PHYSICAL PROTOTYPING

Sketching in Hardware

ABOUT ME

Dr. Rafael 'Tico' Ballagas

Principal Scientist,
Nokia Research Center
Palo Alto, USA

PHYSICAL PROTOTYPING IN INDUSTRIAL RESEARCH

Family Story Play
Pop goes the cell phone

FAMILY STORY PLAY

POP GOES THE CELL-PHONE

OUTLINE

- Why is prototyping important?
- Arduino
- Digital I/O
- Analog Sensors
- Actuators
- State of the art research

WHAT IS A PROTOTYPE?

- Prototyping structures innovation, collaboration, and creativity in the most successful design studios (Kelley and Littman, 2001)

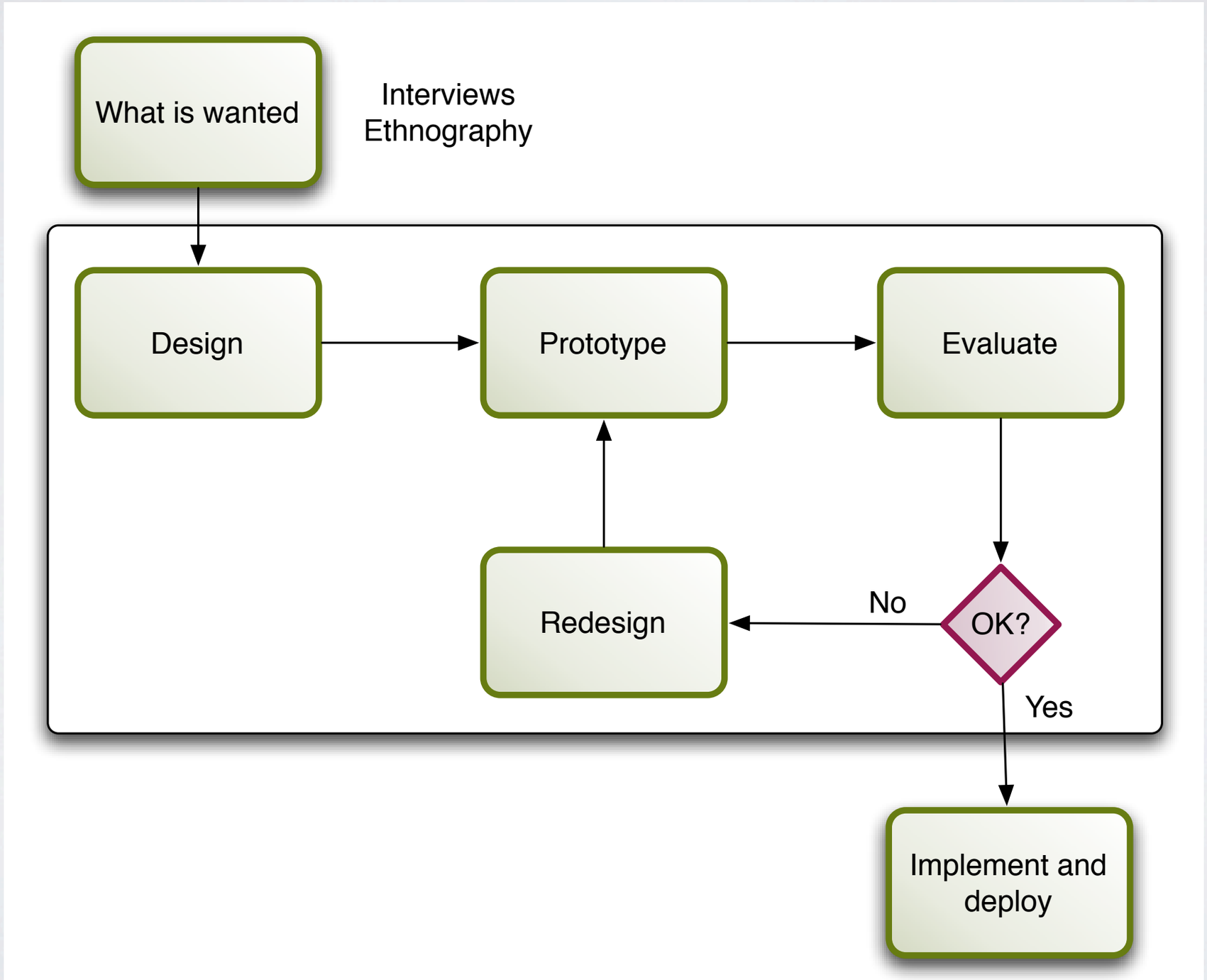
- Designers use prototypes as physical representations of ideas, effectively externalizing cognition and facilitating a "conversation with materials" to uncover surprising problems or generate suggestions for new designs (Schön and Bennett, 1996).

- Prototypes also serve as artifacts that represent tacit knowledge of developers as a communication tool to clients or other members of a design team (Schrage, 1999).

THE NATURE OF PROTOTYPING

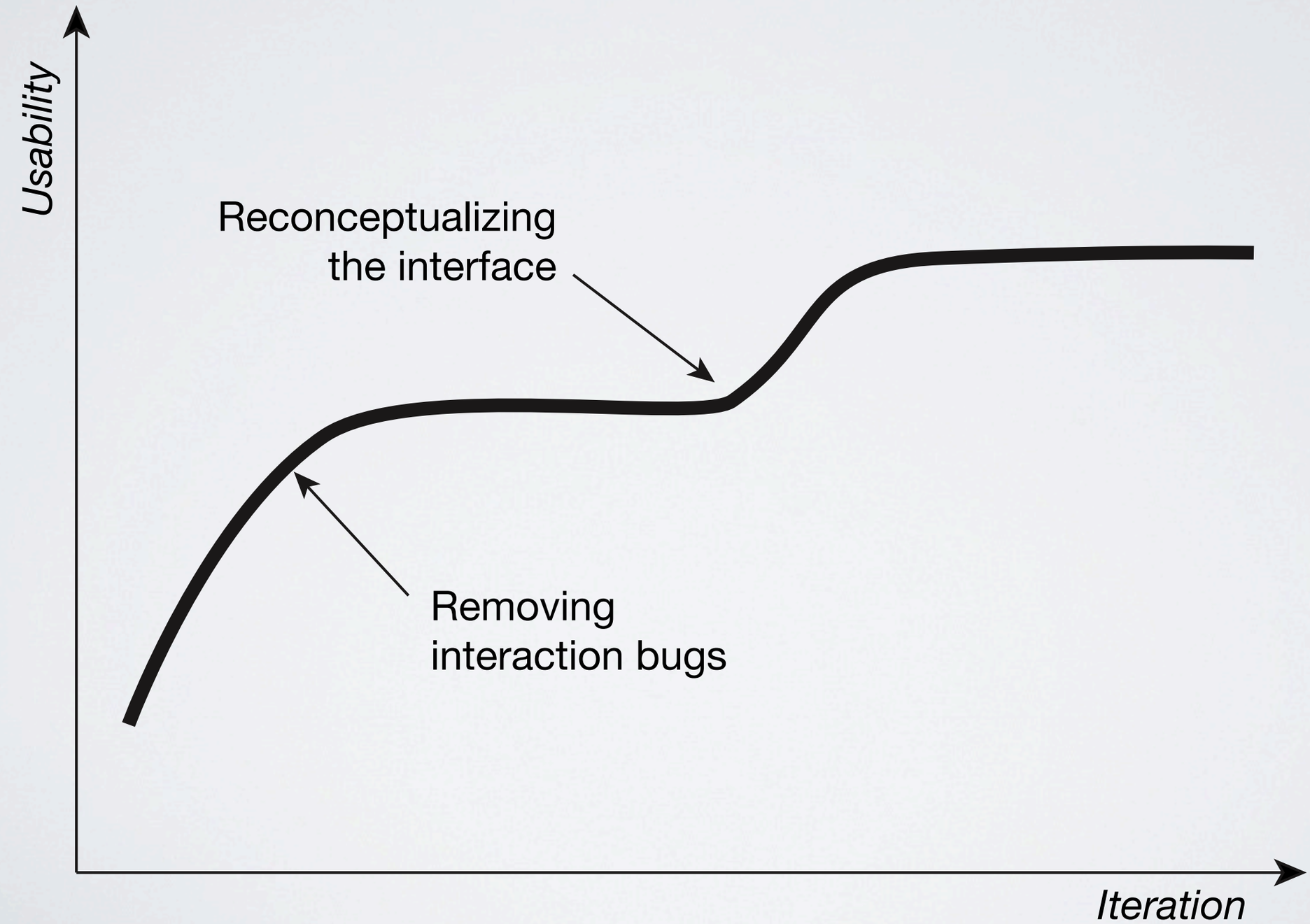
- Problems and solutions co-evolve (Dorst & Cross 2011)
- Subproblems are interconnected (Goel and Pirolli 1992)
- Constraints are often negotiable (Schön 1995)
- Solutions are not right or wrong, only better or worse (Rittle and Webber 1973)

ITERATIVE DESIGN



(Nielsen, 1993)

MORE ITERATIONS = BETTER DESIGNS



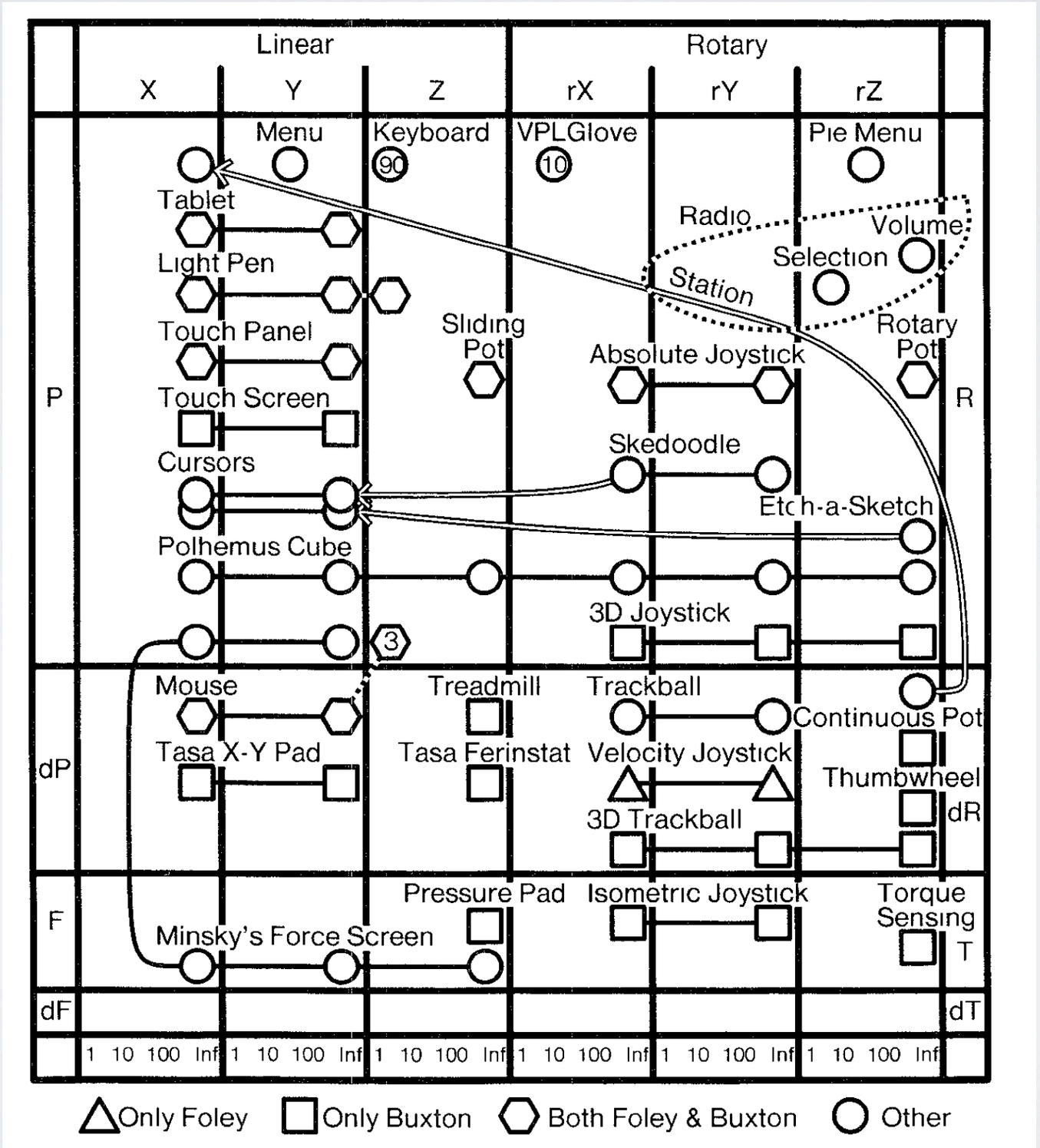
PARALLEL PROTOTYPING LEADS TO BETTER DESIGN

[Dow, 2010]

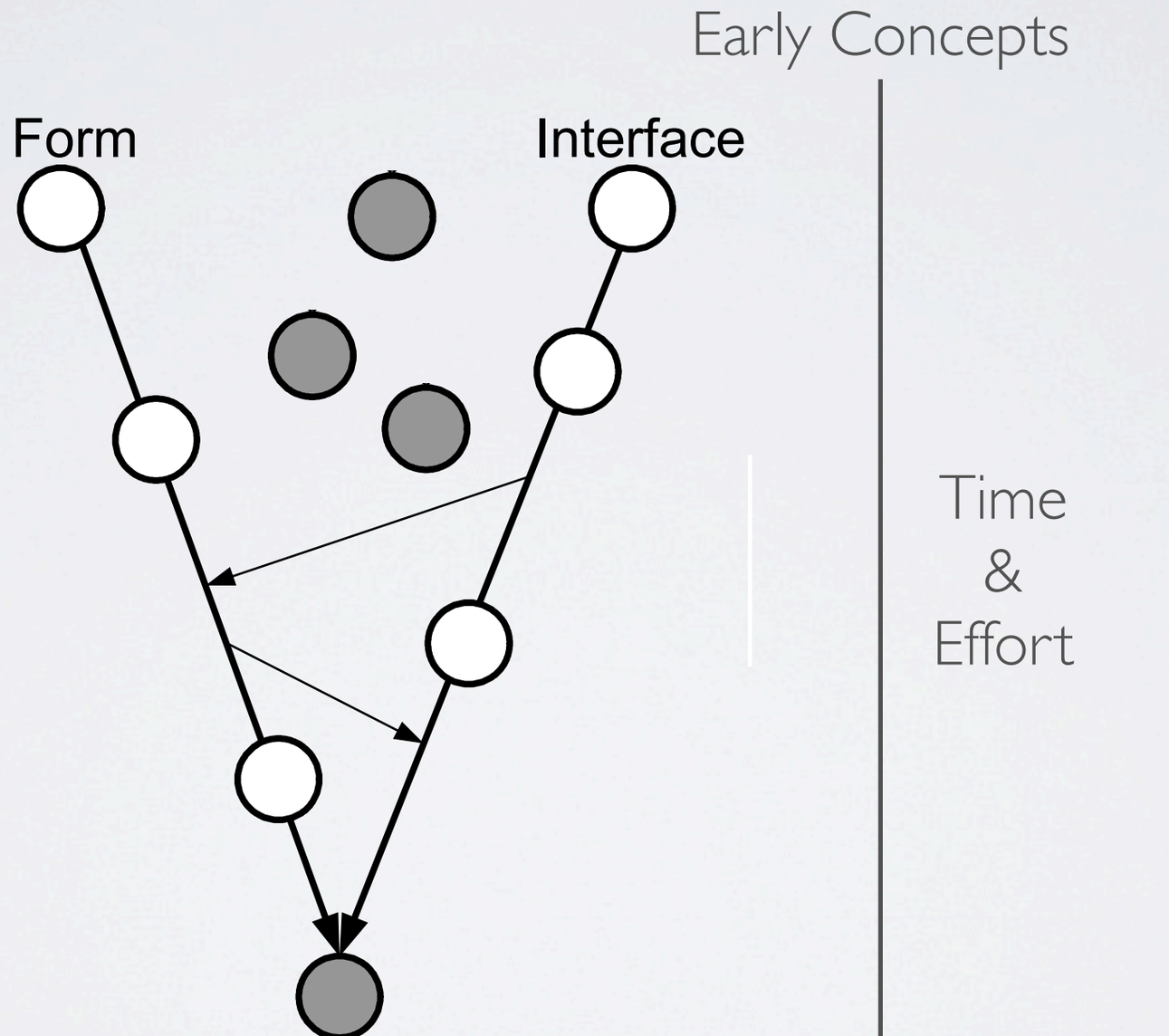
- Study comparing parallel to serial prototyping
- Parallel prototyping outperformed Serial by all objective performance measures
- Parallel prototypers had more divergent ideas
- Parallel prototypers react more positively toward critique

DESIGN SPACE OF INPUT DEVICES

[Card, 1991]



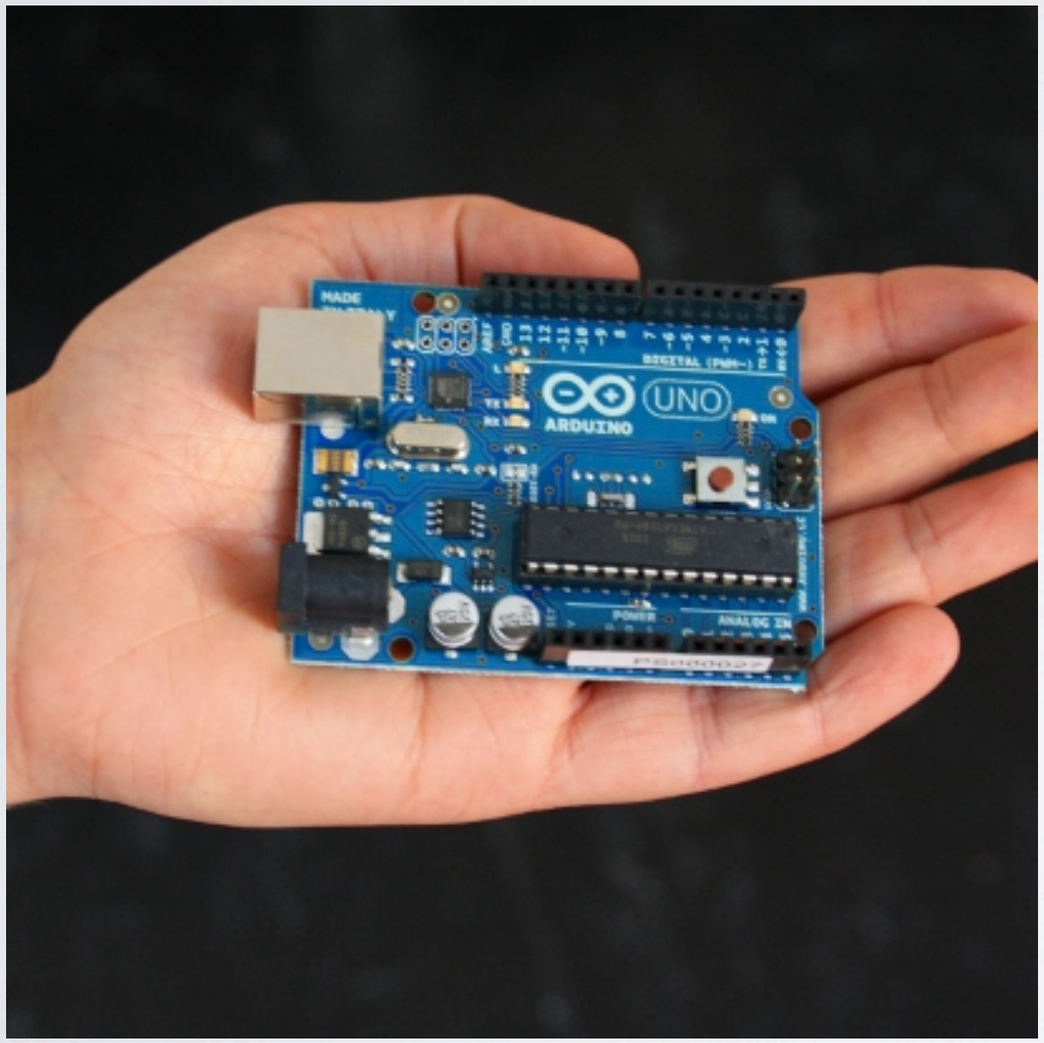
TRADITIONAL PROTOTYPING



PROTOTYPING KITS

ARDUINO UNO BOARD

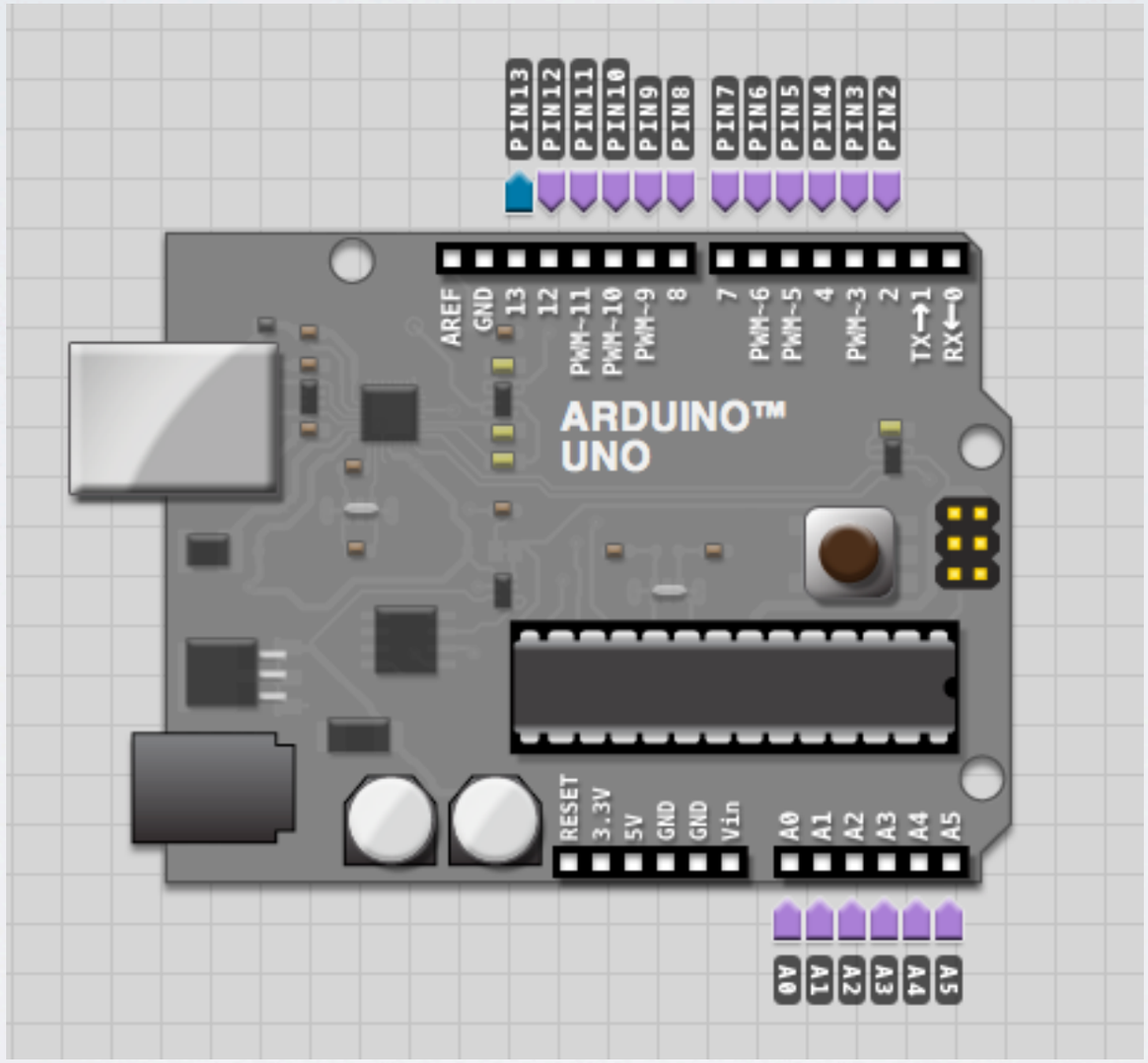
http://www.arduino.cc



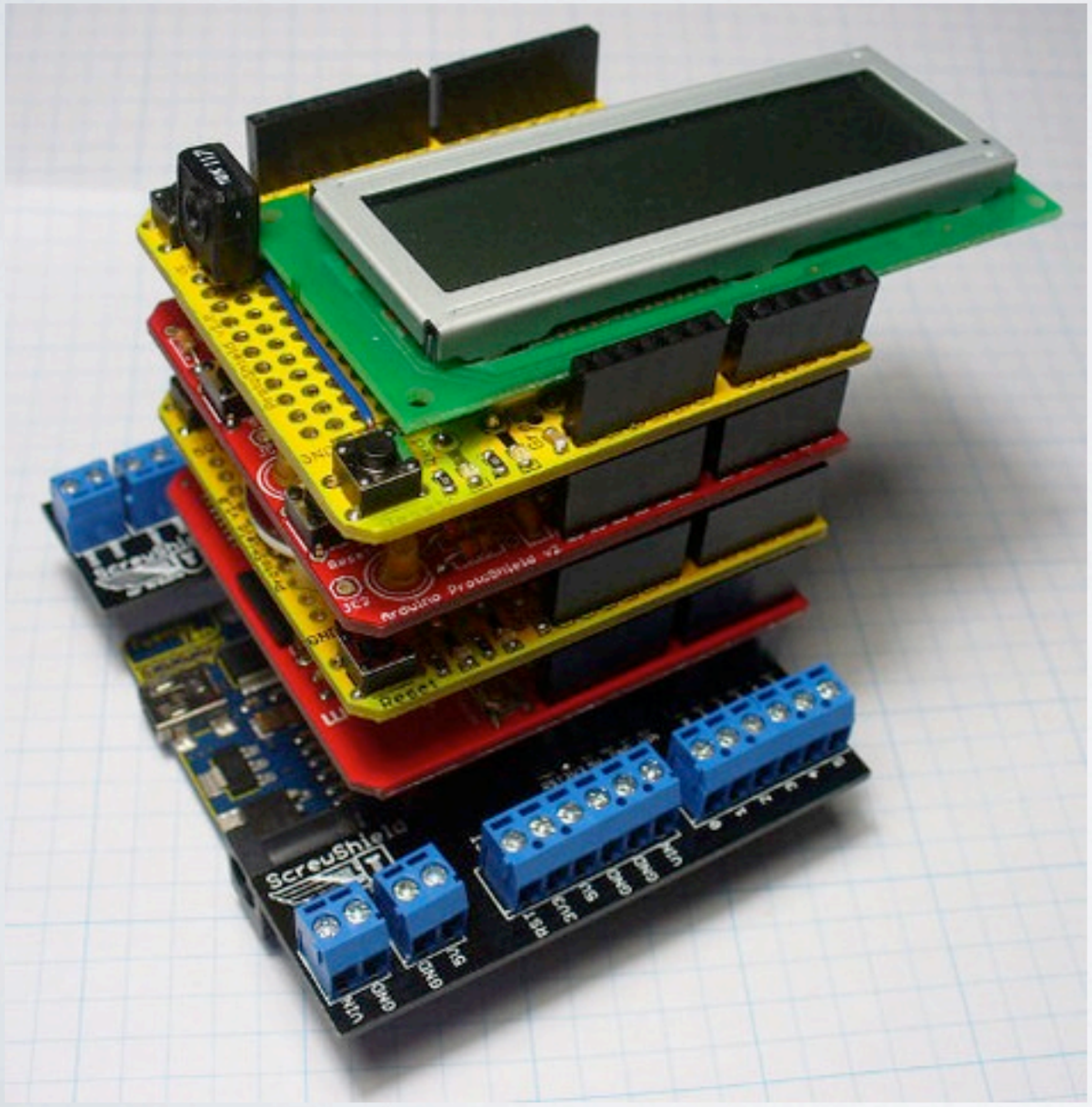
Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

ARDUINO I/O


PIN13
PIN12
PIN11
PIN10
PIN9
PIN8
PIN7
PIN6
PIN5
PIN4
PIN3
PIN2
AREF
GND
13
12
PWM~11
PWM~10
PWM~9
8
7
PWM~6
PWM~5
4
PWM~3
2
TX→1
RX←0
ARDUINO™
UNO
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5
A0
A1
A2
A3
A4
A5

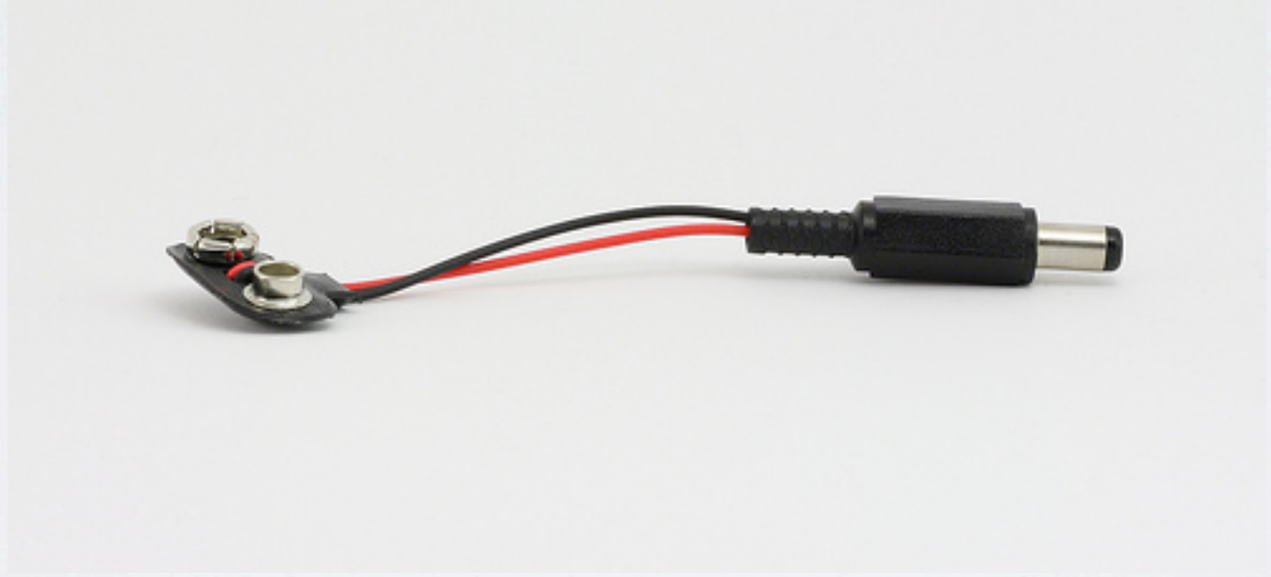
EXTENDING ARDUINO



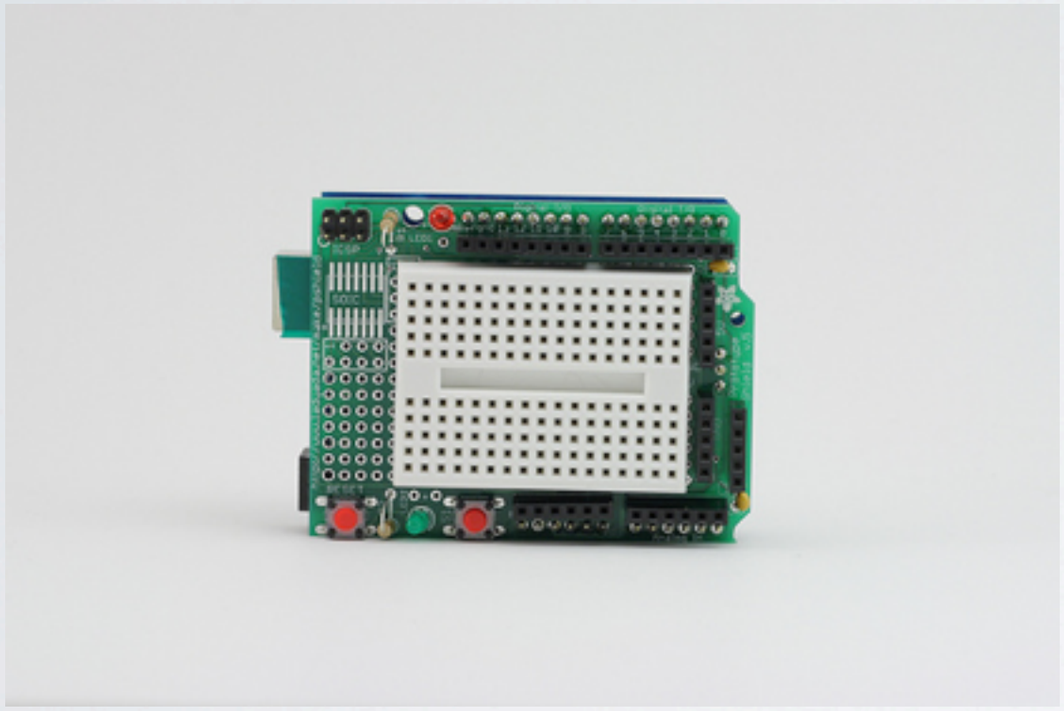
- Ethernet shield
- XBee shield
- Motor shield

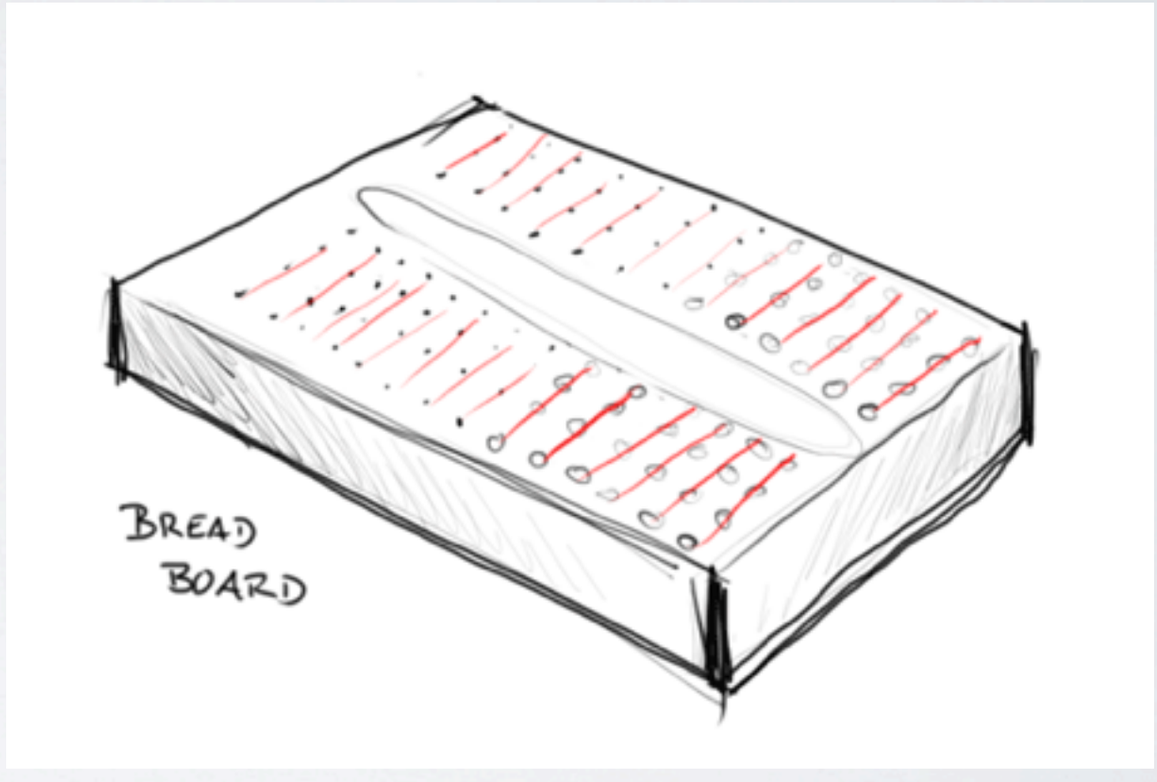
USB CABLE

9V BATTERY CONNECTOR

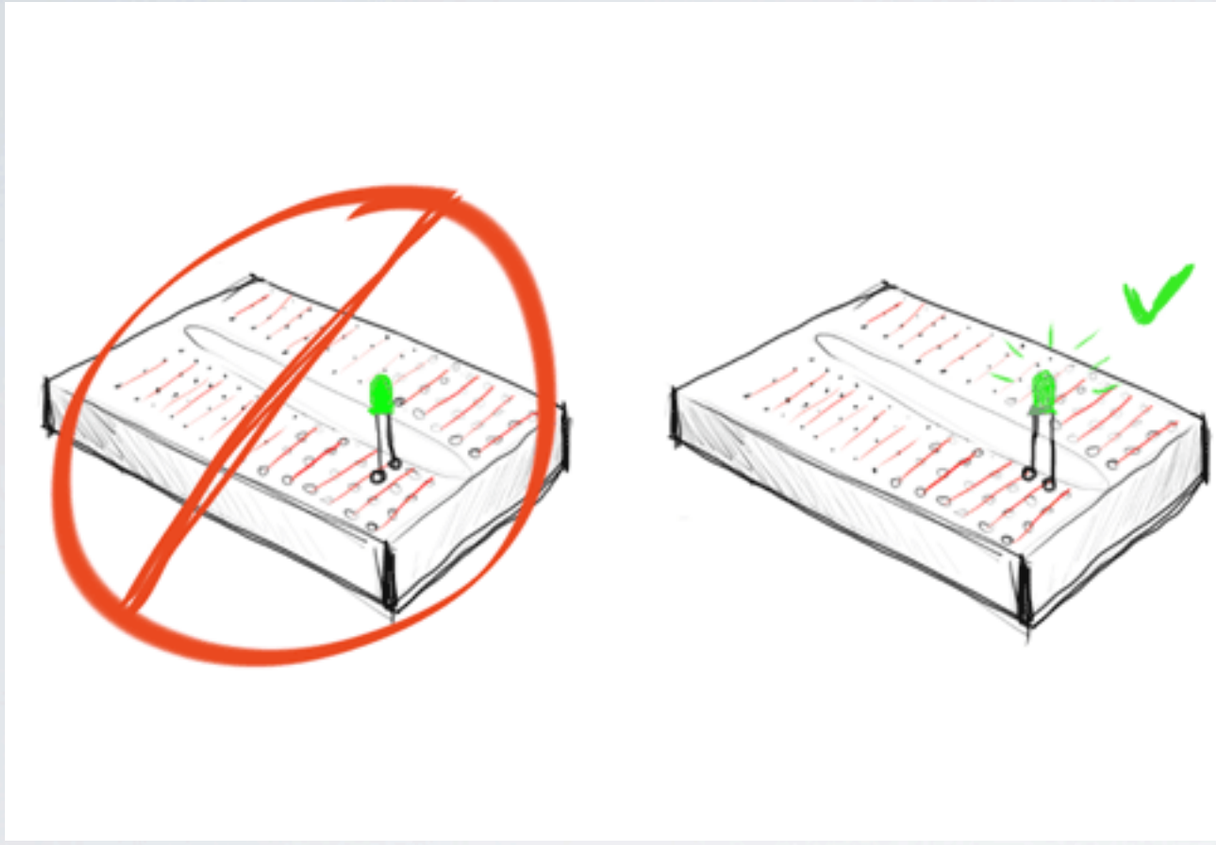


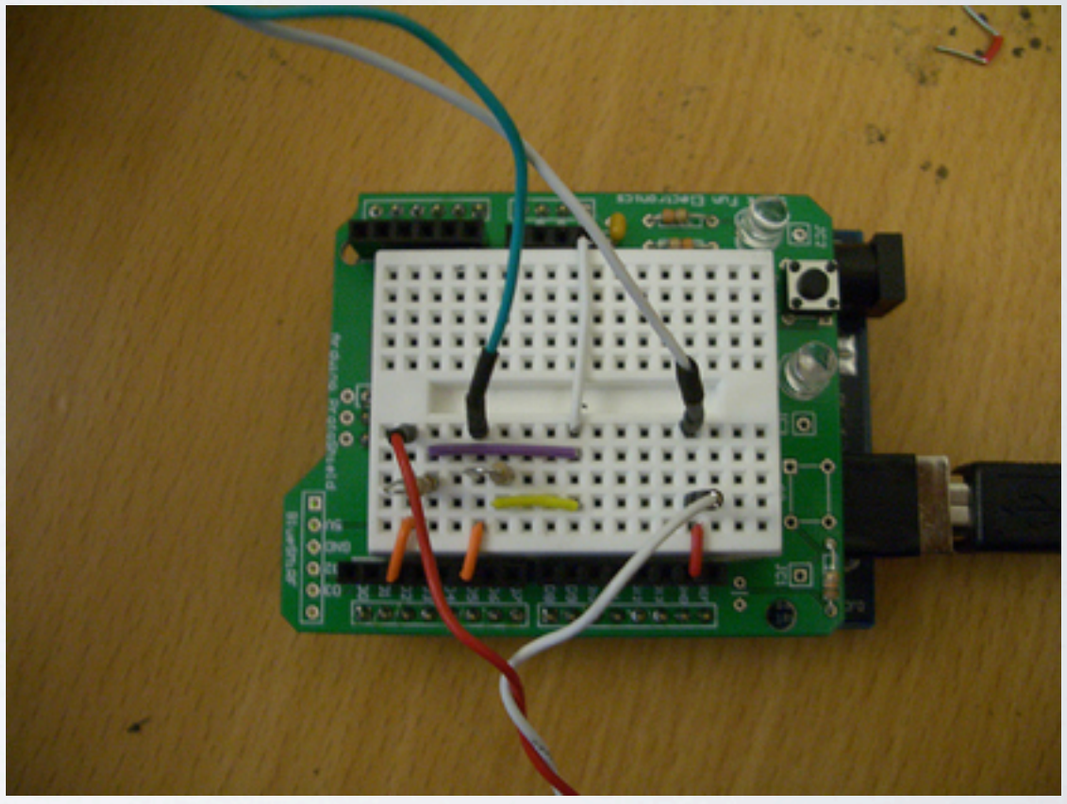
BREADBOARDS





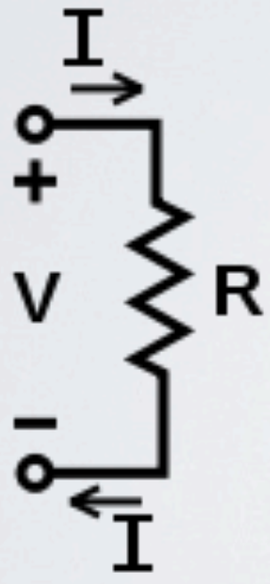
BREADBOARDS IN USE





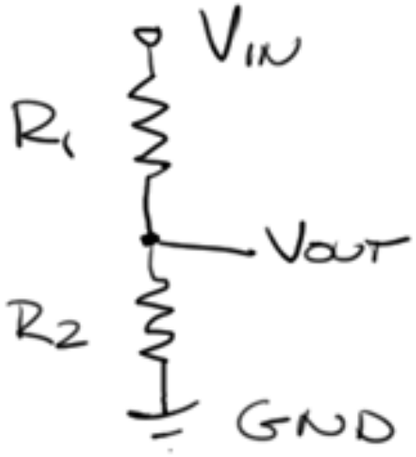
VOLTAGE, CURRENT, RESISTANCE

Ohm's law



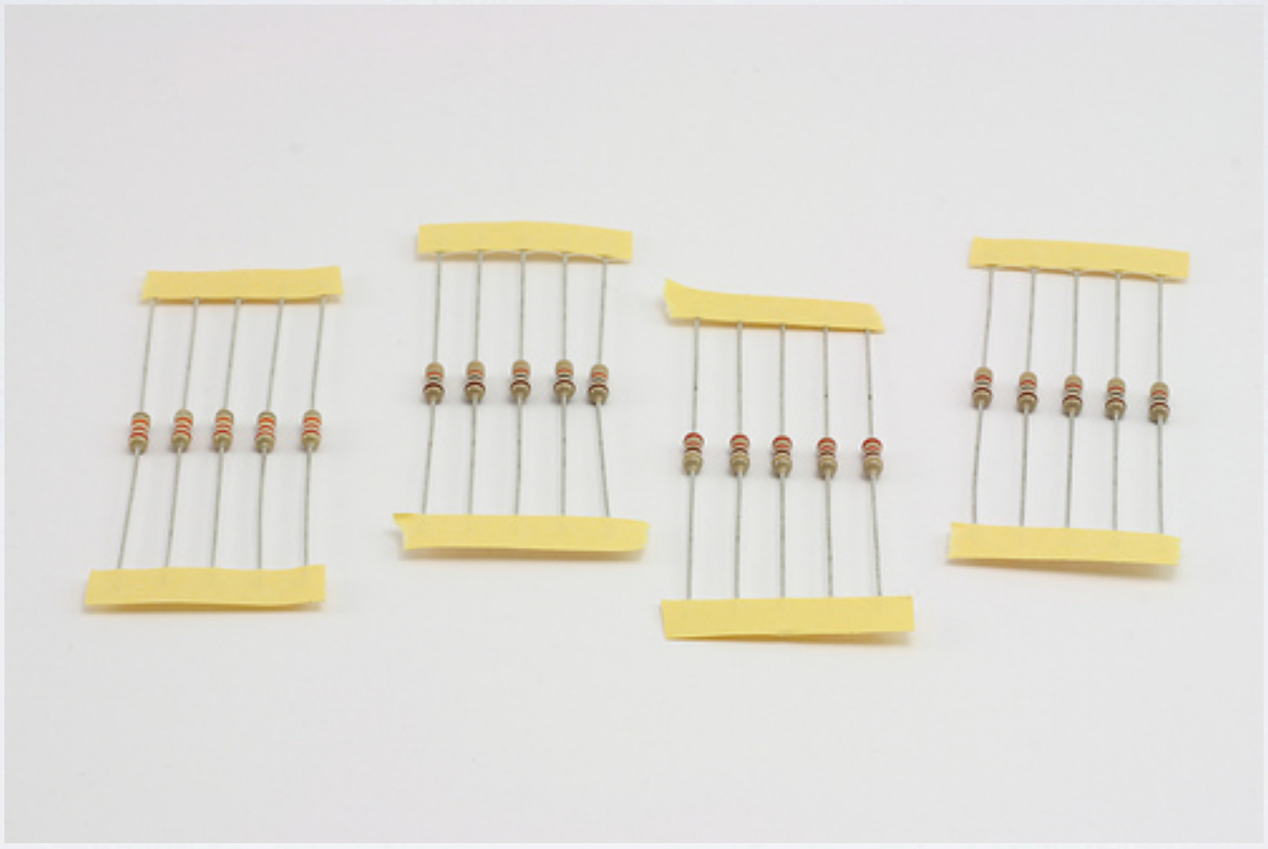
$V=IR$

Voltage Divider



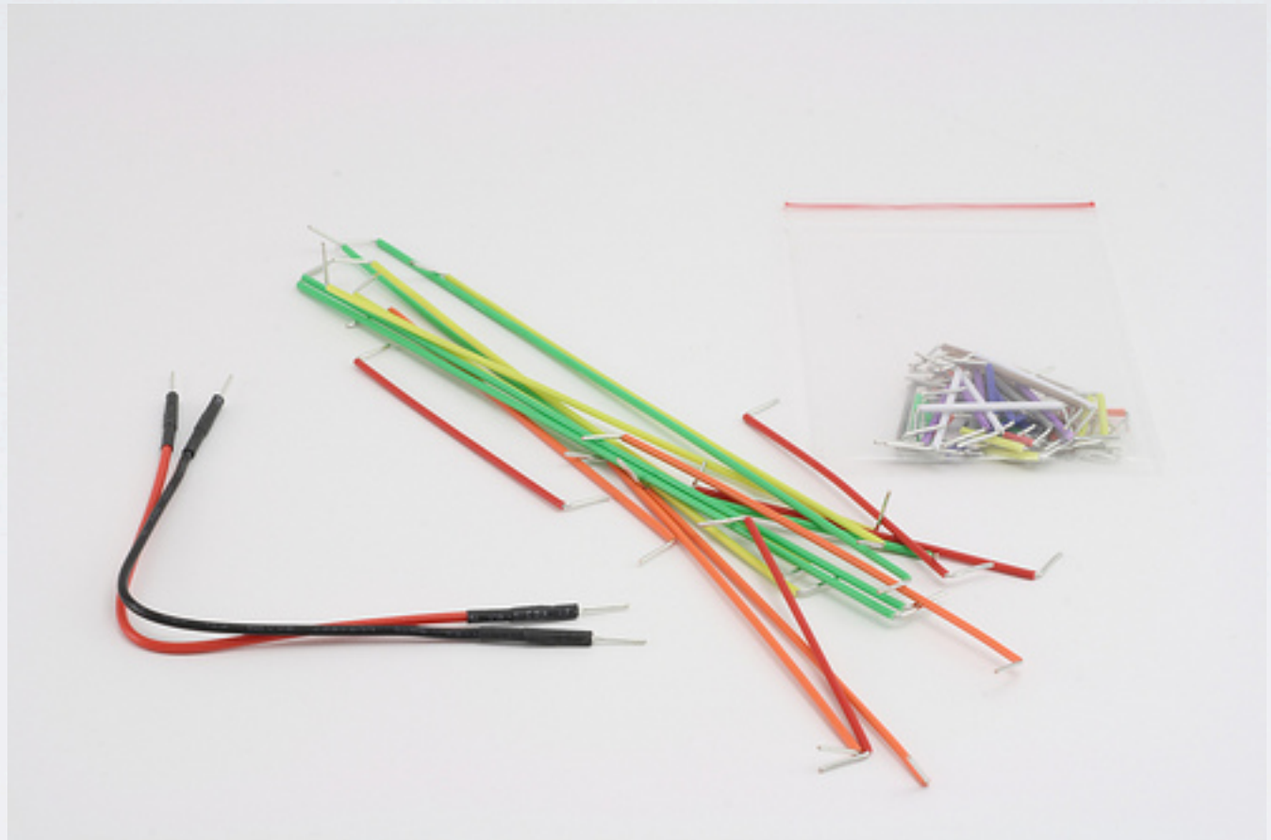
$$V_{out} = \frac{R_2}{R_1 + R_2} \cdot V_{IN}$$

RESISTORS



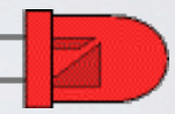
Resistor Color Codes: http://www.kpsec.freeuk.com/components/resist.htm
Resistor Calculator: http://www.dannyg.com/examples/res2/resistor.htm

JUMPERS

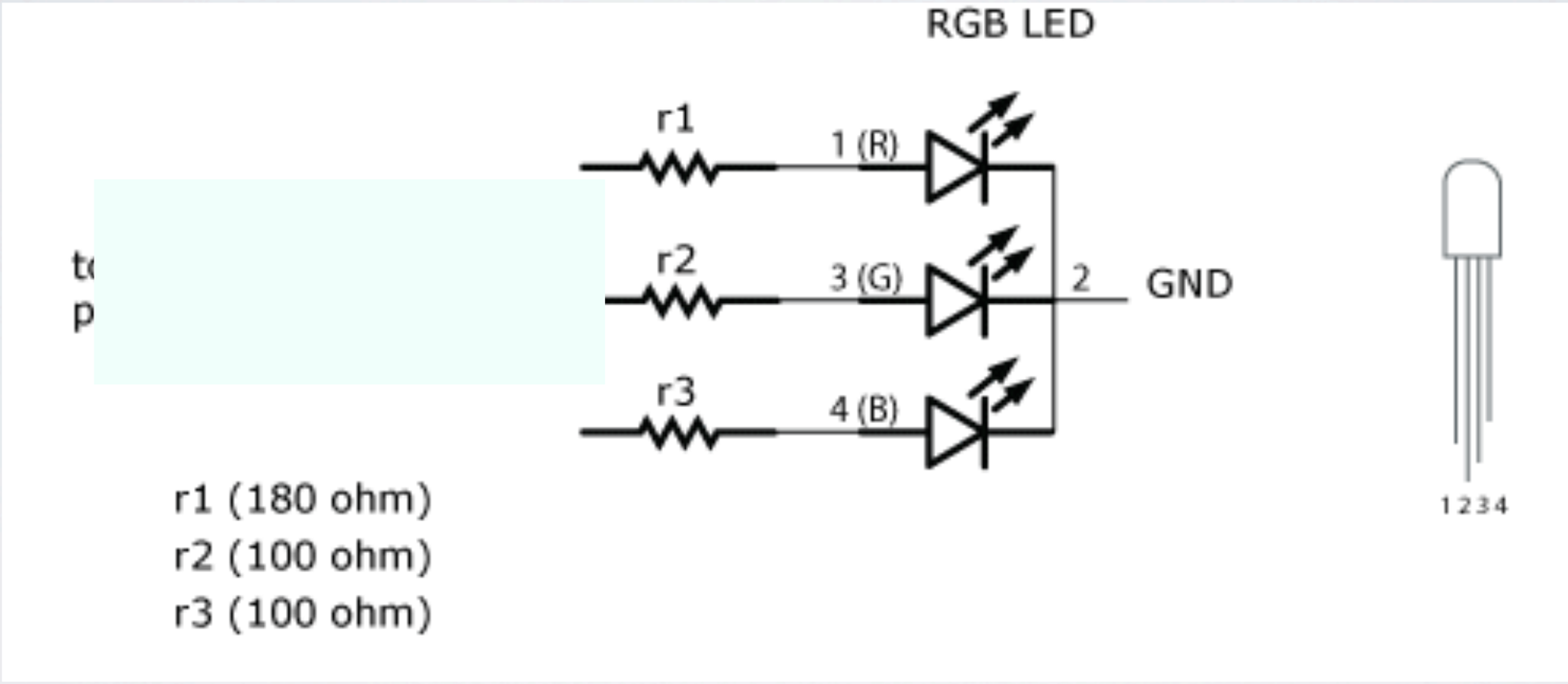


LIGHT EMITTING DIODE (LED)

anode (+)
cathode (-)



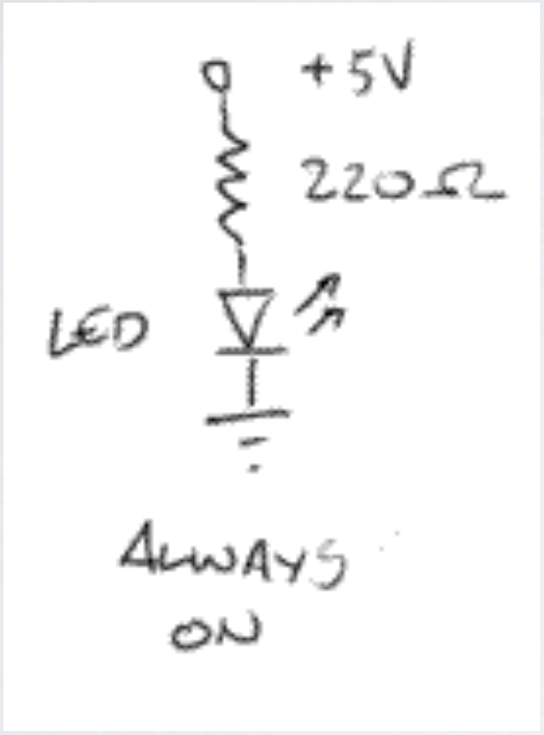
RGB LEDS

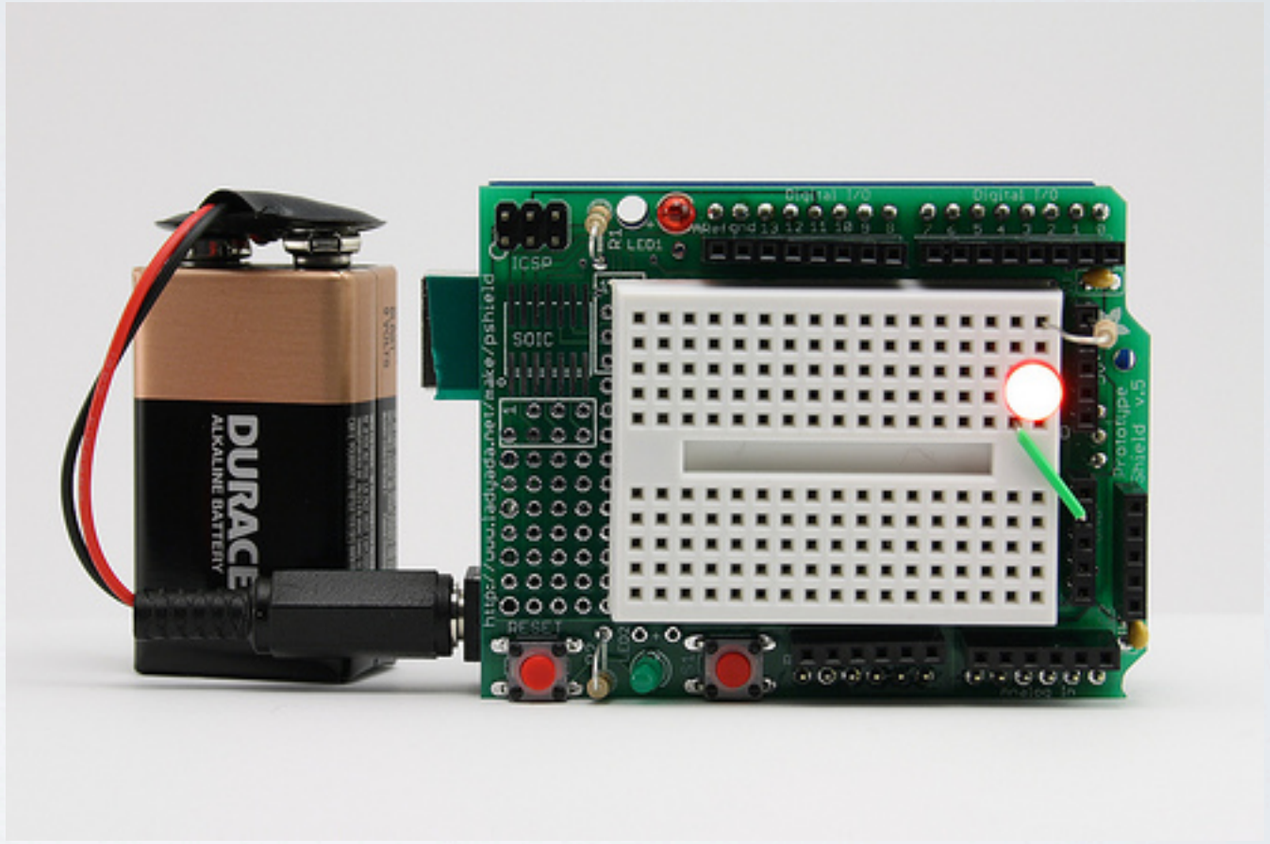

RGB LED
r1
1 (R)
r2
3 (G)
2
GND
r3
4 (B)
1 2 3 4
r1 (180 ohm)
r2 (100 ohm)
r3 (100 ohm)

DIGITAL INPUT & OUTPUT

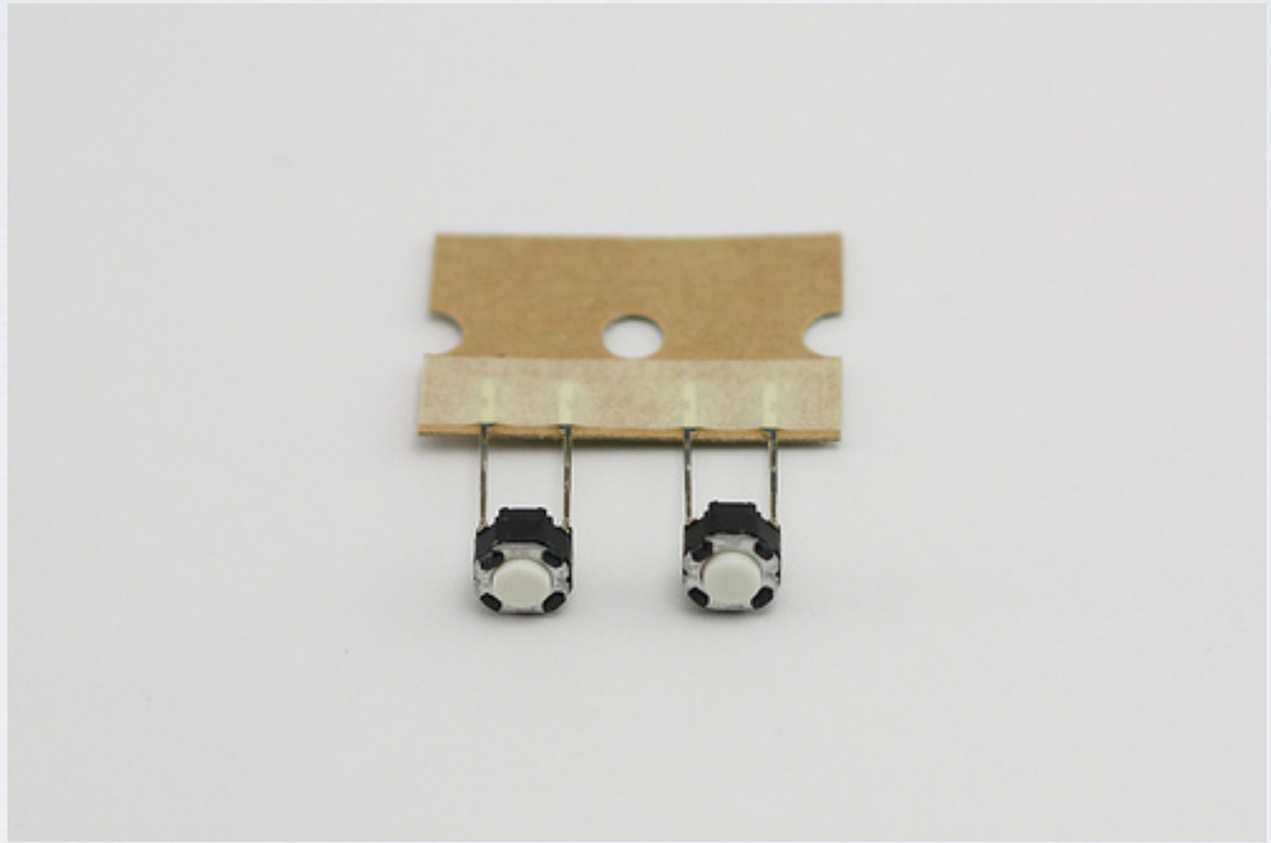
Switches and LEDs

MY FIRST CIRCUIT



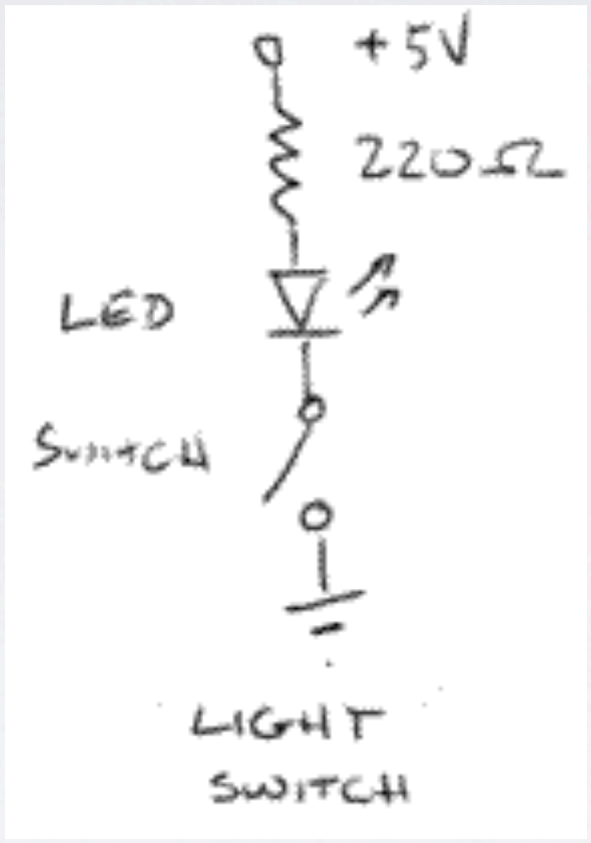


BUTTONS AND SWITCHES

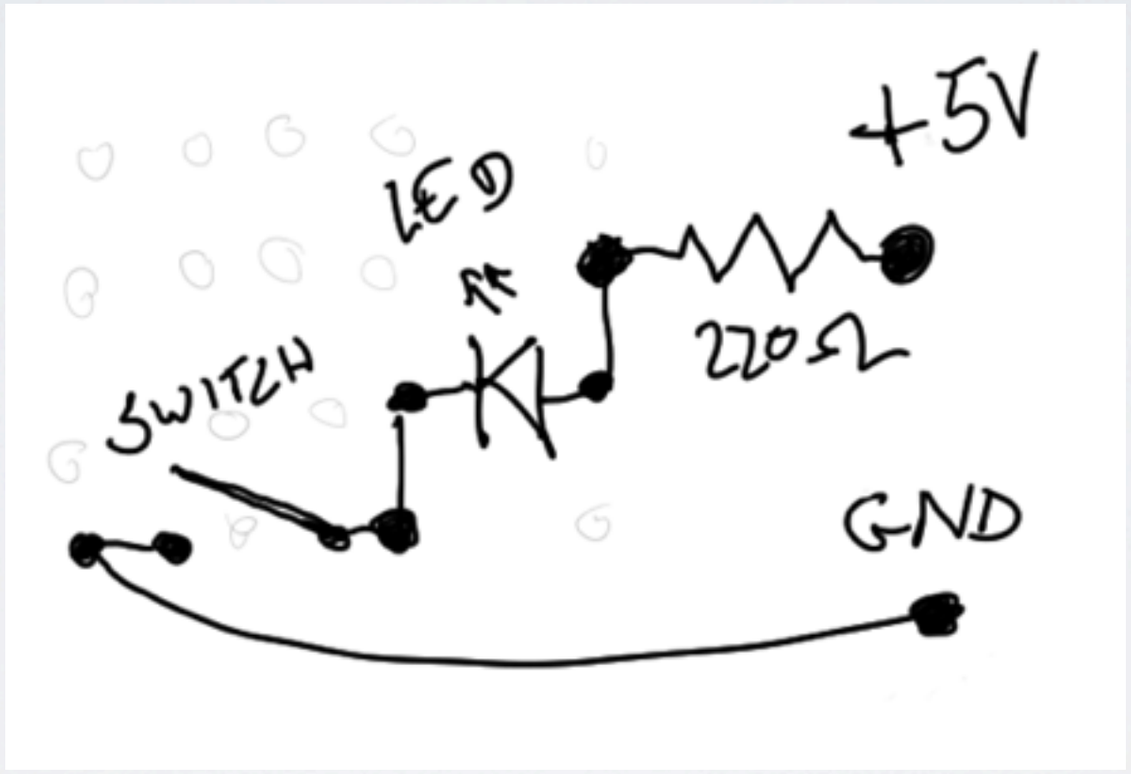


ROCKER SWITCH

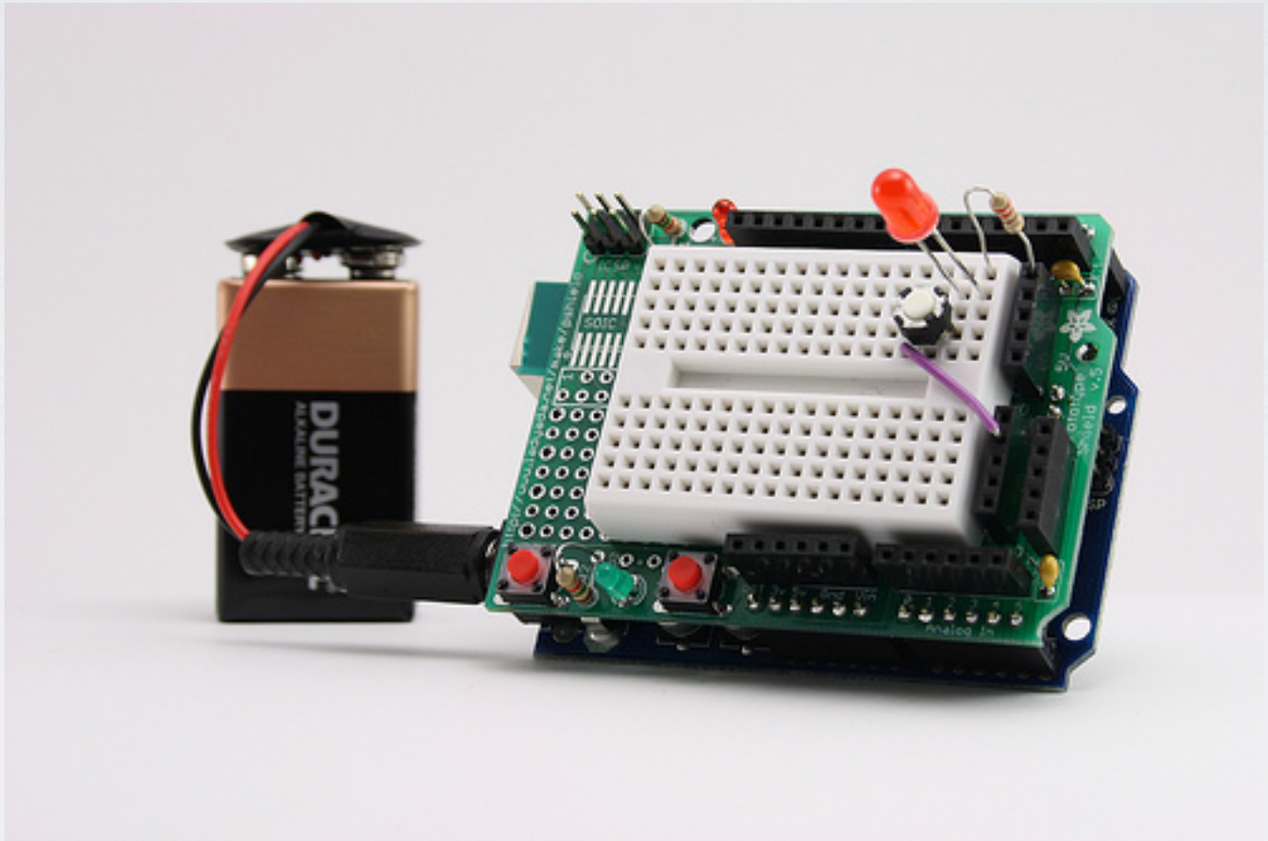
LIGHT SWITCH



LIGHT SWITCH HINT



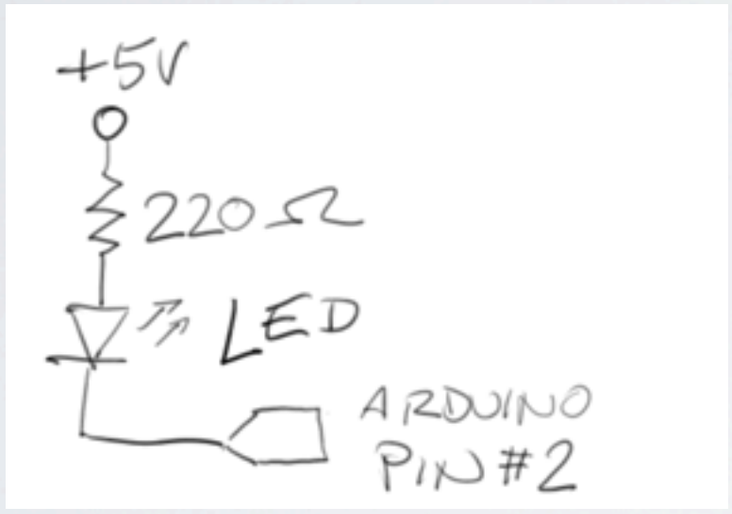
LIGHT SWITCH SOLUTION



MAKE A LIGHT SWITCH PROGRAMMATICALLY

- Step 1: Programmatically turn on LED
- Step 2: Programmatically sense button press
- Step 3: Programmatically link button to turn on LED

STEP 1: LED



sketch_jul02a | Arduino 0022

sketch_jul02a §

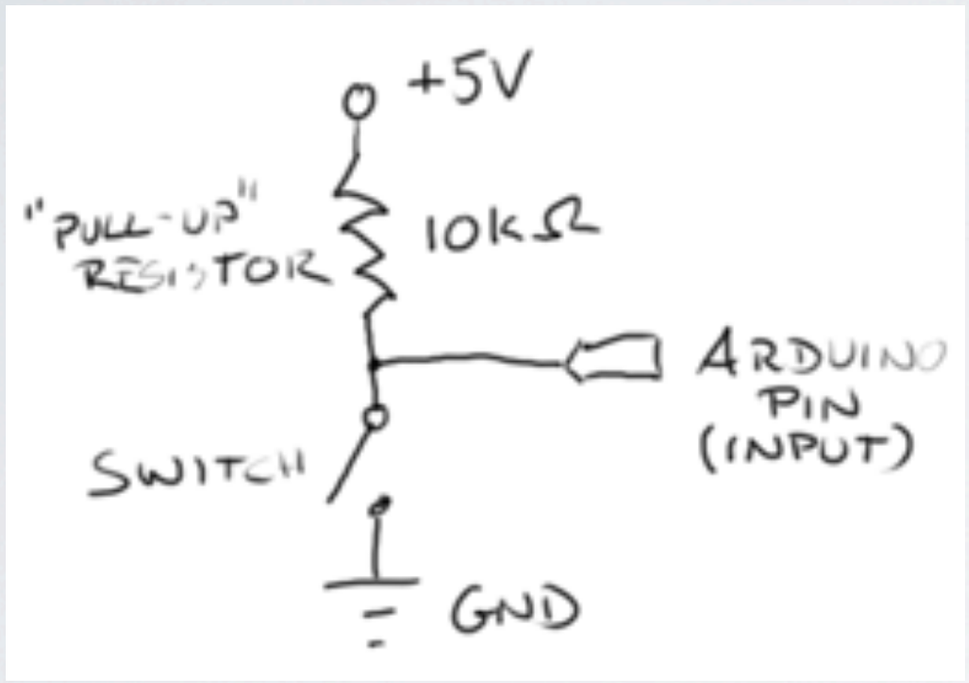
```
int ledPin = 2;
void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  digitalWrite(ledPin, HIGH);   // set the LED on
  delay(1000);                  // wait for a second
  digitalWrite(ledPin, LOW);    // set the LED off
  delay(1000);                  // wait for a second
}
```

Done uploading.

```
Binary sketch size: 1026 bytes (of a 14336 byte maximum)
```

9

STEP 2: BUTTON



Button_Step2 | Arduino 0022

Button_Step2

```
const int buttonPin = 7;     // the number of the pushbutton pin

// variables will change
int buttonState = 0;

void setup() {
  //Create Serial Object
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}

void loop(){

   // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is LOW:
  if (buttonState == LOW) {
    Serial.println("button pressed");
  }
}
```

Done Saving.

Binary sketch size: 2456 bytes (of a 14336 byte maximum)

17

STEP 3: MAPPING I/O

ARDUINO + GUI

PROCESSING

- Download from: http://processing.org

- Processing is an open source programming language and environment for people who want to create images, animations, and interactions

- developed to serve as a software sketchbook and to teach fundamentals of computer programming within a visual context

PROCESSING EXAMPLE

ARDUINO & PROCESSING

- Communication over serial port
- Step 1: Arduino to Processing
- Step 2: Processing to Arduino

ARDUINO TO PROCESSING

Processing_button | Arduino 0022

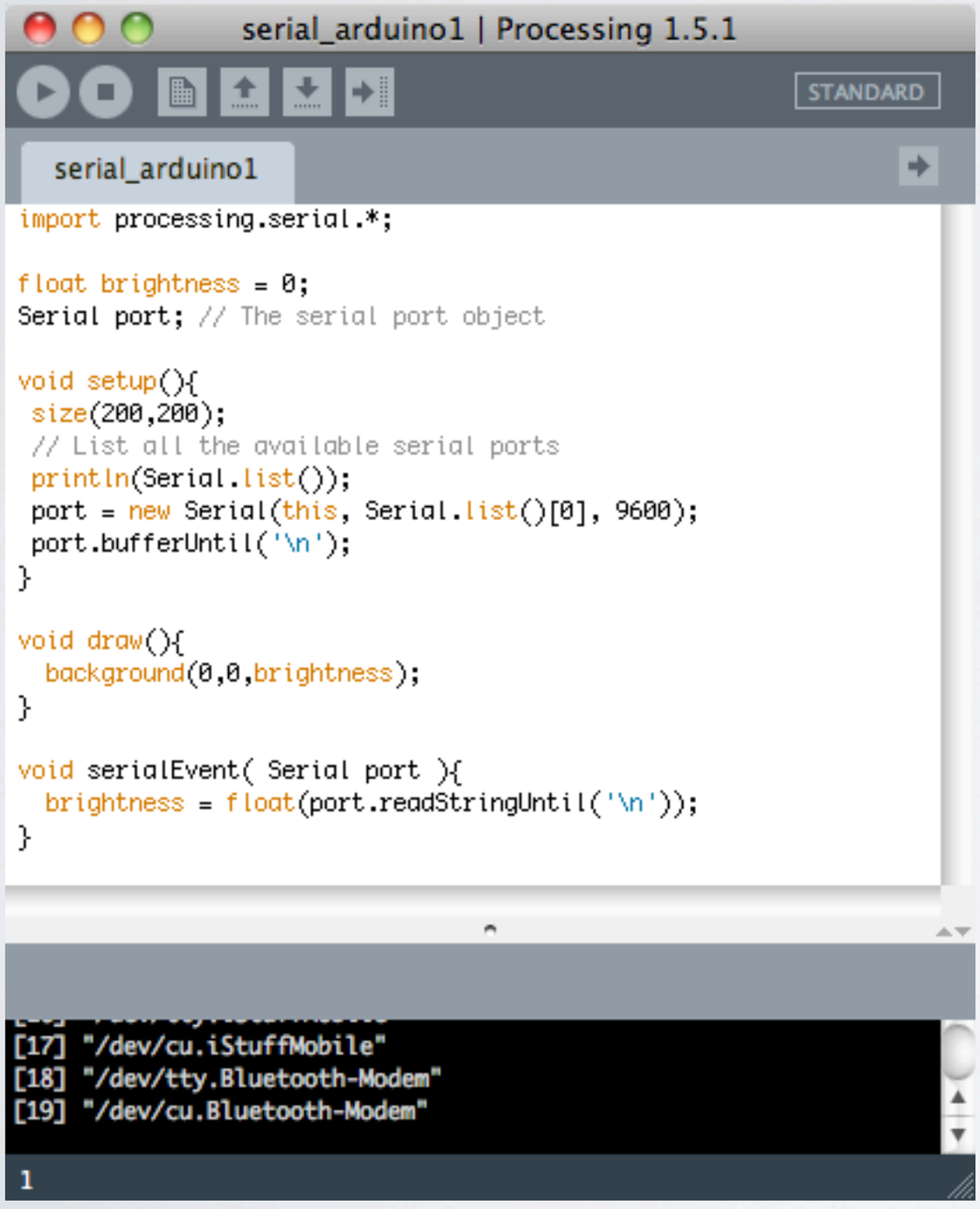
Processing_button

```
const int buttonPin = 7;     // the number of the pushbutton pin

// variables will change
int buttonState = 0;
int lastButtonState = 0;

void setup() {
  //Create Serial Object
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}

void loop(){

   // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  if(buttonState != lastButtonState){
    // check if the pushbutton is pressed.
    // if it is, the buttonState is LOW:
    if (buttonState == LOW) {
      Serial.println(255);
    }else{
      Serial.println(0);
    }
  }
  lastButtonState = buttonState;
}
```

Done Saving.

Arduino

ARDUINO TO PROCESSING



Processing

PROCESSING TO ARDUINO

Processing_button_led | Arduino 0022

Processing_button_led §

```
const int ledPin = 2;
const int buttonPin = 7;     // the number of the pushbutton pin

// variables will change
int buttonState = 0;
int lastButtonState = 0;

void setup() {
  //Create Serial Object
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
  pinMode(ledPin, OUTPUT);

}

void loop(){
   // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  if(Serial.available() != 0){
    int val = Serial.read();
    if(val == 0){
        digitalWrite(ledPin, LOW);
    } else if (val == 1){
        digitalWrite(ledPin, HIGH);
    }
  }

  // check if the pushbutton has changed
  if(lastButtonState != buttonState){
    // if it is, the buttonState is LOW:
    if (buttonState == LOW) {
      Serial.println(255);
    }else{
      Serial.println(0);
    }
  }
  lastButtonState = buttonState;
}
```

Arduino

PROCESSING TO ARDUINO

serial_arduino2 | Processing 1.5.1

STANDARD

serial_arduino2

```
import processing.serial.*;

float brightness = 0;
Serial port; // The serial port object
boolean lastMousePressed = false;

void setup(){
 size(200,200);
 // List all the available serial ports
 println(Serial.list());
 port = new Serial(this, Serial.list()[0], 9600);
 port.bufferUntil('\n');
}

void draw(){
  background(0,0,brightness);
  if(lastMousePressed != mousePressed){
    if(mousePressed == true){
      port.write(0);
    }else{
      port.write(1);
    }
  }
  lastMousePressed = mousePressed;
}

void serialEvent( Serial port ){
  brightness = float(port.readStringUntil('\n'));
}
```

[19] "/dev/cu.Bluetooth-Modem"

Processing

PROCESSING SUPPORT FOR ARDUINO

- http://www.arduino.cc/playground/Interfacing/Processing
- Don't forget to restart processing after installing the libraries

FIRMATA

- Generic protocol for communicating with microcontrollers from software on a host computer.

- It is intended to work with any host computer software package.

- Works with a number of languages.

- The aim is to allow people to completely control the Arduino from software on the host computer.

LOAD FIRMATA FIRMWARE

- Arduino: File > Examples > Firmata > StandardFirmata

FIRST FIRMATA PROGRAM

sketch_jul03b | Processing 1.5.1

STANDARD

sketch_jul03b §

```
import processing.serial.*;
import cc.arduino.*;

Arduino arduino;
int ledPin=2;

void setup()
{
  size(256, 256);
  arduino = new Arduino(this, Arduino.list()[1], 57600);
  arduino.pinMode(ledPin, Arduino.OUTPUT);
  background(0);
}

void draw()
{
  if(mousePressed == true){
      arduino.digitalWrite(ledPin, Arduino.LOW);
  } else {
      arduino.digitalWrite(ledPin, Arduino.HIGH);
  }
}
```

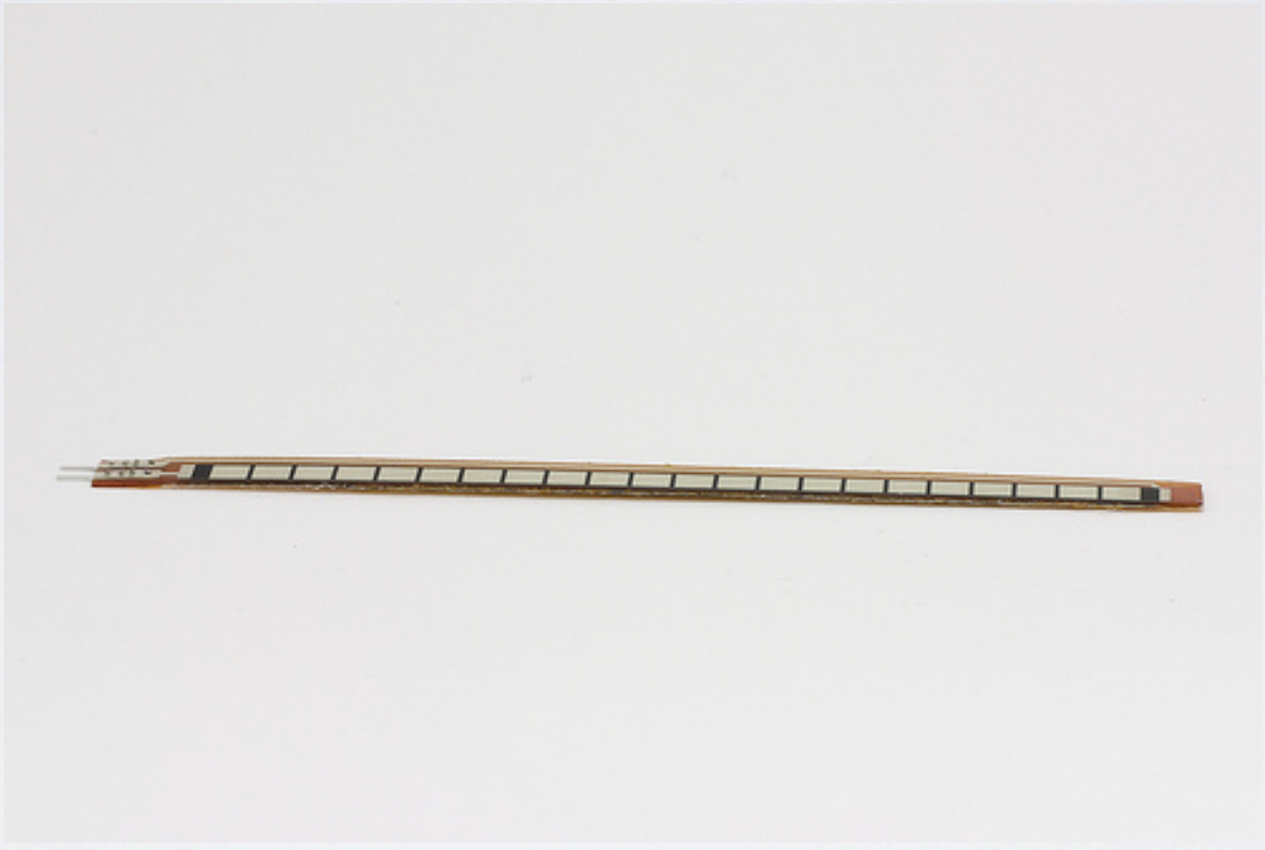
```
native lib Version = RXTX-2.2pre2
Invalid memory access of location 0xb17b9be0 eip=0x24bc690c
```

5

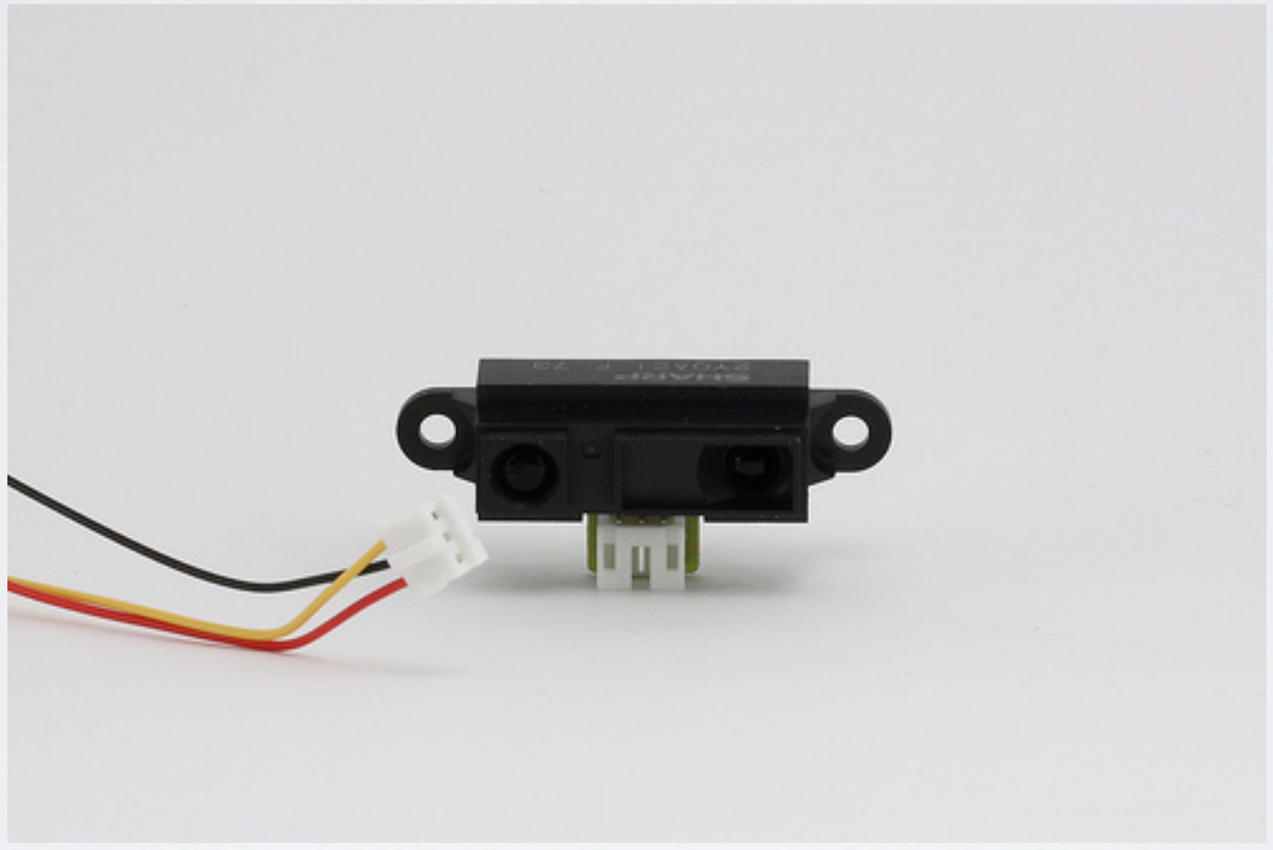
ANALOG SENSORS

POTENTIOMETERS

BEND SENSOR



INFRARED DISTANCE RANGER

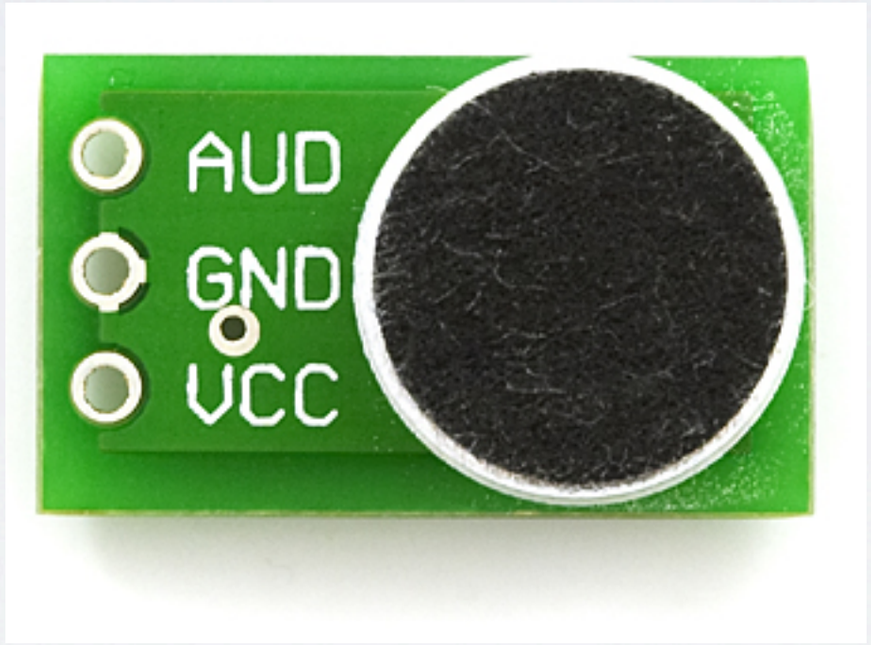


2-AXIS ACCELEROMETER

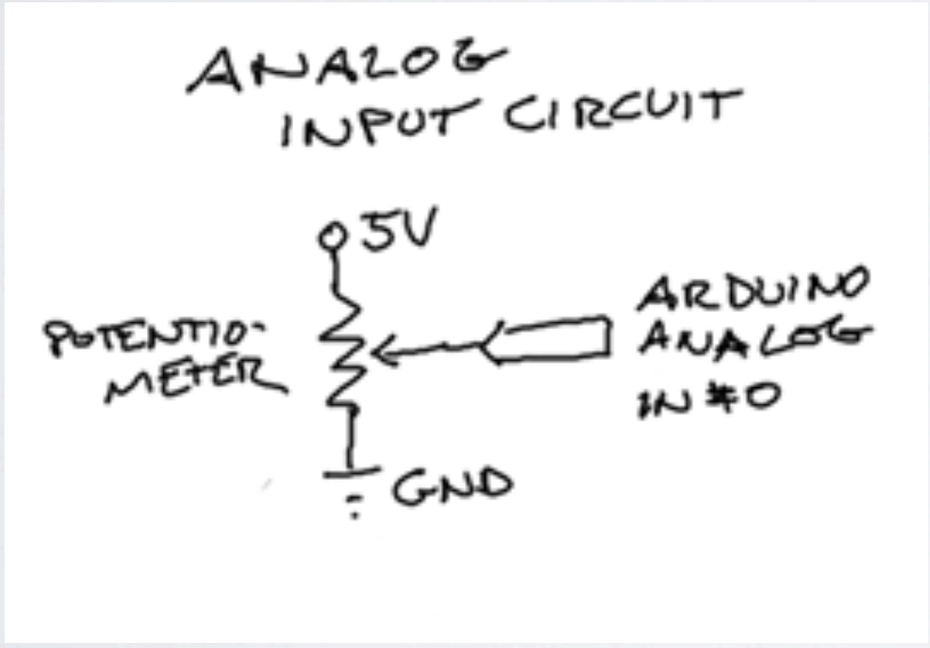
FORCE SENSITIVE RESISTORS

THUMB JOYSTICK

MICROPHONE



PRINCIPLES OF ANALOG INPUT



- 10 bit Analog to Digital Converter
- 2^10 = 1024
- Sampling rate up to ~10 kHz
- V = 5 * s / 1024

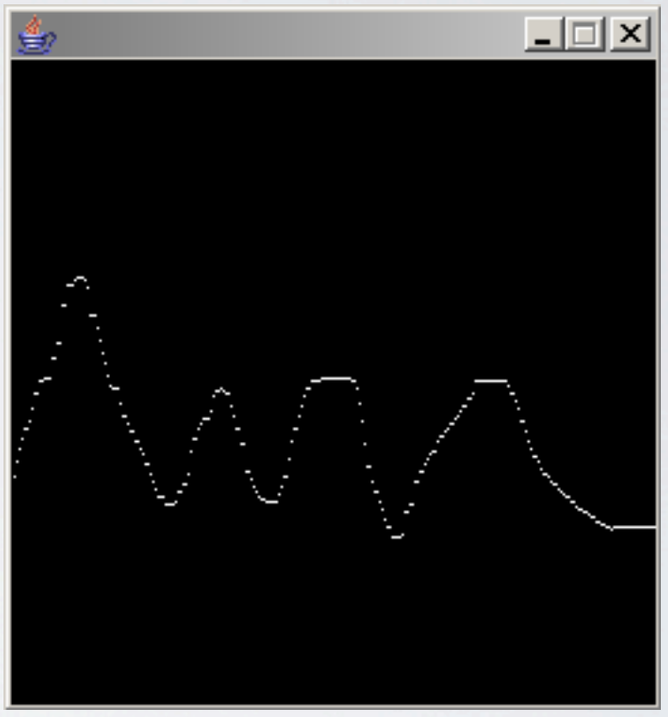
VISUALIZING ANALOG INPUT

analogInput | Processing 1.5.1

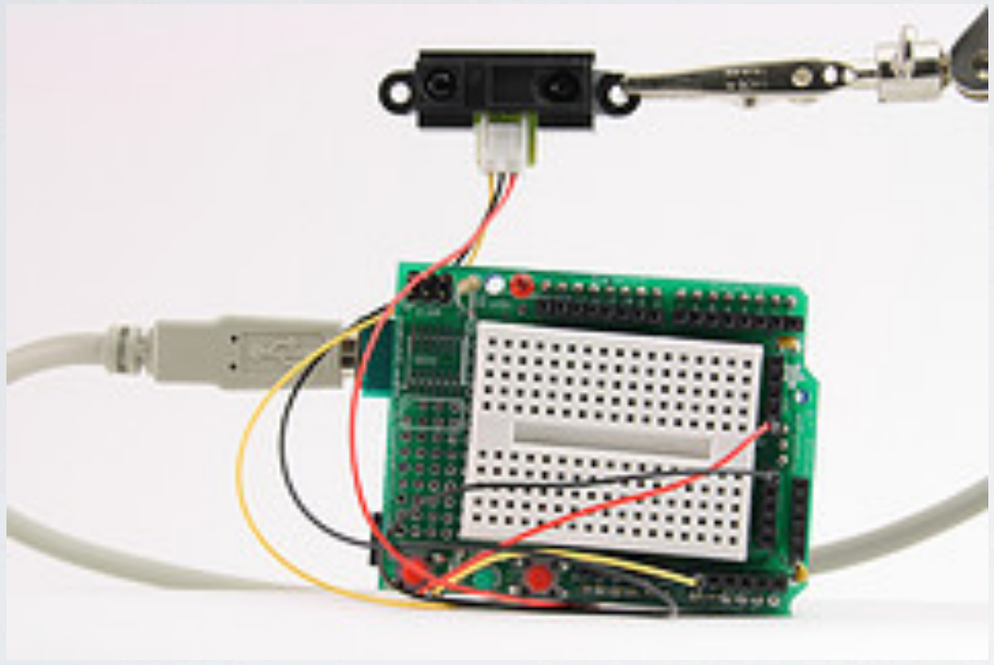
STANDARD

analogInput

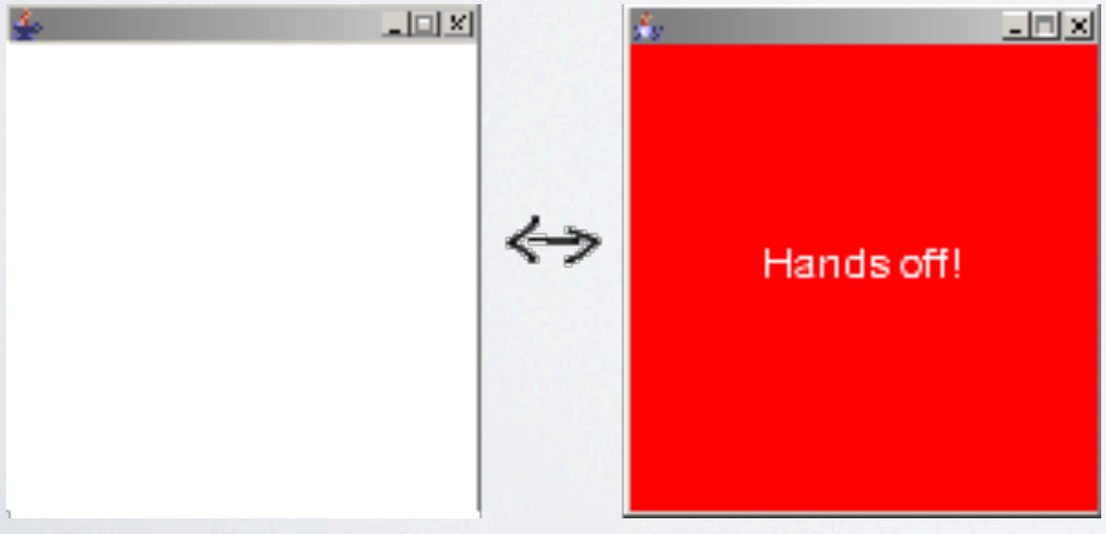
```
/**
* Graph sensor values
*/
import processing.serial.*;
import cc.arduino.*;

int[] xvals;
int arrayindex = 0;
Arduino arduino;
int ledPin=9;
int potPin=0;
void setup()
{
  size(256, 256);
  xvals = new int[width];
  arduino = new Arduino(this, Arduino.list()[1], 57600);
}

void draw()
{
  background(0);
  //shift array left by one
  for(int i=1; i<width; i++) {
    xvals[i-1] = xvals[i];
  }
  // Add the new values to the end of the array
  // read potentiometer (0..1024), divide by four (0..255)
  // to stay within canvas drawing limits
  xvals[width-1] = arduino.analogRead(potPin)/4;
  for(int i=1; i<width; i++) {
    stroke(255);
    point(i, 255-xvals[i]);
  }
}
```



THRESHOLDING



- process of turning continuous data into discrete yes / no decision

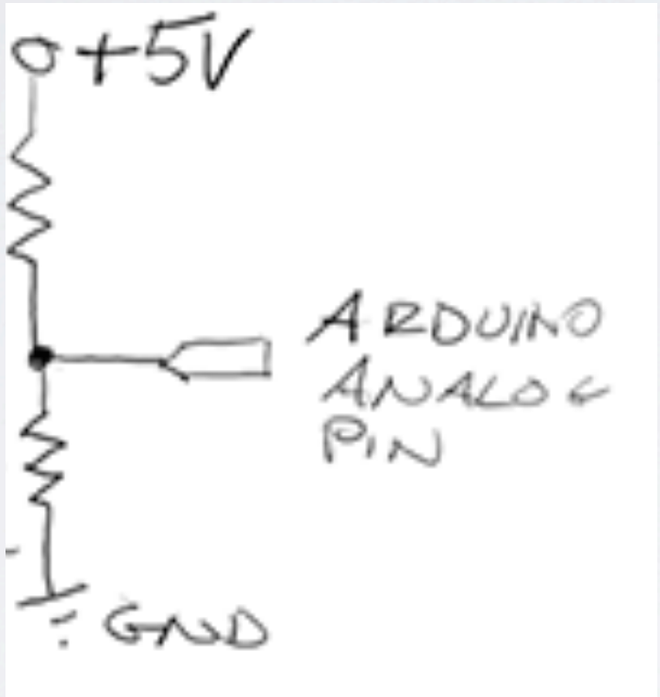


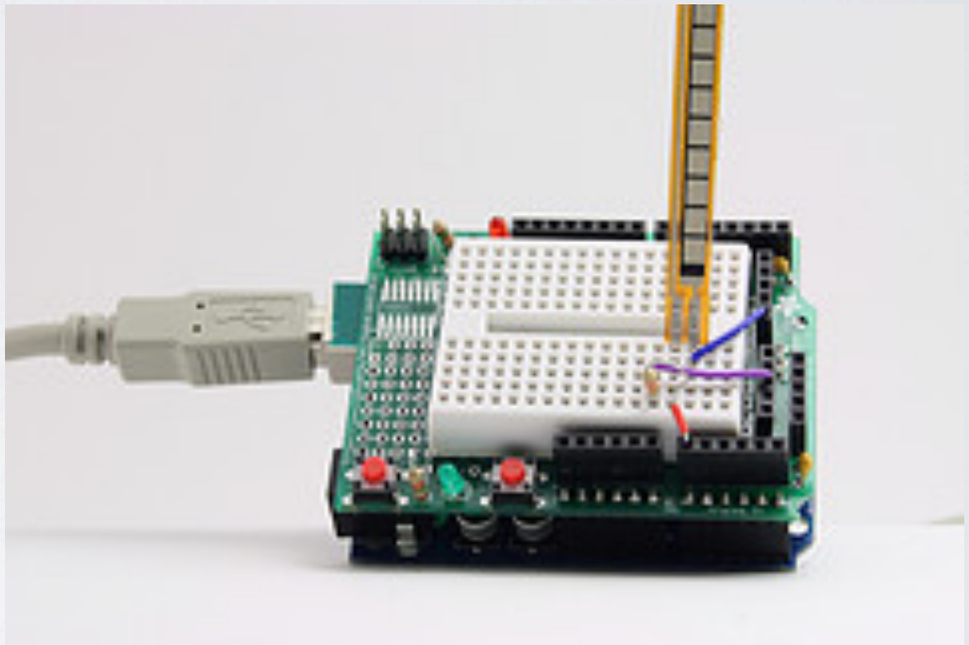
BEND SENSOR

Bend Sensor

10k - 40k

??





CALCULATIONS

$$V_{OUT} = \frac{R_2}{R_1 + R_2} \cdot V_{IN}$$

$V_{in} = 5V$

$R_2 = 22k\Omega$

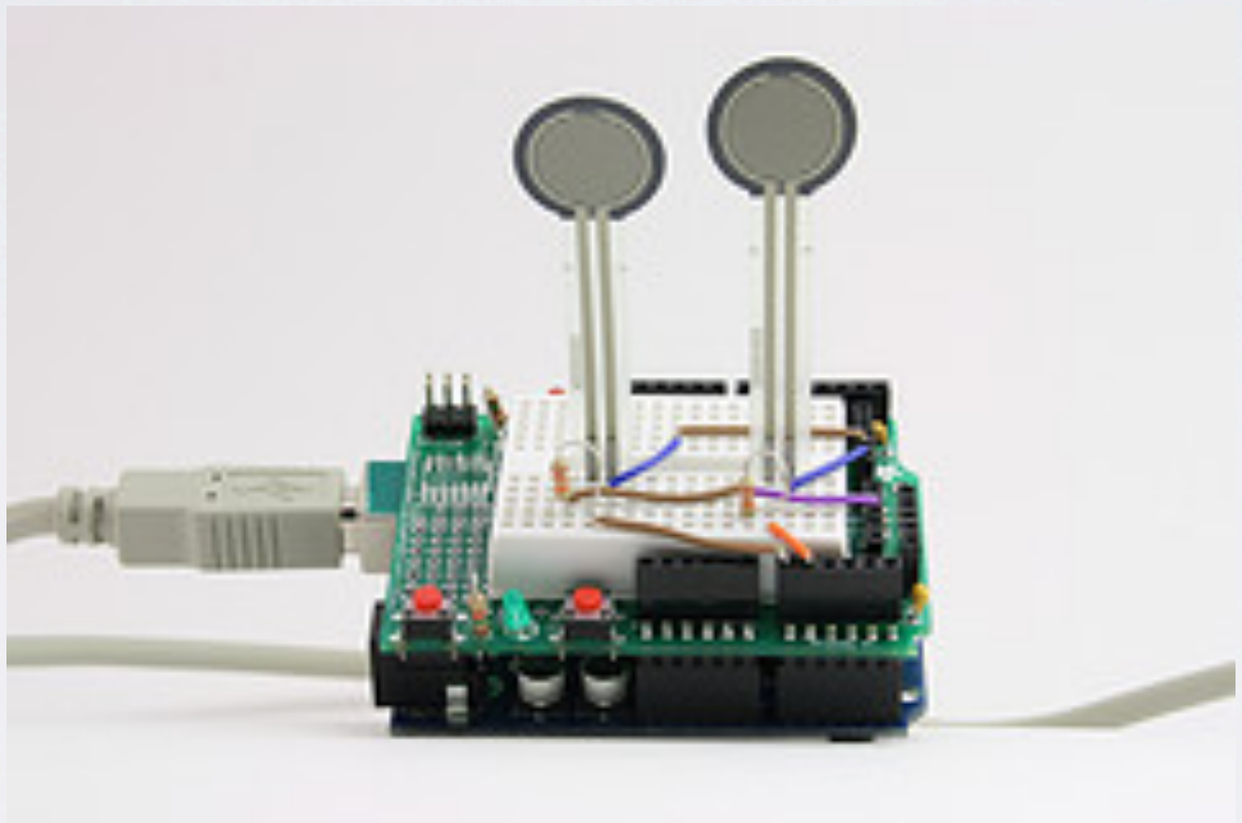
R_1: $10k - 40k\Omega$

WE EXPECT:

$$V_{STRAIGHT} = \frac{22k\Omega}{(10+22)k\Omega} \cdot 5V = 3.44V$$

$$V_{BENT} = \frac{22k\Omega}{40+22k\Omega} \cdot 5V = 1.77V$$

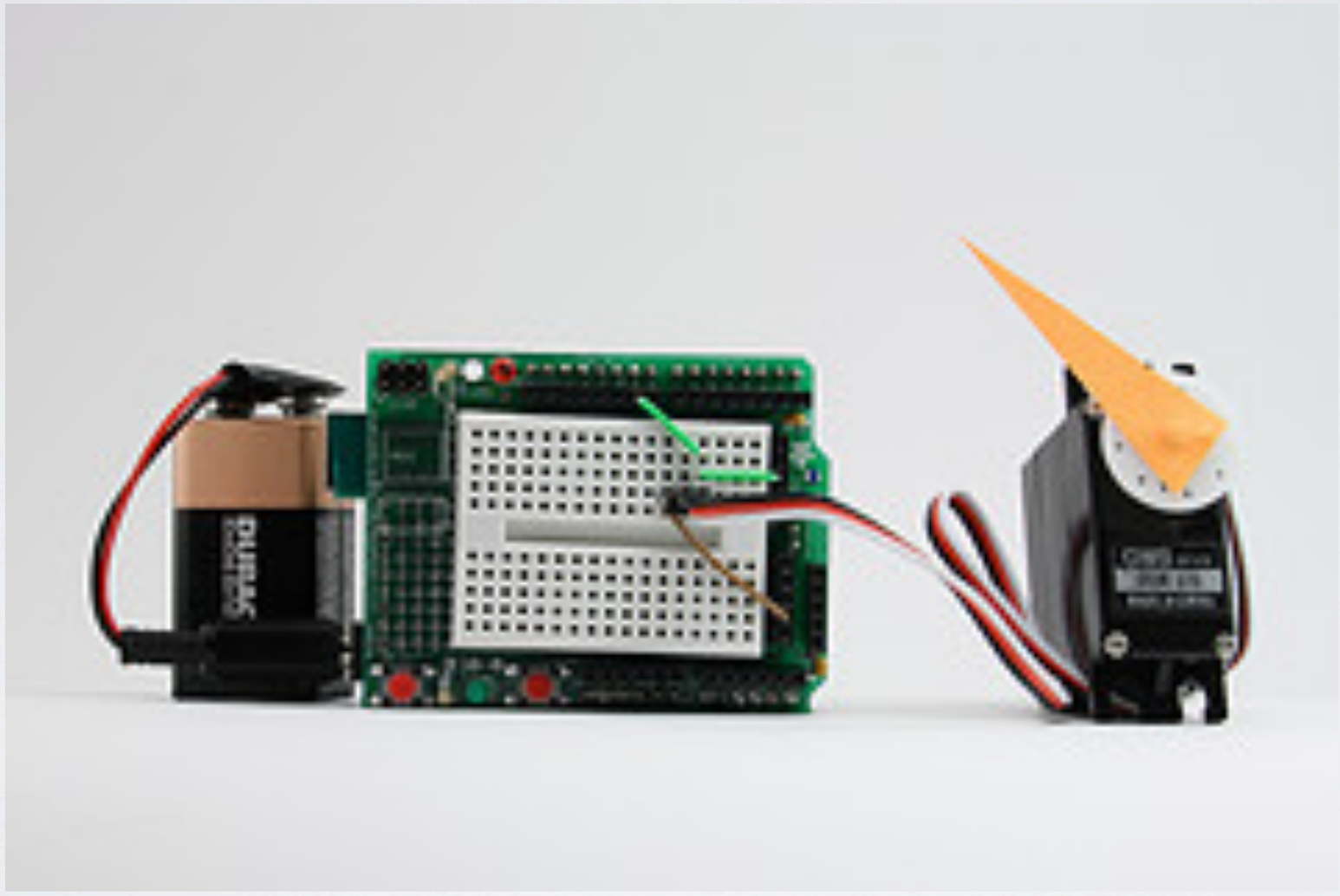
EXAMPLE: FSR THUMB WRESTLING



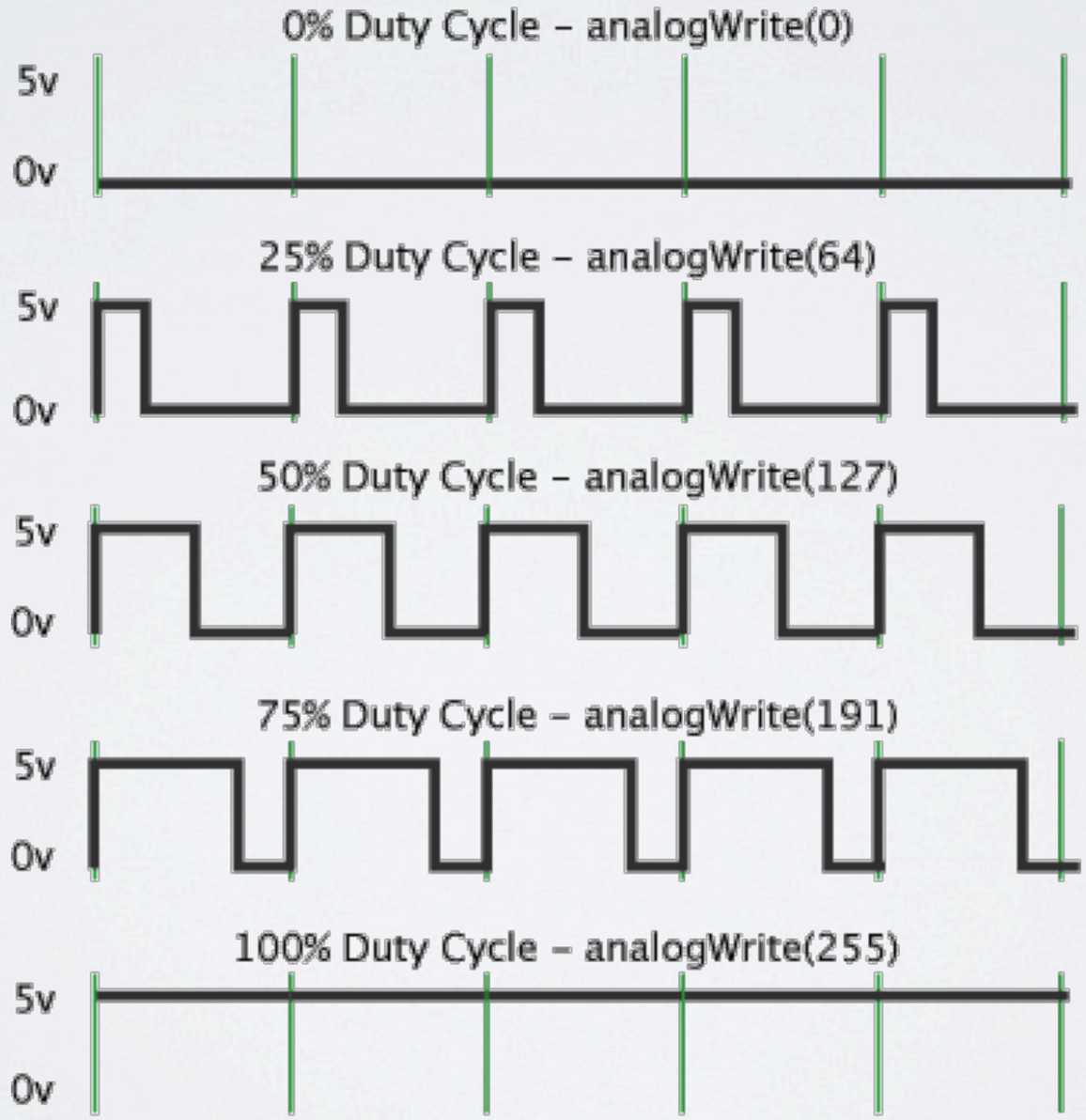
ACTUATORS

Making things move

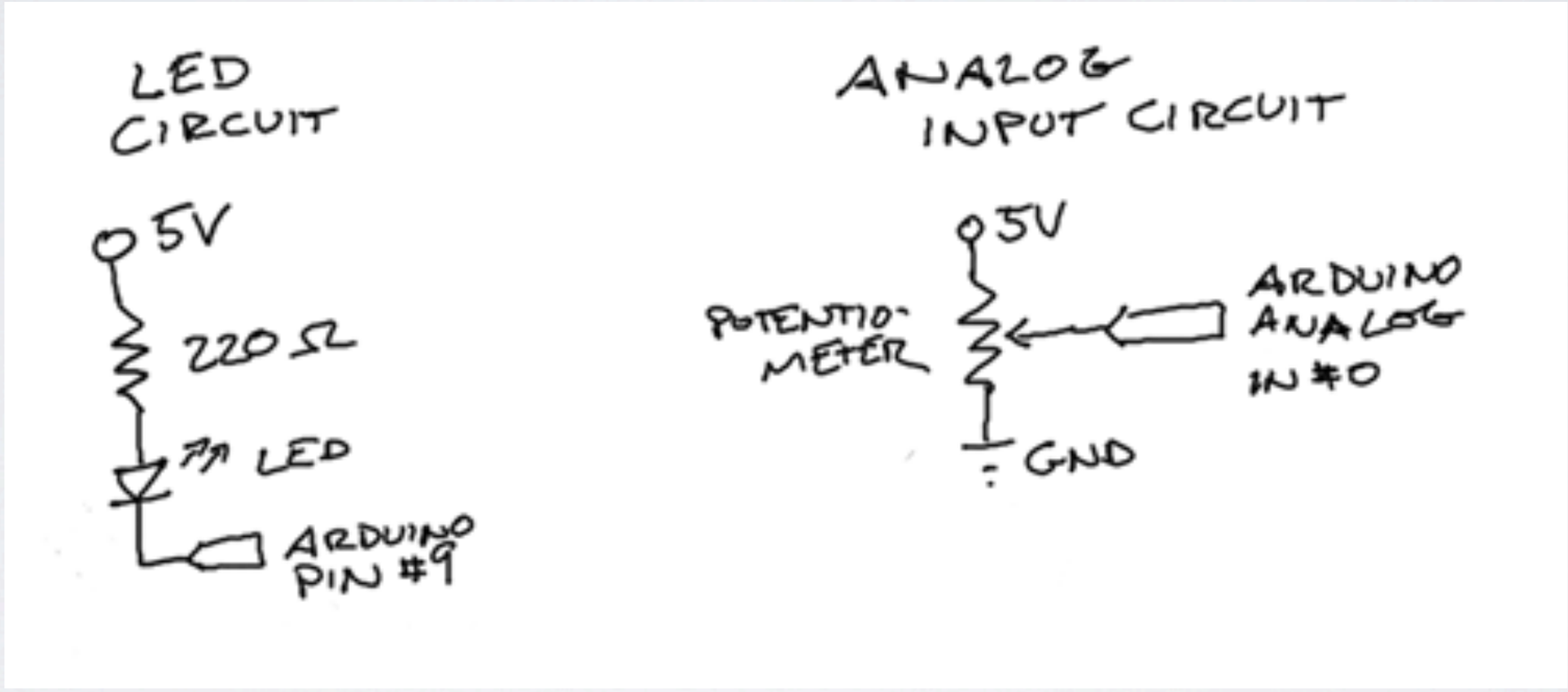
SERVO MOTORS



PULSE WIDTH MODULATION



EXAMPLE: DIMMER



SERVO EXAMPLE

servo | Processing 1.5.1

STANDARD

servo §

```
import processing.serial.*;
import cc.arduino.*;

Arduino arduino;
int pos=0; //servo position in degrees (0..180)
int servoPin=9;

void setup() {
  size(360, 200);

  arduino = new Arduino(this, Arduino.list()[1], 57600); //your

  arduino.pinMode(servoPin, Arduino.OUTPUT);

}

void draw() {
  // read mouseX coordinate
  int newPos = constrain(mouseX/2,0,180);

  // update bg & servo position if mouseX changed
  if(newPos != pos) {
    background(newPos);
    arduino.analogWrite(servoPin,newPos);
    pos=newPos; //update servo position storage variable
  }
}
```

DESIGN EXERCISE

RESEARCH IN PHYSICAL PROTOTYPING

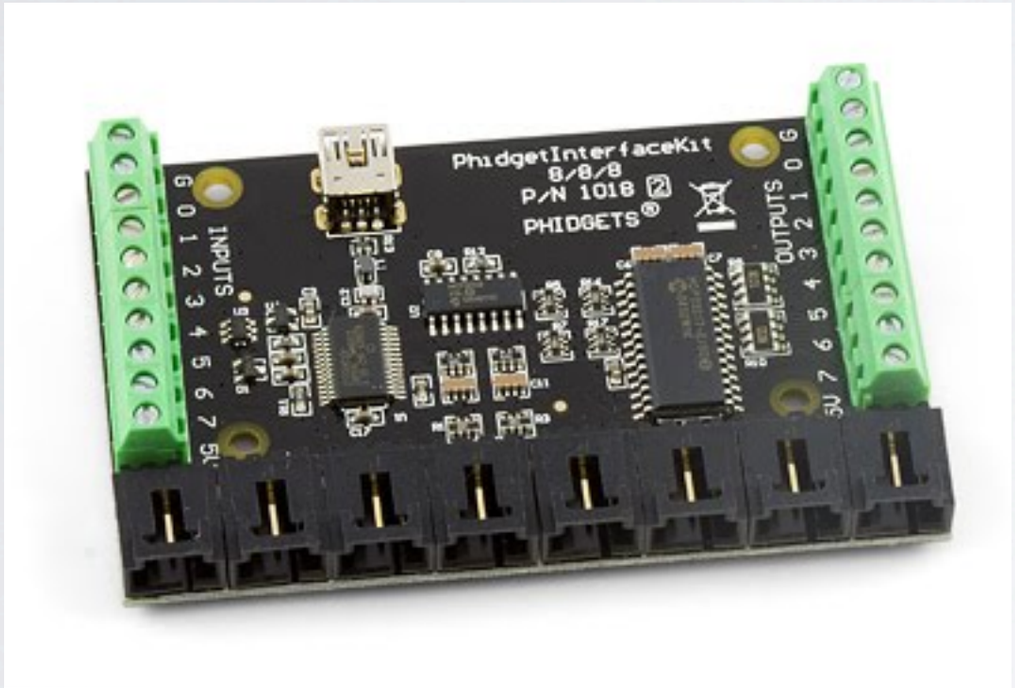
KEY RESEARCH GOAL

- Low Threshold
- High Ceiling

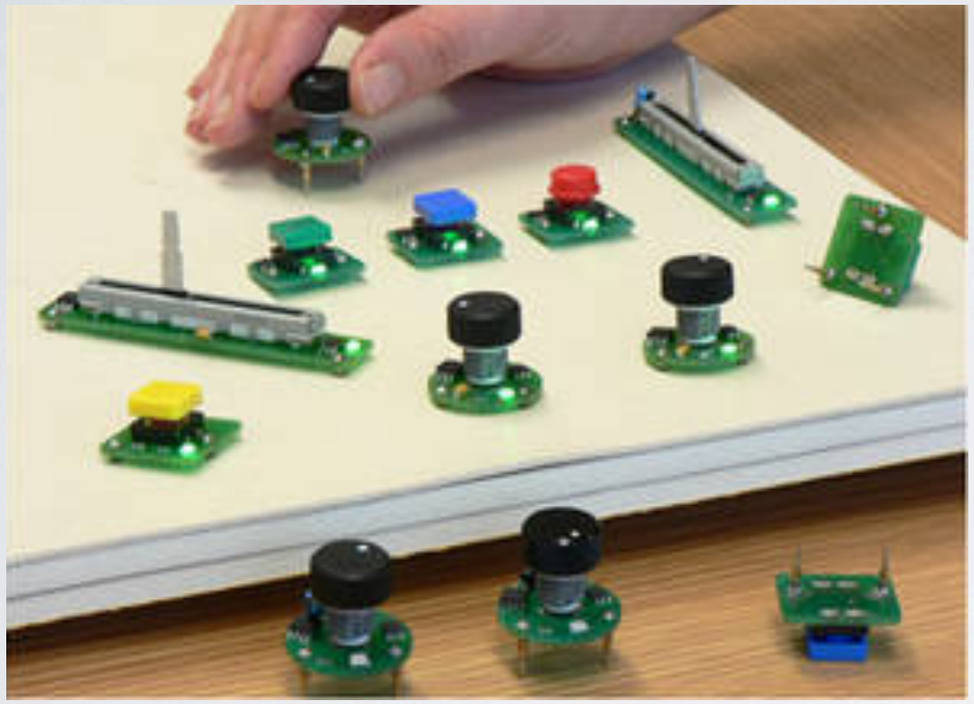
[Greenberg, UIST 2001]

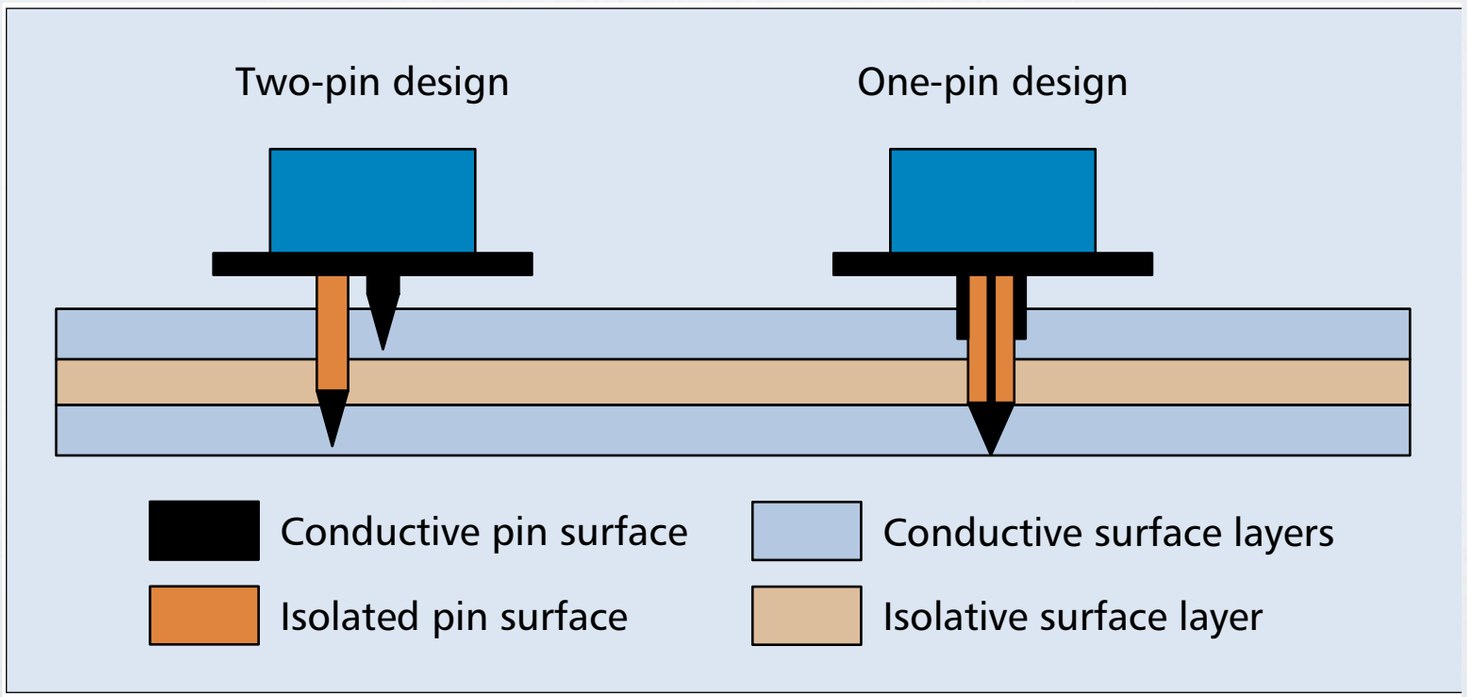
PHIDGETS

www.phidgets.com



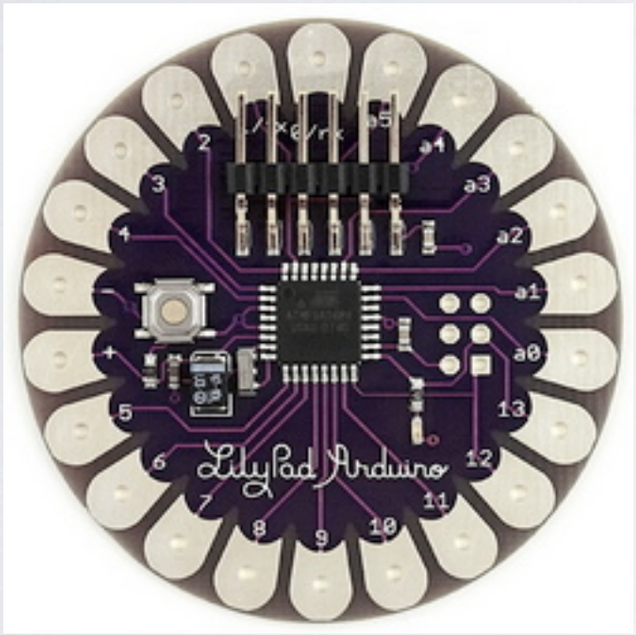
VOODOO I/O

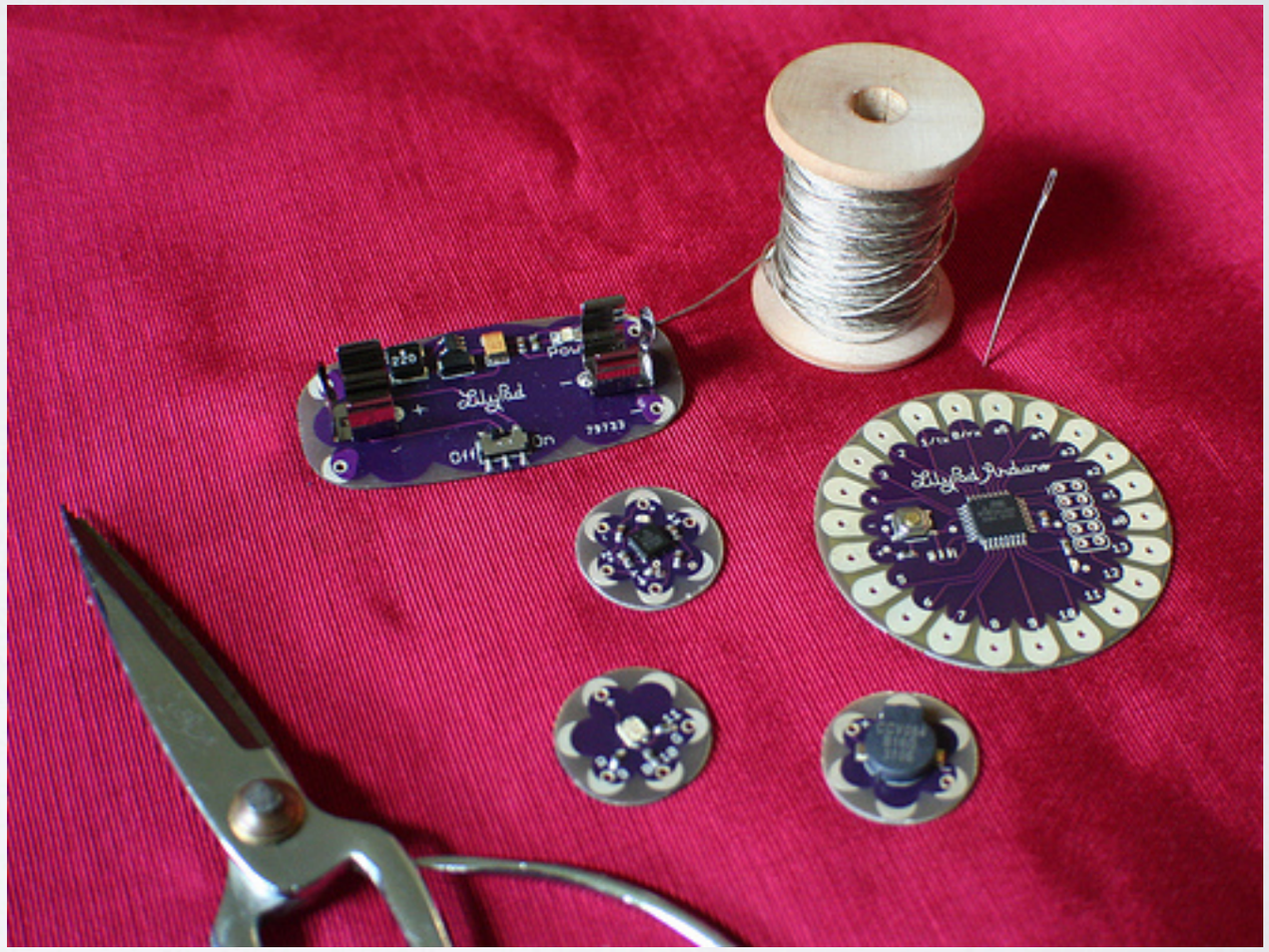



Two-pin design
One-pin design
Conductive pin surface
Isolated pin surface
Conductive surface layers
Isolative surface layer

[Buechley, CHI '08]

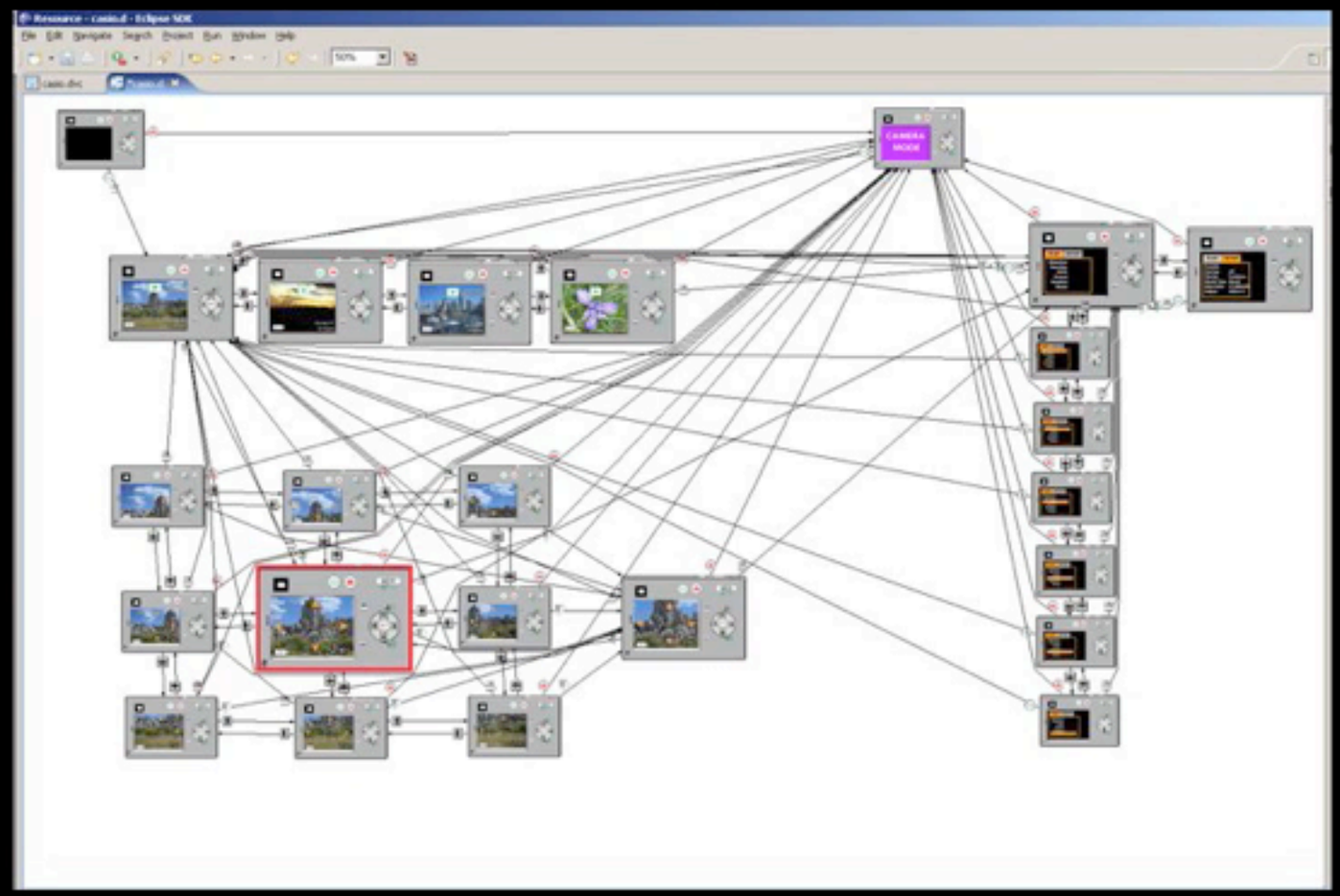
LILYPAD ARDUINO





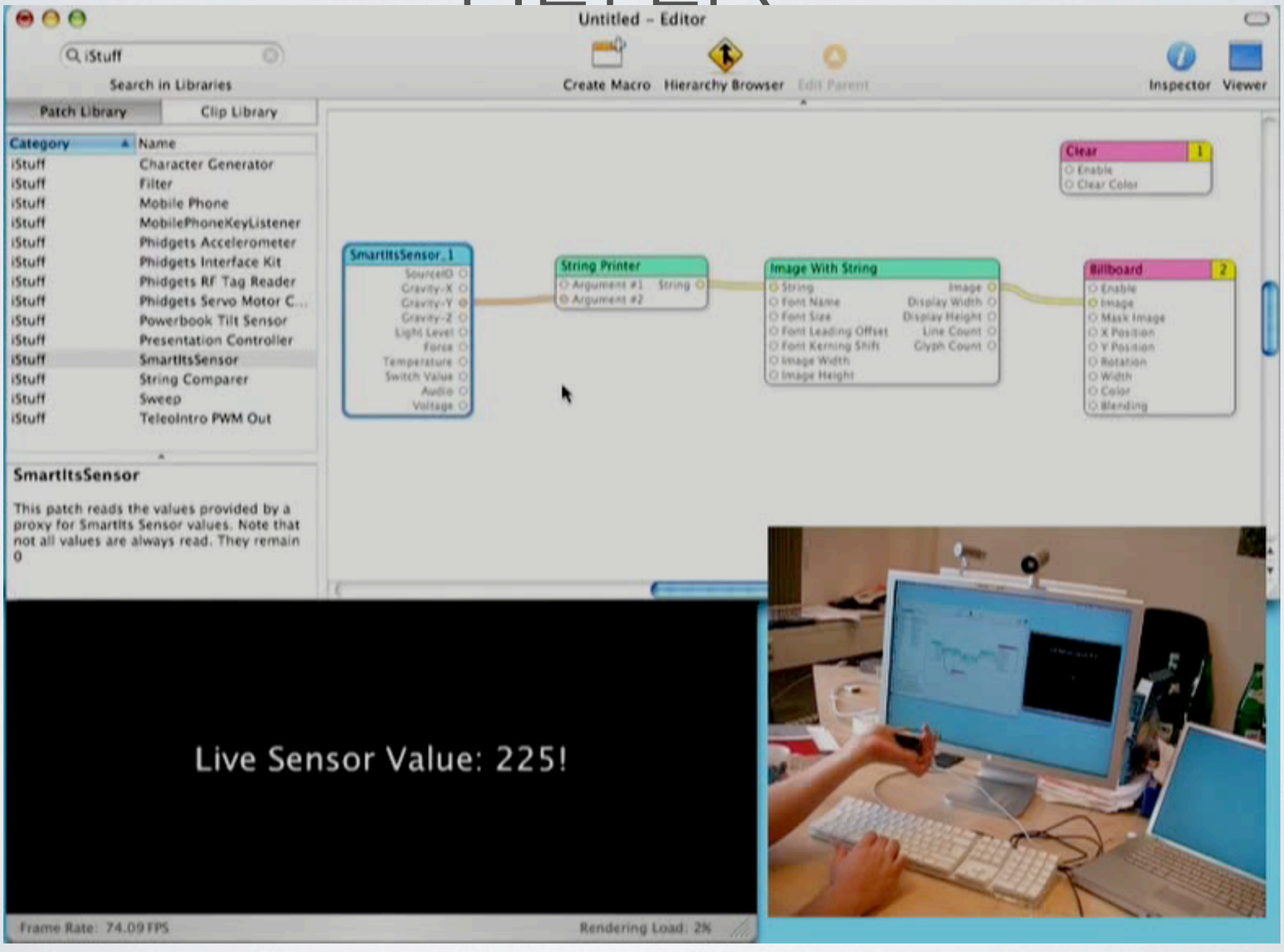
[Hartmann, UIST 2006]

D.TOOLS: STATE CHART EDITING

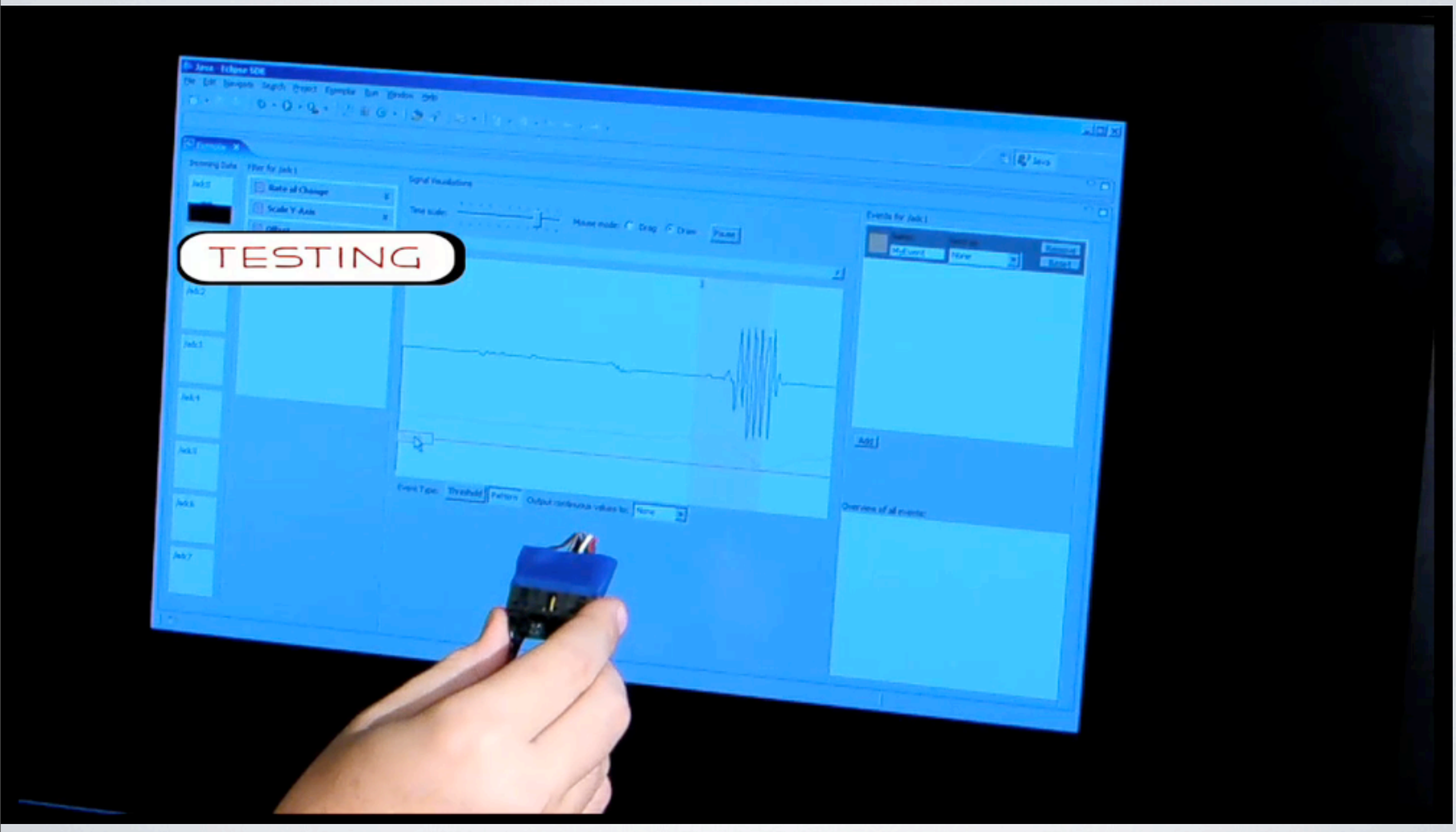


[Ballagas, CHI 2007]

iSTUFF MOBILE: PIPE & FILTER

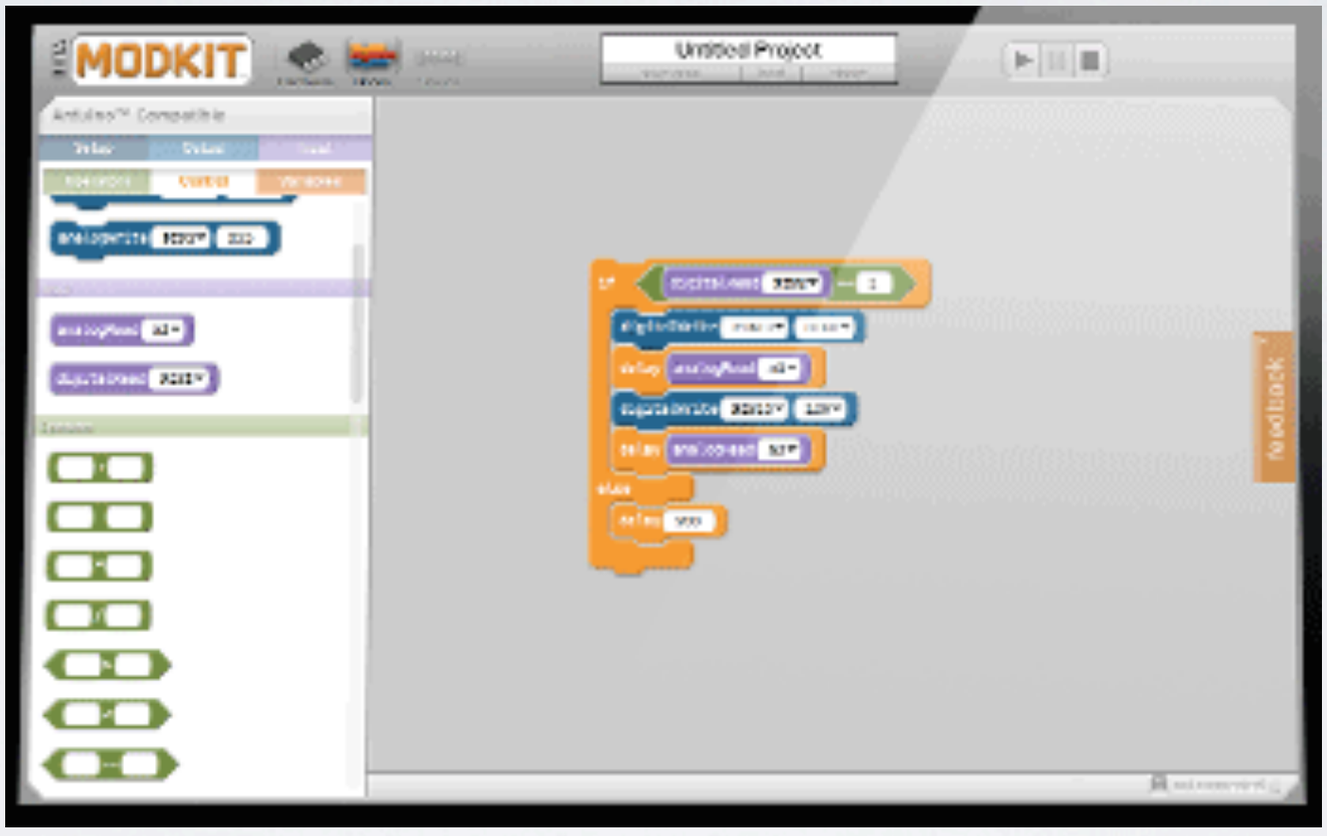


EXEMPLAR: PROG. BY DEMONSTRATION

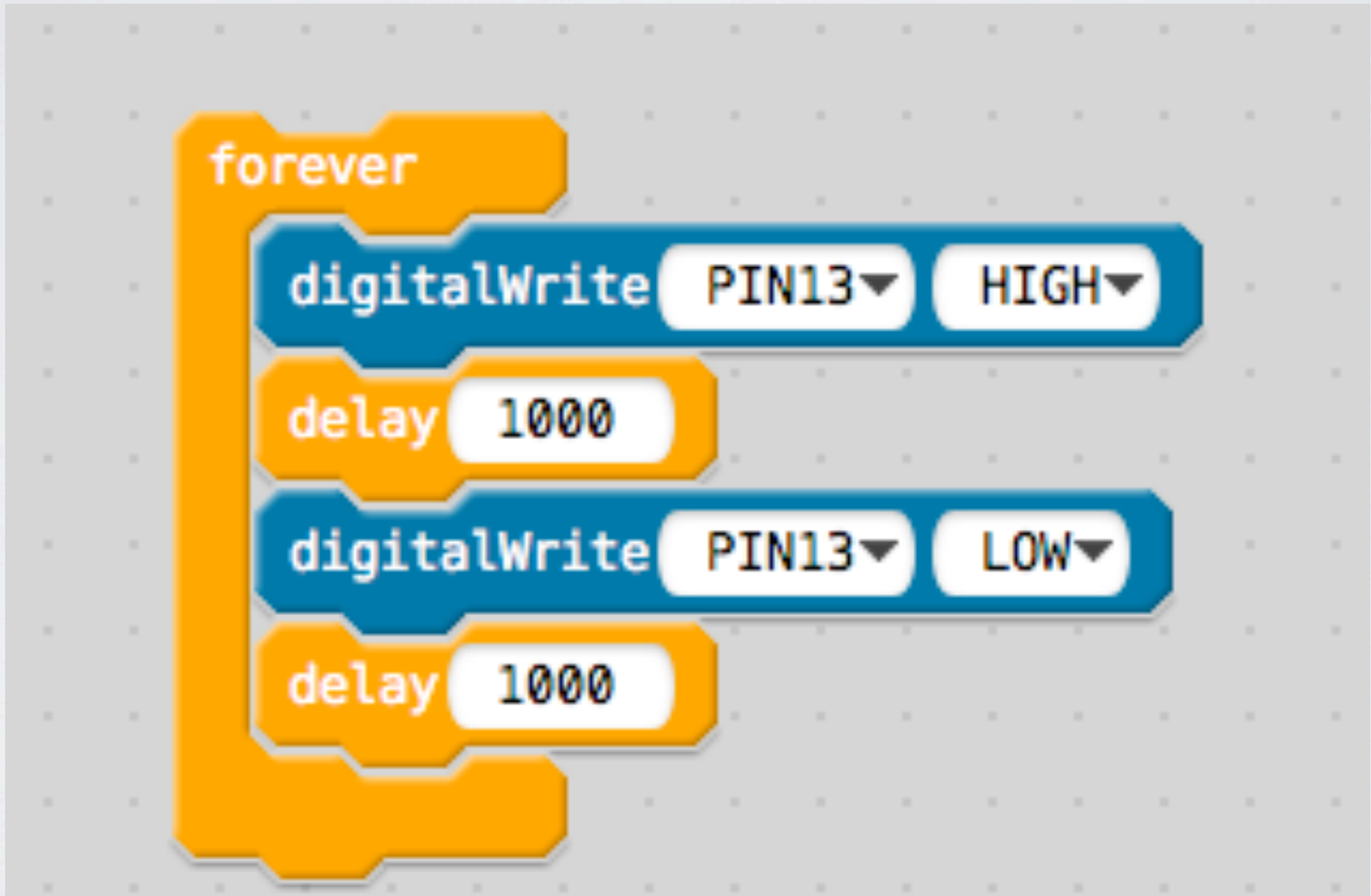


MODKIT

- Scratch Meets Arduino

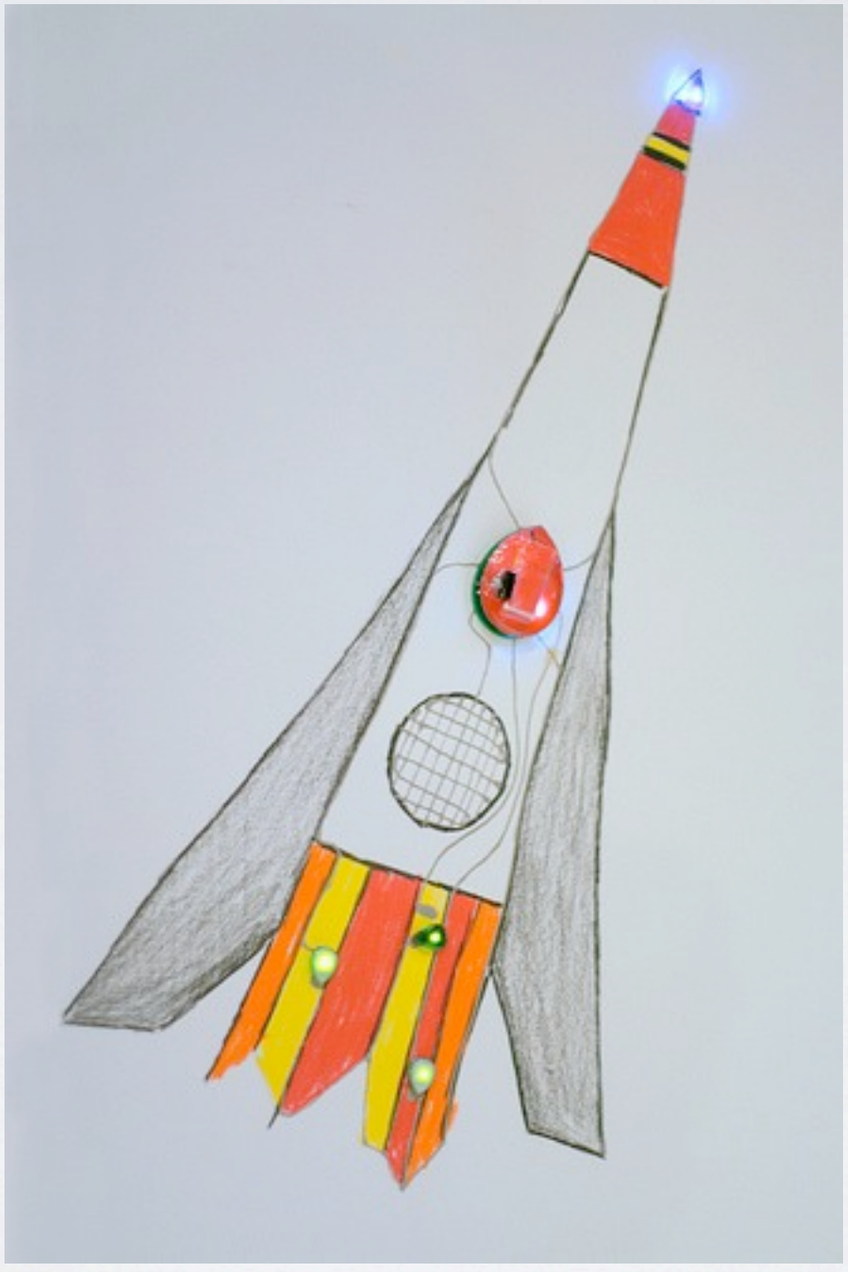


MODKIT EXAMPLE



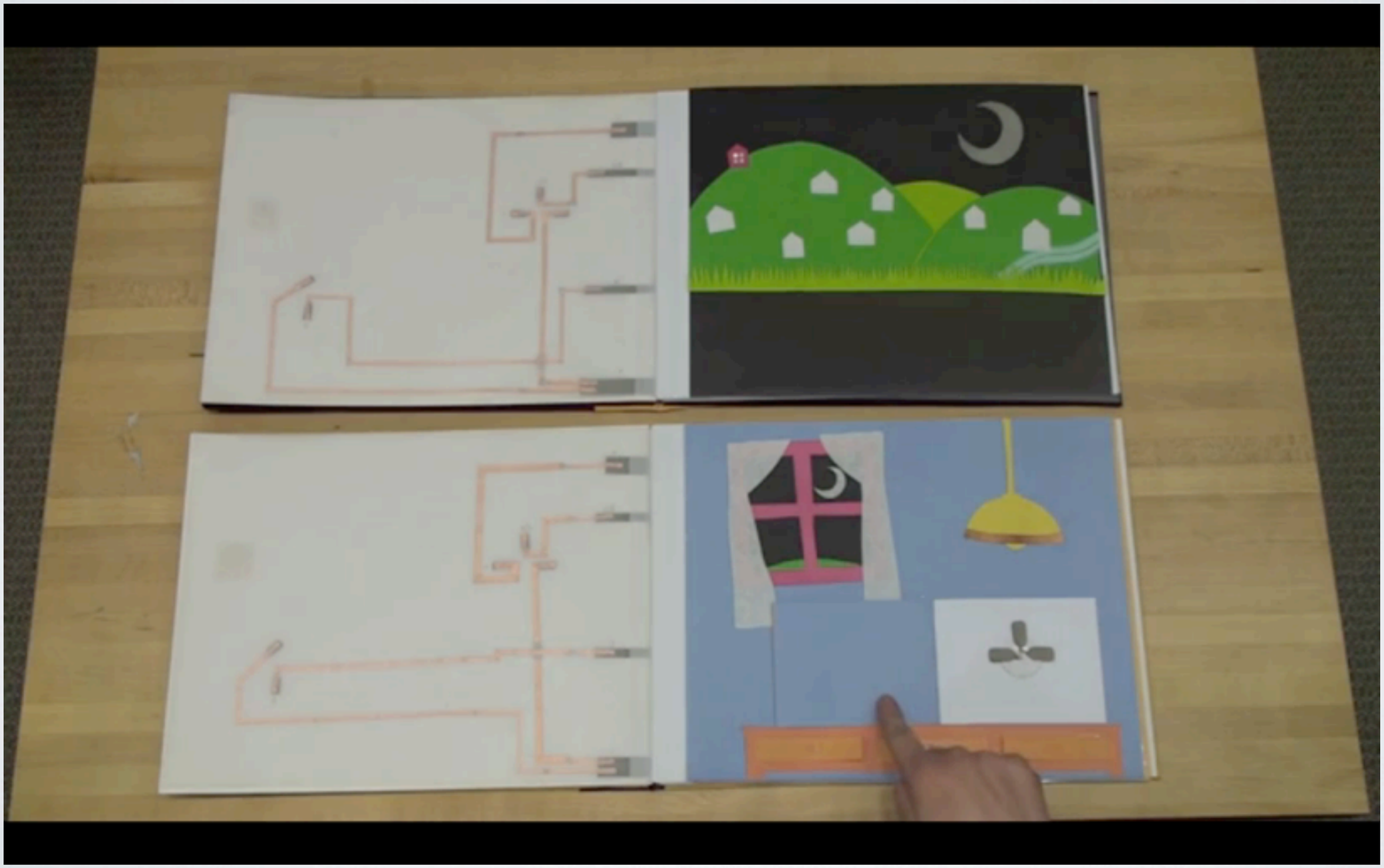
[Buechley, TEI '09]

TEAR DROP



[Freed, IDC '11]

TELESCRAPBOOKS



TAKEAWAYS

- Build lots of prototypes, both parallel and iterative.
- Key is to *fail early and often*
- Good prototyping tools provide a low threshold and a high ceiling.