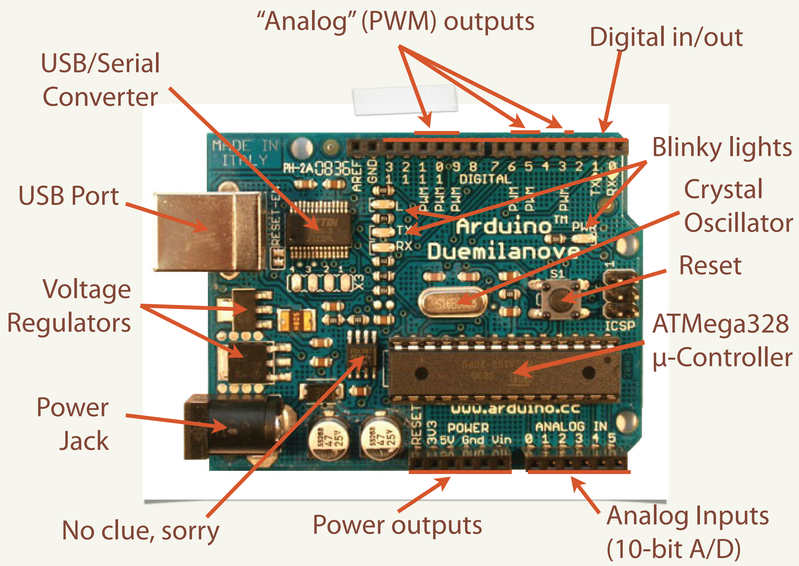
Using Arduino Microcontrollers as Inexpensive Dataloggers

Dr. Eric Ayars, California State University Chico

California State University Chico Physics
Today Decides Tomorrow
www.csuchico.edu/phys

What the Arduino is

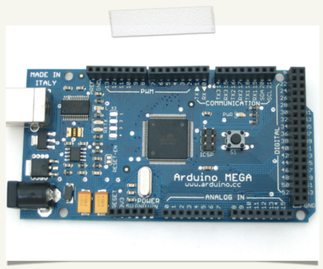


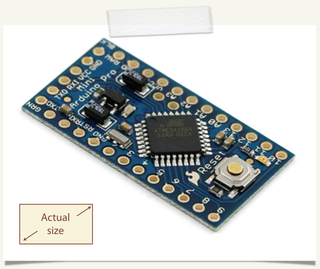
The Arduino is an open-source microcontroller system based on the ATMega series of chips. All of the software *and hardware* is open-source: if you don't want to buy one you can just build your own using the freely-available schematics. (The cost for a professionally-fabricated unit is less than $30, though, so it's easiest to just buy one.) What makes an ATMega chip an Arduino is the "bootloader" code which allows it to be programmed via a serial connection rather than a more complicated (and expensive) hardware programmer. This bootloader modification decreases the available program memory by about 2kB, but the result is a very easy-to-program device.

Arduino programs are written in "Wiring", a simplified version of c/c++. Once the program is written (on your computer), it is uploaded to the Arduino via a USB cable. The program instructions are saved in flash memory on the microcontroller, which runs the code any time power is supplied. The Arduino does not need to be connected to the computer to run, once it has been programmed.

The ATMega328-based Arduino has 30kB of usable program memory, 2kB of RAM for variable storage, and another 1kB of non-volatile EEPROM. It has 14 dedicated digital I/O lines, six of which can be used for Pulse-Width Modulation (PWM) output. It also has six 10-bit A/D inputs. It includes support for UART TTL serial and I2C communications, as well as external interrupts.

The board shown above is the "classic" Arduino. There are numerous other designs, such as the Arduino Mega (with 54 DIO pins, 14 PWM outputs, 16 AI pins, and 124kB of usable program memory) and the Arduino Pro Mini (same specifications as the Arduino, but without on-board power or USB, and the size reduced to 18x33x2mm).





What kind of data the Arduino can collect

Digital Timing

The "standard" Arduino can collect digital data from up to 20 lines with sub-millisecond timing resolution. In addition, two of those digital lines can be configured as hardware interrupts for more precise (microsecond) timing.

Digital data collection is ideal for use with photogates, ultrasonic range-finders, quadrature-based angular position sensors, reed switches, Hall effect switches, contact switches, push-buttons, and any other sensors that produce a logic-level signal.

Direct Analog Measurements

The built-in 10-bit ADCs on the ATMega chips are limited in resolution to one part in 1024, or about 0.1% of the measurement range. This is still sufficient for useful measurements with MEMS accelerometers, light-level sensors, potentiometer-based angular measurements, analog Hall sensors, sound level meters, and other low-precision analog voltage measurements.

Indirect Measurements

The built-in serial and I2C communications of the Arduino allow communications with other instruments (serial) and chips (I2C) that can vastly increase the measurement capabilities of the device. Examples might include the ADS8344, with 8 channels of 16-bit ADC; or the LTC2485 ADC chip with *24-bit* resolution. There are I2C sensor chips manufactured now that can measure capacitance changes of a few femto-Farads, atmospheric pressure changes corresponding to a 20cm change in height, and just about anything else you can imagine.

And while it's collecting that data...

30kB of program memory can hold some pretty sophisticated software, and the Arduino is ideal for controlling servos and relays and other experiment-control hardware. It is quite possible to do things like use a quadrant photodiode and a pair of servos to track a light source while measuring the total intensity, or use software PID control and an H-bridge chip to drive a Peltier junction and control the temperature of an apparatus.

This is a complete computer, with memory comparable to some of the earliest personal computers, measurement capabilities greater than those early computers, and a mass of a couple of grams.

More information available:
Arduino website: http://arduino.cc/
Arduino/Freeduino index: http://www.freeduino.org/
My website: http://phys.csuchico.edu/~eayars/

What collection modes can be used

"Tethered" data collection

The "standard" Arduino has built-in USB/Serial capability. This allows you to connect the Arduino to a computer via the computer's USB port, and transfer the measurements to the computer via serial communications.

In this configuration, the $30 Arduino can duplicate all of the digital capabilities of (for example) a Vernier Software LabPro; but with 18* digital inputs. (Of course, you'd have to write your own software...)

The Arduino is very useful in this configuration as an I2C adaptor. Dedicated I2C-to-serial tranceivers are quite expensive, but the Arduino can perform the duty admirably with a very simple program. This capability opens up an enormous array of I2C sensors to use with LabVIEW or other serial-capable data-collection mechanisms.

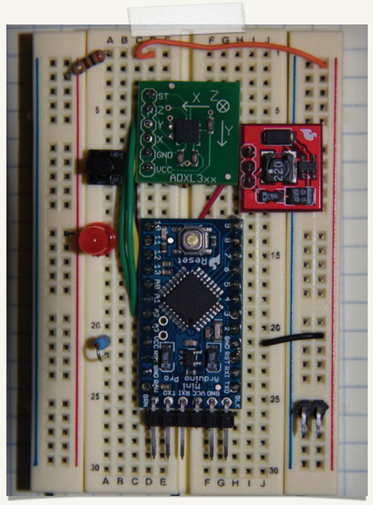
Wireless data collection

Using XBee transceivers, one can operate in "tethered" mode without the wire between the Arduino and the computer. With minimal configuration, a pair of XBees can act as a virtual serial cable with a range of up to 15 miles. One XBee is connected to the serial pins of the Arduino, the other is connected via a USB/Serial adaptor to the USB port of the computer.

Remote collection & storage

One of the most promising uses of the Arduino —particularly the Arduino Pro Mini— is as a remote datalogger. One can build a self-contained device consisting of a power supply, Arduino, any desired sensors, and a data storage mechanism such as a micro-SD memory card. This small and lightweight device could then be incorporated into experiments that would be otherwise be difficult to access, such as weather balloons, rockets, or submersibles. After recovery, the memory card can be removed and the data file read by a computer with an appropriate adaptor.

A prototype remote datalogger is shown here: this particular one measured acceleration on three axes, at 100 samples/second, with a resolution of 0.04 m/s/s.



Special thanks to the students of the CSUC SPS, without whose curiosity and enthusiasm I would never have gotten quite so involved in such interesting stuff.

*Two of the 20 possible digital lines would be needed for serial communications in this tethered configuration, leaving 18 for data collection.