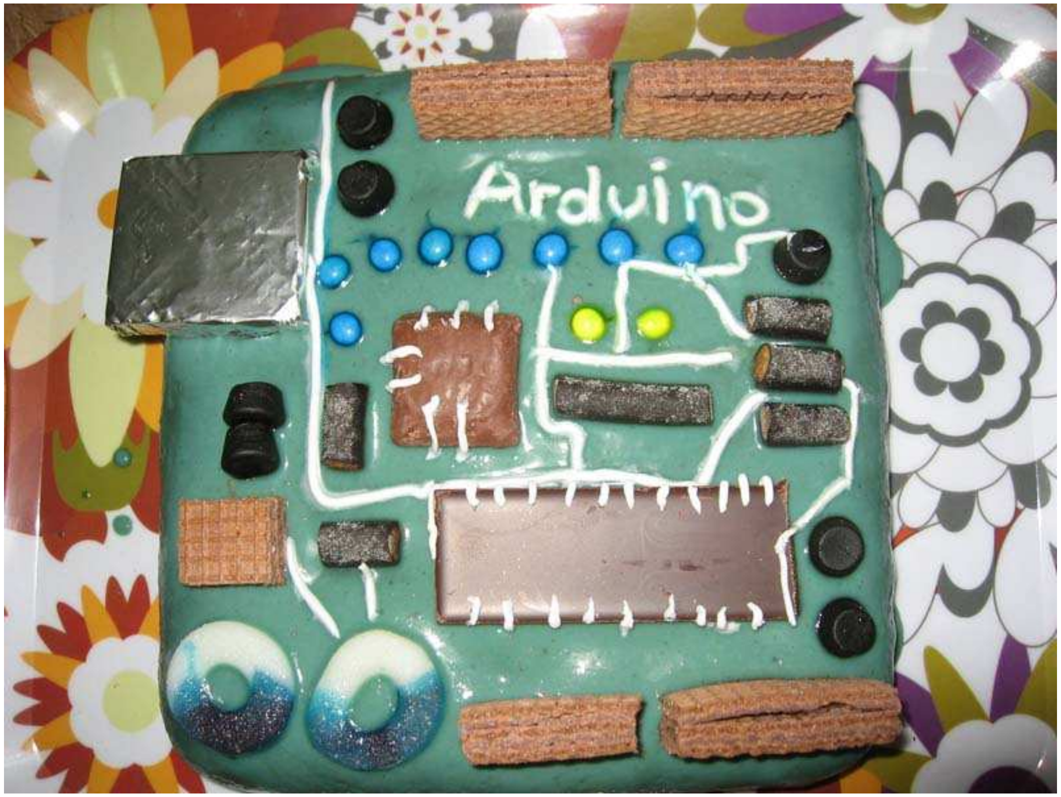
ART2040

Electronics & Physical Computing



Concepts

- Transducers (sensors/actuators)

  Convert between physical/electrical energy

- Input/Output

  Direction of energy flow

- Analog/Digital

  Continuous (i.e. multi-state) vs Binary (i.e. two-state)

- Serial/Parallel

  How events in the flow occur over time

Interaction (break into 3 steps)

- Input (listen)
- Processing (think)
- Output (speak)

Example: movement -> light & sound

Transduction (Transducers)

- Eyes, ears, hands, mouth of physical computing systems
- Input transducers: sensors
- Output transducers: actuators

- Challenge: convert physical energy (light, heat, sound, pressure) to electrical energy and vice versa

Analog & Digital

- Analog has continuous range or multiple states

  i.e. gas pedal of a car

- Digital has only two states: on/off

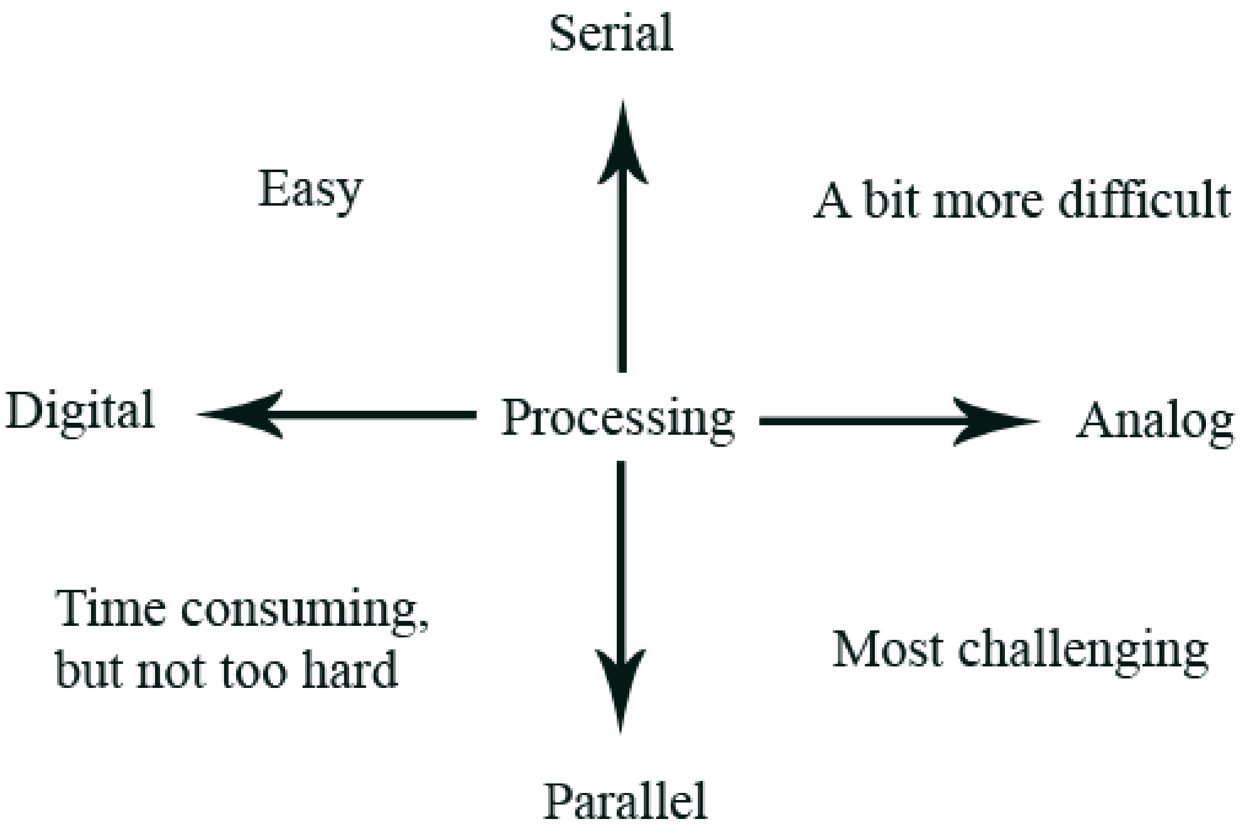
  i.e. a light switch

Parallel & Serial

How are we listening/speaking?

- Serial: events occur one at a time
- Parallel: two or more events occur at the same time

Note: this also applies to electrical flow in circuits



Serial

Easy

A bit more difficult

Digital

Processing

Analog

Time consuming,
but not too hard

Most challenging

Parallel

Getting Started

- First describe *what* happens, not *how* it happens (avoid describing specific technologies – this comes later)
- Break down into input, ouput, and processing
- Identify inputs & outputs as analog or digital

Refer to example: movement -> light & sound

Levels of Abstraction

Food	Software	Circuits
□Highest level: go to restaurant □TV Dinners □Hamburger Helper □Supermarket □Growing own food □Lowest level: “Henry, go kill me a chicken and we’ll have some pot pie tonight”	□MAX,MSP □Puredata □Java □C □ASSEMBLY □MACHINE Language (“100100101110”)	□Buy readymade circuit □Build circuit on breadboard □Build/etch your own PCB □Physically build your own components and circuit

The Tools

- **Circuits:** glue between sensors\actuators & computers
- **Computers:**

1. Microcontrollers (Arduino) - receive info from sensors, control other devices (i.e. motors), communicate with PCs
2. Desktop Computers (i.e. PCs) – run programs like PD

- **Programming:**

1. Puredata
2. Arduino Programming Environment

Electricity

- Flow of electrons from a point of greater electrical energy to a point of lesser electrical energy
- Always follows path of least resistance
- A circuit is a closed loop containing an electrical source (i.e. battery) and a load (i.e. light bulb)

Current/Voltage/Resistance

- Voltage (Volt) – is the difference of electrical energy between two points
- Current (Amps) – is the amount of electrical energy passing through any given point
- Resistance (Ohms) – is the amount that a component resists current flow

Ohm’s Law

- Relationship between voltage, current, resistance

Voltage = Current x Resistance

Likewise: Current = Voltage/Resistance
Resistance = Voltage/Current

Electrical Power (Wattage)

- Determines how much “work” a circuit can do (i.e. turning a motor to lift a weight takes more power than turning on a small light)
- Power is measured in Watts
- Power (Watts) = Voltage (Volts) x Current (Amps)

AC/DC

Two ways to supply electrical power:

1. Direct Current (DC) has constant voltage/current. Batteries supply DC.
2. Alternating Current (AC) alternates voltage in a wave pattern (usually sine).

- Most electronics operating using DC. Electricity supplied to homes is AC therefore we need AC-to-DC converters (transformers) or use batteries.

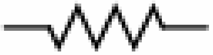
Circuits

- Current flows from positive terminal to negative terminal (ground). Along the way you insert various components (i.e. resistors, capacitors, diodes) to divert the electrons to do your bidding

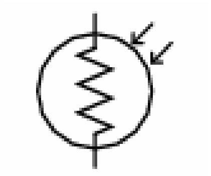
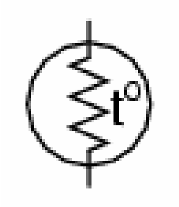
Circuits

- Remember: electricity always favors path of least resistance (avoid short circuits!)
- Also: all the electrical energy (voltage) supplied must be used up in the circuit. If too much energy is supplied components heat up and break!

Resistors

- Give electricity something to do: convert electrical energy to heat (preventing short circuits)
- No polarity (no positive/negative side)
- Measured in ohms, indicated how much resistance they offer
- Schematic symbol: 

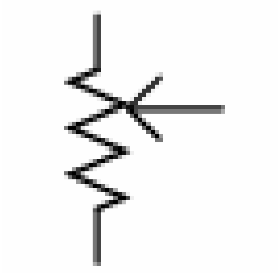
Variable Resistors

- Commonly used as sensors for analog input.
- Photocells:  resistance changes with change in light levels
- Thermistor: 

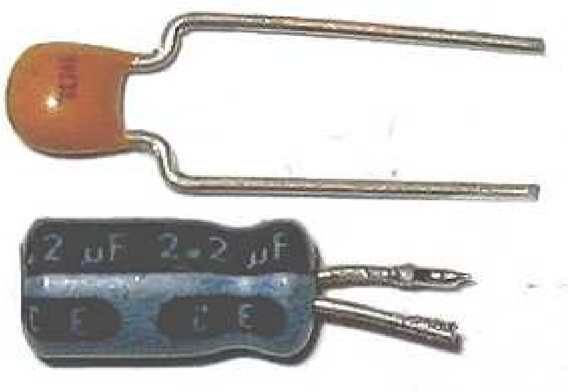
resistance changes with change in heat

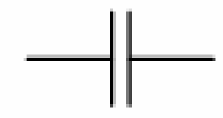
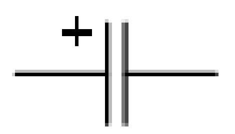
Potentiometer aka Pot (another variable Resistor)

- Changes resistance by turning knob (commonly used as volume knobs)
- Orientation is important: they have three pins
- Schematic symbol:



Capacitors

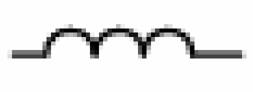


- Store electrical energy (charge)
- Measured in farads (F). Most capacitors are in the range of microfarads (mF or µF), picofarads (pF), nanofarads (nF).
- Some are unpolarized: 
- Some are polarized: 

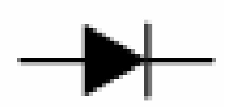
  + goes towards higher voltage

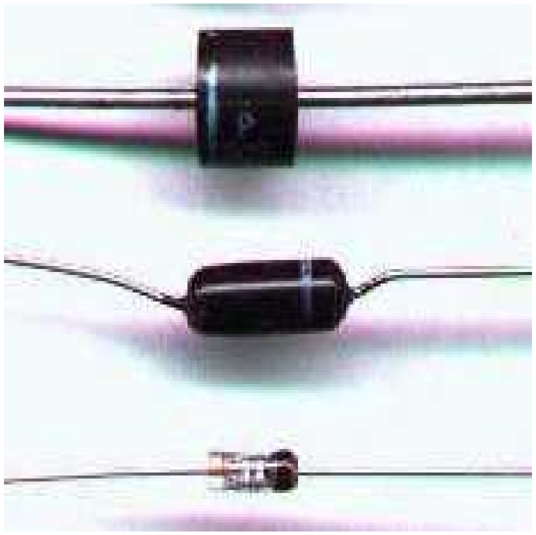
- Capacitors are good for smoothing out erratic electrical flow: they release charge when current dips and store excess charge when the current spikes.

Inductors

- Coil of wire wrapped around a core (air, iron, other magnetic metals)
- Property of inductance: electrical current produces a magnetic field
- Analogous to capacitors: an inductor can be considered as an inverse of a capacitor
- Measured in Henries (H), 

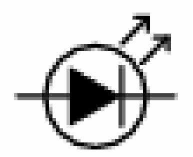
Diodes

- It’s like a one way street: only allows current to flow in one direction
- This means they are polarized (orientation matters)
- Schematic Symbol: 



LED (Light Emitting Diode)

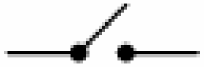
- Low power 'light bulb'

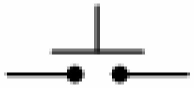


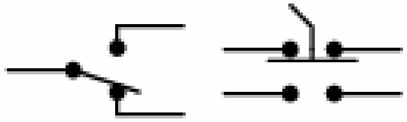
- Usually rated at or below 5V and 20milliamps (0.02A)
- Infrared (IR) LEDs emit infrared light (TV remotes). Good for simple wireless communication

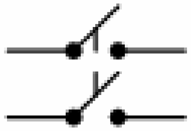
Switches

- Simple concept, but many varieties
- Normally open (N.O.) vs normally closed (N.C)
- Momentary (pushbuttons) vs toggle (light switch)
- Magnetic switches (reed switches)

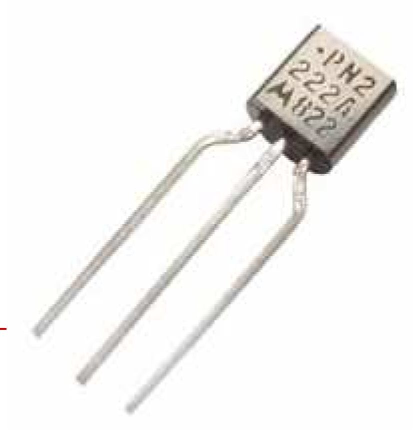

SPST toggle
normally open


Pushbutton
normally open

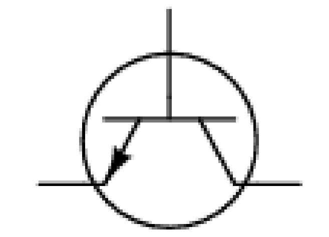

SPDT toggle


DPST toggle

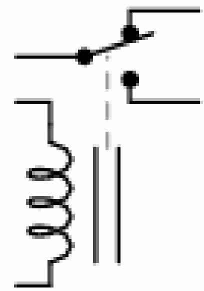
Transistors and Relays



- Switching devices controlled by electronic signals (as opposed to physical)
- Relays work by passing small current through a coil to activate a switch that allows large current to pass
- Transistors are a bit more complicated

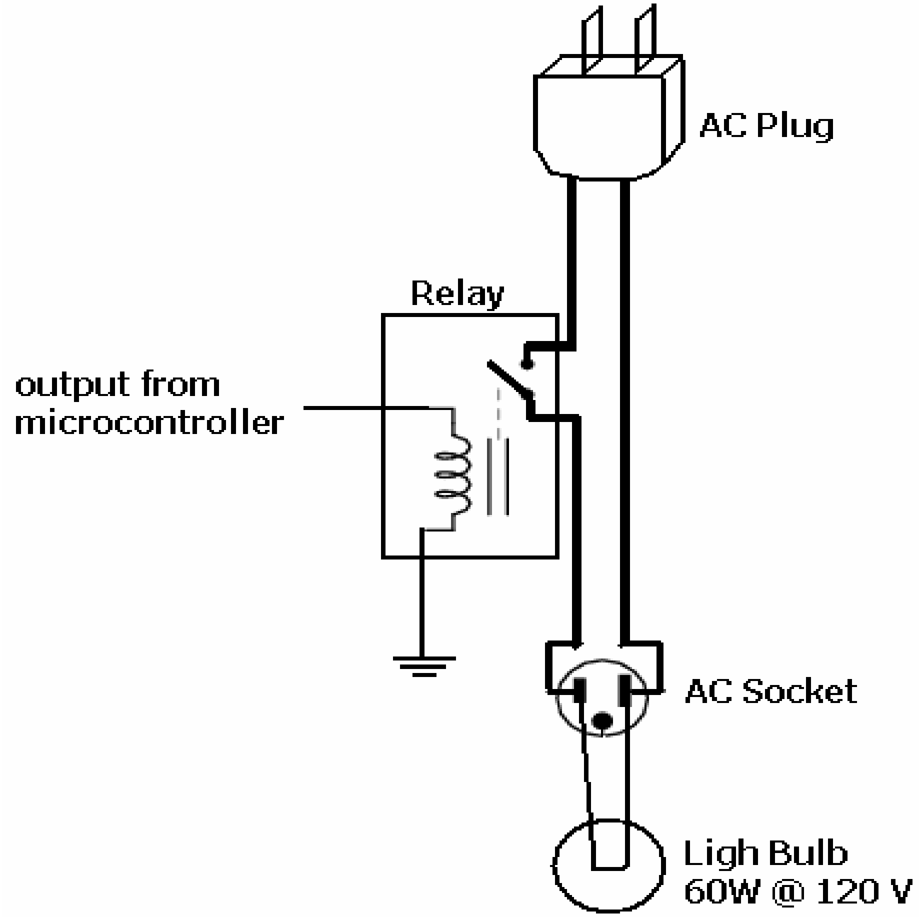


NPN Transistor



Relay

example


AC Plug
Relay
output from microcontroller
AC Socket
Ligh Bulb
60W @ 120 V

Motors: RC Servo

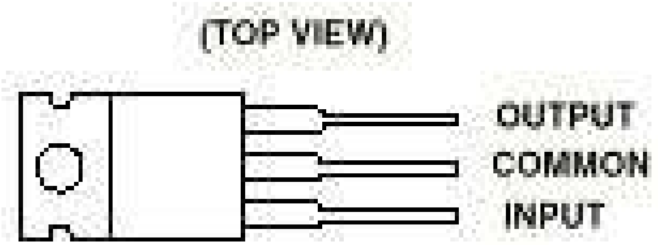
- Unlike traditional motors
- Built in components let user control exact position of the motor
- Use Pulse Width Modulation (PWM) signal to control the motor (PWM is obtained from microcontroller)
- Easy to control (compared to other motors)

Solenoids

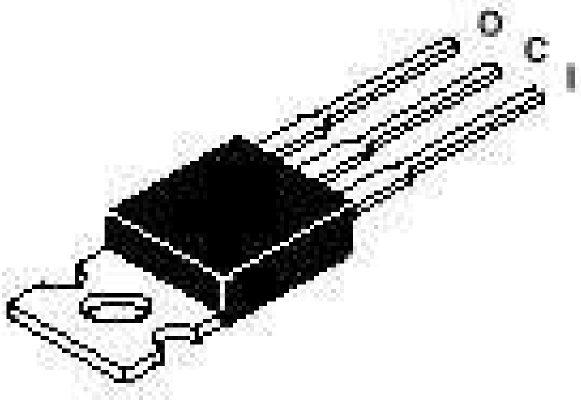
- Device that provides linear motion
- Coil of wire with iron shaft
- Uses property of inductance to generate magnetic field and shaft is pulled or pushed as a result
- Not very difficult to control (usually requires relay/transistor and diode)

Voltage Regulators

- convert a range of voltages (8-15V DC) to a fixed voltage (5V DC)
- Common regulator is the 7805 which converts to 5V DC rated for 1Amp



The common terminal is in electrical contact with the mounting base.



Other components/Tools

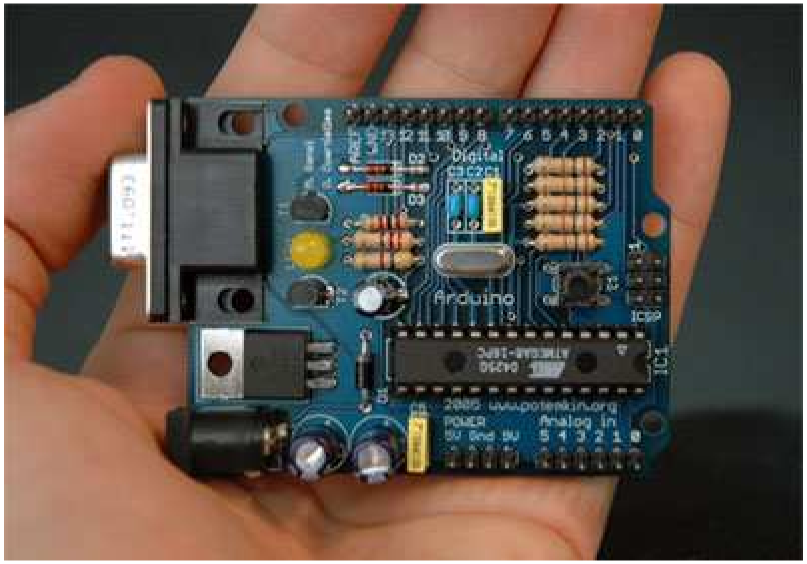
- Connectors: male, female (note: connectors emmiting electrical energy should be female)

- Multimeter:

Measures voltage, current, resistance, sometimes even capacitance

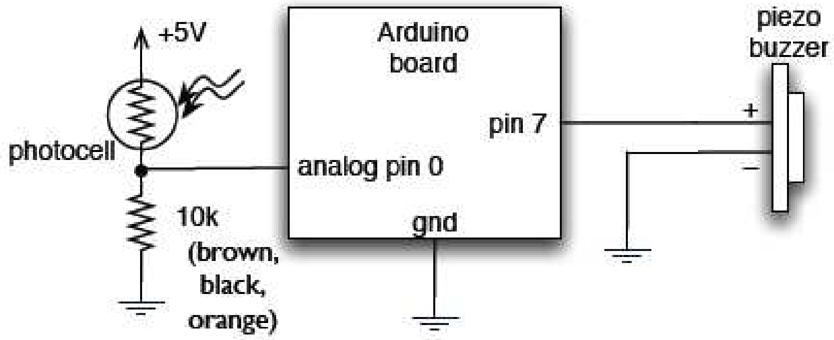
Microcontrollers (Arduino board)

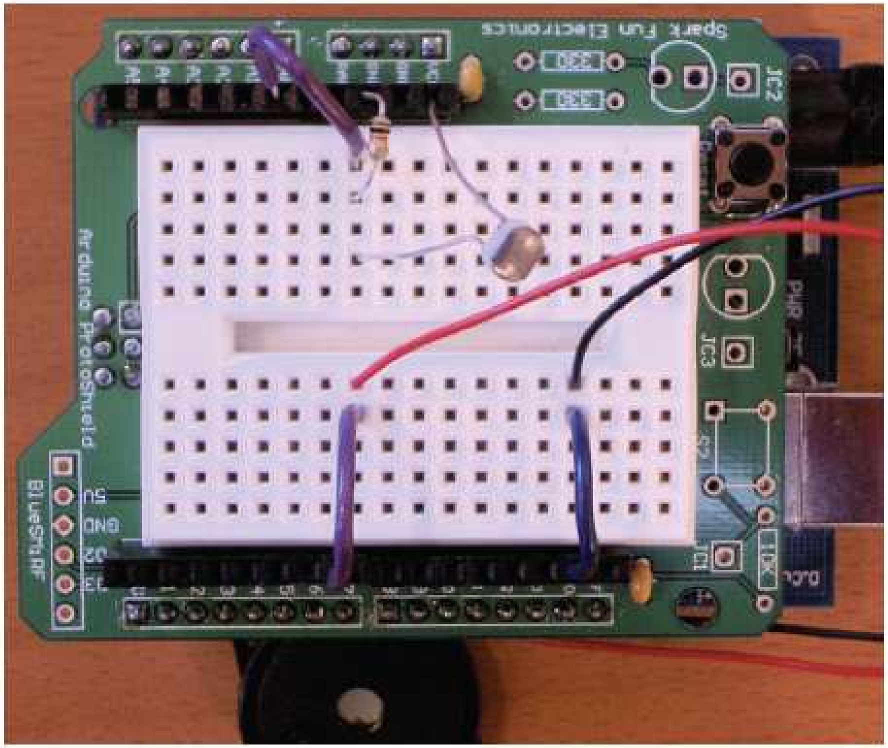
- www.arduino.cc



Building a circuit:
Working from a Schematic

- □ Lets make a Theremin



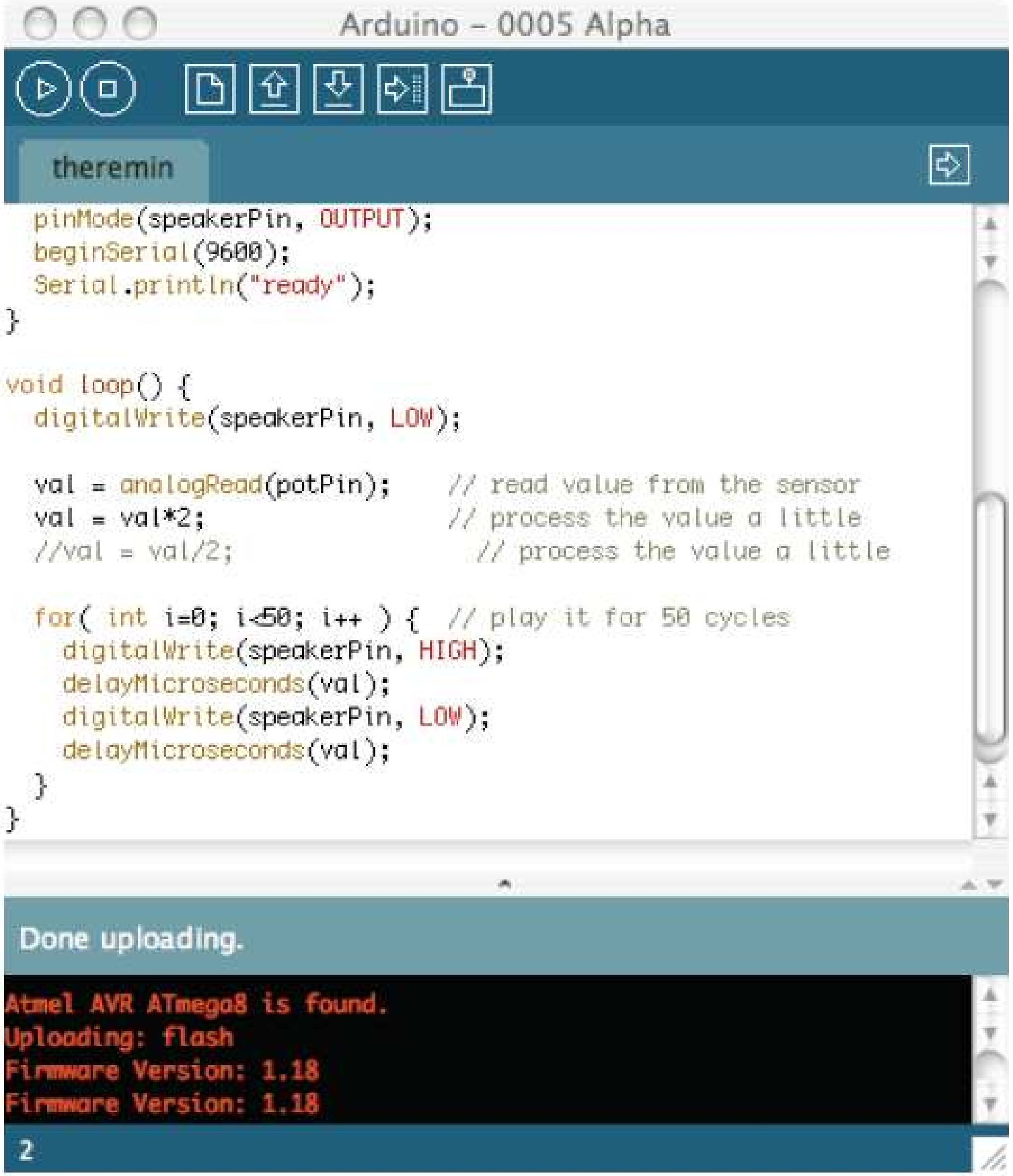


“theremin”

Move hand over photocell to change pitch

Play with val processing & cycles count
to alter sensitivity, pitch and timbre

This is *frequency modulation,*
since you’re changing the frequency



```
  pinMode(speakerPin, OUTPUT);
  beginSerial(9600);
  Serial.println("ready");
}

void loop() {
  digitalWrite(speakerPin, LOW);

  val = analogRead(potPin);    // read value from the sensor
  val = val*2;                 // process the value a little
  //val = val/2;                 // process the value a little

  for( int i=0; i<50; i++ ) {  // play it for 50 cycles
    digitalWrite(speakerPin, HIGH);
    delayMicroseconds(val);
    digitalWrite(speakerPin, LOW);
    delayMicroseconds(val);
  }
}
```

```
Done uploading.
```

```
Atmel AVR ATmega8 is found.
Uploading: flash
Firmware Version: 1.18
Firmware Version: 1.18
```

Resources

- *Physical Computing* by Dan O’Sullivan and Tom Igoe
- http://electronics.howstuffworks.com