

week 02



Physical Computing

Bridging the gap between the physical and virtual

How the Computer Sees Us

Shall we take a better look at ourselves to see our full range of expression?



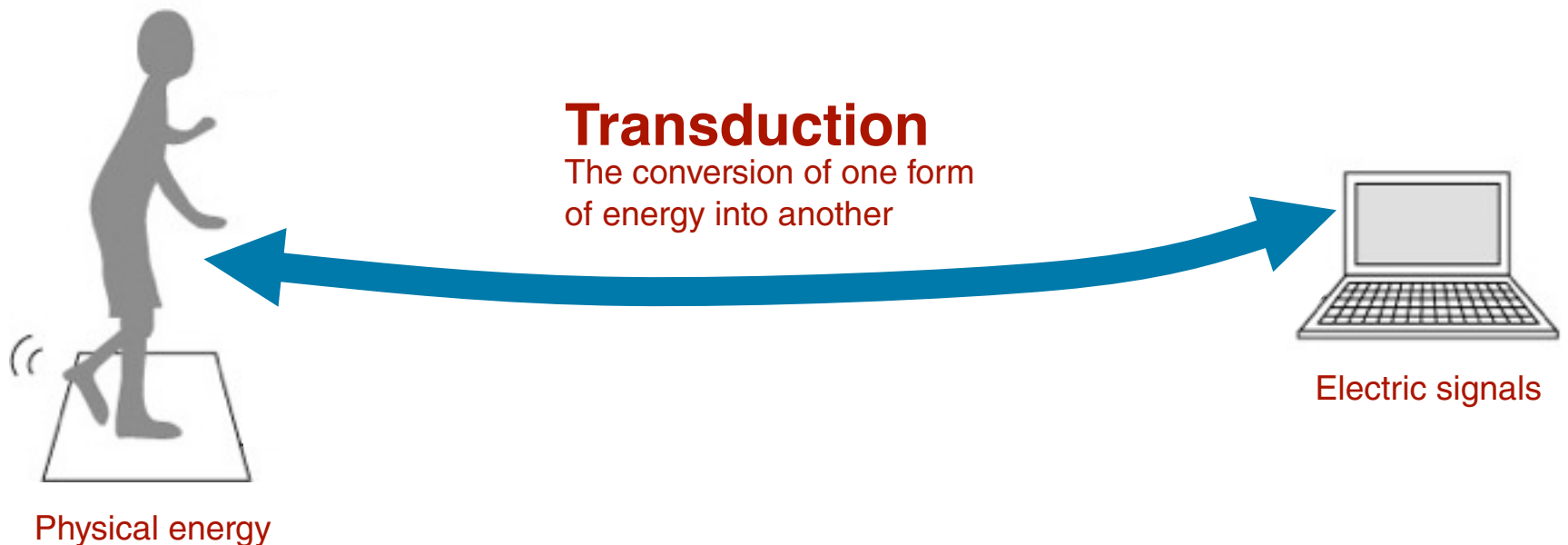
Physical Computing

A conversation between the physical world and the virtual world of the computer.



Physical Computing

A conversation between the physical world and the virtual world of the computer.



Input and Output

Input

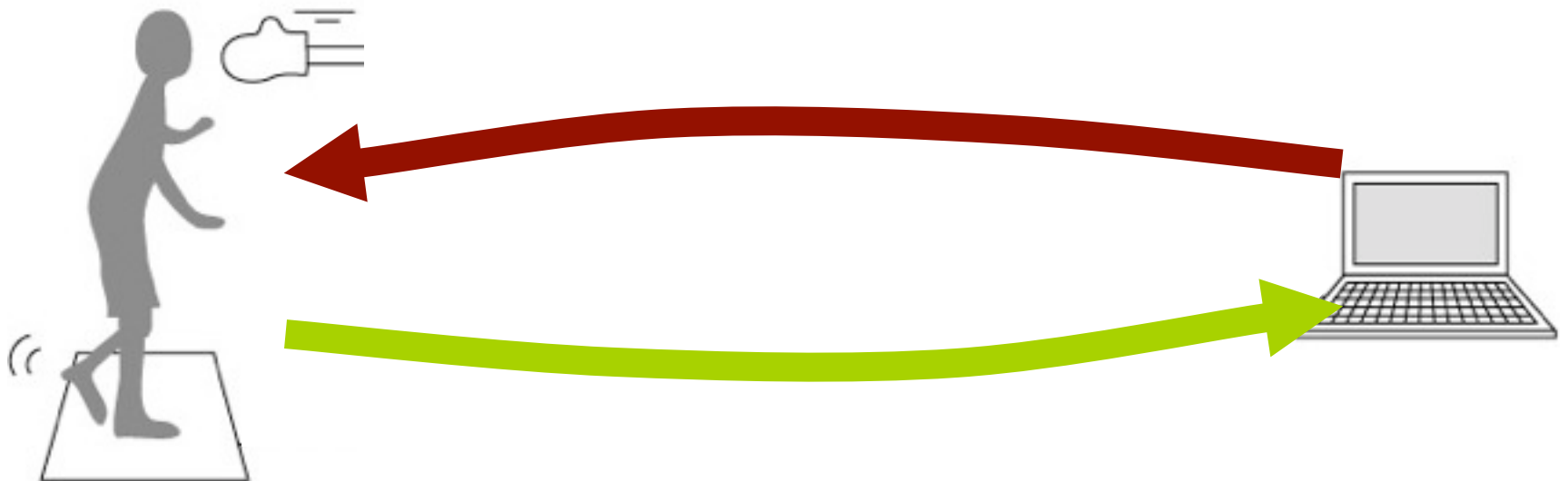
Ways of sensing your physical energy/expressions.
Input is usually easier than output because it takes less energy to sense activity than to move things.



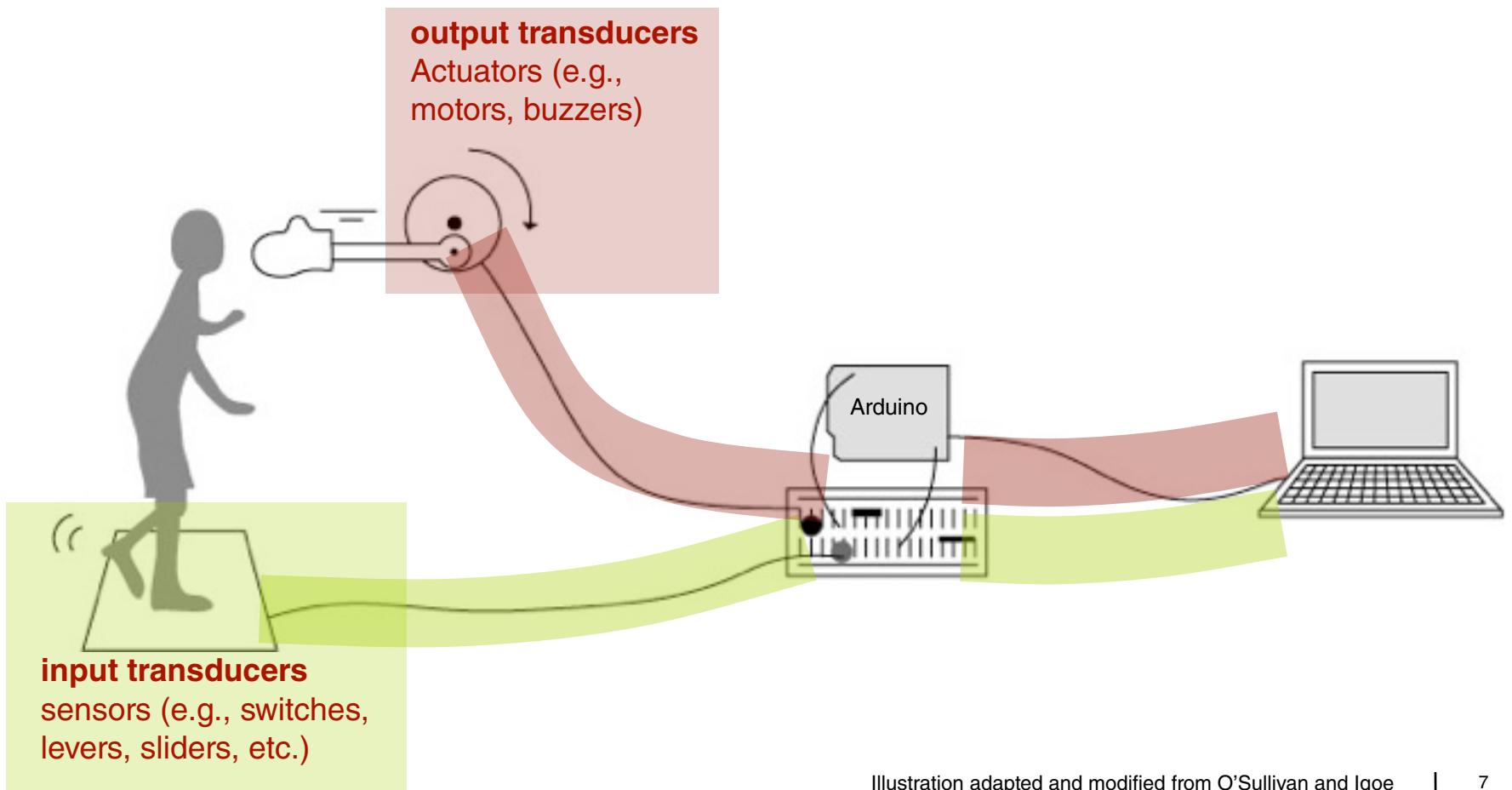
Input and Output

Output

Physical computing is not just about sensing the world, but also about changing it. But moving things are hard (you need electrical and mechanical skills).



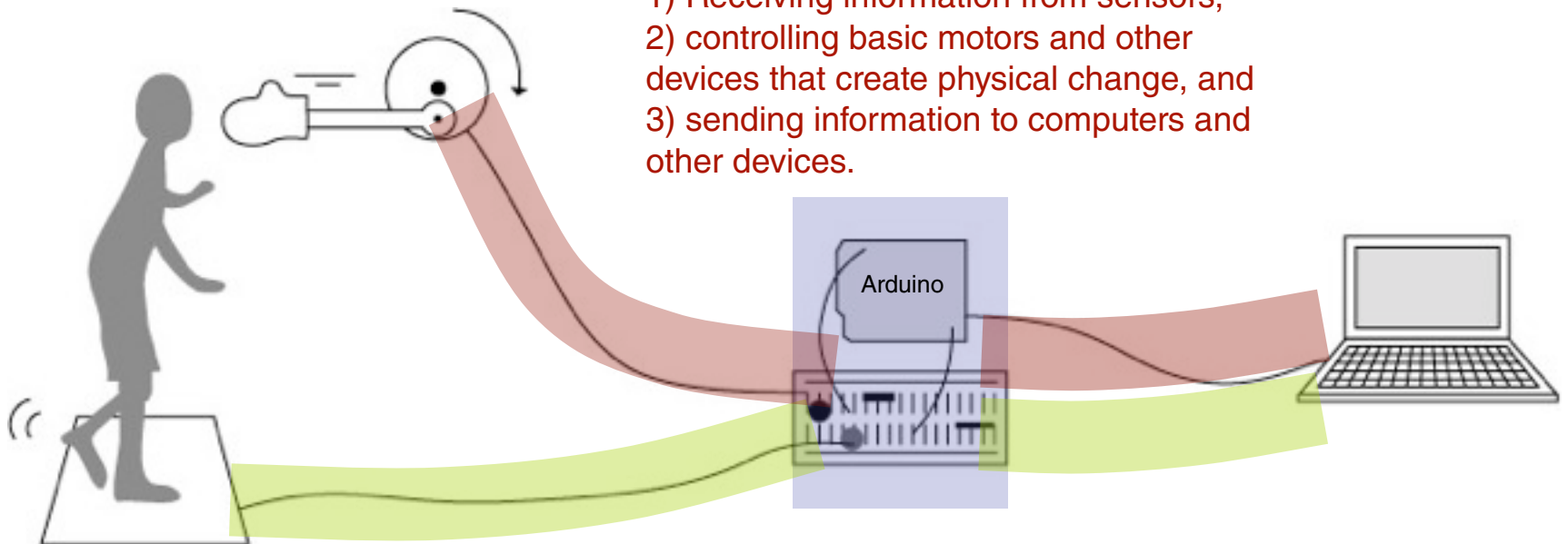
Transducers



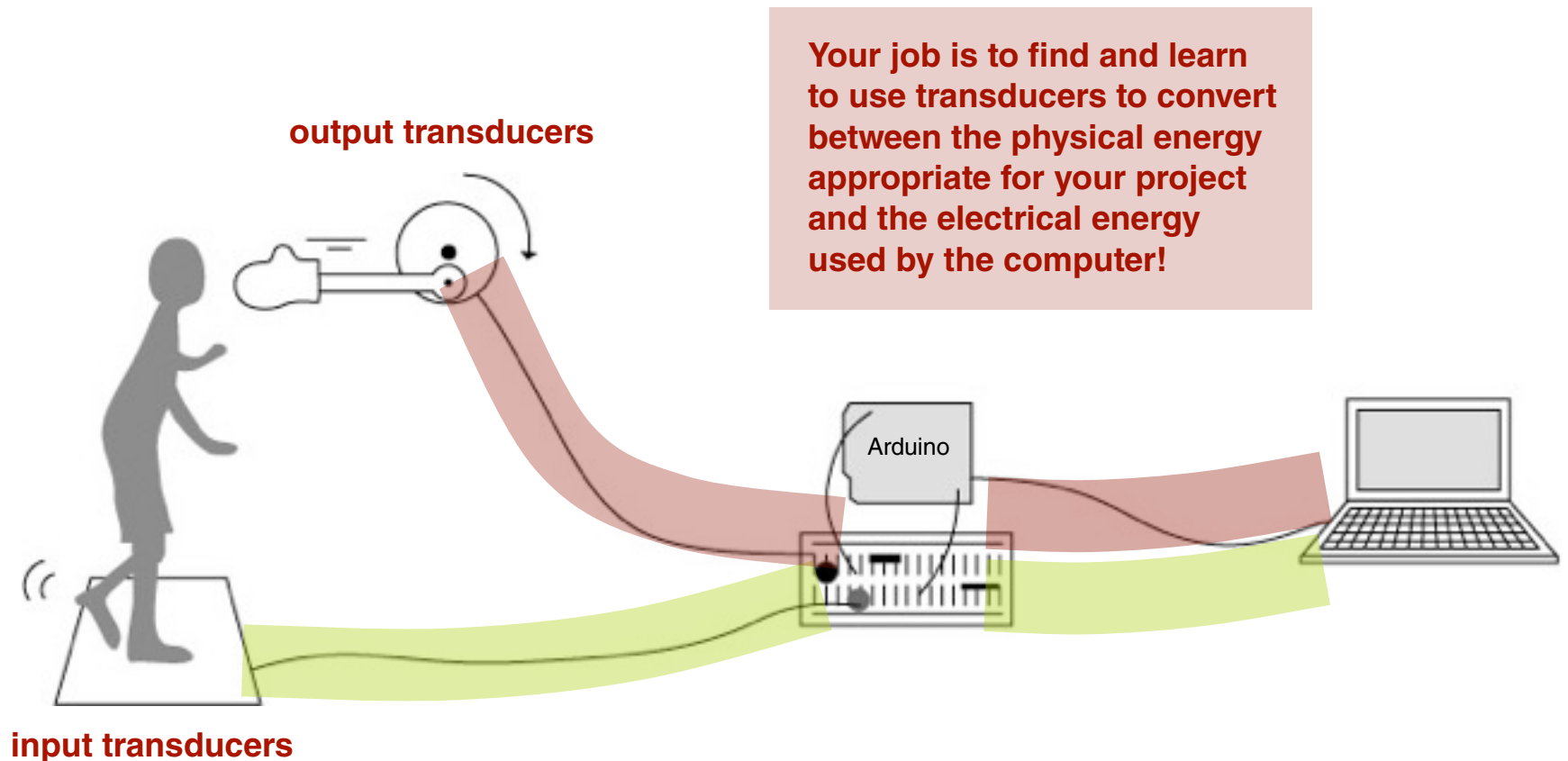
Microcontrollers

Gateway between the physical and the virtual

- 1) Receiving information from sensors,
- 2) controlling basic motors and other devices that create physical change, and
- 3) sending information to computers and other devices.



Transduction



Word of Caution: Your Idea is Important

1. Don't get trapped in technological seduction
2. Don't spin your wheels for so long that you give up your project. There might be an alternative way that makes things easier.

Work at a high level. Talk to us. Ask other people. Take frequent breaks.

Working with Arduino

An open-source electronics prototyping

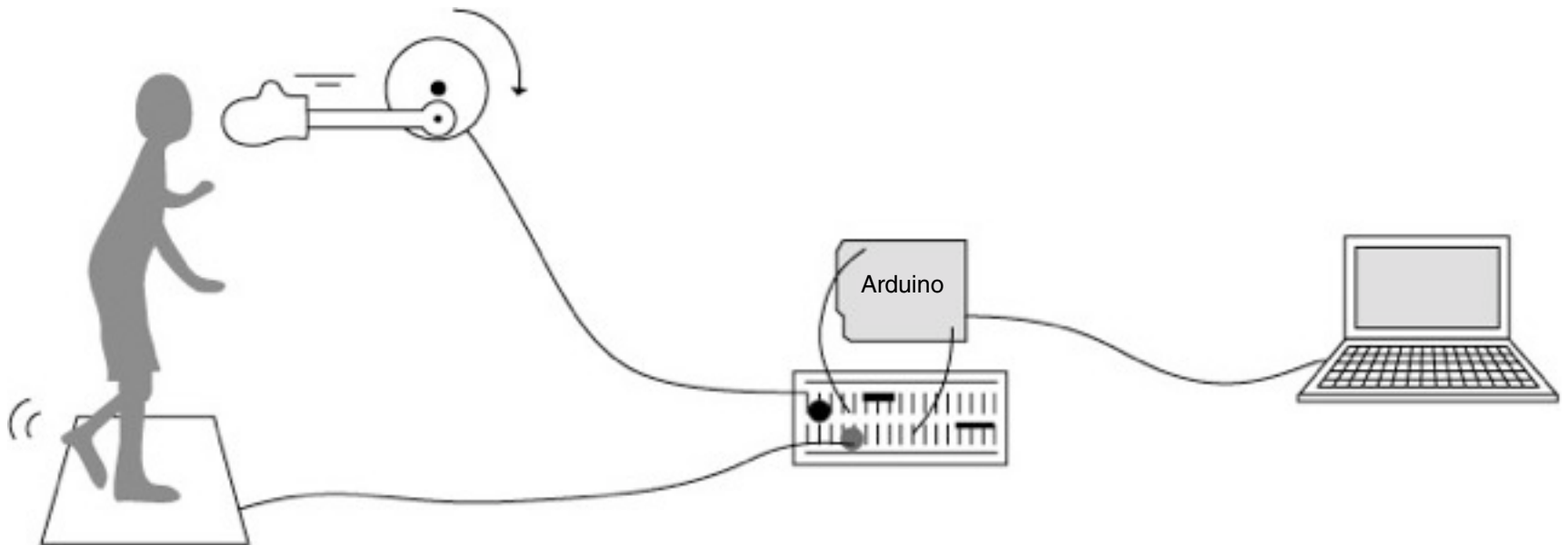
Create interactive objects and environments

What is Arduino

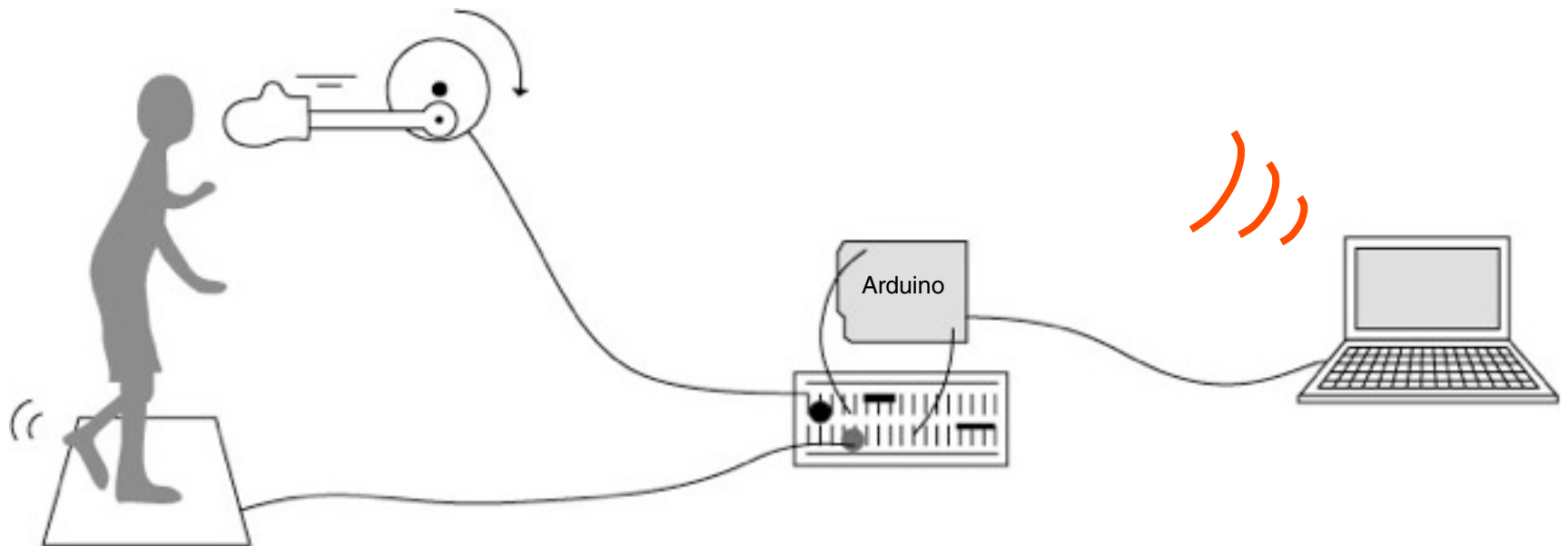
A tiny computer you can program, for rapid prototyping



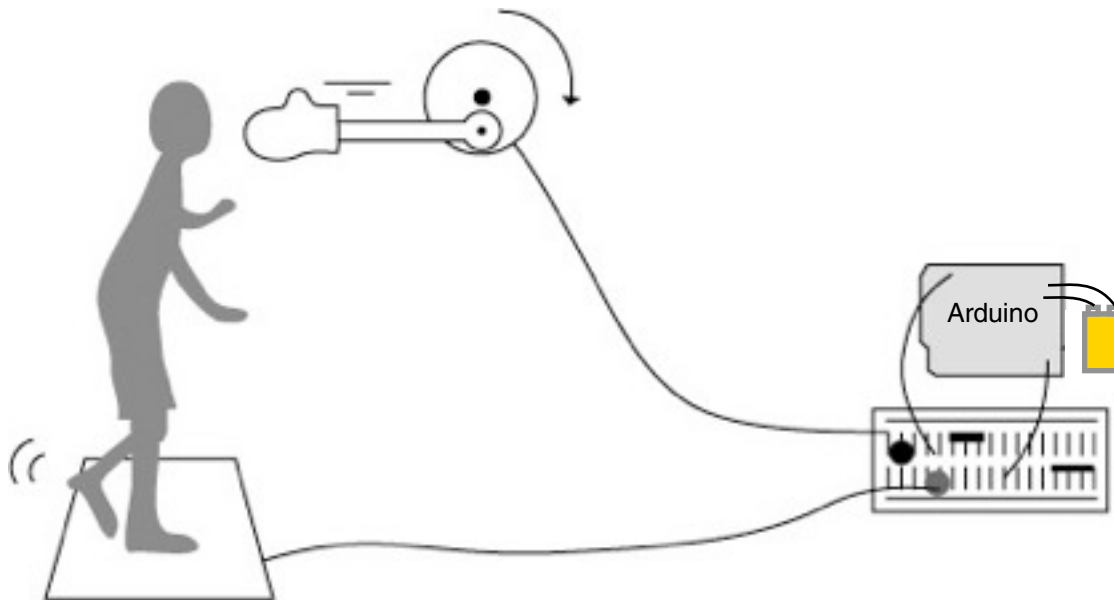
Arduino as an Interface Board



Arduino as an Interface Board



Arduino as an Embedded Computing Device

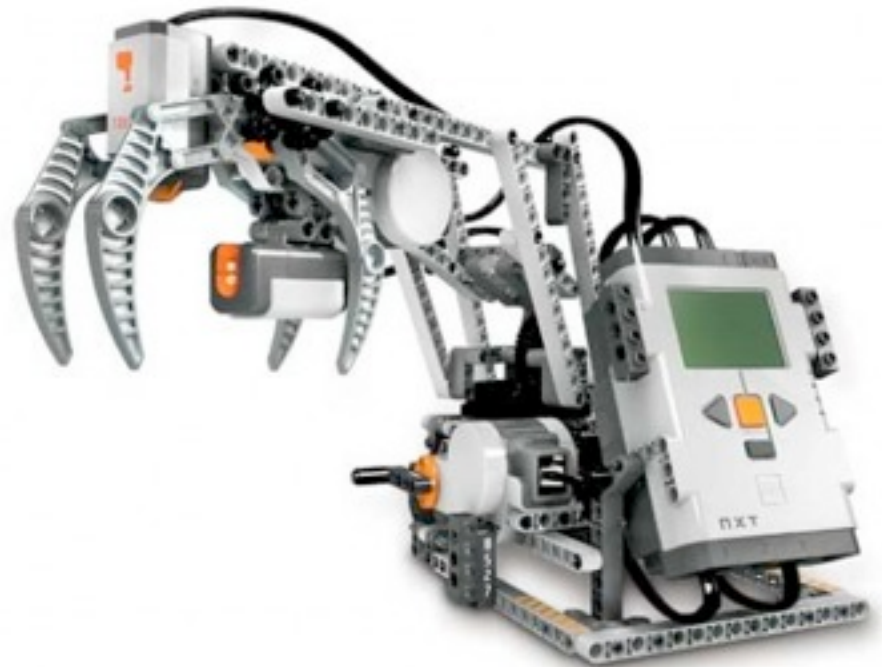


What is Arduino

Relatively cheap (compare with LEGO Mindstorm)



\$31



\$250 ~

What is Arduino

It used to be a bit more complicated...

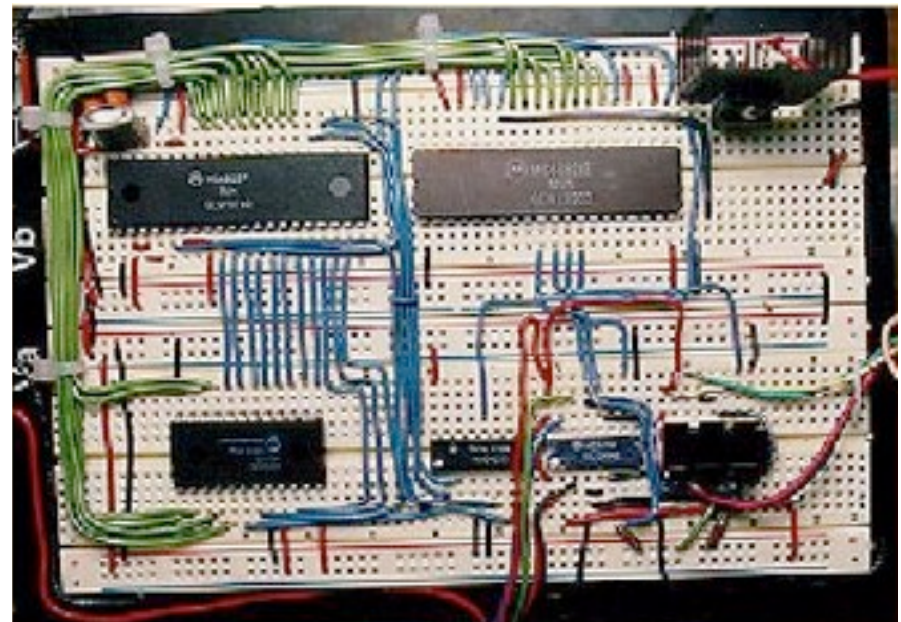


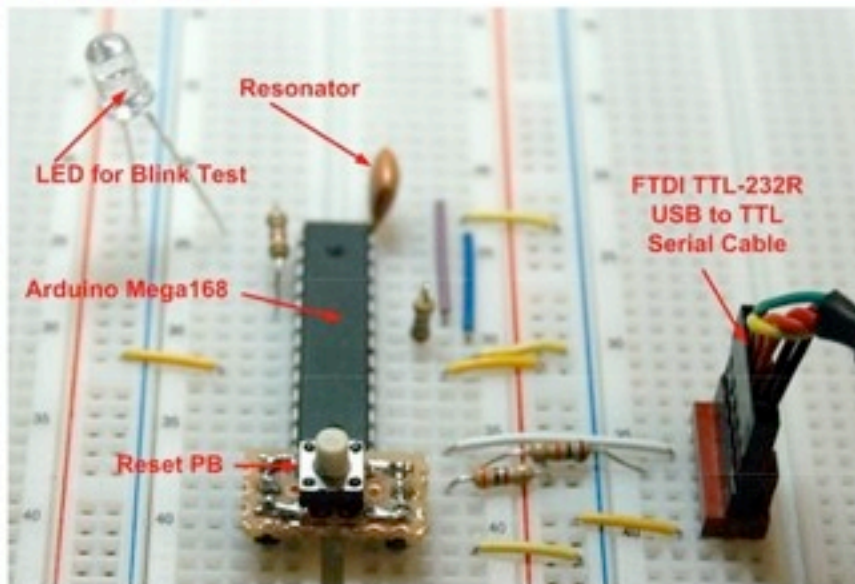
image from www.tangentsoft.net/elec/breadboard.html

Make:

technology on your time

[Blog](#)
[Make Magazine](#)
[Podcasts](#)
[HOW TO - Make an aux-in for the Bose SoundDock](#)
[Main](#)
[Last Day of the Weekend](#)
[Subscribe to the Weekend Projects Podcast](#)

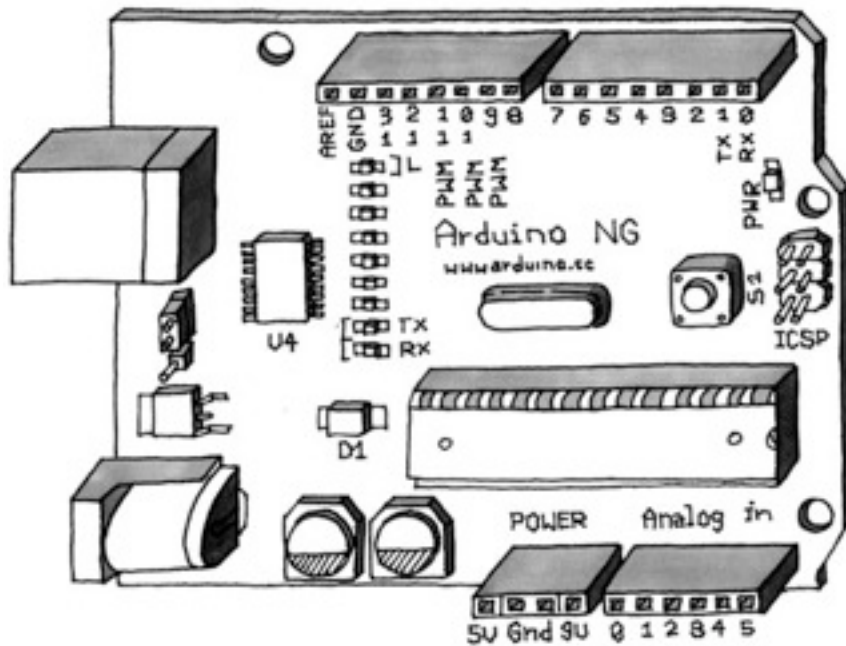
Barebones Arduino on a breadboard



Here is a truly barebones Arduino setup. Just the Arduino chip and a few support parts. This has to be close to the simplest and lowest cost way to play with microcontrollers. The only special parts are the resonator and the Mega168 programmed with the Arduino boot loader. Everything else you should already have as an electronics hobbyist.

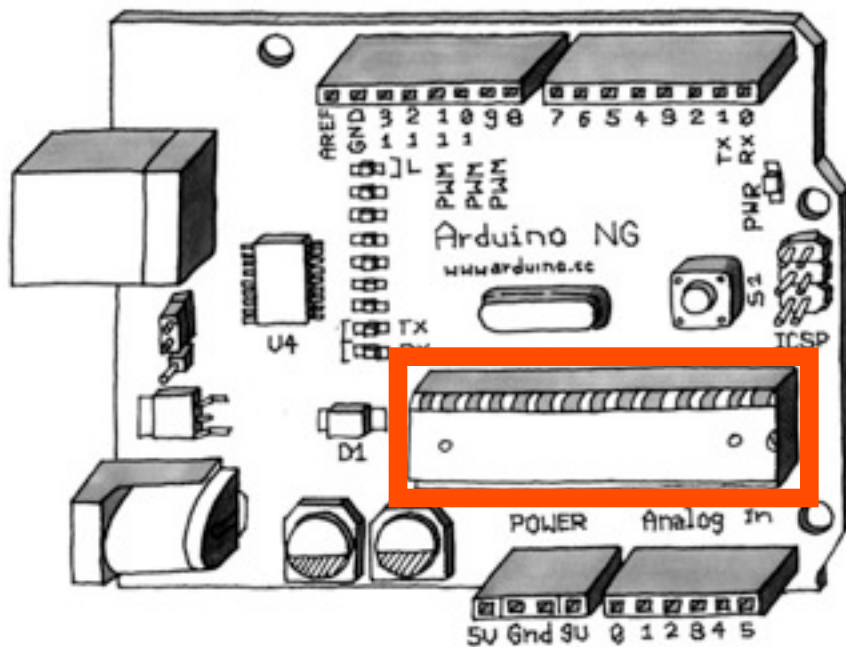
Open source, so you can build one yourself!

Arduino Board Overview



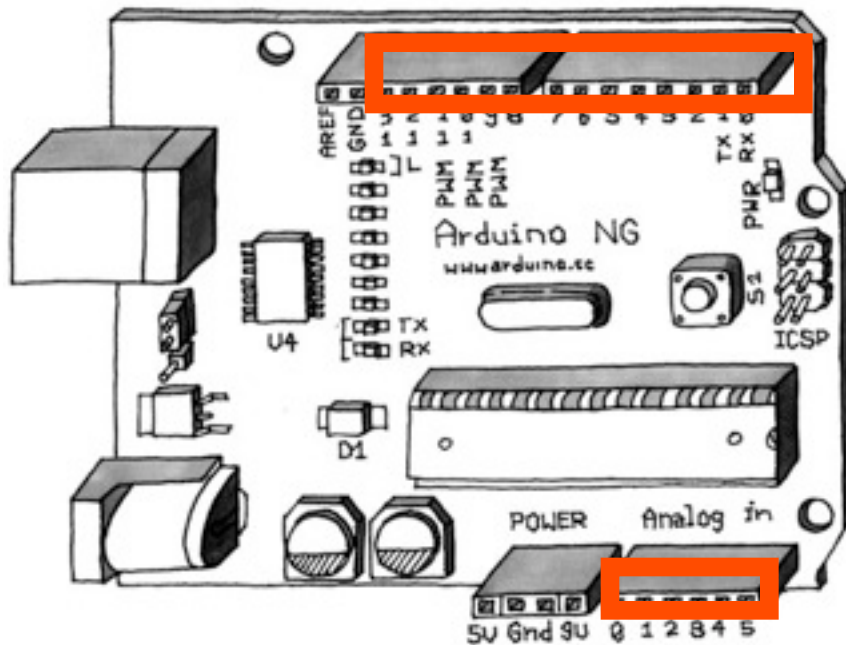
Arduino Board Overview

- 32 kBytes of Flash program memory (your program stays in Arduino when powered off)
- 2 kByte of RAM
- 16 MHz processor speed (c.f., Apple II: 1 MHz)

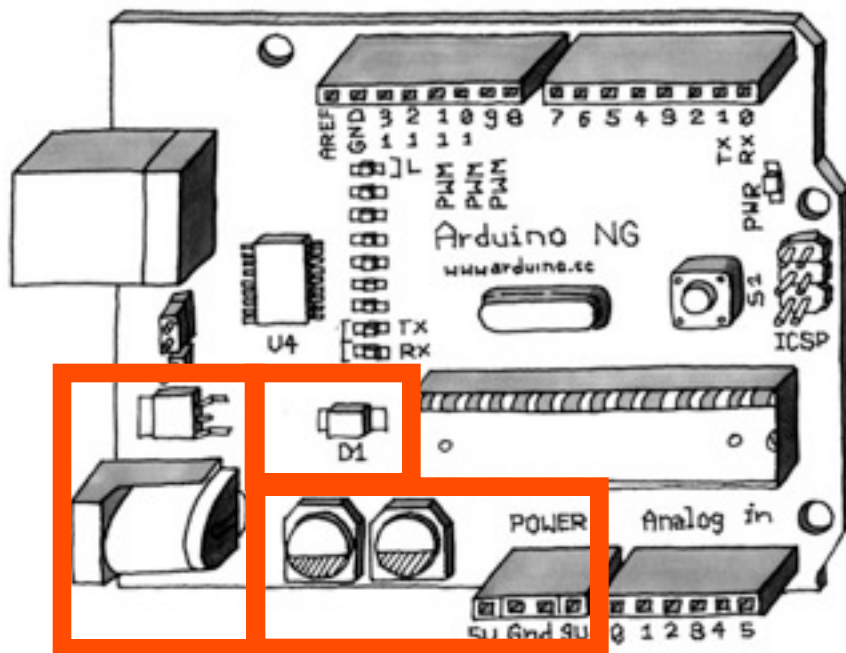


Arduino Board Overview

- 14 digital input/output pins
- 6 analog input pins

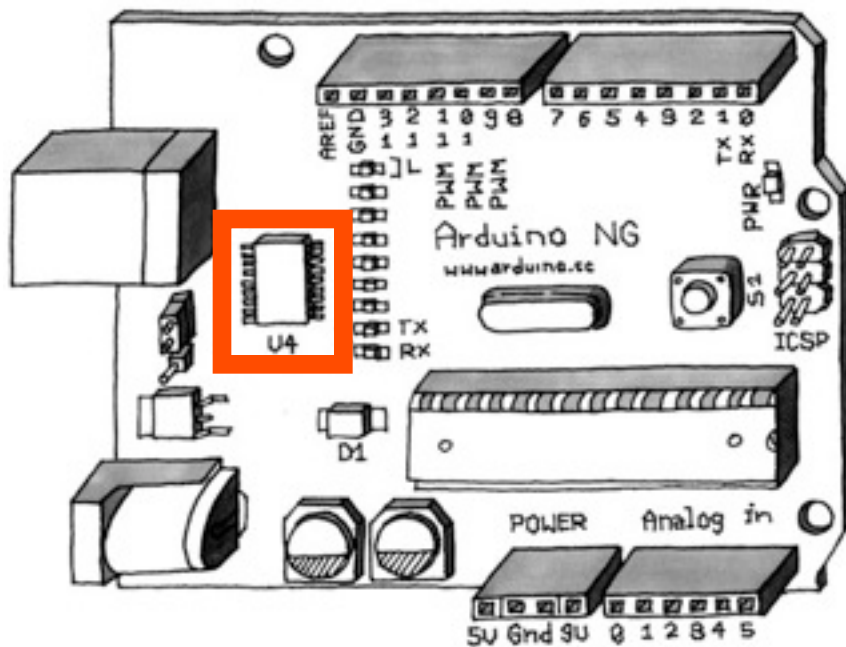


Arduino Board Overview



Arduino Board Overview

- USB to serial chip (converts simple serial signal to USB)



Arduino Software

www.arduino.cc

Reference Blog Playground Forum

Help | Sign in or Register



Buy Download Getting Started Learning Reference Hardware FAQ



Photo by the Arduino Team

[Arduino on Twitter \(more\)](#)

2 days ago

@apfellicie it's now in the hands of 20 beta testers.. more news

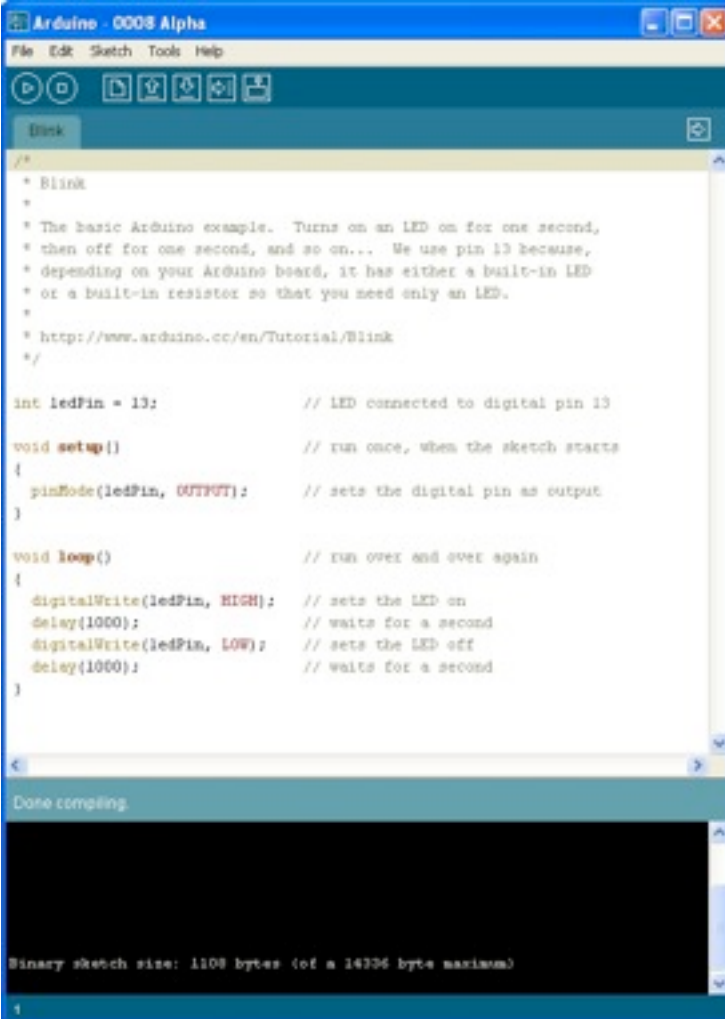
Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments.

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the **Arduino programming language** (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).

The boards can be **built by hand** or purchased preassembled; the software can be **downloaded** for free. The hardware reference designs (CAD files) are **available** under an open-source license, you are free to **adapt them to your needs**.

Arduino received an Honorary Mention in the Digital Communities section of the 2006 Ars Electronica Prix. The Arduino team is: Massimo

Arduino Software



The screenshot shows the Arduino IDE interface with the 'Blink' sketch loaded. The code is as follows:

```
/*
 * Blink
 *
 * The basic Arduino example. Turns on an LED on for one second,
 * then off for one second, and so on... We use pin 13 because,
 * depending on your Arduino board, it has either a built-in LED
 * or a built-in resistor so that you need only an LED.
 *
 * http://www.arduino.cc/en/Tutorial/Blink
 */

int ledPin = 13;          // LED connected to digital pin 13

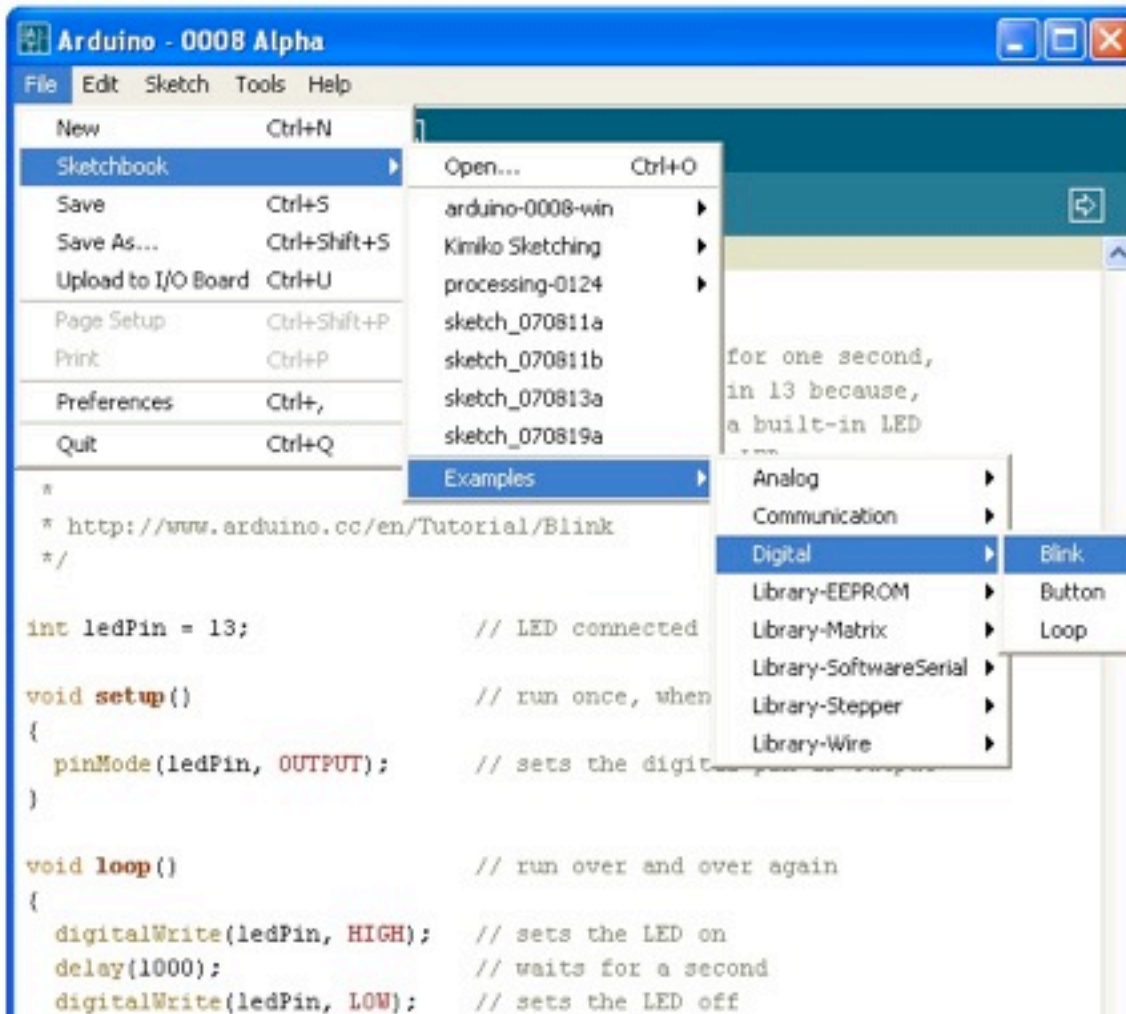
void setup()             // run once, when the sketch starts
{
  pinMode(ledPin, OUTPUT); // sets the digital pin as output
}

void loop()              // run over and over again
{
  digitalWrite(ledPin, HIGH); // sets the LED on
  delay(1000);                // waits for a second
  digitalWrite(ledPin, LOW);  // sets the LED off
  delay(1000);                // waits for a second
}
```

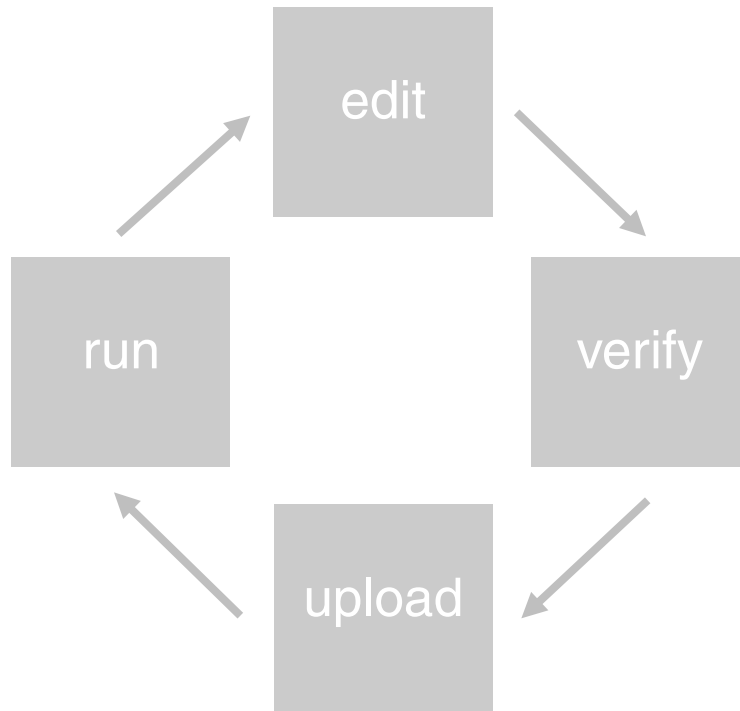
Done compiling

Binary sketch size: 1108 bytes (of a 14336 byte maximum)

Arduino Sketches



Arduino Programming Cycle

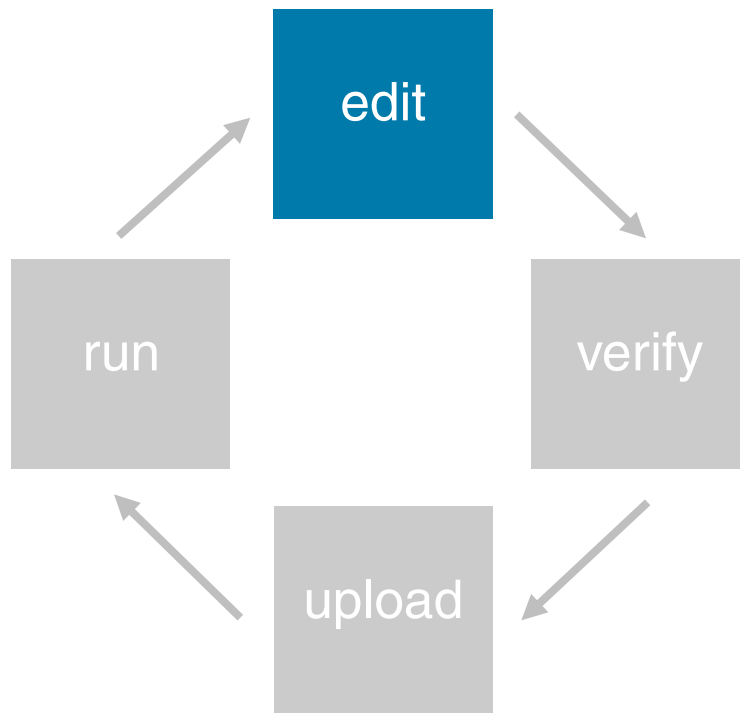


The screenshot shows the Arduino IDE interface with a sketch loaded. The sketch is a simple blink program. The code is as follows:

```
/*  
 * Blink  
 *  
 * The basic Arduino example. Turns on an LED on for one second,  
 * then off for one second, and so on... We use pin 13 because,  
 * depending on your Arduino board, it has either a built-in LED  
 * or a built-in resistor so that you need only an LED.  
 *  
 * http://www.arduino.cc/en/Tutorial/Blink  
 */  
  
int ledPin = 13;           // LED connected to digital pin 13  
  
void setup()              // run once, when the sketch starts  
{  
  pinMode(ledPin, OUTPUT); // sets the digital pin as output  
}  
  
void loop()               // run over and over again  
{  
  digitalWrite(ledPin, HIGH); // sets the LED on  
  delay(1000);                // waits for a second  
  digitalWrite(ledPin, LOW);  // sets the LED off  
  delay(1000);                // waits for a second  
}
```

Below the code editor, the status bar shows "Done compiling" and "Binary sketch size: 1108 bytes (of a 14736 byte maximum)".

Arduino Programming Cycle



The screenshot shows the Arduino IDE interface. The main window displays a sketch for a blinking LED. The code is as follows:

```
* Blink
*
* The basic Arduino example. Turns on an LED on for one second,
* then off for one second, and so on... We use pin 13 because,
* depending on your Arduino board, it has either a built-in LED
* or a built-in resistor so that you need only an LED.
*
* http://www.arduino.cc/en/Tutorial/Blink
*/

int ledPin = 13;           // LED connected to digital pin 13

void setup()              // run once, when the sketch starts
{
  pinMode(ledPin, OUTPUT); // sets the digital pin as output
}

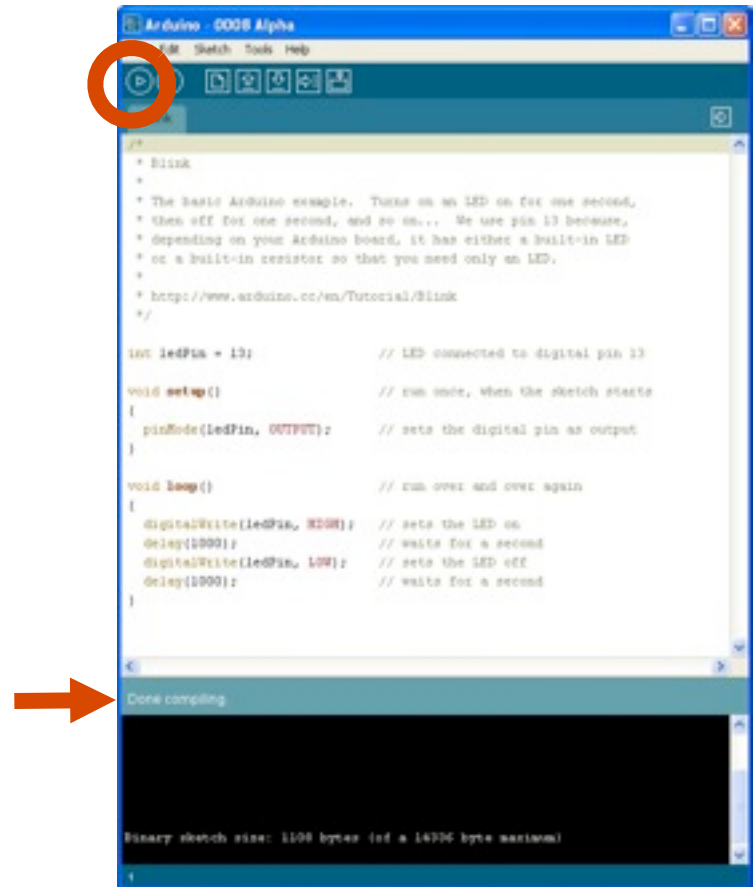
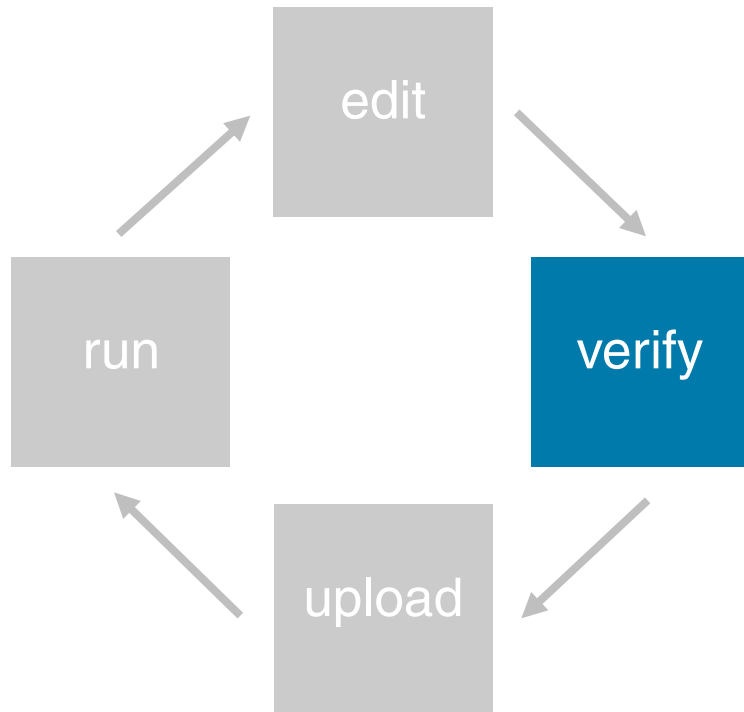
void loop()               // run over and over again
{
  digitalWrite(ledPin, HIGH); // sets the LED on
  delay(1000);                // waits for a second
  digitalWrite(ledPin, LOW);  // sets the LED off
  delay(1000);                // waits for a second
}
```

Below the code editor, the compilation output is shown:

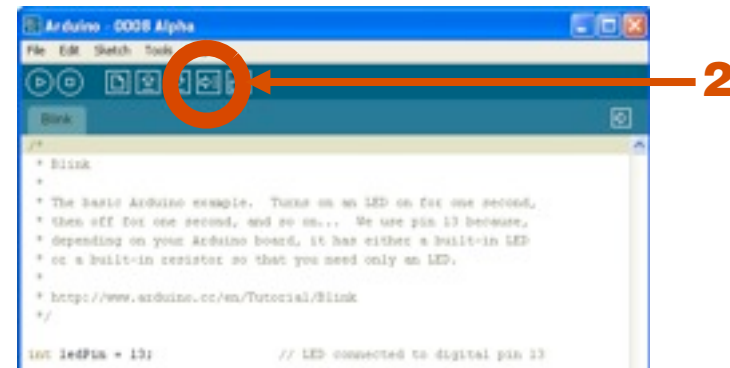
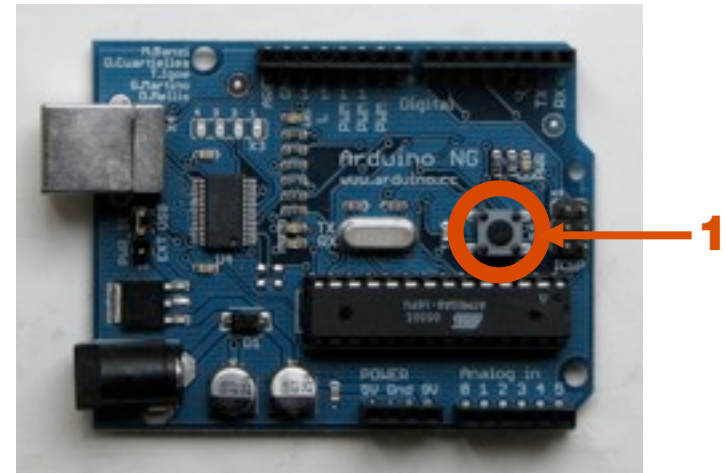
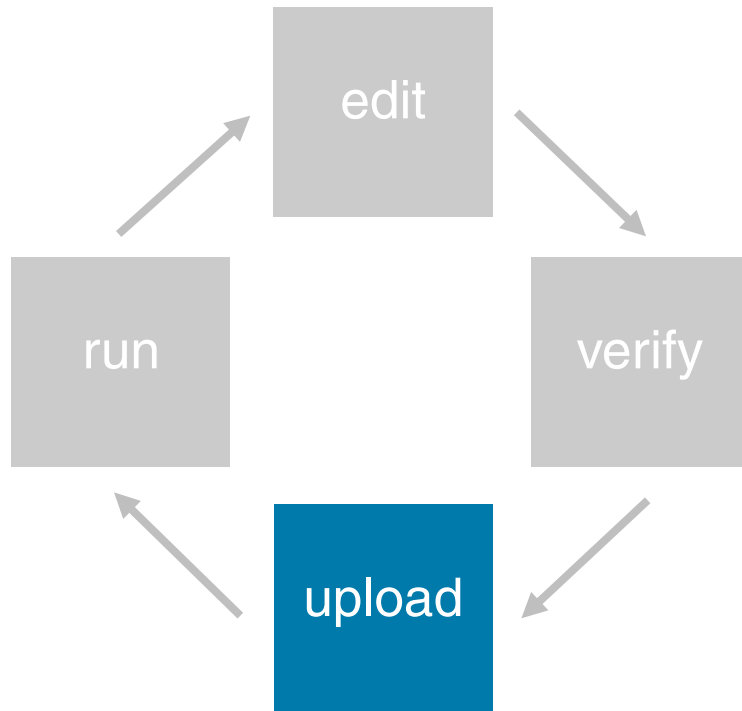
```
Done compiling

Binary sketch size: 1108 bytes (of a 14736 byte maximum)
```

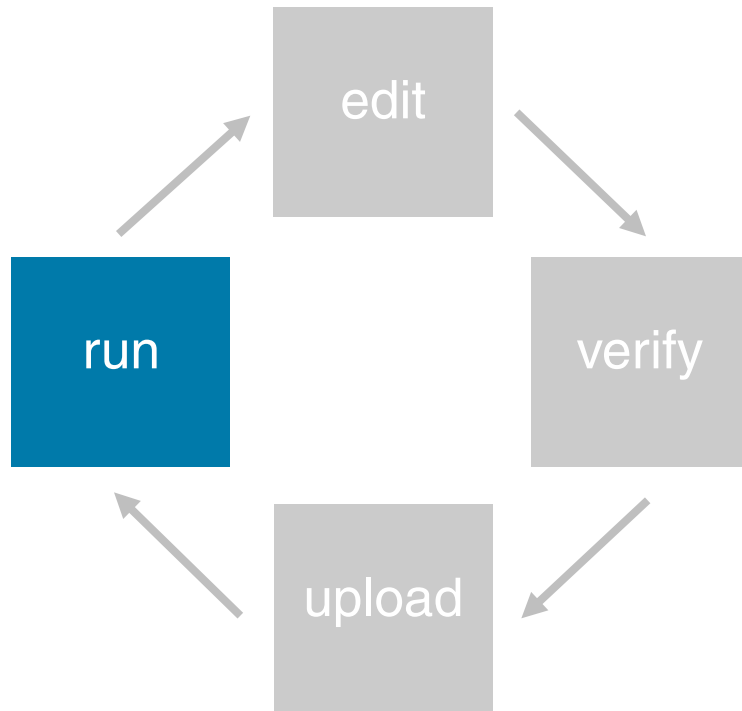
Arduino Programming Cycle



Arduino Programming Cycle



Arduino Programming Cycle



Watch your LED blink!

Arduino Program

Program consists of 3 parts:

1. **Declare** variables at top

2. **Initialize**

`setup()` – run once at beginning, set pins

3. **Run**

`loop()` – run repeatedly, after `setup()`

Arduino Language

Like C but easier

Example functions

- `pinMode()` – set a pin as input or output
- `digitalWrite()` – set a digital pin high/low
- `digitalRead()` – read a digital pin's state
- `analogRead()` – read an analog pin
- `analogWrite()` – write an “analog” PWM value
- `delay()` – wait an amount of time

Example (and many other examples at www.arduino.cc)

```
/*
 * Blink
 *
 * The basic Arduino example. Turns on an LED on for one second,
 * then off for one second, and so on... We use pin 13 because,
 * depending on your Arduino board, it has either a built-in LED
 * or a built-in resistor so that you need only an LED.
 *
 * http://www.arduino.cc/en/Tutorial/Blink
 */

int ledPin = 13;           // LED connected to digital pin 13

void setup()              // run once, when the sketch starts
{
  pinMode(ledPin, OUTPUT); // sets the digital pin as output
}

void loop()               // run over and over again
{
  digitalWrite(ledPin, HIGH); // sets the LED on
  delay(1000);                // waits for a second
  digitalWrite(ledPin, LOW);  // sets the LED off
  delay(1000);                // waits for a second
}
```

Example

 (and many other examples at www.arduino.cc)

```
/*
 * Blink
 *
 * The basic Arduino example. Turns on an LED on for one second,
 * then off for one second, and so on... We use pin 13 because,
 * depending on your Arduino board, it has either a built-in LED
 * or a built-in resistor so that you need only an LED.
 *
 * http://www.arduino.cc/en/Tutorial/Blink
 */

int ledPin = 13;           // LED connected to digital pin 13

void setup()              // run once, when the sketch starts
{
  pinMode(ledPin, OUTPUT); // sets the digital pin as output
}

void loop()               // run over and over again
{
  digitalWrite(ledPin, HIGH); // sets the LED on
  delay(1000);                // waits for a second
  digitalWrite(ledPin, LOW);  // sets the LED off
  delay(1000);                // waits for a second
}
```

Declare variables

Example

 (and many other examples at www.arduino.cc)

```
/*
 * Blink
 *
 * The basic Arduino example. Turns on an LED on for one second,
 * then off for one second, and so on... We use pin 13 because,
 * depending on your Arduino board, it has either a built-in LED
 * or a built-in resistor so that you need only an LED.
 *
 * http://www.arduino.cc/en/Tutorial/Blink
 */

int ledPin = 13;           // LED connected to digital pin 13

void setup()              // run once, when the sketch starts
{
  pinMode(ledPin, OUTPUT); // sets the digital pin as output
}

void loop()              // run over and over again
{
  digitalWrite(ledPin, HIGH); // sets the LED on
  delay(1000);               // waits for a second
  digitalWrite(ledPin, LOW);  // sets the LED off
  delay(1000);               // waits for a second
}
```

Initialize

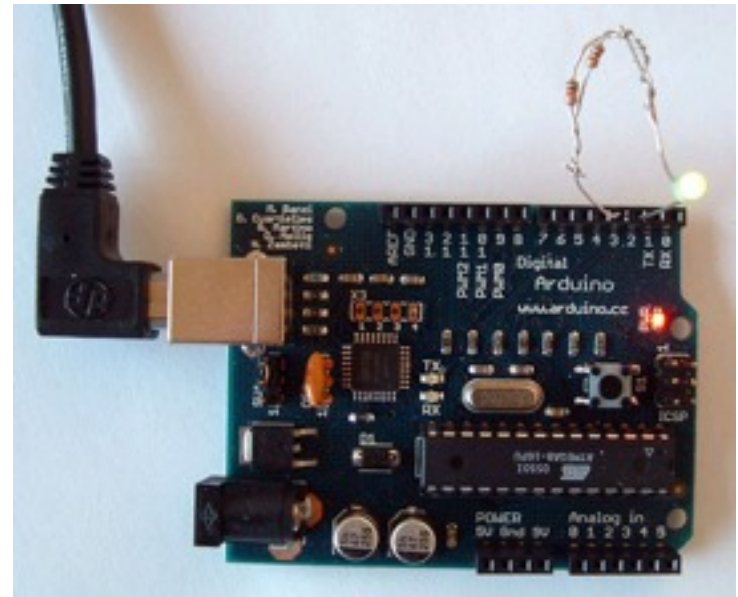
Example

 (and many other examples at www.arduino.cc)

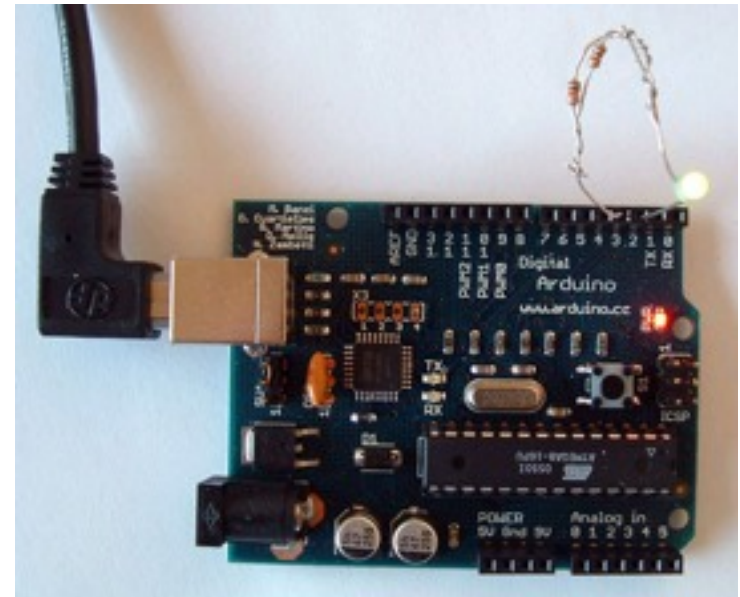
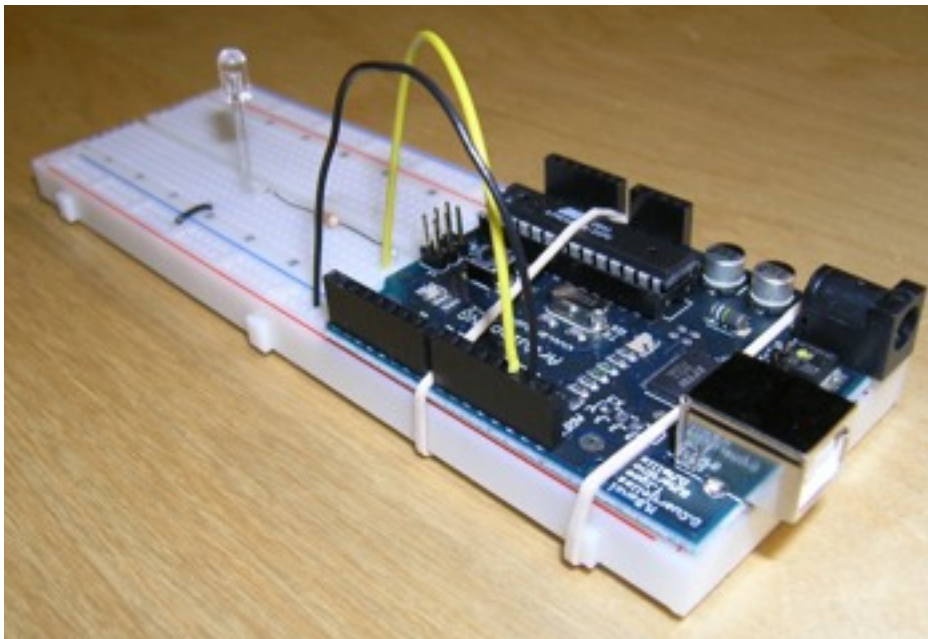
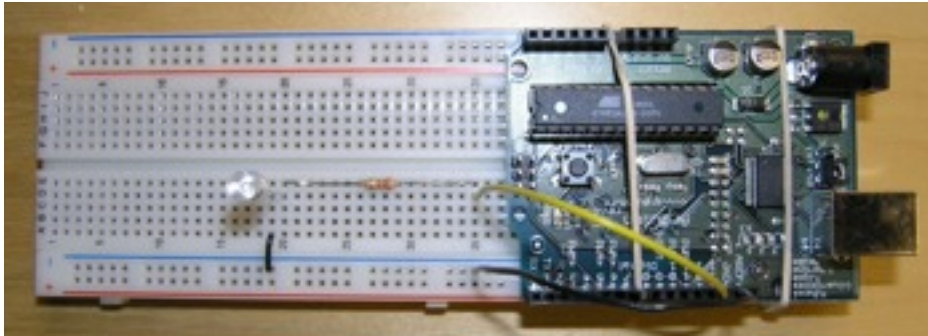
```
/*  
 * Blink  
 *  
 * The basic Arduino example. Turns on an LED on for one second,  
 * then off for one second, and so on... We use pin 13 because,  
 * depending on your Arduino board, it has either a built-in LED  
 * or a built-in resistor so that you need only an LED.  
 *  
 * http://www.arduino.cc/en/Tutorial/Blink  
 */  
  
int ledPin = 13;           // LED connected to digital pin 13  
  
void setup()              // run once, when the sketch starts  
{  
  pinMode(ledPin, OUTPUT); // sets the digital pin as output  
}  
  
void loop()                // run over and over again  
{  
  digitalWrite(ledPin, HIGH); // sets the LED on  
  delay(1000);                // waits for a second  
  digitalWrite(ledPin, LOW);  // sets the LED off  
  delay(1000);                // waits for a second  
}
```

Run

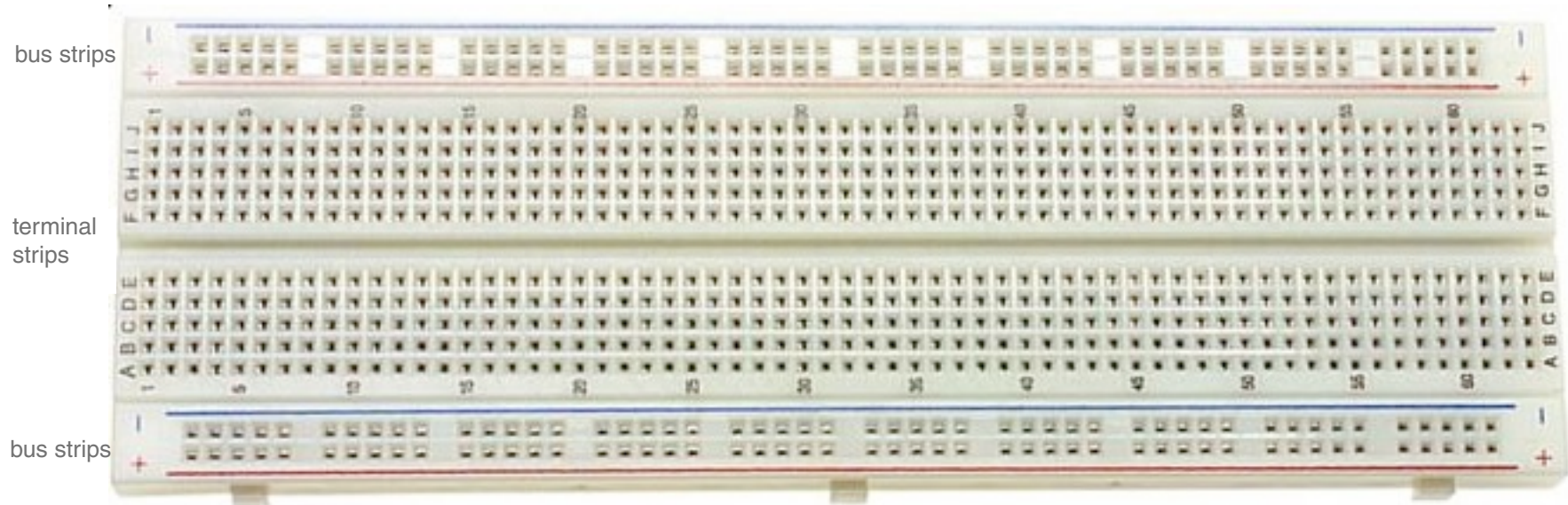
Arduino and Breadboard



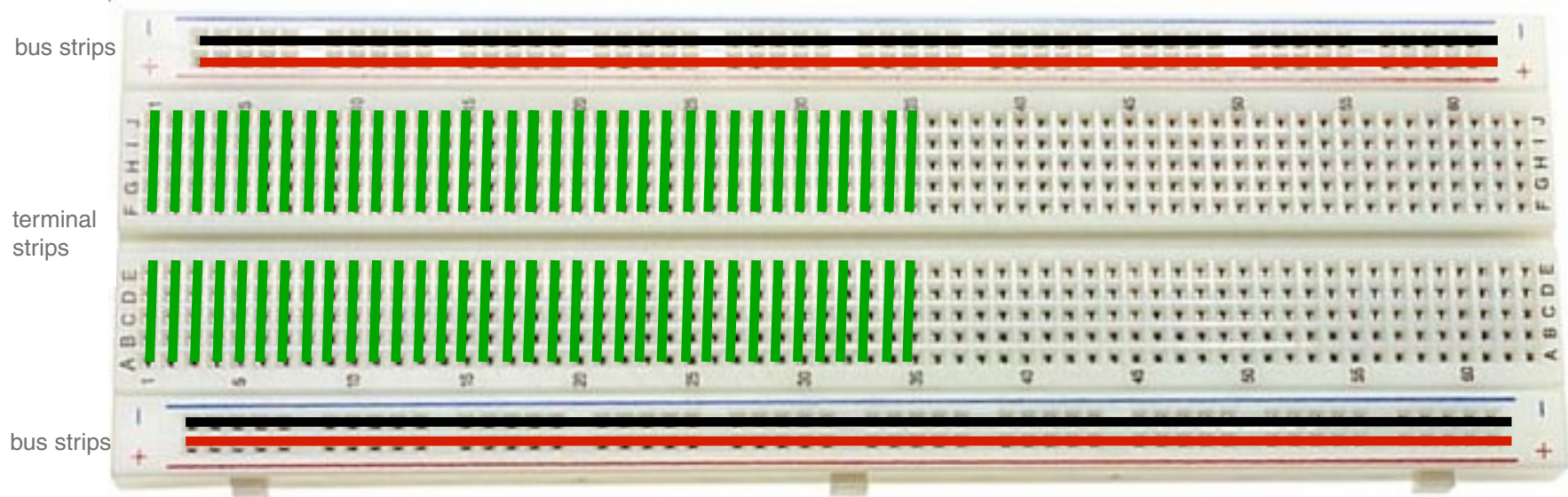
Arduino and Breadboard



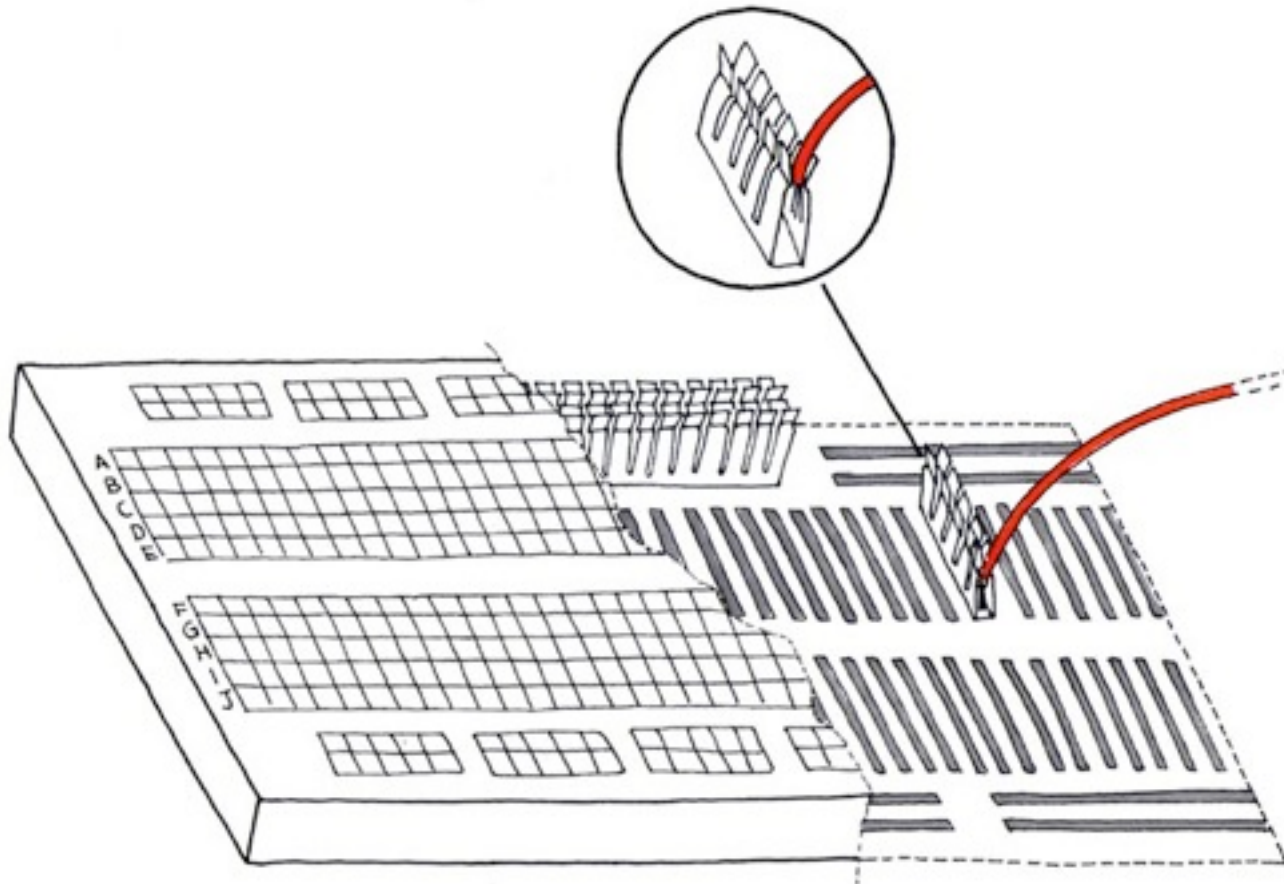
Solderless Breadboard



Solderless Breadboard

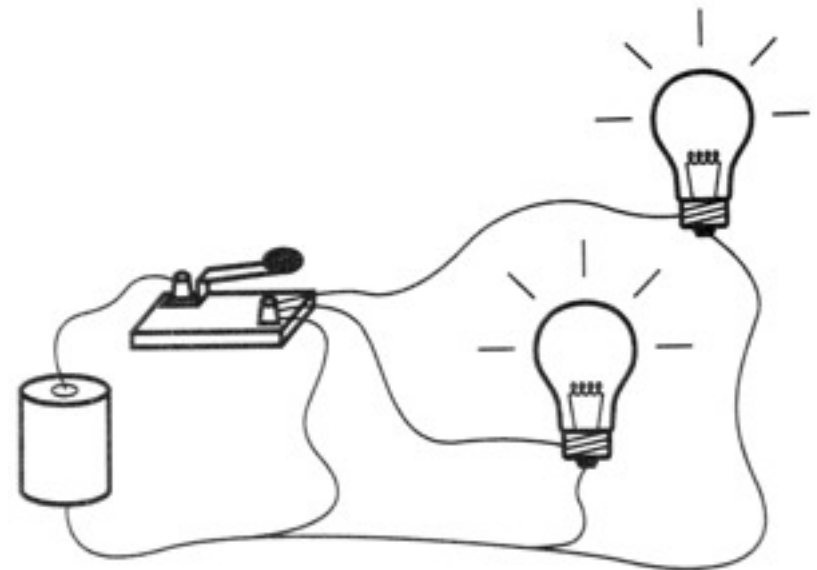
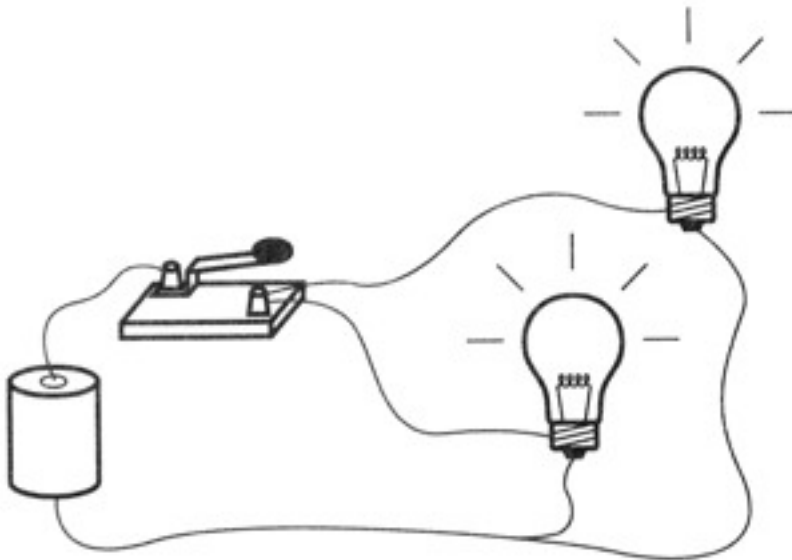


Solderless Breadboard



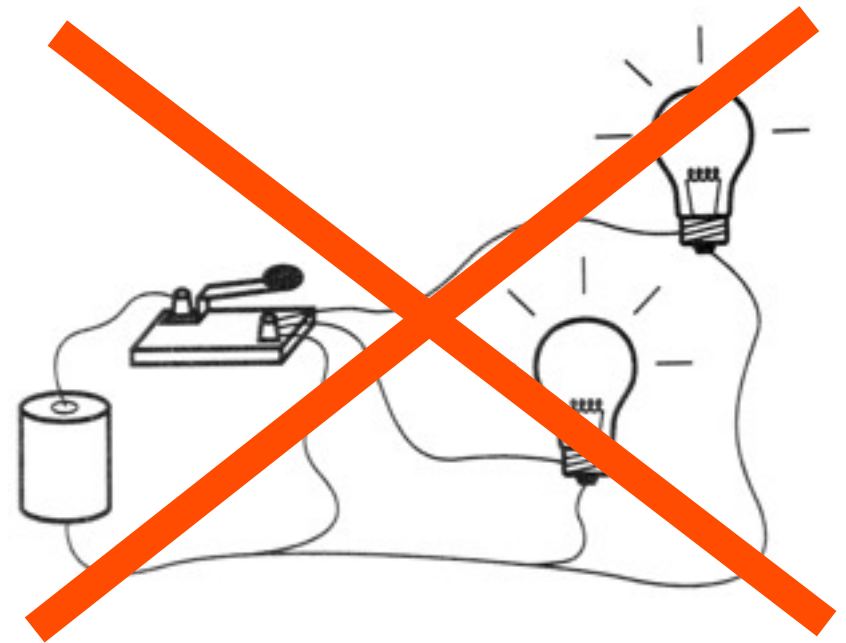
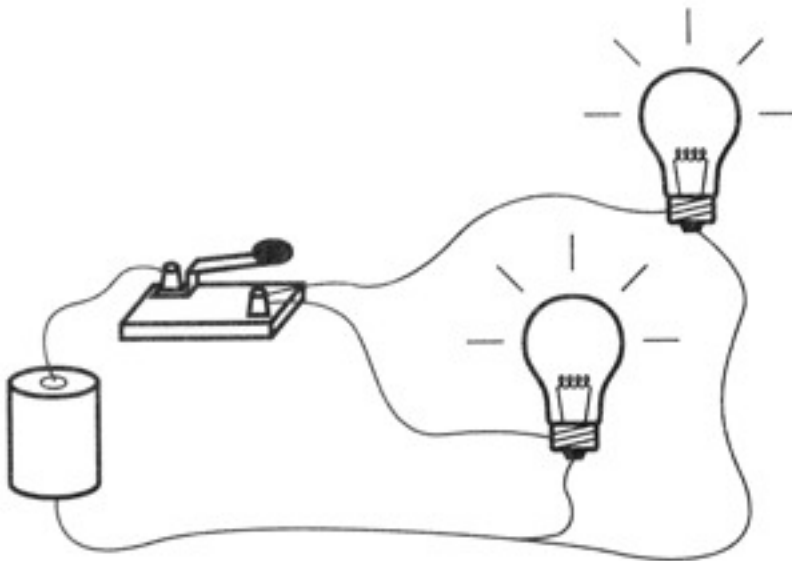
Circuits: Avoid Shortcuts

Electricity always favors the path of least resistance to ground



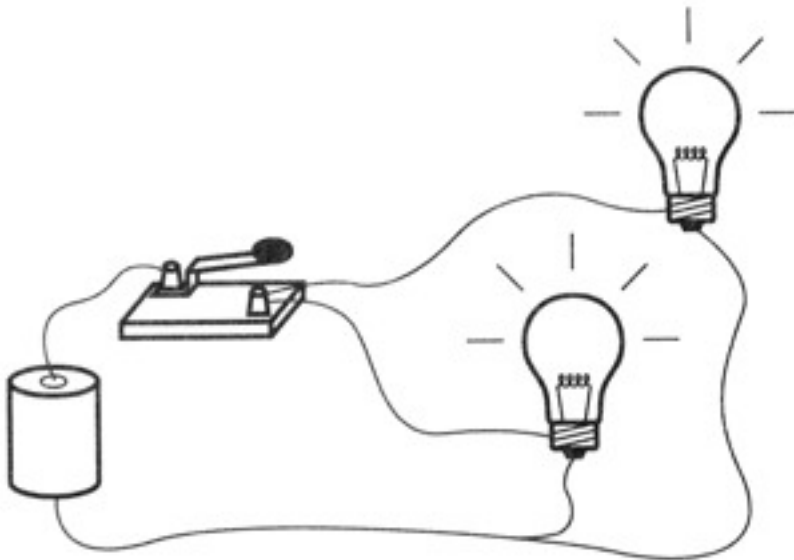
Circuits: Avoid Shortcuts

Electricity always favors the path of least resistance to ground



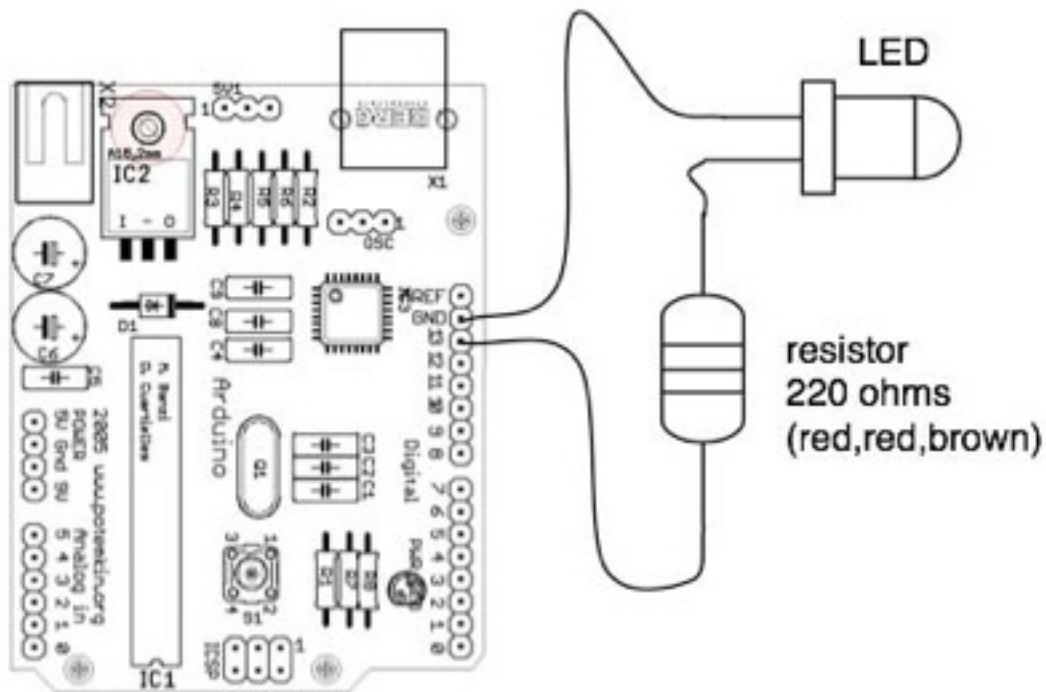
Circuits

All the electrical energy in a circuit must be used



LED

- LED = Light-Emitting Diode
- Needs a “current limiting” resistor, or burns out



Circuits Summary

Avoid deadly shortcuts

- **Flows to the lowest resistance**
- **All the electrical energy in a circuit must be used**

When in doubt, talk to us, we can help

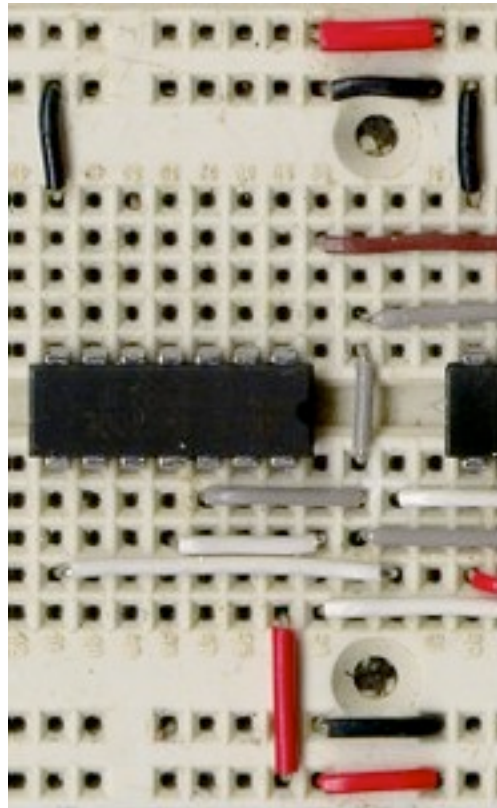
(Your Arduino has some fail safe, but in the worst case, you could fry your board [\$29])

Try to be Neat

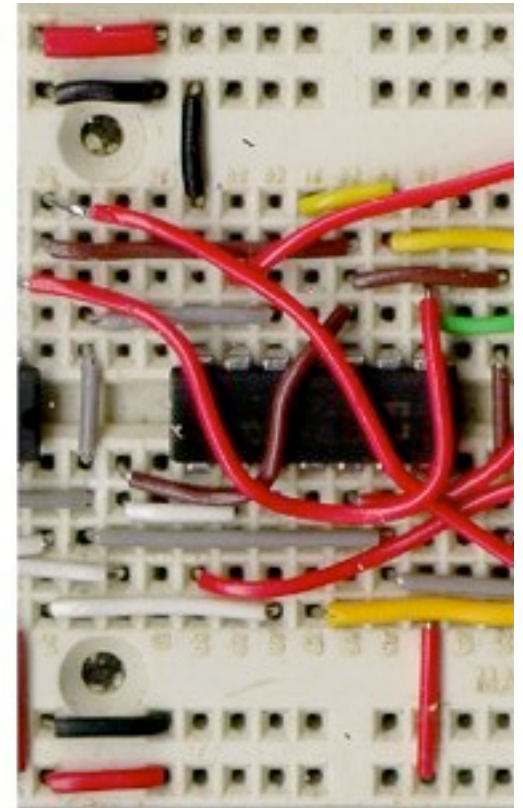
Color code:

- **Red**: power
- **Black**: ground

Be consistent

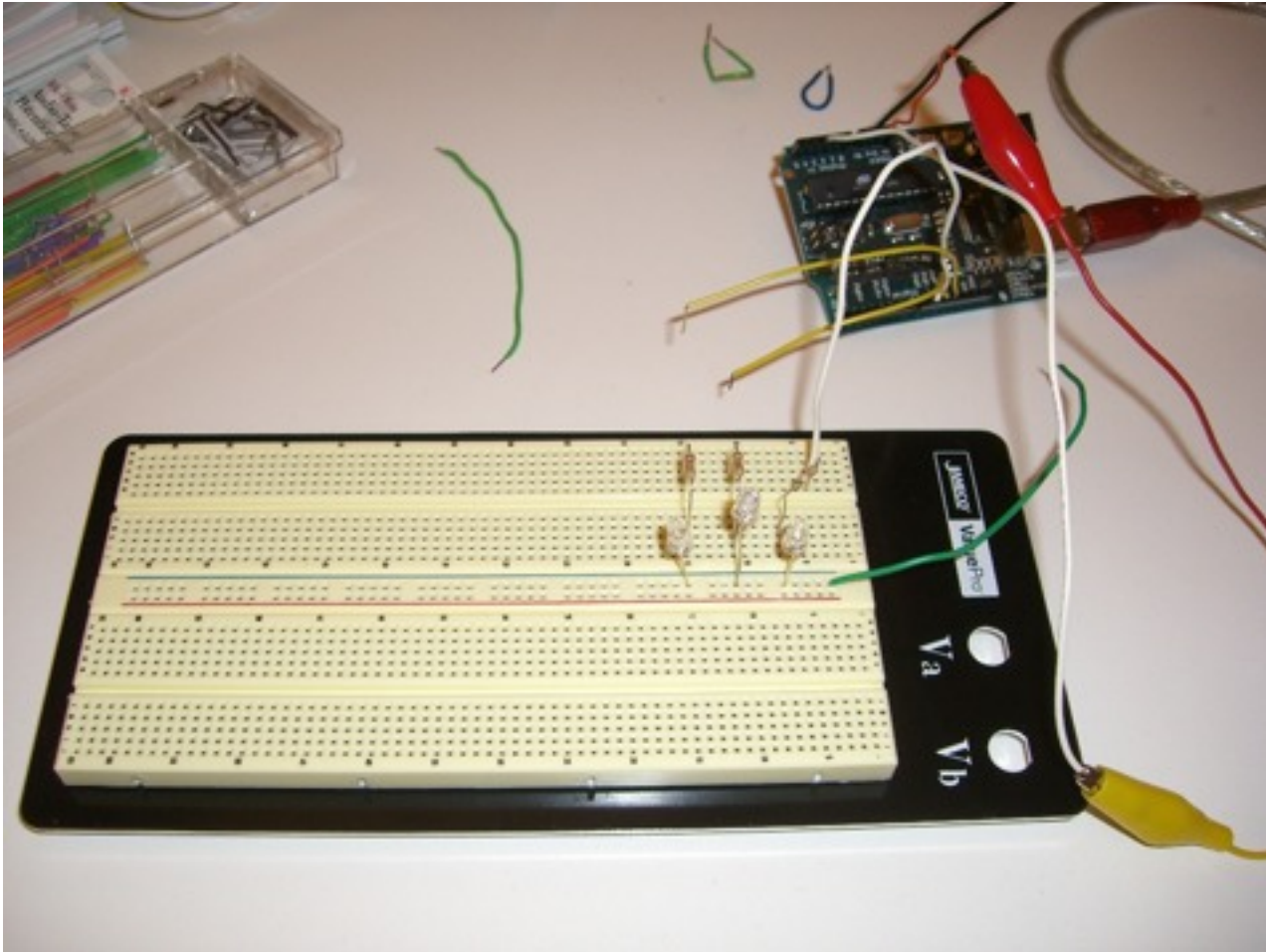


Good



Bad

Be Careful

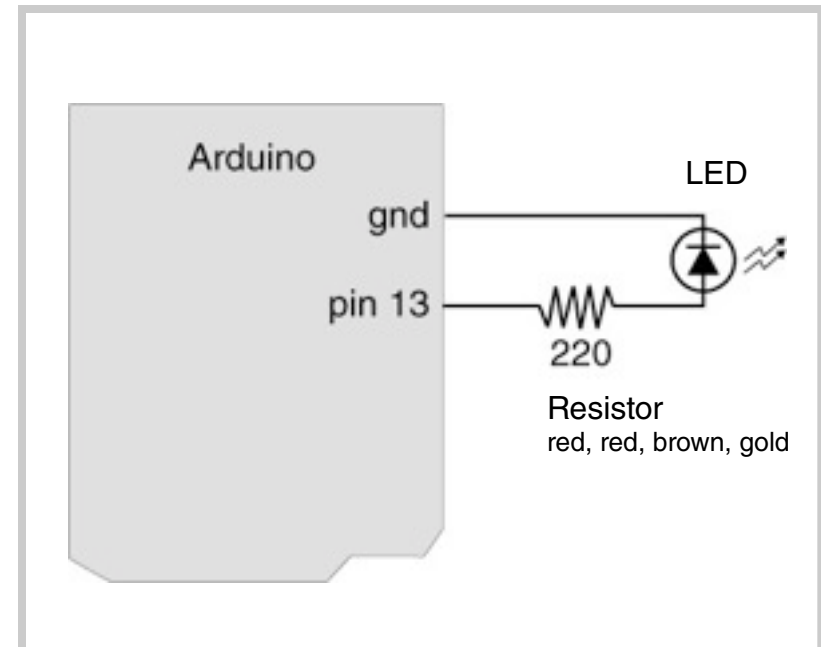


Lab Assignment This Week

- Get the course lab kit
- Download & install Arduino software (www.arduino.cc)
- Make an LED blink
 - Make it blink at different rate
- Create your course web account
- Post the photo of your board and an optional comment on the course website

Lab Assignment: Blinking LED

```
/*  
 * Blink  
 *  
 * The basic Arduino example. Turns on an LED on for one second,  
 * then off for one second, and so on... We use pin 13 because,  
 * depending on your Arduino board, it has either a built-in LED  
 * or a built-in resistor so that you need only an LED.  
 *  
 * http://www.arduino.cc/en/Tutorial/Blink  
 */  
  
int ledPin = 13;           // LED connected to digital pin 13  
  
void setup()              // run once, when the sketch starts  
{  
  pinMode(ledPin, OUTPUT); // sets the digital pin as output  
}  
  
void loop()               // run over and over again  
{  
  digitalWrite(ledPin, HIGH); // sets the LED on  
  delay(1000);                // waits for a second  
  digitalWrite(ledPin, LOW);  // sets the LED off  
  delay(1000);                // waits for a second  
}
```



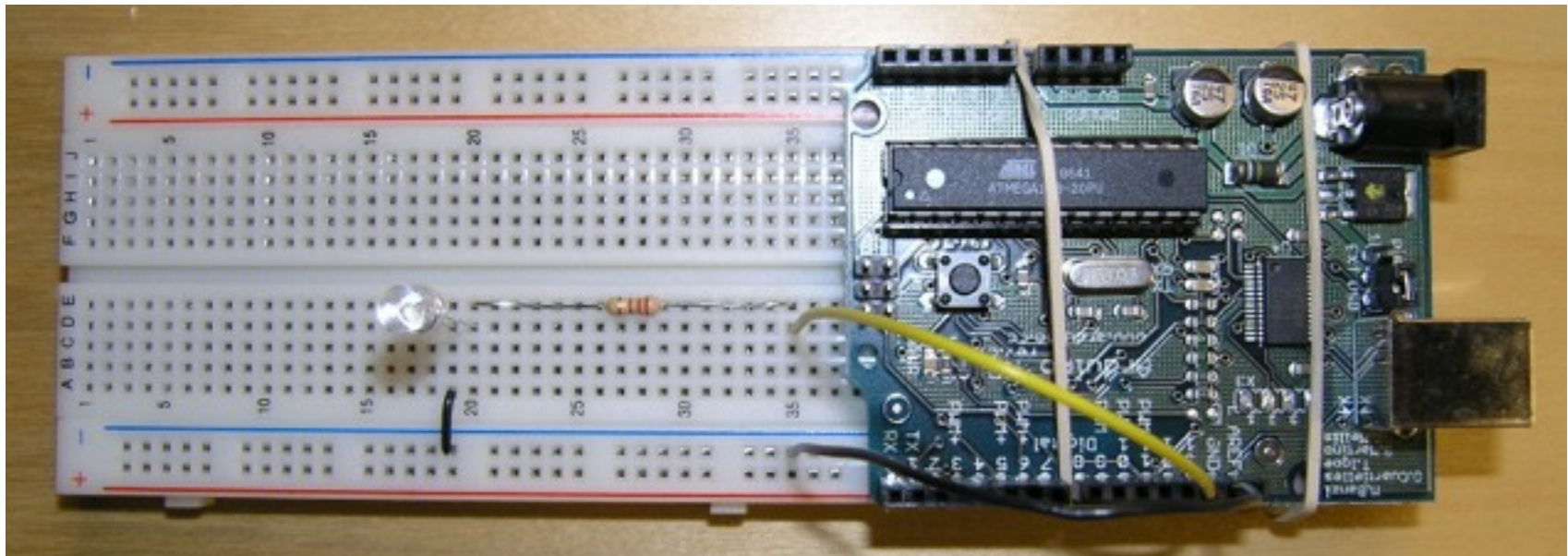
Your assignment looks like this



LED
Shorter leg → ground
Polarity matters



220 ohm resistor
red, red, brown, gold
Polarity does not matter



Next Wednesday

Don't forget to bring your lab kit in

Do use the lab hour to catch up

Create your course account

Post your assignment on the course website

Thanks!