CPEN 291

Computer Engineering Design Studio I

Lab 1: Arduino (Programming and Basic IO)

ECE – UBC

2016 W2

©Farshid Agharebparast

January 2017
Electrical and Computer Engineering
University of British Columbia

Objective

- This lab is on learning the Arduino Uno board, programming the Arduino board, basic analog and digital input/output (IO) interfacing, and external interrupts and AVR features.
  - Arduino Uno board
  - Arduino IDE
    - C programming in the Arduino integrated development environment (Processing)
  - IO:
    - Switches (digital input)
    - LED (digital output)
    - Photocell (analog input)
  - External Interrupts and AVR features
  - Fritzing software
  - Review of basic discrete electronic components' usage

Arduino

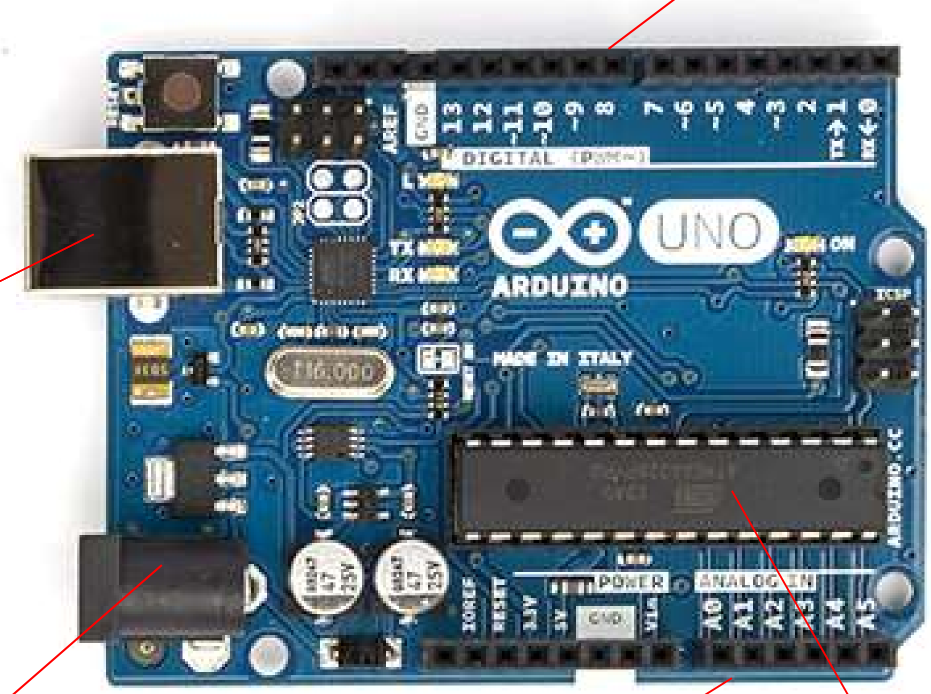
- Arduino is an open-source electronics platform . There are a number of different Arduino boards. We will use the very popular Arduino Uno Rev3 in this course.

Image source: Arduino.cc

- A great amount of info on Arduino can be found at http://arduino.cc. Carefully read: http://arduino.cc/en/Main/ArduinoBoardUno

Arduino UNO Rev3

Digital IO pins



USB

Image source: Arduino.cc

External power (needed if USB is not connected)

Atmel AVR microcontroller

Analog input pins

What else do you identify on the board? Familiarize yourself fully.

Arduino Uno and Atmel Microcontroller

- The Arduino Uno is a microcontroller board based on the ATmega328. It is a popular open-source design.

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 also provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

- You can find the datasheet for the microcontroller here: http://www.atmel.com/devices/ATMEGA328.aspx

Arduino Uno IO and Power

- The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.
- Digital IO: Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms.
- Analog IO: The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values) with analogRead() function. Digital pin 3, 5, 6, 9, 10, and 11 may also act as analog output providing 8-bit PWM output with the analogWrite() function.
- Arduino Uno in also well equipped with pins with specializes IO functions such as serial, interrupts, PWM, SPI, Reset, ….
- More info: http://arduino.cc/en/Main/ArduinoBoardUno

CAUTION!

Be very careful with the voltages you apply to the Arduino board or its shields. Note that Arduino Uno mainly operates at 5 volts.

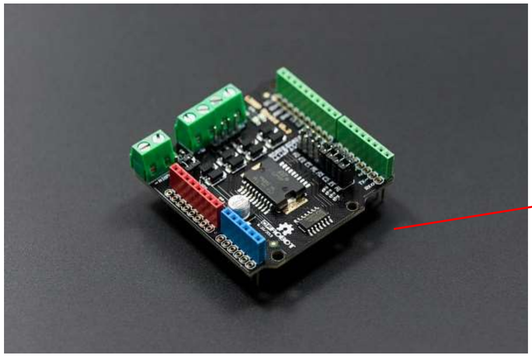
In general, be cautious and mindful when you connect or wire your circuit, and which pins your are using.

Also remember that Arduino saves the programs and run them as soon as it is powered.

If you burn out a component, you will need to purchase and replace it.

Arduino Shields and Modules

- Shields are boards that can be plugged on top of the Arduino board to extend its capabilities. There are large number of commercial shield.



Arduino Motor Shield

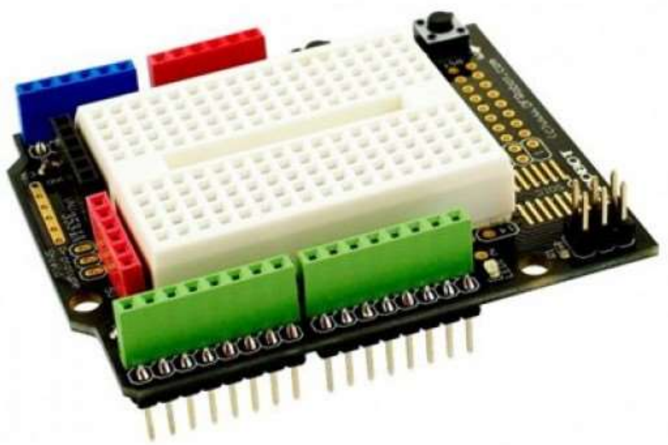
- There are also Arduino compatible modules. They are much smaller and limited.

Ultrasonic Range Finder Module

- We will use many different modules and shields in this course.

Arduino Proto-Shield

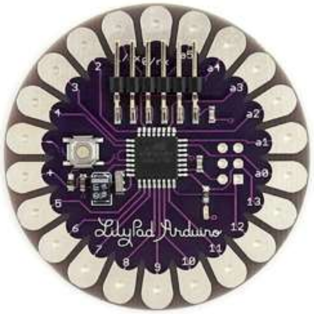
- The prototyping shield has great features and makes prototyping easier on your Arduino.
- It includes
  - a mini detached/attachable breadboard
  - Reset button
  - GND and +5V rails
  - DIP and SOIC prototyping areas
  - Extra button, …



- It will provide a compact and convenient working area for the experiments. If you need more room, then use a larger breadboard.

Other Arduino Boards

- There is a rather large number of Arduino boards for different applications. Nevertheless they are all programmed similarly. See http://arduino.cc/en/main/boards for the complete list.
- Arduio Uno
- Arduino Mega
- Arduino Nano
- Arduino Due (ARM Based)
- Arduino Yun
- LilyPad Arduino
- …



Images source: Arduino.cc

- See a comparison here: http://arduino.cc/en/Products.Compare

Alternative Open-Source Boards

- In addition to Arduino, there are a few other types of open-source microcontroller boards including:
  - Raspberry Pi: Single board Linux computer with video and GPIO ports

    We will Pi in this course.

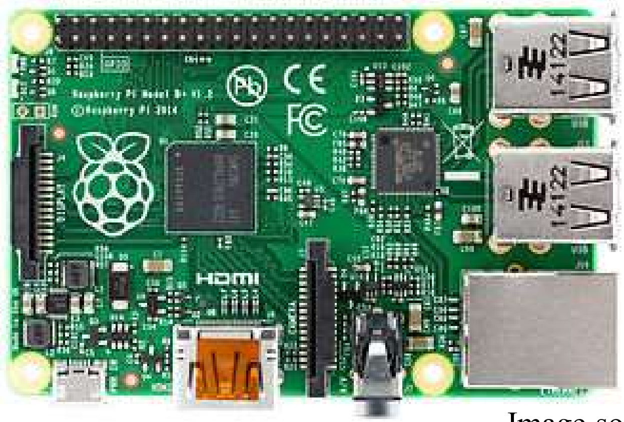


Image source: wikipedia

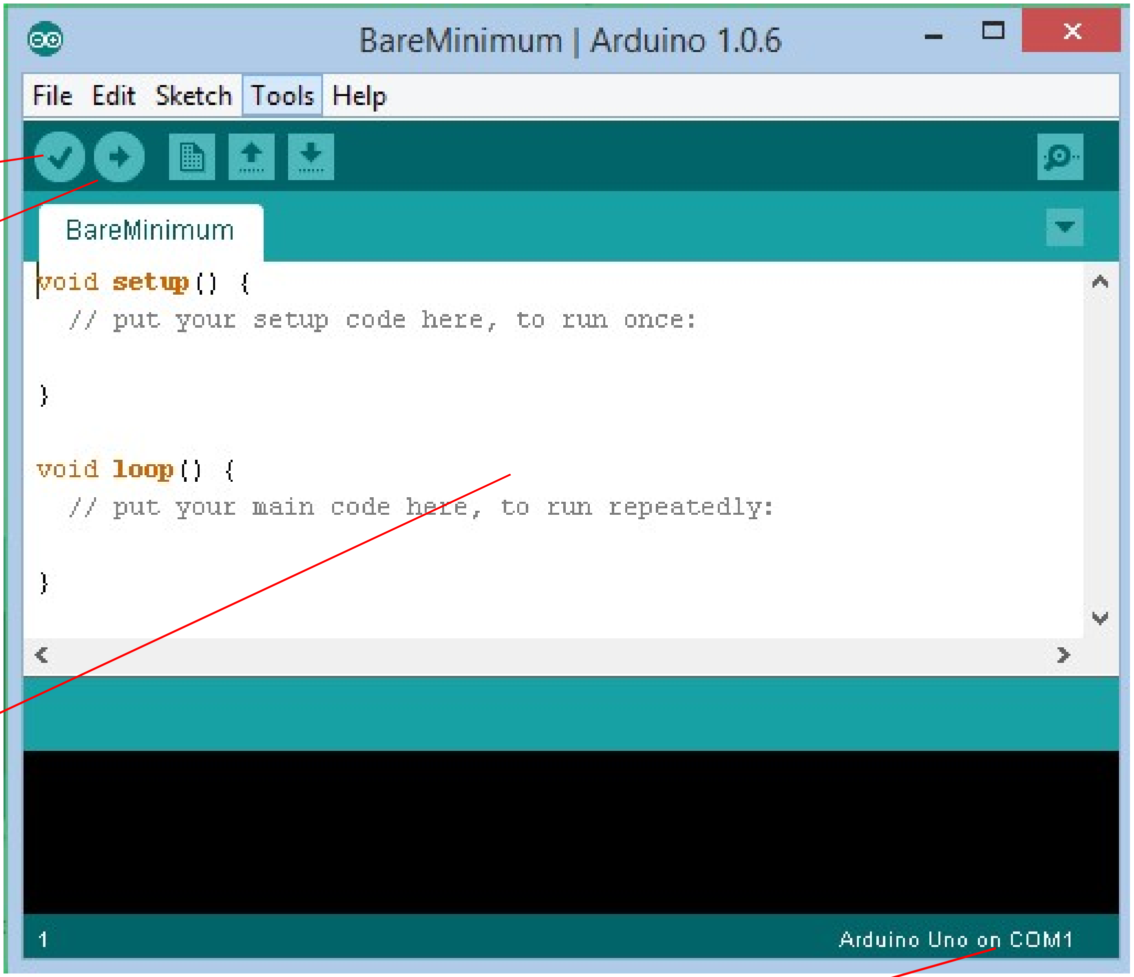
  - Beagle Bone: TI's ARM-based Linux board
  - Netduino: Open-source microcontroller programmed using .NET/C#
  - …

Arduino Software

- Programming Arduino is very straightforward. There is a free and open-source Integrated Development Environment (IDE) for all Arduino boards.
  - You will connect the Arduino board to a computer via USB and then use the open-source free Arduino IDE to write, edit, compile and finally upload the programs to the Arduino board.
- The software is installed on the lab computers but it is a good idea you download and install it on your own laptop as well.
- You will mainly use the C programming language to program Arduino. The alternative is to also use inline assembly.
- More info: http://arduino.cc/en/Main/Software

Arduino IDE

The Ardunio IDE includes all you need to write, compile and load the program to an Arduino and test it, such as: a text editor to write the code, compiler and loader, serial monitor, example programs, ...



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

1 Arduino Uno on COM1

Verify/Compile

Upload

C programing (A bare minimum program skeleton is shown)

It is important to be on the correct COM port

Coding

- You will be using C programming language extensively in this course. There will be a quiz during the first lab on C programming. Make sure you refresh your programming skills.
- Arduino program are essentially C programs with a slight modification.
  - There is no main() function. There are two primary functions in all Arduino programs: setup() and loop()
  - setup() is executed once and include all one time setup and initialization instructions. For example in the setup() function you may set the input or output functionality of the pins you use.
  - loop() is a function that acts like a superloop for the programs. The function will be execution repeatedly as long as the UNO is powered up.
  - Other than the above, every program is essentially a C program. You can add as many more functions as you need.

Arduino Functions for Lab 1

- ❑ setup(): http://arduino.cc/en/Reference/Setup
- ❑ loop(): http://arduino.cc/en/Reference/loop

IO functions (you may remember these from APSC 160 DAQ functions. Their functionalities are identical.)

- ❑ digitalRead(): http://arduino.cc/en/Reference/digitalRead
- ❑ digitalWrite(): http://arduino.cc/en/Reference/digitalWrite
- ❑ analogRead(): http://arduino.cc/en/Reference/analogRead
- ❑ analogWrite(): http://arduino.cc/en/Reference/analogWrite

Also

- ❑ serial.begin(): http://arduino.cc/en/pmwiki.php?n=Serial/Begin
- ❑ delay(): http://arduino.cc/en/reference/delay

External Interrupt and AVR Features

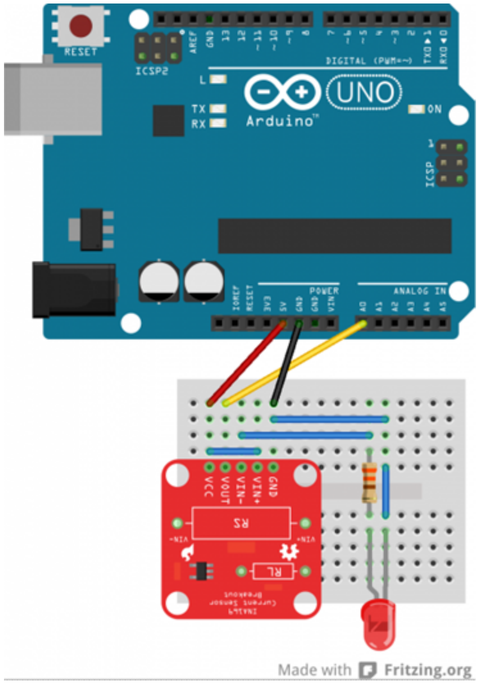
- External interrupt:
  - In this lab you will also use external interrupts in a quite basic setting.
  - You can easily attach a certain pin to a specific interrupt. See http://arduino.cc/en/Reference/AttachInterrupt.
  - For more info see: http://playground.arduino.cc/Code/Interrupts

- As a part of this lab, I will be asking you to look into the AVR features. The knowledge of these features and their usage may make your Arduino code more efficient.
  - Example of such features are: port manipulation, in-line assembly, timers, …
  - See: http://playground.arduino.cc/Main/AVR

Fritzing and Other Software

- ❑ We will use the Fritzing software for the hardware schematics.

Fritzing (http://fritzing.org/download/) is open-source free software and is specially Arduino compatible, so you can not only create the usual circuit schematics, but you can also clearly draw and display hardware connections and wiring, and create the PCB layout for prototyping or manufacturing .

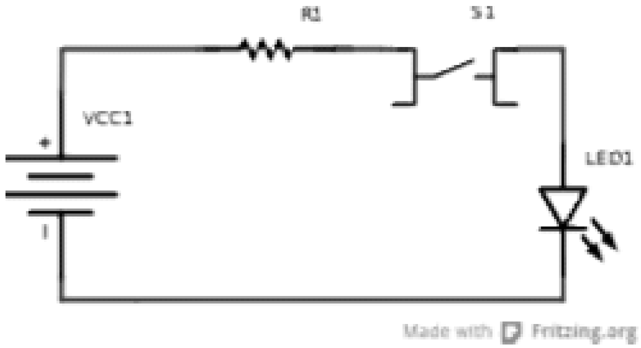


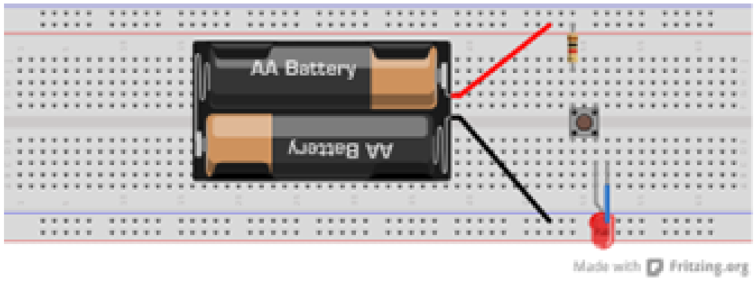
- ❑ Fritzing intro:
  https://www.youtube.com/watch?feature=player_embedded&v=Hxhd4HKrWpg

- ❑ We will also use other software such as: Processing, Matlab (octave), multisim, …

Fritzing Example

- ❑ As an example, the following two figures show a circuit diagram of a simple circuit and the breadboard view of the same circuit drawn with Fritzing.

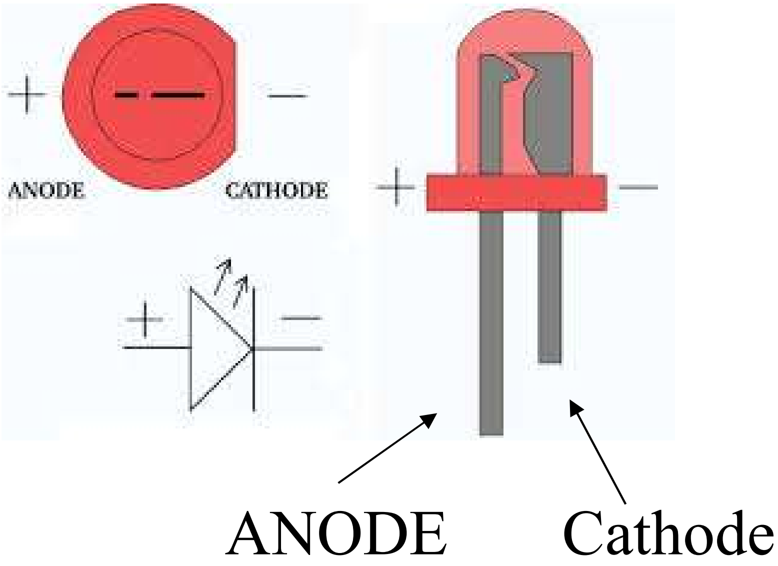




Images source: Wikipedia

IO: LED, Switch, Photocell

- Three basic IO devices are used in lab 1:
  - Switch (toggle switch and push button) as digital input
  - LED as digital output: You have already seen how to interface with an LED in APSC 160. Here in addition, you need to physically connect the LED and program for the Arduino platform.
    - The schematic below shows the two terminals of an LED: anode and cathode.
    - Always protect an LED with a resistor in series with it.



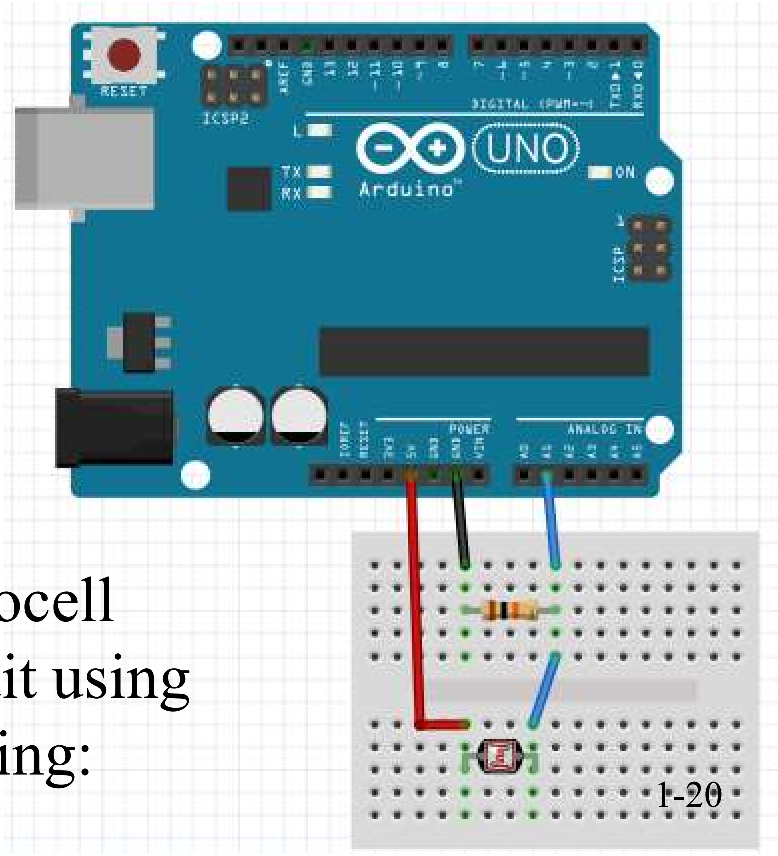
IO: LED, Switch, Photocell

- LED (cont.):
  - Turning LED on: apply 5 Volts across the resistor and LED.
  - Turning LED off: apply 0 Volts across the resistor and LED.
  - In most cases: HIGH = 5V, LOW = 0V

- Photocell as analog input: A photocell would allow us to sense the level of ambient light.

Photocell:

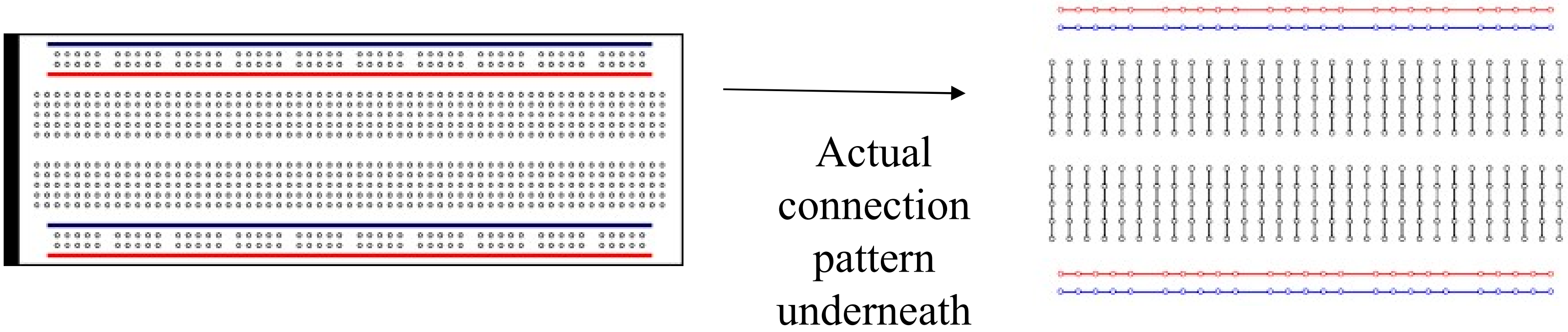
Photocell circuit using Fritzing:



Tools and Components: Breadboard

- ❑ You have learned/will learn more about these concept in ELEC201/EECE251 but for the sake of this experiment you need to have a basic familiarity with breadboards and resistor colour coding.

- ❑ Breadboard: A breadboard (or protoboard) is a construction base for prototyping of electronics.



  - ❖ On how to use a breadboard see: https://learn.sparkfun.com/tutorials/how-to-use-a-breadboard/anatomy-of-a-breadboard

Tools and Components: Resistors

- Resistor Colour coding: four bands of colours are used on a resistor to indicate its value. Each colour has a fixed meaning.

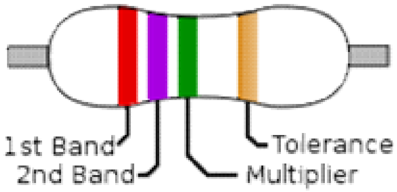


Image source: wikipedia

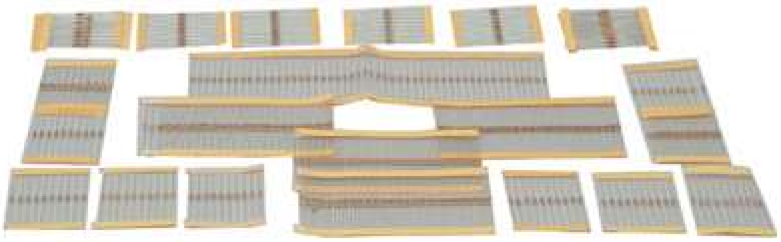
  - See: http://en.wikipedia.org/wiki/Electronic_color_code
  - Use an online tool to help you determine the resistor value, such as: http://www.digikey.ca/en/resources/conversion-calculators/conversion-calculator-resistor-color-code-4-band

- Note: you will need to identify resistor values for this experiment and that can be done with the help of the tool mentioned above or TAs.

More on Resistors

- ❑ There is a resistor sub-kit containing tapes of 30 different resistor values in the lab kit.

540 pieces kit
30 values
10 or 30 for each value



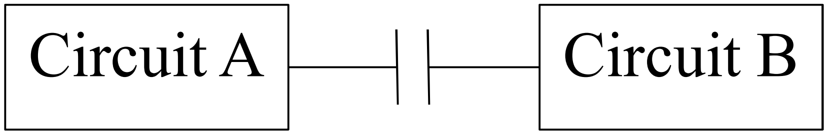
- ❑ The resistors provided in the kit are all 1/4 W. This is the maximum allowed power dissipation. You always want to make sure the resistor always operates well within its recommended power rating. Recall that:

$$P = I^2 R = IV = \frac{V^2}{R}$$

- ❑ Practice/review/learn how to read resistor colour codes, so that you can identify the value of any resistor.

Some Notes on Capacitors

- ❑ From physics and circuit analysis courses, recall that capacitors can store electrical energy when connected to a charging circuit, and when disconnected it can dissipate that stored energy. Capacitor have many applications: bypassing, signal coupling, filters, …
- ❑ Two of the most common and also intuitive electronics applications of capacitors are bypassing and coupling.
- ❑ Coupling (also know as Blocking): since a capacitor is open-circuit at DC, a capacitor can be used to couple a varying signal while blocking the its average DC level.

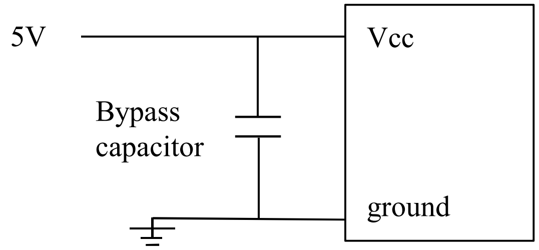


A blocking or coupling capacitor

Reference: *Art of Electronics*

More Notes on Capacitors

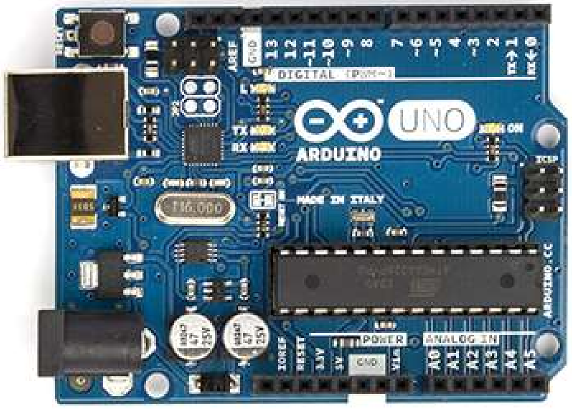
- Bypassing (also know as decoupling): Since a capacitor is short-circuit to “relative” high frequencies, it supresses (bypasses) the signal we do not want, such as noise.
- A very common use of bypass capacitors is to stabilize the power delivered to an IC or a circuit.
  - A bypass capacitor is added as close as possible, between a circuit’s power and ground, or even for each individual IC’s.



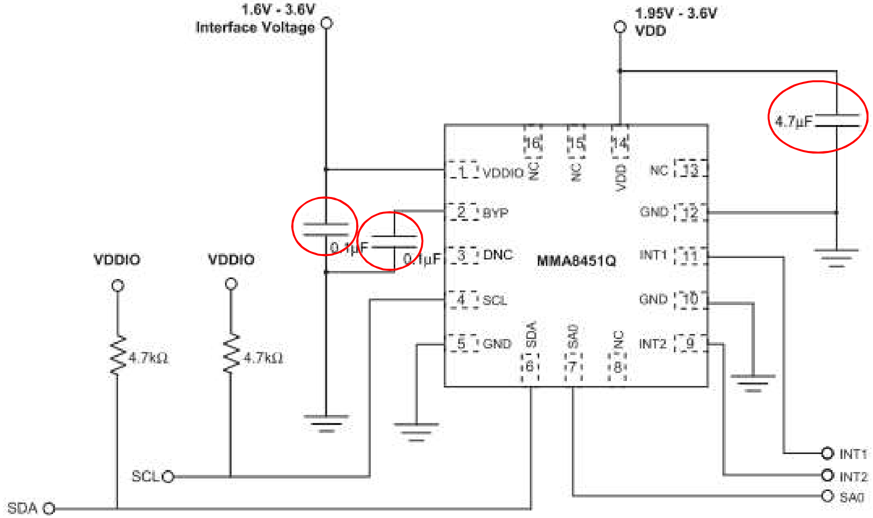
- Use the provided ceramic capacitors (un-polarized).
- Do no use electrolytic capacitors unless necessary and very carefully. If hooked incorrectly, one may explode.

Capacitor Examples

- Examine an Arduino UNO board. Bypass capacitors can be seen on the schematic.

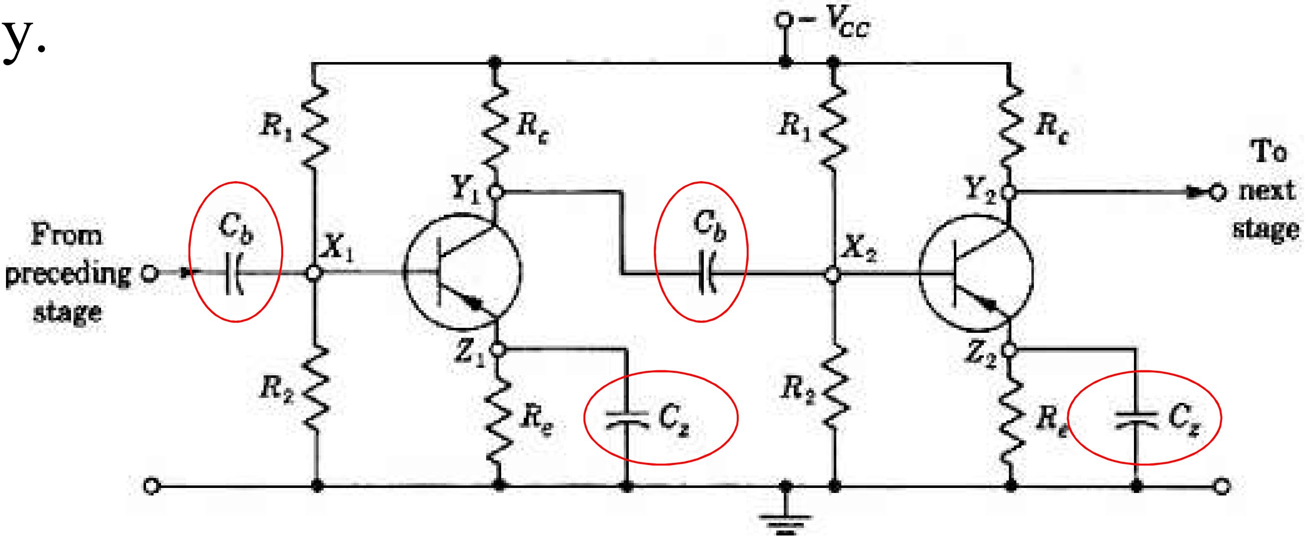


- The following circuit is from the MMA8451Q accelerometer chip datasheet. You can see three suggested bypass capacitors in this circuit.



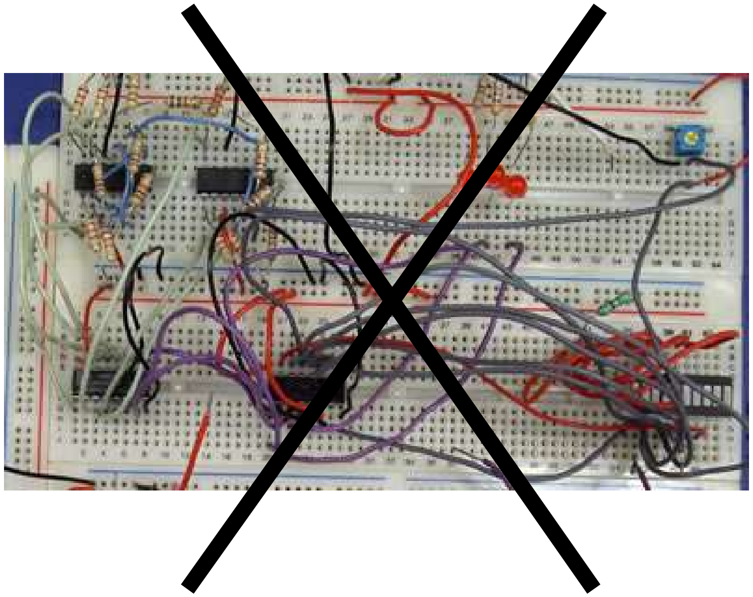
Capacitor Examples 2

- The following circuit is a classic two-stage transistor amplifier circuit. You can see 2 bypass capacitors (*Cz*) and 2 coupling capacitors (*Cb*) on the circuit. (Do not worry about the circuit itself. You will thoroughly examine such a circuit in electronics courses.)
- The bypass capacitors here are for another purpose, to bypass the emitter resistances at the AC, not for stabilizing the power supply.

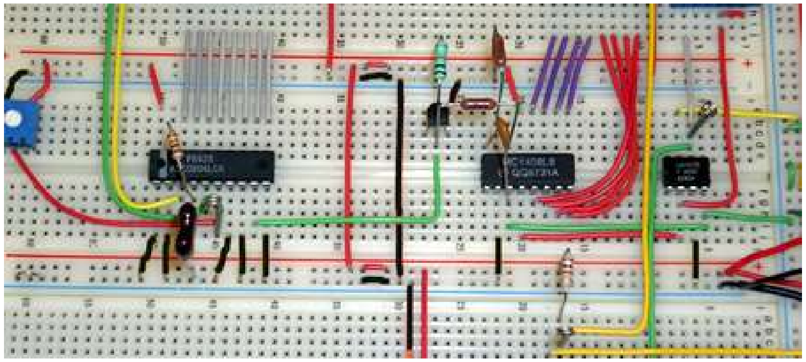


Wiring Breadboards

- ❑ Use jumper wires of correct sizes and proper colours. Do not poorly wire circuit.



Poorly wired circuit (spaghetti)



Much better wired circuit

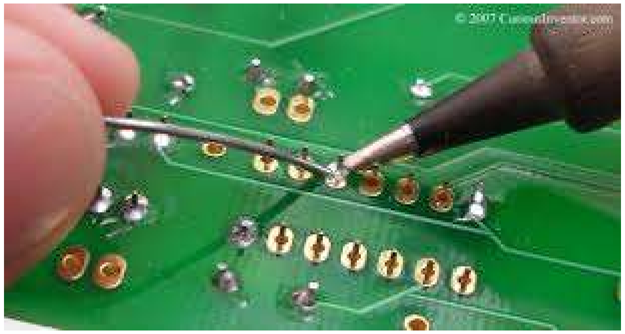
- ❑ You can find many other good examples.

Datasheets

- Datasheets are our primary source of information for any electronic device.
  - “A **datasheet**, **data sheet**, or **spec sheet** is a document that summarizes the performance and other technical characteristics of a product, machine, component (e.g., an electronics component), material, a subsystem (e.g., a power supply) or software in sufficient detail to be used by a design engineer to integrate the component into a system.”
- You must be in the habit of reading the datasheet carefully before using any component.
- In particular, a datasheet is the source for the following info:
  - Exact purpose and application
  - Pin-out or layout of the component
  - Operating voltage, power rating, operating temperature, frequency range, input/output resistance, …
  - Sample application notes and circuits

Soldering

- Soldering is the process of joining components to a board by melting and flowing a filter metal (solder) into the joint.



Soldering station

- You do not need any soldering for lab 1.
  - In general, if you need to solder, get the consent of the lab engineer and follow all safety guidelines (e.g. wearing safety glasses in mandatory for you and everybody around you).
- Here is a guide for soldering: https://learn.adafruit.com/adafruit-guide-excellent-soldering/preparation

Lab1 Experiment and Timeline

- ❑ Lab info: See Connect for the instruction on the lab experiment
  - ❖ L2A and L2B: Lab1 starts on Wed (Jan 11) and the demos are on Fri (Jan 13)
- ❑ Lab deliverables

  1) Prelab (just usual pre-reading and preparation)

  2) Lab report, code and schematic submission

     - Submit by the deadline. No late submission is allowed.

  3) Demo:

     - There will be a grade 30% penalty if you miss your allocated demo time in this lab. It must be rescheduled asap within the same lab time.

References

- There is a great amount of good information on Arduino and related software on the Internet.
  - Arduino: http://arduino.cc/
  - Fritzing: http://fritzing.org/
  - Arduino Getting Started tutorial: http://arduino.cc/en/Guide/HomePage
    Arduino Reference: http://arduino.cc/en/Reference/HomePage

    Fritzing reference: http://fritzing.org/learning/full_reference/
  - There are also many good books …
- Datasheets:
  - See the datasheets posted on Connect