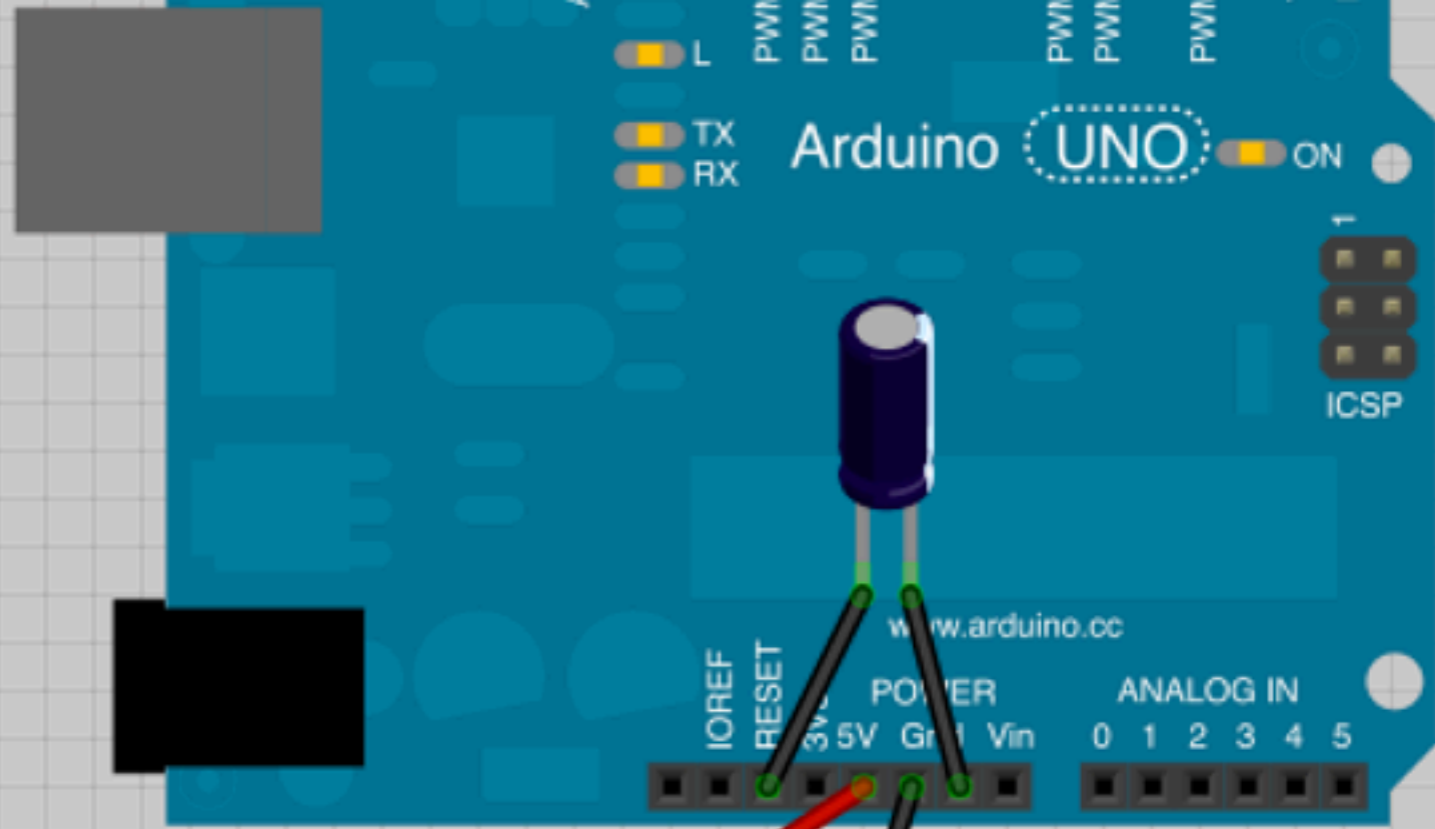
CONNECTING THE ATTINY TO THE ARDUINO

## ATTiny45 / ATTiny85





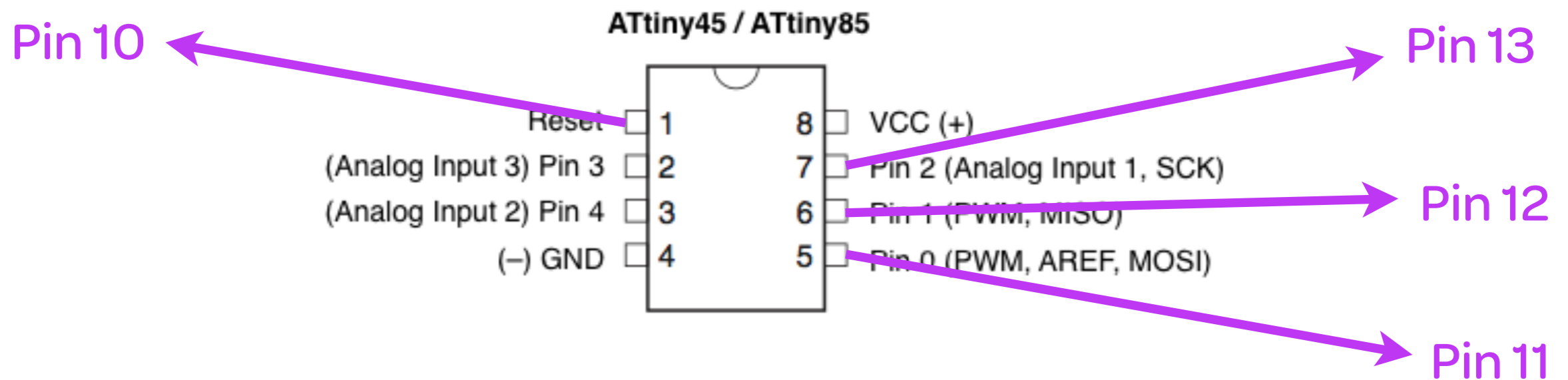
Power

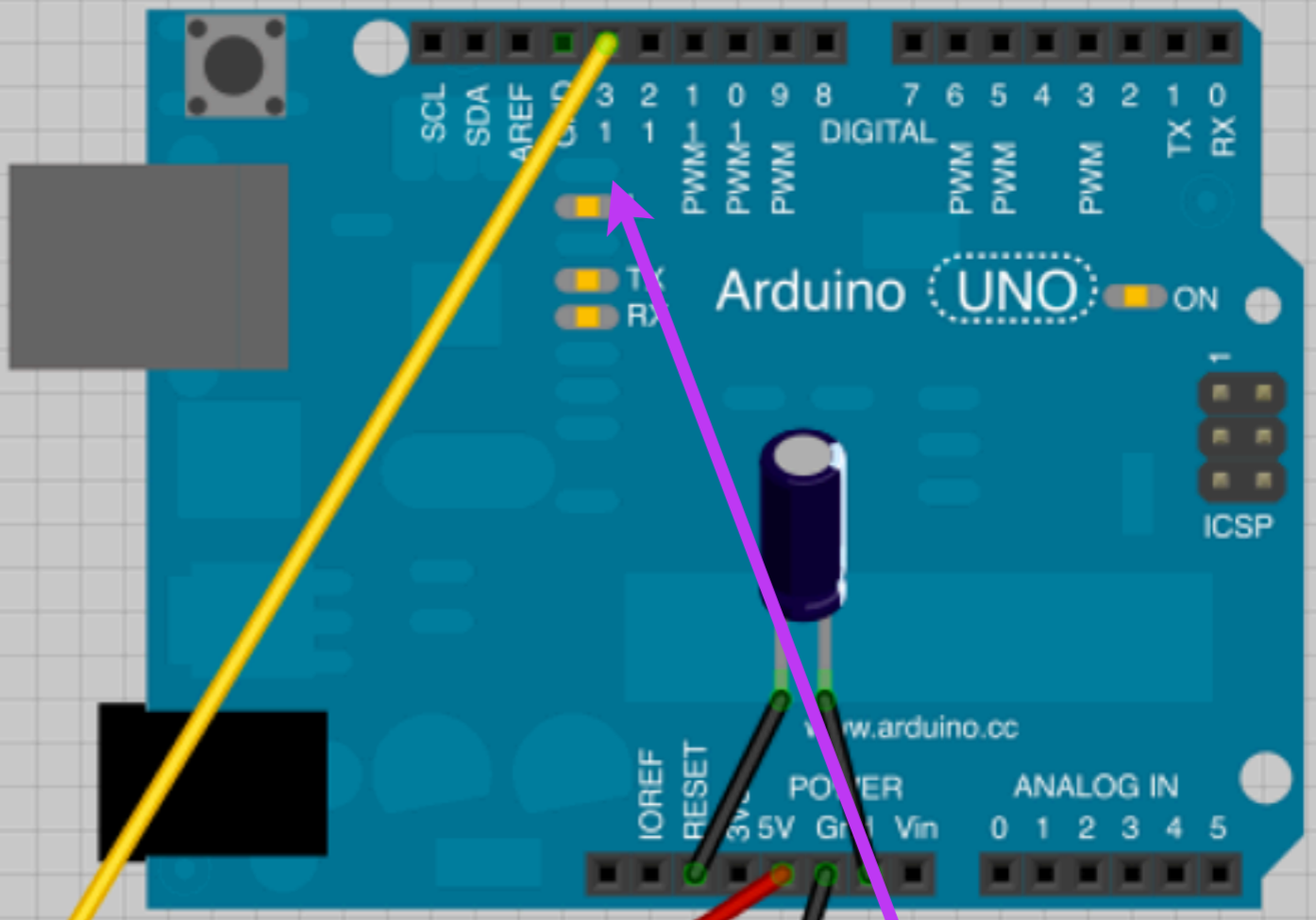


Ground

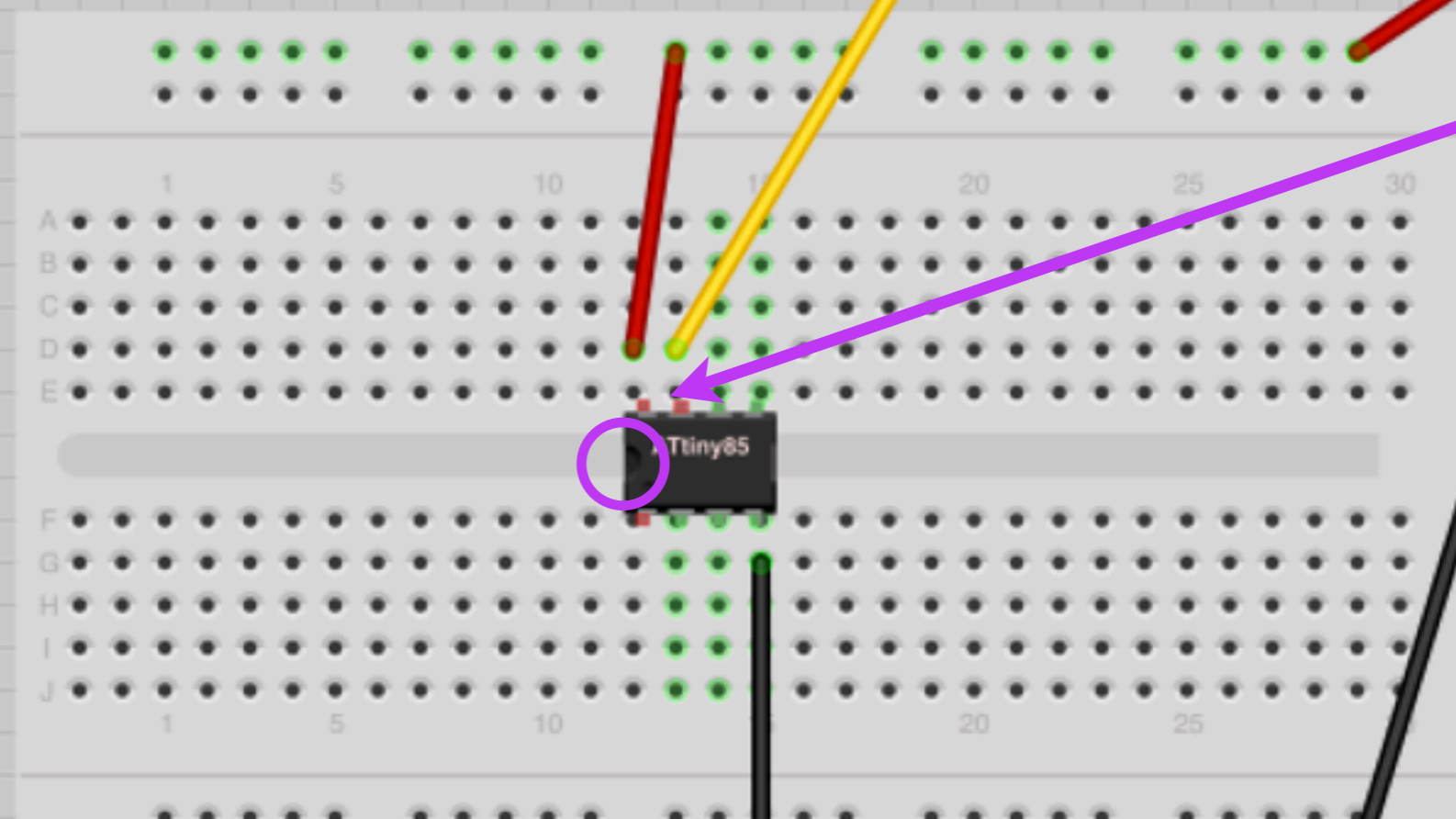
# hookin it up: attiny45/85

CONNECTING THE ATTINY TO THE ARDUINO

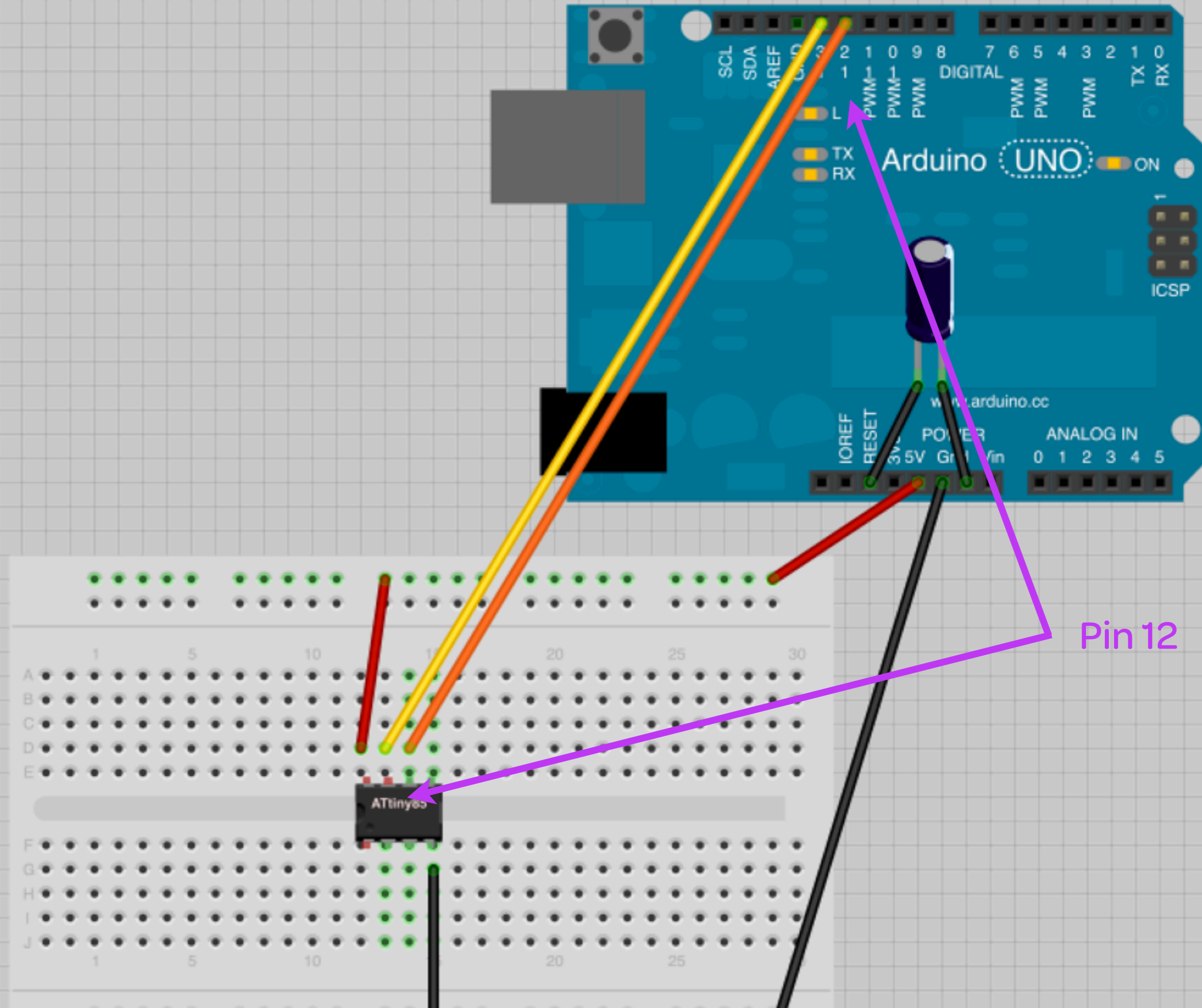


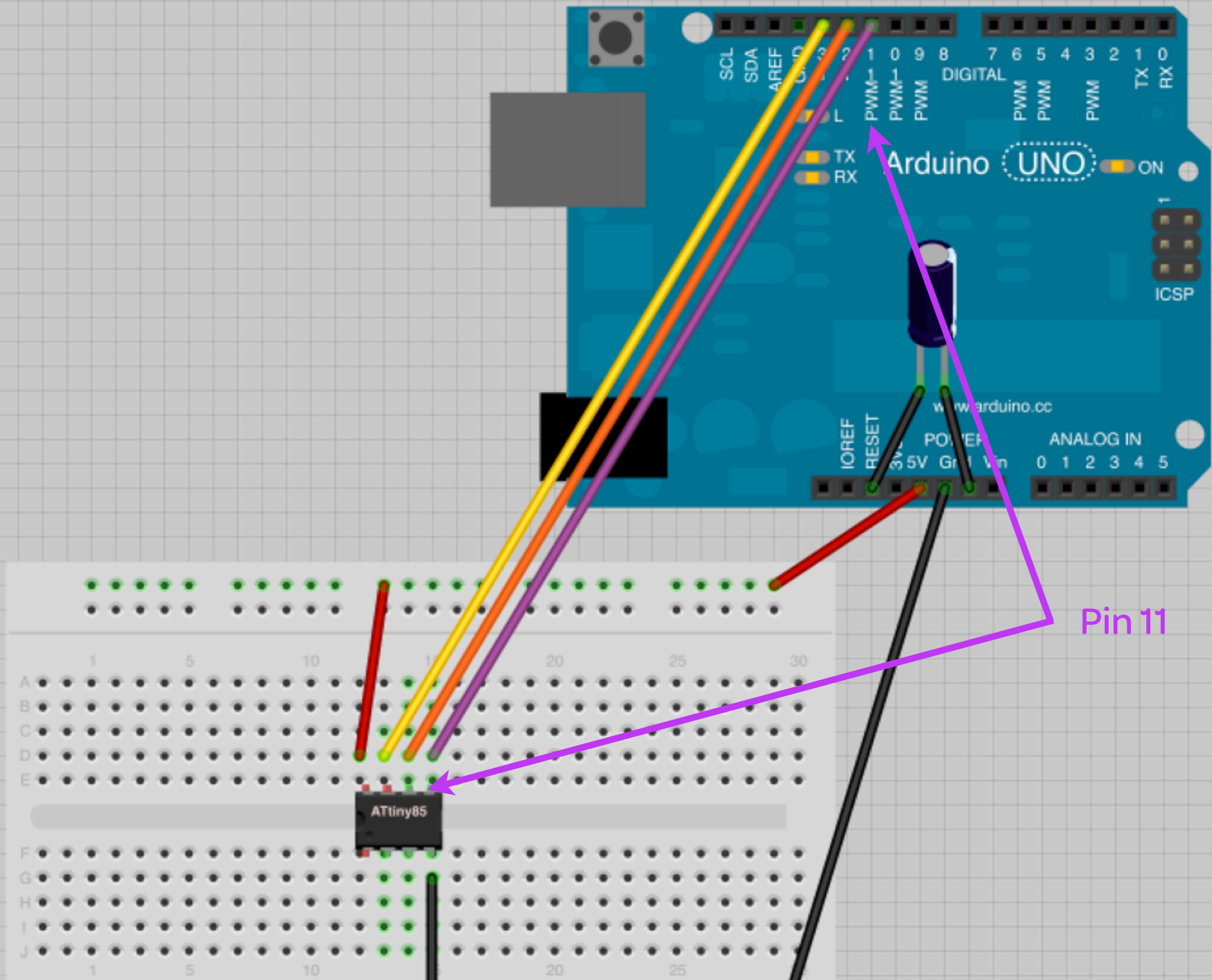


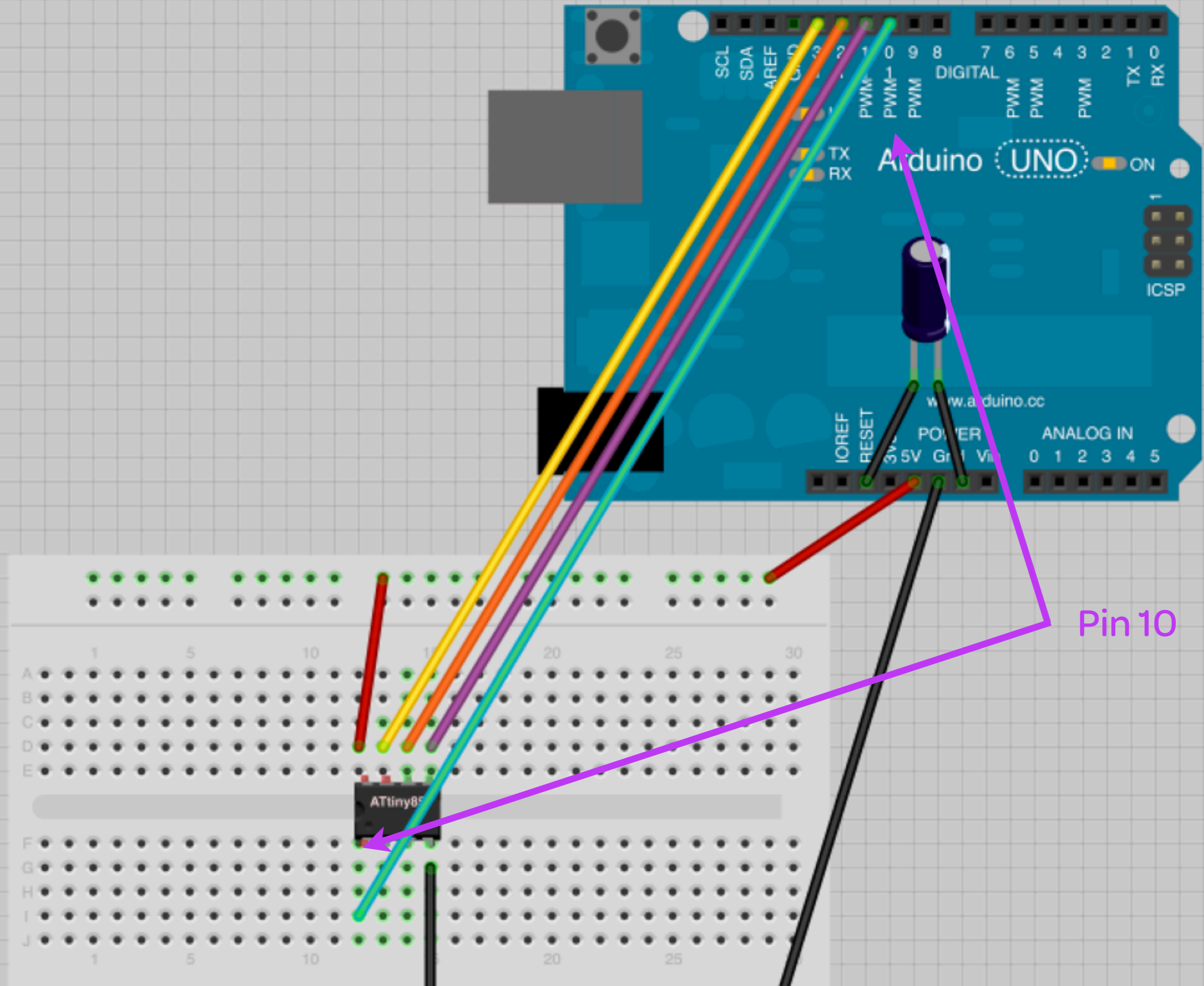
Pin 13



Ttiny85







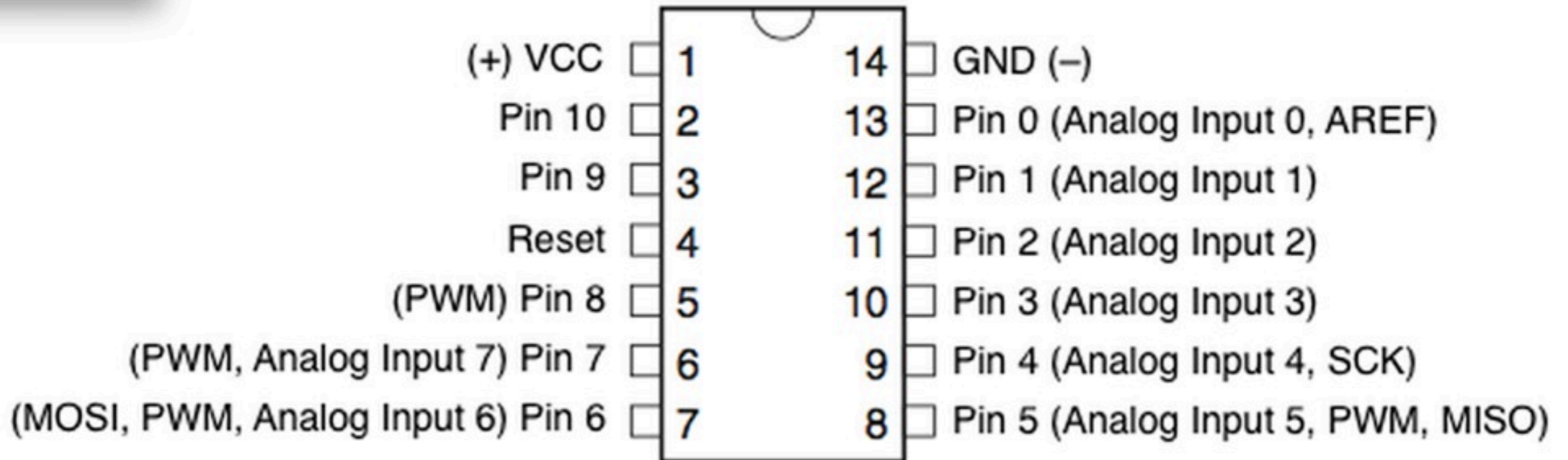


# pinout: attiny44/84

FOR REFERENCE

ATtiny44-84

## ATtiny44 / ATtiny84



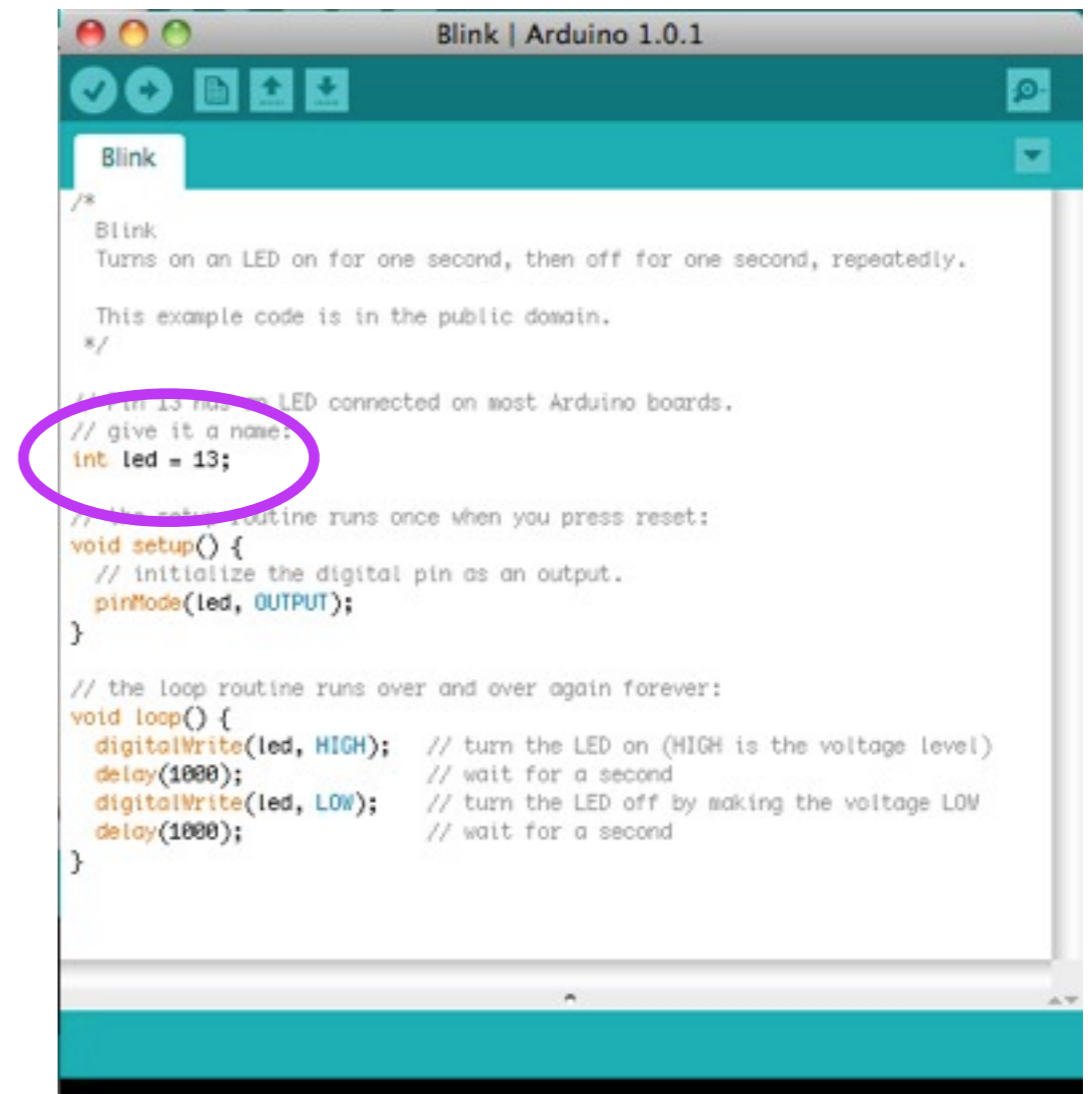
step 4

# open the sketch

PROGRAMMING THE ATTINY

Go into **Files** > **Examples** > **.01Basics** > **Blink**.

Change **int led = 13** to **int led = 3**.

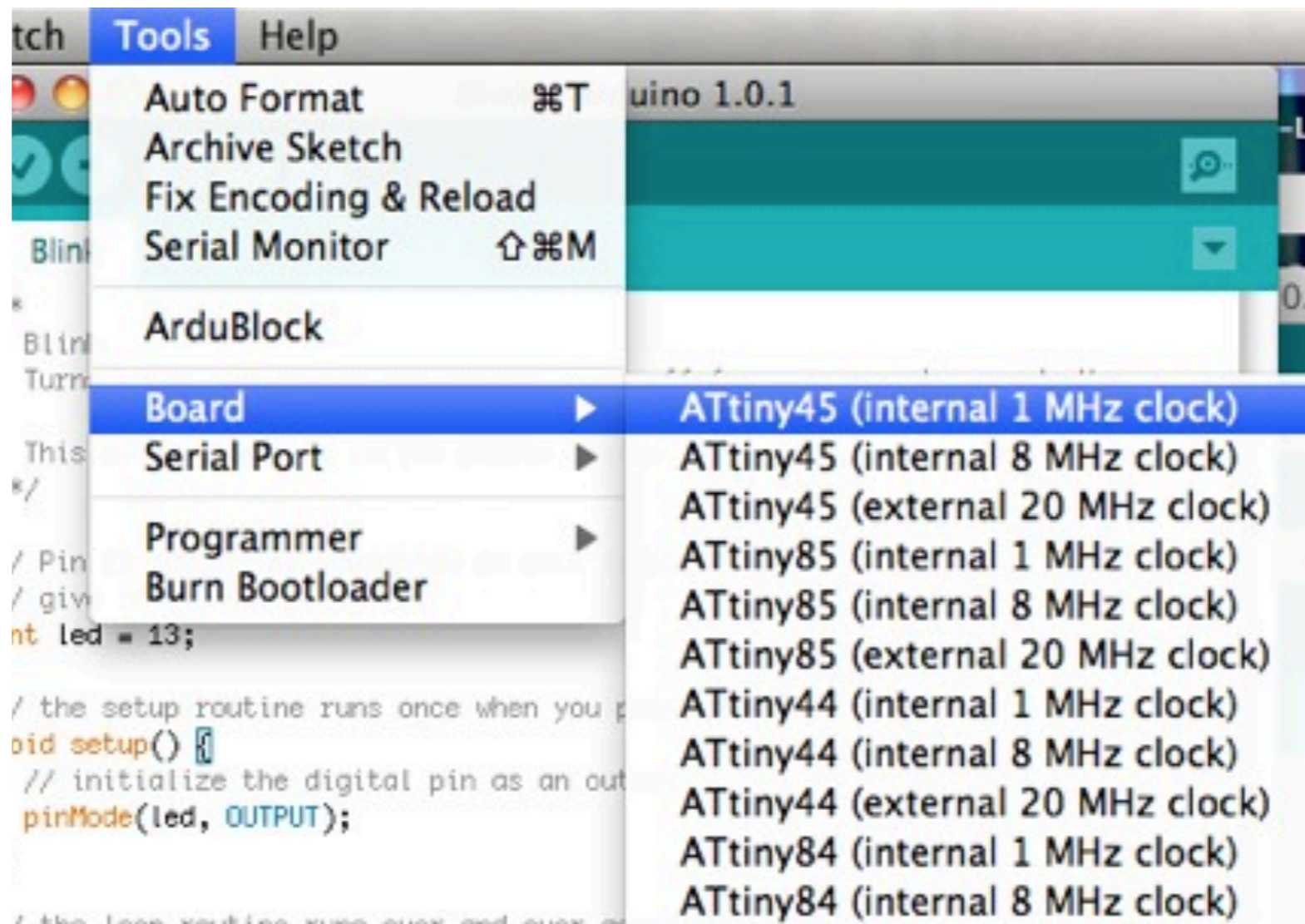


```
Blink | Arduino 1.0.1
Blink
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeatedly.
 *
 * This example code is in the public domain.
 */
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;
// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}
// the loop routine 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
}
```

# pick the right board

PROGRAMMING THE ATTINY

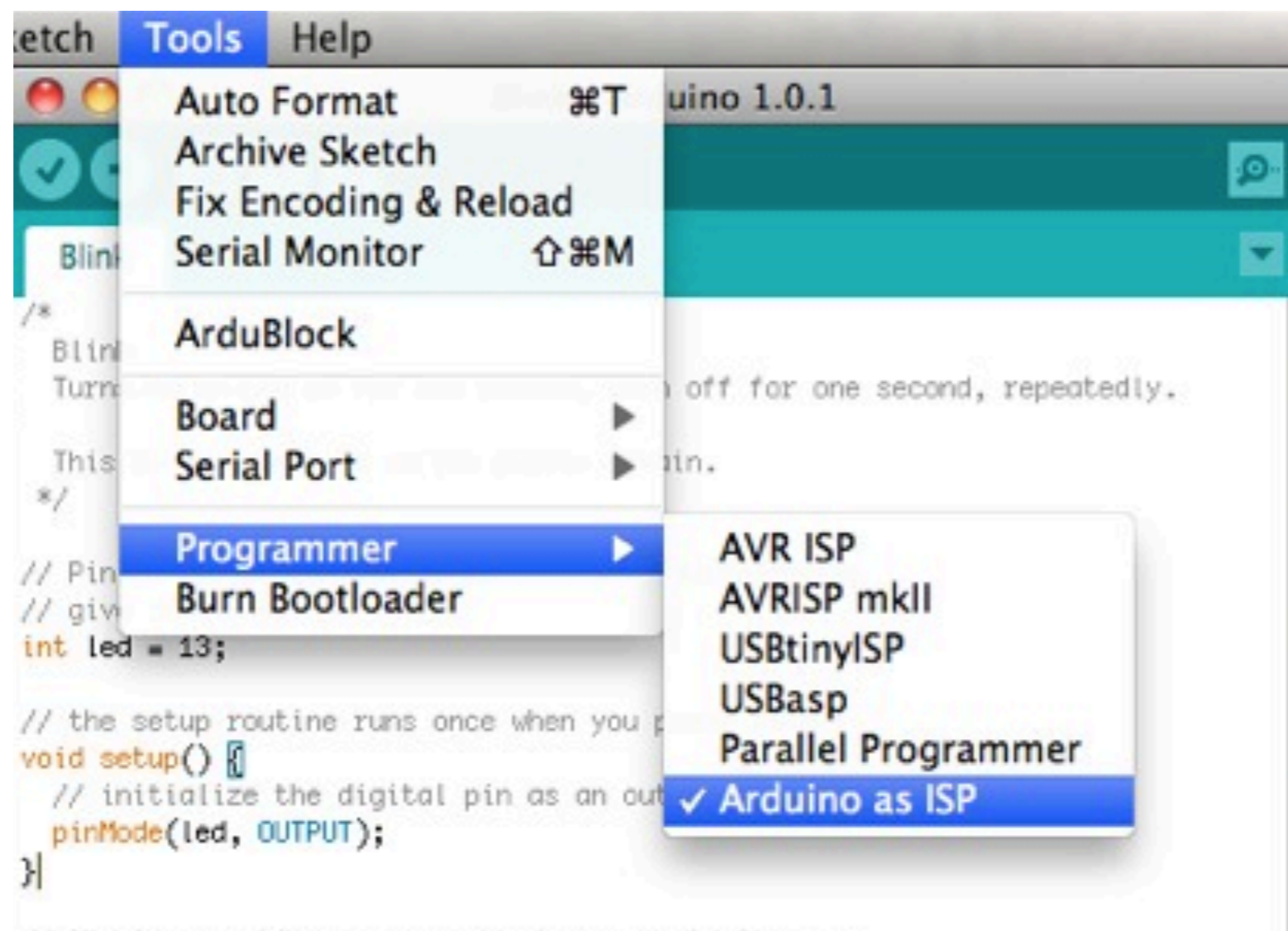
Go into **Tools > Boards > ATtiny45 (Internal 1MHz Clock)** or **ATtiny45 (Internal 1MHz Clock)**



# assign the programmer

PROGRAMMING THE ATTINY

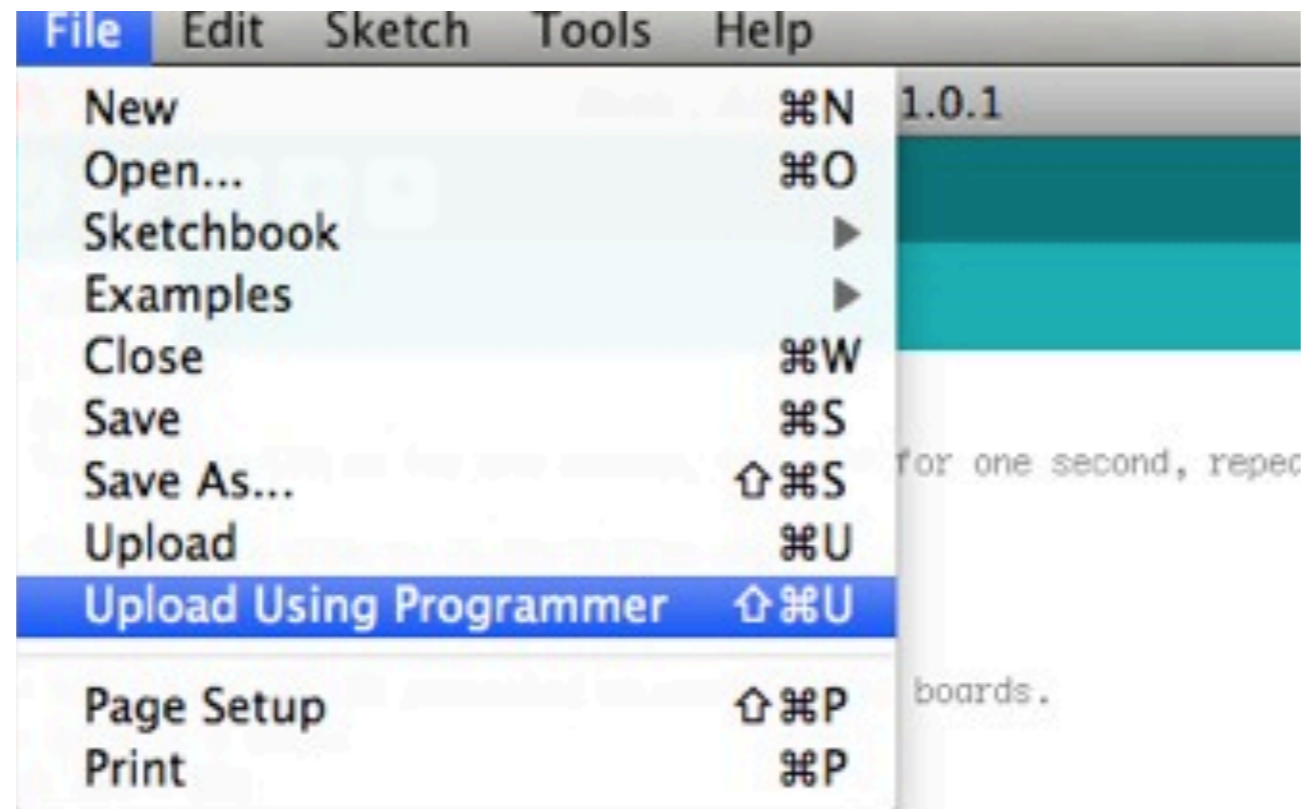
Go into **Tools** > **Programmer** > **Arduino as ISP**



# upload the sketch

PROGRAMMING THE ATTINY

Go into **File** > **Upload using programmer**



```
/ the setup routine runs once when you press reset:  
void setup() {  
  // initialize the digital pin as an output.  
  pinMode(led, OUTPUT);  
}
```

# upload the sketch

PROGRAMMING THE ATTINY

This is **OK**:

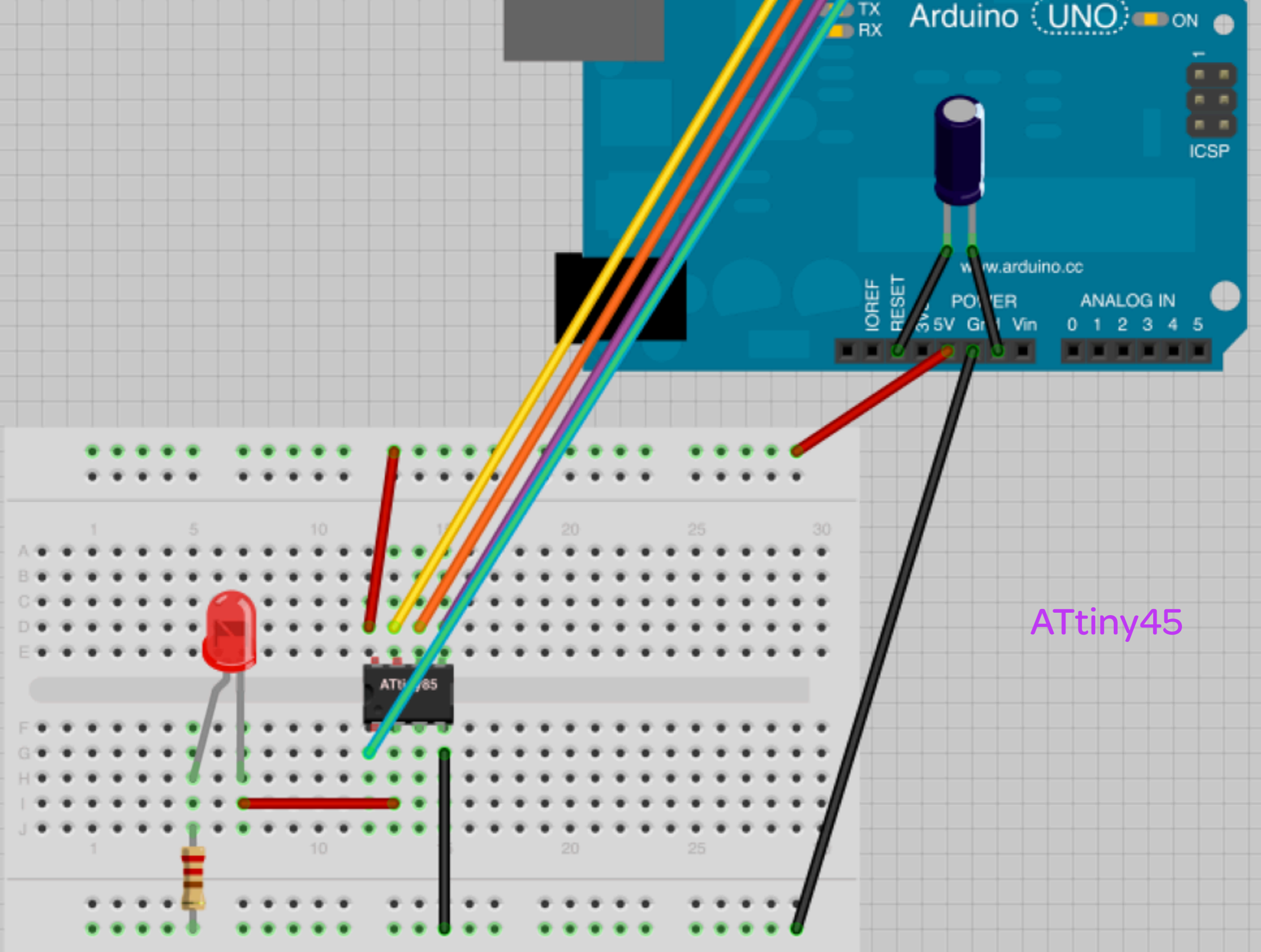
```
avrdude: please define PAGEL and BS2 signals in the configuration file for part ATtiny85  
avrdude: please define PAGEL and BS2 signals in the configuration file for part ATtiny8
```

# make light!

PROGRAMMING THE ATTINY

Insert an LED on pin 3 for the ATtiny45/85





debug time

# next steps

TRY SOMETHING ELSE!

Take it off the Arduino and connect it to a 3 volt.  
//One more reason why the ATtiny is AWESOME.

Try to make a light (or two!) fade.  
//Using PWM

Try this capacitance sketch  
//<http://hlt.media.mit.edu/?p=1653>

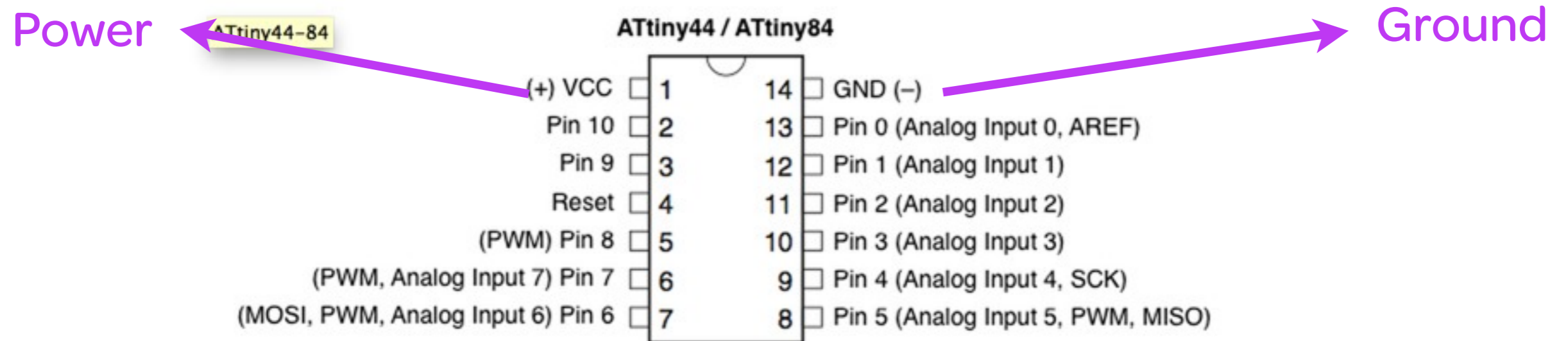
A note on sound...

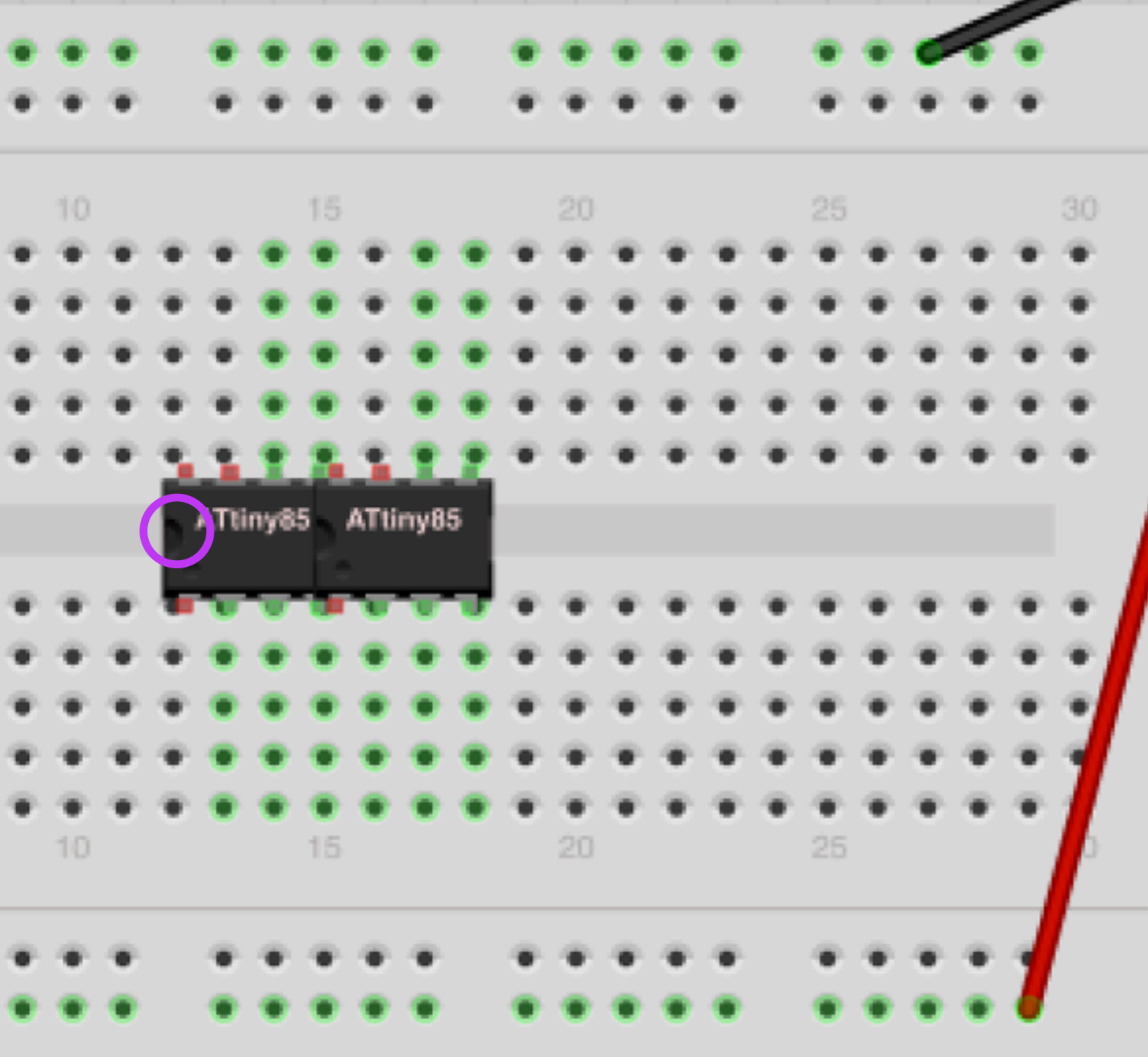
BREAK!

[then midterm sharings]

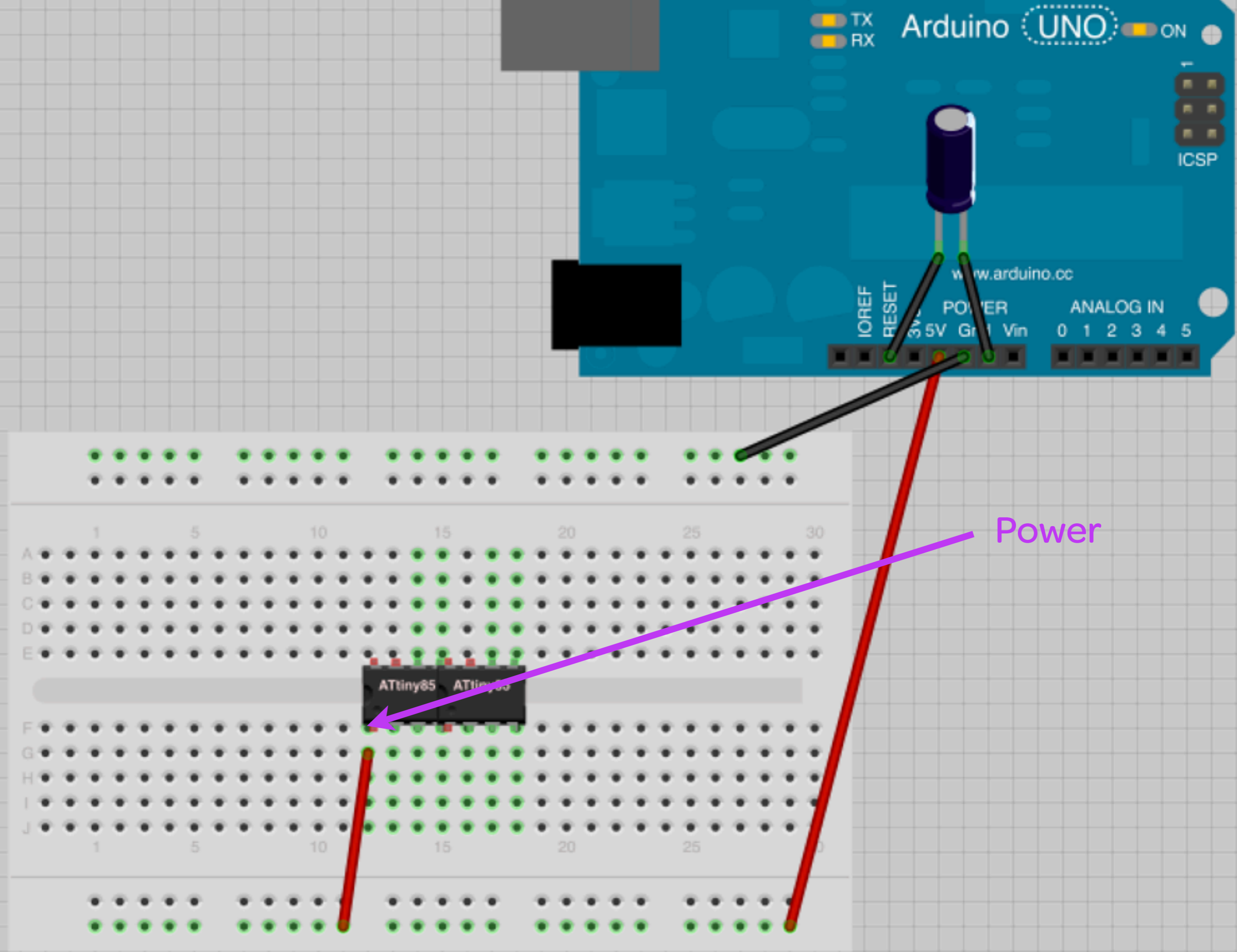
# hookin it up: attiny44/84

CONNECTING THE ATTINY TO THE ARDUINO

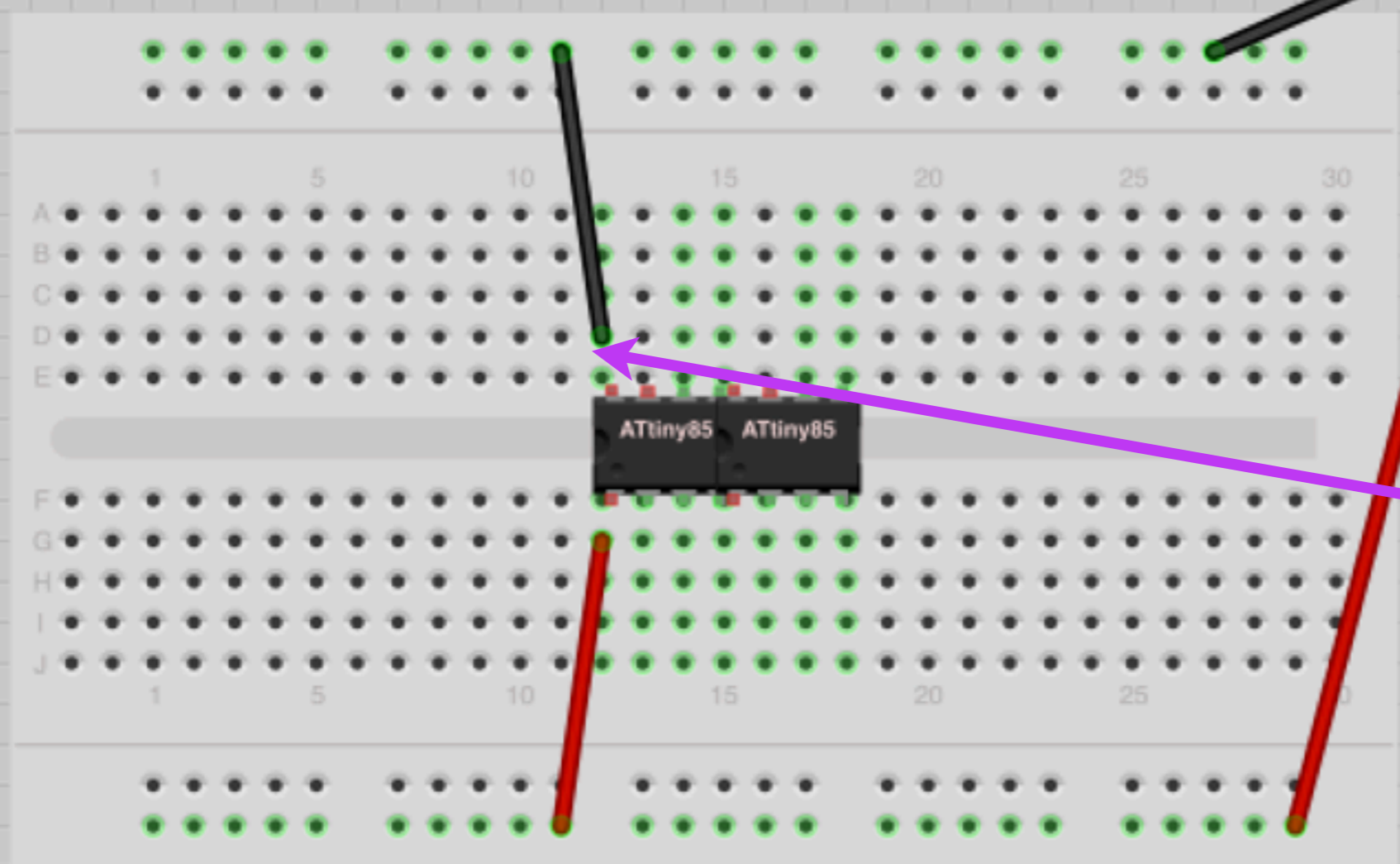




ATtiny85 ATtiny85



Power

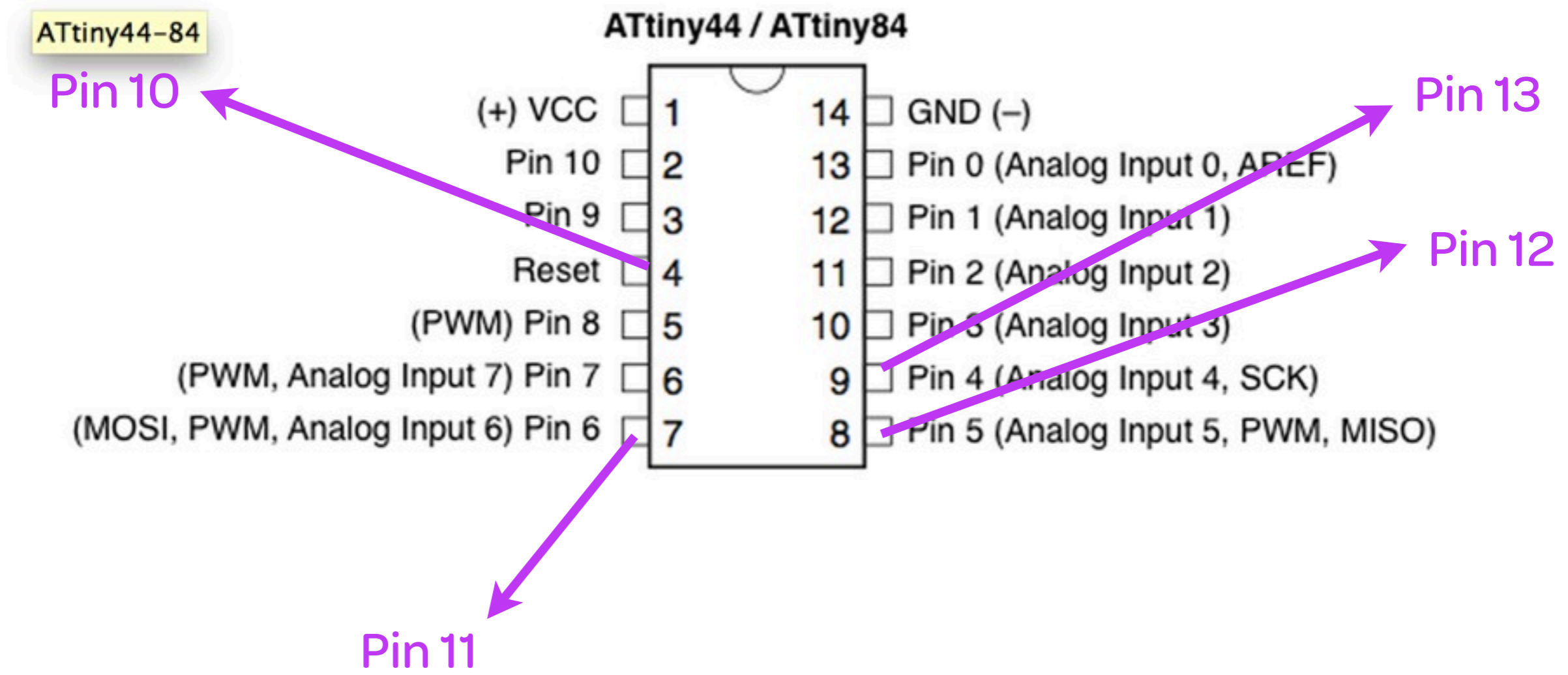


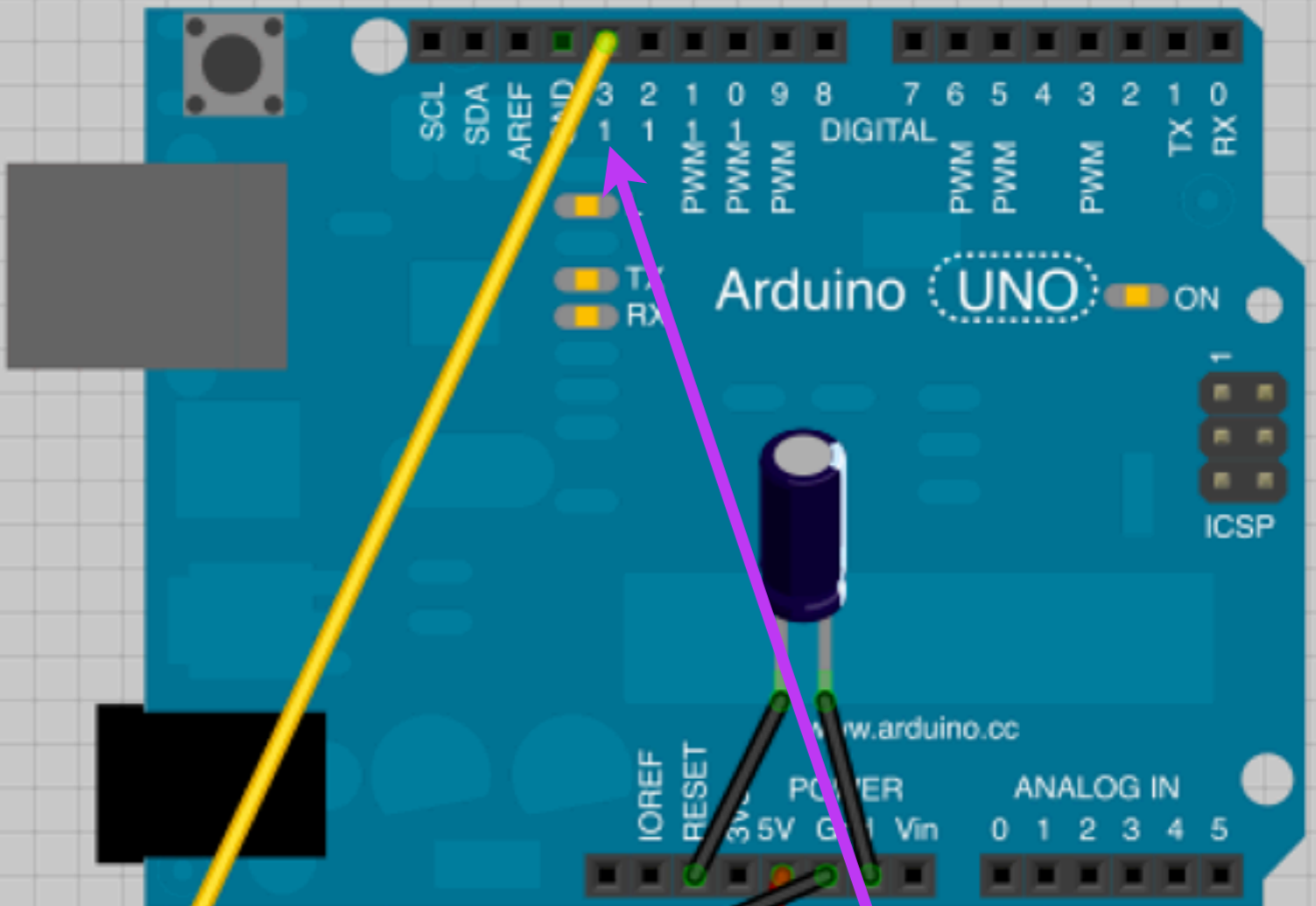
Ground



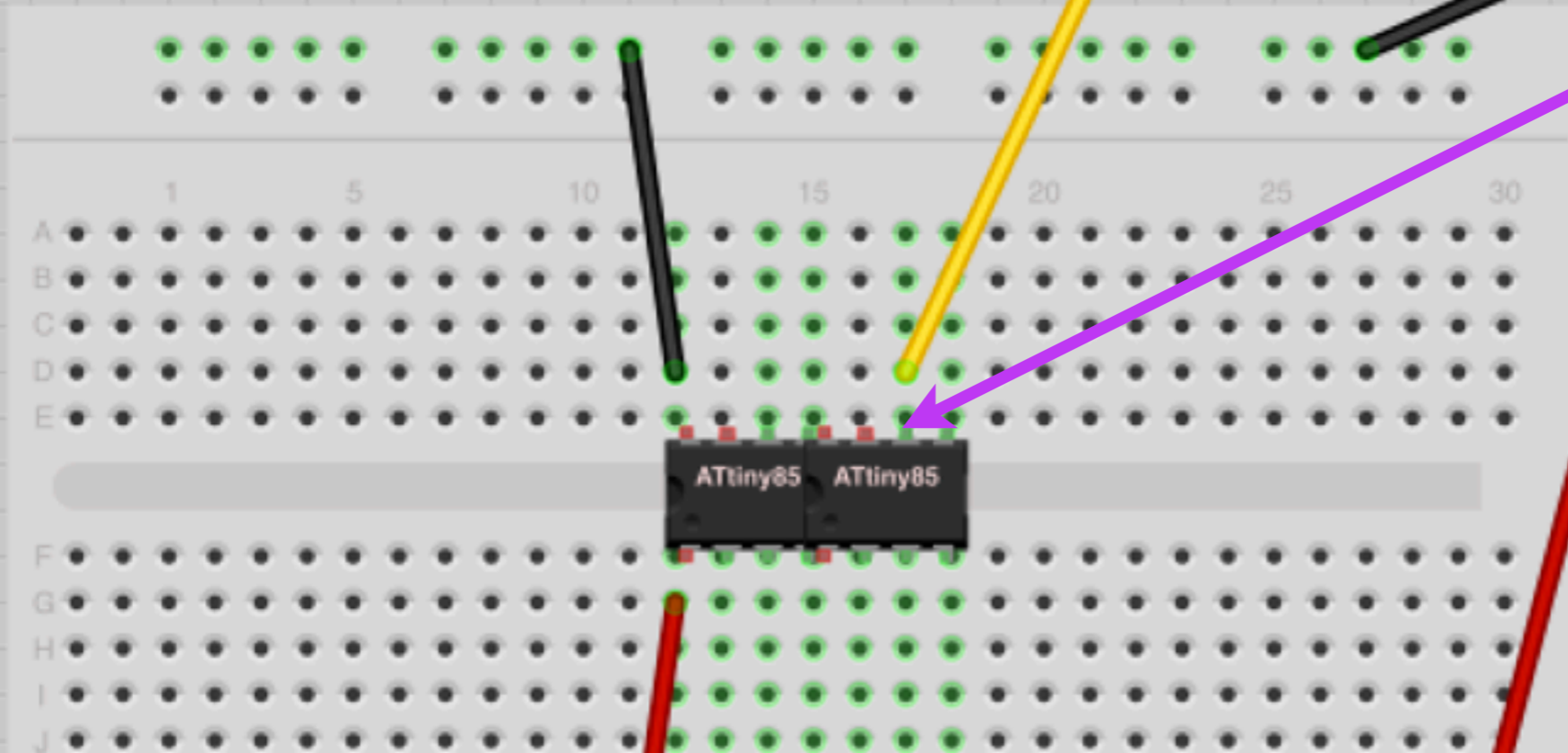
# hookin it up: attiny44/84

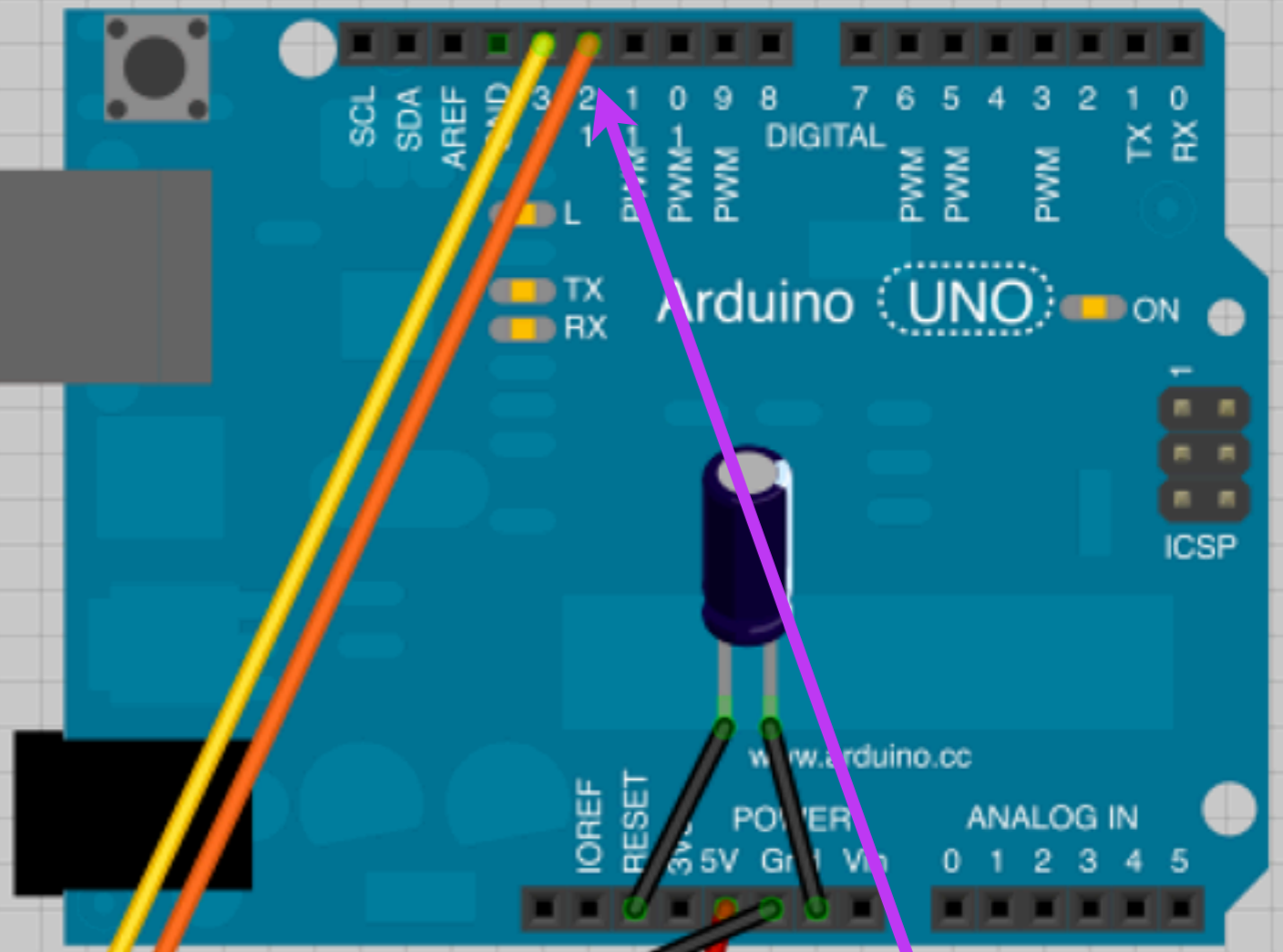
CONNECTING THE ATTINY TO THE ARDUINO



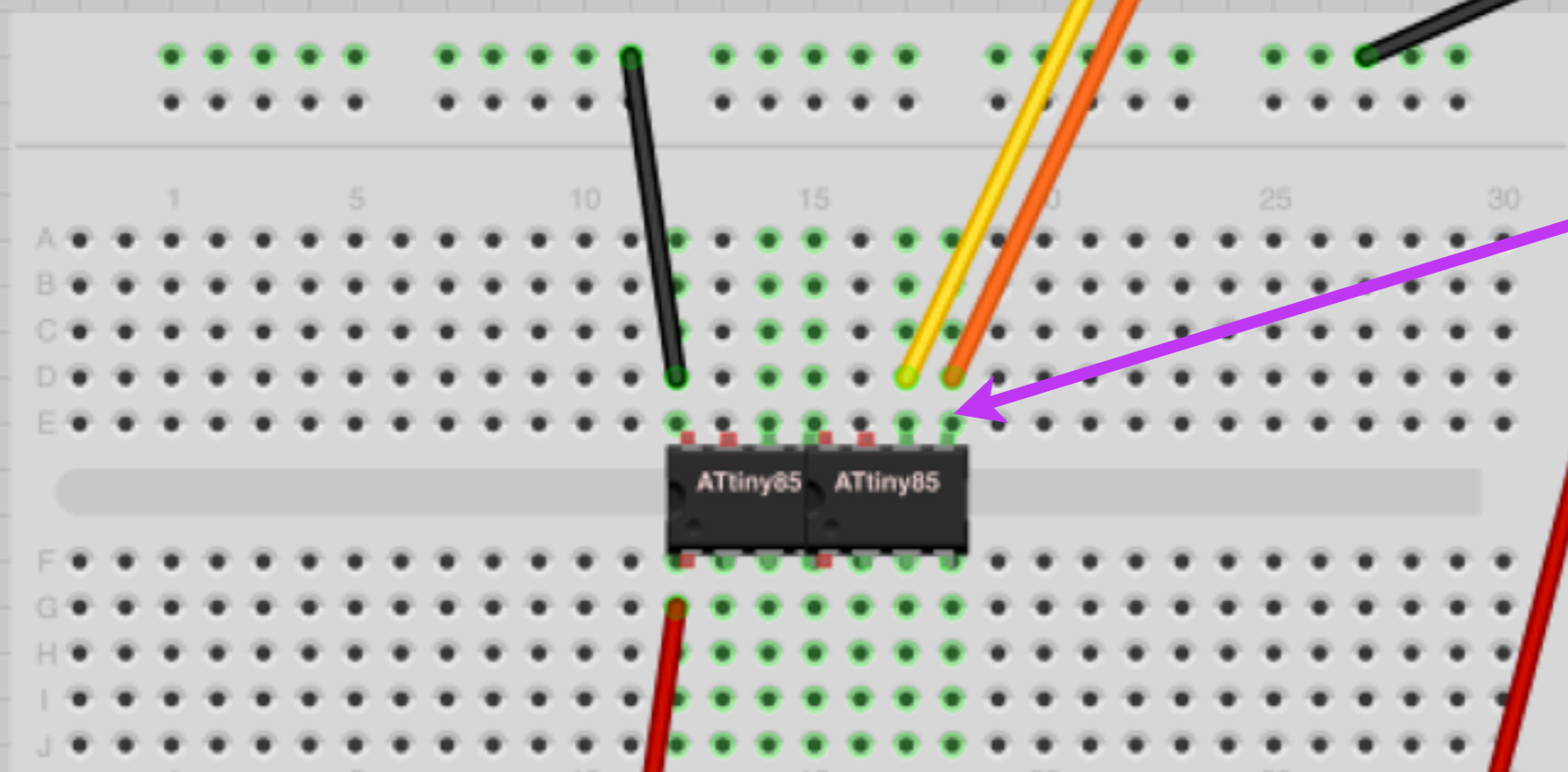


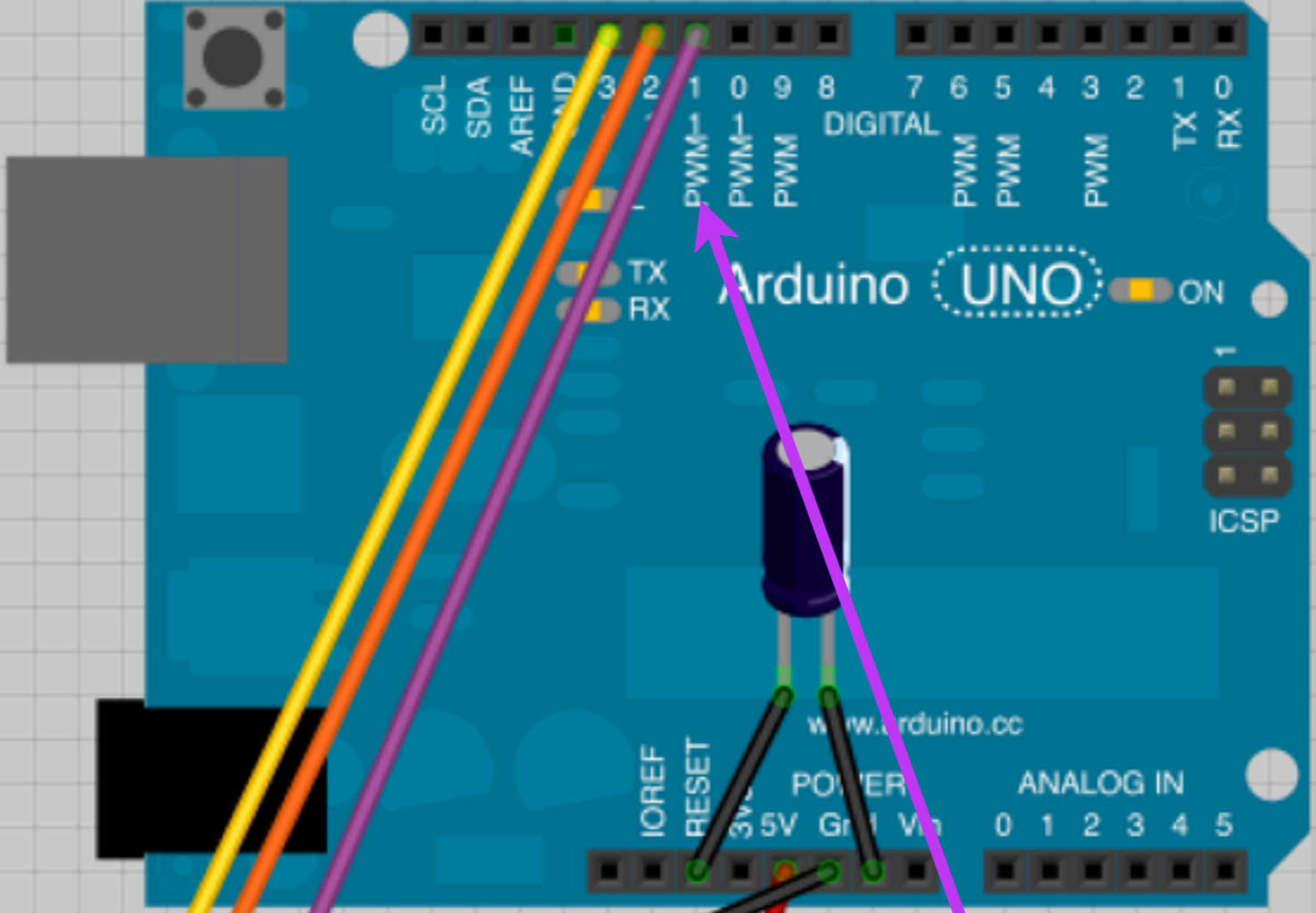
Pin 13





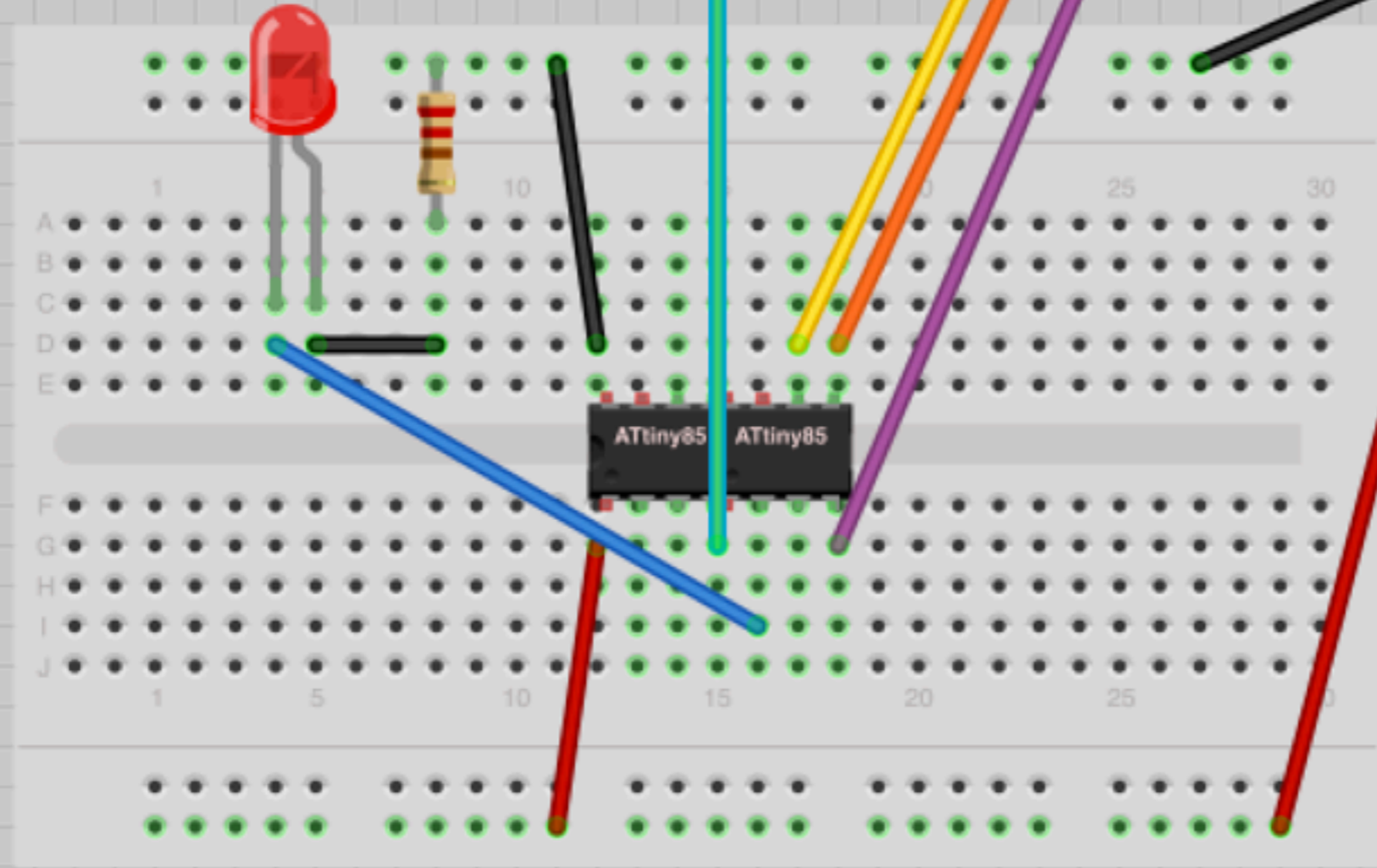
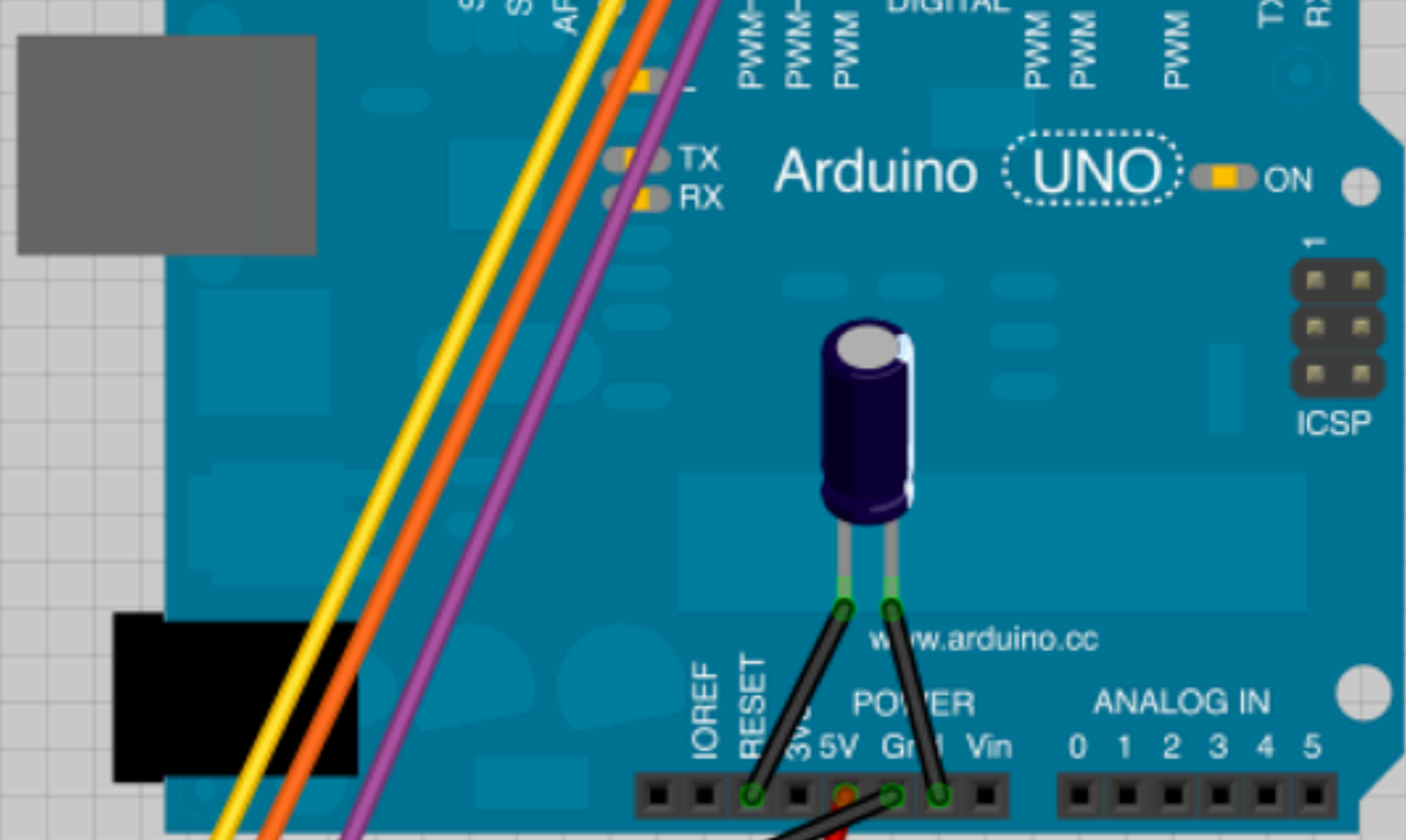
Pin 12





Pin 11





ATtiny84