

*Editor's note: These are the listing for the article, "ChatMagLev: Levitating the AI Way" (P. Neufeld, Elektor Mag, Nov/Dec 2023). <https://www.elektormagazine.com/magazine-archive>*

### **Listing 1: Extension of handwritten code.**

This was ChatGPT's response on my question about an extended version of my code, with the ability to control trigger value and hysteresis level by simple commands, sent via the serial line:

*Here's the extended code for the Arduino C++ program you provided:*

```
#include <Preferences.h>

Preferences preferences;

int TRIGGER = 2740;

int HYST = 35;

int HALL_PIN = 33;

int HALL_VAL = 0;

int RELAIS_PIN = 23;

int X = 0;

void setup() {

    Serial.begin(115200);

    pinMode(RELAIS_PIN, OUTPUT);
```

```
Serial.print("Magnetic Levitation: START ");

// Load values from non-volatile RAM
// Initialize preferences with a namespace
preferences.begin("levitation", false);

if (preferences.getBytesLength("config") > 0) {
  preferences.getBytes("config", &TRIGGER,
    sizeof(TRIGGER));
  preferences.getBytes("config", &HYST,
    sizeof(HYST));
}

preferences.end();
}

void loop() {

  if (Serial.available() >= 3) {
    String command = Serial.readStringUntil('\r');
    // Read the command until a
    // carriage return (\r) is encountered
```

```
if (command.startsWith("T")) {
    TRIGGER++;

    Serial.println("Trigger increased: "
        + String(TRIGGER));
}

else if (command.startsWith("t")) {
    TRIGGER--;

    Serial.println("Trigger decreased: "
        + String(TRIGGER));
}

else if (command.startsWith("H")) {
    HYST++;

    Serial.println("Hysteresis increased: "
        + String(HYST));
}

else if (command.startsWith("h")) {
    HYST--;

    Serial.println("Hysteresis decreased: "
        + String(HYST));
}

else if (command.startsWith("S")) {
    // Store current values in non-volatile RAM

    // Initialize preferences with a namespace
```

```

preferences.begin("levitation", false);

preferences.putBytes("config", &TRIGGER,
                    sizeof(TRIGGER));

preferences.putBytes("config", &HYST,
                    sizeof(HYST));

preferences.end();

Serial.println(
    "Values stored in non-volatile RAM.");
}
}

HALL_VAL = analogRead(HALL_PIN);

if (HALL_VAL < (TRIGGER + X)) {
    digitalWrite(RELAIS_PIN, HIGH);
    X = HYST;
} else {
    digitalWrite(RELAIS_PIN, LOW);
    X = 0 - HYST;
}
}
}

```

*In this code, the preferences library is used to store and load the trigger and hysteresis values in non-volatile RAM. The new commands 'T', 't', 'H', and 'h' are used to increase the trigger,*

*decrease the trigger, increase the hysteresis, and decrease the hysteresis values, respectively. The command 'S' stores the current values in non-volatile RAM. These values will be used at startup if present. Please note that you need to install the "Preferences" library for this code to work correctly. You can find the library on the Arduino Library Manager or download it from the Arduino website.*

## **Listing 2: Generated code and response from ChatGPT.**

```
#include <EEPROM.h>

const int magnetPin = 23;

const int hallSensorPin = 33;

const int triggerAddr = 0;

const int hysteresisAddr = 1;

float triggerValue = 2.4;

float hysteresisValue = 0.05;

void setup() {

  Serial.begin(115200);

  pinMode(magnetPin, OUTPUT);

  pinMode(hallSensorPin, INPUT);

  // Load trigger and hysteresis values

  // from EEPROM if available

  EEPROM.begin(2);
```

```
float storedTriggerValue =
    EEPROM.readFloat(triggerAddr);

float storedHysteresisValue =
    EEPROM.readFloat(hysteresisAddr);

if (storedTriggerValue != 0x00)
    triggerValue = storedTriggerValue;

if (storedHysteresisValue != 0x00)
    hysteresisValue = storedHysteresisValue;

Serial.println("Magnetic Levitation Setup");
Serial.println("Commands:");
Serial.println("T - Increase trigger value");
Serial.println("t - Decrease trigger value");
Serial.println("H - Increase hysteresis value");
Serial.println("h - Decrease hysteresis value");
Serial.println(
    "S or s - Save trigger and hysteresis values");
Serial.println();
Serial.print("Trigger Value: ");
Serial.println(triggerValue);
Serial.print("Hysteresis Value: ");
```

```

Serial.println(hysteresisValue);
}
◦
void loop() {
    // Read the analog value from the Hall sensor
    int hallSensorValue = analogRead(hallSensorPin);

    // Check if magnet is close to the sensor
    if (hallSensorValue >
        (triggerValue + hysteresisValue) * 1023) {
        // Magnet is close, turn off the electromagnet
        digitalWrite(magnetPin, LOW);
    }
    else {
        // Magnet is far, turn on the electromagnet
        digitalWrite(magnetPin, HIGH);
    }

    // Check for serial input
    if (Serial.available() > 0) {
        char command = Serial.read();
        if (command == 'T') {
            triggerValue += 0.01;

```

```
}  
else if (command == 't') {  
    triggerValue -= 0.01;  
}  
else if (command == 'H') {  
    hysteresisValue += 0.01;  
}  
else if (command == 'h') {  
    hysteresisValue -= 0.01;  
}  
else if (command == 'S' || command == 's') {  
    // Save trigger and hysteresis values  
    // to EEPROM  
    EEPROM.writeFloat(triggerAddr, triggerValue);  
    EEPROM.writeFloat(hysteresisAddr,  
        hysteresisValue);  
    EEPROM.commit();  
    Serial.println(  
        "Trigger and hysteresis values saved.");  
}  
  
// Output trigger and hysteresis values  
Serial.print("Trigger Value: ");
```



```
Serial.println(triggerValue);

Serial.print("Hysteresis Value: ");

Serial.println(hysteresisValue);

}

}
```

*Note: This is a compact version of the code without extensive comments. It assumes you have the necessary libraries installed, including the EEPROM library. Make sure to include that library in your Arduino IDE if you don't have it already.*

### **Listing 3: Enhanced code.**

```
#include <Preferences.h>

const int magnetPin = 23;

const int hallSensorPin = 33;

const char* prefNamespace = "levitation";

const char* triggerKey = "trigger";

const char* hysteresisKey = "hysteresis";

float triggerValue = 2.4;

float hysteresisValue = 0.05;

Preferences preferences;
```

```
void setup() {  
  
    Serial.begin(115200);  
  
    pinMode(magnetPin, OUTPUT);  
    pinMode(hallSensorPin, INPUT);  
  
  
    preferences.begin(prefNamespace, false);  
    if (preferences.getBool("initialized", false)) {  
        triggerValue =  
            preferences.getFloat(triggerKey, triggerValue);  
        hysteresisValue =  
            preferences.getFloat(hysteresisKey,  
                hysteresisValue);  
    }  
    else {  
        preferences.putBool("initialized", true);  
    }  
  
    Serial.println("Magnetic Levitation Setup");  
    Serial.println("Commands:");  
    Serial.println("T - Increase trigger value");  
    Serial.println("t - Decrease trigger value");  
    Serial.println("H - Increase hysteresis value");  
    Serial.println("h - Decrease hysteresis value");  
}
```

```

Serial.println(
  "S or s - Save trigger and hysteresis values");
Serial.println();
Serial.print("Trigger Value: ");
Serial.println(triggerValue);
Serial.print("Hysteresis Value: ");
Serial.println(hysteresisValue);
}

void loop() {
  int hallSensorValue = analogRead(hallSensorPin);

  if (hallSensorValue > (triggerValue * 1023) &&
    digitalRead(magnetPin) == HIGH) {
    digitalWrite(magnetPin, LOW);
  }

  else if (hallSensorValue <
    ((triggerValue - hysteresisValue) *
    1023) && digitalRead(magnetPin) == LOW) {
    digitalWrite(magnetPin, HIGH);
  }

  if (Serial.available() > 0) {
    char command = Serial.read();

```

```
if (command == 'T') {
    triggerValue += 0.01;
} else if (command == 't') {
    triggerValue -= 0.01;
} else if (command == 'H') {
    hysteresisValue += 0.01;
} else if (command == 'h') {
    hysteresisValue -= 0.01;
} else if (command == 'S' || command == 's') {
    preferences.putFloat(triggerKey,
        triggerValue);
    preferences.putFloat(hysteresisKey,
        hysteresisValue);
    preferences.end();
    Serial.println(
        "Trigger and hysteresis values saved.");
}

Serial.print("Trigger Value: ");
Serial.println(triggerValue);
Serial.print("Hysteresis Value: ");
Serial.println(hysteresisValue);
}
```

}

*This version addresses the two issues mentioned:*

*The code now includes hysteresis in the comparison by checking the magnetPin state in both conditions. If the electromagnet is already in the desired state, the condition for turning it on or off is not evaluated.*

*Instead of using the EEPROM library, this code uses the Preferences library provided by the ESP32 core. The preferences are stored in the non-volatile memory of the ESP32 using the Preferences API. The values are saved when the 'S' or 's' command is received and are read from the memory during setup. The preferences namespace and keys are specified for the levitation setup to avoid conflicts with other preferences.*