

In.Ra.Ci.

Avenue Jupiter, 188  
1190 Bruxelles

Technique de qualification  
Electronique

Année scolaire 2023-2024

# La main robotisée

Thomas Giarrizzo

89 Jan Baptist Wautersstraat

1600 Sint-Pieters-Leeuw



In.Ra.Ci

Avenue Jupiter, 188  
1190 Bruxelles

Technique de qualification  
Electronique

Année scolaire 2023-2024

# La main robotisée

Thomas Giarrizzo

89 Jan Baptist Wautersstraat

1600 Sint-Pieters Leeuw

Pour commencer, je tiens à remercier toutes les personnes qui m'ont  
aidé à la réalisation de mon projet.

Je remercie mes parents pour leur soutien, qui m'ont beaucoup aidé au niveau de la  
mécanique et de l'achat du matériel.

Je souhaite exprimer ma gratitude à mon promoteur, Monsieur F. Kapita  
pour tout ce qu'il m' a apporté.

Sa façon de travailler, sa patience et son sérieux m'ont été précieux  
pour la réalisation de mon projet.

Je remercie également tous mes professeurs d'option, M. Mazzeo et M. Block,  
pour tout ce qu'ils m'ont apporté lors de mon apprentissage en électronique.

Je remercie aussi Mme Hekster pour ses conseils et pour les corrections qu'elle a apportées à  
mon travail.

Je remercie mon camarade Daoud avec qui j'ai eu la chance de partager la réalisation de ce  
projet.

## Table des matières

1. Introduction .....	6
2. Schéma bloc .....	7
2.2 Schéma bloc du gant .....	7
2.3 Schéma bloc de la main .....	8
3. Les caractéristiques .....	9
4. Principes mis en jeu .....	11
4.1 Etude détaillée de l' interruption timer et de la réentrance d'une fonction en « c » .....	11
4.2 Etude détaillée de la transmission Bluetooth, modulation physique et émission UART .....	17
5. Etude détaillée et mesures .....	22
5.1 Le schéma complet .....	22
5.2 Etude des capteurs de flexion .....	24
5.2.1 Mesure des capteurs .....	27
5.3 Etude des servomoteurs .....	30
5.3.1 Mesure des servomoteurs .....	31
5.4 L'ADS 1115 .....	34
5.5 Le bus I2C .....	36
5.6 Étude de l'écran e-link de la M5STACK PAPER .....	39
5.7 Encodeur EC11 .....	41
6. La programmation .....	43
6.1 L'ordinogramme général de la transmission .....	43
6.2 L'interruption .....	44
6.2 Réception .....	45
6.4 Le programme .....	46
7. La fabrication .....	48
8. La mise au point .....	49
9. La conclusion .....	50
Les annexes .....	51
1. Schéma de principe sans connecteur .....	51
2. Schéma de principe avec connecteur .....	52
3. PCB .....	53
4. Sérigraphie du PCB .....	53
5. Dessin d'implantation dans le boîtier .....	54
6. Vue 3D .....	55
Support m5 paper .....	55
6.1 Support pour module adc1115 .....	56

6.2 Support pour la main robotique .....	57
6.3 Vue d'ensemble du projet assemblé.....	58
6.4 Vue de la main robotique .....	59
7. Mise en plan .....	60
Support pour la main robotique .....	60
7.1 Support m5 paper .....	61
7.2 Support module ADC 1115 .....	63
8. Fiches techniques des composants peu courants.....	64
8.1 Datasheet Feather ESP32.....	64
8.2 Datasheet M5stack paper .....	65
8.3 Datasheet servomoteurs .....	66
8.4 Datasheet adc 1115 .....	67
8.5 Datasheet capteur de flexion ZD10-100.....	68
8.6 Datasheet des encodeurs .....	71
Sources du projet.....	72

## 1. Introduction

Comme projet de fin d'études, j'ai opté pour la réalisation d'une main robotisée contrôlée à l'aide d'un gant connecté en Bluetooth.

Mon projet est composé de deux parties.

**La première partie** concernée est le gant Bluetooth.

Dans ce gant, il y a 5 capteurs de flexion (flex sensor) qui sont connectés en série à une résistance de  $8k2\Omega$ . Ces cinq résistances sont soudées sur deux cartes ADS qui sont reliées ensemble.

D'une de ces deux cartes, une fiche est reliée à ma carte de programmation M5PAPER.

L'ensemble est attaché à mon bras à l'aide de deux pièces 3D que j'ai personnellement dessinées en 3D et imprimées avec mon imprimante 3D.

Dans la M5PAPER, se trouve mon programme avec 3 modes qui me permettent de faire différentes manipulations (mode avec le gant et mode manuel).

**La deuxième partie** se compose d'une main robotisée qui contient 5 servomoteurs.

Ceux-ci sont reliés à un microcontrôleur ESP32 qui récupèrent en Bluetooth les informations nécessaires pour faire fonctionner la main.

A l'entrée de la carte ESP32, il y a également 8 encodeurs qui me permettent de faire lever ou abaisser les doigts de la même façon que le gant.

J'ai voulu effectuer ce projet car j'ai été attiré par le fait de pouvoir commander une main en Bluetooth et je me suis dit que, par la suite, je pourrais l'améliorer et potentiellement en faire une prothèse.

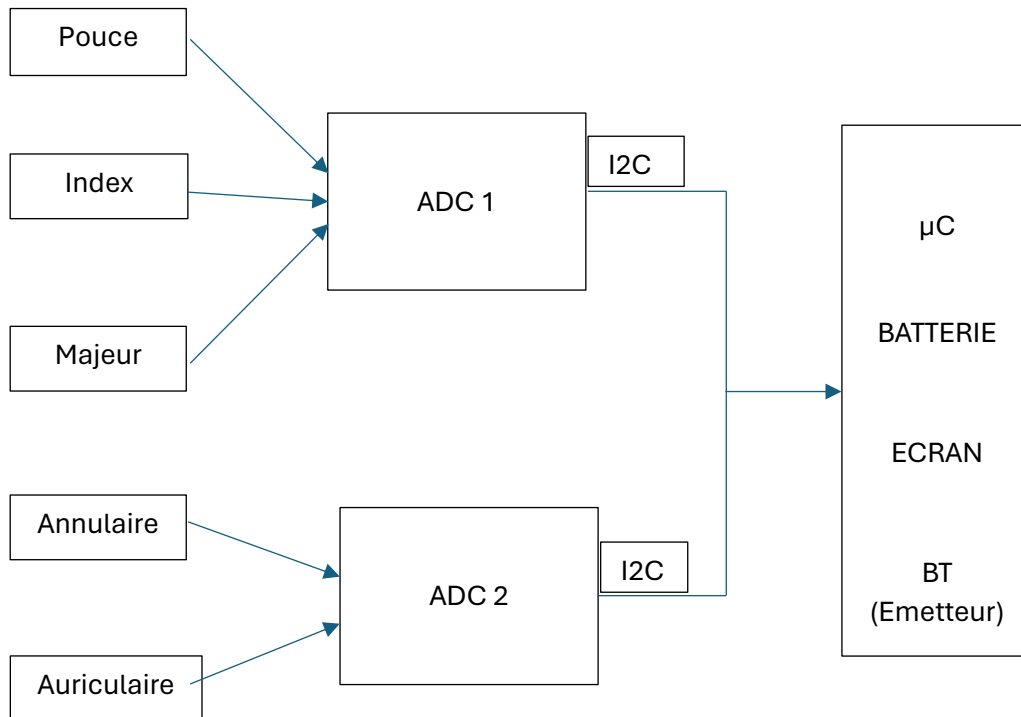
J'ai choisi ce projet assez complexe, car je pense avoir les compétences nécessaires pour le réaliser. Le fait de choisir un projet dont l'envie est présente pour le réaliser est une force, car la motivation sera là, tout au long de sa conception.

Etant un grand fan de Iron Man, le souhait de réaliser ce gant m'est venu à l'idée, car toutes ces technologies donnent envie d'être réalisées.

## 2. Schéma bloc

### 2.2 Schéma bloc du gant

#### **Résistances variables**



#### **Explication**

Les trois premières résistances variables (capteurs de flexions ZD10-100) sont connectées sur un premier module ADC (Analog to Digital Converter).

Les deux autres résistances variables (capteurs de flexions ZD10-100) sont connectées sur un second module ADC.

La raison est qu'un module ADC ne contient que 4 entrées.

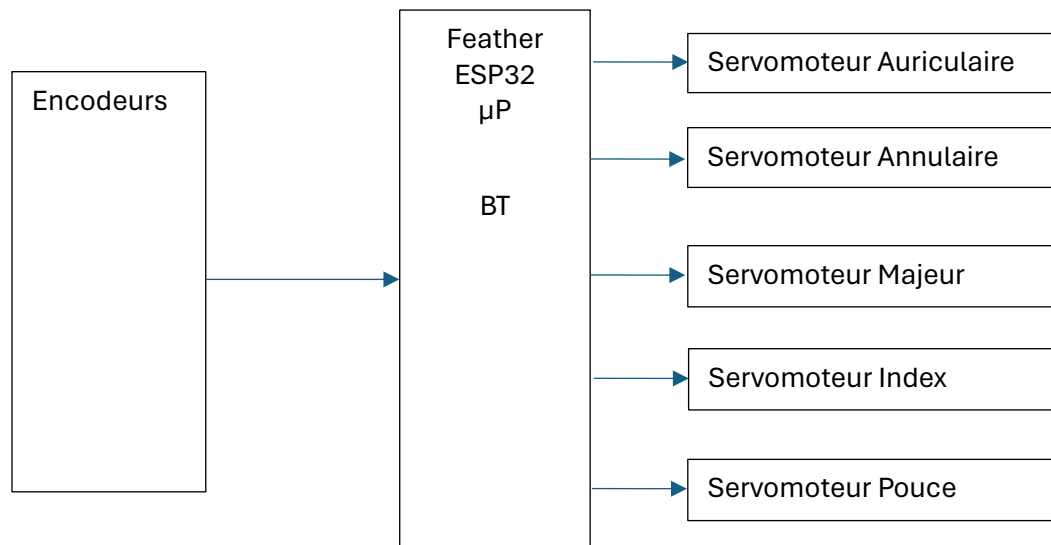
Les deux modules ADC sont reliés en parallèle, ce qui va nous permettre de sortir de ces deux modules en seulement 4 fils qui sont connectés dans le port A de la M5 PAPER.

Les modules ADC vont permettre de convertir avec précision les signaux analogiques en une forme numérique afin que le microcontrôleur puisse les traduire.

Le microcontrôleur (émetteur) va envoyer les informations en Bluetooth au microprocesseur (récepteur) de la main robotisée.



### 2.3 Schéma bloc de la main



#### **Explication**

Nous avons plusieurs encodeurs (huit) qui vont être connectés aux microprocesseurs (récepteur).

Cinq encodeurs sur huit sont utilisés pour contrôler chacun un doigt bien défini.

Les trois autres encodeurs sont utilisés pour commander les leds qui se trouvent au-dessus de chaque encodeur.

L'encodeur utilisé est le « M5Stack 8-Encoder Unit ».

Le microprocesseur est programmé de telle sorte que, si on pousse ou qu'on tourne un des encodeurs, le servomoteur du doigt choisi est commandé.

Le microprocesseur est également programmé de telle sorte que, si le microcontrôleur du gant lui envoie des informations, il lit la (ou les) donnée(s) pour que les servomoteurs soient contrôlés par le gant.

### 3. Les caractéristiques

#### Caractéristiques générales

##### Main :

- Robotisée
- 5 servomoteurs pour le pilotage
- Utilisation intérieure et extérieure
- Feather ESP32
- 8 encodeurs
- 255 possibilités de positions

##### Gant :

- 5 capteurs de flexion
- Feather ESP32 M5PAPER
- 2 pièces 3D
- 2 cartes ADC 1115
- Ecran tactile capacitif

#### Caractéristiques électroniques

##### Main :

- Alimentée en 3,3V
- Contrôlée en Bluetooth (2,4GHz) via une Feather (ESP32)
- Autonomie à déterminer en fonction de la batterie
- 300 mA/doigts
- Tension de fonctionnement des servomoteurs de 5 à 6 V
- Protocole de contrôle : 0,5ms à 2,5ms pour des servomoteurs 180°

##### Gant :

- Alimenté en 3,3V DC
- 5 capteurs de flexion commandés en 3,3V
- Autonomie +- 4h20
- Capacité 1150mAh
- Contrôlé en Bluetooth (2,4GHz) via une Feather

## Caractéristiques mécaniques:

### Main :

- Hauteur 189,27mm
- Largeur 66 mm
- Poids 943g
- Matériau Acrylique noir 5.0
- Diverses parties en métal
- Support 3D en PLA

### Gant :

- Gant souple
- Taille L
- Matériau tissu
- Pièces 3D en PLA

## 4. Principes mis en jeu

### 4.1 Etude détaillée de l'interruption timer et de la réentrance d'une fonction en « c »

Les timers sont des registres présents dans le microprocesseur : ils permettent de compter et de synchroniser des signaux. Ils s'incrémentent en fonction des impulsions d'horloge ou d'impulsions externes.

Ces compteurs internes permettent de mesurer le temps ou de générer des événements périodiques. De plus, il est possible de les configurer pour modifier leur comportement.

La valeur des timers est stockée dans un registre propre à eux.

La plupart des microprocesseurs possèdent 3 timers :

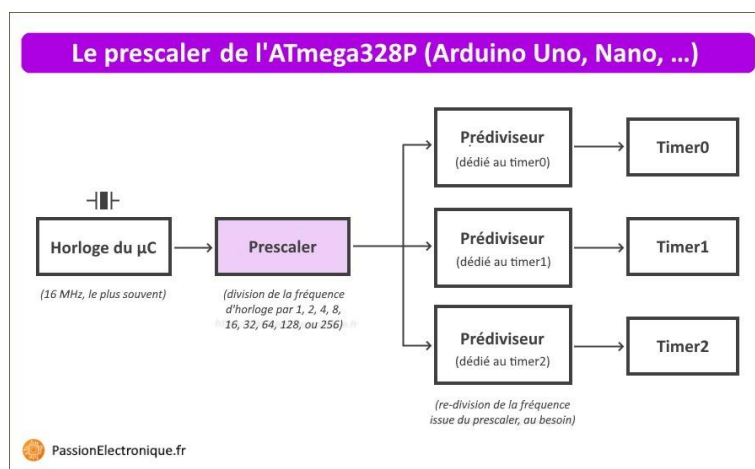
- Le timer 0 est de 8 bits (0-256). Celui-ci va commander le PWM et on l'utilise aussi pour les fonctions delay(), millis.
- Le timer 1 est de 16 bits (0-65535). Il est également utilisé pour le PWM sur des broches différentes du timer 0. On l'utilise surtout pour la librairie Servo.h.
- Le timer 2 est de 8 bits et est utilisé par la fonction Tone et est aussi utilisé pour le PWM.

Ces compteurs commencent à zéro et avancent un à un. Une fois qu'ils ont fini de compter, ils repassent immédiatement à zéro.

A chaque fois qu'ils reviennent à zéro, un signal est émis.

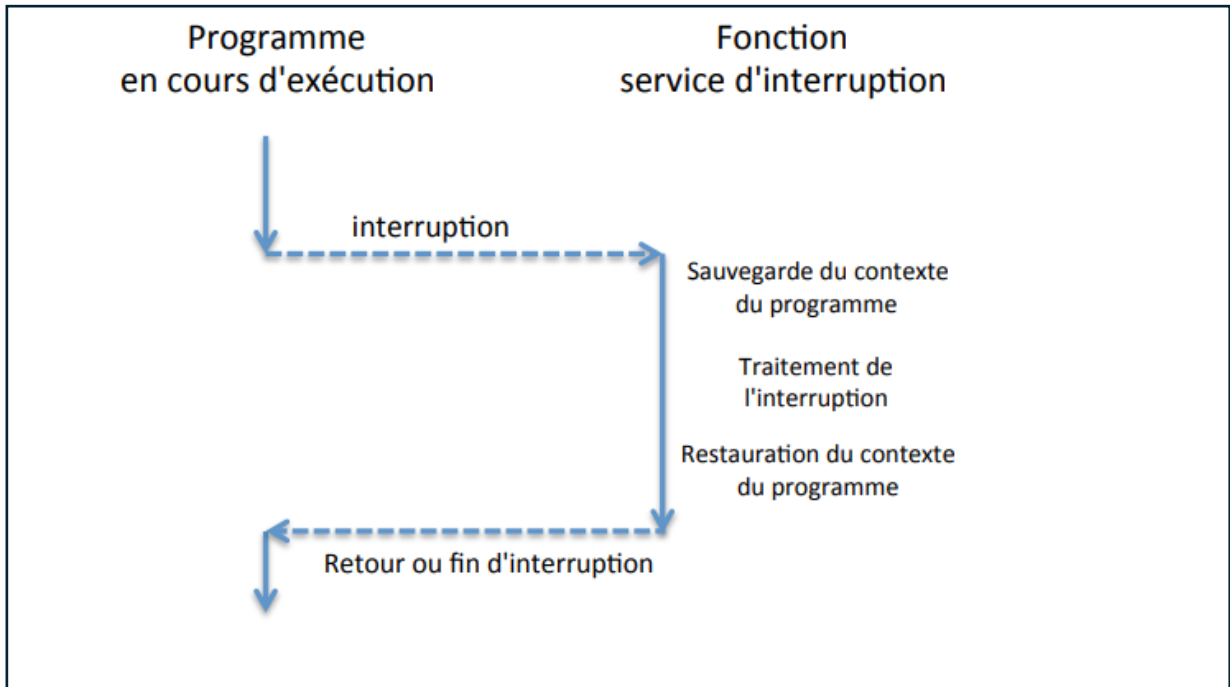
Même si nos timers commencent à zéro, il est possible de forcer leurs valeurs pour qu'ils commencent à compter à partir d'un certain nombre.

Il faut savoir que la cadence de ces timers se fait à partir de la fréquence d'horloge d'un microcontrôleur, mais ils peuvent aussi être divisés par 2 diviseurs de fréquence d'horloge : le prescaler et le prédiviseur.



Source : <https://passionelectronique.fr/introduction-timer-arduino/>, consulté le 15 novembre 2023.

L'interruption est un évènement imprévisible qui va provoquer l'arrêt d'un programme qui est en cours d'exécution, afin d'exécuter un autre programme (routine) d'interruption. A la fin du programme d'interruption, le microcontrôleur va reprendre le programme là où il s'est arrêté.



Source : <https://polytech-prog.gricad-pages.univ-grenoble-alpes.fr/polytech-microc/documents/cours5.pdf>, consulté le 15 novembre 2023.

Il existe deux types d'interruption : l'interruption interne et externe.

L'interruption externe se fait sur les pins du  $\mu\text{p}$  : exemple un bouton poussoir.

L'interruption interne est déclenchée par le débordement d'un timer ou par le déroulement du programme, par exemple le résultat d'un calcul.

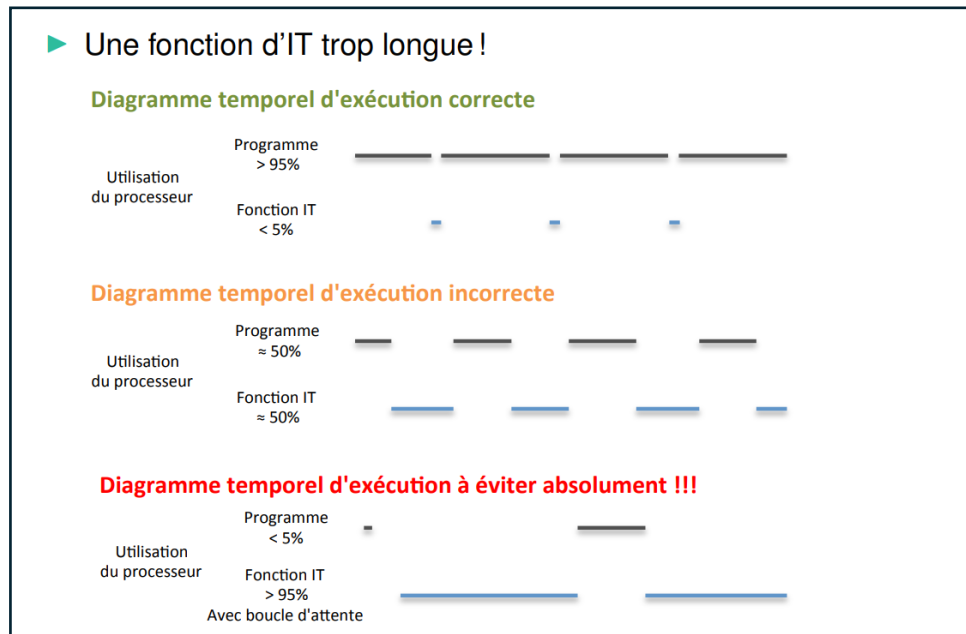
Il faut savoir que toutes les interruptions internes se font sur 3 bits :

- Le bit indicateur : Le bit est mis à 1 lorsque l'interruption correspondante surgit.
- Le bit d'activation : Il active ou désactive l'interruption spécifique.
- Le bit d'activation globale : Il permet d'activer ou de désactiver toutes les interruptions en une seule opération.

Tous ces bits sont regroupés dans des registres de configuration des interruptions.

Exemple : INTCON (registre principal de gestion des interruptions), PIE1, PIR1 et PIR2.

Il faut également éviter les interruptions trop longues, sinon le programme ne fonctionnera pas correctement.



Source : <https://polytech-prog.gricad-pages.univ-grenoble-alpes.fr/polytech-microc/documents/cours5.pdf>, consulté le 15 novembre 2023.

Les interruptions timers permettent d'exécuter des actions à des intervalles réguliers (ex : toutes les 50ms) sans perturber le reste du programme. On les utilise souvent pour la génération de signaux périodiques comme pour la mesure du temps.

Dans mon cas, je les utilise pour le traitement de données de mes ADC.

Le problème de réentrance d'une fonction en « c » se produit lorsqu'on appelle une même fonction à partir d'elle-même, plusieurs fois sans que celle-ci ne soit finie.

Une fonction est donc dite réentrante si elle peut être appelée par plusieurs parties du programme sans qu'il y ait de conflit.

Le problème de réentrance est souvent dû à l'utilisation des variables globales, car si plusieurs parties du programme utilisent la même variable, cela peut créer des interférences les unes avec les autres.

Une autre priorité est qu'une fonction réentrante ne peut pas appeler une fonction non réentrante.

```
int my_function(int x) {  
    return x * 10;  
}  
int my_second_function(int x) {  
    return my_function(x) * 20;  
}
```

Dans ce cas-ci les deux fonctions sont réentrantes, car le « x » est différent pour les deux fonctions.

```
int x;  
int my_function() {  
    return x * 10;  
}  
int my_second_function() {  
    return my_function() * 20;  
}
```

Dans ce cas-ci les deux fonctions ne sont pas réentrantes, car le « x » est commun aux deux fonctions.

## Caractéristiques

### Interruption :

- Rapide
- Exécute plusieurs tâches simultanément
- Interne ou externe

### Timers :

- S'incrémentent à chaque impulsion
- Le prédiviseur : chaque timer en a un propre à lui (divise la fréquence d'horloge)
- Le prescaler (diviseur de fréquence générale)



## Programme de test d'une interruption

```
esp_timer_handle_t timer;
```

Je commence par déclarer ma variable timer.

```
void IRAM_ATTR onTimer(void *param) {  
}
```

Ici, je crée mon interruption dans laquelle j'envoie mes données en Bluetooth.

```
void setup() {  
  
    delay(1000);  
    M5.begin();  
  
    // #ifdef USE_bluetooth  
    init_bluetooth();  
    // #endif  
    secondWire.begin(I2C_SDA, I2C_SCL, (uint32_t)400000U); // Initialisation du  
    deuxième bus I2C avec les broches SDA et SCL définies  
    init_screen(90, 90, 540, 960, 5);  
    init_ads1();  
    init_ads2();  
    // ***** Configuration de l'interruption du timer *****  
    esp_timer_create_args_t timerArgs; // crée une structure pour configurer le  
    timer  
    timerArgs.callback = &onTimer; // Définition de la fonction de rappel  
    timerArgs.arg = NULL; // rien de plus n'est passé à la fonction de rappel  
    esp_timer_create(&timerArgs, &timer); // Création du timer  
    esp_timer_start_periodic(timer, 20000); // Déclencher toutes les 20 ms  
}
```

Ici je le déclare dans mon setup. J'initialise le temps de déclenchement et je crée mon timer.

## 4.2 Etude détaillée de la transmission Bluetooth, modulation physique et émission UART

### Le Bluetooth

Le Bluetooth est une technologie de réseau sans fil, mais utilisable sur une faible portée (on parle d'une dizaine de mètres).

Actuellement, les dernières versions permettent d'atteindre 100m de portée.

Le Bluetooth utilise le système maître et esclave pour communiquer entre les appareils.

Le maître va s'occuper de l'initialisation et de la connexion entre les appareils.

L'esclave, quant à lui, va juste se connecter à l'appareil.

Quand un appareil qui utilise le Bluetooth est à la recherche d'autres appareils, il va envoyer des signaux pour détecter les différents appareils disponibles. Lorsqu'un appareil esclave est détecté, le maître va envoyer des requêtes de connexion à l'esclave. Celui-ci pourra accepter ou refuser la connexion.

Quand les deux appareils sont connectés, ils vont s'envoyer des données sous forme de paquet. Chaque paquet va contenir des informations sur la taille des données, l'adresse du destinataire et un code qui va vérifier l'intégralité des données.

Il faut savoir que, dans le Bluetooth, il y a un émetteur et un récepteur. La communication peut donc être half duplex ou full duplex.

Il existe deux types de Bluetooth le BLE et le Bluetooth classique.

Dans le cadre de mon projet, j'utilise le Bluetooth classique.

Dans le cadre du cours d'électronique, nous nous sommes intéressés à la couche radio.

Caractéristique de la couche radio :

- Modulation GFSK (Gaussian Frequency Shift Keying).  
Notre signal numérique entre dans le filtre avant d'entrer dans le démodulateur. Cela va permettre de réduire la raideur des flancs et donc de limiter la bande passante après la modulation.
- La bande de fréquence ISM est de 2,4 GHz à 2,483 GHz avec 2MHz de garde basse et 3,5 MHz de garde haute. Il va nous rester une bande de 2,402 GHz à 2,480 GHz qui est divisée en 79 canaux de 1 MHz.
- Saut de fréquence FHSS (Frequency Hopping Spread Spectrum). Il y a 1600 sauts de fréquence pseudo aléatoire. Cela permet de réduire les interférences avec d'autres appareils.
- On retrouve 3 classes de fonctionnement
  1. La classe 1 avec une puissance d'émission de 20 dBm et une portée de 100m.
  2. La classe 2 avec une puissance d'émission de 4 dBm et une portée de 14 m.
  3. La classe 3 avec une puissance d'émission de 0 dBm et une portée de 10 m.

## Modulation

Le but de la modulation est de modifier un paramètre d'un signal proportionnellement au signal que l'on souhaite transmettre.

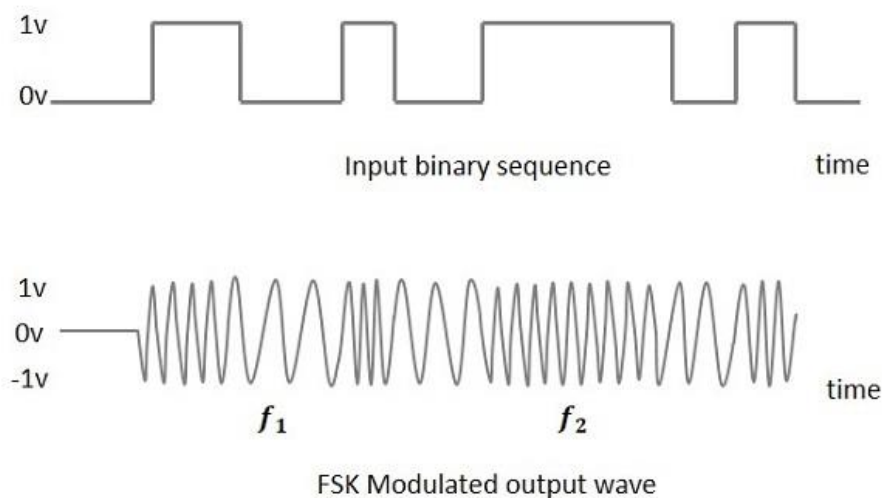
La modulation modifie le contenu spectral du signal.

### Quelles sont les raisons de moduler un signal ?

1. On ne peut pas avoir plusieurs émetteurs.
2. On a une très basse fréquence → très grande longueur d'onde → problème de propagation.

Il existe plusieurs type de modulation la FSK, AM, FM, PSK, etc.

La modulation FSK (Frequency Shift Keying) consiste à modifier la fréquence de la porteuse en fonction de la données à transmettre.



Source : [FSK](#) , consulté le 30 mai 2024.

Dans le cadre de mon projet c'est la modulation de type GFSK qui est utilisée.

La modulation GFSK est une variante de la modulation FSK (Frequency Shift Keying) qui utilise un filtre gaussien avant la modulation pour limiter la bande passante du signal.

Les données binaires vont d'abord passer à travers un filtre gaussien avant la modulation de la porteuse. Ce filtre permet d'adoucir la transition entre les bits, ce qui réduit ainsi la raideur des flancs du signal.

Le filtre gaussien atténue les composantes de haute fréquence, réduisant ainsi la bande passante du signal modulé. Cela permet de diminuer les interférences et permet une utilisation plus efficace du spectre radio.

Comment fonctionne la modulation ?

Le principe de la modulation est similaire à celui de la modulation FSK, mais les transitions sont plus douces grâce au filtrage.

La fréquence de la porteuse est modifiée en fonction des données filtrées, représentant les bits 0 et 1 avec des fréquences distinctes.

Avantages du GFSK

**Réduction de la bande passante** : Le filtrage gaussien avant la modulation limite la bande passante, rendant le signal plus efficace et moins susceptible d'interférer avec d'autres signaux.

**Moins de distorsion** : Les transitions douces réduisent les pics et les sursauts dans le signal, diminuant ainsi la distorsion et les erreurs de transmission.

Applications

GFSK est utilisé dans de nombreuses applications modernes de communication sans fil, telles que :

**Bluetooth** : Utilise GFSK dans certaines versions pour la transmission de données.

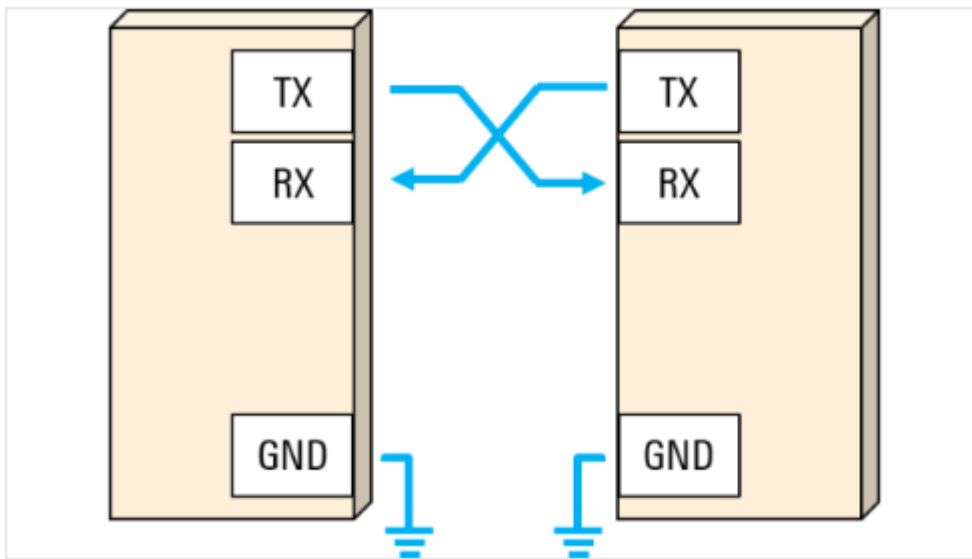
## L'émission UART

L'UART est un protocole série qui permet de faire communiquer et d'échanger des données entre deux appareils. Sur l'ESP32, on retrouve 3 UART.

Quand on utilise la console `Serial.println()` c'est l'UART1 qui est utilisé.

L'émission UART avec l'ESP32 est le processus de transmission de données séries asynchrones entre l'ESP32 et un autre périphérique.

L'UART est donc un protocole de communication série qui utilise deux fils pour envoyer des données. Un fil pour la transmission (TX) et un fil pour la réception (RX).



Source : [https://www.rohde-schwarz.com/fr/produits/test-et-mesure/essentials-test-equipment/digital-oscilloscopes/comprehension-uart\\_254524.html](https://www.rohde-schwarz.com/fr/produits/test-et-mesure/essentials-test-equipment/digital-oscilloscopes/comprehension-uart_254524.html), consulté le 15 janvier 2024.

## Constitution d'une trame UART

Une trame UART est toujours composée des bits suivants :

- Le bit de start qui est toujours à 0 et qui sert à la synchronisation du récepteur.
- Données : la taille des données est entre 5 et 9 bits. On commence par le LSB (bit de poids faible) au MSB (bit de poids fort).
- Bit de parité qui n'est pas toujours présent. Il peut être pair ou impair.
- Le bit de fin : celui-ci est toujours à 1.

### Vitesse de transmission

Pour faciliter la communication entre différents périphériques tels que les PC, les microcontrôleurs et les modems, les vitesses de transmission sont standardisées en utilisant des multiples et des sous-multiples de 9600 bauds. Le baud représente une unité de transmission par seconde.

Ces valeurs standardisées, telles que 9600 bauds, ainsi que d'autres valeurs comme 110, 300, 1200, etc., garantissent une compatibilité optimale et une communication fluide entre les périphériques.

Pour pouvoir échanger les données, les deux appareils doivent être connectés sur le même nombre de bauds. Dans mon cas, j'utilise 115200 bauds pour communiquer entre mes deux appareils.

### Caractéristiques

#### UART :

1. Asynchrone : Pas de signal d'horloge commun entre les deux dispositifs.
2. Nombres de bits de données : 8 bits.
3. Simplex ou duplex :
  - En simplex, un appareil communique à la fois.
  - En duplex, les deux peuvent communiquer en même temps.
4. Connecteurs : 2 fils RX et TX.

## 5. Etude détaillée et mesures

### 5.1 Le schéma complet

#### Schéma de la main robotique

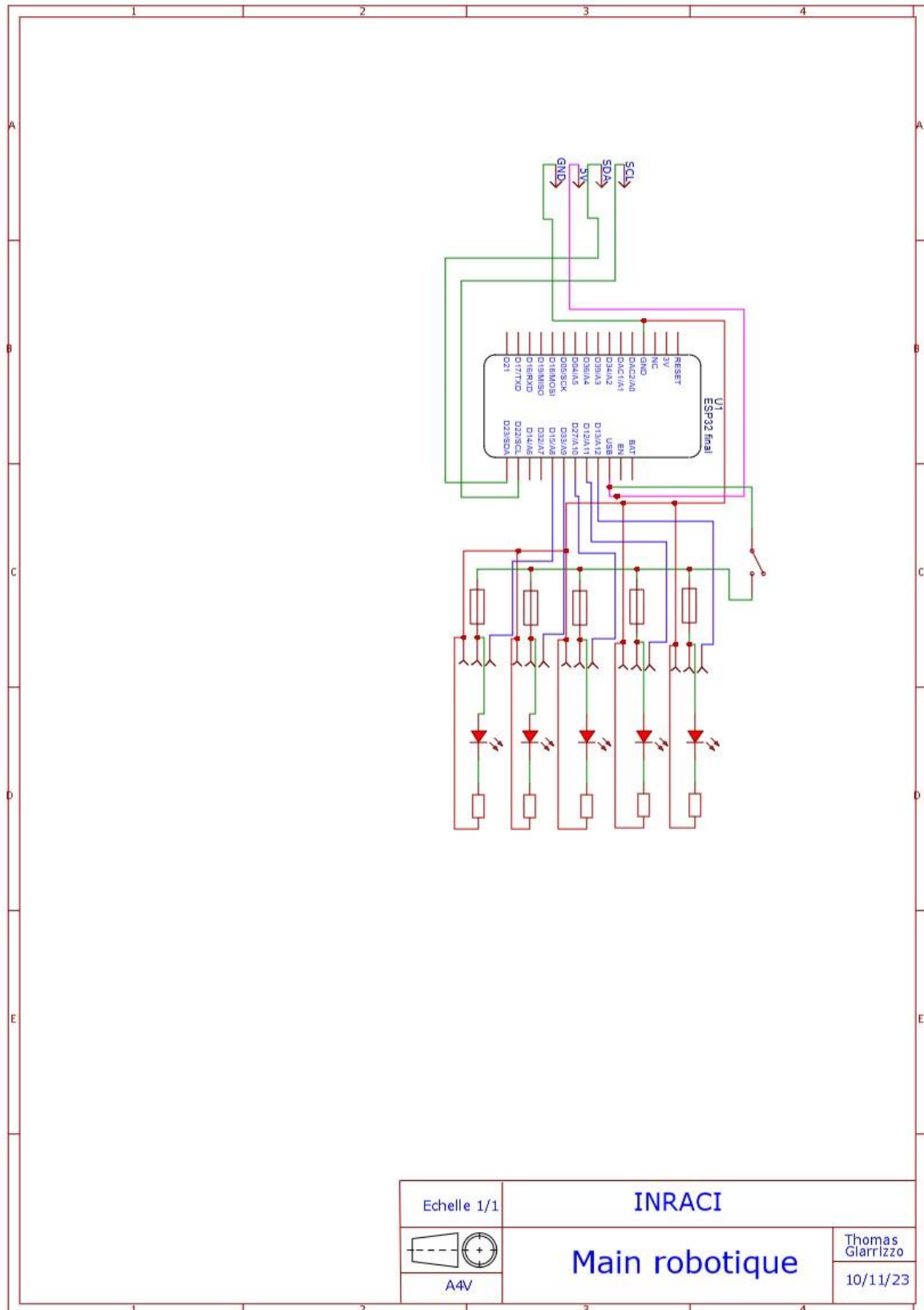
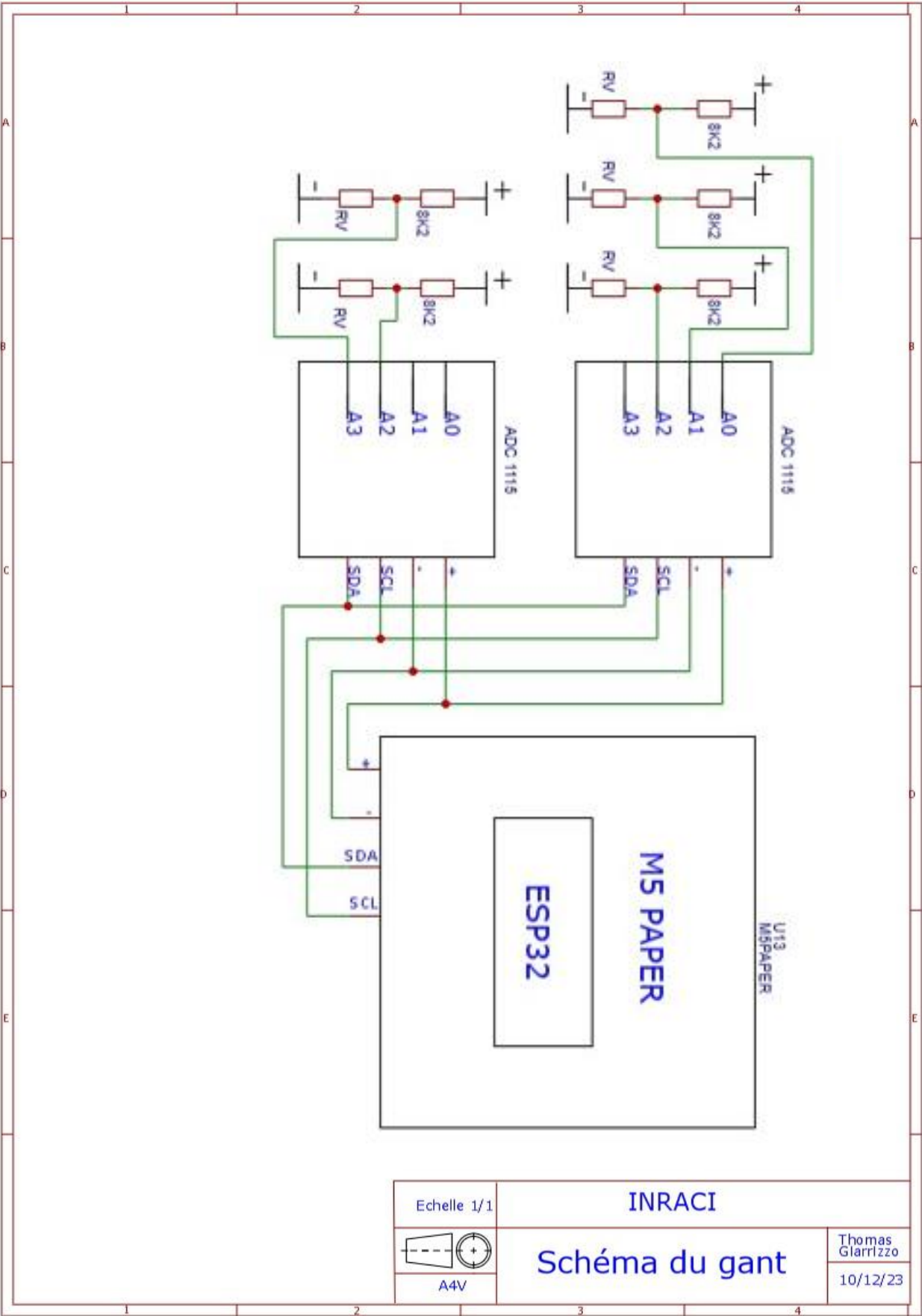


Schéma de principe du gant





## 5.2 Etude des capteurs de flexion



### Introduction

Un capteur de flexion est un capteur électronique qui mesure le taux de flexion d'un objet.

Il va agir comme un potentiomètre.

Le rôle d'un capteur de flexion est également de transformer le mouvement en un signal analogique afin de, par exemple, imiter un mouvement.

Un de ses avantages est de s'adapter à n'importe quelle forme (exemple : doigt d'une main).

Un capteur est constitué de 3 composants :

- La zone sensible qui permet de choisir la technologie des capteurs afin que ceux-ci soient adaptés à l'application souhaitée.
- L'électronique de traitement qui convertit le signal analogique en un signal numérique.
- La sortie du signal va contenir l'électronique qui sera reliée au reste du projet.

C'est pourquoi ces capteurs sont surtout utilisés en robotique, pour les jeux virtuels ou les instruments de musique ou dans le domaine médical.

### Explication du fonctionnement

Ce capteur contient une encre polymère qui possède des particules conductrices.

Lorsque ce capteur est plié, les particules s'éloignent et, par conséquent, la conductivité est réduite et la résistance augmente.

Au repos, c'est-à-dire à plat, les capteurs de flexion sont l'équivalent d'une résistance de 30k $\Omega$ .

Cette valeur varie en fonction du nombre de degrés à laquelle le capteur est exposé.

Par exemple à 90°, le capteur est l'équivalent d'une résistance de 70k $\Omega$ .

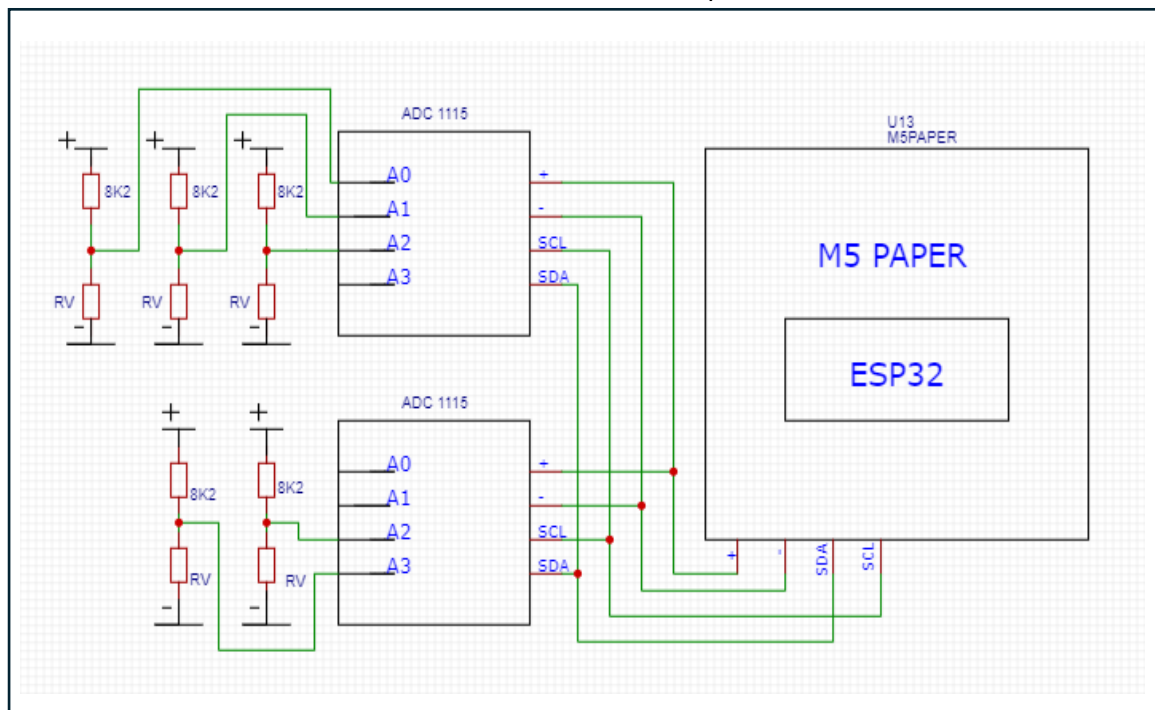
Pour utiliser ces capteurs de flexion avec un microcontrôleur, on utilise une résistance de  $10k\Omega$  qui va permettre de faire un diviseur de tension. Grâce à cela, nous allons avoir une tension variable qui va pouvoir être lue par un convertisseur analogique/numérique.

Le capteur peut donc être utilisé pour une entrée analogique avec une résistance de pull-up en série ou une entrée numérique avec un condensateur de  $0,1\mu F$ .

Caractéristiques (datasheet)

- Résistance plate : 25 000 Ohms
- Tolérance de résistance :  $\pm 30\%$
- Plage de résistance de la flexion : 45 000 à 125 000 Ohms (en fonction du rayon de courbure)
- Puissance nominale : 0,50 Watts en continu. 1 watt culminé

Schéma de fonctionnement des capteurs de flexions



## Programme de test

```
#include "BluetoothSerial.h"
#include <Wire.h>
#include "DFRobot_ADS1115.h"
#include <M5EPD.h>

int16_t adc0, adc1, adc2;
M5EPD_Canvas canvas(&M5.EPD);
DFRobot_ADS1115 ads1(&Wire);
```

```
void setup() {

M5.begin();
  M5.EPD.SetRotation(180);
  M5.TP.SetRotation(180);
  M5.EPD.Clear(true); // actualise l'ecran
  canvas.createCanvas(540, 960);
  canvas.setTextSize(5);
  Serial.begin(115200);
  canvas.pushCanvas(0, 0, UPDATE_MODE_DU4);

  ads1.setAddr_ADS1115(ADS1115_IIC_ADDRESS1); // 0x48
  ads1.setGain(eGAIN_TWOTHIRDS); // Définit le gain à deux tiers
  ads1.setMode(eMODE_SINGLE); // Définition du mode de fonctionnement en mode
unique
  ads1.setRate(eRATE_128); // Taux d'échantillonnage à 128SPS (défaut)
  ads1.setOSMode(eOSMODE_SINGLE); // Définition du mode de déclenchement en
mode unique
  ads1.init(); // initialise l'appareil avec tous les paramètres au-dessus
}
```

Nous sommes dans la partie Setup : c'est là que se fait l'initialisation. Dans ce cas-ci, je commence par l'initialisation de l'écran en lui disant la rotation de l'écriture et la plage dans laquelle je peux écrire. J'initialise le port série et tout ce qu'il faut pour mon module ADS 1115.

```
void loop() {
  canvas.pushCanvas(0, 0, UPDATE_MODE_DU4);
  if (ads1.checkADS1115()) {

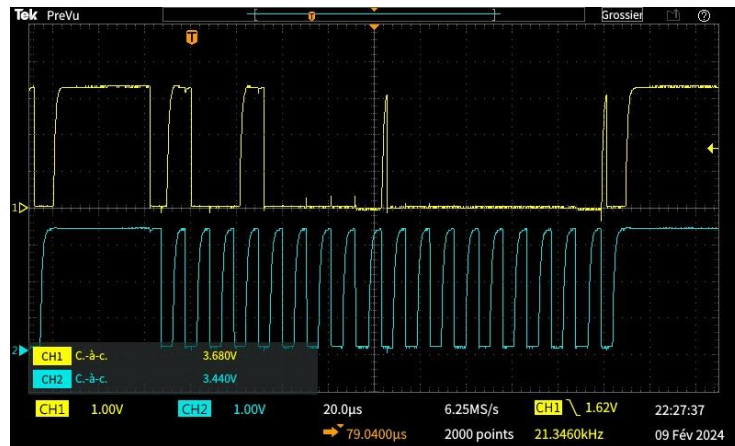
    adc0 = ads1.readVoltage(0);
    canvas.drawString("A0: " + String(adc0), 10, 60);
  }
}
```

Dans le loop, je « reset » l'écran dans un premier temps. Ensuite, je dis que si le module est prêt, ADC 0 est égal à la tension sur la pin 0 de l'ADC 1.

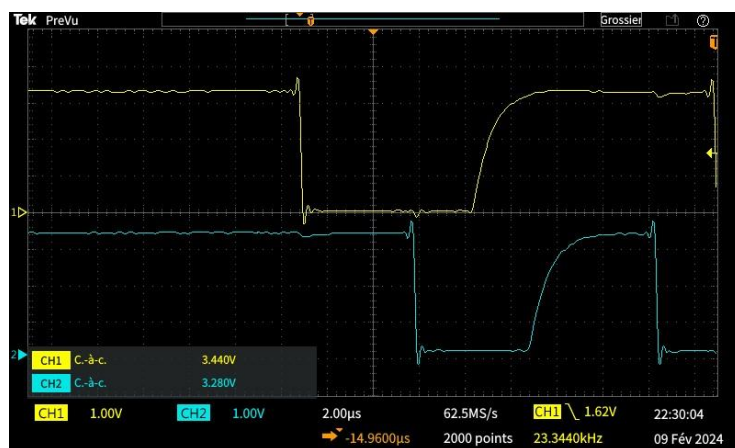
### 5.2.1 Mesure des capteurs

3 captures d'écran

#### Capteur de flexion à plat



#### Zoom sur le bit de start



#### Capteur totalement plié



### Tableau de trois mesures différentes

	Ohmmètre	Voltmètre	Positions des capteurs (ADC / 25)	Valeurs des capteurs sans le /25 en binaire
Capteur à plat	18 k $\Omega$	2,29V	245	6143 0b 0001 0111 1111 1111
Capteur plié à 90°	7300 $\Omega$	1,8V	131	4262 0b 0001 0000 1010 0110
Capteur totalement plié	2300 $\Omega$	560mV	83	489 0b 0001 1110 1001

### Explication

Quand on prend le capteur seul (sans le module ADC et sans le diviseur de tension), on peut observer que la valeur du capteur diminue.

A 90 degrés, la valeur du capteur est environ à 50% de la valeur du capteur à plat.

C'est la même chose pour la tension : quand on est à 90 degrés, on est environ à 50% de la valeur du capteur à plat.

On observe également que le capteur peut aller de 6143 à 489. Ces valeurs signifient la position de mon doigt.

J'ai également calculé la sensibilité de mes doigts.

La tension est de 3,3V.

L'exposant « n » correspond au nombre de bits. Dans mon projet, c'est 16 bits pour le CAN.

Grâce à ces valeurs, je peux calculer la sensibilité des doigts.

$$Q = \text{Tension d'entrée} / 2^n - 1$$

$$Q = 3,3V / 2^{16} - 1$$

$$Q = 50 \mu V$$

On observe que mes capteurs ainsi que la résistance de 8k2 forment un diviseur de tension.  
On peut donc calculer la tension qu'il y aura sur les entrées analogiques :

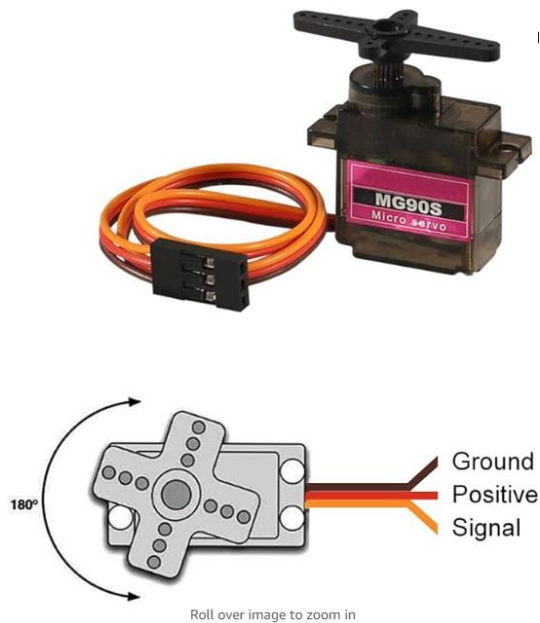
#### **Minimum**

- $U_{\text{Capteur totalement plié}} = (RV / RV + 8200) \times 3,3V$
- $U_{\text{Capteur totalement plié}} = (18000 / 18000 + 8200) \times 3,3V$
- $U_{\text{Capteur totalement plié}} = 722mV$

#### **Maximum**

- $U_{\text{Capteur totalement à plat}} = (RV / RV + 8200) \times 3,3V$
- $U_{\text{Capteur totalement à plat}} = (2300 / 2300 + 8200) \times 3,3V$
- $U_{\text{Capteur totalement à plat}} = 2,27V$

### 5.3 Etude des servomoteurs

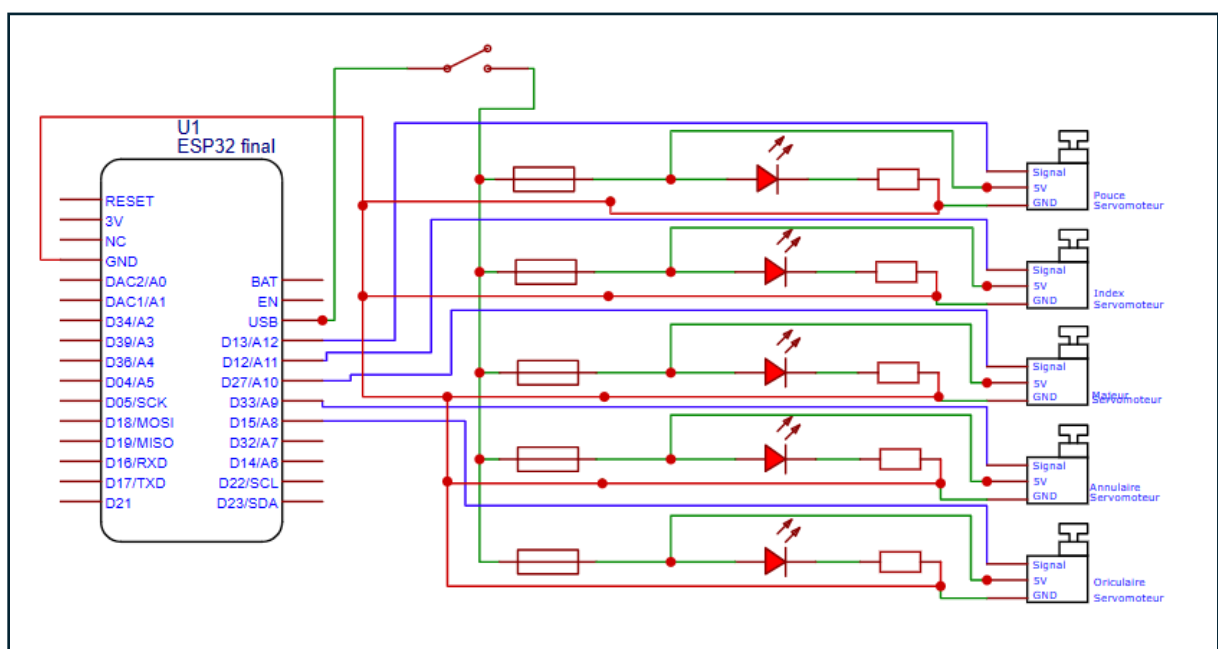


Source : <https://www.amazon.com/Compatible-Raspberry-Project-Helicopter-Airplane/dp/B0925V3X2S?th=1>, consulté le 20 octobre 2023.

#### Explication

Les servomoteurs possèdent trois pins, GND, 5V et la pin signal qui permet de gérer un signal PWM, ce qui va permettre de faire varier la vitesse du servomoteur.

#### Schéma d'utilisation des moteurs



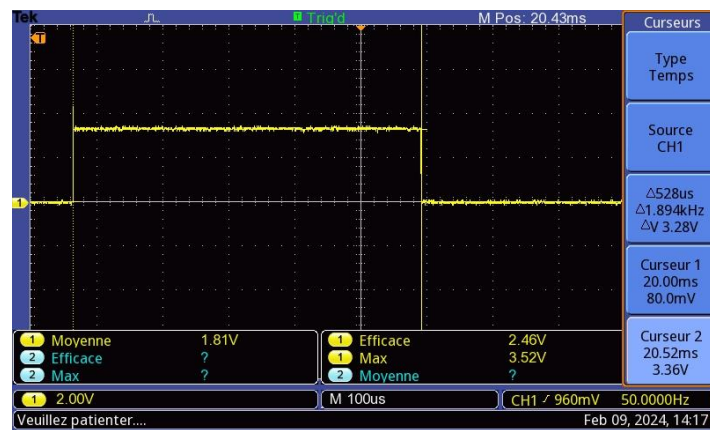
### 5.3.1 Mesure des servomoteurs

Six captures d'écran

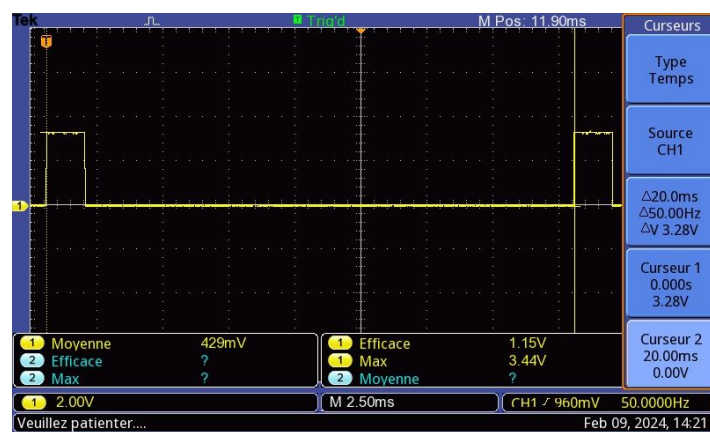
0°



Zoom sur Ton

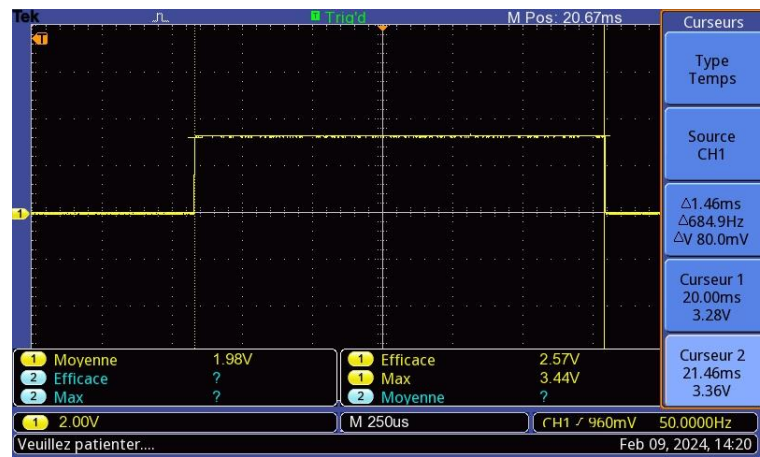


90°

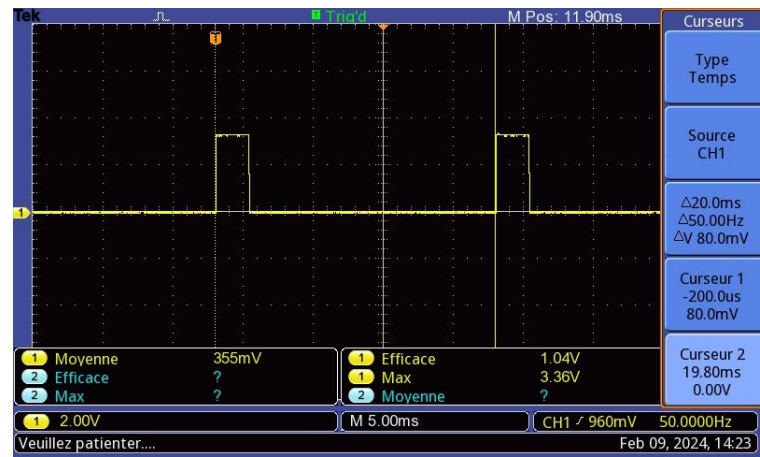




Zoom sur T on



180°



Zoom sur T on

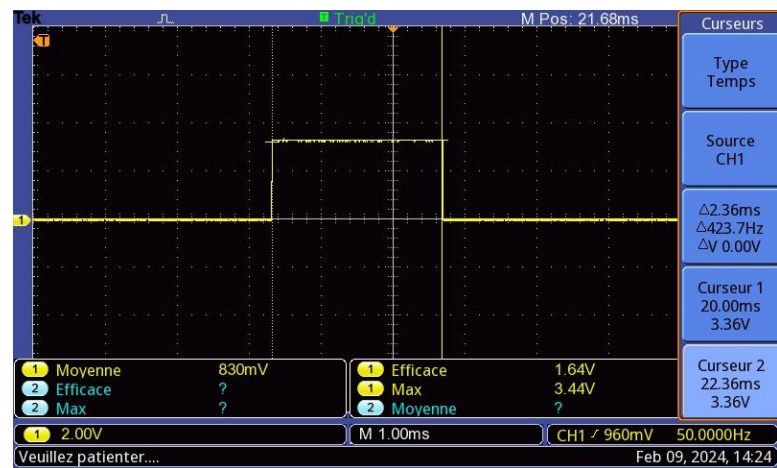


Tableau des mesures

	T (ms)	f (Hz)	t <sub>on</sub> (ms)	Rapport cyclique r (%)	Umoy calculé
0°	20ms	50 Hz	0,528ms	$0,528/20 \times 100 = 2,64\%$	0,08V
90°	20ms	50Hz	1.46ms	$1,46/20 \times 100 = 7,3 \%$	0.23V
180°	20ms	50Hz	2.36ms	$2.36/20 \times 100 = 11,8\%$	0.38V

### **Explication**

On observe que le T ne change pas.

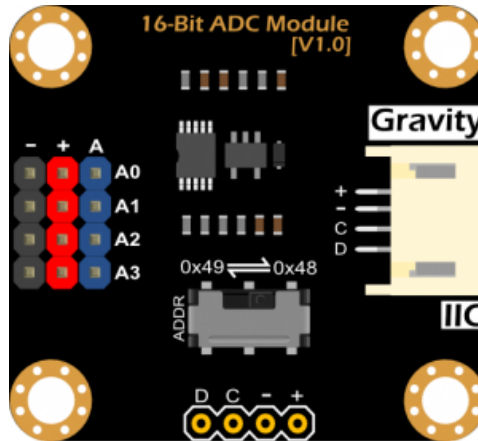
La fréquence est également la même pour tous les servos.

On peut également dire que plus l'angle est grand, plus le temps d'allumage (T on) est long.

## 5.4 L'ADS 1115

Dans le cadre de mon projet, l'utilisation des modules ADS 1115 m'est très utile.

Il me permet de convertir la tension analogique et numérique pour que ma carte puisse les lire.



Source : [ADS1115](#) , consulté le 27 mai 2024

### Caractéristique

- Alimentation : 3,3/5V
- Gnd
- SCL : Sérial clock
- SDA : Sérial data
- 4 entrées analogiques : A0, A1, A2, A3
- 2 adresses : 0x48 et 0x49 qui permettent de mettre 2 modules sur le bus I2C
- Travaile sur 16 bits

### Schéma de fonctionnement des ADS 1115

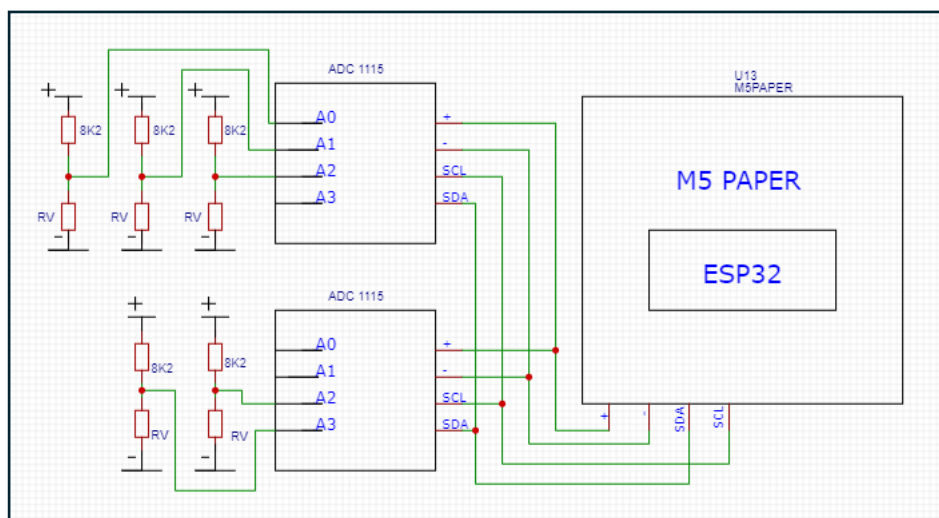
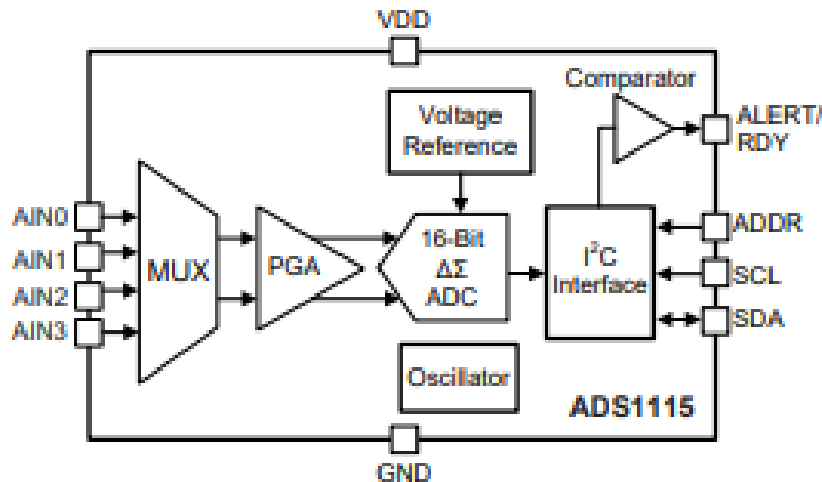


Schéma interne de l'ADS 1115



Source : [ADS1115](#), consulté le 29 mai 2024.

Nous retrouvons le schéma interne du module ADS 1115.

Au centre du schéma, nous pouvons retrouver le bloc de conversion analogique-numérique (CAN) de 16 bits. Il va convertir les signaux analogiques (tension) en valeurs numériques (bits). La résolution est sur 16 bits. Cela signifie que l'on peut représenter 65536 niveaux de tension différents.

En entrée, on retrouve nos entrées analogiques reliées au multiplexeur. Ce multiplexeur permet de choisir l'entrée analogique que l'on souhaite.

```
adc7 = ads2.readVoltage(3);
```

Dans l'intérieur de cette fonction se passe le choix de l'entrée à choisir.

Après le multiplexeur, se trouve un PGA (Amplificateur programmable). Il permet d'amplifier le signal d'entrée. On peut également régler le gain de 1x, 2x, 4x, 8x pour adapter la plage de tension mesurée.

A droite du bloc de conversion, on retrouve une interface I2C qui permet de communiquer avec 2 fils seulement SDA et SCL.

Au-dessus de l'I2C se trouve un comparateur qui va comparer la tension d'entrée avec la tension de référence. Il va générer une alerte si la tension dépasse un seuil défini.

On retrouve également une horloge interne (oscillateur). L'ADS 1115 utilise une horloge interne pour synchroniser la conversion. Cela permet de garantir une précision lors de la conversion.

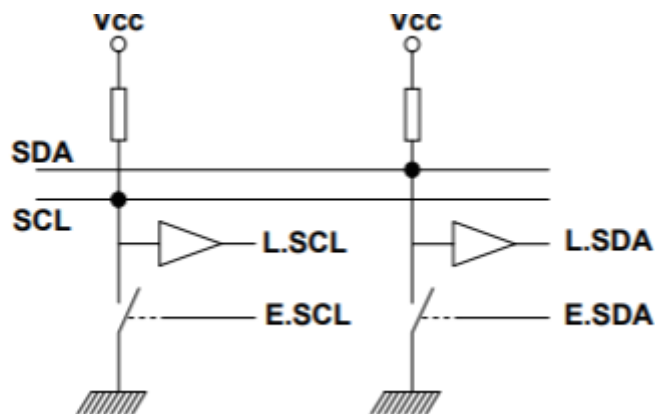
## 5.5 Le bus I2C

Le bus I2C, également connu sous le nom de bus Inter Integrated Circuit, est un canal de communication série synchrone composé de trois fils principaux :

- SDA (Signal de Données)
- SCL (Signal d'Horloge)
- La masse

Son rôle essentiel est de faciliter la communication entre différents circuits intégrés présents sur un même PCB. Ce bus opère à une vitesse de transfert de données de 100 Kbits/sec dans sa version standard et de 400 Kbits/sec dans sa version étendue.

Principe du bus I2C



Source : cours d'électronique.

Cette image ne concerne qu'un module : le premier comparateur permet de lire le signal d'horloge. En-dessous, le transistor permet d'écrire sur le signal d'horloge. Le deuxième comparateur permet de lire les données et le transistor permet d'écrire les données.

L'une des caractéristiques distinctives du système I2C est sa nature multi-maîtres. Cela signifie que n'importe quel circuit intégré peut prendre le contrôle du bus et agir en tant que maître. En tant que tel, il est capable d'initier des opérations de lecture ou d'écriture avec d'autres circuits, qui agissent alors en tant qu'esclaves.

L'état initial est SDA et SCL qui sont tous deux, au niveau haut.

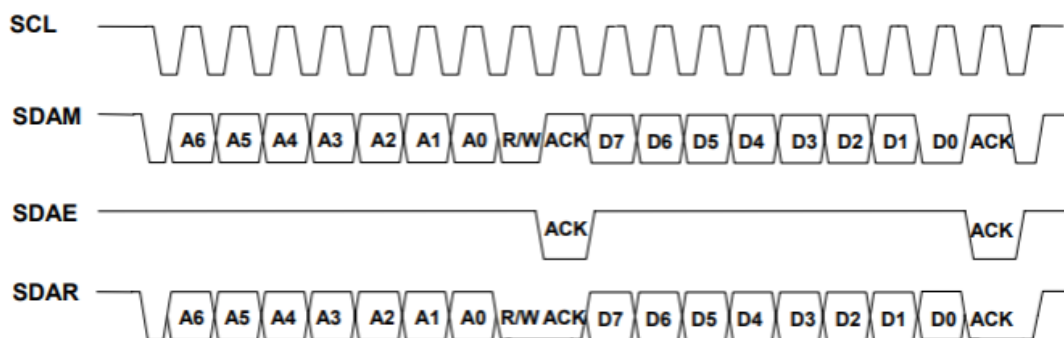
Le processus d'échange de données commence lorsque SDA passe à un niveau bas et SCL est maintenu à un niveau logique haut. Lorsque le signal de données (SDA) passe au niveau bas, le maître commence à émettre des instructions. Pour terminer l'échange, les signaux SDA et SCL doivent tous deux revenir à un niveau logique haut.

Les données sont échangées selon un protocole précis :

Le maître commence par :

1. L'adresse du périphérique ciblé est transmise.
2. Une indication est donnée pour spécifier si l'opération est une écriture ou une lecture.
3. Les données sont transmises.

Echange d'une trame



La première fois, les adresses seront envoyées sous 7 bits, car le huitième est consacré à l'écriture ou lecture. Dans ce cas-ci c'est une écriture ; cependant la deuxième fois qu'il enverra les données, il les enverra sous 8 bits.

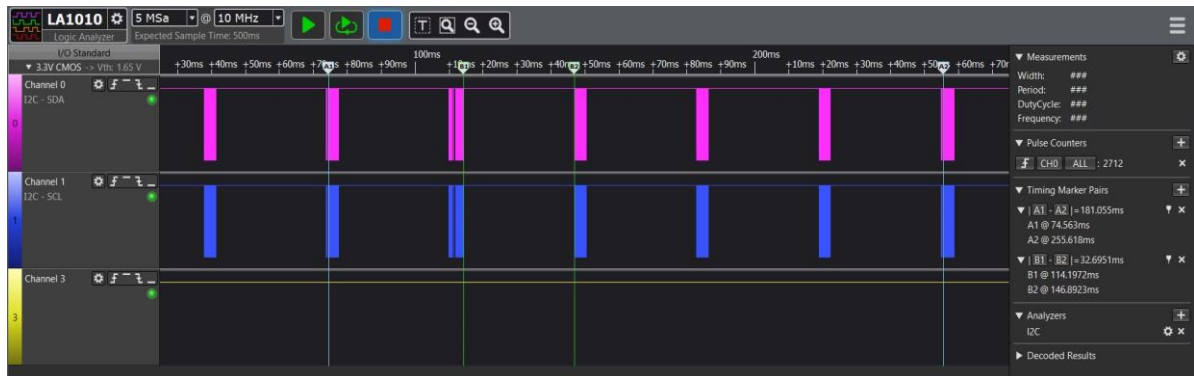
Le bit ACK est l'esclave qui répond au maître pour dire qu'il peut envoyer les informations.

Le maître laisse un niveau haut et l'esclave met un niveau bas. Le maître sait alors que l'esclave lui a répondu.

Ce protocole strict assure un échange fiable et synchronisé entre les composants connectés au bus I2C.

Plusieurs de mes modules utilisent l'I2C. Je l'utilise dans le cas de mes encodeurs, mais encore pour mes modules ADC1115 qui me permettent de convertir de l'analogique en numérique.

## Mesure de l'I2C et interruption



J'ai mesuré le temps que prenait mon interruption et j'ai pu observer qu'elle prenait 182ms et qu'entre chaque doigt il y a 33ms.

## 5.6 Étude de l'écran e-link de la M5STACK PAPER

Mon projet se divise en deux parties, chacune étant équipée d'une carte de programmation similaire avec quelques différences. Les deux cartes sont des ESP32, mais l'une est équipée d'un écran tactile et d'une batterie, celle-ci est la M5STACK PAPER. Son écran unique est un écran e-link, couramment utilisé dans les liseuses.

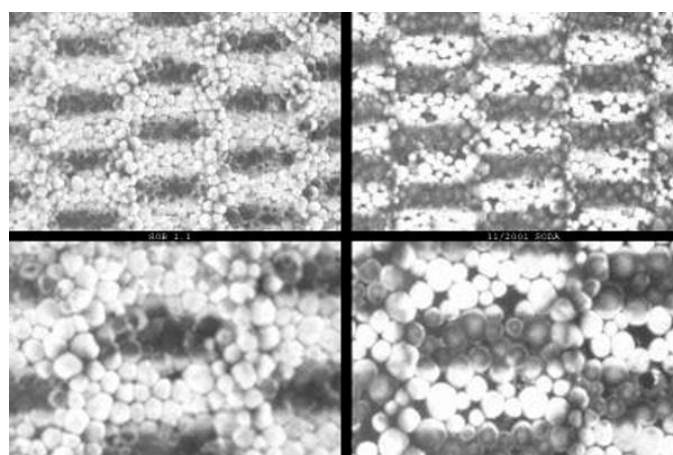
Qu'est-ce qu'un écran e-link ou encre électronique ?

Il s'agit d'un type d'écran qui offre une expérience proche de celle de l'impression papier lors de l'affichage. Contrairement aux écrans traditionnels, il ne nécessite pas de rétroéclairage pour être lisible. Sa surface est réfléchissante, ce qui signifie que la lumière ambiante, comme celle du soleil ou d'une source lumineuse artificielle, permet de visualiser son contenu. L'encre électronique est principalement utilisée sur des surfaces en plastique rigide ou souple, bien qu'elle puisse être utilisée sur d'autres types de surfaces également.

Comment fonctionne un écran e-ink ?

Il est composé de microcapsules ayant la forme de sphères. Ces microcapsules sont situées entre deux feuilles de plastique. Chaque microcapsule contient de l'huile et de petites particules chargées électriquement. Pour afficher la couleur noire par exemple, les particules doivent être chargées négativement, tandis que pour afficher du blanc, elles doivent être chargées positivement. Lorsqu'un courant électrique négatif est appliqué sur une capsule, les particules blanches se dirigent vers une extrémité de la capsule et les particules noires vers l'autre extrémité, permettant ainsi de passer de la couleur blanche à la couleur noire lors de l'affichage.

Sur l'image ci-dessous, on retrouve les particules qui constituent l'écran e-link



Source : <https://www.liseuses.net/comment-fonctionne-encre-electronique/>, consulté le 11 mai 2024.



Pour obtenir des nuances de gris, différentes proportions de particules blanches et noires sont utilisées.

Exemple : 75% de blanc et 25% de noir ou 50% de blanc et 50% de noir, etc.

Une fois que le courant électrique est appliqué, les particules restent en place, ce qui signifie que l'écran n'a plus besoin d'être alimenté en énergie une fois l'affichage réalisé. Un écran e-link est composé de millions de ces microcapsules. Pour afficher d'autres couleurs, un filtre est placé au-dessus des capsules, formant une matrice de filtres permettant d'afficher plus de 4096 couleurs différentes.

## 5.7 Encodeur EC11

Concernant la partie manuelle de mon projet, j'utilise l'encodeur « unit 8encodeur ». Ces encodeurs me permettent de contrôler les 5 doigts de ma main comme mon gant. Les encodeurs utilisés sont les EC11 de type optique. Ils sont incrémentaux et envoient les informations en I2C au microcontrôleur ESP32.

Les encodeurs sont donc similaires aux potentiomètres, mais ils possèdent 4 fils : deux pour l'alimentation et deux pour les signaux.

Comment fonctionne l'encodeur ?

L'encodeur est constitué de deux parties : le rotor et le stator.

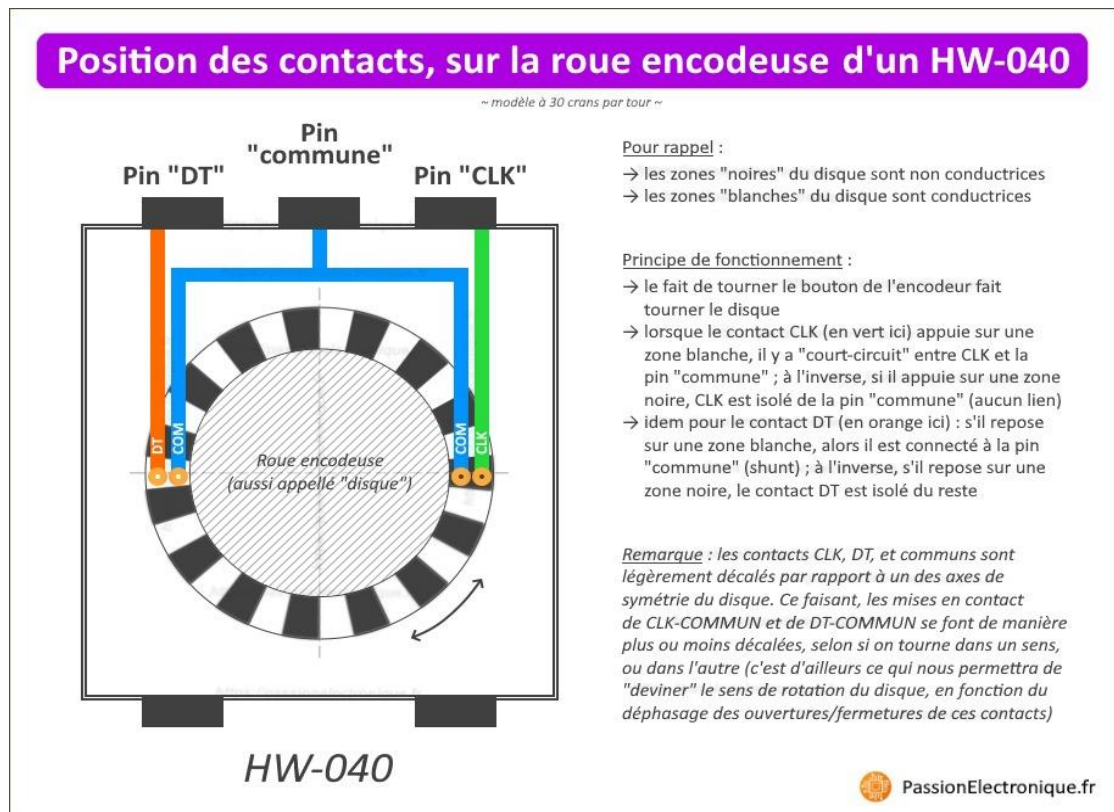
Le rotor est la partie rotative qui est reliée à l'élément dont on souhaite mesurer la position, la vitesse ou la distance.

Le stator est la partie fixe qui contient le capteur permettant de mesurer les données souhaitées, telles que la vitesse ou la position.

Principe de fonctionnement des encodeurs rotatifs incrémentaux

Un encodeur fonctionne selon les étapes suivantes :

- Rotation et contacts électriques : Le bouton rotatif central de l'encodeur est relié à une roue encodeuse qui est partiellement conductrice. Deux contacts électriques reposent sur cette roue. En tournant le bouton, ces contacts vont s'ouvrir et se fermer selon leur position sur les zones conductrices ou isolantes du disque.

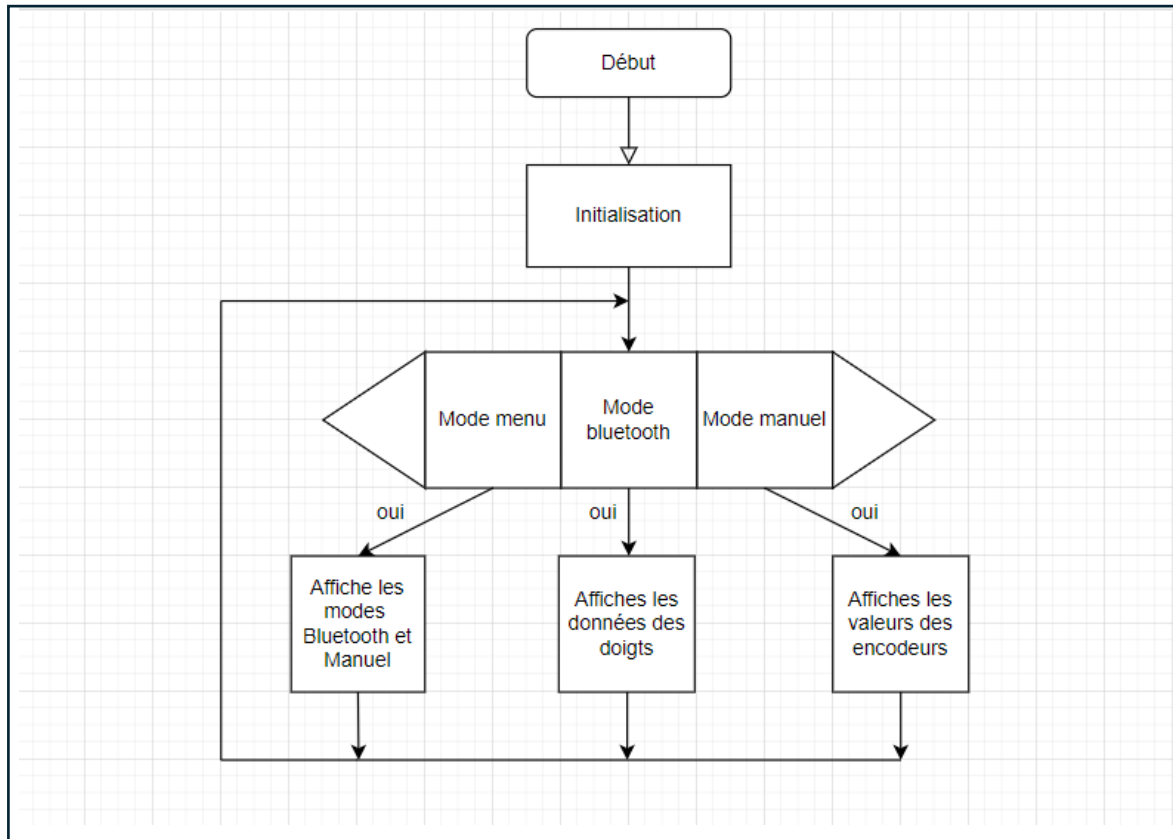


Source : [Encodeur](#), consulté le 15 mai 2024.

- Déphasage des signaux : les contacts électriques (CLK et DT, ou A et B) sont utilisés de telle sorte que si on les tourne, ils s'ouvrent et se ferment de manière déphasée. Ceci permet de déterminer le nombre de crans parcourus, mais également le sens de rotation.
- Transmission de données : les changements d'état des contacts sont convertis en signaux électriques qui sont envoyés en I2C au microcontrôleur. Ces signaux vont nous permettre de calculer la position, la vitesse,...

## 6. La programmation

### 6.1 L'ordinogramme général de la transmission



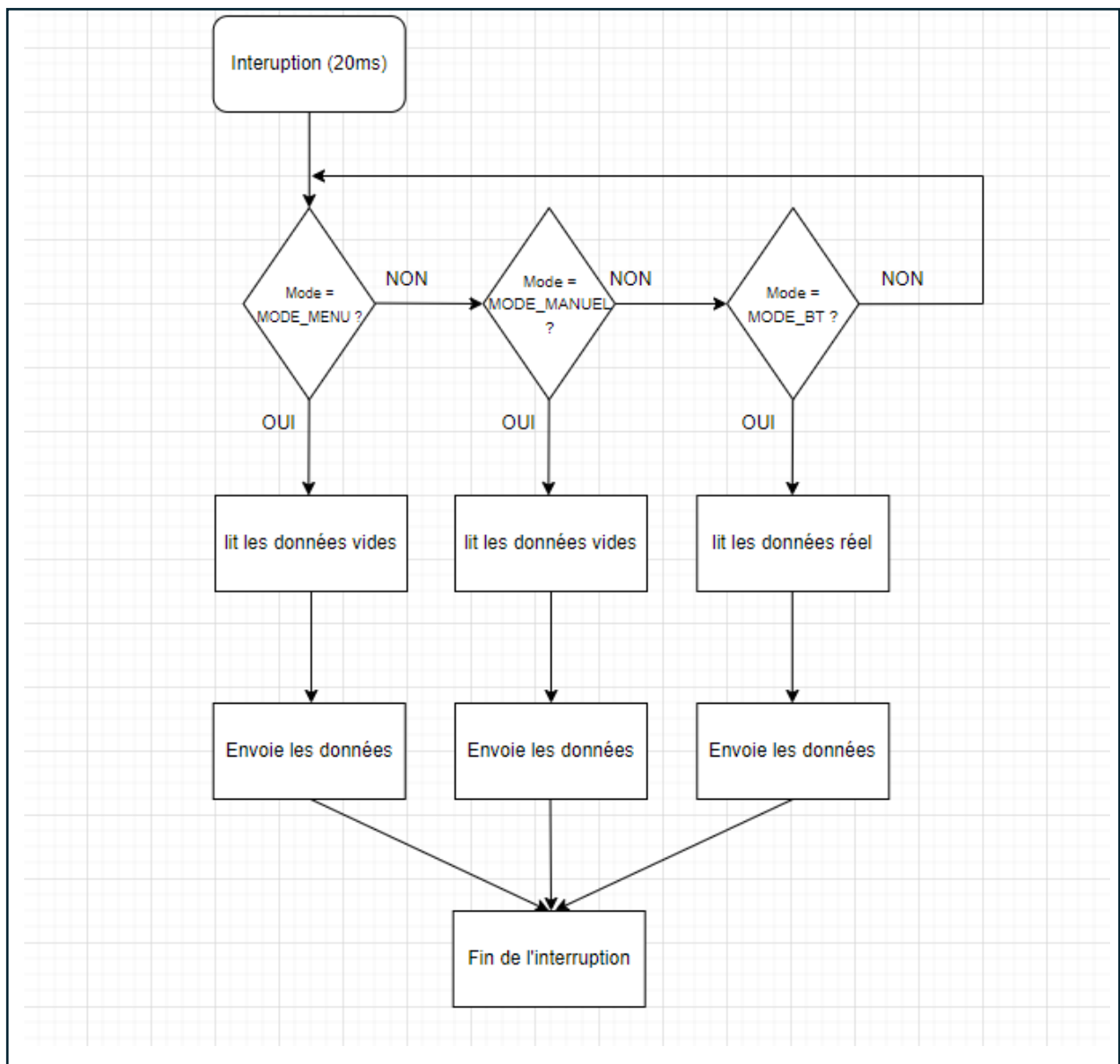
Dans mon programme principal, je choisis le mode que je veux et, ensuite, j'affiche les données.

Pour le mode Bluetooth, j'affiche les données de mes capteurs.

Pour le mode menu, j'affiche les différents modes à choisir.

Pour le mode manuel, j'affiche la position des encodeurs.

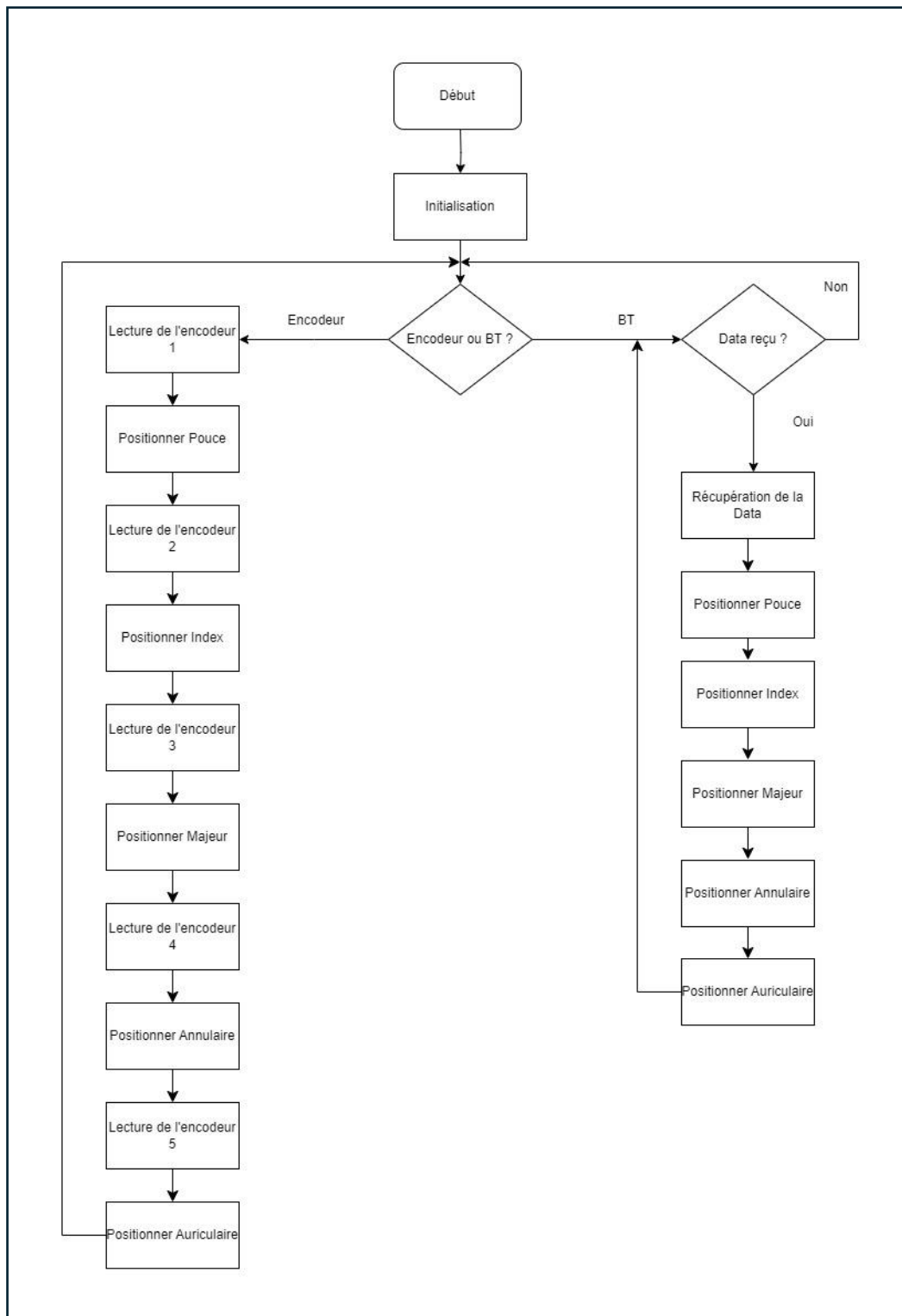
## 6.2 L'interruption



Pour le mode Bluetooth je lis et envoie les données de mes modules ADC (Analog to digital converter).

Si, le caractère initial du mode menu ou mode Bluetooth n'est pas reçu correctement, le récepteur peut détecter cette erreur grâce au contrôle d'intégrité et à la structure du paquet et il peut prendre des mesures appropriées (comme demander une retransmission).

## 6.2 Réception



L'ordinogramme de réception a été fait par Daoud Oulad El Fadel.

## 6.4 Le programme

```
void IRAM_ATTR onTimer(void *param) {
    digitalWrite(PIN_G33, HIGH); //1

    if (Mode == MODE_BT) {
        if (ads2.checkADS1115()) { // gère les données du deuxième module

            adc7 = ads2.readVoltage(3);
            Serial.printf("A03:%5d ", adc7 / 25);

            adc6 = ads2.readVoltage(2);
            Serial.printf("A02:%5d ", adc6 / 25);
        } else {
            Serial.println("ADS1115-gauche Disconnected!");
        }

        if (ads1.checkADS1115()) { // gère les données du premier module

            adc2 = ads1.readVoltage(2);
            Serial.printf("A12:%5d ", adc2 / 25);
            adc1 = ads1.readVoltage(1);
            Serial.printf("A11:%5d ", adc1 / 25);
            adc0 = ads1.readVoltage(0);
            Serial.printf("A10:%5d \n", 255 - adc0 / 25);
        } else {
            Serial.println("ADS1115-droite Disconnected!");
        }

        sendtab[0] = '#';
        sendtab[1] = char(255 - adc0 / 25);
        sendtab[2] = char(adc1 / 25); // divisée par 25 car pas besoin de 6123
positions.
        sendtab[3] = char(adc2 / 25);
        sendtab[4] = char(adc6 / 25);
        sendtab[5] = char(adc7 / 25);
        sendtab[6] = sendtab[1] ^ sendtab[2] ^ sendtab[3] ^ sendtab[4] ^
sendtab[5]; //byte de controle d'intégrité de donnée
        for (char cptSend = 0; cptSend < NB_data; cptSend++) {
            SerialBT.print(sendtab[cptSend]);

#ifdef debug_ADC
            Serial.println(sendtab[cptSend]);
#endif
        }
    }
}
```

```

if(Mode== MODE_MENU){
    sendtab[0] = '&';
    sendtab[1] = 0;
    sendtab[2] = 0;
    sendtab[3] = 0;
    sendtab[4] = 0;
    sendtab[5] = 0;
    sendtab[6] = sendtab[1] ^ sendtab[2] ^ sendtab[3] ^ sendtab[4] ^
sendtab[5]; //byte de controle d'intégrité de donnée
for (char cptSend = 0; cptSend < NB_data; cptSend++) {
    SerialBT.print(sendtab[cptSend]);
}
}
if(Mode== MODE_MANUEL){

    sendtab[0] = '@';
    sendtab[1] = 0;
    sendtab[2] = 0;
    sendtab[3] = 0;
    sendtab[4] = 0;
    sendtab[5] = 0;
    sendtab[6] = sendtab[1] ^ sendtab[2] ^ sendtab[3] ^ sendtab[4] ^
sendtab[5]; //byte de controle d'intégrité de donnée

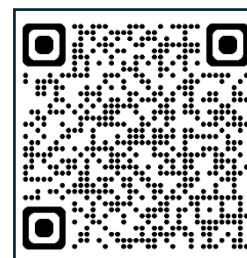
for (char cptSend = 0; cptSend < NB_data; cptSend++) {
    SerialBT.print(sendtab[cptSend]);
}
}
digitalWrite(PIN_G33, LOW); //0
}

```

Ci-dessus, vous pouvez retrouver mon interruption dans laquelle, si je choisis le mode manuel ou BT, je lis et j'envoie mes données sous forme de tableau. Le premier caractère du tableau me permet de reconnaître, lors du programme de réception, le mode dans lequel je suis.

Pour voir l'intégralité de mon code, je vous invite à cliquer sur ce lien Github dans lequel vous retrouverez une vidéo du projet ainsi que quelques explications.

Lien Github: [Thomas2809/transmission \(github.com\)](https://github.com/Thomas2809/transmission).





## 7. La fabrication

Le commencement de mon projet a débuté par la sélection des composants appropriés. Avec mon promoteur, nous avons opté pour deux module ADC, un convertisseur analogique-numérique, idéal pour traiter les données provenant de mes capteurs de flexion afin que le microcontrôleur puisse les interpréter avec précision. Ensuite, sur les conseils avisés de mon promoteur, j'ai opté pour une carte de programmation M5Stack Paper équipée d'un écran tactile, une décision qui m'a immédiatement fait imaginer plusieurs options pour la suite.

Par la suite, j'ai entrepris l'assemblage méticuleux de la structure du gant connecté pour faciliter les tests de mon programme. Une fois cette étape franchie, j'ai développé un programme permettant de lire les données provenant de mes capteurs.

La phase suivante a été consacrée à l'intégration du Bluetooth, celle-ci est essentielle pour établir une communication efficace entre nos deux cartes afin de transmettre les données. J'ai donc suivi une formation (auprès de mon professeur M. Kapita durant les vacances de Noël) sur le Bluetooth et étudié un code de test pour comprendre son fonctionnement. Ensuite, nous avons travaillé sur le programme de réception.

Après cette étape, j'ai testé l'ensemble des programmes et constaté que les données étaient transmises, mais à une vitesse insatisfaisante. Mon promoteur m'a alors recommandé d'ajouter une interruption pour garantir une transmission beaucoup plus rapide des données.

Avec les deux programmes opérationnels, j'ai pu me concentrer sur la fabrication de la structure de la main, en commençant par la conception d'un circuit imprimé (PCB). Pendant l'attente de la livraison du PCB, j'ai pu concevoir le support de la main.

Une fois la mécanique de mon projet entièrement finalisée, j'ai commencé à concevoir un menu permettant d'intégrer tous les modes de fonctionnement.

## 8. La mise au point

Tout au long de la conception de mon projet, je n'ai pas rencontré de problèmes importants avec ma main robotisée, mais uniquement quelques détails à régler ou plutôt à améliorer.

Au niveau de la mécanique de la main, je n'ai pas rencontré de problème mais uniquement des améliorations nécessaires, que j'ai apportées au fur et à mesure de la finalisation de mon projet.

Dans un premier temps, je n'avais pas prévu de support pour accueillir ma batterie permettant d'alimenter ma main.

J'ai donc réfléchi à une amélioration en redessinant ma pièce 3D afin d'y accueillir une batterie externe qui allait alimenter celle-ci.

De ce fait, ma main robotisée est totalement autonome.

J'ai rencontré un problème de connexion avec mes modules ADC.

Le problème était que je devais constamment reconnecter mes câbles à mes modules ADC, car ceux-ci ne tenaient pas bien et pour le transport j'étais obligé de les déconnecter.

J'ai donc opté pour une fiche qui me permet de connecter mes capteurs au module sans problème. J'ai également protégé les câbles avec de la gaine tressée afin de ne pas exercer une force sur ceux-ci.

Là où j'ai rencontré un peu plus de problèmes est au niveau de mon code.

Le problème se situait au niveau des encodeurs. Si je voulais utiliser mes encodeurs, je devais constamment faire plusieurs tours avant que mes encodeurs ne trouvent leurs plages de fonctionnement, ce qui posait un problème. Je me suis donc renseigné et j'ai trouvé une ligne de code me permettant de résoudre ce problème.

```
MM.setAbsCounter(1, 220); // le numéro de l'encodeur, la limite
```

## 9. La conclusion

De manière générale, la conception de la main robotisée s'est très bien déroulée.

Je suis très satisfait et surtout très fier du résultat final de mon projet.

Étant motivé par mon projet et ayant appris énormément de choses durant mes deux dernières années, l'envie d'améliorer la main robotisée en réalisant un bras complet m'est venu à l'idée.

La réalisation de ce projet m'a permis de me lancer divers défis tout au long de sa réalisation.

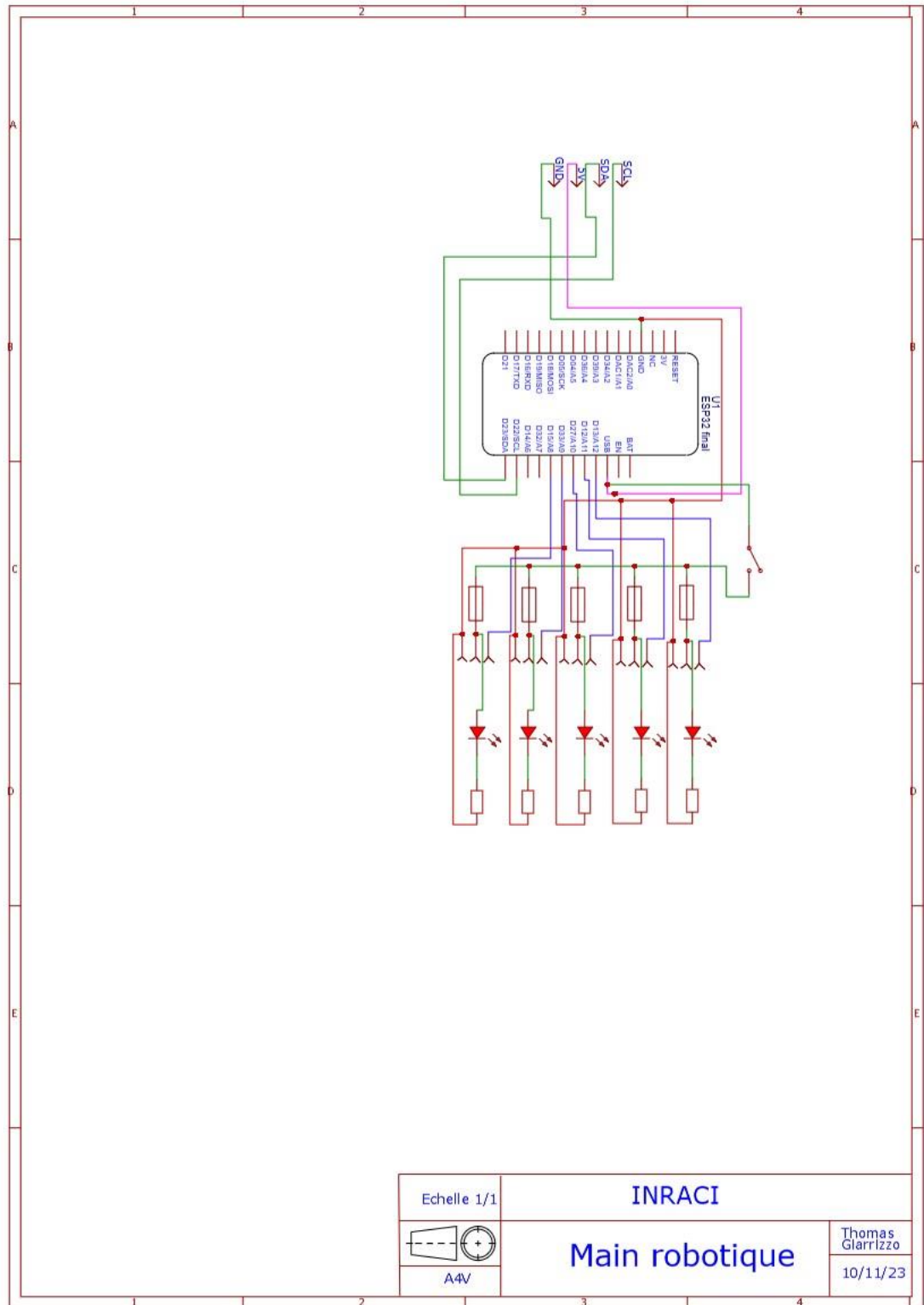
Le fait d'avoir réussi à mener ceux-ci à bien, m'encourage énormément à continuer mes études afin d'explorer d'autres branches de l'électronique, de la mécanique,...

Il y a deux ans, mon idée était de terminer ma sixième année et d'aller travailler dans la société de mes parents.

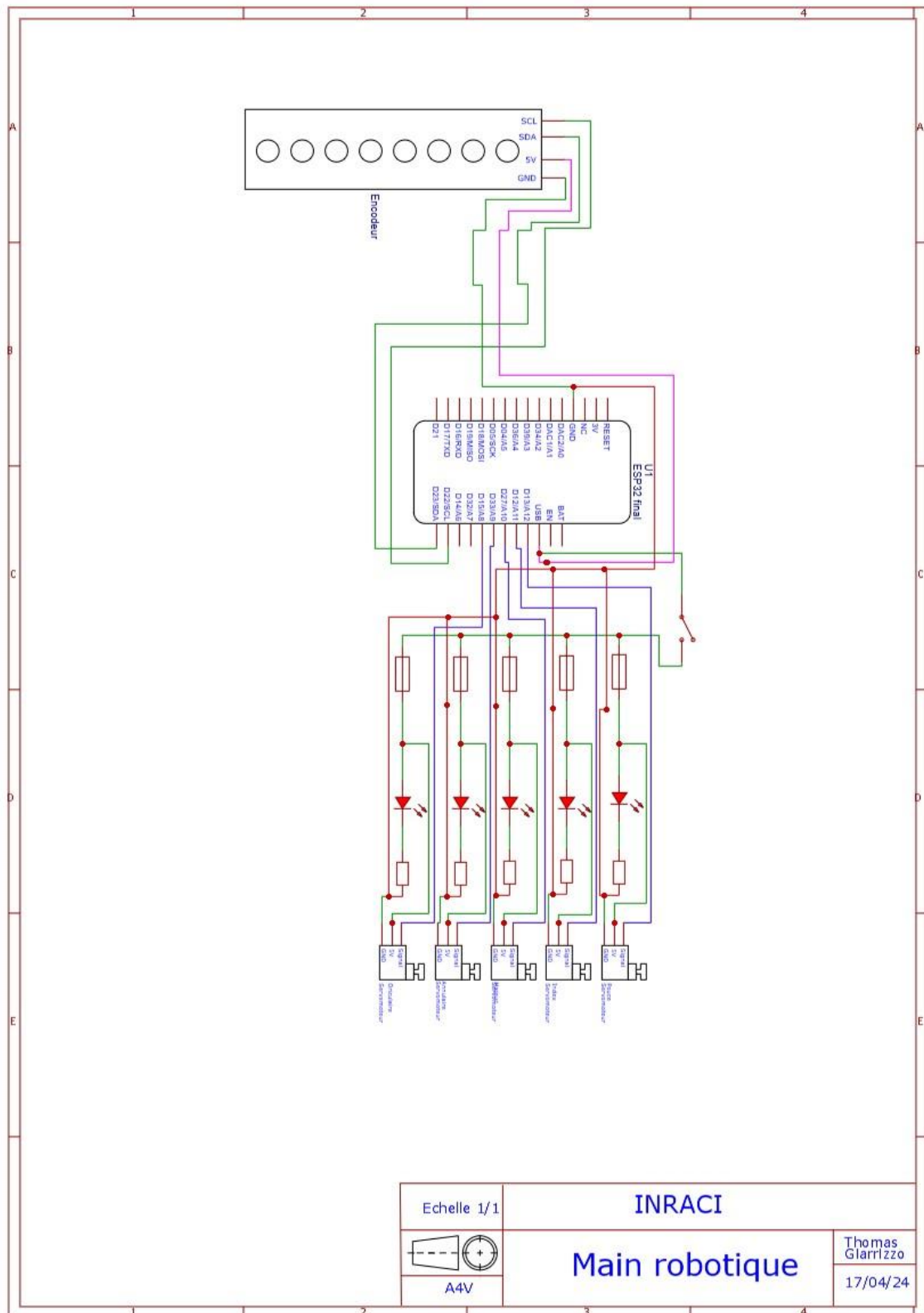
La réalisation de mon projet m'a permis de me rendre compte de mes capacités et m'a motivé à continuer mes études.

## Les annexes

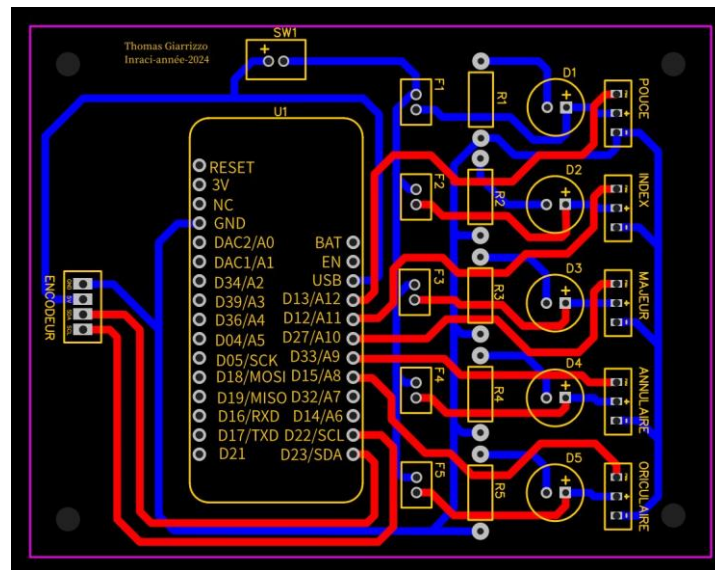
### 1. Schéma de principe sans connecteur



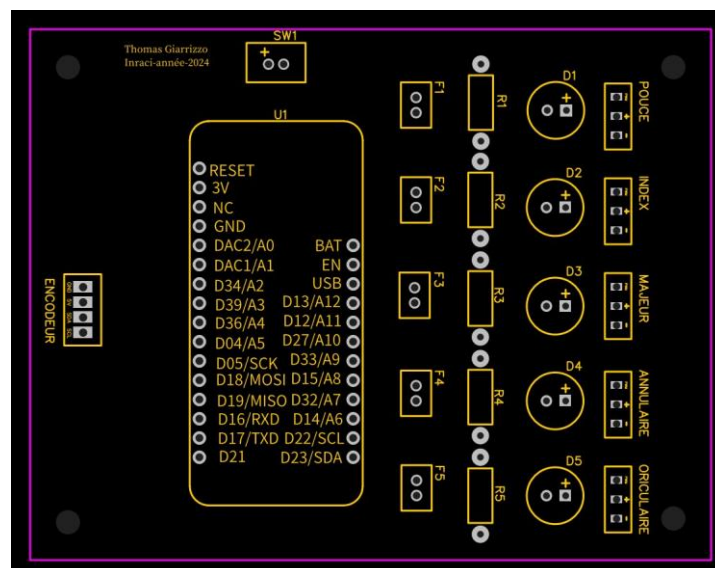
## 2. Schéma de principe avec connecteur



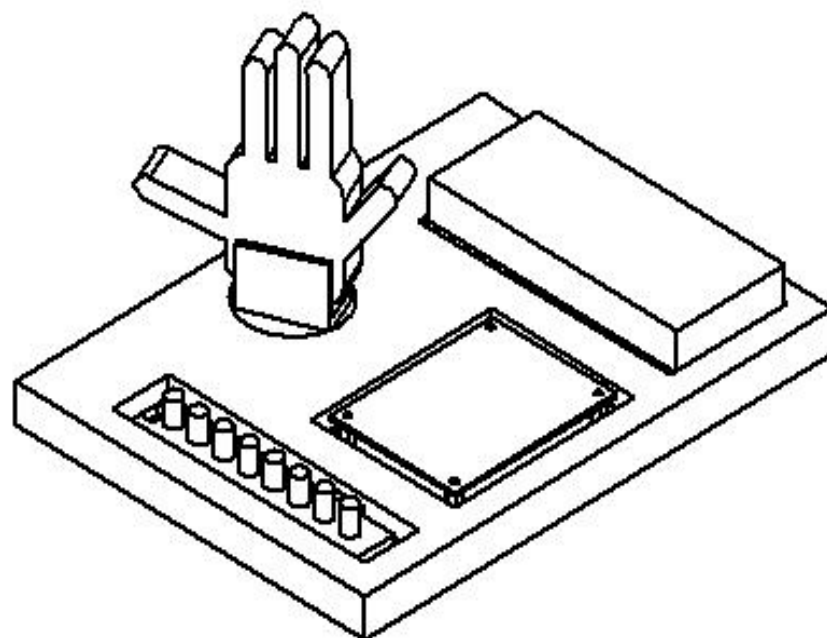
### 3. PCB



### 4. Sérigraphie du PCB

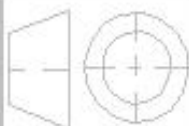


## 5. Dessin d'implantation dans le boîtier



Echelle 1/2

Inraci



A4V

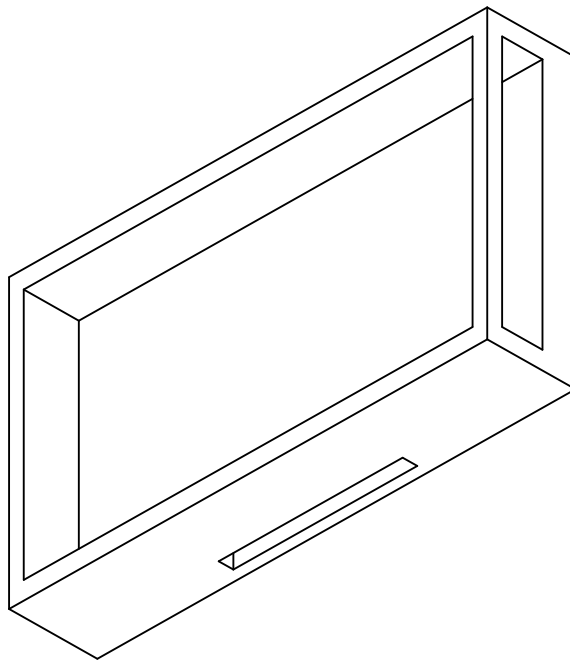
Main robotisée

Thomas  
Gianizzo

7/02/24

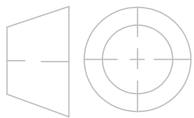
## 6. Vue 3D

Support m5 paper



Echelle 1/1

Inraci



A4V

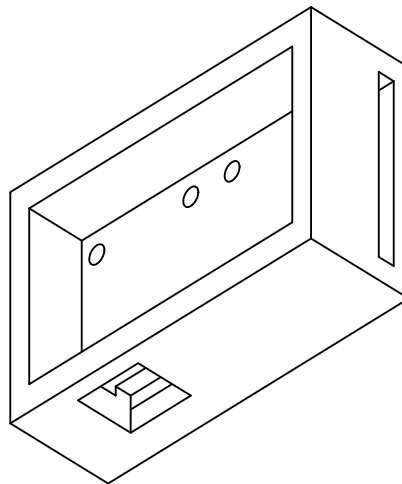
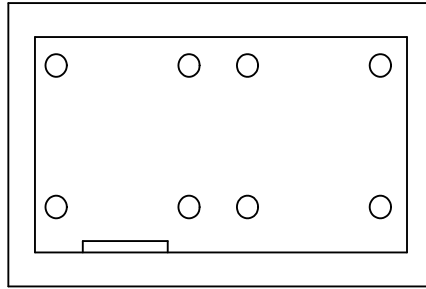
Support M5PAPER

Thomas  
Giarrizzo

19/11/23



## 6.1 Support pour module adc1115



Echelle 1/1

Inraci



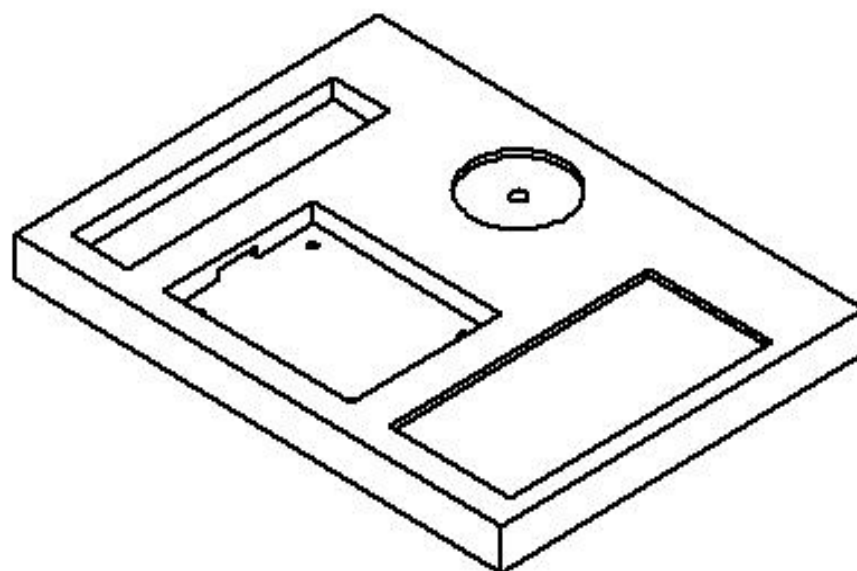
A4V

Support ADS

Thomas  
Giarrizzo

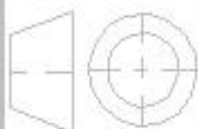
16/11/ 23

## 6.2 Support pour la main robotique



Echelle 1/1

Inraci



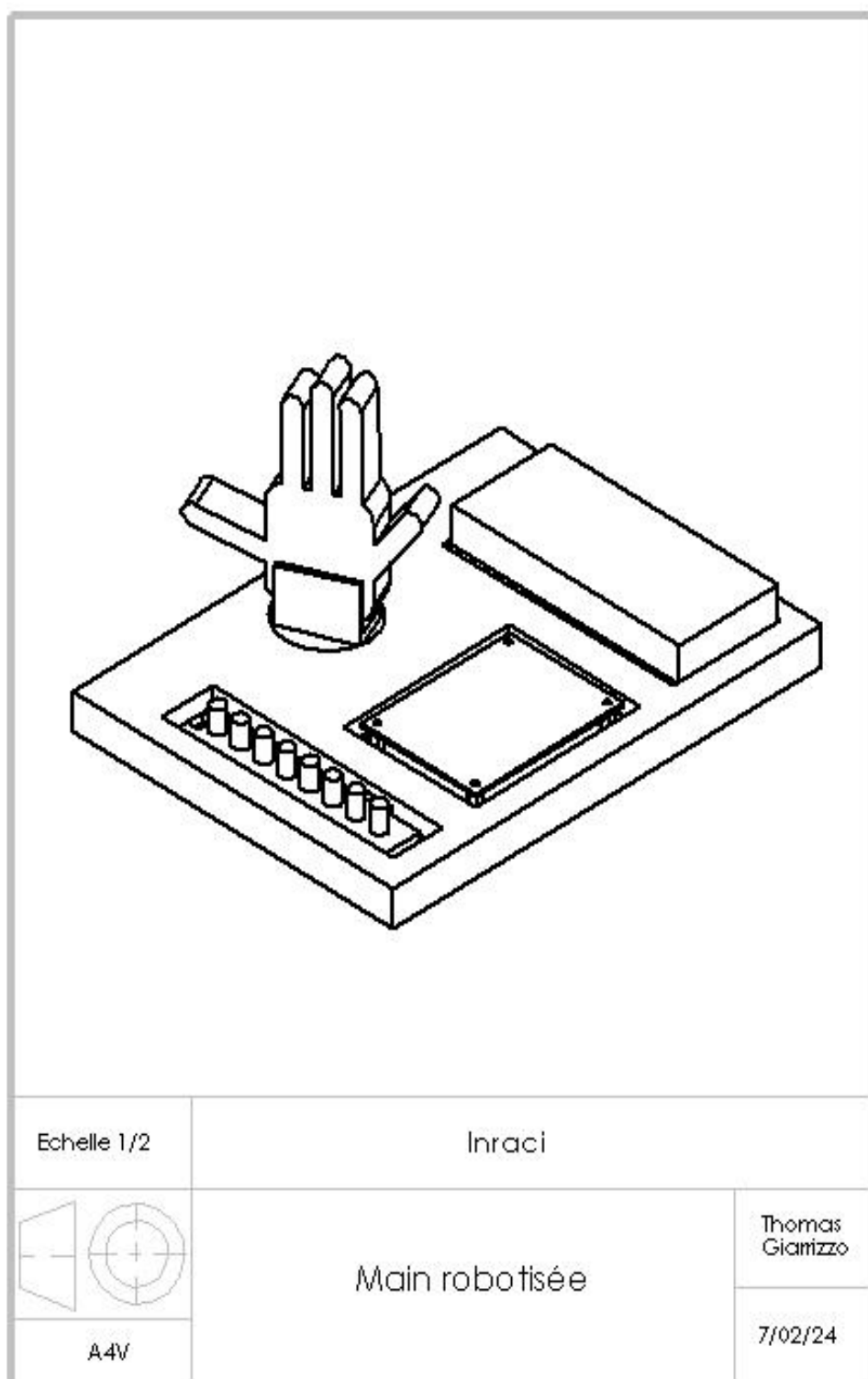
A4V

Support main robotisée

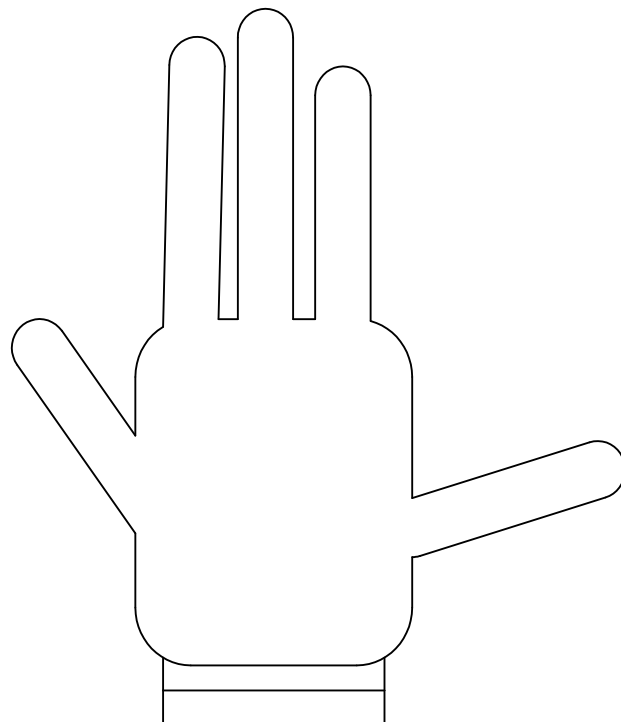
Thomas  
Giarizzo

8/02/24

### 6.3 Vue d'ensemble du projet assemblé



#### 6.4 Vue de la main robotique



Echelle 1/1

Inraci



A4V

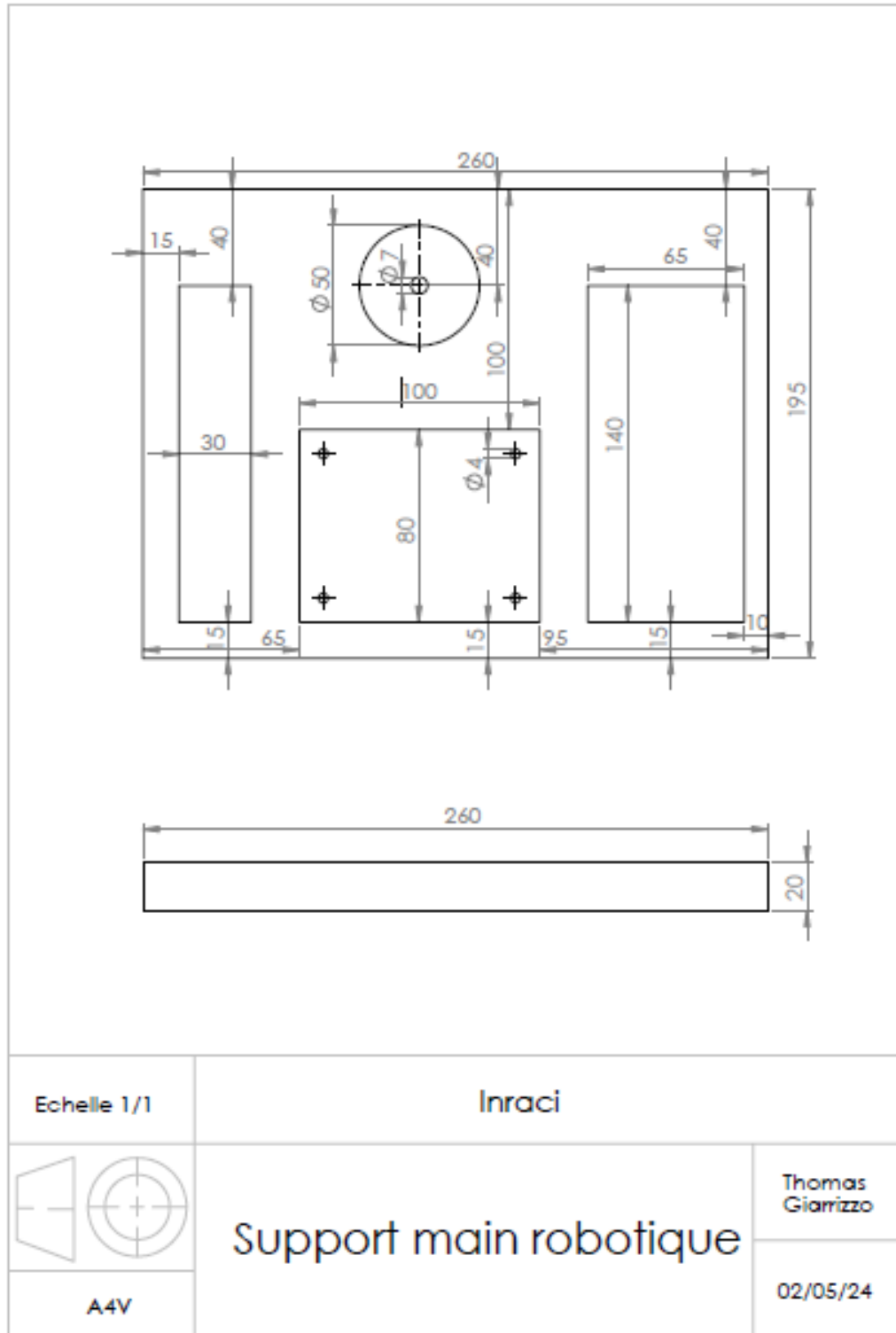
Main robotique

Thomas  
Giarrizzo

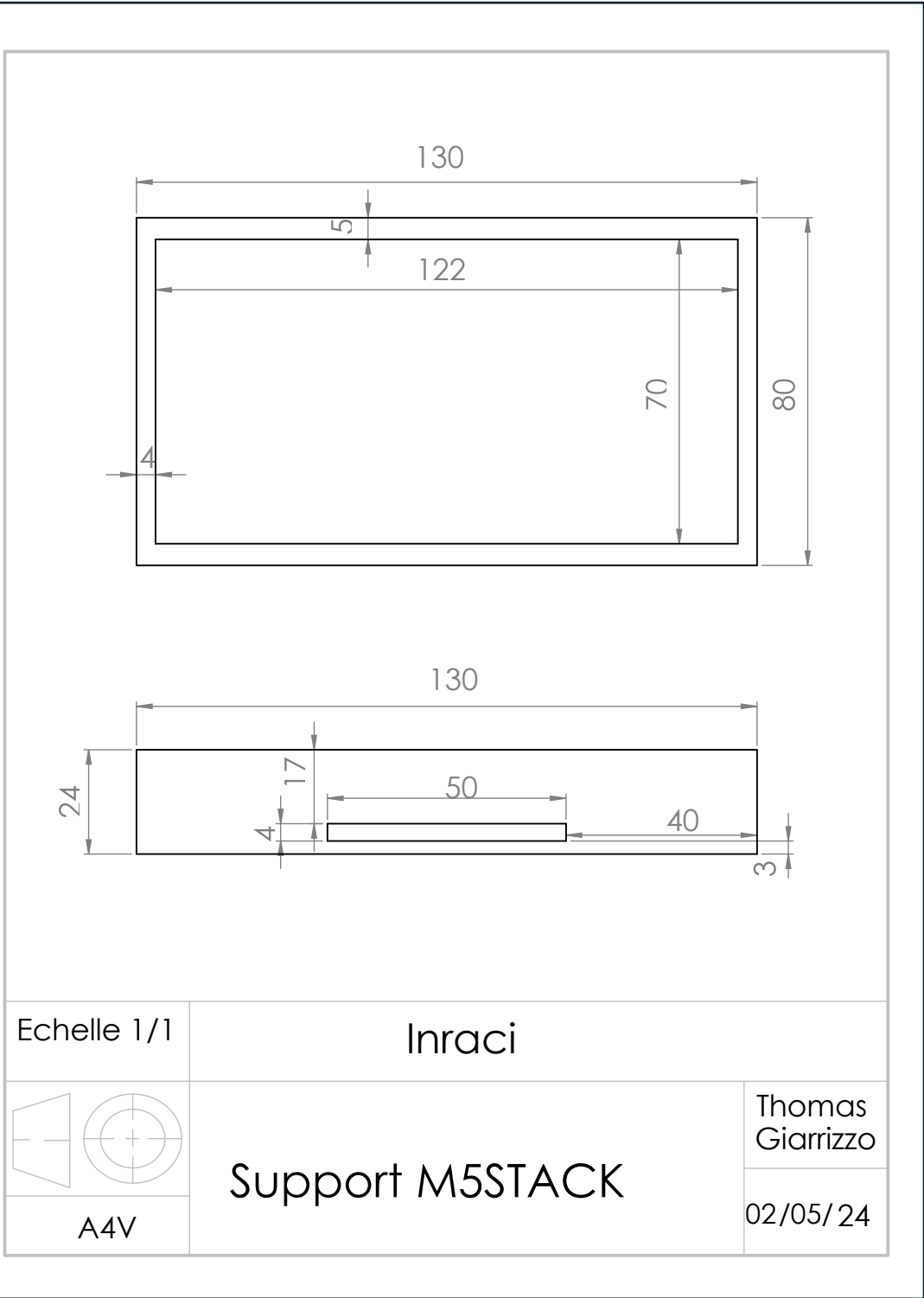
23/05/ 24

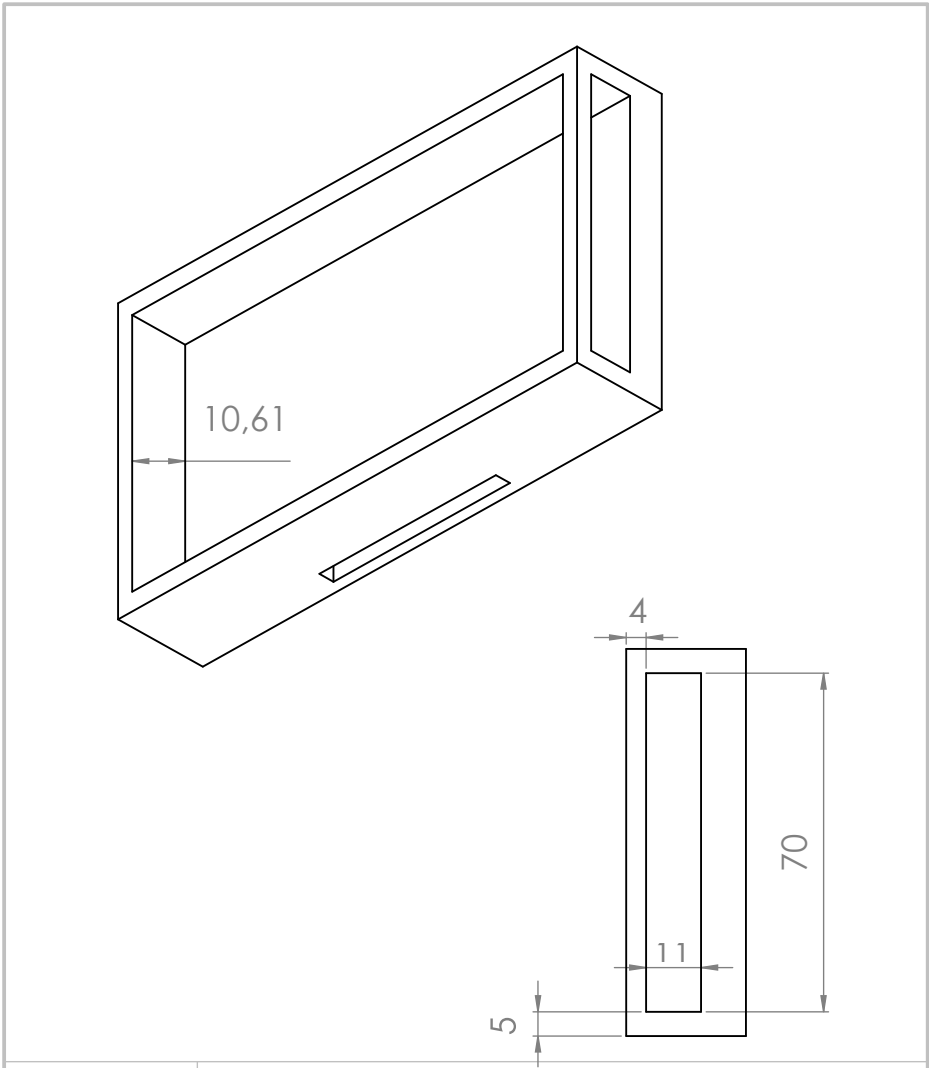
## 7. Mise en plan


### Support pour la main robotique



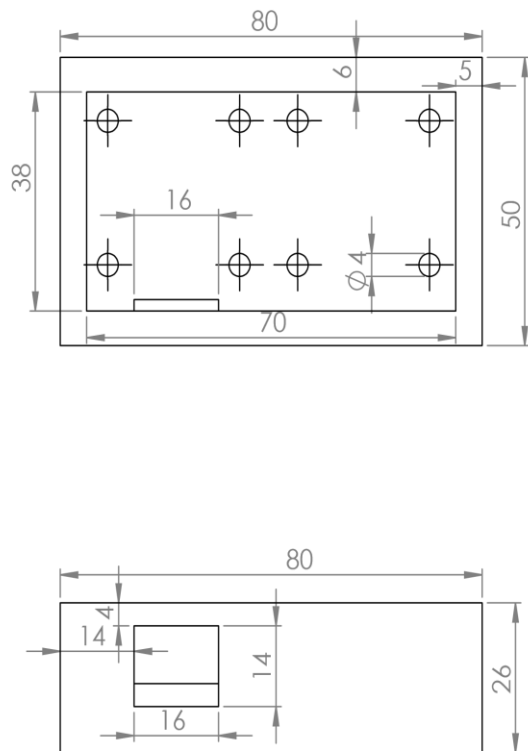
7.1 Support m5 paper





Echelle 1/1	Inraci	
 A4V	Support M5STACK PAPER	
	Thomas Giarrizzo 02/05/24	

7.2 Support module ADC 1115



Echelle 1/1

Inraci



A4V

Support ADS

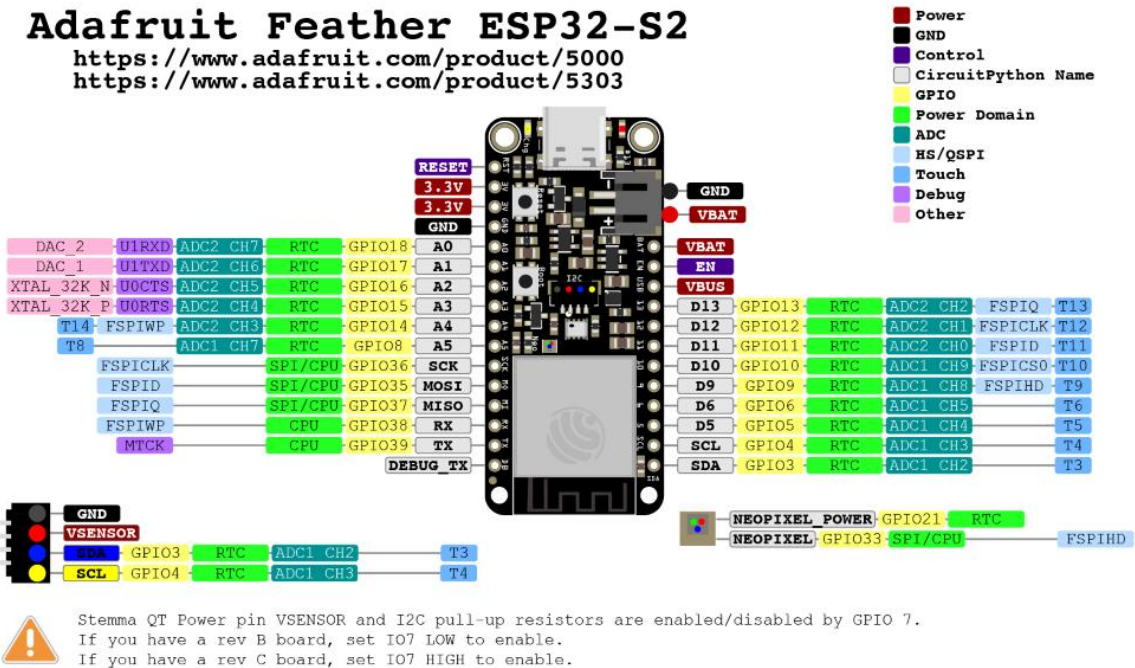
Thomas  
Giarrizzo

02/05/24



## 8. Fiches techniques des composants peu courants

### 8.1 Datasheet Feather ESP32



Source : [Pinouts | Adafruit ESP32-S2 Feather | Adafruit Learning System](#), consulté le 17 mars 2024

## 8.2 Datasheet M5stack paper

Resources	Parameter
ESP32-D0WDQ6-V3	240MHz dual core, 600 DMIPS, 520KB SRAM, Wi-Fi
Flash	16MB
PSRAM	8MB
Input Voltage	5V @ 500mA
Ports	TypeC*1, HY2.0-4P*3 , TF-card(microSD) slot
E-Ink Display	Model Number:EPD_ED047TC1   540*960@4.7"   Gray scale : 16 Levels   Display area : 58.32*103.68mm   Display Driver : IT8951E
Physical Button	Multi-function button*1 , Reset Button*1
RTC	BM8563
Antenna	2.4G 3D Antenna
PINS	G25, G32, G26, G33, G18, G19
Battery	1150mAh@3.7V
Working Temp	0°C to 60°C

Net Weight	86g
Gross Weight	100g
Product Dimension	118*66x*10mm
Packaging Dimension	120*70x*14mm
Casing Material	Plastic ( PC )

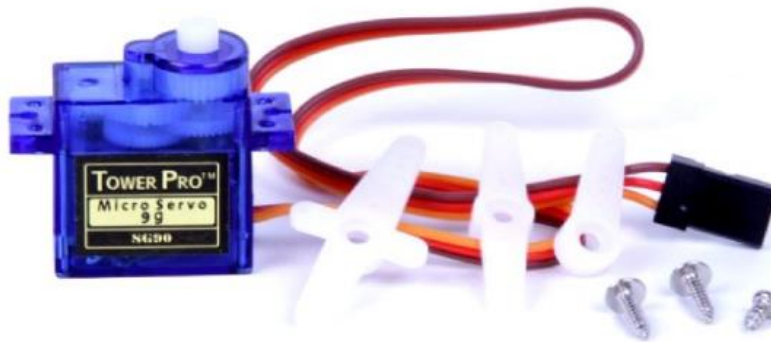
Source : [https://docs.m5stack.com/en/core/m5paper\\_v1.1](https://docs.m5stack.com/en/core/m5paper_v1.1), consulté le 17 mars 2024.

### 8.3 Datasheet servomoteurs

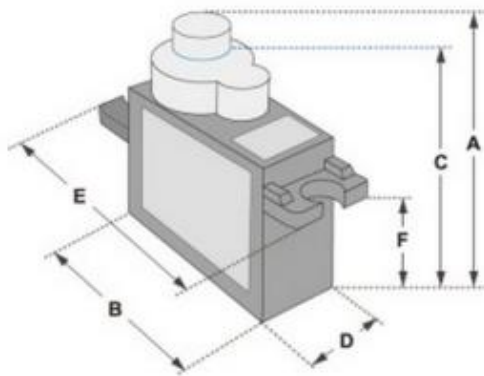
Servomoteur 5-6V

SERVO MOTOR SG90

DATA SHEET



Petit et léger avec une puissance de sortie élevée, le servo peut pivoter d'environ 180 degrés (90 dans chaque direction) et fonctionne comme les types standard, mais plus petit.





Dimensions & Specifications	
A (mm) :	32
B (mm) :	23
C (mm) :	28.5
D (mm) :	12
E (mm) :	32
F (mm) :	19.5
Speed (sec) :	0.1
Torque (kg-cm) :	2.5
Weight (g) :	14.7
Voltage :	4.8 - 6


Source : [SERVO MOTOR SG90 DATA SHEET \(ic.ac.uk\)](https://www.ic.ac.uk/), consulté le 18 octobre 2023.


Autre lien : [SG90.pdf \(akizukidenshi.com\)](https://akizukidenshi.com/)


## 8.4 Datasheet adc 1115


 Product Folder

 Order Now

 Technical Documents

 Tools & Software

 Support & Community



ADS1113, ADS1114, ADS1115  
SBAS444D –MAY 2009–REVISED JANUARY 2018

### ADS111x Ultra-Small, Low-Power, I<sup>2</sup>C-Compatible, 860-SPS, 16-Bit ADCs With Internal Reference, Oscillator, and Programmable Comparator

#### 1 Features

- Ultra-Small X2QFN Package:  
2 mm × 1.5 mm × 0.4 mm
- Wide Supply Range: 2.0 V to 5.5 V
- Low Current Consumption: 150  $\mu$ A  
(Continuous-Conversion Mode)
- Programmable Data Rate:  
8 SPS to 860 SPS
- Single-Cycle Settling
- Internal Low-Drift Voltage Reference
- Internal Oscillator
- I<sup>2</sup>C Interface: Four Pin-Selectable Addresses
- Four Single-Ended or Two Differential Inputs  
(ADS1115)
- Programmable Comparator (ADS1114 and  
ADS1115)
- Operating Temperature Range:  
–40°C to +125°C

#### 2 Applications

- Portable Instrumentation
- Battery Voltage and Current Monitoring
- Temperature Measurement Systems
- Consumer Electronics
- Factory Automation and Process Control

#### 3 Description

The ADS1113, ADS1114, and ADS1115 devices (ADS111x) are precision, low-power, 16-bit, I<sup>2</sup>C-compatible, analog-to-digital converters (ADCs) offered in an ultra-small, leadless, X2QFN-10 package, and a VSSOP-10 package. The ADS111x devices incorporate a low-drift voltage reference and an oscillator. The ADS1114 and ADS1115 also incorporate a programmable gain amplifier (PGA) and a digital comparator. These features, along with a wide operating supply range, make the ADS111x well suited for power- and space-constrained, sensor measurement applications.

The ADS111x perform conversions at data rates up to 860 samples per second (SPS). The PGA offers input ranges from  $\pm 256$  mV to  $\pm 6.144$  V, allowing precise large- and small-signal measurements. The ADS1115 features an input multiplexer (MUX) that allows two differential or four single-ended input measurements. Use the digital comparator in the ADS1114 and ADS1115 for under- and overvoltage detection.

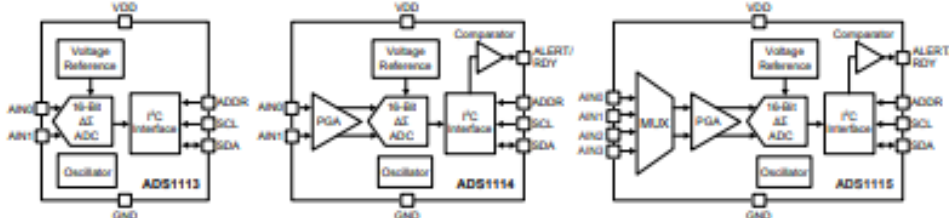
The ADS111x operate in either continuous-conversion mode or single-shot mode. The devices are automatically powered down after one conversion in single-shot mode; therefore, power consumption is significantly reduced during idle periods.

#### Device Information<sup>(1)</sup>

PART NUMBER	PACKAGE	BODY SIZE (NOM)
ADS111x	X2QFN (10)	1.50 mm × 2.00 mm
	VSSOP (10)	3.00 mm × 3.00 mm

(1) For all available packages, see the package option addendum at the end of the data sheet.

#### Simplified Block Diagrams



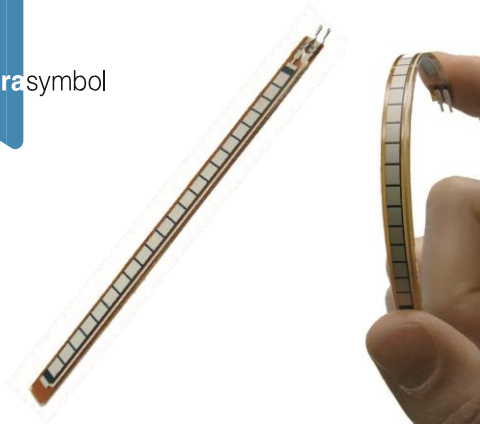
Copyright © 2018, Texas Instruments Incorporated



An IMPORTANT NOTICE at the end of this data sheet addresses availability, warranty, changes, use in safety-critical applications, intellectual property matters and other important disclaimers. PRODUCTION DATA.

Source :[https://www.ti.com/lit/ds/symlink/ads1115.pdf?ts=1713340087408&ref\\_url=https%253A%252F%252Fwww.google.de%252F](https://www.ti.com/lit/ds/symlink/ads1115.pdf?ts=1713340087408&ref_url=https%253A%252F%252Fwww.google.de%252F), consulté le 17 mars 2024.

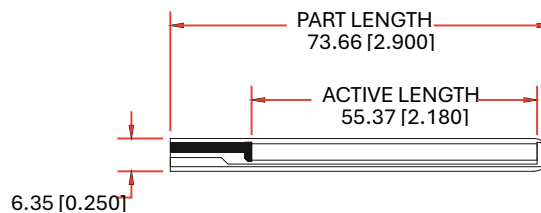
## 8.5 Datasheet capteur de flexion ZD10-100



- Musical Instruments
- Physical Therapy
- Simple Construction

- Bends and Flexes physically with motion device
- Possible Uses
- Robotics
- Gaming (Virtual Motion)
- Medical Devices
- Computer Peripherals

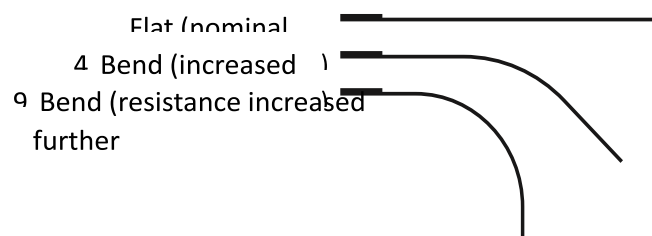
### Dimensional Diagram - Stock Flex Sensor



### How to Order - Stock Flex Sensor

FS	-	L	-	0055	-	253	-	ST
Series		Model		Active Length		Resistance		Connectors
FS = Flex Sensor		L = Linear		0055 = 55.37mm		253 = 25K Ohms		ST = Solder Tab

### How It Works



-Low

### Mechanical

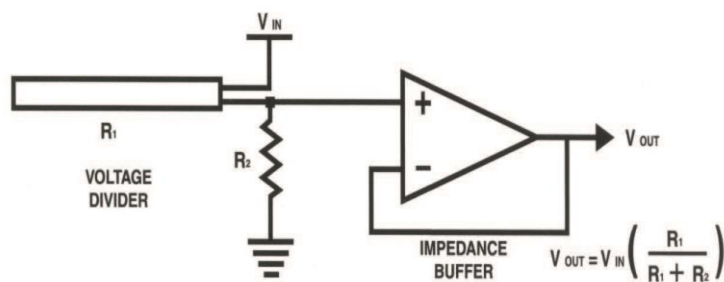
-Life Cycle: >1  
-Height: 0.43mm )  
-Temperature Range: -35°C to

### Electrical

-Flat Resistance: 25K  
-Resistance Tolerance:  
-Bend Resistance Range: 45K to 125K  
(depending on bend  
angle)  
-Power Rating : 0.50  
Watts continuous. 1

### Schematics

#### BASIC FLEX SENSOR CIRCUIT:



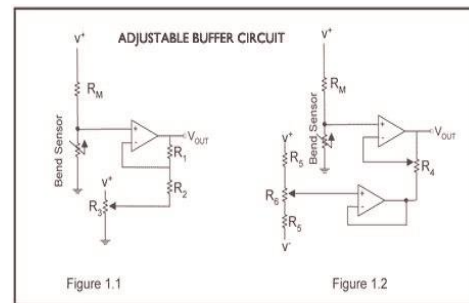
*Following are notes from the ITP Flex Sensor Workshop*

"The impedance buffer in the [Basic Flex Sensor Circuit] (above) is a single sided operational amplifier, used with these sensors because the low bias current of the op amp reduces error due to source impedance of the flex sensor as voltage divider. Suggested op amps are the LM358 or LM324."

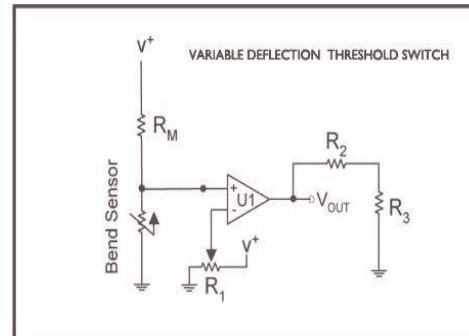
"You can also test your flex sensor using the simplest circuit, and skip the op

**"Adjustable"** - a potentiometer can be added to the

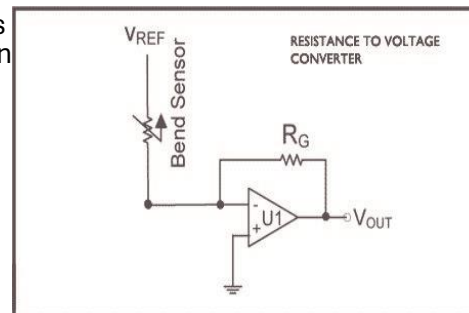
circuit to adjust the sensitivity range."



**"Variable Deflection"** - an op amp is used and outputs either high or low depending on the voltage of the inverting input. In this way you can use the flex sensor as a switch without going through a microcontroller."

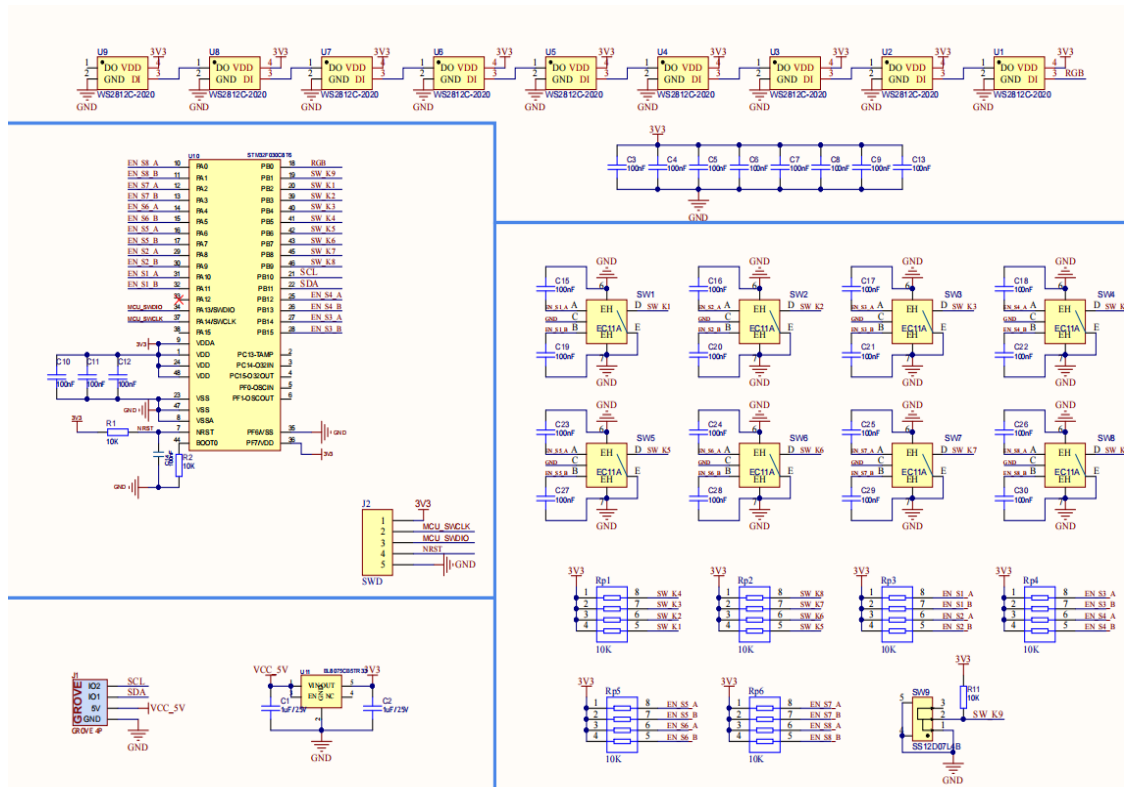


**"Resistance to Voltage"** - use the sensor as input of a resistance to voltage converter using op-amp. A negative reference voltage will give a positive output. Should be used in situations when you want output at a low degree of bending."



## 8.6 Datasheet des encodeurs

Resources	Parameters
MCU	STM32F030C8T6
RGB	WS2812C-2020
Input voltage	5v
I2C communication address	0x41
Product Size	128mm × 24mm × 22.7mm
Package Size	130mm × 27.7mm × 27.7mm
Product Weight	42.8g
Package Weight	52.4g



Source : [m5-docs \(m5stack.com\)](https://m5-docs.m5stack.com/), consulté le 20 janvier 2024.



## Sources du projet

M5STACK PAPER :

- <https://shop.m5stack.com/products/m5paper-esp32-development-kit-v1-1-960x540-4-7-eink-display-235-ppi>, consulté le 21 septembre 2023.



Feather ESP32 :

- <https://www.adafruit.com/product/3405>, consulté le 21 septembre 2023.



Timer :

- <https://passionelectronique.fr/introduction-timer-arduino/>, consulté le 21 septembre 2023.



- <https://passionelectronique.fr/introduction-timer-arduino/> , consulté le 21 septembre 2023.



Interruption :

- <https://polytech-prog.gricad-pages.univ-grenoble-alpes.fr/polytech-microc/documents/cours5.pdf> , consulté le 2décembre 2023.



- <https://www.technologuepro.com/cours-microcontrolleurs-mikroc/chapitre-10-interruptions-en-mikroc.pdf>, consulté le 2décembre 2023



- <https://polytech-prog.gricad-pages.univ-grenoble-alpes.fr/polytech-microc/documents/cours5.pdf>



Fonction réentrante :

- <https://prograide.com/pregunta/12079/quest-ce-quune-fonction-reentrante>, consulté le 2décembre 2023.



- <https://www.tutorialspoint.com/what-is-a-reentrant-function-in-c-cplusplus>, consulté le 2décembre 2023.



- <https://blog.the-pans.com/reentrant/>, consulté le 2décembre 2023.



Emission UART :

- [https://www.rohde-schwarz.com/fr/produits/test-et-mesure/essentials-test-equipment/digital-oscilloscopes/comprehension-uart\\_254524.html](https://www.rohde-schwarz.com/fr/produits/test-et-mesure/essentials-test-equipment/digital-oscilloscopes/comprehension-uart_254524.html), consulté le 18 février 2024.



Contenu sur le Bluetooth :

- <https://lewebpedagogique.com/isneiffel/files/2017/05/Bluetooth.pdf>, consulté le 18 février 2024.



- <http://tvaira.free.fr/bts-sn/activites/activite-ble/activite-ble-esp32.html>, consulté le 18 février 2024.



- <https://elainnovation.com/quest-ce-que-le-bluetooth-low-energy/>, consulté le 18 février 2024.



Capteur ZD10-100 :

- <https://idealblasting.com/thin-film-pressure-sensor-flex-bend-sensor-zd10-100-500g-resistance-type-fsr-sensor-thin-film-pressure-sensor-force-sensing-resistor-force-sensitive-resistor/> , consulté le 21 septembre 2023.



Servomoteur :

- <https://www.amazon.com/Compatible-Raspberry-Project-Helicopter-Airplane/dp/B0925V3X2S?th=1>, consulté le 21 septembre 2023.



Bibliothèque utilisée pour les servomoteurs :

- <https://github.com/jkb-git/ESP32Servo>, consulté le 28 octobre 2023.



Bibliothèque utilisée pour le m5rotate:

- <https://github.com/RobTillaart/M5ROTATE8>, consulté le 17 décembre 2023.



Bibliothèque utilisée pour le M5EPD:

- <https://github.com/m5stack/M5EPD>, consulté le 18 novembre 2023.



Bibliothèque utilisée pour le Bluetooth :

- <https://randomnerdtutorials.com/esp32-bluetooth-classic-arduino-ide/>, consulté le 3 janvier 2024.



#### Datasheet ADS 1115

- [https://www.ti.com/lit/ds/symlink/ads1115.pdf?ts=1713340087408&ref\\_url=https%253A%252F%252Fwww.google.de%252F](https://www.ti.com/lit/ds/symlink/ads1115.pdf?ts=1713340087408&ref_url=https%253A%252F%252Fwww.google.de%252F), consulté le 11 octobre 2023.



#### Datasheet m5stack paper

- [https://docs.m5stack.com/en/core/m5paper\\_v1.1](https://docs.m5stack.com/en/core/m5paper_v1.1), consulté le 21 septembre 2023.



