Chip Limeburner

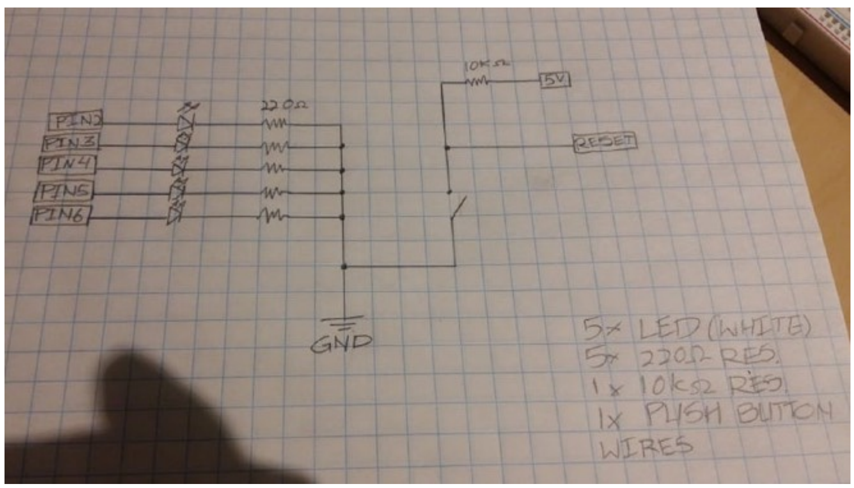
Student ID: 40177255

CART 360

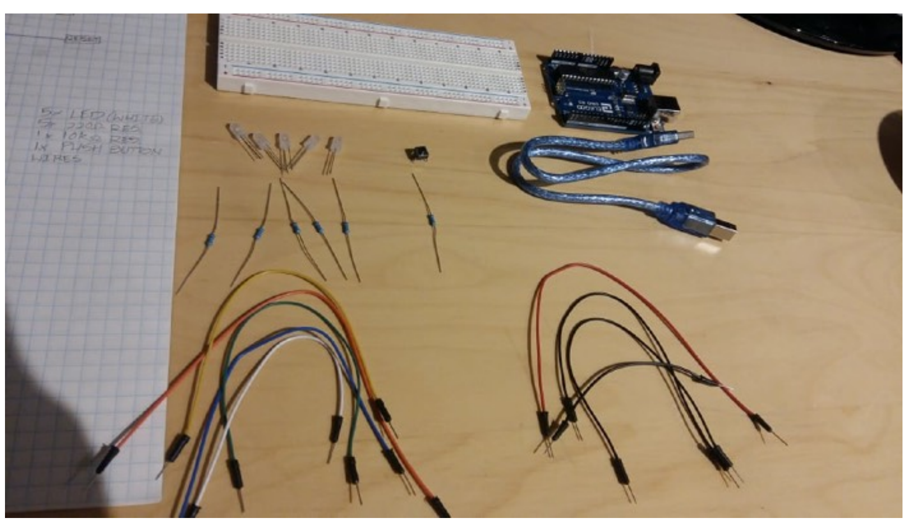
Oct. 1, 2021

Etude 1

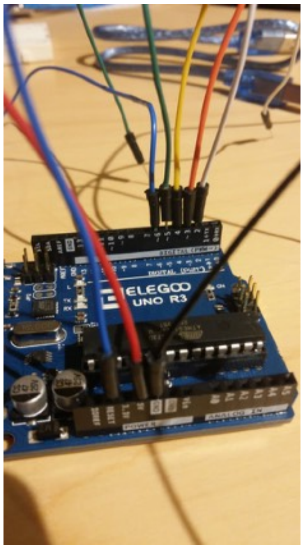
1) The first step I took to assembling the circuit was to re-draw it using standard diagram notation. This was simply a personal preference given my background in STEM. This also gave me the opportunity to double-check resistor values, and create an itemized list of the main components I would need.

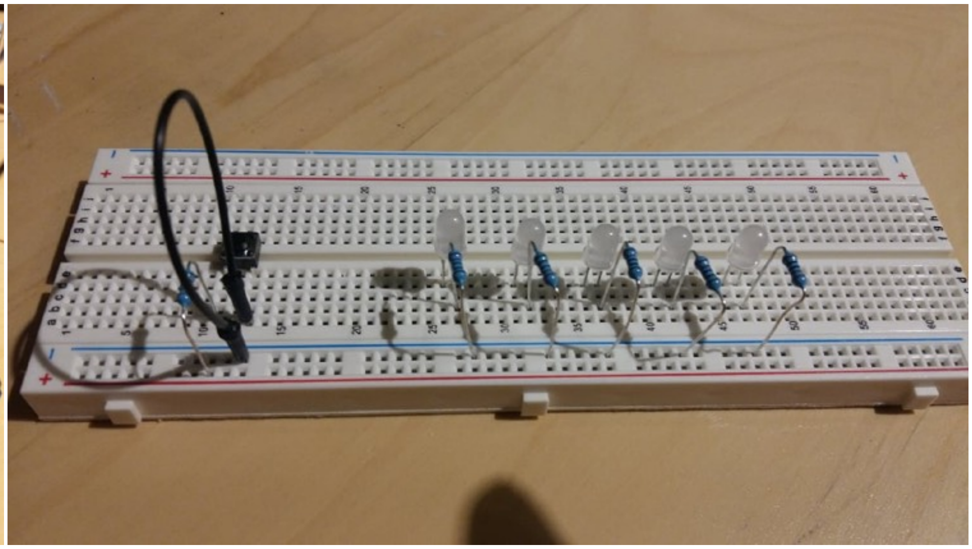


The Next step I took was to assemble all the components I'd previously listed along with the breadboard/Arduino/cables etc. As I laid them out, I grouped them roughly by the "sub-circuit" they belonged to, either the reset circuit or the LED circuit.

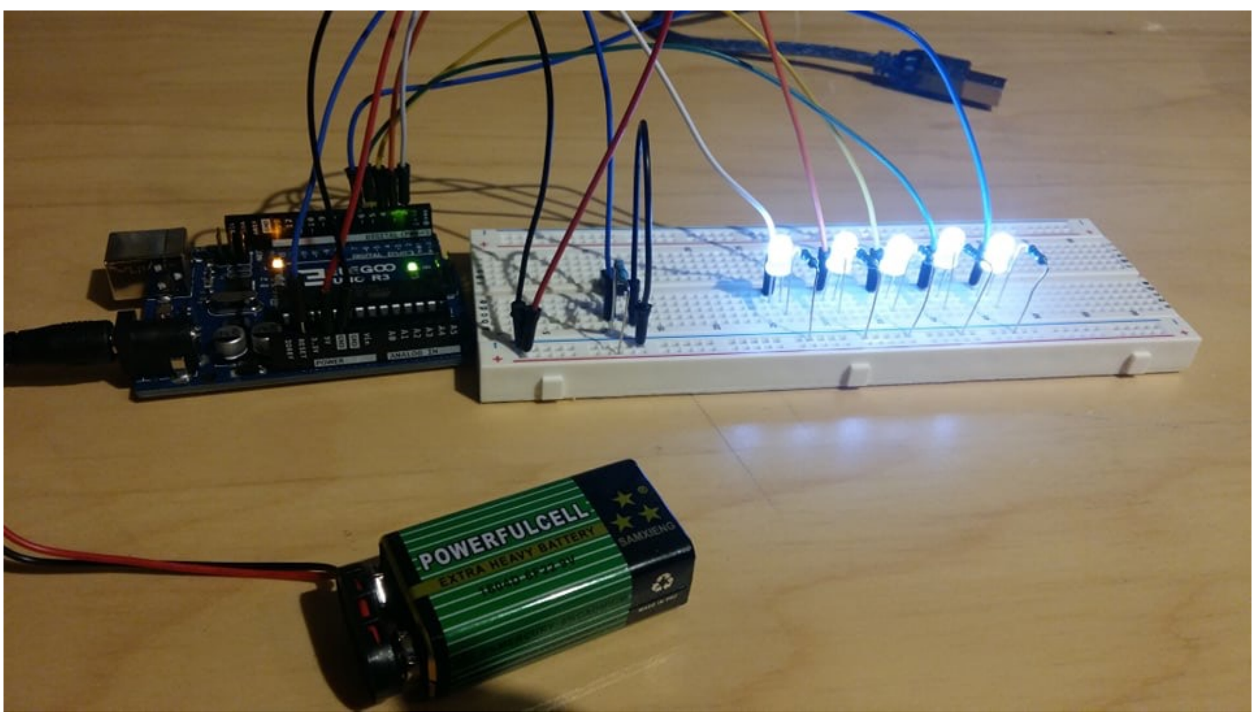


I then proceeded to assemble the components on the breadboard and separately attach the wires I anticipated needing for the Arduino board. Once both these steps were complete, I took a moment to review component placement to ensure they were properly connected.

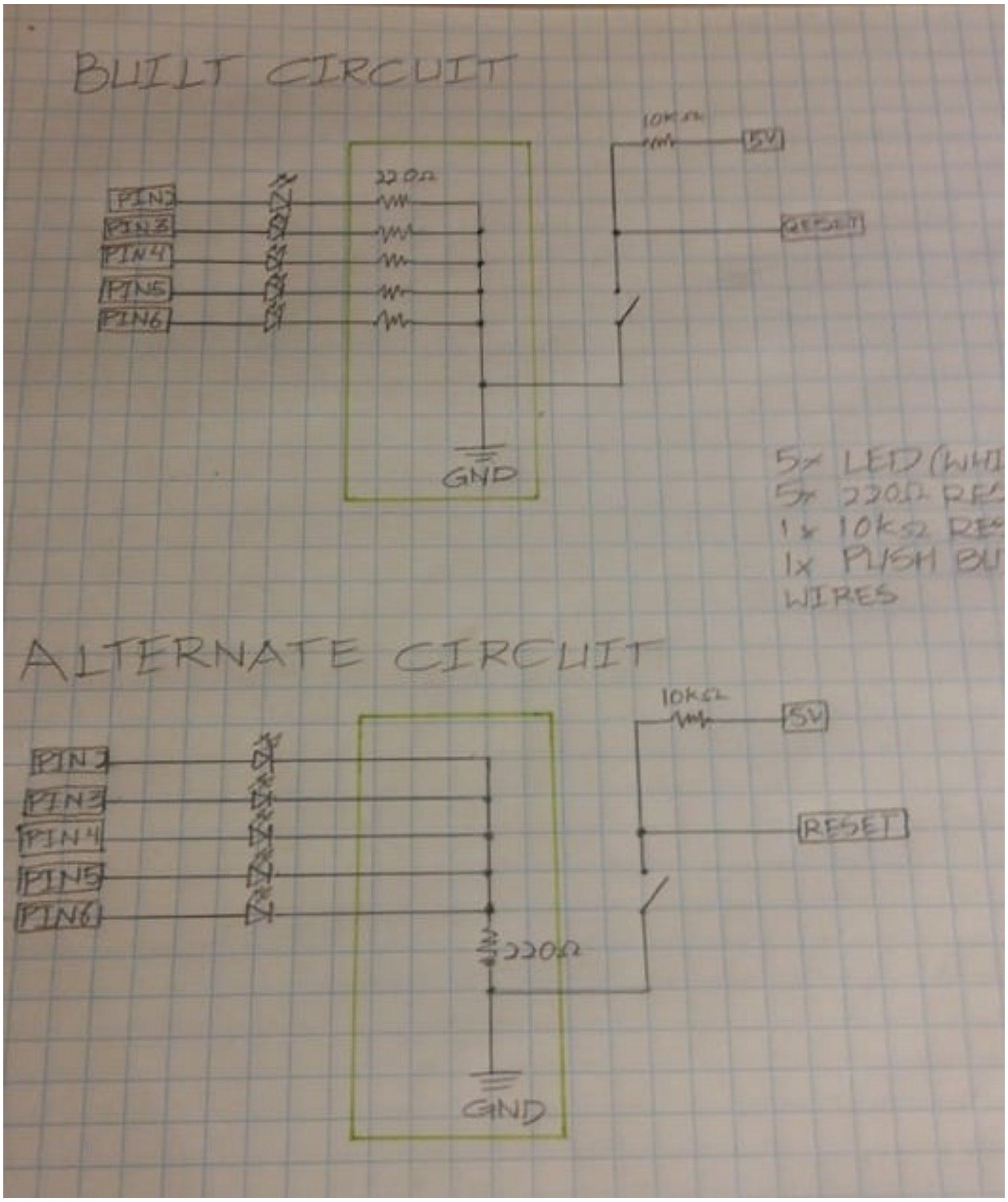




With the Arduino board and breadboard separately in order, I connected the two, making sure to connect the correct wire to the appropriate part of the circuit. After a final review of the connections, I connected the Arduino to my computer, uploaded the code to it, then swapped the power supply for the 9V battery to make the whole device portable.



2) The difference between the two circuits lies entirely in the regions highlighted below:



In the “built” circuit, each LED is in series with its own current-limiting resistor, whereas in the “alternate” circuit, the five LEDs share a common current-limiting resistor. I would expect the “built" circuit to be the more reliable one, for the following reason.

If we carry out some cursory circuit analysis on the “built” circuit, we know by Kirchhoff’s voltage law that the net voltage from a pin to ground is going to be zero. Thus:

(1) $$V_{pin} = V_{drop-LED} + V_{drop-resistor}$$

By Kirchhoff's current law, we also know that the current across any given pair of LED and resistor will be the same. Using Ohm's law and this observation, the above equation becomes:

(2a) $$V_{pin-n} = IR_n + I220$$

(2b) $$V_{pin-n} = I(R_n + 220)$$

For a given LED n in the "built" circuit. What's notable about this is that V_{pin-n} is a discrete value (0V or 5V), and so if a pin is HIGH, then the voltage and current supplied to its respective LED will be consistent, regardless of the action of the other pins.

On the other hand, in the "alternate" circuit Kirchhoff's voltage law tells us again that the voltage drop across any of the LEDs and then the resistor must equal the input voltage, so similarly:

(3a) $$V_{pin-n} = V_{drop-LED-n} + V_{drop-resistor}$$

(3b) $$V_{drop-LED-n} = V_{pin-n} - V_{drop-resistor}$$

For pin and LED n, but at the same time, Kirchhoff's current law tells us the net current coming all the LEDs must equal the current entering the common resistor[1], or:

(4) $$I_{resistor} = \sum_{n=1}^{5} I_{LED-n}$$

By Ohm's law, we can combine 3b and 4 to produce:

(5) $$V_{drop-LED-n} = V_{pin-n} - 220\sum_{n=1}^{5} I_{LED-n}$$

While a bit busy, equation 5 essentially states that the voltage drop across a given LED n is a function of both the initial voltage applied to it *and the current across all other LEDs in the circuit*. As pins switch between HIGH and LOW, the state of current in all LEDs will vary over time, resulting in variable voltage supplied to any given LED. **For this reason, we can expect less stable voltage and current supply subject to how many pins are HIGH in the "alternate" circuit, and we might therefore deem the "built" circuit more reliable.**

---

[1] This is a convenient simplification of Kirchhoff's law based on the reasonable assumption no current is flowing backwards through any of the LEDs.

3) The highlighted area is an external reset button for the Arduino. Pulling the RESET pin of the Arduino LOW will reset the device, causing it to re-run the code uploaded to the board. In this particular implementation, there is a 10KΩ pull-up resistor that, when the button isn't being pushed, pulls the RESET pin HIGH, ensuring that a "floating pin" situation doesn't result in electrical noise causing an erroneous reset. When the button is pushed, the circuit between 5V and RESET connects to GND, pulling the RESET pin LOW, and resetting the board.
4) For my custom message, I chose the phrase, "full marks?" by substituting the string passed as input to the displayString function within the loop function of the provided code.