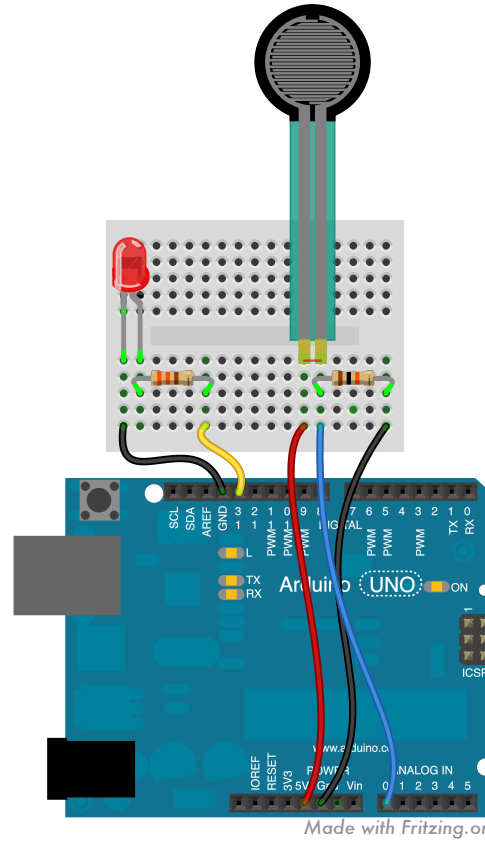
VIII. READING ANALOG VALUES



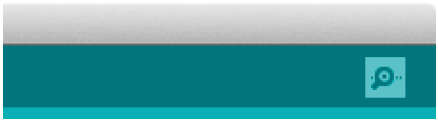
Disconnect USB. Remove the pushbutton and connect a **force-sensitive resistor**, the 10 kΩ resistor and wires to analog input **A0** as shown.

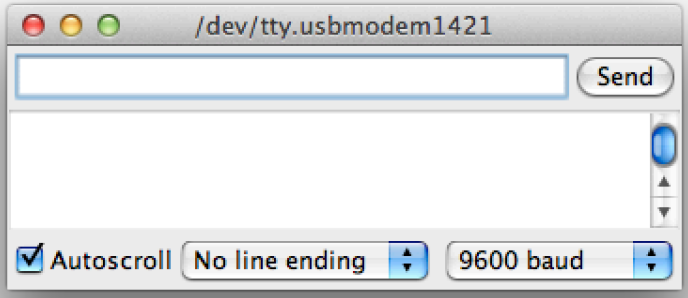
You've created a **voltage divider**. Normally the force sensor has a resistance much higher than 10 kΩ, so A0 gets pulled to near 0V by the 10 kΩ resistor. Press on the force sensor, and you lower its resistance. It can go way below 10 kΩ, and the voltage on A0 then goes up to almost 5V. So: more pressure – higher voltage. That's how you can read continuous (analog) values.

Load **File:Examples:03. Analog:AnalogInput.** It uses **analogRead(...)** to read the voltage on A0, and turns the LED on and off with a delay determined by that voltage – the harder you press, the slower the LED will blink. analogRead() returns values from 0 (0V) to **1023** (5V).

IX. DEBUGGING USING SERIAL OUTPUT

It's hard to really see what values you're getting from the force sensor through the analog pin just by looking at the blink frequency of that LED. To see the actual numbers, open **File:Examples:01.Basics:AnalogReadSerial.** Upload it, then click on the **magnifier** toolbar button in the top right to open the **Serial Monitor** window. You'll see values changing as you press the sensor.



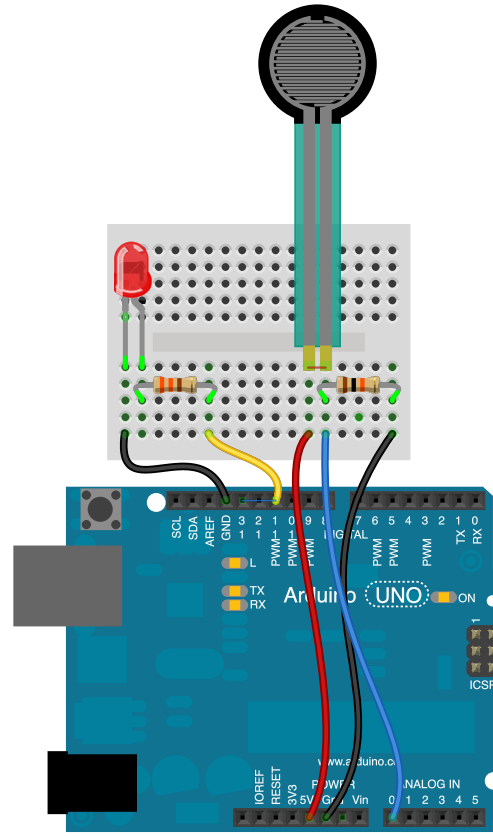


The code uses **Serial.begin(9600)** to open a serial connection back to your computer in setup(), and **Serial.println(...)** to output ("print") numbers to that serial connection, which end up in the Serial Monitor window on your screen. It's also possible to send data back to the Arduino that way, using **Serial.read(...).**

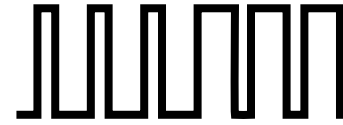
X. ANALOG OUTPUT AND PWM

Disconnect USB. Move the yellow wire from pin 13 to pin 11. Pin 11 has a **tilde (~)** on the board, which means it can output analog values.

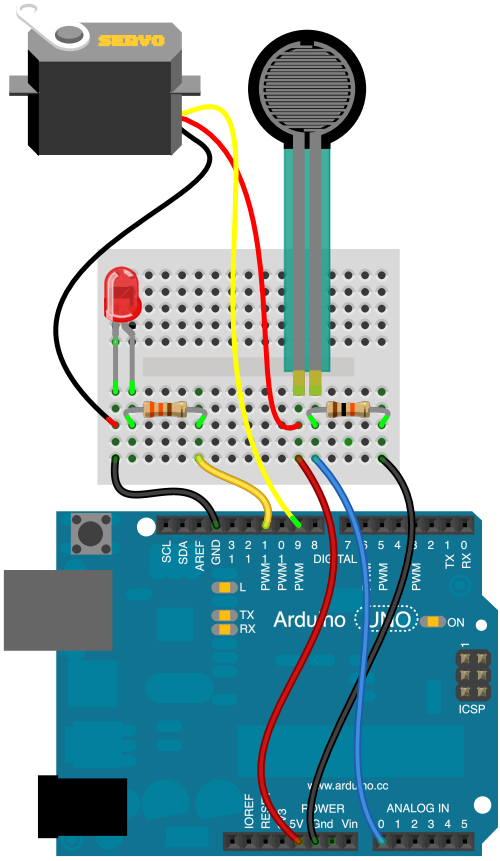
Change your loop() to control the LED with **analogWrite(...)**. Analog values for output go from 0 to **255**, not 1023, so divide the value from analogRead(...) by 4 before writing it to the LED pin.



Connect USB and upload your code. Now you can control the brightness of your LED by pressing on the force sensor.



Arduino uses **Pulse-Width Modulation (PWM)** to create analog values — it'll turn the output on (5V) and off (0V) at 500 Hz, and increase the duty cycle (relative on-time) of that square wave signal to represent higher analog values. 500 Hz is too fast for the human eye, so the LED looks like it's always on, just more or less bright.



Made with Fritzing.org

XI. CONTROLLING SERVOS

Disconnect USB, and add a servo to your setup: Connect its **black or dark brown lead to GND**, its **red lead to 5V**, and its **orange, yellow or white lead** (the "**signal**" lead) to pin 9, by sticking jumper wires into the servo connector.

Load the sample sketch **File:Examples:Servo:Knob** (it's further down in the list). Instead of the potentiometer (an adjustable resistor) mentioned in the code we'll just use the force sensor voltage divider we already have to provide an analog value to input A0.

Run the code, and you can **control the angular turning position of the servo by pressing on the sensor.**

Internally, servos are also controlled by PWM – the longer the signal, the further to the right they turn, usually in a range of 0..180 degrees.

However, the **Servo library** for Arduino takes care of all this. Note the **#import** statement, the **Servo** object declaration, and how the sample code then sends data to the servo using angular values. You can easily declare, create, and control two or more Servo objects this way – essential for your robot!

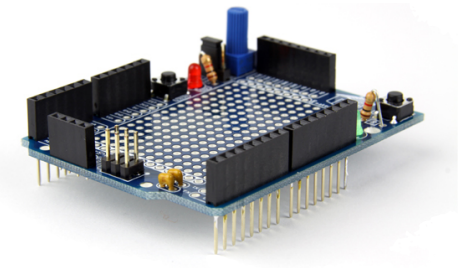
Servos can take around 0.5–1s to reach their target position. For a simple walking movement, you can just send the maximum angle, wait, then send the minimum angle, wait, and so on.

Tip: If you run out of 5V pins on the Arduino, **bring 5V over to a column** on the breadboard, and connect things from there (similar for GND). Use red and black wires to keep your sanity! On larger breadboards like the large one in the MAKE Pack, use the horizontal connector **rows** ("rails") along the top and bottom – put 5V only to the top red rail, and GND only to the bottom blue rail, to avoid plugging things into the wrong one.

XII. SHIELDS

Shields are PCBs that **stack on top of the Arduino** and connect to all Arduino pins to add all kinds of hardware features. There are shields to play MP3 files, for WiFi, Bluetooth, Ethernet, Zigbee, MIDI, GPS, to log data, drive big motors, etc. — shieldlist.org lists over 200!

Good shields are **stackable** — they have the same female header pins as the Arduino on top.



The MAKE Pack contains a kit for a **MakerShield** prototyping shield (see photo). **Solder** it together (URL for instructions on the pack), stick the mini breadboard onto it, and you have a very useful board with some built-in LEDs, buttons, even a potentiometer for quick experiments. It's not a perfect fit for the Arduino, but it works well enough. **SparkFun** has a similar **ProtoShield**.

RadioShack also carries a simpler, but more **modern prototyping shield** designed by the Arduino team, to solder your own parts onto. In the future, look for shields like that one, with **18 pins at the top**, which is the **new Arduino standard**.

XIII. SOME POINTERS

To learn more about Arduino and build fun projects, check out these resources:

arduino.cc is your first stop for help with the IDE, the Arduino language reference, board specifications, new boards and software versions, excellent mostly up-to-date tutorials, and libraries for a lot of stuff you may want to hook up to your Arduino.

Sparkfun.com has great components, shields, sensors, breakout boards, etc. They support open-source hardware and have all schematics online. Decent pricing, great community. Epic Friday New Product Post videos.

Adafruit.com is smaller but similar, with great products, learning resources and community support.

MAKE Magazine has an Arduino blog with great tutorial videos at blog.makezine.com/arduino/ and lots of Arduino projects at makeprojects.com/c/Arduino. The paper magazine is also great fun, especially with kids.

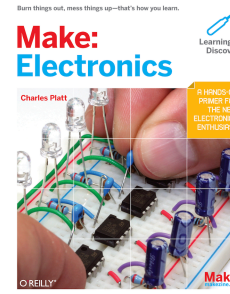
RadioShack.com has all essential parts, a bit pricey, but lets you check online if a part is available at your local store around the corner – perfect for those last-minute needs that always seem to come up.

Digikey.com, **Mouser.com**, and **Farnell.com** are professional electronic component vendors. They carry and have datasheets for **everything**, at the best prices if you know what you're looking for, but they are **overwhelming** to beginners – try SparkFun, MakerShed, Adafruit or RadioShack at first.

Fritzing.org has a nice free tool to document your Arduino breadboard designs, and to design shields that can then be made by submitting your files to an online PCB maker. I used it for all the Arduino diagrams here.

In general, if you want to hook up X to an Arduino, **google "Arduino X"** and you're likely to find a solution. :) Look for the above sites among the search results.

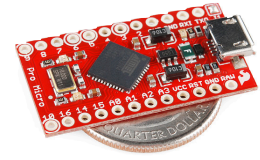
There are countless Arduino **books** out there; the gentlest is probably Massimo Banzi's **Getting Started With Arduino** (he designed the Arduino board). The PDF is ten bucks at the Makershed. Tom Igoe's **Making Things Talk** is excellent and beautifully designed, focusing on making Arduinos and other electronic devices connect and share information. **Arduino Bots and Gadgets** is interesting if you want to build robots. However, **all** Arduino books become **outdated quickly** because the Arduino boards and IDE have changed slightly almost every year so far. Look for a book edition that's **no older than a year.**



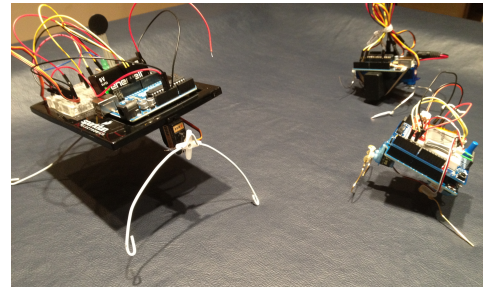
For a current book on **basic electronics** (not Arduino), there is no better choice than **Make: Electronics** by Charles Platt. Beautifully illustrated, starts with the basics, very accessible and fun. And experiment #1 is licking a 9V battery. I read it cover to cover to refresh my electronics knowledge.

In general, **OReilly.com** has excellent books on Arduino and other techie topics, with DRM-free versions for iPads and in PDF, lifetime access and dropbox syncing, **frequent updates** of their PDF editions, 3-for-2 deals, and special pricing if you own the printed book.

There are dozens of **different Arduino boards** out there. Stick with those documented on arduino.cc at first. Their new **Leonardo** board, e.g., can act as a USB keyboard or mouse, the new **Due** is faster, and SparkFun's **Arduino Pro Micro** (right) is a great tiny option, but for a beginner they're not quite as smooth to use yet.



Without a doubt this version of this guide will become outdated just as quickly as all those books. **Check back** at the URL on the front cover for **updates**, and meanwhile, have fun hacking and making with Arduino!



About the author

Jan Borchers is a professor of computer science and head of the Media Computing Group at RWTH Aachen University. He works in human-computer interaction, usability, and digital fabrication, and has taught Arduino to students and kids since 2008.

Feel free to use this booklet for yourself, with your friends, or in noncommercial classes. **Instead of hosting a local copy, please link back to the URL** on the cover so we can keep old versions from floating around. I'll keep older versions of the booklet there for reference. Thanks!

Version history

2013-08-05 (1.8): Added QR code, adjusted cover page graphics.

2013-07-31 (1.7): Clarified p.4, updated SparkFun prices, corrected pinMode typo, updated for IDE 1.0.5.

2013-01-24 (1.6): Added target audience, author info, premium headers on wishlist. Updated pushbutton declaration. Updated for IDE 1.0.2 & 1.0.3. Cosmetic corrections throughout.

2012-08-17 (1.5): Corrected button use in Ch.VI. Changed title page, tips layout, last page layout.

2012-08-15 (1.4): Added shieldlist.org.

2012-08-15 (1.3): Updated acknowledgements, Java reference, +5V pin, enabling pullups, page footers, layout. Added Creative Commons license terms.

2012-08-15 (1.2): Added: cropmarks, boldface in first chapters, Raspberry Pi, SparkFun wish list, other starter kits, Electronic Toolbox, Arduino Pro Micro, Make: Electronics cover, double-sided layout for binding. Edited: shields, O'Reilly, title, headings.

2012-08-12 (1.1): Added: missing sensor wire in ch. X+XI diagrams, Circuit Playground, different boards, version history. Fixed typos.

2012-08-09 (1.0): Initial release, see Acknowledgements.