An Introduction to Arduino
a low cost digital prototyping platform

Aileen Drohan, David Kirwan, and Martin Walshe

South East Makerspace,
Old Printworks, Thomas Hill,
Waterford City, X91 TW63
info@southeastmakerspace.org
https://www.southeastmakerspace.org

Abstract. What is an Arduino? Some information about what this workshop hopes to achieve etc. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean imperdiet congue nisi, eu ultrices elit sollicitudin nec. Vestibulum vitae pretium enim. Mauris eu nulla lectus. Donec eu arcu sem. Pellentesque quis metus eget quam efficitur fringilla tristique vitae neque. Nulla at erat ac felis mollis vulputate id vitae justo. Donec ac rutrum neque. Pellentesque at mollis arcu, quis blandit orci. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

Keywords: arduino, digital electronics, prototyping, introduction

Table of Contents

Introduction

"Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike." [Arduino, 2015a]

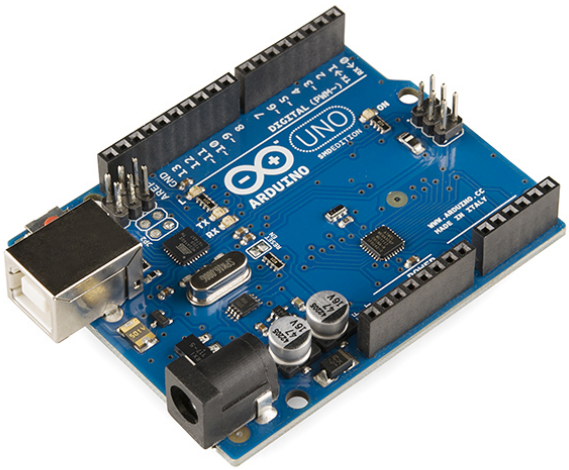

Fig. 1: Arduino Uno R3 [Wikipedia, 2013]

Where might an Arduino be used?

A few contrived examples of where one might use an Arduino in order to automate some process:

Automatic Dog's Water Bowl

An dog owner wants to ensure her pet is never left without water. She attaches a system for measuring the water level in the dog's bowl. Her Arduino is programmed to measure this value every 5 minutes. If this level falls below a certain value, a valve is opened and a water pump is activated to fill up the water bowl. It then sends an SMS to the owner to let her know that the dog is in safe hands.

- Read inputs from a water level sensor
- Control a valve which lets water flow
- Control speed of a water pump
- Send SMS to owner about giving the dog water

Bird Table Camera

A rising social media ornithologist wishes to share pictures from all the visitors to the bird table in his garden. He mounts an infra-red movement sensor on the bird table attached to an Arduino which is configured to record an image and send it to Twitter. His neighbours marvel at how many crows he's feeding.

- Read inputs from a movement sensor
- Control a camera shutter
- Transmit image back to PC
- Send tweet with picture of the bird table visitor

Fingerprint Door Lock

A student is sick of forgetting his keys and being locked out of his house. He uses a fingerprint scanner and an Arduino to make a biometric fingerprint door lock. He needs only scan his thumb print now and the door will unlock.

- Read inputs from a fingerprint sensor
- Compares the finger print against an authorised fingerprint
- Records the time and date a finger was pressed on the scanner
- Makes audio error tone if the fingerprint was invalid
- If valid fingerprint it unlocks the door

Workshop Aims

In this workshop the aim is to give you a crash course in digital electronics, and providing you the basic skills to start using the Arduino micro-controllers in your future projects.

Workshop Requirements

Each person will require the following:

- PC, either Linux, Mac or Windows can be used
- Arduino IDE pre-installed (Internet at the makerspace is flaky!)
- A sambo to keep you going

Learning Outcomes

Each person will leave with:

- Arduino starter kit
- Crash course in digital electronics
- Confidence to use Arduino in future projects

Arduino Starter Kit Contents

The Arduino starter kit contains the following components, which we will be making use of during the workshop.

- 1 × Arduino Compatible R3 Uno
- 1 × Breadboard
- 16 × jumper wires various colours
- 20 × 5mm LED's assorted colours
- 10 × 10k ohm resistors
- 10 × 330ohm resistors
- 1 × RGB LED
- 1 × photo resistor
- 2 × push buttons
- 1 × temperature sensor

Basic Circuit Theory

In an electrical circuit there is a fundamental relationship between voltage, current and resistance and it is explained by Ohms Law [ElectronicsTutorials, 2015].

Voltage

Voltage, (SI Unit: V - Volts) is the potential energy of an electrical supply stored in the form of an electrical charge. Voltage can be thought of as the force that pushes electrons through a conductor and the greater the voltage the greater is its ability to push the electrons through a given circuit [ElectronicsTutorials, 2015].

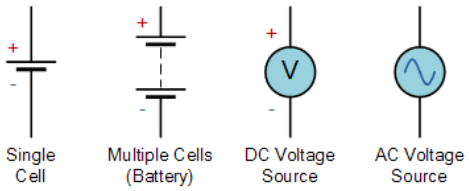

Fig. 2: Voltage Symbols [ElectronicsTutorials, 2015]

Current

Current, (SI Unit: A - Ampere) is the movement or flow of electrical charge and is measured in Amperes. It is the continuous and uniform flow (called a drift) of electrons (the negative particles of an atom) around a circuit that are being pushed by the voltage source [ElectronicsTutorials, 2015].

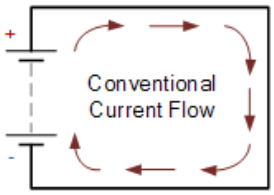

Fig. 3: Current Symbols [ElectronicsTutorials, 2015]

Resistance

Resistance, (SI Unit: Ω - Ohms) of a circuit is its ability to resist or prevent the flow of current (electron flow) through itself making it necessary to apply a greater voltage to the electrical circuit to cause the current to flow again. Note that Resistance cannot be negative in value only positive [ElectronicsTutorials, 2015].

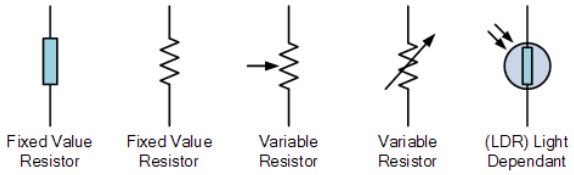

Fig. 4: Resistance Symbols [ElectronicsTutorials, 2015]

Ohm's Law

The following equation Ohm's Law explains the relationship between Voltage, Current and Resistance for an electrical circuit.

$$V = I \times R \tag{1}$$

$$I = \frac{V}{R} \tag{2}$$

$$R = \frac{V}{I} \tag{3}$$

Fig. 5: Ohm's Law

Analogue vs Digital Signals

The world we live in is an analogue one. There are an infinite number of possible combinations of colours smells and sounds. The digital world however is not infinite. It is discrete and finite where we are limited by several factors, such as memory and computational capabilities. In order to represent a real world thing digitally, it is by necessity that some information is lost.

Analogue Signal Examples

The following are some examples of analogue signals:

- Temperature
- Sound
- EM Radiation

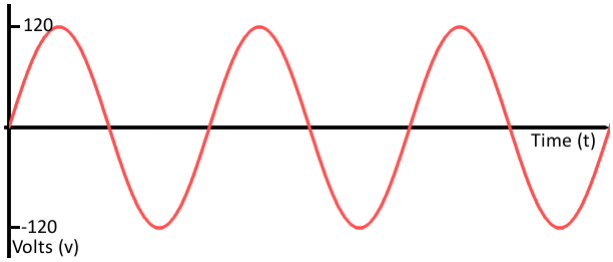

Fig. 6: Analogue Signal [Lindblom, 2015]

Digital Signal Examples

The following are some examples of digital signals:

- Morse Code
- WIFI
- Binary

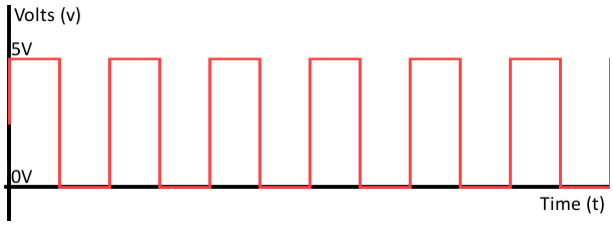

Fig. 7: Digital Signal [Lindblom, 2015]

Basic Arduino Coding Concepts

Variables

A variable is a way of naming and storing a value for later use by the program, such as data from a sensor or an intermediate value used in a calculation [Arduino, 2015b]. There are other more advanced data types such as Arrays and

Type	Example
char	'a'
string	'Hello there'
byte	0 to 255
int	-32,768 to 32,767
unsigned int	0 to 65,535
long	-2,147,483,648 to 2,147,483,647
unsigned long	0 to 4,294,967,295
float	-3.4028235E38 to 3.4028235E38

Table 1: Variable Types in Wiring

Pointers, which we may use, but won't explain them in this introduction. More information can be found regarding these data types online. See the Arduino online reference library at: https://www.arduino.cc/en/Reference/HomePage.

setup() function

The setup() function is called when a sketch starts. Use it to initialize variables, pin modes, start using libraries, etc. The setup function will only run once, after each powerup or reset of the Arduino board [Arduino, 2015c].

```
// setup initializes serial and the button pin
void setup()
{
  Serial.begin(9600);
}

// write to serial, then wait 2 seconds
void loop(){
  Serial.write("Hello World");
  delay(2000);
}
```

loop() function

After creating a setup() function, which initializes and sets the initial values, the loop() function does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the Arduino board [Arduino, 2015d].

```
// Create int variable to represent
// a physical pin on the Arduino
int buttonPin = 3;

// setup initializes serial and the button pin
void setup()
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}

// loop checks the button pin each time,
// and will send serial if it is pressed
void loop()
{
  if (digitalRead(buttonPin) == HIGH)
    Serial.write('H');
  else
    Serial.write('L');

  delay(1000);
}
```

Experiment 1 - Blink

This example is what one might call the *hello world* example for Arduino sketches. We wire up the experiment as shown in the diagram fig: 8. And upload the sketch code in the next section on page: 12.

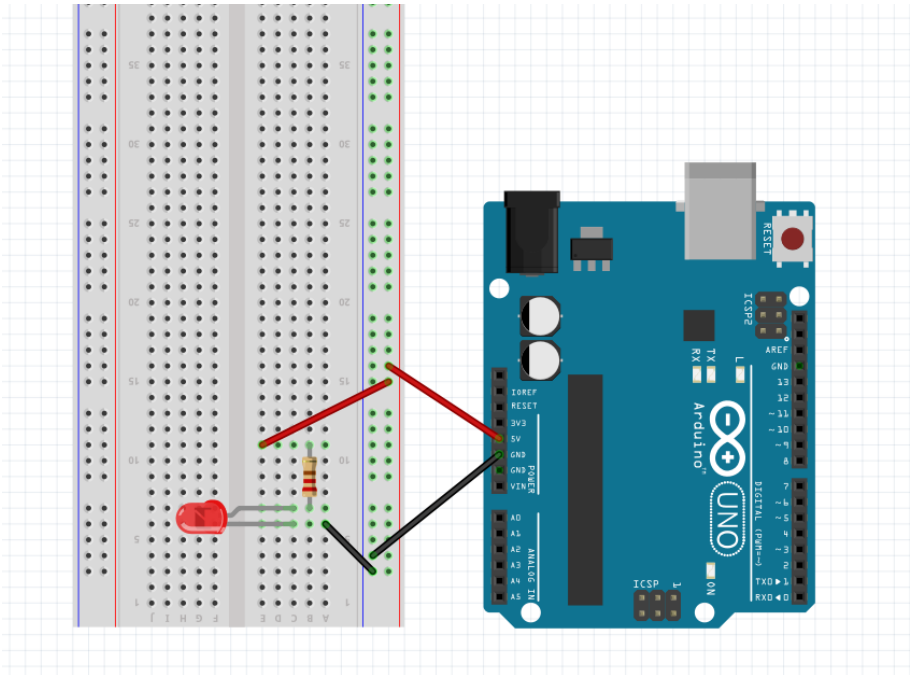

Fig. 8: LED Blink [Fritzing, 2015]

When the Arduino boots, the led should flash on for a second, then off for a second and repeat.

Sketch Code

```
/*
Blink
Turns on an LED on for one second, then off for one second, repeatedly.

This example code is in the public domain.
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  // turn the LED on
  // (HIGH is the voltage level)
  digitalWrite(13, HIGH);

  //wait for 1000 milliseconds
  delay(1000);

  // turn the LED off by making
  // the voltage LOW
  digitalWrite(13, LOW);

  // wait for 1000 milliseconds
  delay(1000);
}
```

Experiment 2 - Button

We wire up the experiment as shown in the diagram fig: 9. And upload the sketch code in the next section on page: 14.

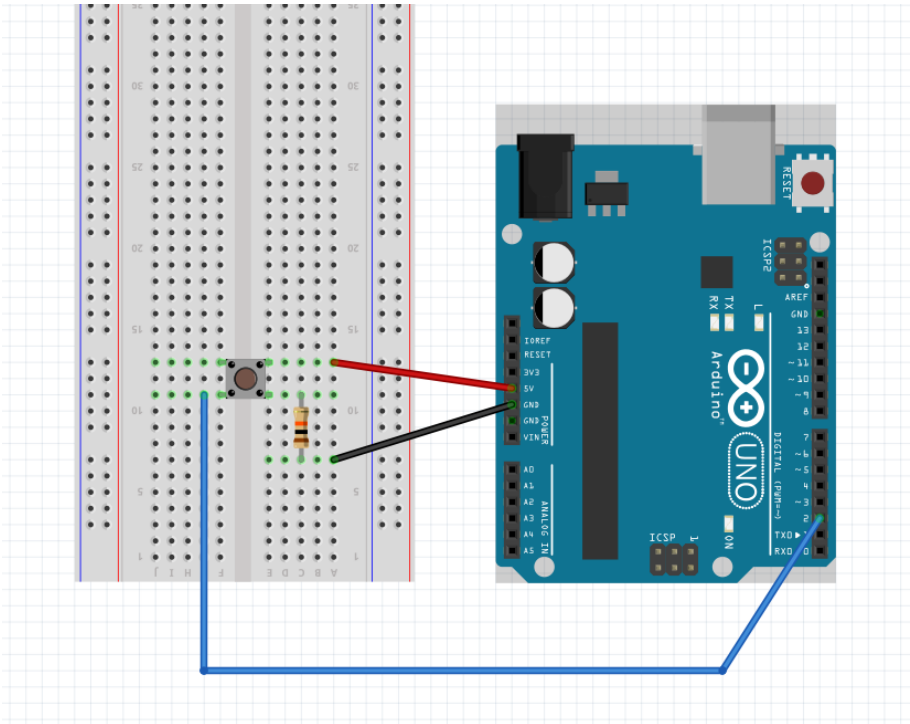

Fig. 9: Read a buttons input [Fritzing, 2015]

Open the *serial monitor* inside the Arduino IDE, every second a 0 should be printed in the console window. When the button is pressed, a 1 will be printed instead.

Sketch Code

```
/*
DigitalReadSerial
Reads a digital input on pin 2, prints the result to the serial monitor

This example code is in the public domain.
*/

// digital pin 2 has a pushbutton attached to it. Give it a name:
int pushButton = 2;
int buttonState;

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input pin:
  buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  // delay in between reads for stability
  delay(100);
}
```

Experiment 3 - Generating Music

We wire up the experiment as shown in the diagram fig: 10. And upload the sketch code in the next section on page: 16.

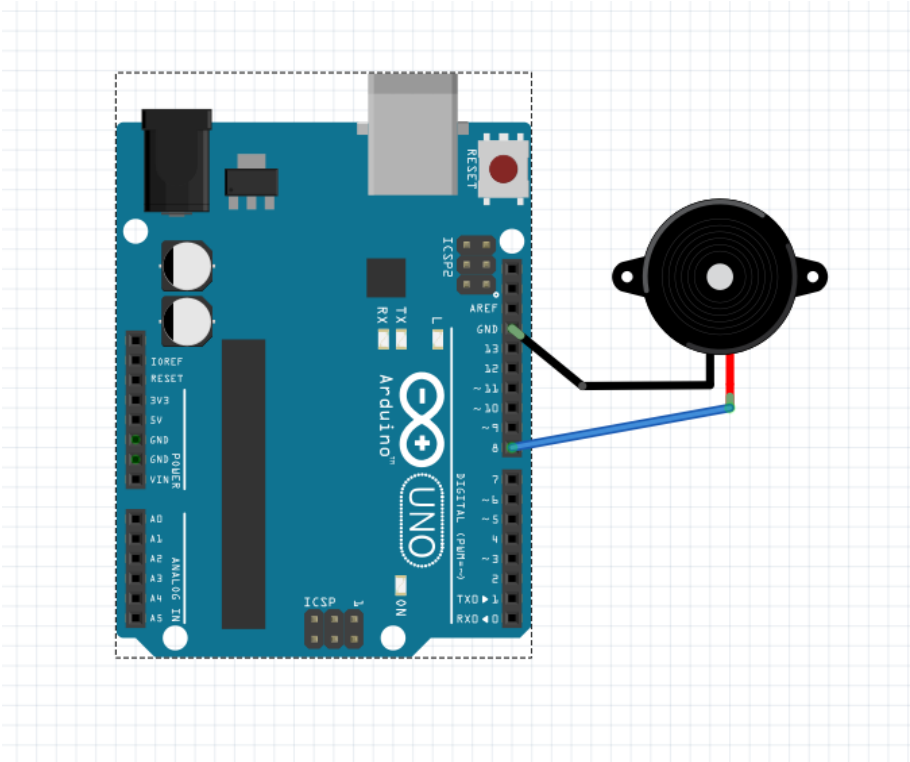

Fig. 10: Play musical notes through a speaker [Fritzing, 2015]

As soon as the Arduino boots it should play a series of musical notes and then stop. For more information see the online reference page for the functions used in this experiment here: https://www.arduino.cc/en/Reference/Tone

Sketch Code

```
/*
Melody - Plays a melody
This example code is in the public domain.
*/
#include "pitches.h"

int melody[] = { // notes in the melody:
  NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4
};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
  4, 8, 8, 4, 4, 4, 4, 4
};

void setup() {
  // iterate over the notes of the melody:
  for (int thisNote = 0; thisNote < 8; thisNote++) {
    // to calculate the note duration, take one second
    // divided by the note type.
    //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
    int noteDuration = 1000 / noteDurations[thisNote];
    tone(8, melody[thisNote], noteDuration);
    // to distinguish the notes, set a minimum time between them.
    // the note's duration + 30% seems to work well:
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
    noTone(8);
  }
}

void loop() {
  // no need to repeat the melody.
}
```

```
// ********************* pitches.h ************************

/************************************************
* Public Constants
************************************************/

#define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
#define NOTE_A1  55
#define NOTE_AS1 58
#define NOTE_B1  62
#define NOTE_C2  65
#define NOTE_CS2 69
#define NOTE_D2  73
#define NOTE_DS2 78
#define NOTE_E2  82
#define NOTE_F2  87
#define NOTE_FS2 93
#define NOTE_G2  98
#define NOTE_GS2 104
#define NOTE_A2  110
#define NOTE_AS2 117
#define NOTE_B2  123
#define NOTE_C3  131
#define NOTE_CS3 139
#define NOTE_D3  147
#define NOTE_DS3 156
#define NOTE_E3  165
```

```
#define NOTE_F3  175
#define NOTE_FS3 185
#define NOTE_G3  196
#define NOTE_GS3 208
#define NOTE_A3  220
#define NOTE_AS3 233
#define NOTE_B3  247
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_B4  494
#define NOTE_C5  523
#define NOTE_CS5 554
#define NOTE_D5  587
#define NOTE_DS5 622
#define NOTE_E5  659
#define NOTE_F5  698
#define NOTE_FS5 740
#define NOTE_G5  784
#define NOTE_GS5 831
#define NOTE_A5  880
#define NOTE_AS5 932
#define NOTE_B5  988
#define NOTE_C6  1047
#define NOTE_CS6 1109
#define NOTE_D6  1175
#define NOTE_DS6 1245
#define NOTE_E6  1319
```

```
#define NOTE_F6  1397
#define NOTE_FS6 1480
#define NOTE_G6  1568
#define NOTE_GS6 1661
#define NOTE_A6  1760
#define NOTE_AS6 1865
#define NOTE_B6  1976
#define NOTE_C7  2093
#define NOTE_CS7 2217
#define NOTE_D7  2349
#define NOTE_DS7 2489
#define NOTE_E7  2637
#define NOTE_F7  2794
#define NOTE_FS7 2960
#define NOTE_G7  3136
#define NOTE_GS7 3322
#define NOTE_A7  3520
#define NOTE_AS7 3729
#define NOTE_B7  3951
#define NOTE_C8  4186
#define NOTE_CS8 4435
#define NOTE_D8  4699
#define NOTE_DS8 4978
```

Experiment 4

Outline each experiment in a separate chapter

Experiment 5 - Phenakistoscope Creation

Outline each experiment in a separate chapter
[Kalif, 2013] [Kalif, 2015]

Conclusions

- item 1
- item 2
- item 3
- item 4

Glossary

Arduino an open-source electronics platform based on easy-to-use hardware and software. It's intended for anyone making interactive projects. https://www.arduino.cc. 3, 4

Bibliography

Arduino. What is arduino?, 2015a. URL https://www.arduino.cc/en/Guide/Introduction. Accessed: 2015-10-18.

Arduino. Variables, 2015b. URL https://www.arduino.cc/en/Reference/VariableDeclaration. Accessed: 2015-10-18.

Arduino. setup(), 2015c. URL https://www.arduino.cc/en/Reference/Setup. Accessed: 2015-10-18.

Arduino. loop(), 2015d. URL https://www.arduino.cc/en/Reference/Loop. Accessed: 2015-10-18.

ElectronicsTutorials. Relationship between voltage, current and resistance, 2015. URL http://www.electronics-tutorials.ws/dccircuits/dcp_1.html. Accessed: 2015-10-18.

Fritzing. Fritzing - electronics made easy, 2015. URL http://fritzing.org/home/. Accessed: 2015-10-23.

Will Kalif. Make a phenakistoscope, 2013. URL https://www.youtube.com/watch?v=tQVdTXEo2qM. Accessed: 2015-10-18.

Will Kalif. Make a phenakistoscope, 2015. URL http://www.stormthecastle.com/stop-motion-animation/how-to-make-a-phenakistoscope.htm. Accessed: 2015-10-18.

Jim Lindblom. Analog vs. digital, 2015. URL https://learn.sparkfun.com/tutorials/analog-vs-digital. Accessed: 2015-10-18.

Wikipedia. Arduino uno r3, 2013. URL https://commons.wikimedia.org/wiki/File:Arduino_Uno_-_R3.jpg. Accessed: 2015-10-18.