Introduction to Arduino

Presented by SEAS Computing Facility    November 3, 2018
Link to These Slides

seascf.seas.gwu.edu/workshops
What is an Arduino?

- Single-board microcontroller
- Originated in Italy
- Many variations from different makers
- Cannot be directly connected to a monitor, keyboard, mouse, etc.
- Does not normally have an operating system
Arduino Uno

- Power In
  - Barrel Jack
  - USB

- Power Out
  - (3.3v and 5v)

- Ground
  - Power In
  - Analog In

- Reset

- Digital In/Out
  - PWM (3, 5, 6, 9, 10, 11)

- RX/TX

- ARef

- Ground
Raspberry Pi vs. Arduino

**Raspberry Pi**
- Allows graphical user interface
- Can be directly connected to Internet
- More powerful and more memory
- Can be used with more programming languages

**Arduino**
- Low power consumption
- Can directly read analog inputs
- Requires less hardware (monitor, mouse, etc.) to get started
- No operating system needs to be installed
Top 10 Arduino Projects
Running a Basic Arduino Program

1. Open the Arduino Desktop IDE (install required)
   a. There is also an Arduino Web IDE (account required)
2. Connect the Arduino to the computer using a USB cable
3. Open example sketch from File > Examples > 01.Basics > Blink
4. Select the board type Arduino/Genuino Uno from Tools > Board
5. Select the port with the Arduino from Tools > Port
6. Click the upload button 🔄
7. The built-in LED near pin 13 should start to blink

Programs will stay on the Arduino until overridden by another program.

Programs start whenever the Arduino is powered on or reset.
Running a Basic Arduino Program

Code:

```cpp
// 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); // Each command ends with a semicolon
}

// 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 by making the voltage LOW
    delay(1000); // wait for a second
}
```
Useful Arduino Functions

- `pinMode(pin, mode) - Sets the pin to be INPUT or OUTPUT
- `digitalWrite(pin, value) - Writes HIGH or LOW to the pin`
Analog vs. Digital

- Digital - finite set of discrete values (in this case 0 or 1)
- Analog - continuous range of values
Useful Arduino Functions

- `pinMode(pin, mode)` - Sets the pin to be INPUT or OUTPUT
- `digitalWrite(pin, value)` - Writes HIGH or LOW to the pin
- `analogWrite(pin, value)` - Writes a value between 0 and 255 to the pin
Serial Output

- Serial is a computer communication protocol
- Serial can be used to send data from the Arduino to the computer
- View the serial output using the Serial Monitor
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.begin(9600);  // Configure the serial output to the computer
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  Serial.println("LED on");  // Write to the serial output
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  Serial.println("LED off");  // Write to the serial output
  delay(1000);
}
Useful Arduino Functions

- `pinMode(pin, mode)` - Sets the pin to be INPUT or OUTPUT
- `digitalWrite(pin, value)` - Writes HIGH or LOW to the pin
- `analogWrite(pin, value)` - Writes a value between 0 and 255 to the pin
- `Serial.begin(speed)` - Configures the serial output for the specified speed (9600 is typical)
- `Serial.println(val)` - Prints a value or string to the Serial monitor followed by a new line
Breadboard Setup
Jumper Cables
Connecting to a Breadboard

- Connect a male-to-male jumper cable from a power rail to 5V on the Arduino
- Connect the ground rail to GND
- Connect the rails on each side of the breadboard
Basic LED Circuit
Light Emitting Diode

- A **diode** is a component that only allows flow of current in one direction.
- A **light emitting diode (LED)** emits light when current passes in the correct direction.
Circuit Basics

- Ohm’s Law: $V = IR$
  - $V$: Voltage (volts)
  - $I$: Current (amperes)
  - $R$: Resistance (ohms)

- LEDs have a maximum current

- Ohm’s Law (rewritten): $I = \frac{V}{R}$
  - To keep current ($I$) low, resistance ($R$) must be high enough
Building the LED Circuit

- Two jumper cables
- LED
- 220 Ω resistor
Building the LED Circuit

- Add an LED to the breadboard
- Connect the anode (long end) of the LED to the resistor and then to the power rail
- Connect the cathode (short end) of the LED directly to the ground rail
- The LED should glow!
Building the LED Circuit

- Move the wire from the resistor out of the power rail and attach it to pin 2 on the Arduino
Blinking an LED

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

void loop() {
    digitalWrite(2, HIGH);  // Set pin 2 to high voltage (turn LED on)
    delay(1000);            // Wait a second
    digitalWrite(2, LOW);   // Set pin 2 to low voltage (turn LED off)
    delay(1000);            // Wait a second
}
Modulo Operation

- Operation for the remainder after division of two numbers
- Represented in many programming languages by %
- Example: $9 \div 4 = 2 \quad 9 \mod 4 = 1$
Blinking an LED (Alternative)

**Code:**

```c
int i = 0;  // Create an integer called i set to 0

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

void loop() {
    digitalWrite(2, i % 2); // Set pin 2 to remainder of i / 2
    delay(1000); // Wait a second
    i++; // Add 1 to i
}
```
Blinking an LED (Alternative)

Code:
```c
const int pinLed = 2;  // Set pinLed to 2; const means this cannot be changed
int i = 0;

void setup() {
    pinMode(pinLed, OUTPUT);
}

void loop() {
    digitalWrite(pinLed, i % 2);
    delay(1000);
    i++;
}
```
Scope

- Variables cannot be used outside of the scope in which they are declared

```c
const int pin_led = 2;

void setup() {
    const int button_led = 3;
    // pin_led can be used here
}

void loop() {
    // pin_led can be used here
    // button_led cannot be used here!
}
```
Button-Activated LEDs
Push Button

- All four pins are **connected** when pressed on.
- If off, the **left** and **right** are separated.
Adding a Button

- Insert the button into your breadboard
- Connect one side of the button to 5 V
- Connect the other side to Arduino pin 3
- Also connect that side to a resistor to ground
  - When the button is not pressed, the Arduino input will default to off
const int pin_led = 2;    const int pin_button = 3;    int buttonState = 0;

void setup() {
    // Set pin 2 (LED) as output
    // Set pin 3 (button) as input
}

void loop() {
    buttonState = digitalRead(pin_button);    // Read whether the button is pressed
    if (buttonState == HIGH) {
        // If the button is on,
        // Turn the LED on
    } else {
        // Otherwise,
        // Turn the LED off
    }
}
```cpp
const int pin_led = 2;  
const int pin_button = 3;  
int buttonState = 0;

void setup() {
  pinMode(pin_led, OUTPUT);  // Set pin 2 (LED) as output
  pinMode(pin_button, INPUT);  // Set pin 3 (button) as input
}

void loop() {
  buttonState = digitalRead(pin_button);  // Read whether the button is pressed
  if (buttonState == HIGH) {
    digitalWrite(pin_led, HIGH);  // If the button is on,
    digitalWrite(pin_led, LOW);  // Otherwise,
  } else {
    digitalWrite(pin_led, LOW);  // Turn the LED on
    digitalWrite(pin_led, LOW);  // Turn the LED off
  }
}
```
Useful Arduino Functions

- `pinMode(pin, mode)` - Sets the pin to be INPUT or OUTPUT
- `digitalWrite(pin, value)` - Writes HIGH or LOW to the pin
- `analogWrite(pin, value)` - Writes a value between 0 and 255 to the pin
- `Serial.begin(speed)` - Configures the serial output for the specified speed (9600 is typical)
- `Serial.println(val)` - Prints a value or string to the Serial monitor followed by a new line
- `digitalRead(pin)` - Returns HIGH or LOW depending on the voltage of the specified pin
- `analogRead(pin)` - Returns a value from 0 to 1023 representing the voltage of the specified pin between 0 and 5 V
Motion Detector
PIR Motion Sensor Detector Module

- **PIR**: Passive Infrared
  - Senses infrared radiation from objects
- Range is adjustable up to 7 meters
- Viewing area is approximately a 120° cone

[Image of PIR sensor module]

PIR Motion Sensor Detector Module

- **Time Delay Adjust**
  - Clockwise increases delay
- **Sensitivity Adjust**
  - Clockwise decreases range
- **Trigger Selection Jumper**
  - See next slide
- **Pins**
  - Power: Should be between 5 and 20 V input
  - Ground: Should be connected to ground
  - Output: Will be 3.3 V if activated, 0 if not

Diagram from [Henry’s Bench](https://henrysbench.com)
PIR Motion Sensor Detector Module

Single Trigger Mode – Time
Delay is started immediately upon detecting motion.
Continued detection is blocked

Repeatable Trigger Mode – Time
Delay is re-started every time motion is detected.

Diagram from Henry’s Bench
Replace Button with Motion Sensor

- Disconnect button, but leave wire from pin 3
- Connect
  - Power to 5V
  - Ground to GND
  - Output to Arduino pin 3
const int pin_led = 2;  
const int pin_motion = 3;  
int motionState = 0;

void setup() {  
  pinMode(pin_led, OUTPUT);  // Set pin 2 (LED) as output
  pinMode(pin_motion, INPUT);  // Set pin 3 (motion) as input
}

void loop() {  
  motionState = digitalRead(pin_motion);  // Read whether motion is detected
  if (motionState == HIGH) {  // If motion is detected,
    digitalWrite(pin_led, HIGH);  // Turn the LED on
  } else {  // Otherwise,
    digitalWrite(pin_led, LOW);  // Turn the LED off
  }
}
Buzzers
Piezo Buzzer

- Piezoelectric buzzers use the piezoelectric effect to convert electrical to mechanical energy
- Connect one pin to ground and the other to a digital output pin
- Active buzzer: generates the sound using internal oscillator
  - Turn on with `digitalWrite(pinNumber, HIGH)`
  - Turn off with `digitalWrite(pinNumber, LOW)`
- Passive buzzer: converts the input signal to sound
  - Turn on with `tone(pinNumber, frequency)`
  - Turn off with `noTone(pinNumber)`
Motion-Activated LED and Buzzer

- Connect the side of the buzzer with the plus to Arduino pin 4
- Connect the other side to ground
Motion-Activated LED and Buzzer

Code:

```c
const int pin_led = 2;
const int pin_motion = 3;
// constant setting pin_buzzer to 4
int motionState = 0;

void setup() {
    pinMode(pin_led, OUTPUT);
    pinMode(pin_motion, INPUT);
    // set pin_buzzer pin mode
}

void loop() {
    motionState = digitalRead(pin_motion);
    if (motionState == HIGH) {
        digitalWrite(pin_led, HIGH);
        tone(pin_buzzer, 5000);
    } else {
        digitalWrite(pin_led, LOW);
        noTone(pin_buzzer);
    }
}
```
const int pin_led = 2;
const int pin_motion = 3;
const int pin_buzzer = 4;

int motionState = 0;

void setup() {
  pinMode(pin_led, OUTPUT);
  pinMode(pin_motion, INPUT);
  pinMode(pin_buzzer, OUTPUT);
}

void loop() {
  motionState = digitalRead(pin_motion);
  if (motionState == HIGH) {
    digitalWrite(pin_led, HIGH);
    tone(pin_buzzer, 5000);
  } else {
    digitalWrite(pin_led, LOW);
    noTone(pin_buzzer);
  }
}
Motors
Motor

- Motors convert electric current to rotating motion
Motor

- Motor draws more current than digital output can supply
- Transistor can be used as a switch to enable power to motor
- Switch wires on motor to change direction

Code:
```cpp
const int motorPin = 3;

void setup() {
    pinMode(motorPin, OUTPUT);
}

void loop() {
    analogWrite(motorPin, 255); // Run motor at max speed
}
```
Other Sensors

Ultrasonic Sensor

Water Sensor

Sound Sensor
Paper Piano
Paper Piano

- Capacitive touch sensing is a way of human touch sensing through materials such as thin plastic, wood, or any other type of covering
- Piano uses capacitive sense to form a capacitor using the sensor plate (the piano) and your body when you touch the sensor
- Use the CapacitiveSense library to have the Arduino uses a send pin to communicate with receive pins
  - To check if you have this library installed, go to Tools > Manage Libraries > and search “CapcitiveSense” if installed then will say installed, otherwise download
Paper Piano

- Assign keys to input pins (used 3, 4, 5, for first set, 8, 9, 10 for second set)
- Will be using a 1M ohm resistor per “key”
- Use cable in pin 2 to carry power to resistors
- Stagger resistors and place cable in line with one resistor and connect to paper clip on piano
Paper Piano

- Add speaker/buzzer by connecting to ground and pin 11
- Then use additional male to female cables to attach to speaker/buzzer
  - Attach “longer” cable to pin 11 (power), and attach “shorter” cable to ground
- Go to go.gwu.edu/5b4 for the first part of the code
More Resources
Ultrasonic Sensor

Measure distance using sound

https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/
Sound Sensor

Measure when sound exceeds a set level

Other Kit Components
Useful Arduino Functions

- `pinMode(pin, mode)` - Sets the pin to be INPUT or OUTPUT
- `digitalRead(pin)` - Returns HIGH or LOW depending on the voltage of the specified pin
- `analogRead(pin)` - Returns a value from 0 to 1023 representing the voltage of the specified pin between 0 and 5 V
- `analogWrite(pin, value)` - Writes a value between 0 and 255 to the pin
- `digitalWrite(pin, value)` - Writes HIGH or LOW to the pin
- `Serial.begin(speed)` - Configures the serial output for the specified speed (9600 is typical)
- `Serial.println(val)` - Prints a value or string to the Serial monitor followed by a new line
Link to These Slides

seascf.seas.gwu.edu/workshops

Survey: go.gwu.edu/fall2018survey