Using Your Arduino, Breadboard and Multimeter

Work in teams of two!

EAS 199A     Fall 2012
Your Multimeter

pincer clips – good for working with breadboard wiring

(probe these onto probes)

probes

leads

Turn knob to select the type of measurement.

You will use the multimeter to understand and troubleshoot circuits, mostly measuring DC voltage, resistance and DC current.
Power can be provided through the USB cable (+5V from the computer) or externally (7-12V supply recommended)
Measure $V_{in}$

$V_{in}$ is the voltage of the power supply. The USB supplies a nominal 5V (4.43V was measured when this photo was taken).
Change power source and measure $Vin$

*In this photo, a 7V DC power supply was plugged into the power jack of the Arduino.*
Check Voltage at 5V Power Pin

The on-board voltage regulator maintains the voltage on the 5V pin at about 5V.

The measured voltage is close to 5V target.
Check Voltage at 3.3V Pin

The FIDI chip on the Arduino, which helps the microcontroller talk with your computer through the USB cable, also has an on-board voltage regulator that outputs 3.3V.

If you need less than 5V for a project, you can use the 3.3V pin, which provides about 3.3V. The current draw from the 3V3 pin is limited to 50mA.

\[
\text{max power} = V \cdot I = 3.3V \cdot 0.05A = 0.165W = 165mW
\]
Select Resistors

*Find the 330Ω and the 10kΩ resistors from your parts kit.*

<table>
<thead>
<tr>
<th>color</th>
<th>digit</th>
</tr>
</thead>
<tbody>
<tr>
<td>black</td>
<td>0</td>
</tr>
<tr>
<td>brown</td>
<td>1</td>
</tr>
<tr>
<td>red</td>
<td>2</td>
</tr>
<tr>
<td>orange</td>
<td>3</td>
</tr>
<tr>
<td>yellow</td>
<td>4</td>
</tr>
<tr>
<td>green</td>
<td>5</td>
</tr>
<tr>
<td>blue</td>
<td>6</td>
</tr>
<tr>
<td>violet</td>
<td>7</td>
</tr>
<tr>
<td>gray</td>
<td>8</td>
</tr>
<tr>
<td>white</td>
<td>9</td>
</tr>
</tbody>
</table>

Example: 330Ω resistor:

3 = orange
3 = orange
Add 1 zero to 33 to make 330, so 1 = brown

So, 330 = orange, orange, brown

Now, find the 10kΩ resistor.
Check Resistance of Resistors

R ≈ 330 Ω

Switch to Ω

330 Ω
Building a circuit on a breadboard

Rows are *not connected* across the gap

Rows are connected

Columns are not connected

Power rails on both sides

Gap fits the standard DIP package
LED circuit:
Two equivalent pictures
Building an LED Circuit

**Supplies:**
- 2 two jumper wires – colors don’t matter, but red is usually used for positive, and black is used for negative
- LED
- 330 Ω and 10kΩ resistors
- Arduino
- Breadboard
- USB cable from your computer
LEDs

LED = Light Emitting Diode

Electricity can only flow one way through an LED (or any diode). The flat spot on the LED must be connected to ground (GND).
Building an always-on LED Circuit

Short leg of LED connects to ground wire
Breadboard LED circuit

Connections
The Circuit

These circuit diagrams are equivalent.

Symbol for ground (GND)

Black = Gnd
Red = +5V
Replace the 330Ω Resistor with the 10kΩ Resistor

What happens and Why??

**ANSWER:** The smaller resistor (330Ω) provides less resistance to current than the larger resistor (10kΩ). For the same applied voltage, increasing the resistance decreases the current.

Therefore, replacing the 300Ω resistor with the 10kΩ resistor reduces the current and causes the LED to glow less brightly.

**What would happen if you forgot to put in a resistor?** You would probably burn up your LED.
Arduino program to blink an LED

- Build the circuit on the breadboard
  - A slight modification to always-on LED circuit
- Write your first Arduino program
- Use the digital (on/off) output to turn LED on and off
Connect the Power Wire to Pin 2
(Use P2 as a digital output)

Switch power from 5V constant to pin 2

Enter and run the following program:

```c
void setup() {
    // initialize pin as an output:
    pinMode(2, OUTPUT);
}

void loop() {
    // turn the LED on
    digitalWrite(2, HIGH);
    // wait 1 second = 1000 ms
    delay(1000);
    // turn the LED off
    digitalWrite(2, LOW);
    // wait for 500 ms
    delay(500);
}
```
How the Program Works

void setup() {
    pinMode(2, OUTPUT); // initialize pin 2 as an output
}

void loop() {
    digitalWrite(2, HIGH); // set pin 2 to HIGH (5V)
    delay(1000); // wait 1000 ms
    digitalWrite(2, LOW); // set pin 2 to LOW (0V)
    delay(500); // wait 500 ms
}

<table>
<thead>
<tr>
<th>Time (ms)</th>
<th>Voltage (V)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0V</td>
</tr>
<tr>
<td>500</td>
<td>5V</td>
</tr>
<tr>
<td>1000</td>
<td>0V</td>
</tr>
</tbody>
</table>

HIGH = 5V and LOW = 0V (Always!!!!)
Now Experiment on Your Own!

(1) Try changing the time to 1.5 seconds on and 1 second off

(2) Connect the resistor to digital pin 5 and change the program to match

(3) Blink out SOS in Morse code (dot-dot-dot-dash-dash-dash-dot-dot-dot-dot)
   a. three short pulses (0.25 seconds each) followed by . . .
   b. three long pulses (0.75 second each) followed by . . .
   c. three short pulses (0.25 seconds each) followed by . . .
   d. a brief pause (1 second)
   e. repeat a through d using an infinite loop

Show your instructor when you have completed exercise (3)
Find each command in the reference section of arduino.cc
*(discuss each command with others at your table)*

```cpp
void setup() {
  // initialize the digital pin as an output:
  pinMode(2, OUTPUT);
}

void loop() {
  digitalWrite(2, HIGH);  // set the LED on
  delay(1000);           // wait for a second
  digitalWrite(2, LOW);  // set the LED off
  delay(500);            // wait for 500 ms
}
```