Application Note

Implementing the ATmega328 Microcontroller on a PCB

by Katherine Cayago

# Introduction

 The ATmega328 chip is an 8-bit microcontroller from which the Arduino UNO microcontroller board is based. When designing a PCB that involves the use of the Arduino UNO, it is useful to use the ATmega328 chip from the Arduino Uno board to save space on the PCB. The ATmega328 chip contains most of the functions of the Arduino UNO microcontroller board, but must be wired and setup in a certain way in order to function on its own. This application note will cover how to implement the ATmega328 chip onto a PCB.

 In some cases, it may be necessary to order new ATmega328 chips to implement onto your PCB. However, these chips must often be bootloaded with software from the Arduino UNO in order for a user the upload his or her programs. This application note will cover how to load the necessary software onto the ATmega328 chip using the Arduino UNO.

PCB Implementation

The ATmega328 chip has a different pinout scheme than the Arduino UNO. Below is a pinout diagram of the ATmega328 with the corresponding Arduino pins courtesy of the Arduino website.



 One of the differences between the ATmega328 chip and the Arduino UNO board is that the UNO contains a 16MHz oscillator while the ATmega328 chip does not. A separate 16MHz crystal must be attached to pins 9 and 10 on the diagram. This crystal can be through-hole or surface mount, although a surface mount crystal may be desirable to reduce the space taken up on the PCB.

 Another thing to note are the VCC, GND, reset, and REF pins. Pins 8 and 22 should be hooked to ground, while Pin 7 should be hooked to a 5 volt power supply. Pins 1, 20, and 21 should generally be left alone unless an analog reference is needed or a reset is required. Otherwise, connecting those pins may result in output errors, even if they are connected to ground.

 Additional PCB tips include creating an outline of the chip on your PCB board with a notch indicating which side the first pin is located. This makes it easier to identify the orientation of the chip on your board. It is also useful to solder chip sockets on the PCB so that the ATmega328 chip can be removed easily. When doing so, make sure to solder all surface mount components first before soldering the chip sockets. This is because you may need to use an oven to solder surface mount components and it is possible that the high heat will ruin your chip socket.

# Bootloading the ATmega328

 If your project is dependent upon the ATmega microcontroller chip, it is beneficial to have several backup chips you can use in case your ATmega chip breaks or stops working. Replacement chips can be bought online at a low cost. However, these chips will not automatically work with the Arduino Uno and must be bootloaded with the correct software. Bootloading is the term used to describe the process of loading software from a working, Arduino-compatible ATmega chip to an unconfigured, default ATmega chip. The bootloading process outlined in this application note refers specifically to the use of the ATmega328P-PU chip with the Arduino Uno, courtesy of a tutorial written by Nick Gammon ([link](http://www.gammon.com.au/forum/?id=11637)). It is important to note that the bootloading procedure may be different for different models of the ATmega and different versions of the Arduino board.

 You will need an Arduino Uno that contains a working ATmega chip, an unloaded ATmega chip, a breadboard, a 10kohm resistor, and two .1 uf capacitors. Connect the following pins together on your breadboard:

Uno ATmega on Breadboard

D10 Pin 1

D11 Pin 17

D12 Pin 18

D13 Pin 19

GND GND

VCC (+5v) VCC

 Place the 10kohm resistor across pin 1 and the 5 volt rail and then place a .1 uf capacitor between GND and VCC on the two rails on either side of the chip. Upload the “board detector” sketch to ensure that the target ATmega chip is detected. From the instructions, you may choose to use an oscillator or to use the internal clock provided by the Uno to bootload the ATmega. Ideally, you may want to order an oscillator in case the provided tutorial does not work for whatever reason. This is because most bootloading tutorials involving the Uno require an oscillator. However, this useful tutorial outlines one of the few ways the ATmega can be bootloaded without an oscillator, which is very useful in case one does not have the time or resources to get an oscillator.

 After verifying that the software is on the new ATmega, test it by uploading a code onto your bootloaded chip. It should compile correctly. If not, try bootloading the chip again. If you used the set of instructions using no oscillator, try bootloading with an oscillator. It may be the some chips do need an oscillator to be bootloaded.