Report

SESSION 2 *(19/12/22)*

Vahan Komaryan

I made the code of the stepper motors. I thought that to make the code more readable and easy to use it would be better to use a class for the motors especially since we have to control two of them. So I wrote this class. The code is too long to be fully rendered here so it is accessible in the ***archive/session2/main*** folder on GitHub.

One of the most important things to take into account in the code is the fact that we could not use delays. Indeed, to produce the desired effect, the motors must be able to run simultaneously, with a delay we would stop the program completely when one motor is running while the other is not, which we don't want.

To solve this problem, I had the idea to use time varibles and the millis function. So, each engine object has its own time variables like the last time we updated its position or the next time we will update it. So we can get rid of the delay and update the engines only when a certain time has passed.

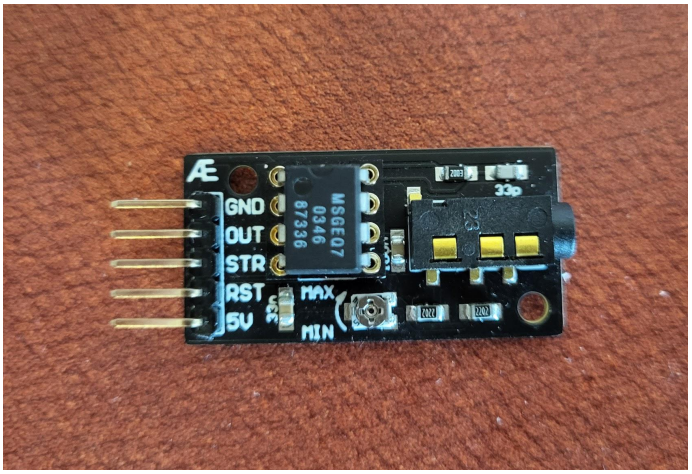
I realized too late that the "dir" input of one of the drivers doesn't work, which I will change at the next session.

I also discovered how to disassociate code in several files to make it more readable and organized.

I finally started to create some functions to move the motors which should for example draw a line on the wall but I haven't finished yet.

Benjamin Choiselat

This session, I have started working on the MSGEQ7 module. It has 5 pins, including the Out pin which is used to read the information, and the Strobe and Reset pins, which are used to calibrate the sound recording.



The output is connected in analog to the arduino, as it returns values between 0 and 1023 for each frequency depending on their intensity, while the reset and strobe pins are connected in digital, as they only need to be HIGH or LOW. I then wrote a code to try it out :

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(strobePin, OUTPUT);
  pinMode(resetPin, OUTPUT);
  pinMode(OutPin, INPUT);

  digitalWrite(resetPin,  LOW);
  digitalWrite(strobePin, LOW);
  delay(1);

  digitalWrite(resetPin,  HIGH);
  delay(1);
  digitalWrite(resetPin,  LOW);
  digitalWrite(strobePin, HIGH);
  delay(1);
```

First, we initialize the Reset and Strobe pins to LOW.

Then, we have to do a Reset sequence, required before starting to read. We put the Reset pin HIGH for at least 100 ns (1ms here), before putting it in LOW again. Now, each time the Strobe pin is put in HIGH, it will record the sound frequency by frequency (from 63 Hz to 16kHz and then back to 63Hz automatically).

In the loop, I just have to put the Strobe pin to HIGH, read the Out pin, keep the value in an array, and put the Strobe pin to LOW.

Finally, I tried to find a way to sort the first frequency to detect when there is a bass or not in the music, and tried to make a running average of the frequencies to then compare it to the new frequencies. But it's not working yet. I have put the code in the archives.