Sajana Wijesinghe

**Title:**

* *Lab number 3*

**Statement of the Problem:**

* *What question(s) are you trying to answer?*
  + How are we able to use an Arduino and the IDE program to display the contents of an EEPROM are preserved across power outages?
* *Preliminary observations or background information required*
  + *Programming in C*

**Materials:**

* 1 Arduino Uno
* 1 USB Cable
* Arduino software installed on the PC

**Procedure:**

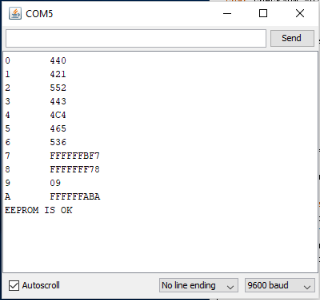
1. *Start with an Arduino Uno board. Using the USB cable connect the Arduino to the computer and load up the Arduino IDE.*
2. *On a new sketch in the IDE, we declare two functions, setup() and loop().*
3. *Use checkSum and magic numbers to be able to verify the contents of  EEPROM*
4. *To check suitable values: we had to check them at random, since that's how magic numbers work.*
5. *Had some error handling cases just in case magic number or checksum is getting a value that isn't supposed to be recognized or is matched.*
6. *Had to include header file #include <EEPROM.h> to be able to use the read and write methods.*
7. *We created the setup method which outputs memory space of EEPROM from beginning to end*
8. *Now we tested setup function with actual values to make it read and write out the bytes from EEPROM. We compared the value we got with the last location of EEPROM*
9. *First time it didn't work, but the we had to change the code, which represents the values of our configuration data.*
10. *We ran a bitwise exclusive-or on all the bytes that were present in EEPROM, except for the last one. In the last one we just store the values we are currently dealing with as a checkSum. Then print a message saying “EEPROM has been reinitialized.”*
11. *If correct, print “EEPROm check is okay”*
12. *We unplugged our board and reconnected/reset button*
13. *Swapped Arduino boards with another group to see their result.*

**Code:**

#include <EEPROM.h>  
char checkSUM()  
{  
  char checkSUM =0;  
  for(int i = 0; i < 0x07; i++)  
  {  
    checkSUM = checkSUM ^ EEPROM.read(i);  
  }  
  return ~checkSUM;  
}  
void setup()  
{  
   
 Serial.begin(9600);  
 int checksum\_slot= 124;  
  
 for(int address =0; address < 0x0B; address++)  
{  
  char value = EEPROM.read(address);  
  Serial.print(address, HEX);  
  Serial.print("\t");  
  Serial.print(value, HEX);  
  Serial.print(address, HEX);  
  Serial.println();  
    
}  
  
 // EEPROM.write(checksum\_slot, 'D');  
   
 if((checkSUM() & 0xff) != EEPROM.read(checksum\_slot))  
 {  
   EEPROM.write(0, 'D'); EEPROM.write(1, 'B');  
   EEPROM.write(2, 'U'); EEPROM.write(3, 'D');  
   EEPROM.write(4, 'L'); EEPROM.write(5, 'F');  
   EEPROM.write(6, 'S');   
   EEPROM.write(checksum\_slot, checkSUM());   
   Serial.println("EEPROM has been reinitialized");  
 }  
 else  
   Serial.println("EEPROM IS OK");   
}  
  
void loop(){  
}

**Results:**

* *The serial monitor displays the following:*



**Conclusions:**

* *We learn about how EEPROM works with its memory space. We learned how to deal with configuration data and how to ask the computer for what we need and how magic numbers work. EEPROm is able to save data when power outages occur, which is what we saw when we hit the reset button and unplugged the Arduino board. The values/content stayed the same and EEPROM was reinitialized.*
* *Potential Errors:*
  + *Not selecting the correct COM port.*
  + *Both messages weren’t appearing sometimes*
    - *“EEPROM has been reinitialized.”*
    - *“EEPROM is Okay”*