

Interaction Design & Firmware Programming

September 4, 2018

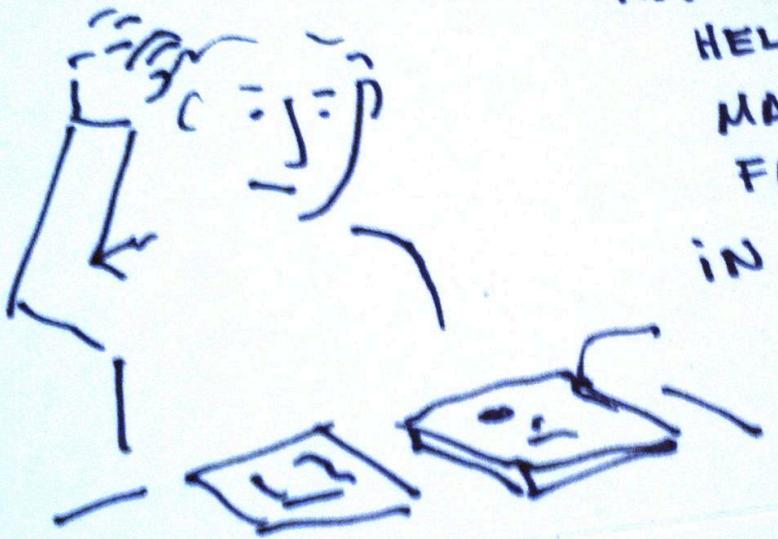
Lab Review!

I like, I wish
Cool Frankenlights
1 hour rule

Hacking

Culture | Understanding how things work
Being rough and ready

HOW CAN WE
HELP OUR STUDENTS
MAKE SENSE +
FIND OPPORTUNITY
IN THE STUFF
THAT SURROUNDS
US ALL?



Hacker:
[a] person who
delights in having an
intimate
understanding of the
internal workings of a
system, computers
and computer
networks in particular



Homebrew: Several very high- profile Silicon Valley hackers and IT entrepreneurs emerged from the DIY computer movement of the mid 1970's.

NEWSLETTER
Homebrew Computer Club

Robert Reiling, Editor □ Post Office Box 626, Mountain View, CA 94042 □ Joel Miller, Staff Writer
Typesetting, graphics and editorial services donated by Laurel Publications, 17235 Laurel Rd., Los Gatos, CA 95030 (408) 363-3609

RANDOM DATA
By Robert Reiling

Computer clubs continue to form around the country... E. Brooner would like to have material to help him get started with the "Flathead Computer Society" in the Kalispell area. His Address is P.O. Box 236, Lakeside, Montana 59922.

Did you see the SOL terminal demonstrated by Bob Marsh at the Sept. 1st meeting? An excellent design that will interest hobbyists and commercial users alike. It's available from Processor Technology, 6200 Hollis St., Emeryville, CA 94608. Write them for prices and specifications.

The OSI Systems Journal has been sent to all OSI customers (free—at least for the time being). It's a bimonthly magazine with plans to go monthly in the future. There are 28 pages in the first issue (August 1976, Vol. 1, No. 1) with a hardware feature covering the OSI 440 Video Graphics System and software, features concerning Tiny BASIC for the 6800 and a Graphics Editor for the 6502. It also includes OSI product and software catalog data. The BASIC is, of course, the 2K Tiny BASIC developed by Tom Pittman. Many of you have met Tom at the Homebrew computer Club meetings. The OSI Systems Journal is a good way to learn more about the OSI computer hardware and software along with helpful user information. The contact address is: The OSI Systems Journal, P.O. Box 134, Hiram, Ohio 44234.

KIM-1 users now have a newsletter. Eric Rehneke is producing the newsletter every 5-8 weeks, MOS Technology, Inc. helped get it started by sending copies to all known KIM owners. The user group, however, is independent of MOS Technology, Inc. The newsletter is devoted to KIM-1 support. Subscriptions are \$5.00 for the next six issues. Contact "KIM-1 User Notes," c/o Eric C. Rehneke, Apt. 207, 7656 Broadway Rd., Parma, Ohio 44134.

The BAMUG club has a new contact address. It is BAMUG, c/o Timothy O'Hare, 1211 Santa Clara Ave., Alameda, CA 94501. Write Timothy for club information. I suggest you include a stamped, self-addressed envelope.

Beware of board snatchers! Glenn Ewing reports 11 boards were taken out of his IMSAI computer. The boards are: MPU, 4 RAM-4's, SIO-2, P10-4, PIC-8, PROM-4, IFM and FIB. Glenn suggests you consider providing good security for your computer and associated equipment. In his case the computer was in a locked office which was burglarized. In the event you have information on the above boards, write Lt. Glenn Ewing, Code 62E1, Naval Post Graduate School, Monterey, CA 93940.

For family and friends of people who always wanted to know about computers, but didn't want to ask them, four easy-going classes are available starting Oct. 19th on Tuesdays from 7 to 9 p.m. You can learn how computers work and what they can and can't do. You will also have some of the jargon deciphered, see what you can do with a computer, play some games and learn to program. The cost is \$25. Contact the Community Computer Center, 1919 Menlo Ave., Menlo Park, CA 94025, phone (415) 325-4444.

A call for papers in personal computing has been issued by the 1977 National Computer Conference. The conference is scheduled for June 13-16, 1977. I have a few copies of the guidelines if you would like to submit a paper.

The First West Coast Computer Faire will be held April 16 and 17, 1977 at the San Francisco Civic Auditorium. This faire is shaping up rapidly. If you would like to lead a conference or participate in a conference session, please contact me. More information about the Faire is in the accompanying article.□

THE FIRST WEST COAST COMPUTER FAIRE
A Call For Papers And Participation

The San Francisco Bay Area is finally going to have a major conference and exhibition exclusively concerned with personal and home computing—The First West Coast Computer Faire. And, it promises to be a massive one! It will take place in the largest convention facility in Northern California: The Civic Auditorium in San Francisco. It will be a two-and-a-half day affair, starting on Friday evening and running through Sunday evening, April 15-17.

It is being sponsored by a number of local and regional hobbyist clubs, educational organizations and professional groups. These include:

•The two largest amateur computer organizations in the United States—the Homebrew Computer Club and the Southern California Computer Society.
•Both of the Bay Area chapters of the Association Of Computing Machinery—the San Francisco Chapter and the Golden Gate Chapter
•Stanford University's Electrical Engineering Department

I



ifixit



Photo from articulate.com



Photo from Ambidextrous Mag



Photo from photobucket: pacapo



Photo from Bunnie Studios

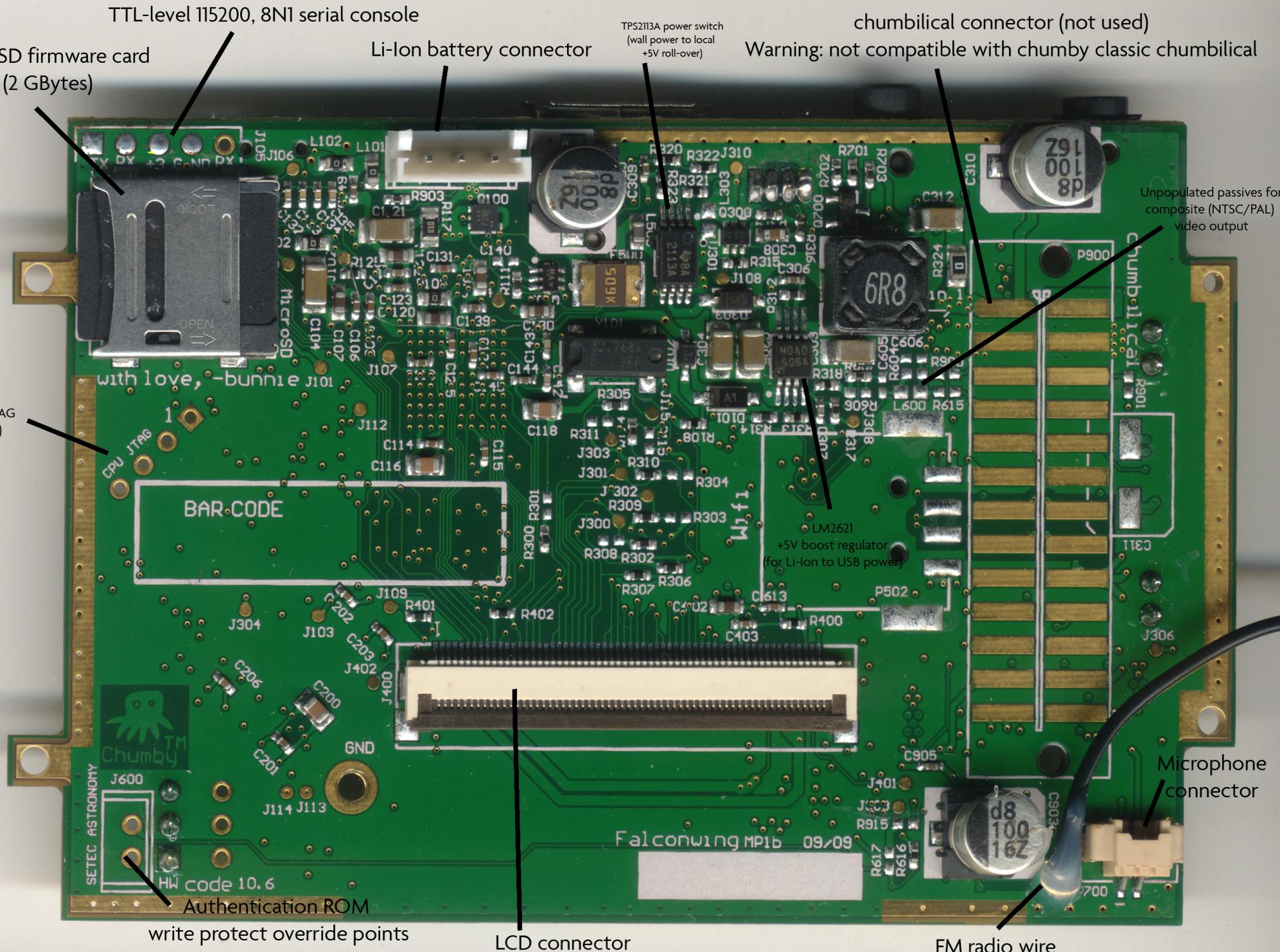
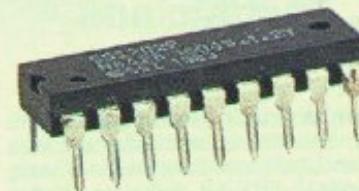


Photo from Bunnie Studios

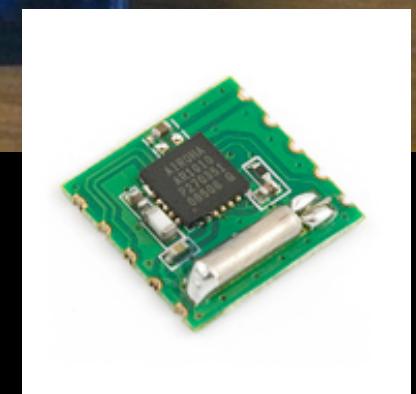


FM Receiver On-A-Chip



11.95

TDA7000. Combines RF, mixer, IF and demodulator stages in one monolithic IC! Mute circuit reduces spurious reception. Frequency-locked-loop system with non critical 70 KHz IF. With data. 276-1304 .11.95





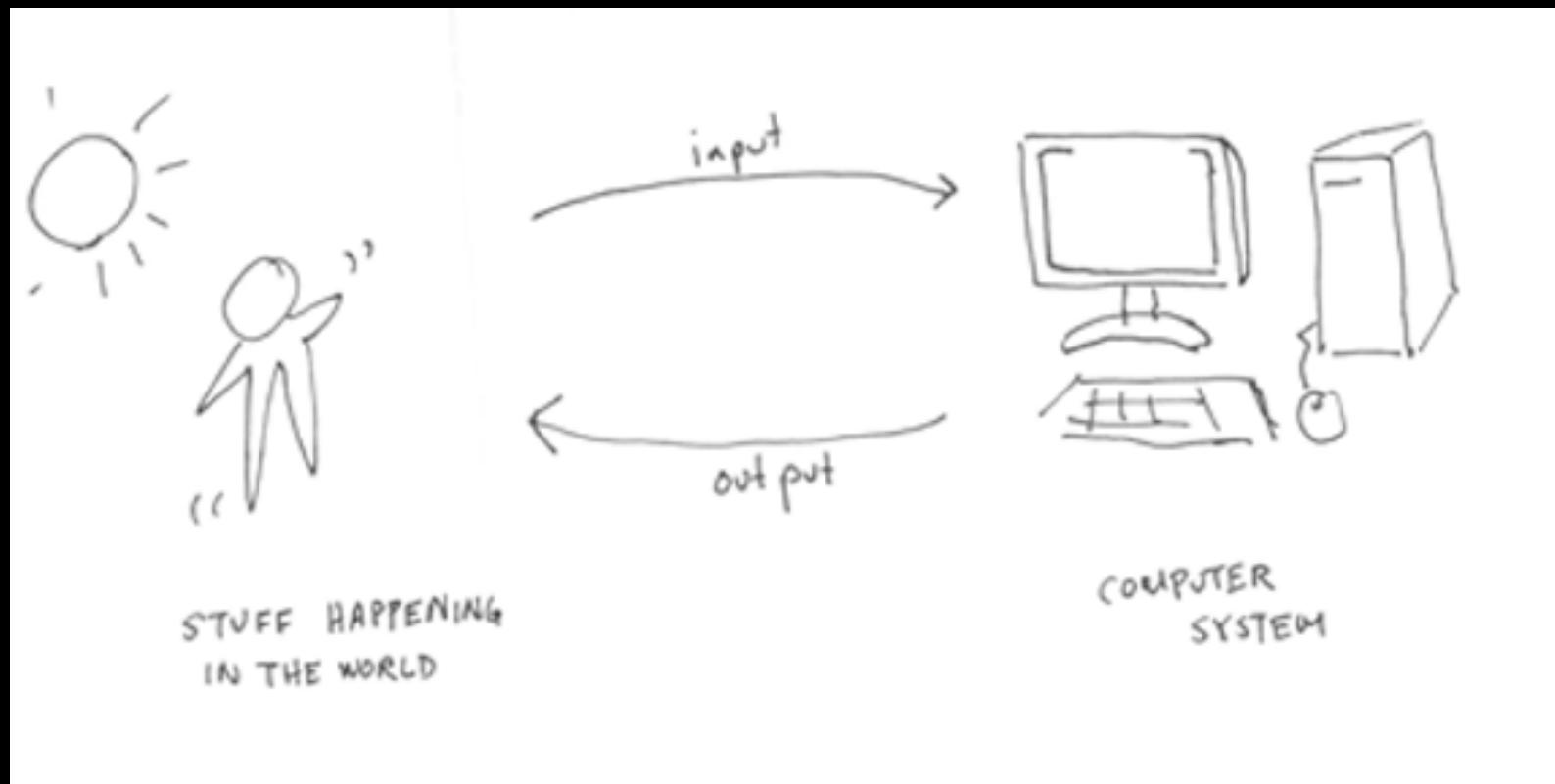
An
inflatable
corset



Multimeter demonstration

Voltage
Resistance
Connections

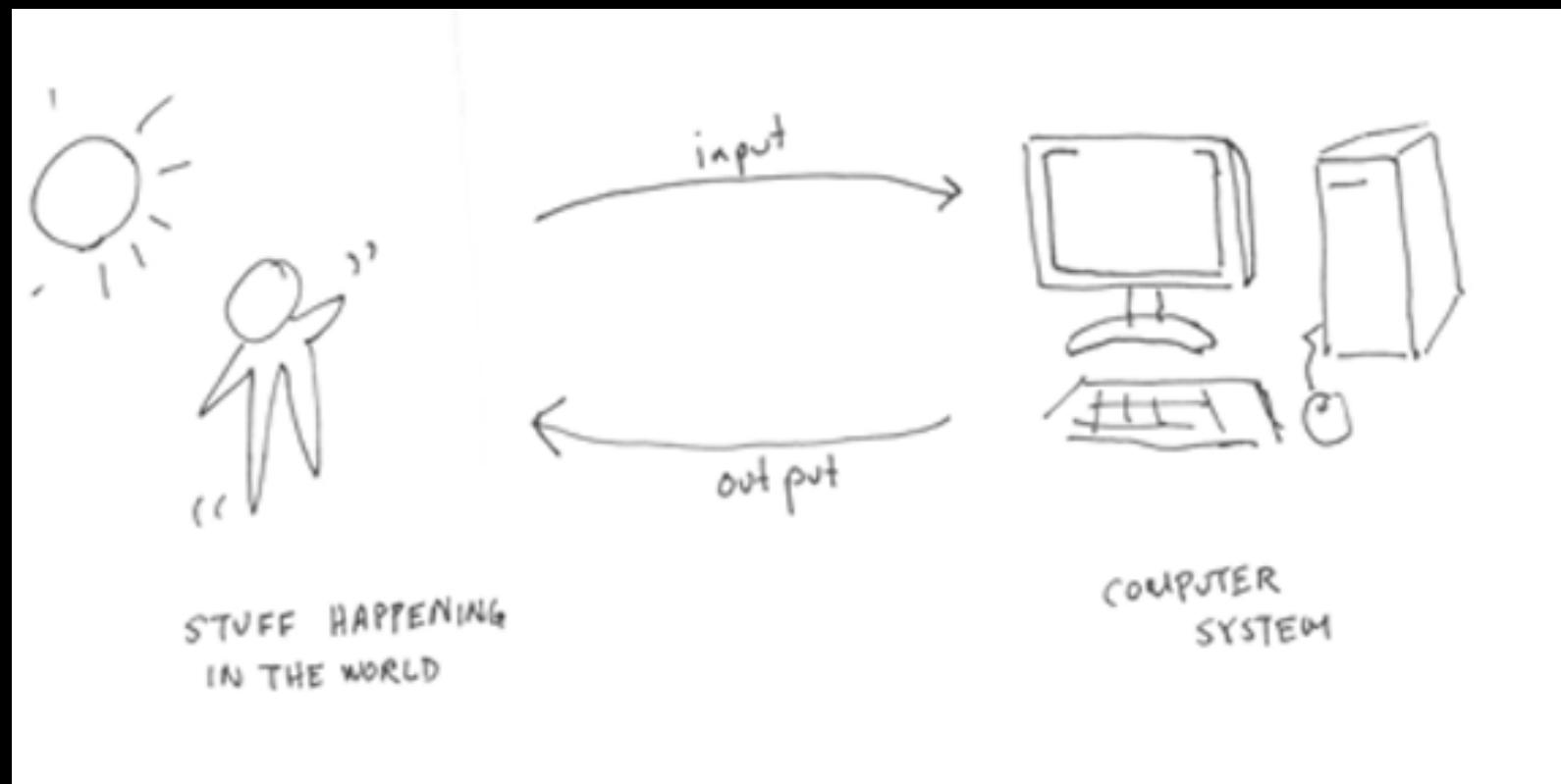
This week's preLab



How do we sketch ideas for interactions?

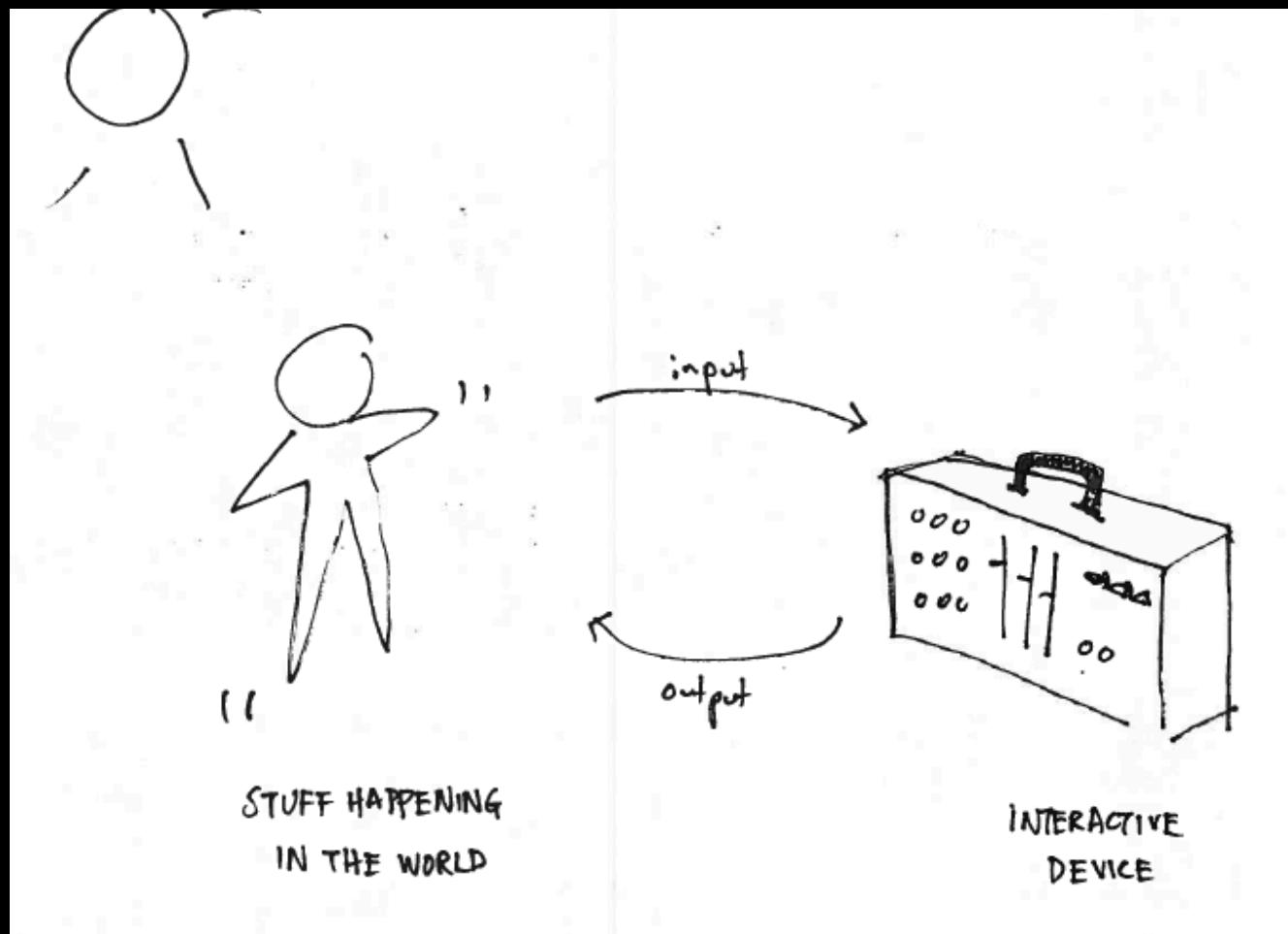
Interacting with Interactive Devices

some sketches



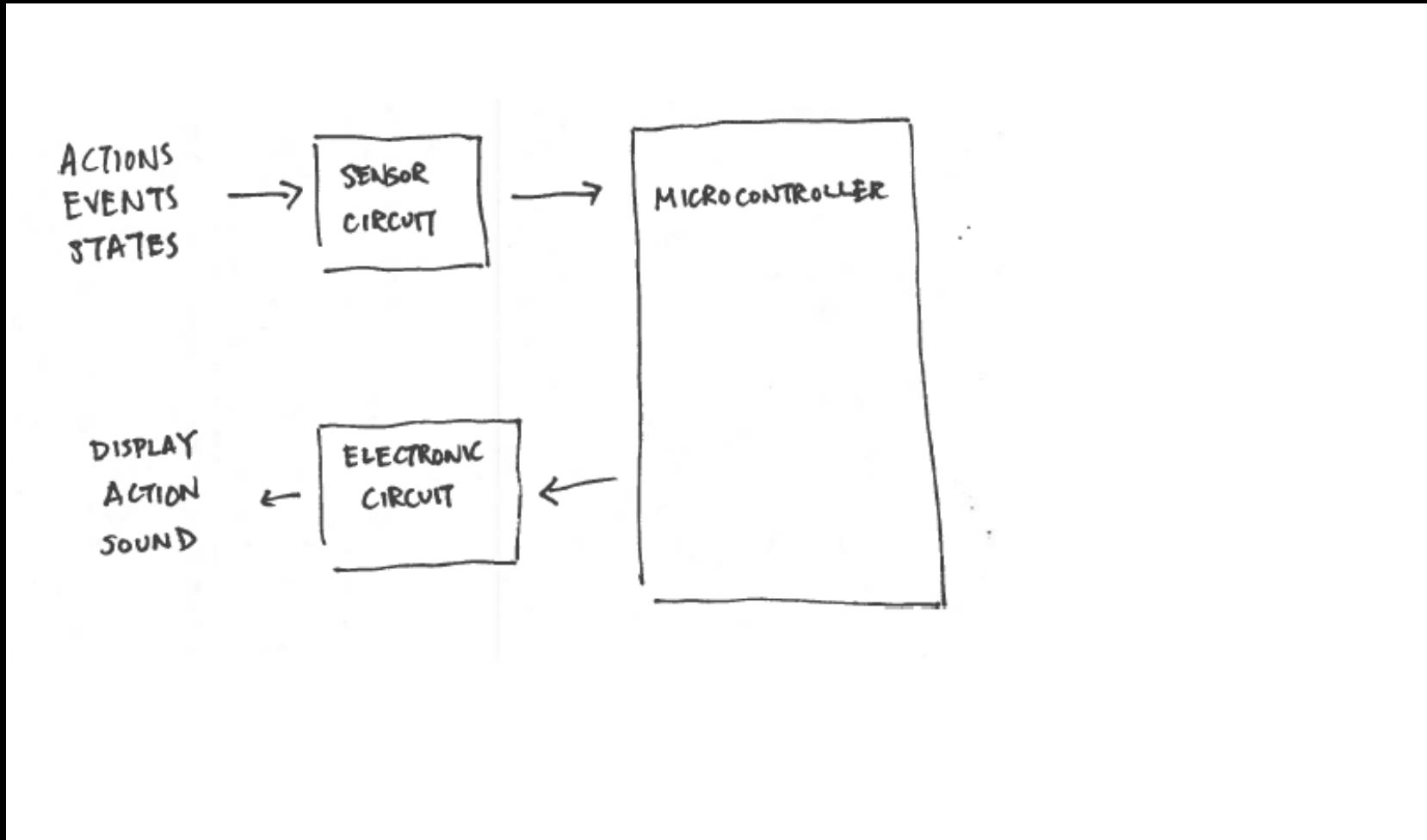
Interacting with Interactive Devices

some sketches



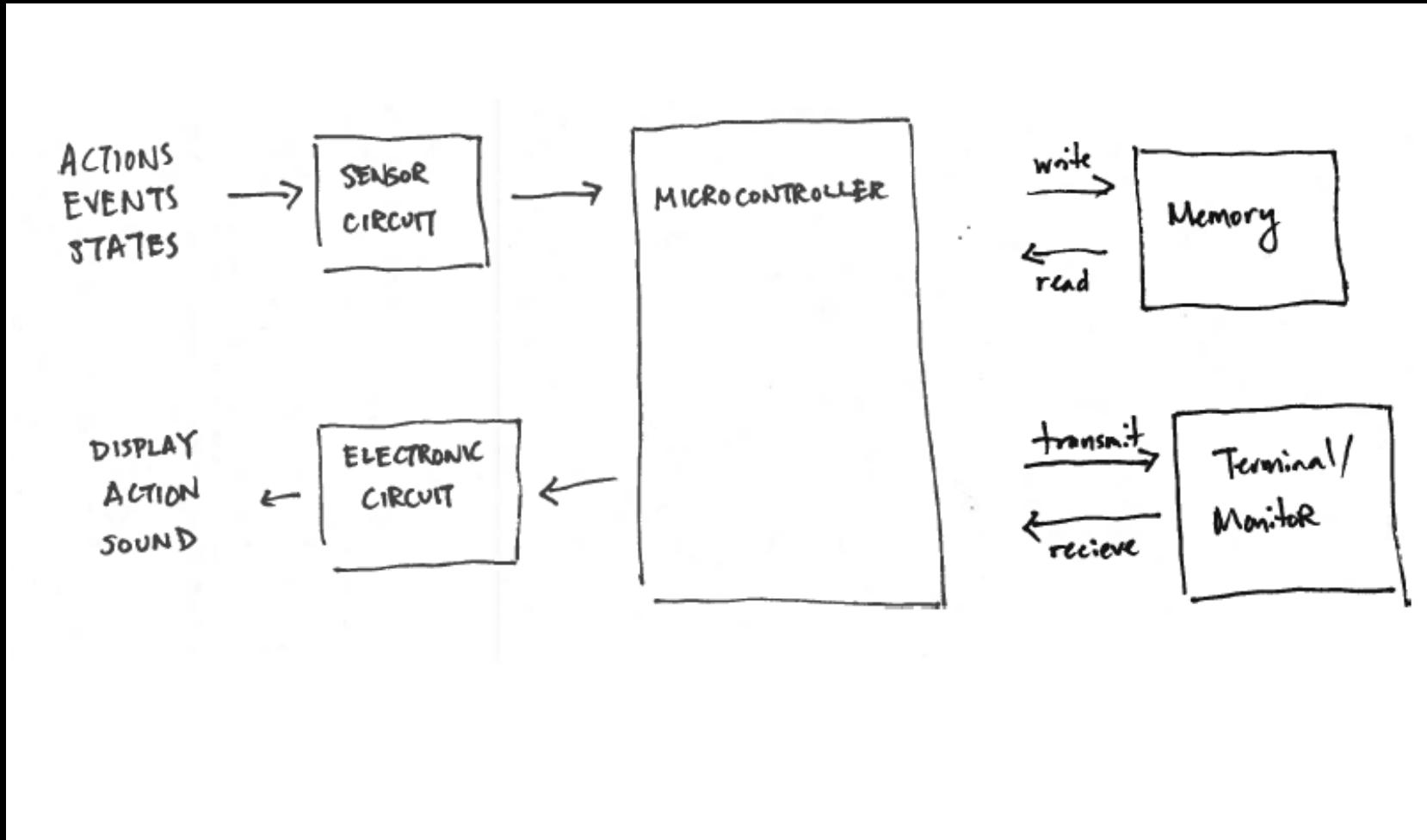
Interacting with Interactive Devices

some sketches



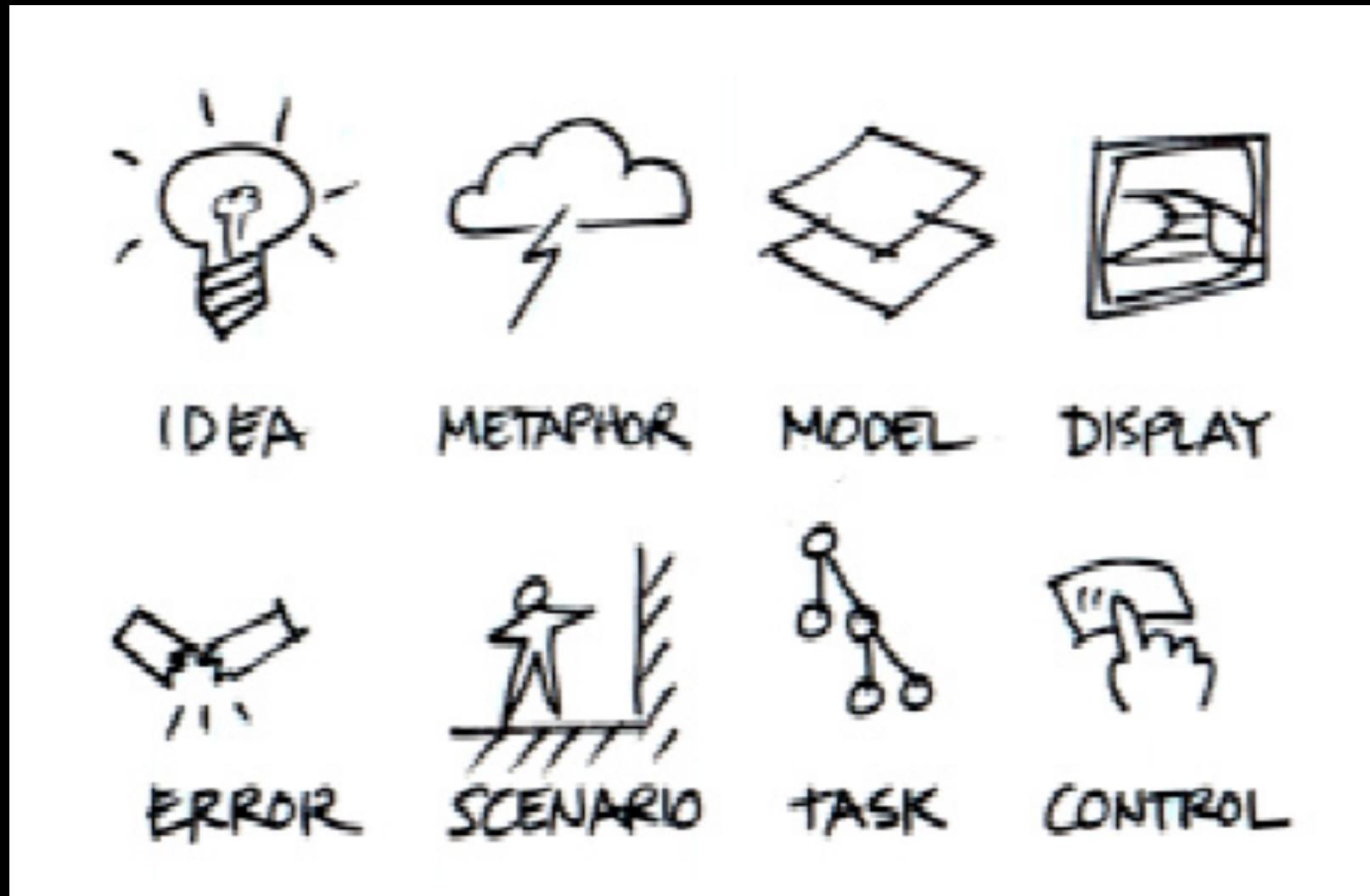
Interacting with Interactive Devices

some sketches



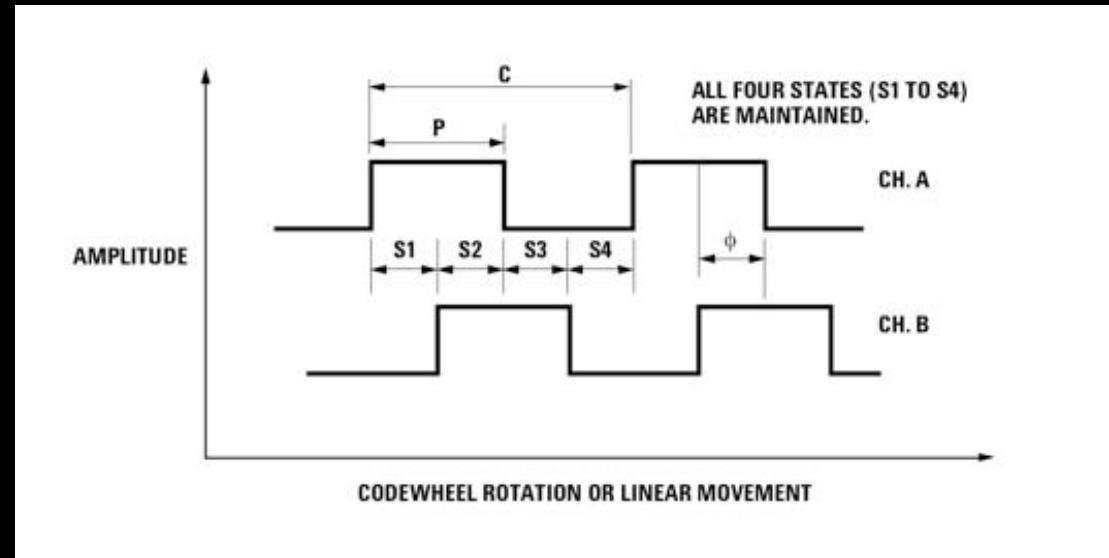
Interacting with Interactive Devices

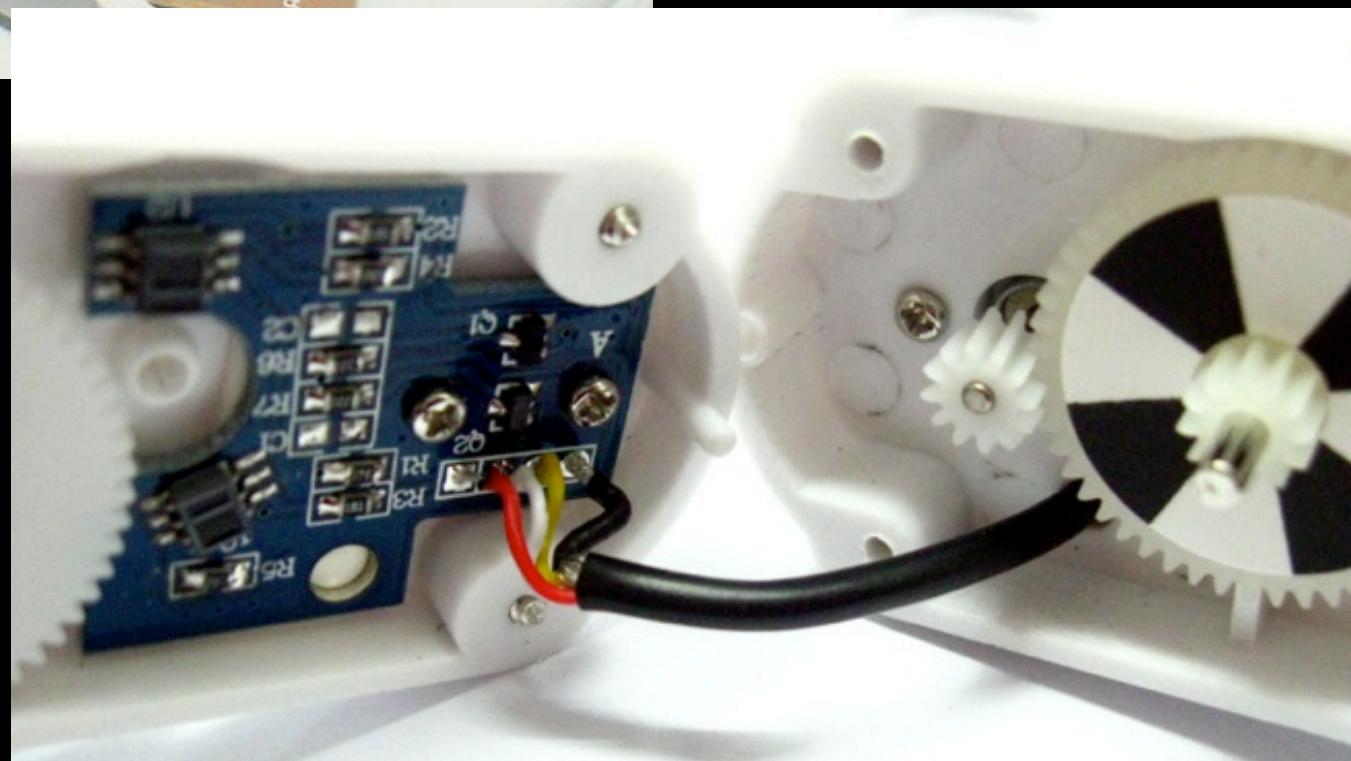
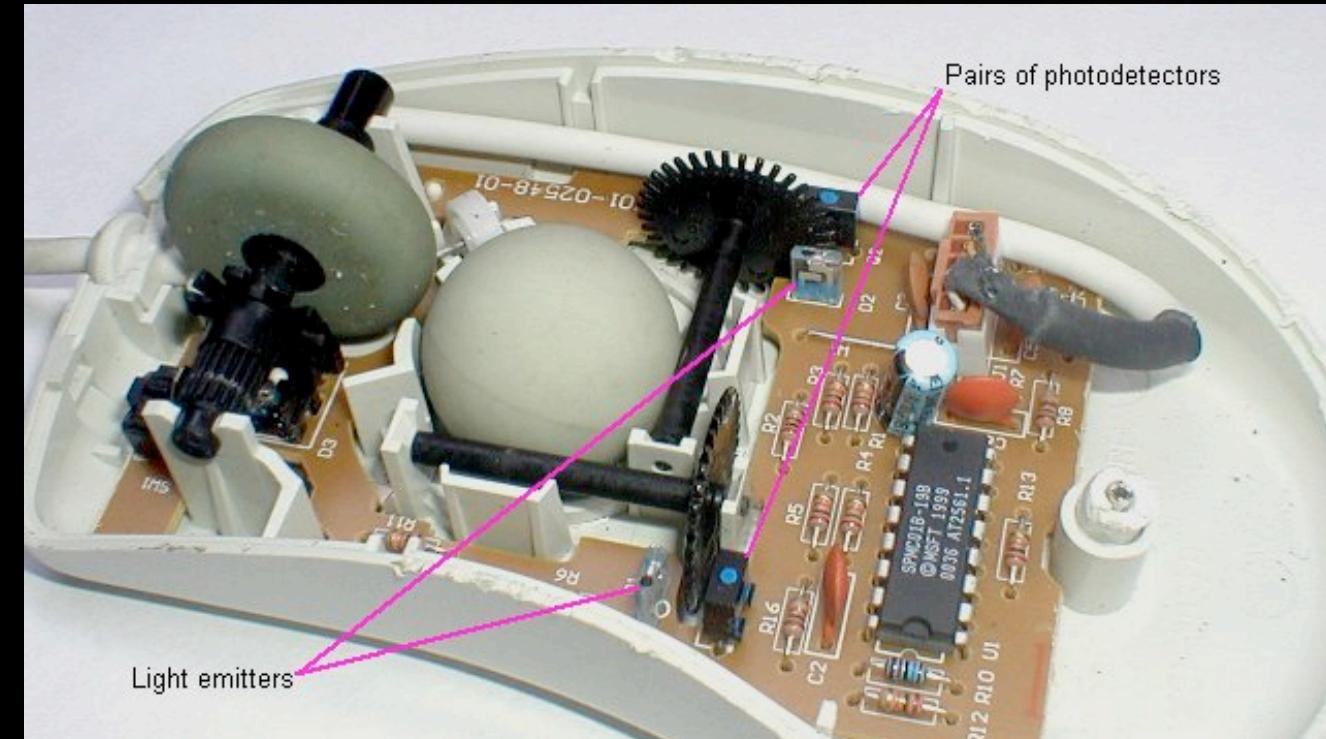
some sketches

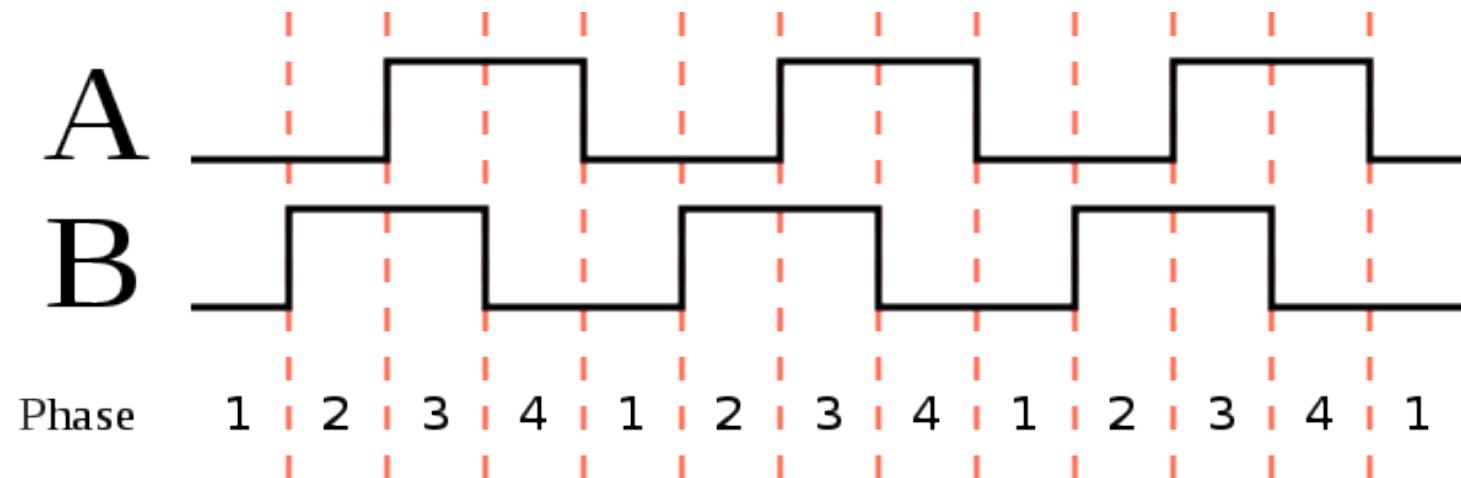
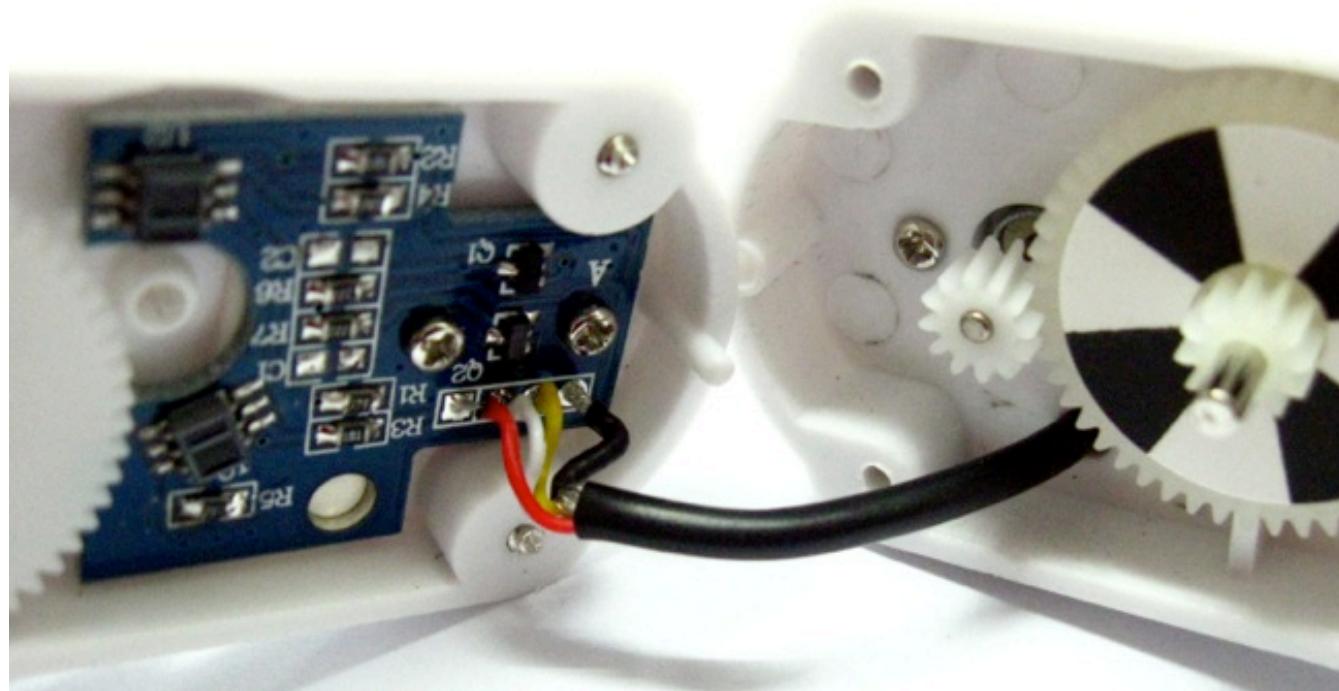


Sensors

Time/Count varying sensors







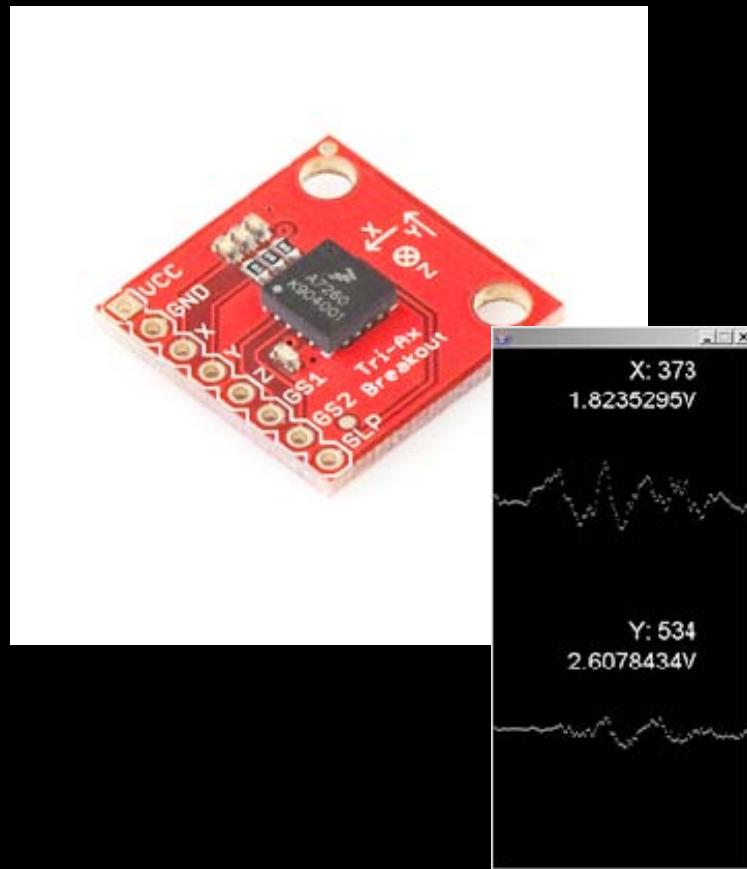
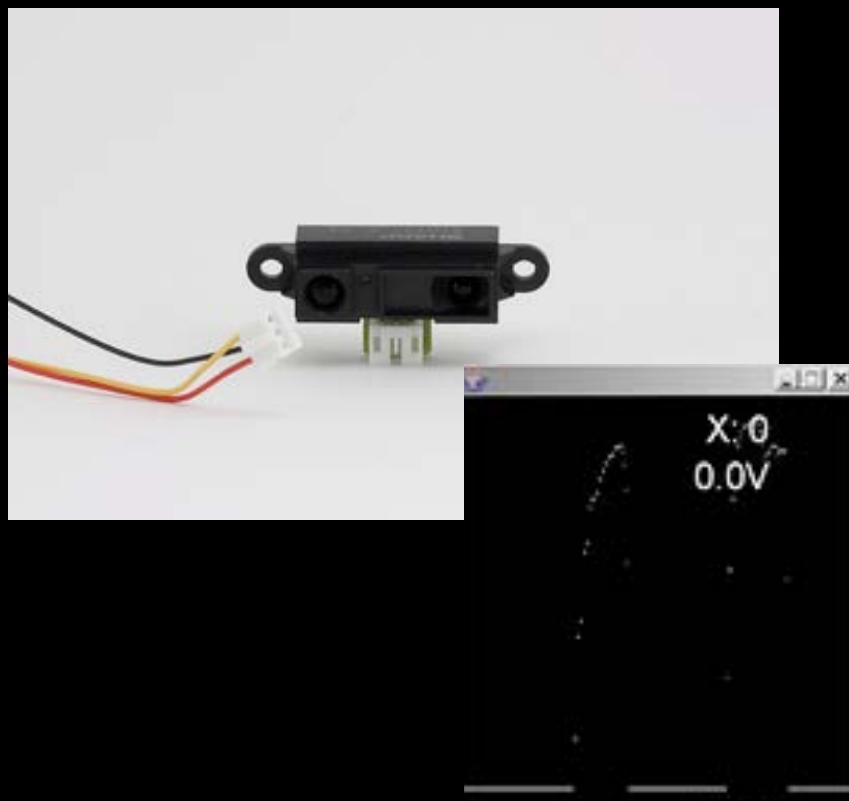
Sensors

Resistance varying sensors



Sensors

Voltage varying sensors



Where do we get cool sensors, displays and actuators?

Tinkersphere (152 Allen St.)

Adafruit

Sparkfun

Jameco

Mouser

Digikey

Firmware Programming

Arduino IDE review, questions
Behind the Scenes
Finite State Machines
Modules

Where were we?

Set preferences to see verbose mode
when compiling, uploading.
Let's look at the files.

Blink | Arduino 1.0.1

Verify

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.
// Pin 11 has the LED on Teensy 2.0
// Pin 6 has the LED on Teensy++ 2.0
// 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);
}

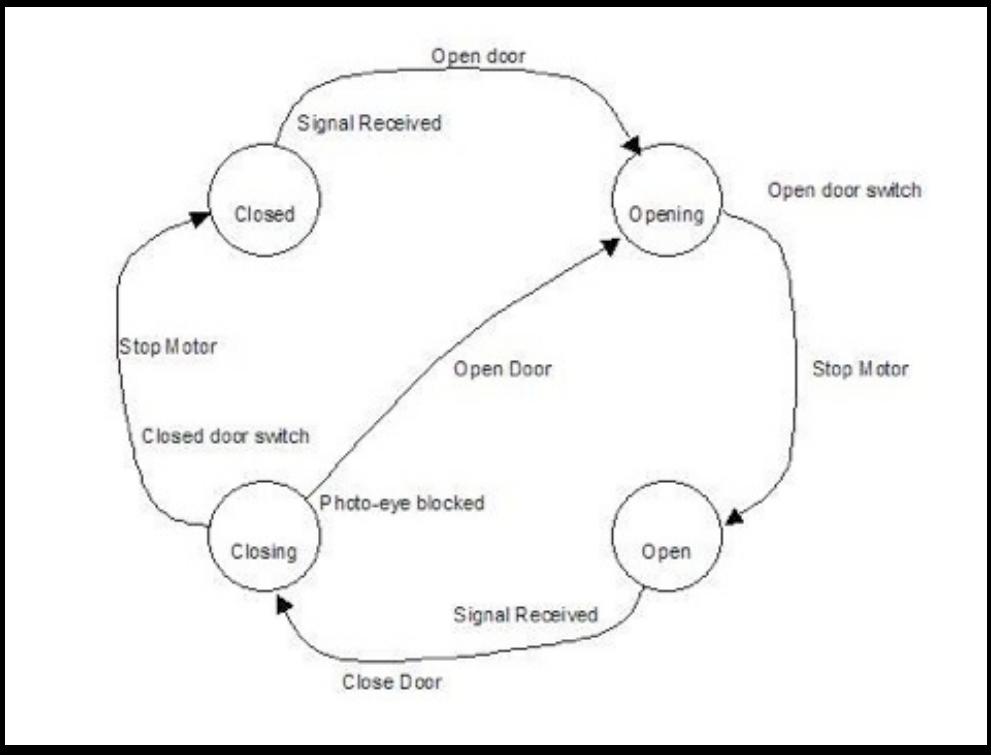
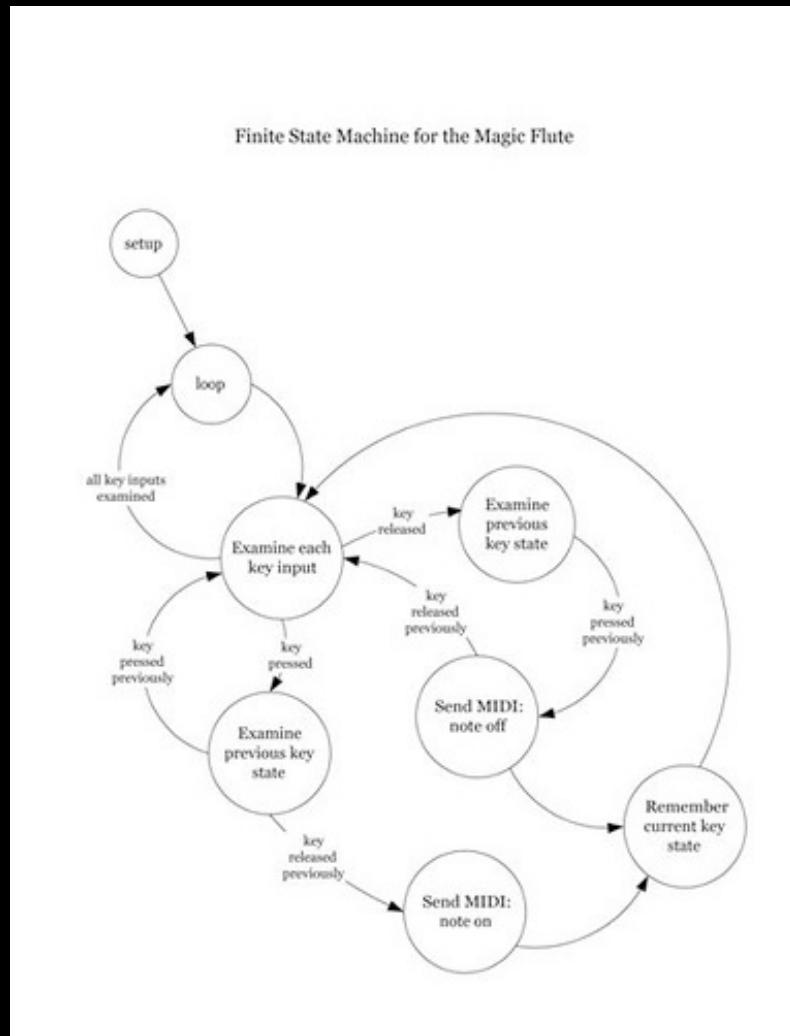
// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);    // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);     // turn the LED off by making the voltage LOW
  delay(1000);               //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

2 null on /dev/tty.usbmodem1411

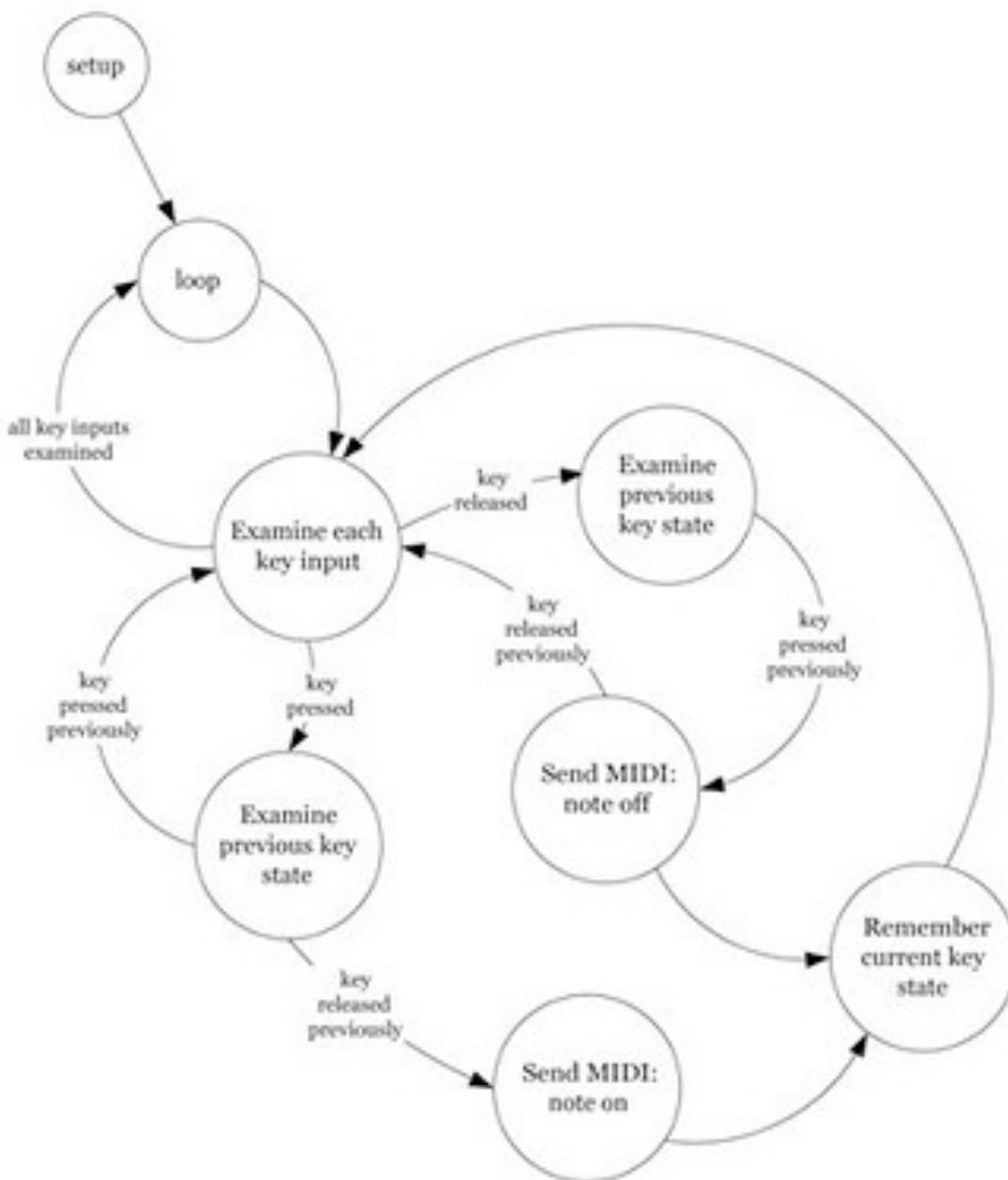
```
drwxr-xr-x 21 wendyju wendyju 714 Apr 11 15:43 .
drwx----- 16 wendyju wendyju 544 Apr 11 15:47 ..
-rw-r--r-- 1 wendyju wendyju 559 Apr 11 15:42 Blink.cpp
-rw-r--r-- 1 wendyju wendyju 13 Apr 11 15:43 Blink.cpp.eep
-rwxr-xr-x 1 wendyju wendyju 13913 Apr 11 15:43 Blink.cpp.elf
-rw-r--r-- 1 wendyju wendyju 2881 Apr 11 15:43 Blink.cpp.hex
-rw-r--r-- 1 wendyju wendyju 3716 Apr 11 15:42 Blink.cpp.o
-rw-r--r-- 1 wendyju wendyju 17868 Apr 11 15:43 HardwareSerial.cpp.o
-rw-r--r-- 1 wendyju wendyju 31996 Apr 11 15:43 Print.cpp.o
-rw-r--r-- 1 wendyju wendyju 16264 Apr 11 15:43 Tone.cpp.o
-rw-r--r-- 1 wendyju wendyju 5676 Apr 11 15:43 WInterrupts.c.o
-rw-r--r-- 1 wendyju wendyju 7068 Apr 11 15:43 WMath.cpp.o
-rw-r--r-- 1 wendyju wendyju 57548 Apr 11 15:43 WString.cpp.o
-rw-r--r-- 1 wendyju wendyju 184770 Apr 11 15:43 core.a
-rw-r--r-- 1 wendyju wendyju 3168 Apr 11 15:43 main.cpp.o
-rw-r--r-- 1 wendyju wendyju 3288 Apr 11 15:42 pins_arduino.c.o
-rw-r--r-- 1 wendyju wendyju 9392 Apr 11 15:43 wiring.c.o
-rw-r--r-- 1 wendyju wendyju 6776 Apr 11 15:43 wiring_analog.c.o
-rw-r--r-- 1 wendyju wendyju 9256 Apr 11 15:43 wiring_digital.c.o
-rw-r--r-- 1 wendyju wendyju 6812 Apr 11 15:43 wiring_pulse.c.o
-rw-r--r-- 1 wendyju wendyju 5344 Apr 11 15:43 wiring_shift.c.o
```

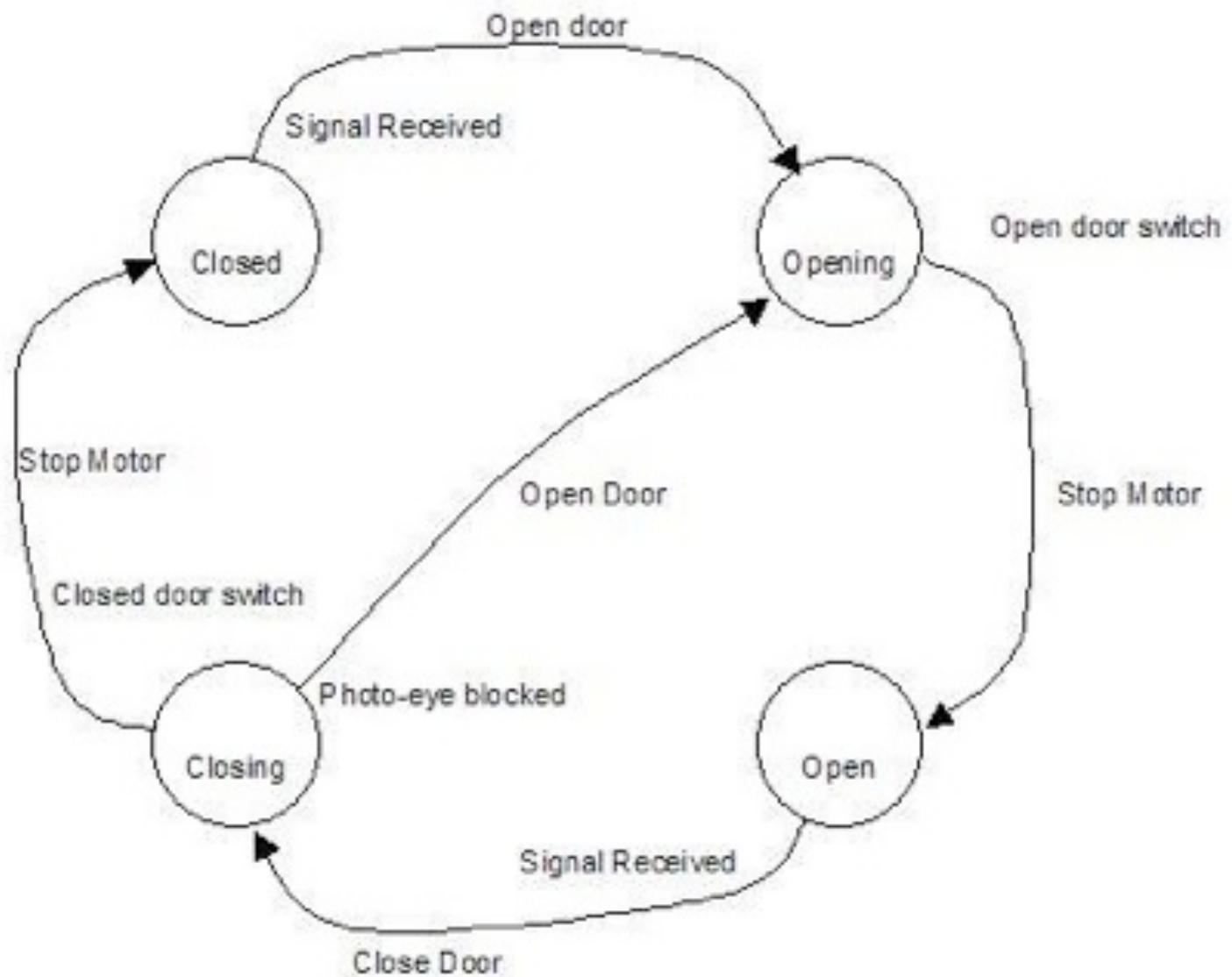
Finite State Machine Model

Sketching Interactive Device Behavior



Finite State Machine for the Magic Flute



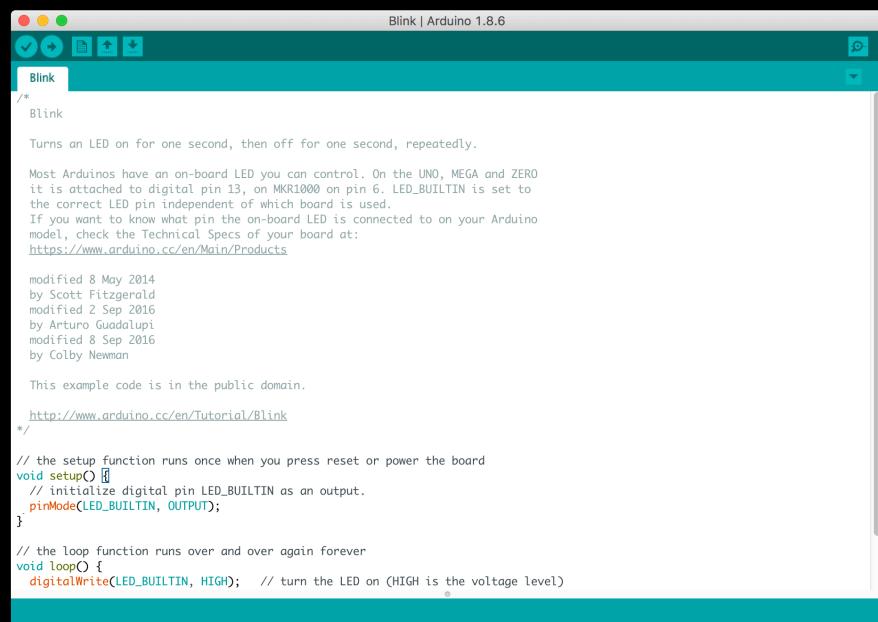


Adapting Found Code

How do we merge two programs?

Adapting Found Code

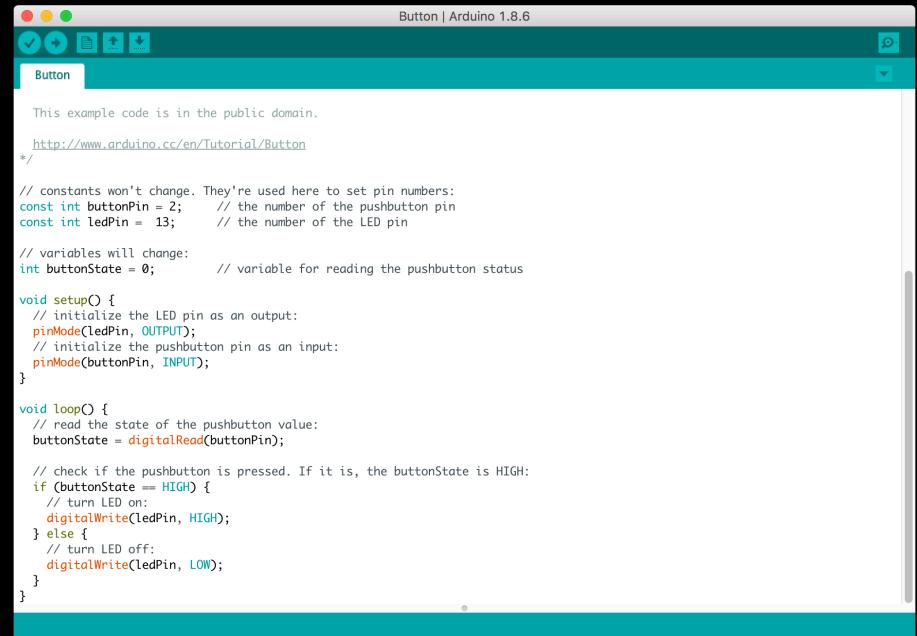
How do we merge two programs?



The screenshot shows the Arduino IDE interface with the title bar "Blink | Arduino 1.8.6". The code editor contains the standard Blink sketch. The code is as follows:

```
/*
 * Blink
 *
 * Turns 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, MEGA and ZERO
 * it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
 * 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 Arduino
 * model, check the Technical Specs of your board at:
 * https://www.arduino.cc/en/Main/Products
 *
 * modified 8 May 2014
 * by Scott Fitzgerald
 * modified 2 Sep 2016
 * by Arturo Guadalupi
 * modified 8 Sep 2016
 * by Colby Newman
 *
 * This example code is in the public domain.
 * http://www.arduino.cc/en/Tutorial/Blink
 */
// the setup function runs once when you press reset or power the board
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 voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);     // turn the LED off (LOW is the voltage level)
  delay(1000);                      // wait for a second
}
```



The screenshot shows the Arduino IDE interface with the title bar "Button | Arduino 1.8.6". The code editor contains the Button sketch. The code is as follows:

```
This example code is in the public domain.
http://www.arduino.cc/en/Tutorial/Button
*/
// constants won't change. They're used here to set pin numbers:
const int buttonPin = 2;      // the number of the pushbutton pin
const int ledPin = 13;         // the number of the LED pin

// variables will change:
int buttonState = 0;          // variable for reading the pushbutton status

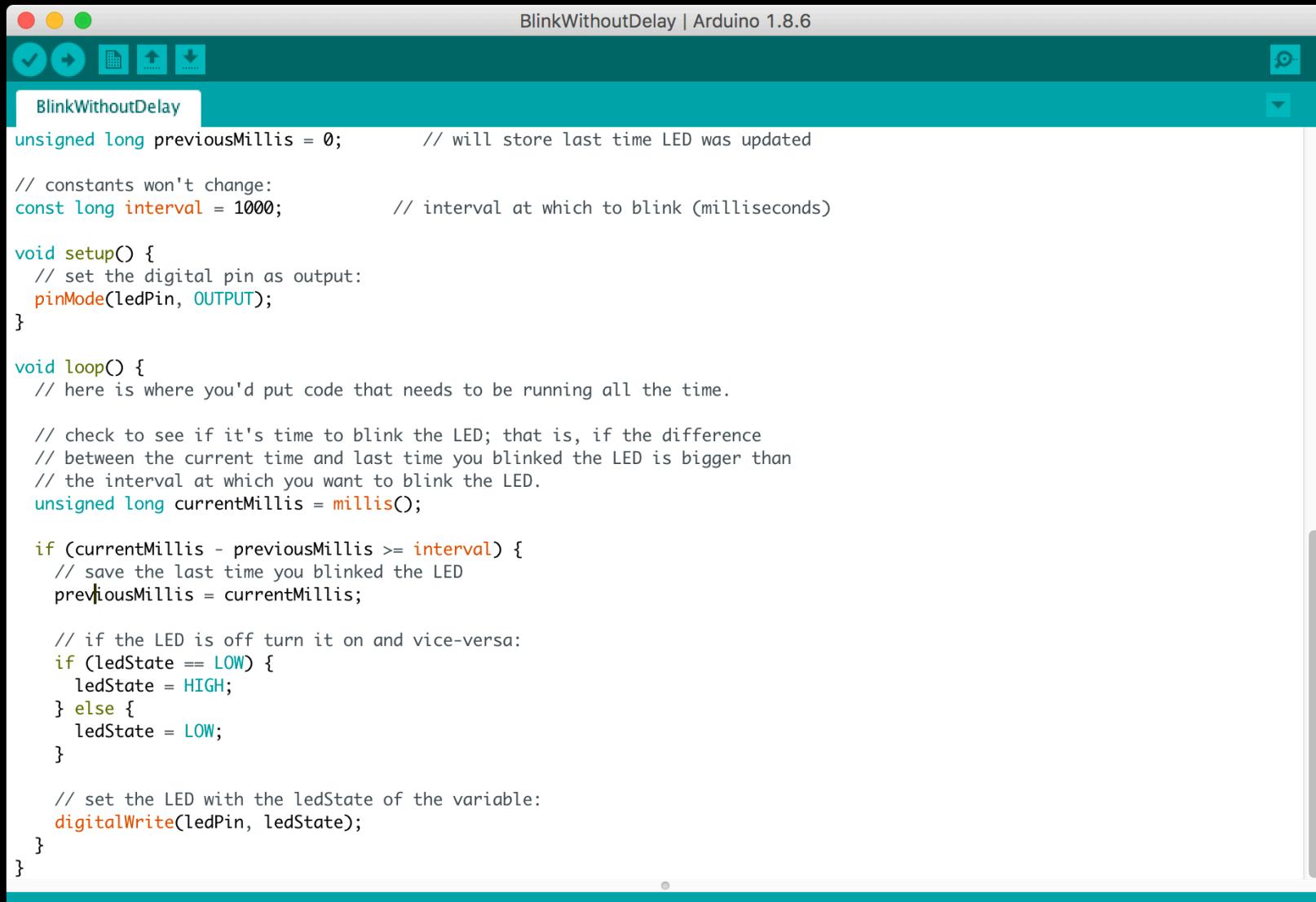
void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

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

  // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  } else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

Non-blocking Code

How do we write code that doesn't block execution?



The screenshot shows the Arduino IDE interface with the title bar "BlinkWithoutDelay | Arduino 1.8.6". The code editor contains the following non-blocking code:

```
unsigned long previousMillis = 0;          // will store last time LED was updated

// constants won't change:
const long interval = 1000;                // interval at which to blink (milliseconds)

void setup() {
  // set the digital pin as output:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // here is where you'd put code that needs to be running all the time.

  // check to see if it's time to blink the LED; that is, if the difference
  // between the current time and last time you blinked the LED is bigger than
  // the interval at which you want to blink the LED.
  unsigned long currentMillis = millis();

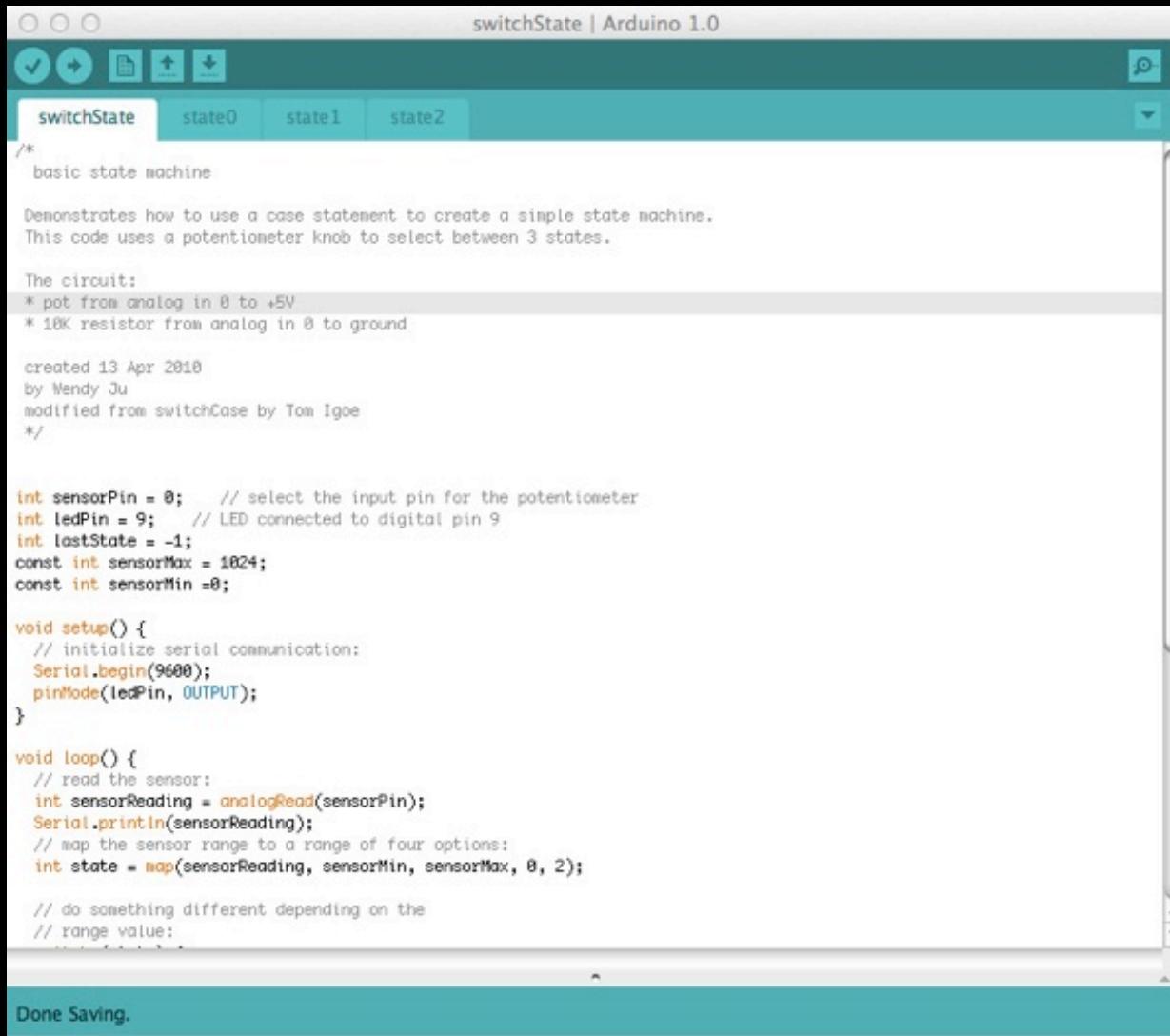
  if (currentMillis - previousMillis >= interval) {
    // save the last time you blinked the LED
    previousMillis = currentMillis;

    // if the LED is off turn it on and vice-versa:
    if (ledState == LOW) {
      ledState = HIGH;
    } else {
      ledState = LOW;
    }

    // set the LED with the ledState of the variable:
    digitalWrite(ledPin, ledState);
}
```

Modules & Modes

Organizing functions and Behaviors



The screenshot shows the Arduino IDE interface with the title bar "switchState | Arduino 1.0". The code editor contains a sketch named "switchState". The code is a basic state machine example. It includes comments explaining the purpose, circuit setup, and history. The code defines variables for sensor and LED pins, initializes serial communication, and implements a loop that reads the sensor value, maps it to a state (0, 1, or 2), and then performs different actions based on the state.

```
switchState | Arduino 1.0

switchState state0 state1 state2

/*
  basic state machine

Demonstrates how to use a case statement to create a simple state machine.
This code uses a potentiometer knob to select between 3 states.

The circuit:
* pot from analog in 0 to +5V
* 10K resistor from analog in 0 to ground

created 13 Apr 2010
by Wendy Ju
modified from switchCase by Tom Igoe
*/

int sensorPin = 0;    // select the input pin for the potentiometer
int ledPin = 9;      // LED connected to digital pin 9
int lastState = -1;
const int sensorMax = 1024;
const int sensorMin = 0;

void setup() {
  // initialize serial communication:
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the sensor:
  int sensorReading = analogRead(sensorPin);
  Serial.println(sensorReading);
  // map the sensor range to a range of four options:
  int state = map(sensorReading, sensorMin, sensorMax, 0, 2);

  // do something different depending on the
  // range value:
  if (state == 0) {
    digitalWrite(ledPin, HIGH);
  } else if (state == 1) {
    digitalWrite(ledPin, LOW);
  } else if (state == 2) {
    digitalWrite(ledPin, HIGH);
  }
}
```

Done Saving.