Everyone’s had a little programming experience, right? Who’s had any electrical experience?
What’s for Today

- Introduction to Arduino
- Building an LED flashlight
- Making some blinky LED eyes
What’s in your goodie bag
Class Kit Manifest

- Arduino NG USB board
- Arduino ProtoShield
- Solderless breadboard
- USB cable
- RC servo
- piezo buzzer
- 6m hookup wire in ghastly colors
- potentiometer with knob
- R,G,B and mystery LEDs
- two push switches
- 9V battery and connector
- 220, 330, 10k, and 1M resistors
- light sensitive resistor
- 5.1v zener diode
- square of velcro
- scary eyeballs

And other bits as we progress
Ignore most of the kit for now, just use Arduino board and USB cable
Complete kit manifest with part numbers will be online
A Word on Safety

- Electronics are toxic to you
  - Lead in some of the parts
  - Wash up afterwards

- You are toxic to electronics
  - Static-sensitive: don’t shuffle your feet
  - Wires only bend so much
What is Arduino?

- Open Source Physical Computing Platform
  - open source: free to inspect & modify
  - physical computing. er, what? ubiquitous computing, pervasive computing, ambient intelligence, calm computing, everyware, spimes, blogjects, smart objects...

- A physical board, a programming environment, a development philosophy

- Tiny computer you can program
  - Completely stand-alone, talks to other devices

Physical computing as invisible computing
Can run off a battery
Can talk to other computers, cell phones, etc.
What is Arduino?

• Based on AVR-GCC, avr-libc, AVRlib and Processing (all open source projects)
• Very similar to Basic Stamp (if you know it)
  • but cheaper, faster, & open
• Uses AVR ATmega8 microcontroller chip

Basic Stamp uses PIC microcontroller chip.
PICs and AVRs are very comparable, one’s not necessarily better than the other
AVRs are a little better if you’re using a language like C (stack-based)
Don’t need to worry about the chip particulars for now
What is Arduino?

- Why not just use a bare AVR ATmega8 chip?

- Arduino is also a standardized “bootloader”
  - A tiny program that loads other programs
  - It’s alive during first 5 seconds

A bootloader is akin to an BIOS on a real computer. It handles the startup of the chip. After 5 seconds, your program runs. Don’t need special programmer board with a bootloader. Arduino can work with other AVR chips, some are smaller than your fingernail, cost ~ 40 cents.
What is Arduino?

• Capabilities
  • 8 kBytes of Flash program memory
  • 1 kByte of RAM
  • 12 MHz (Apple II: 1 MHz)
• Inputs and Outputs
  • 13 digital input/output pins
  • 5 analog input pins

Digital I/O can read switches and buttons, control LEDs and motors
Analog input can read knobs or other varying sensors
Analog output can be done with PWM
What is Arduino?

But how do you program it?

• Write programs on your PC
• Download them into the Arduino board
• Arduino board can then be used by itself
Arduino Board

- Test LED
- Digital input/output
- Power LED
- TX/RX LEDs
- Reset button
- Analog input
- USB input
- Power input
- ICSP programming header

Also: USB input, power input, ICSP programming header
Arduino Board

Diagrammatic version, to simplify
But of a slightly older version of the board
Digital? Analog?

- Digital – only has two values: on/off
- Analog – has many (infinite) values

Computers don’t really do analog

So they fake it, with quantization

Quantization = breaking up the analog range into bins. The number of bins is the resolution. More bins = higher accuracy, but is more complex. Digital can be thought of as only two bins.
That’s the full code for blinking an LED, btw. Arduino defines several useful functions like digitalWrite() and delay(). More on that later.

Processing and Wiring not needed.
Arduino & Processing

http://processing.org/

build generative art or other applets easily

not needed for Arduino, but can work with it

Arduino has essentially the same GUI as Processing
Easier than Arduino, since all software
Though similar UI and philosophy, Arduino is a different language
We’ll use Processing later in the class to let the computer control Arduino & vice-versa
Installing Arduino

- Download software: http://arduino.cc/
- Mac OS X PPC or Intel (must pick)
- Windows 2000/XP
- Install drivers
  - In “drivers” folder, pick appropriate one
  - Windows: unzip driver, plug in board, setup
  - “macosx-setup-command” for Mac folk
- Reboot

Different Arduino downloads for each operating system
Different drivers for each OS too
“macosx-setup-command” must be run before reboot, but, it will go away in next version
Using Arduino

- Programs are called “sketches”
- Load up example sketch “led_blink”
Errors

“Programmer is not responding”

Must select serial port

Programmer is not responding, RESET the board right before exporting
What’s my serial port?

Mac: It’s called “/dev/tty.usbserial-something”
Windows: Use Device Manager to find COM port
Using Arduino

- Write program
- Compile (check for errors)
- Reset board
- Upload to board

Try it out with "led_blink"!

On reset, board will flash on-board pin 13 LED really fast for a split-second to indicate bootloader exists.
When uploading, TX/RX lights will flash as data is transferred.
Then the board resets, pin 13 will flash fast again.
Finally, your program will run.

```c
void setup() {
  pinMode(ledPin, OUTPUT);  // sets t
}
void loop() {
  digitalWrite(ledPin, HIGH);  // sets t
  delay(1000);  // waits
  digitalWrite(ledPin, LOW);  // sets t
  delay(1000);  // waits
}
```
Arduino Board Lifecycle

- Power on
- Reset
  - Blinks p13 fast
  - Waits 5 secs for upload
  - Upload new sketch!
- Execute existing sketch!
- Press reset
Take a Break
Making Circuits

heart pumps, blood flows  voltage pushes, current flows

Ventricles

Pulmonary Arteries

Venae Cavae

Aorta

Right Atrium

Left Atrium

Right Ventricle

Left Ventricle

current

voltage
LEDs

- LED = Light-Emitting Diode
  - electricity only flows one way in a diode
- Needs a “current limiting” resistor, or burns out

Many types of integrated lenses.
Some project a narrow beam (like the ones in this class), some project a very wide beam
All LED circuits are essentially this: power source, current limiter, LED

Flat part of LED goes to negative, like bar in schematic

The higher the resistance, the dimmer the LED; the lower, the brighter
LED flashlight

Take out solderless breadboard, resistor, LED, and battery and make a circuit.
LEDs have been marked a little as to what color they are, but color doesn’t matter here.
Solderless Breadboards

Insert wires into holes to make a connection.
*Much* easier, quicker than soldering
But, they wear out, are expensive ($8 for this little one)
Using Solderless Breadboards

Using needle nose pliers can help

Grab wire or lead toward end and push into hole
Blinky LED circuit

“hello world” of microcontrollers

In schematics signals often flow from top–left to bottom–right
Common nodes like “gnd” are given their own symbol
Pick any digital pin to hook up to, doesn’t matter which
Blinky LED circuit

- Plug shield on top of Arduino board
- Stick breadboard to shield
Blinky LED Software

You’ve already seen it.

```c
int ledPin = 13; // LED connected to digital pin 13
void setup()
{
    pinMode(ledPin, OUTPUT); // sets the digital pin as output
}

void loop()
{
    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
}
```
Arduino Sketch Structure

- Declare variables at top
- Initialize
  - `setup()` – run once at beginning, set pins
- Running
  - `loop()` – run repeatedly, after `setup()`

Pins can be changed in `loop()` too, but conceptually easier in `setup()`
Arduino “Language”

- Language is standard C (but made easy)
- Lots of useful 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
  - `millis()` – get the current time
- And many others. And libraries. And examples!

Also: serial library, LCD library, servo examples
Development Cycle

- Make as many changes as you want
- Not like most web programming: edit $\rightarrow$ run
- Edit $\rightarrow$ compile $\rightarrow$ upload $\rightarrow$ run
More Blinky Madness

Add LEDs

```cpp
int ledAPin = 7;
int ledBPIn = 6;
int ledCPin = 5;

void setup()
{
    pinMode(ledAPin, OUTPUT);
    pinMode(ledBPIn, OUTPUT);
    pinMode(ledCPin, OUTPUT);
}

void loop()
{
    digitalWrite(ledAPin, HIGH);  // sets the LED on
    digitalWrite(ledBPIn, LOW);   // sets the LED off
    delay(1000);                   // waits for a second
    digitalWrite(ledAPin, LOW);   // sets the LED off
    digitalWrite(ledBPIn, HIGH);  // sets the LED on
    delay(1000);                   // waits for a second
    digitalWrite(ledCPin, HIGH);  // sets the LED on
    delay(100);                    // waits for a second
    digitalWrite(ledCPin, LOW);   // sets the LED off
    delay(100);                    // waits for a second
    digitalWrite(ledCPin, HIGH);  // sets the LED on
    delay(100);                    // waits for a second
}```
Next Week

- Reading buttons
- Reading analog values (knobs)
- Detecting the dark
- More complex LED circuits
- Stand-alone Arduino
END Class 1

Tod E. Kurt

tod@todbot.com
ATmega8 & Arduino

Arduino Pin Mapping

digital pin 0 (RX)
digital pin 1 (TX)
digital pin 2
digital pin 3
digital pin 4
digital pin 5
digital pin 6
digital pin 7
digital pin 8

(RESET) PC6 □ 1
(RXD) PD0 □ 2
(TXD) PD1 □ 3
(INT0) PD2 □ 4
(INT1) PD3 □ 5
(XCK/T0) PD4 □ 6
VCC □ 7
GND □ 8
(XTAL1/TOSC1) PB6 □ 9
(XTAL2/TOSC2) PB7 □ 10
(T1) PD5 □ 11
(AIN0) PD6 □ 12
(AIN1) PD7 □ 13
(ICP1) PB0 □ 14

28 □ PC5 (ADC5/SCL)
27 □ PC4 (ADC4/SDA)
26 □ PC3 (ADC3)
25 □ PC2 (ADC2)
24 □ PC1 (ADC1)
23 □ PC0 (ADC0)
22 □ GND
21 □ AREF
20 □ AVCC
19 □ PB5 (SCK)
18 □ PB4 (MISO)
17 □ PB3 (MOSI/OC2)
16 □ PB2 (SS/OC1B)
15 □ PB1 (OC1A)

analog input 5
analog input 4
analog input 3
analog input 2
analog input 1
analog input 0
digital pin 13 (LED)
digital pin 12
digital pin 11 (PWM)
digital pin 10 (PWM)
digital pin 9 (PWM)

www.arduino.cc