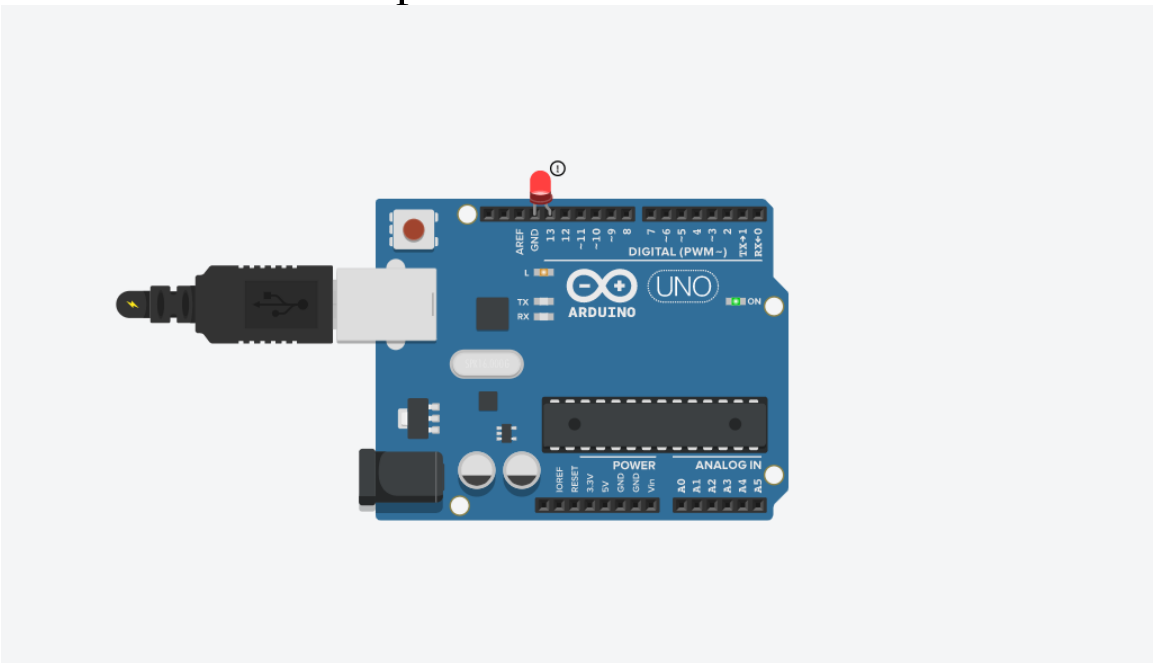
LAB REPORT 4

Ben Giftakis
3/2/21

Screenshot + Components:



- Arduino – single board computer
- Led- takes on pins and turns it into light

Summary:

- To do this lab I first had to find the address range I needed to work with, I used the slides to find it was the 20's. then I used the existing library functions to find the state of these addresses at a LOW and HIGH state. I then followed the example in the slide to manipulate those memory addresses and their values according to the screenshot below.

Results:

```
020: 00 00 00 00 00 00 00 00 00 01 00 00 00 00 00 00

020: 00 00 00 20 20 20 00 00 00 01 00 00 00 00 00 00
```

This is the memory data at low and high

Conclusions:

- Learned about how the pins are controlled via memory mappings
- Learned how to manually manipulate that memory

- I was surprised to find that in my testing the Arduino gets mad when you declare something twice, but only if its twice in the loop, since it needs to declare again when the loop repeats

Code:

```
/*
  Memory Mapped I/O Exploration
 */

/* Figure out what bits control pin 13. make use of the provided functions
*  to do this.
*/
void setup() {
  // Setup for Serial output
  Serial.begin(9600);
/*
  displayRAM((char*) 0x20,(char*)  0x2f, true);
  int led = 13;
  pinMode(led, OUTPUT);
  digitalWrite(led, HIGH);

    displayBits((char *)0x23);
  displayBits((char *)0x24);
  displayBits((char *)0x25);

  //displayRAM((char*) 0x20,(char*)  0x2f, true);
  */
//displayRAM((char*) 0x20,(char*)  0x2f, true);

}

/*once you know which bit can be used to turn pin 13 on and off,
* try to blink an led in loop without using digitalWrite().
*/
void loop() {
  char *a = (char *) 0x23;
  *a ^=0x20;
  char *b = (char *) 0x24;
  *b ^=0x20;
  char *c = (char *) 0x25;
  *c ^=0x20;
  delay(1000);
  //displayRAM((char*) 0x20,(char*)  0x2f, true);
 // char *a = (char *) 0x23;
```

```
  *a ^=0x0;
  //char *b = (char *) 0x24;
  *b ^=0x0;
  //char *c = (char *) 0x25;
  *c ^=0x0;
  delay(1000);
}

/* example call displayRAM((char *) 0x8E0, (char *) 0x8FF, true);
* if hex is false, letters and numbers will be printed, and all other valu
es will be represented as '.'  */
void displayRAM(char *start, char *endd, bool hex) {
  char *array;
  for(array = start; array < endd; array += 0x10) {
    //create row number
    if (array < (char *)0x10)
      Serial.print('0');
    if (array < (char *)0x100)
      Serial.print('0');
    Serial.print((int)array, HEX);
    Serial.print(": ");
    //for each index (0 through 15 inclusive)
    for(int i = 0; i < 0x10; i++) {
      if(hex) {
        if (array[i] >= 0x00 && array[i] < 0x10)
          Serial.print('0');
        Serial.print(array[i] & 0xFF, HEX); //0xFF is our bitmask
      } else {
        Serial.print((array[i] >= ' ' && array[i] <= 'z') ? array[i] : '.'
);
      }
      Serial.write(' ');
    }
    Serial.println();
  }
  Serial.println();
}

//Example call displayBits((char *) 0x100);
//pretty prints an address in binary
void displayBits(char *address) {
  Serial.print("0x");
  Serial.print((int) address, HEX);
  Serial.print(": ");
  Serial.println(address[0], BIN);
```

```
}
```