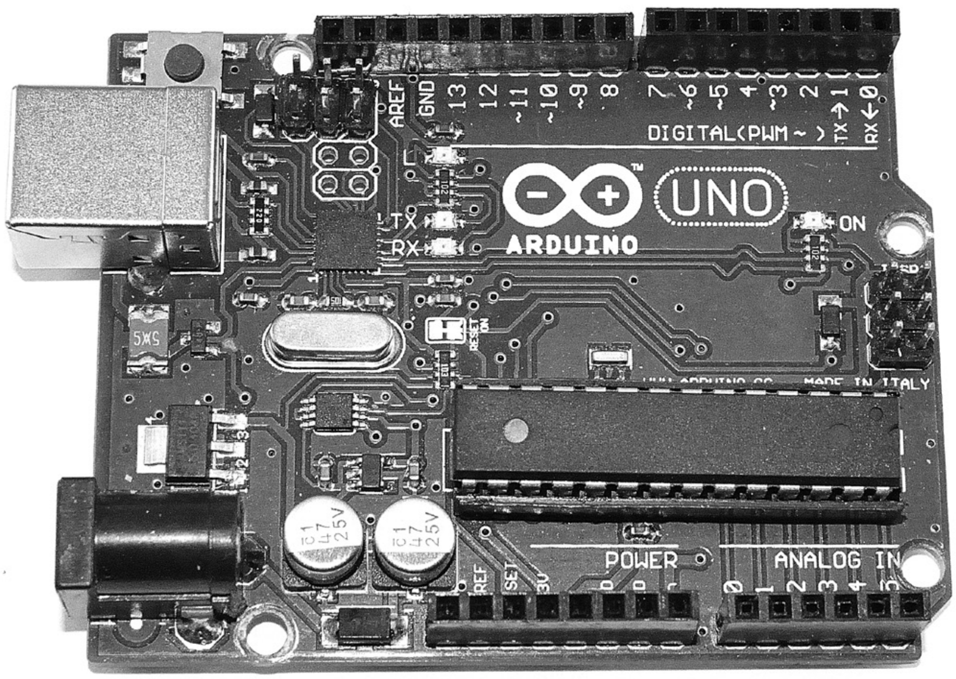
Introduction to the Arduino



The Arduino Uno.

The Arduino has become wildly popular among the hobbyist community. In 2011, there were an estimated 300,000 Arduino boards in use, not counting the many "clone" boards produced under the Arduino's unique Open Source licensing model. With its standalone single-board design, the Arduino can interface with a wide variety of sensors and controls easily and inexpensively. Based on the Atmel series of microcontrollers, the Arduino, with its onboard digital and analog I/O (input/output), is an easy and inexpensive way to build extremely versatile electronic projects.

Released under the Open Source Creative Commons Attribution Share-Alike license, the Arduino is totally Open Source, as described later in this chapter. From the board designs and schematic files, to the Arduino programs (known as "sketches") and libraries, everything is Open Source. You are free to do whatever you desire, as long as you properly credit the authors in your work and share any changes you make to the existing code and libraries. For the most part, this means that everything about the Arduino is either free or very low cost.

One of the main benefits of Open Source is that you have a whole community of hobbyists developing and sharing their projects freely. This can save you many hours of work if someone is working on projects similar to yours. You can freely integrate their libraries and code into your project, turning what could have been a months-long programming ordeal into a much shorter, more enjoyable path to a finished project

Along with the Arduino board itself, there is a vast selection of components and modules designed to interface with the Arduino. These devices use the various device communication protocols such as the Serial Peripheral Interface (SPI) and Inter-Integrated Circuit (I^2C) already built into the Arduino, allowing simple connections to the Arduino using only a few wires. Now you can create complex projects without having to dig through datasheets and solder for months as you had to in days gone by. For example, the Lightning Detector project presented later in this book needs only 11 wires to connect the lightning detector module and the Nokia LCD display to the Arduino. Since the libraries to communicate with these modules already existed, all that I had to do was include the libraries in the project and get right down to the brass tacks of what I wanted the project to be.

The Hardware

Although there are now numerous variations on the Arduino, the most common Arduino, the Uno consists of an Atmel ATmega328 8-bit microcontroller with a clock speed of 16 MHz. The ATmega328 has 32 KB of flash memory, 2 KB of static RAM (SRAM), and 1 KB of electrically erasable programmable read-only memory (EEPROM) onboard. The Arduino has 14 digital I/O pins. Six of these pins can also do pulse width modulation (PWM), and six 10-bit analog inputs can also be used for digital I/O pins. Two of the digital pins also directly support external hardware interrupts, and all 24 I/O pins support pin state change interrupts, allowing external hardware control of program execution.

Typically powered via the USB programming port, with its low current drain and onboard power regulator the Arduino is ideally suited for battery powered projects. The Arduino supports multiple communication protocols, including standard Serial, Serial Peripheral Interface (SPI), Two-Wire (also known as Inter-Integrated Circuit or I^2C), and 1-Wire. Designed for expandability, the Arduino I/O and power connections are brought out to a series of headers on the main board. The header layout is standard among the majority of the Uno-type boards and many of the basic Arduino add-ons, also known as *shields*, can be plugged directly into these headers and stacked one on top of the other, providing power and I/O directly to the shield without any additional wiring needed.

Many types of shields are available, including all manner of displays, Ethernet, Wi-Fi, motor driver, MP3, and a wide array of other devices. My personal favorite is the prototyping shield, which allows you to build your own interface to an even wider array of Arduino-compatible components, modules, and breakout boards. You can find GPS, real time clock, compass, text-to-speech, and lightning detection modules, for example, along with an endless list of sensors such accelerometers, pressure, humidity, proximity, motion, vibration, temperature, and many more. We'll explore some of these modules and sensors in projects presented in later chapters of this book.

History

As living proof that necessity is the mother of invention, the Arduino was created at the Interaction Design Institute Ivrea, in the northern Italian town of Ivrea. Originally designed as an inexpensive Open Source tool for students, replacing the more expensive and less powerful Parallax "Basic Stamp" development platform then used by students at the institute, the Arduino began as a thesis project in 2003 by artist and design student, Hernando Barragán, designed for a non-technical audience.

This project, known as *Wiring*, was based on a ready-to-use circuit board with an Integrated Development Environment (IDE) based on the *Processing* language created by Ben Fry and one of Barragán's thesis advisors, Casey Reas. Wiring was then adapted in 2005 by a team co-founded by another of Barragán's thesis advisors, Massimo Banzi. This team consisted of Hernando Barragán, Massimo Banzi, David Cuartielles, Dave Mellis, Gianluca Marino, and Nicholas Zambetti. Their goal was to further simplify the Wiring platform and design a simple, inexpensive Open Source prototyping platform to be used by non-technical artists, designers, and others in the creative field. Banzi's design philosophy regarding the Arduino is best outlined in his quote "Fifty years ago, to write software you needed people in white aprons who knew everything about vacuum tubes. Now, even my mom can program."

Unfortunately, at the same time, due a lack of funding the Institute was forced to close its doors. Fearing their projects would not survive or be misappropriated, the team decided to make the entire project Open Source. Released under the Open Source Creative Commons license, the Arduino became one of the first, if not the first, Open Source hardware products. Needing a name for the project, the team decided to name it Arduino after a local pub named "Bar Di Re Arduino" which itself honors the memory of Italian King Arduin.

Everything about the Arduino is Open Source. The board designs and schematic files are Open Source, meaning that anyone can create their own version of the Arduino free of charge. The Creative Commons licensing agreement allows for unrestricted personal and commercial derivatives as long as the developer gives credit to Arduino, and releases their work under the same license. Only the name Arduino is trademarked, which is why the various Arduino-compatible boards have names like Iduino, Ardweeny, Boarduino, Freeduino, and so on. Typically these boards are fully compatible with their official Arduino counterpart, and they may include additional features not on the original Arduino board.

Massimo Banzi's statement about the Arduino project, "You don't need anyone's permission to make something great," and Arduino team member David Cuartielles's quote, "The philosophy behind Arduino is that if you want to learn electronics, you should be able to learn as you go from day one, instead of starting by learning algebra" sums up what has made the Arduino so popular among hobbyists and builders. The collective knowledgebase of Arduino sketches and program libraries is immense and constantly growing, allowing the average hobbyist to quickly and easily develop complex projects that once took mountains of datasheets and components to build. The Arduino phenomenon has sparked the establishment of a number of suppliers for add-on boards, modules, and sensors adapted for the Arduino. The current (as of mid-2014) Arduino team consisting of Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, and David Mellis has continued to expand and promote the Arduino family of products.

Since its inception, the Arduino product line has been expanded to include more powerful and faster platforms, such as the 86 MHz 32-bit Arduino Due, based on the Atmel SAM3X8E ARM Cortex-M3 processor, and the dual-processor Arduino Yun, which contains the Atheros AR9331 running an onboard Linux distribution in addition to the ATmega32u4 processor that provides Arduino functionality. With the Arduino Tre, a 1-GHz Sitara AM335x processor based Linux/Arduino dual-processor design, the Arduino now has the power needed to support processing-intensive applications and high speed communications. The Arduino variants are discussed in more detail in Chapter 2.

What is Open Source?

Generally speaking, Open Source refers to software in which the source code is freely available to the general public for use and/or modification. Probably the best example of Open Source software is the Linux operating system created by Linus Torvalds. Linux has evolved into a very powerful operating system, and the vast majority of applications that run on Linux are Open Source. A large percentage of the web servers on the Internet are Linux-based, running the Open Source *Apache Web Server*. The popular Firefox web browser is also Open Source, and the list goes on. Even the Android phone operating system is based on Linux and is itself Open Source. This ability to modify and adapt existing software is one of the cornerstones of the Open Source movement, and is what had led to its popularity and success.

The Arduino team took the concept of Open Source to a whole new level. Everything about the Arduino — hardware and software — is released under the Creative Commons Open Source License. This means that not only is the Integrated Development Environment (IDE) software for the Arduino Open Source, the Arduino hardware itself is also Open Source. All of the board design file and schematics are Open Source, meaning that anyone can use these files to create their own Arduino board. In fact, everything on the Arduino website, **www.arduino.cc**, is released as Open Source.

As the Arduino developer community grows, so does the number of Open Source applications and add-on products, also released as Open Source. While it may be easier to buy an Arduino board, shield or module, in the vast majority of cases, everything you need to etch and build your own board is freely available for you to do as you wish. The only real restriction is that you have to give your work back to the Open Source community under the same Open Source licensing. What more could a hobbyist ask for? Everything about the Arduino is either free or low cost. You have a whole community of developers at your back, creating code and projects that you can use in your own projects, saving you weeks and months of development. As you will see in some of the projects in this book, it takes longer to wire and solder things together than it does to actually get it working. That is the true power of Open Source, everyone working together as a collective, freely sharing their work, so that others can join in on the fun.

Open Source Licensing and How it Works

There are several main variations on the Open Source licensing model, but all are intended to allow the general public to freely use, modify, and distribute their work. The most common Open Source license models you will encounter include the GNU General Public License (GPL), Lesser GPL (LGPL), MIT, and the Creative Commons licenses. As a general rule, for the average hobbyist, this means you are free to do as you wish. However, there will always be those of us that come up with that really cool project we can package up and sell to finance our next idea. It is important for that group to review and understand the various license models you may encounter in the Open Source world.

The GNU GPL

As with all Open Source licensing models, the GNU General Public License (GPL) is intended to guarantee your freedom to share, modify, and distribute software freely. Developers who release software under the GPL desire their work to be free and remain free, no matter who changes or distributes the program. The GPL allows you to distribute and publish the software as long as you provide the copyright notice, disclaimer of warranty, and keep intact all notices that refer to the license. Any modified files must carry prominent notices stating that you changed the files and the date of any changes.

Any work that you distribute and publish must be licensed as a whole under the same license. You must also accompany the software with either a machine-readable copy of the source code or a written offer to provide a complete machine readable copy of the software. Recipients of your software will automatically be granted the same license to copy, distribute, and modify the software. One major restriction to the GPL is that it does not permit incorporating GPL software into proprietary programs.

The copyright usage in the GPL is commonly referred to as "copyleft," meaning that rather than using the copyright process to restrict users as with proprietary software, the GPL copyright is used to ensure that every user has the same freedoms as the creator of the software.

There are two major versions of the GPL, Version 2, and the more recent Version 3. There are no radical differences between the two versions; the changes are primarily to make the license easier for everyone to use and understand. Version 3 also addresses laws that prohibit bypassing Digital Rights Management (DRM). This is primarily for codecs and other software that deals with DRM content. Additional changes were made to protect your right to "tinker" and prevent hardware restrictions that don't allow modified GPL programs to run. In an effort to prevent this form of restriction, also known as Tivoization, Version 3 of the GPL has language that specifically prevents such restriction and restores your right to make changes to the software that works on the hardware it was originally intended to run on. Finally, Version 3 of the GPL also includes stronger protections against patent threats.

The Lesser GNU General Public License (LGPL)

The LGPL is very similar to the GPL, except that it permits the usage of program libraries in proprietary programs. This form of licensing is generally to encourage more widespread usage of a program library in an effort for the library to become a de-facto standard, or as a substitute for a proprietary library. As with the GPL, you must make your library modifications available under the same licensing model, but you do not have to release your proprietary code. In most cases, it is preferable to use the standard GPL licensing model.

The MIT License

Originating at the Massachusetts Institute of Technology, the MIT license is a permissive free software license. This license permits reuse of the software within proprietary software, provided all copies of the software include the MIT license terms. The proprietary software will retain its proprietary nature even though it incorporates software licensed under the MIT license. This license is considered to be GPL-compatible, meaning that the GPL permits combination and redistribution with software that uses the MIT License. The MIT license also states more explicitly the rights granted to the end user, including the right to use, copy, modify, merge, publish, distribute, sublicense, and/or sell the software.

The Creative Commons License

There are multiple versions of the Creative Commons License, each with different terms and conditions:

•Attribution (CC BY) — This license allows others to distribute, remix, tweak, and build upon a work, even commercially, as long as they credit the creator for the original creation.

•Attribution-NonCommercial (CC BY-NC) — This license allows others to remix, tweak, and build upon a work non-commercially. While any new works must also acknowledge the creator and be non-commercial, any derivative works are not required to be licensed on the same terms.

•Attribution-ShareAlike (CC BY-SA) — This is the most common form of the Creative Commons License. As with the Attribution license, it allows others to distribute, remix, tweak, and build upon a work, even commercially, as long as they credit the creator for the original creation, and license their new creation under the same license terms. All new works based on yours convey the same license, so any derivatives will also allow commercial use.

•Attribution-NonCommercial-ShareAlike (CC BY-NC-SA) — This license allows others to distribute, remix, tweak, and build upon a work non-commercially, as long as they credit the creator and license their new creations under the identical licensing terms.

•Attribution-No Derivs (CC BY-ND) — This license allows for redistribution, both commercial and non-commercial, as long as it is passed along unchanged and in its entirety, with credit given to the original creator.

•Attribution-NonCommercial-NoDerivs (CC BY-NC-ND) — This is the most restrictive of the Creative Commons licenses, only allowing others to download a work and share them with others as long as they credit the creator. Works released under this license cannot be modified in any way, nor can they be used commercially.

The Arduino is released under the Creative Commons Attribution-ShareAlike (CC BY-SA) license. You can freely use the original design files and content from the Arduino website, **www.arduino.cc**, both commercially and non-commercially, as long as credit is given to Arduino and any derivative work is released under the same licensing. So, if by chance you do create something that you would like to sell, you are free to do so, as long as you give the appropriate credit to Arduino and follow the requirements outlined in the FAQ on the Arduino website, as you may not be required to release your source code if you follow specific guidelines. If you include libraries in your work, be sure you use them within their licensing guidelines. The core Arduino libraries are released under the LGPL and the Java-based IDE is released under the GPL.

In Conclusion

It is this Open Source licensing that has made the Arduino so popular among hobbyists. You have the freedom to do just about anything you want and there are many others developing code and libraries you can freely incorporate in your code, which helps make developing on the Arduino so much fun. For example, I know very little about Fast Fourier transforms, but there is a fully functional library out there just waiting for me to come up with a way to use it. That's the beauty of the Arduino and Open Source. You don't have to be a programming genius to create fully functional projects as long as you have the entire Open Source developer community to draw upon. And, when you do start creating wonderful new things, remember to share them back to the community, so that others following in your footsteps can benefit from your work and create wonderful new things of their own.

References

Arduino — **www.arduino.cc**
Arduino Shield List — **www.shieldlist.org**
Atheros Communications — **www.atheros.com**
Atmel Corp — **www.atmel.com**
Creative Commons — **http://creativecommons.org**
GNU Operating System — **www.gnu.org**
Open Source Initiative **— http://opensource.org**
Texas Instruments — **www.ti.com**