Build Your Own Arduino

Project #1 Circuit Design

With a few inexpensive parts and a solderless breadboard you can quickly and easily build your own Arduino. This concept works great when you want to prototype a new design idea, or you don't want to tear apart your design each time you need your Arduino.

The example below shows how to hook up the components on your breadboard. We will go into further detail throughout this project.

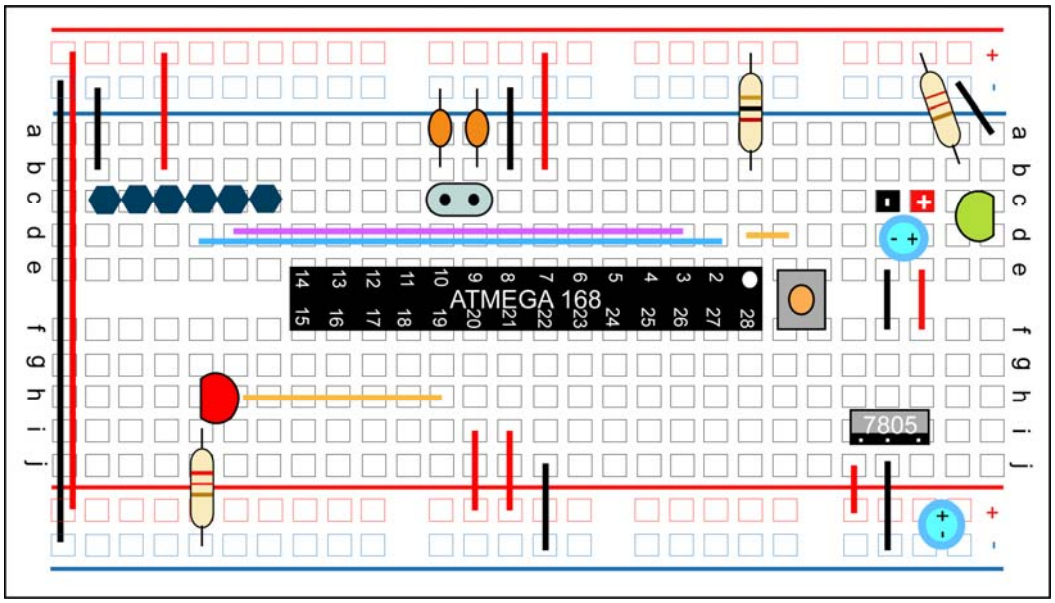

Figure 1-1: Breadboard Arduino with USB programming ability.

Before we get started, make sure you have all the necessary items in the component list box.

If you need to purchase parts you can do so from my site at www.ArduinoFun.com or in the back of this book is a listing of several sites that sell Arduino related items.

*** See note** about the TTL-232R cable in programming options before purchasing.

Component List

- Breadboard (440 or 840 Tie Point)
- 22 AWG Wire (Various Colors)
- 7805 Voltage regulator
- 2 LEDs (Any Color)
- 2 220 Ohm Resistors (Red, Red, Brown)
- 1 10k Ohm Resistor (Brown, Black, Red)
- 2 10 uF Capacitors
- 16 MHz Clock Crystal
- 2 22 pF Capacitors
- Small Momentary Tact Switch
- 1 Row Male Header Pins
- TTL-232R-3V3 USB - Serial Converter Cable

Build Your Own Arduino

Project #1 Step by Step

The first thing you need to do is set up power. With your breadboard and components in front of you… let's get started!

With this step, you will be setting the breadboard Arduino up for constant +5Volts power using a 7805 voltage regulator.

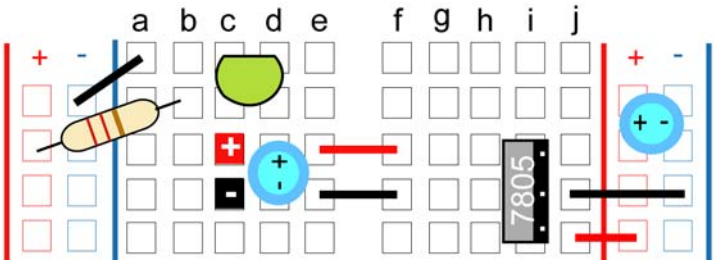

Figure 1-2: Power setup with LED indicator.

In order for the voltage regulator to work, you need to provide more than 5V power. A typical 9V battery with a snap connector would work just fine for this.

Power is going to come into the breadboard where you see the red and black **+** and **–** squares. Then add one of the 10uF capacitors. The longer leg is the Anode (Positive) and the shorter leg is the Cathode (Negative). Most capacitors are also marked with a stripe down the negative side.

Across the empty space on the breadboard (the channel) you will need to place two hook-up wires for positive (red) and ground (black) to jump power from one side of the breadboard to the other.

Now add the 7805 voltage regulator. The 7805 has three legs. If you are looking at it from the front, the left leg is for voltage in (Vin) the middle leg is for ground (GND) and the third leg is for voltage out (Vout). Make sure the left leg is lined up with your positive power in, and the second pin to ground.

Coming out of the voltage regulator and going to the power rail on the side of the breadboard you need to add a GND wire to the ground rail and then the Vout wire (3rd leg of the voltage regulator) to the positive rail. Add the second 10uF capacitor to the power rail. Paying attention to the Positive and Negative sides.

It's a good idea to include an LED status indicator which can be used for troubleshooting. To do this you need to connect the right side power rail with the left power rail. Add positive to positive and negative to negative wires at the bottom of your breadboard.

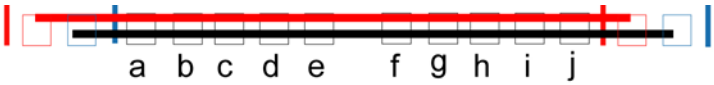

Figure 1-3: Left and Right Power Rail Connections.

Having power on the left and right power rail will also help to keep your breadboard organized when providing power to the various components.

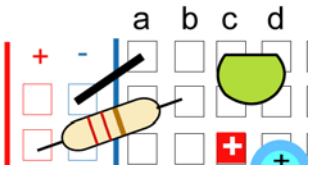

Figure 1-4: For the LED status indicator, connect a 220Ω resistor (colored as: red, red, brown) from power to the anode of the LED (positive side, longer leg) and then a GND wire to the cathode side.

Congratulations, now your breadboard is set up for +5V power. You can move onto the next step in the circuit design.

Now we want to prepare the ATmega168 or 328 chip. Before we begin, let's take a look at what each pin on the chip does in relationship to the Arduino functions. NOTE: The ATmega328 runs pretty much the same speed, with same pinout, but features more than twice the flash memory (30k vs 14k) and twice the EEPROM (1Kb vs 512b).

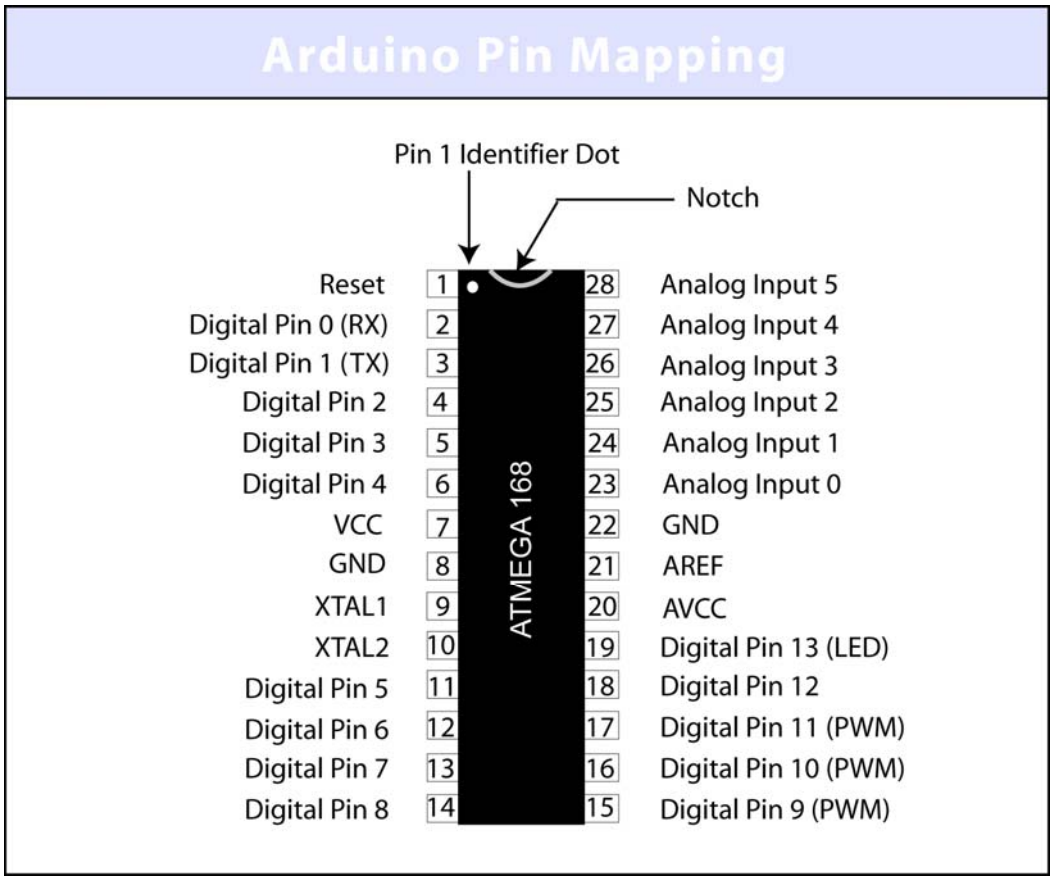

Figure 1-5: Arduino Pin Mapping

The ATmega168 chip is created by Atmel. If you look up the datasheet you won't find that the above references are the same. This is because the Arduino has its own functions for these pins, and I have provided them only on this illustration. If you would like to compare or need to know the actual references for the chip, you can download a copy of the datasheet at www.atmel.com. Now that you know the layout of the pins, we can start hooking up the rest of the components.

To start, we will build the supporting circuitry for one side of the chip and then move on to the other side. Pin one on most chips has an identifier marker. Looking at the ATmega168 or 328 you will notice a u-shaped notch at the top as well as a small dot. The small dot indicates that this is pin 1.

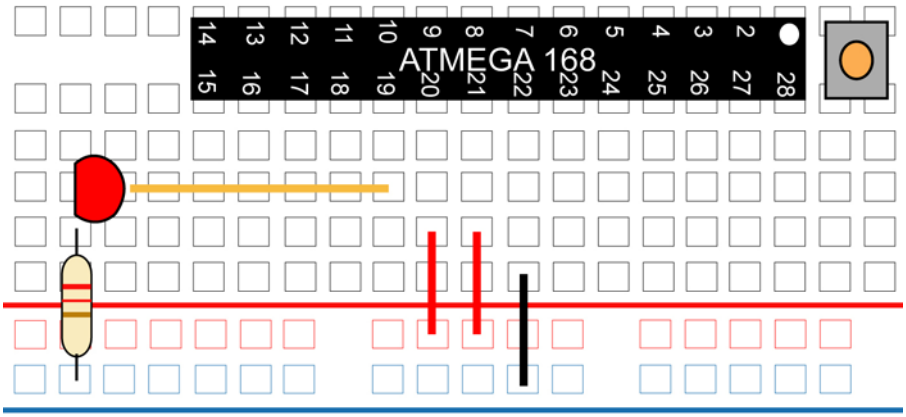

Figure 1-6: Supporting circuitry pins 15-28

From the GND power bus, add a jumper wire to pin 22. Next, from the positive power bus, add jumper wires to pin 20 (AVCC - *Supply voltage for the ADC converter. Needs to be connected to power if ADC isn't being used and to power via a low-pass filter if it is* (a low pass filter is a circuit that cleans out noise from the power source, we aren't using one)

Then add a jumper wire from the positive bus to pin 21 (*Analog reference pin for ADC)*.

On the Arduino, pin 13 is the LED pin. Note that on the actual chip the pin is number 19. When uploading your sketch code and for all projects you will still reference this as Pin 13.

To hook up the LED, add a 220Ω resistor from GND to the cathode of the LED. Then from the anode of the LED add a jumper wire to pin 19.

Now we can move onto the other side of the chip. You are almost finished!

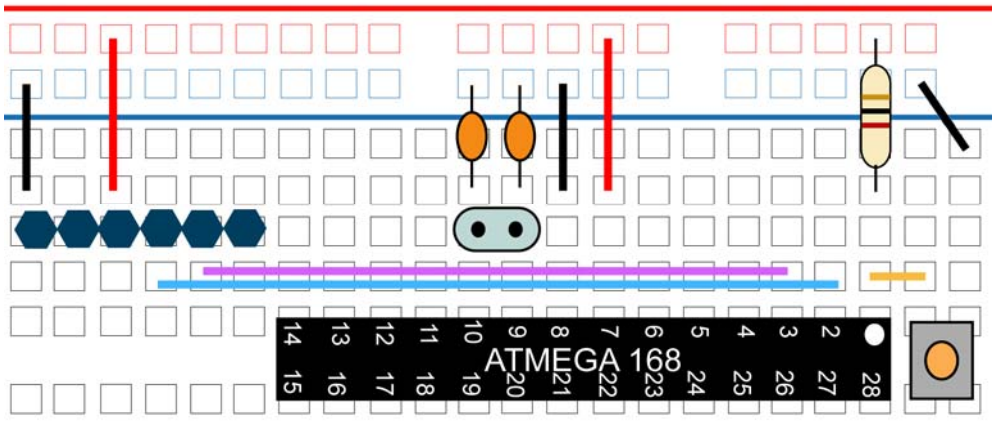

Figure 1-6: Supporting circuitry pins 1-14

Above the ATmega168 chip near the pin 1 identifier, place the small tact switch. This switch is used for resetting the Arduino. Right before you upload a new sketch to the chip you will want to press this once. Now add a small jumper wire from pin 1 to the bottom leg of the switch then add the 10K resistor from power to the pin 1 row on the breadboard. Finally add a GND jumper wire to the top leg of the switch.

Add power and GND jumpers to pin 7(VCC) and pin 8 (GND). Add the 16MHz clock crystal to pin 9 and 10 and then the two .22pF capacitors from pins 9 and 10 to GND. (See note below for alternative method).

Your basic breadboard arduino is now complete. You could stop right here if you wanted to and swap an already programmed chip from your Arduino board to the breadboard, but since you came this far, you might as well finish off by adding some programming pins. This will allow you to program the chip from the breadboard.

NOTE: Instead of using the 16MHz clock crystal, you can use a 16 MHz ceramic resonator with built-in capacitors, three-terminal SIP package. You will have to arrange your breadboard a little differently, the resonator has three legs. The middle leg will go to ground and the other two legs will go to pins 9 & 10 on the ATmega168 chip.

Referring to Figure 1-6, locate a spot where you have 6 columns on the breadboard that are not in contact with anything else. Place a row of six male header pins here.

With the breadboard facing you, the connections are as follows:

GND, NC, 5V, TX, RX, NC, I am also calling these pins 1,2,3,4,5,6. From your power bus rail, add the GND wire to pin 1 and a wire from power for pin 3. NC means not connected, but you can connect these to GND if you want to.

From pin 2 on the ATmega168 chip, which is the Arduino RX pin, you will connect a wire to pin 4 (TX) of your programming headers. On the ATmega168 chip, pin 3 Arduino TX gets connected to pin 5 (RX) on your header pins.

The communication looks like this: ATmega168 RX to Header Pin TX, and ATmega168 TX to Header Pin RX.

Now you can program your breadboard Arduino.

Programming Options

The first option is to buy a TTL-232R 3.3V USB – TTL Level Serial Cable. These can be purchased at www.adafruit.com or www.ftdichip.com

The other two options, which I prefer are to buy one of two breakout boards from SparkFun.com. They are:

- FT232RL USB to Serial Breakout Board, SKU: BOB-0071 (This option takes up more space on your breadboard)
- FTDI Basic Breakout - 3.3V SKU: DEV-08772 (This option, and using right angle male headers works the best out of all three because it is secured better on the breadboard)

Build Your Own Arduino

Project #1 Programming the ATmega168 chip

Double check your connections, make sure your 9V battery is not connected and hook up your programming option. Open up the Arduino IDE and in the Example sketch files, under Digital, load the Blink sketch.

Under the file option Serial Port, select the COM port that you are using with your USB cable. i.e. COM1, COM9, etc.

Under the file option Tools/Board, select either:

- Arduino Duemilanove w/ATmega328
- Arduino Decimila, Duemilanove or Nano w/ATmega128

(depending on which chip you are using with your breadboard Arduino)

Now press the upload icon and then hit the reset button on your breadboard. If you are using one of the SparkFun breakout boards, you will see the RX and TX lights blink. This lets you know that the data is being sent. Sometimes you need to wait a few seconds after pressing the upload button before pressing the reset switch. If you have trouble, just experiment a little with how fast you go between the two.

This sketch if uploaded properly will blink the LED on pin 13 on for one second, off for one second, on for one second… until you either upload a new sketch or turn off the power.

Once you have uploaded the code, you can disconnect the programming board and use your 9V battery for power.

Troubleshooting

- No Power – Make sure your source power is above 5V.
- Power but nothing works – recheck all your connection points.
- Uploading error – Refer to www.arduino.cc and do a search on the particular error message you receive. Also check the forums as there is a lot of great help there.