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.

Transduction
The conversion of one form of energy into another

Physical energy

Electric signals
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.
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

input transducers
sensors (e.g., switches, levers, sliders, etc.)

output transducers
Actuators (e.g., motors, buzzers)

Illustration adapted and modified from O’Sullivan and Igoe
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.

Illustration adapted and modified from O’Sullivan and Igoe
Transduction

Your job is to find and learn to use transducers to convert between the physical energy appropriate for your project and the electrical energy used by the computer!
# Digital and Analog

<table>
<thead>
<tr>
<th>Digital signal</th>
<th>Analog signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Difference between two possible states</td>
<td>continuous range of possible states</td>
</tr>
<tr>
<td>“whether or not”</td>
<td>“how much” or “stronger” “faster” “brighter”</td>
</tr>
<tr>
<td>Is the cat on the mat or not?</td>
<td>How heavy is the cat that’s on the mat?</td>
</tr>
</tbody>
</table>
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.

From O’Sullivan and Igoe
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

Illustration adapted and modified from O’Sullivan and Igoe
Arduino as an Interface Board

Illustration adapted and modified from O’Sullivan and Igoe
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
Open source, so you can build one yourself!

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.
Arduino Board Overview
Arduino Board Overview

- 16 kBytes of Flash program memory (your program stays in Arduino when powered off)
- 1 kByte of RAM
- 12 MHz (Apple II: 1 MHz) processor speed
Arduino Board Overview

- 13 digital input/output pins
- 5 analog input pins
Arduino Board Overview
Arduino Board Overview

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

www.arduino.cc

Arduino

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 assembled 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 Honorable Mention in the Digital Communities section of the 2006 Ars Electronica Prix. Credits

Arduino News (archive)

2008.09.03 Arduino.cc on the move!! During the next week we will migrate all our online services to a new and better server, we apologize for eventual out times while in the process.

2008.07.11 New distributors: Robotec in Bulgaria and Linux Center in Russia. Full list on the buy page.


2008.04.29 Check out: Mechaboshop, our latest distributor in Japan.

Buy an Arduino Board

Download the Arduino Software

Getting Started

To get started, follow the instructions for your operating
Arduino Software

```cpp
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
}
```

Binary sketch size: 1108 bytes (of a 14386 byte maximum)
Arduino Sketches

```cpp
int ledPin = 13; // LED connected

void setup() // run once, when
{
    pinMode(ledPin, OUTPUT); // sets the digital
}   

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
}
```
Arduino Programming Cycle

```c
// 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.
//

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 Programming Cycle

```
// Blink

// The basic Arduino example. Things 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.


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 Programming Cycle

- **edit**
- **upload**
- **verify**
- **run**

```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
}
```

Done compiling.
Arduino Programming Cycle

1. **edit**
2. **upload**
   - `int ledPin = 13;` // LED connected to digital pin 13
3. **run**
4. **verify**
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/lowlow
- `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)

```cpp
//
* 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()
{
    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
}

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
}
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
}
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

Illustration from O’Sullivan and Igoe
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 [$33.00])
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

```c
/*
 * 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() { 
    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 Thursday**

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!