GNG2101 Lab Exam Review

Circuits

LEDs (Light Emitting Diode)

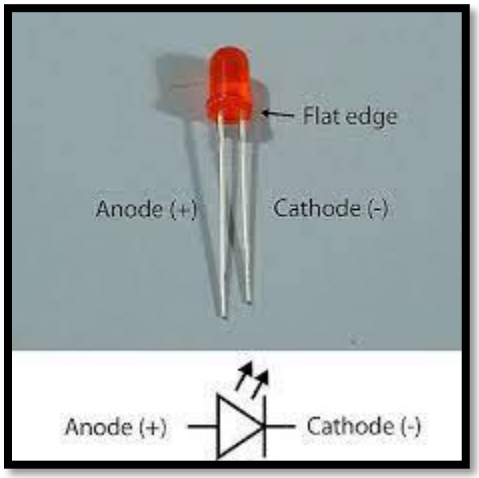

Figure 1: A Red LED

1. Current can only flow from anode to cathode (diodes are unidirectional)
2. When current is flowing through a diode, there is a constant voltage drop across the diode regardless of the current (0.7V for typical diode, 2V for red LED, 3V for blue LED), can be modeled as a constant voltage source in this case
3. If you put more than 20 to 30 mA of current through a LED, smoke will come out of it and it will smell bad

Voltage Sources and Batteries

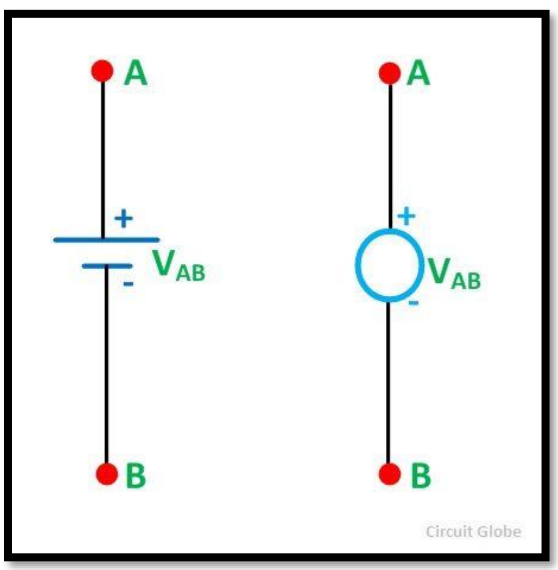

Figure 2: Battery (left), Voltage Source (right)

1. Produce a constant or known voltage
2. Current through the source depends on the circuit attached to the source
3. Can be connected in series, when this is done, the overall voltage is the sum of the individual voltages of each source
4. A battery is a type of voltage source
5. When sources (with different voltages) are connected in parallel, smoke comes out of the sources

Resistors

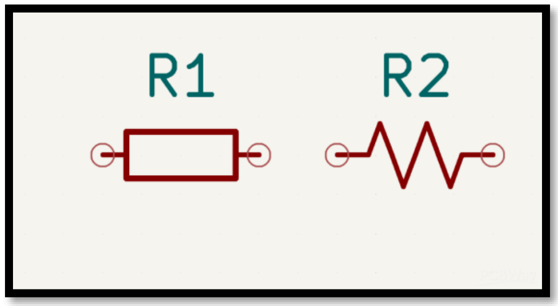

Figure 3: Resistors, EU symbol on the left and US symbol on the right

1. Typically can only handle 0.25W max, when this is exceeded smoke comes out
2. Obeys ohms law (at low frequencies/ in this class) $V = IR$
3. Resistors in series add, resistors in parallel can be combined as follows: $R_{eq} = \frac{R_1 R_2}{R_1 + R_2}$, for 2 resistors, apply this recursively for more than 2 resistors

Motors

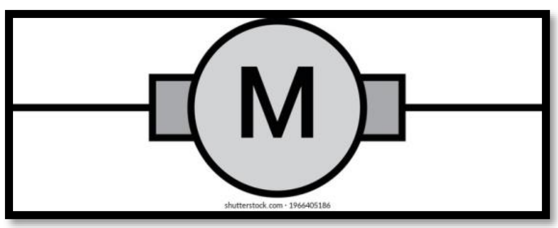

Figure 4: Typical Single Phase DC Motor

1. A piece of wire wrapped around something in a special way with a magnetic field present
2. Really just an inductor, which means when there is a change in the motor current, the motor produces a voltage spike which can destroy a connected circuit if designed incorrectly

Capacitors and Inductors

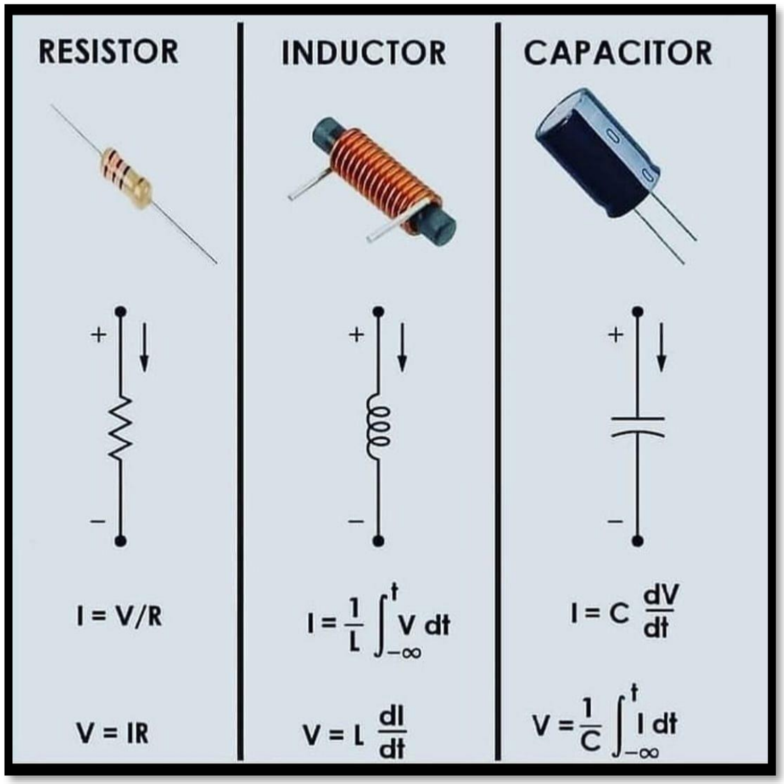

Figure 5: RLC Components

	Capacitors	Inductors
Differential i-v	$i = C\frac{dv}{dt}$	$v = L\frac{di}{dt}$
Integral i-v	$v_C(t) = \frac{1}{C}\int_{-\infty}^{t} i_C(\tau)\,d\tau$	$i_C(t) = \frac{1}{L}\int_{-\infty}^{t} v_L(\tau)\,d\tau$
DC equivalent	Open-circuit	Short-circuit
Two in series	$C_{eq} = \frac{C_1C_2}{C_1+C_2}$	$L_{eq} = L_1 + L_2$
Two in parallel	$C_{eq} = C_1 + C_2$	$L_{eq} = \frac{L_1L_2}{L_1+L_2}$
Stored energy	$W_C = \frac{1}{2}Cv_C^2$	$W_L = \frac{1}{2}Li_L^2$

Figure 6: Capacitor and Inductor Formulas

1. An inductor resists changes in current, a capacitor resists changes in voltage
2. An inductor is just a piece of wire wrapped around something (usually Fe containing)
3. A capacitor is 2 parallel conductors with an insulator in the middle
4. A capacitor stores energy in its electric field, the inductor stores energy in its magnetic field

A Switch

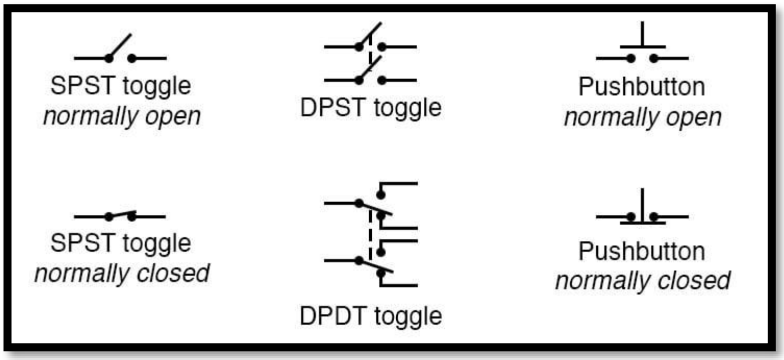

Figure 7: Various Kinds of Switches

1. Used to disconnect a portion of a circuit
2. Can sometimes be found on battery compartments/ holders, although this can be expensive
3. A pain in the ass to analyze when doing circuit analysis

Breadboarding

It is useful to be able to convert a picture of a breadboard into a schematic. The following is harder than you would see on a lab exam for GNG2101

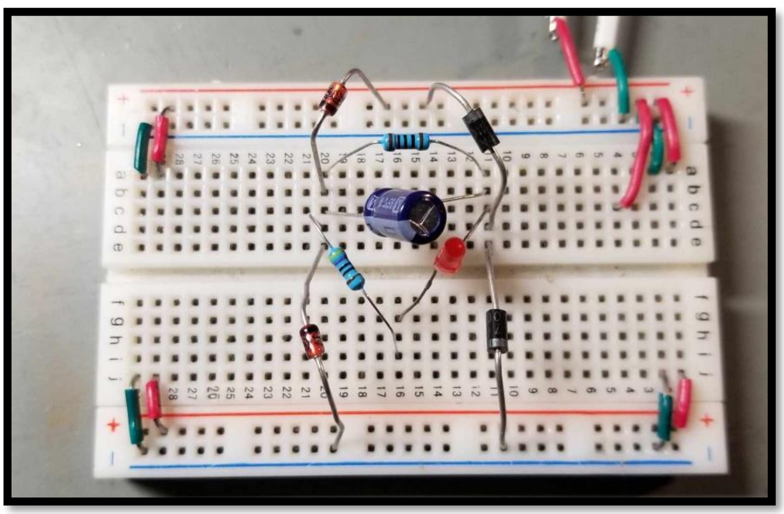

Figure 8: Some Circuit

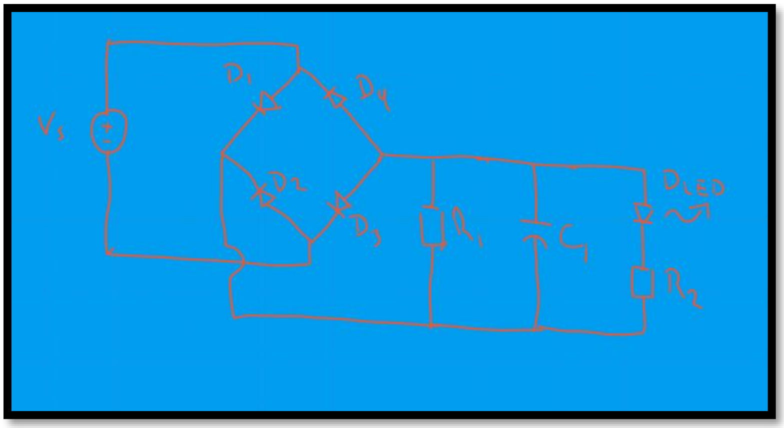

Figure 9: The Same Circuit, but drawn as a circuit diagram

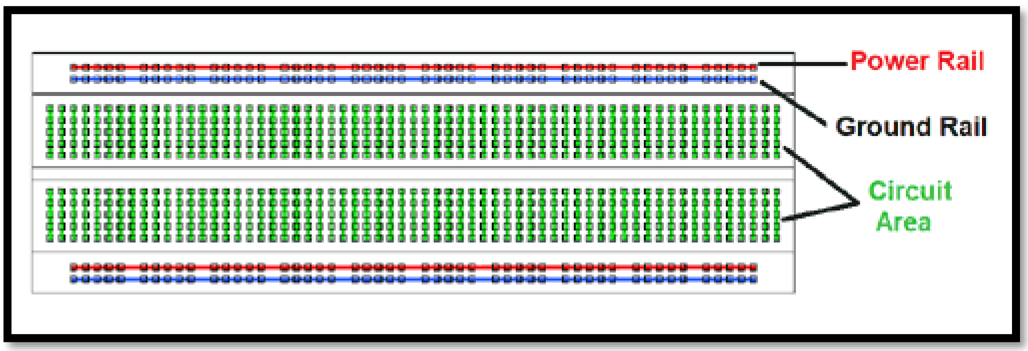

Figure 10: Breadboard Connections

Formulas

$V = IR \quad P = IV$

$$Battery\ Capacity\ (Ahr) = Current\ (A) * time\ (hr)$$

$$Energy\ Stored\ in\ Battery\ (Whr) = Battery\ Capacity\ (Ahr) * Battery\ Voltage\ (V)$$

V: voltage (Volts), I: Current (Amps), R: Resistance (Ohms), W: Power (Watts)

Note: The voltage of a battery decreases as it is discharged, that means the equation above for energy in a battery is only an approximation. To actually calculate the energy in a battery, integrate the voltage*current as the battery discharges

$$R_{currentLimiting} = \frac{V_{supplu} - V_{led}}{I_{led}}$$

Ex: if I have a 26 Ahr battery, I can get 26 A out of it for 1 hr, or 13 A out of it for 2 hr, or 8.7 A out of it for 3 hours

Ex: If I have a 2V LED, 12V supply, and I want 10 mA through the LED, the current limiting resistor value should be: $R_{currentLimiting} = \frac{12-2}{0.01} = 1k\Omega$

Soldering

Things to remember:

1. The end of the soldering iron is hot. Do not touch it and burn yourself
2. Keep the iron temperature at 340°C. If needed, bump it up to 370°C, but the lower the temperature the better (less likely to lift pads)
3. Pads are the metal part of the PCB or protoboard that surrounds the hole that you stick a component in
4. Don't add too much or too little solder (see image below)
5. Don't remove the iron too quickly, or you'll get a cold joint (see below)
6. Clean the soldering iron before you put it down on the steel sponge or a wetted sponge
7. Do not touch the iron to anything that isn't metal
8. When soldering, bring the iron in with one hand to heat up the pad and the component
9. Then bring in the solder with your other hand and touch the joint with the solder (it should melt)
10. Bring both the solder and the iron away from the joint
11. It can be helpful to coat the iron in solder before initially touching it to the joint to heat it

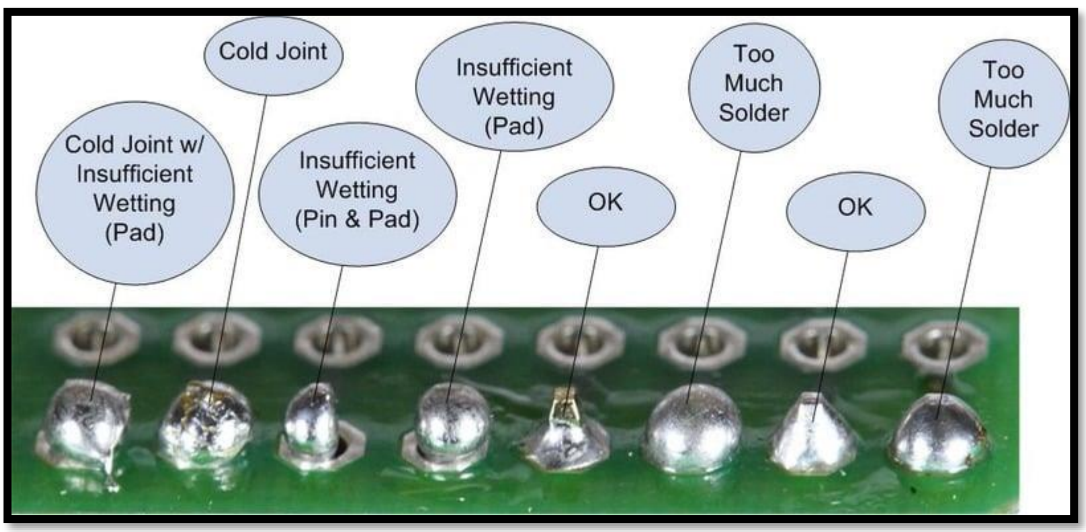

Figure 11: Various Solder Joints

Arduino Programming

1. **Setup and Loop Functions**:
    - **setup()**: Runs once at the start, used to initialize variables, pin modes, or libraries.
    - **loop()**: Runs continuously after **setup()**, where the main code is executed repeatedly (in an infinite while loop).
2. **pinMode()**:
    - Used to configure a specific pin as either INPUT or OUTPUT.
    - Syntax: **pinMode(pin, mode);**

- mode can either be INPUT or OUTPUT (sometimes INPUT_PULLUP or INPUT_PULLDOWN), pin is the pin number
- Ex: pinMode(13, OUTPUT)

3. **digitalWrite()**:
   - Writes a HIGH or LOW value to a digital pin.
   - Useful for turning on/off an LED, relay, etc.
   - Syntax: **digitalWrite(pin, value);**
4. **digitalRead()**:
   - Reads the value from a specified digital pin, either HIGH or LOW.
   - Used for reading the state of a digital input like a button.
   - Syntax: **digitalRead(pin);**
5. **analogRead()**:
   - Reads the value from an analog pin and returns a number between 0 and 1023.
   - Useful for reading sensors that give variable voltage like a temperature sensor.
   - Syntax: int x = **analogRead(pin);**
6. **analogWrite()**:
   - Outputs a variable voltage (via PWM) on a pin, value between 0 and 255.
   - Used for controlling the brightness of an LED or the speed of a motor.
   - Syntax: **analogWrite(pin, value);**
7. **if/else Statements**:
   - Used for making decisions based on conditions.
   - Syntax: **if (condition) { /* code to execute if condition is true */ } else { /* code to execute if condition is false */ }**
8. **for Loop**:
   - A loop that runs a specific number of times, useful for repetitive tasks.
   - Syntax: **for (initialization; condition; increment) { /* code to execute */ }**
9. **Serial Communication**:
   - Used for communication between the Arduino and a computer or other devices.
   - **Serial.begin(speed);** to initialize serial communication. speed is the baud rate (ex 9600)

- **Serial.print()** or **Serial.println()** to send data.

10. **AFMotor Library (Adafruit Motor Shield):**
    - Designed for controlling motors using the Adafruit Motor Shield.
    - AF_DCMotor motor(number): Create an instance of a DC motor. The 'number' is the motor number (1-4) on the shield.
    - motor.setSpeed(speed): Set the speed of the motor (0-255).
    - motor.run(command): Control the motor's operation. command can be FORWARD, BACKWARD, RELEASE, etc.
    - Also supports controlling stepper motors with functions like AF_Stepper(steps, number) and stepper.step(steps, direction, style).
11. **Bluetooth Library (HC-05 Module):**
    - Used for wireless communication via Bluetooth.
    - begin(speed): Initialize the serial communication with the Bluetooth module at a specified baud rate.
    - print(data)/println(data): Send data to the connected Bluetooth device.
    - available(): Check if there is data available to read from the Bluetooth module.
    - read(): Read incoming data from the Bluetooth module.
    - Note: The specific functions may vary slightly depending on the library used for the Bluetooth module.

Mill/ Lathe

Things to remember

1. Doing something on a mill/ lathe usually takes time
2. The direction of rotation of the tool or workpiece matters
3. The speed at which the tool or workpiece matters
    a. $RPM = \frac{4*CS}{D}$, all in imperial
4. Locking any unused axis is a good idea
5. Tying loose hair or clothing back is important
6. Safety glasses
7. Tool/ workpiece needs to be in the center of the chuck
8. Where the tool is in relation to the workpiece matters
9. Proper workpiece holding is important