# Arduino

10/13/2019

Cal State Fullerton, CPSC 311 Technical Writing, Professor Peralta

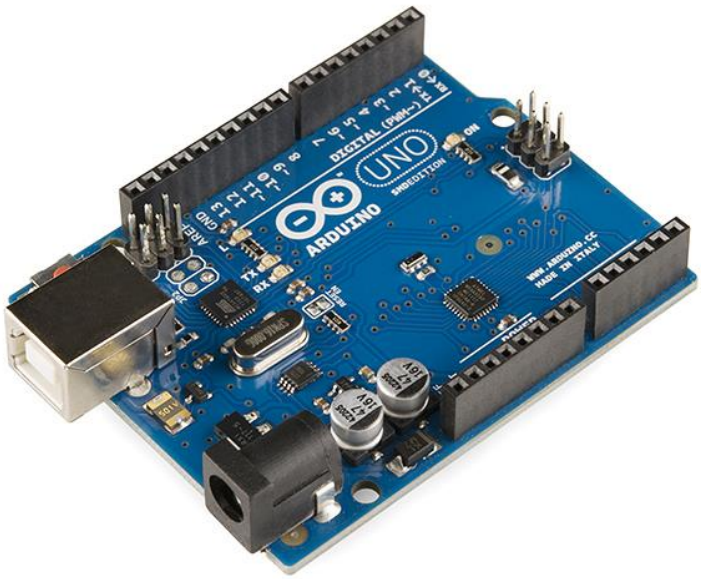
Group Project 2: Research Paper

Kevin Nguyen

Steven Steele

Daniel Banuelos

[1]




[1]Image taken from Wikipedia

Outline

# Abstract

The market offers many microcontrollers and miniature computers. Any business deciding which platform to use for rapid prototyping could be easily overwhelmed with the options. However, for this company's purposes, the Arduino Uno is the best option. There are cheaper microcontrollers, and on the other extreme, more robust SOC's that run desktop operating systems. However, the Arduino Uno has the best trade-offs. With a relatively low cost and robust feature set, it is an excellent option for quick turnaround time for quickly prototyping ideas. This paper will outline the strengths and weaknesses of the Arduino platform, along with technical information on its use.

# What is Arduino

Arduino is an affordable open-source electronics platform available for anyone to use. It comes with hardware and software necessary to program it. It's ideal for anyone who wants to try out using electronics to make it more accessible for them. Generally, an Arduino board consists of a microcontroller using the Arduino programming language, which is C/C++ language, flash memory, pins, and other features.

# Info on Arduino

## Strengths

Arduino has a few advantages it does over other microcontrollers. Some of them being how easy to use and accessible it is to beginners, the library of example code it has, and its community.

## Easy to use

Arduino is an easy to use platform. It is ideal for those who are beginners, hobbyists, or anyone else who wants to do a project. The board comes with its bare necessities for users so they can plug it in through the USB port and start working on it after downloading the IDE. Many people try to use the Arduino because it makes things easier. The simplified version of C++ and the already made Arduino microcontroller(atmega328 microcontroller [1]) that you can program, erase, and reprogram at any given time make the Arduino board easy to use. (Badamasi, 2019)

## Library

Another of its strengths is the abundance of code examples in its software. Compared to other microcontrollers like a ATmega8 microcontroller, where users would have to learn the entire process of running a program, whereas compared to Arduino, all they would need to do is go into the program, click on file, go to the examples, and look through the selection there for the code and add to it if they need to. It has libraries meant for working with the standard electronic components like LCD screens, sensors, motors, and so forth.

## Community

Arduino's community is vast. It contains a lot of libraries, documentation, help, and anything else for a user to explore. There are a lot of projects using Arduino, so the user can look at them to get some ideas. If they need help, they can go around the forums, look through Github or tutorials for the issue they're trying to fix. The community has a significant number of resources available to the user to find what they need.

## Weakness

Likewise, Arduino also has disadvantages compared to other microcontrollers. One is that even though it is easy to use, it does have limitations on itself, causing issues for the user in the future if they want to gain a deeper understanding. The cost being the second issue and the fact that there is no debugger

### Limitations to the user

The learning curve for Arduino is simple. It's easy to learn and get started with all the tools at the user's disposal, and because it has a large community, users can get help from anyway. It's suitable for those who want to do simple projects without learning the inner mechanisms. But for those who wish to gain a better understanding with microcontrollers, Arduino isn't that. Arduino hides away the complicated stuff so that users can have a more beginner-friendly experience, which is different from the usual setting where the user would have to learn from scratch how things work and do something like a simple hello world.

### Cost

Most Arduinos generally range from 10-40 dollars, so the price isn't that bad If the user is only looking to use the Arduino for learning or messing around. If an industry needs to use many boards, then it can be quite costly. There are most likely better alternatives for the industry that's less costly and would work just as well.

### No Debugger

One of the issues that Arduino has is that it doesn't have an internal debugger or an official one in the Arduino IDE. It can cause problems down the line for the user since it'll be much harder for them to debug things. The only way to debug with Arduino is to run it to see what's missing or start small and add to it slowly checking to make sure each code works.

However, the serial port is easy to access and can write to an external device, or send output back to the IDE while connected to a computer. So while there is no debugger, the debugger can be significantly assisted by quickly writing output to the serial peripheral interface and viewing it on a PC.

# Usage of Arduino

## Arduino IDE and Programming Language

The official webpage for Arduino allows the user to download the official IDE for free. The software is open source. The free IDE is very feature-robust and available for multiple platforms.

> The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. (Arduino.cc, 2019)

The IDE allows you to write code targeting the Arduino on whatever computer you choose. The IDE offers the features of any modern IDE, including syntax highlighting, error checking, autocomplete, and auto-formatting. The IDE will cross-compile your program and allow you to quickly transfer the code to the onboard flash memory of any Arduino board via USB. The way Arduino works is different from a Raspberry Pi. The user writes the code as a single primary function (which can still reference outside functions), which is automatically run in a loop on the board itself. It does not run a full operating system like the Raspberry Pi and is ideal for relatively limited programs that lend themselves well to comparatively simple tasks.

There are only 31.5 KB's of flash memory for any uploaded program on the most current revision of the Uno (Store.arduino.cc, 2019).

The IDE uses a language based on a stripped-down version of C++. It includes many of the data structures, algorithms, and features of C++. Also, there is an extensive library available for many of the most commonly used electronics. The IDE allows simple, out-of-the-box support for almost any project, without any need to write custom code to interface with components.

## Included Hardware

An Arduino Uno board comes with useful input/output options built-in. There are 14 general-purpose input/output pins (GPIO), 6 of which can use pulse width modulation (PWM) to approximate analog output. Additionally, there are six analog input pins available. There are also TX and RX pins for serial communication. (Store.arduino.cc, 2019)

These included pins are extremely easy to interact with using the official IDE. There are library functions that make it extraordinarily easy to read analog input (from 0 to 5 volts), read and write the GPIO pins to HIGH or LOW, and send and receive serial communications. By calling simple functions and passing in an integer for the desired pin number, it is fast and easy to read input or write output to the included pins. Unlike some other microcontrollers, it is unnecessary to deal with things like memory mapping to interact with the input/output pins. Just a few simple function calls are all it takes.

## Interfacing with Other Hardware

There are many libraries to make it easy to interface with other components. For example, there is an included library that makes writing to an LCD screen from serial very easy. It is only necessary to set variables beforehand to let the compiler know to which pins the display

connects. The user can handle everything else, and the user does not have to worry about technical details like which ASCII integers to write to the data pins, setting the read/write pin, writing to the enable pin, etc. The included library can even handle things like scrolling for the user.

Many other hardware devices can easily incorporate into a project using the included libraries. For example, there are libraries included for motors (standard motors and step motors), piezoelectric speakers (both passive and active), RF transmitters and receivers, IR sensors, laser diodes, photoresistors, phototransistors, photodiodes, temperature sensors, humidity sensors, soil moisture sensors, and almost anything else you can imagine. Also, because of the vast Arduino support, even if the IDE doesn’t include a library for a component you want to use, chances are excellent that either the company itself or someone in the community has already made one that you can quickly download.

# Compare and Contrast

As stated, Arduino boards are a safe and user-friendly gateway into MCU's. This section will make comparisons to a variety of alternative controllers. We will assume the reader seeks to find alternative MCU's along the same level of usability as Arduino boards. Note that there are various Arduino boards, and each of their boards tends to different needs. The same goes for the competitors; there are many out there and many different variations.

The design of Arduino is open source, enabling a wide variety of "competitors" or alternatives to model themselves after Arduino. There will be some tradeoff between the popular Arduino Boards and the other options that will be listed. My job is to share with you the differences in the "clone" boards. Your job is to know what your needs are and choose accordingly. When working with microcontrollers such as Arduino boards, there are a few

components we must consider when deciding which board we want to use. I will highlight four aspects of the boards that are the most important when choosing the direction you wish to pursue.

## Criteria

The first is power consumption (efficiency), the second is speed, the third is versatility, and the final component is price.

The topic of power consumption may not be relevant to some, but for those who wish to leave their program/microcontroller active for long periods, this will pertain to you. If you intend to run simple programs that will stay on for a while, then you may want to consider something like the MSP430 Launchpad. This microcontroller is an Ultra-Low-Power controller, built around a 16-bit CPU. Most Arduino's will run an ATmega328P processor, which is 8-bits.

If speed is something you are interested in, you are probably running large programs that need/use a lot of memory. Perhaps you need more computational power. If that is the case, then make sure to check out the Teensy 3.6. The Teensy 3.6 has a clock speed of 180MHz as opposed to an average of 20MHz among most Arduino boards. A drawback of this is, of course, the consumption of power.

Using a versatile board is essential, especially when you are beginning and want to explore the fruits MCU's provide. Arduino boards are a good start, but if you wish to try something new, then a good suggestion is the STM32 boards, aka Blue Pill. These boards are like Arduino boards in the sense that they are both versatile and have good specs all around. With the Blue Pill, you're looking at 72MHz, low-power consumption, and 64kb of flash memory. The downside to this alternative is that it is very fragile and has no USB chip on the board.

Now, when discussing price, I'm assuming, the cheaper, the better. Fortunately, Arduino Boards cover this well. Most of their boards are very affordable. Though, if you want something that falls within their range, then the NodeMCU is something to consider.

These are just a few boards that we are comparing. There are tons more out there; you can even create your own!

# Conclusion

In conclusion, the Arduino platform is the correct choice for our business for the reasons outlined above, including low cost, robust toolchain, and an extensive community. The platform is perfect for quickly prototyping ideas and testing them. It is so user-friendly that I was able to create a laser communicator relatively quickly. I only used a few simple components and two Arduino Uno boards. It uses Morse code to transmit a message over laser to a receiver, which decodes the message and displays it. It can be seen working here: https://www.youtube.com/watch?v=WGSFJiVaN_k&feature=youtu.be

Works Cited

Arduino.cc. (2019). *Arduino - Software*. [online] Available at: https://www.arduino.cc/en/Main/Software [Accessed 15 Oct. 2019].

"Arduino Comparison Guide." Standard Arduino Comparison Guide - SparkFun Electronics, https://www.sparkfun.com/standard_arduino_comparison_guide.

Badamasi, Y. (2019). *The working principle of an Arduino - IEEE Conference Publication*. [online] Ieeexplore.ieee.org. Available at: https://ieeexplore.ieee.org/abstract/document/6997578 [Accessed 15 Oct. 2019].

Buckley, Ian. "6 Best Arduino Alternative Microcontrollers." *MakeUseOf*, 2 Feb. 2018, https://www.makeuseof.com/tag/best-arduino-alternative-microcontrollers/.

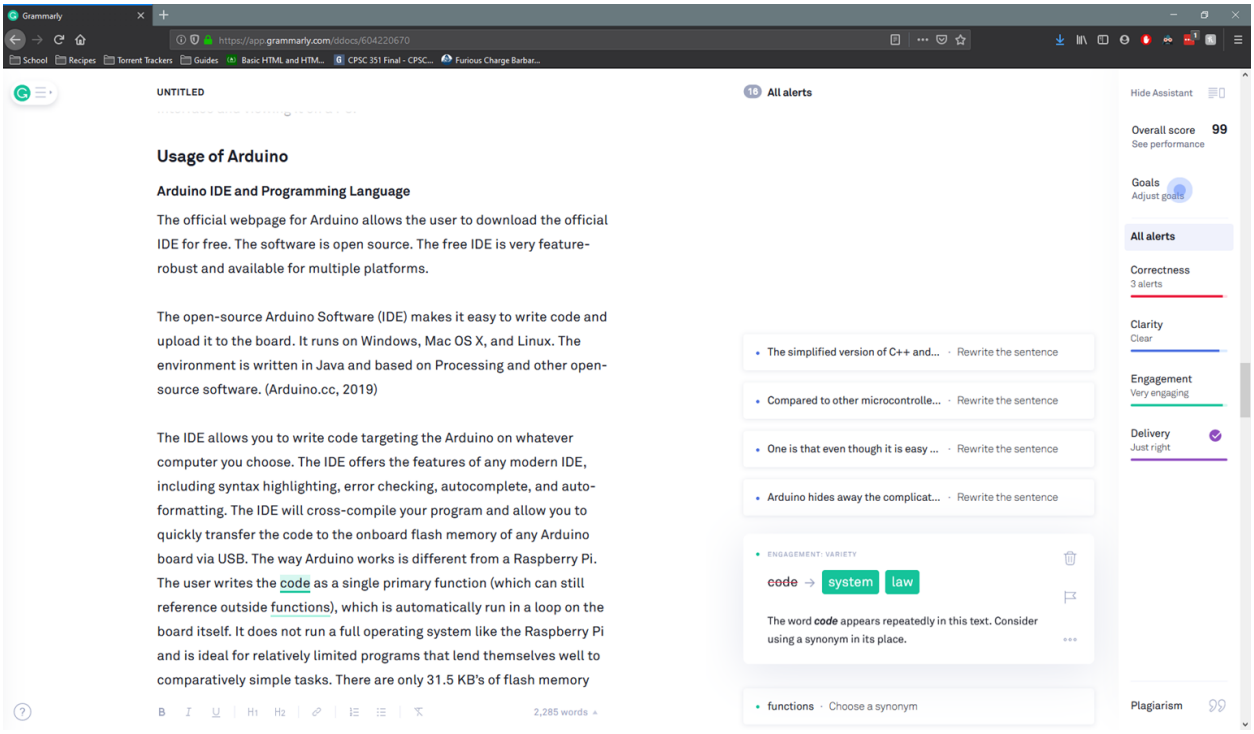
Sarwar, Ismail, et al. "Advantages and Disadvantages of Using Arduino." *Engineer Experiences*, 1 Oct. 2016, http://engineerexperiences.com/advantages-and-disadvatages.html. [Accessed 14 Oct. 2019]

Store.arduino.cc. (2019). *Arduino Uno Rev3*. [online] Available at: https://store.arduino.cc/usa/arduino-uno-rev3 [Accessed 15 Oct. 2019].

Torrone, Phillip. "Why the Arduino Won and Why It's Here to Stay: Make:" *Make*, 10 Feb. 2011, https://makezine.com/2011/02/10/why-the-arduino-won-and-why-its-here-to-stay/.

Zait, Anat. "4 Simple Steps for Debugging Your Arduino Project." *Circuito.io Blog*, 2 Apr. 2018, https://www.circuito.io/blog/arduino-debugging/. [Accessed 14 Oct. 2019]

# Grammarly




Youtube video of Grammarly running:
https://www.youtube.com/watch?v=gwe1PEicUyc&feature=youtu.be