Brevet De Technicien Supérieur

Épreuve E6.2

ÉTUDE D'UN SYSTÈME

PARTIE ÉLECTRONIQUE

« Rucher connecté »

DOSSIER TECHNIQUE

SOMMAIRE

Présentation du système masse

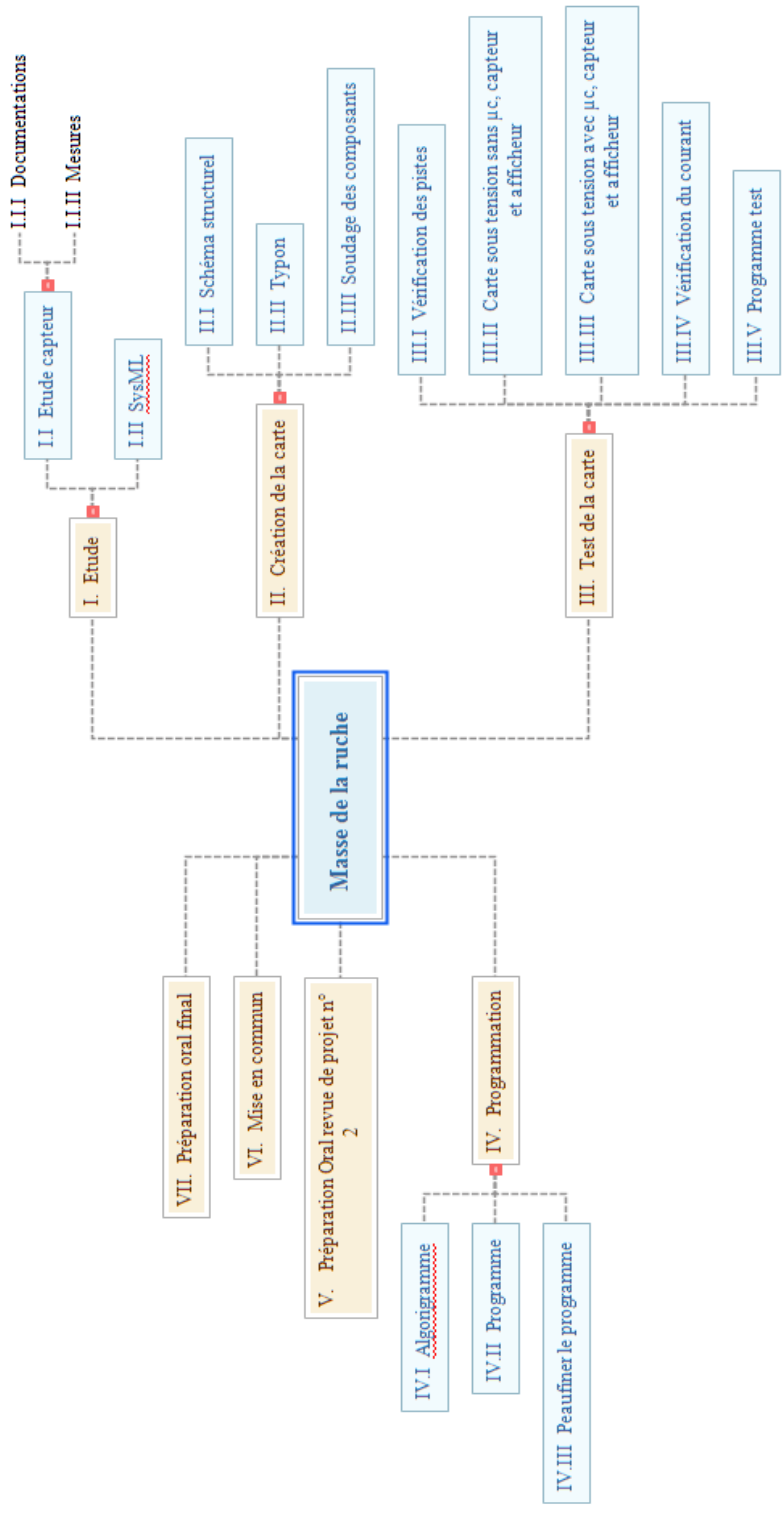
- **Présentation du système**

  Le système de masse du rucher connecté réalise une mesure de la masse pour connaître l'état de la ruche.
  La mesure se fait de deux façons différentes :

- Avec un capteur optique de distance
- Avec potentiomètre linéaire

  De plus il nous a été demander d'ajouter un pour pouvoir tarer.
  Dans cette partie nous ne nous intéresserons seulement au capteur optique de distance.

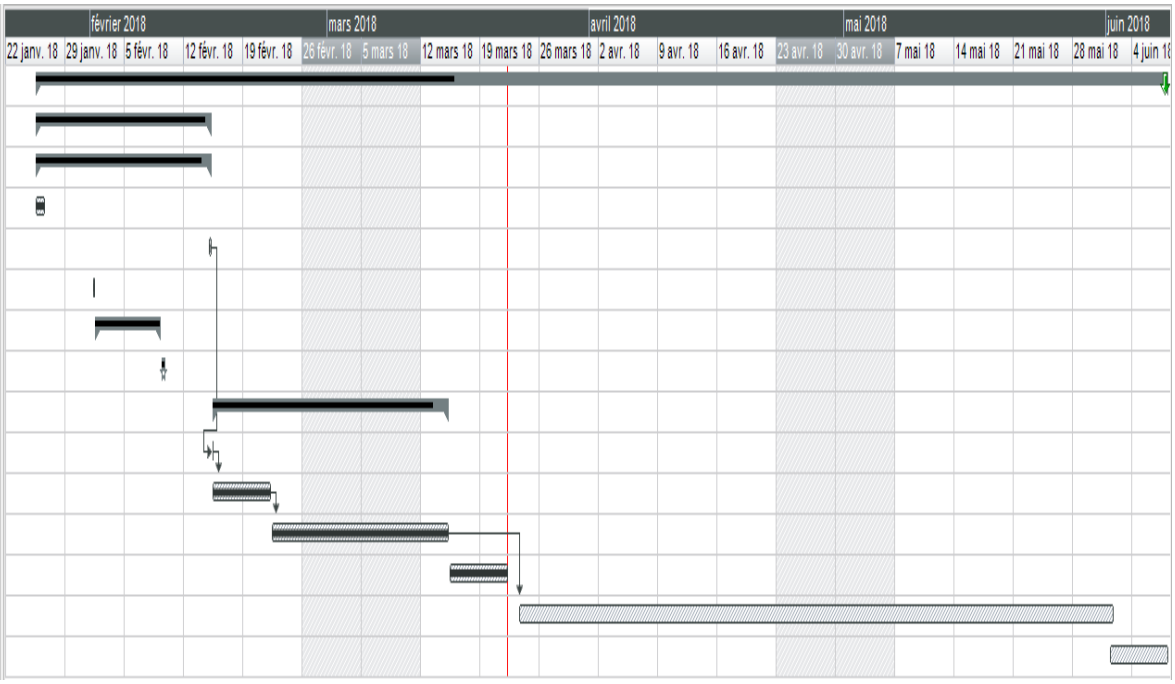
- **Carte Mentale**



Carte mentale réalisé avec le logiciel Mindview

- **Diagramme de GANTT prévisionnel**

	Nom de tâche	Durée	Début	Fin
1	Masse de la ruche	25,37 jours	25/01/2018	08/06/2018
2	⊟I. Etude	5 jours	25/01/2018	15/02/2018
3	⊟I.I Etude capteur	5 jours	25/01/2018	15/02/2018
4	I.I.I Document...	9 hr	25/01/2018	26/01/2018
5	I.I.II Mesures	2 hr	15/02/2018	15/02/2018
6	I.II SysML	4 hr	01/02/2018	01/02/2018
7	⊞II. Création de la...	2,25 jours	01/02/2018	09/02/2018
11	⊞III. Test de la carte	0,5 jour	09/02/2018	09/02/2018
17	⊟IV. Programmation	3,62 jours	15/02/2018	15/03/2018
18	IV.I Algorigramme	1,5 hr	15/02/2018	15/02/2018
19	IV.II Programme	11,5 hr	15/02/2018	22/02/2018
20	IV.III Amélioratio...	13 hr	22/02/2018	15/03/2018
21	V. Préparation Ora...	13 hr	15/03/2018	22/03/2018
22	VI. Mise en commun	105 hr	23/03/2018	01/06/2018
23	VII. Préparation or...	13 hr	01/06/2018	08/06/2018



GANTT réalisé avec le logiciel Mindview

Le projet a donc débuté le 25/01/2018 et s'est achevé le 08/06/2018. Comme le montre la carte mentale ci-dessus et le GANTT, la mise en œuvre du système a duré environ 25 jours et se divise en 4 grandes catégories. L'étude du systèmes, la conception de la carte , puis vient les tests de celle-ci et la programmation (de l'algorigramme à la programmation).

Puis vient la mise en commun du système (donc avec les autres membres du groupe). Cette partie a été relativement plus longue et plus compliqué que la conception du système de la masse.

Fonctionnement du système

Voici quelques schémas pour comprendre le système :

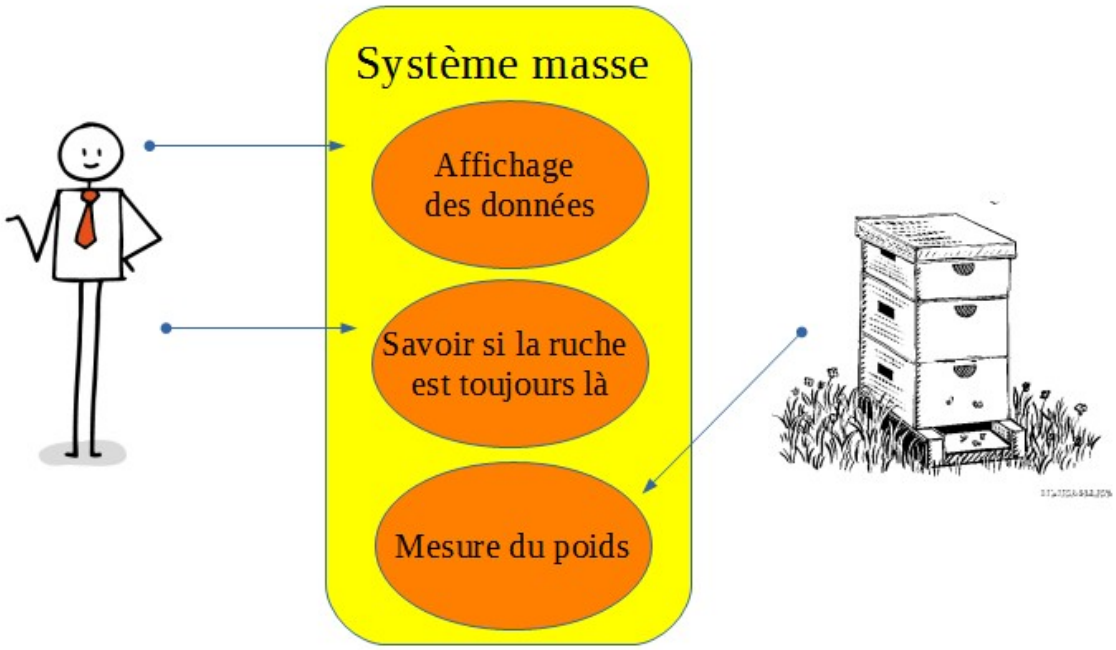


Diagramme SYSML de contexte

Le système récolte poids de la masse sur une balance, et permet à l'utilisateur de connaître celle-ci et par la même occasion de savoir si la ruche est toujours en place (si elle n'a pas été renversé ou volé).

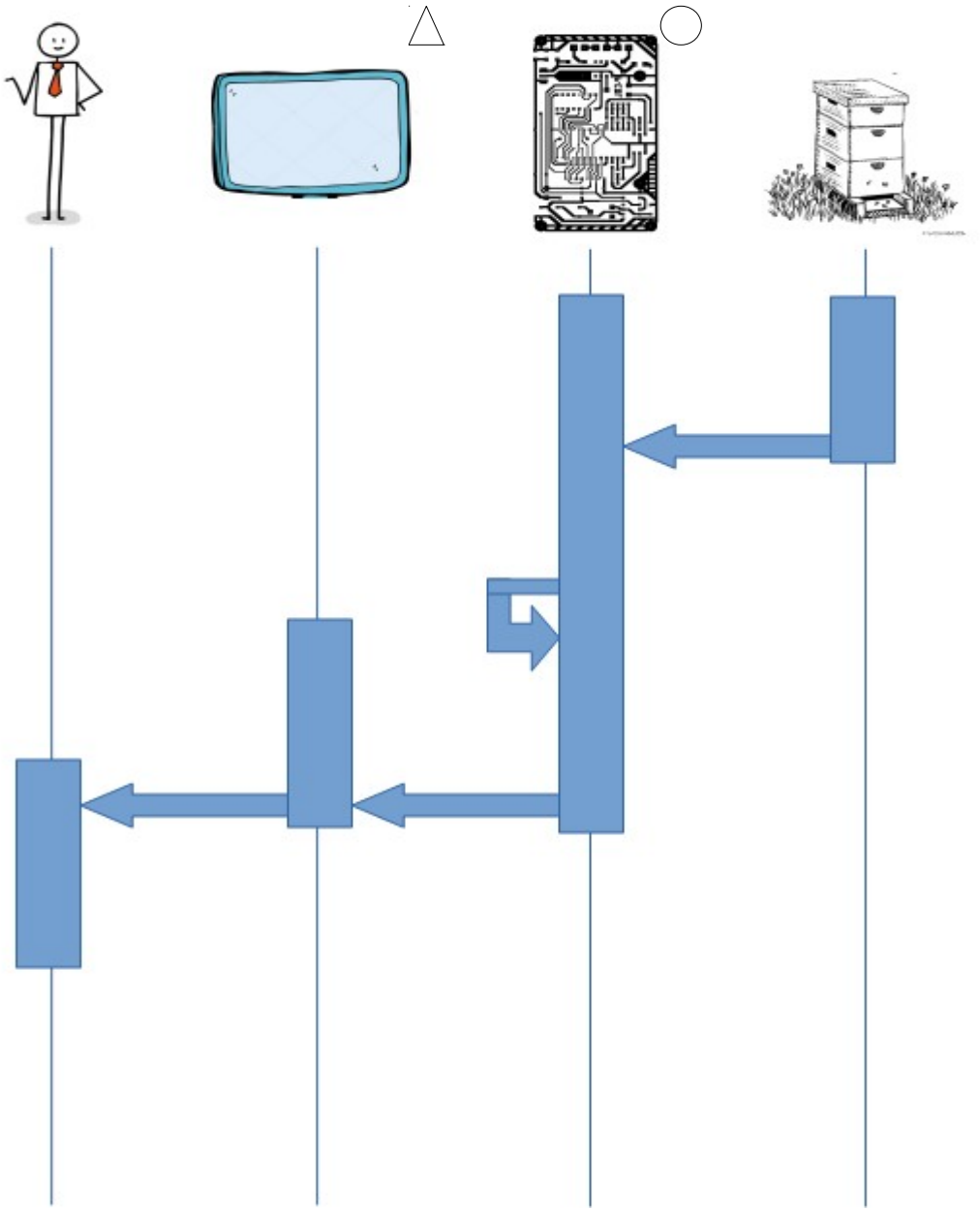


Diagramme SYSML de séquence

△ Écran

◯ Système

La ruche est posée sur une balance, le système traite les informations du capteur optique pour déterminer le poids et transmet les valeurs à un écran qui lui permettra à l'utilisateur de consulter le poids de la ruche.

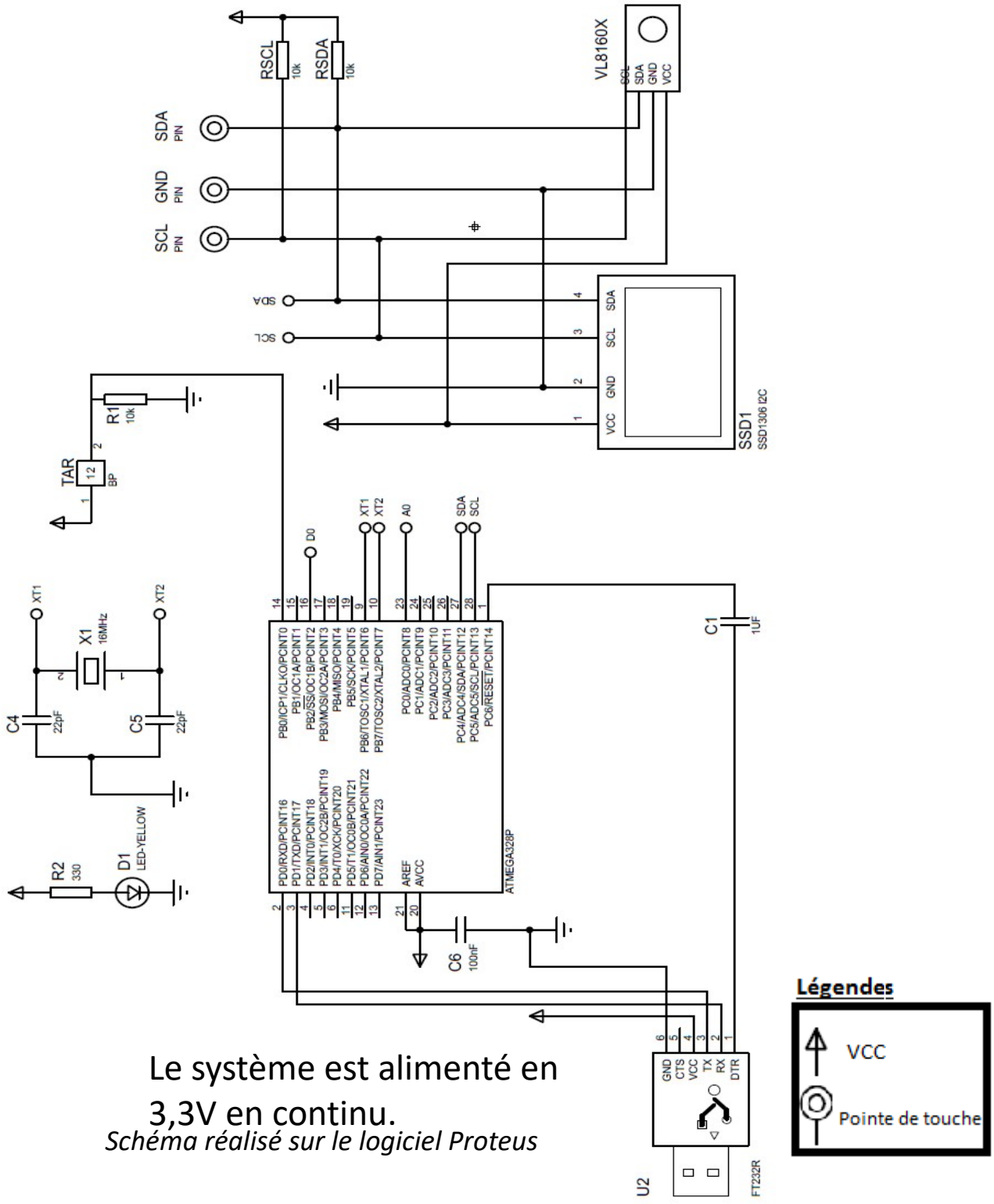
Étude du système masse

- **Éléments du système**

Voici la liste des composants utiles au système de la masse

Nom	Référence sur le schéma	Valeur
Condensateur x1	C1	1 UF
Condensateur x2	C4/C5	22 pF
Condensateur x1	C6	100 nF
Résistance x1	R2	330 Ω
Résistance x3	R1/RSCL/RSDA	10 kΩ
ATMEGA328P x1	U1	
FT232R x1	U2	
LED x1	D1	
Pointe de touche x3	SDA/SCL/GND	
Capteur de distance (VLX8160X) x1		
Quartz x1	X1	16 MHz
Afficheur x1	SSD1306	
Bouton Poussoir x1	TAR	

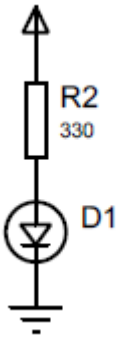
- Schéma structurel simplifié du système de la masse



Le système est alimenté en 3,3V en continu.

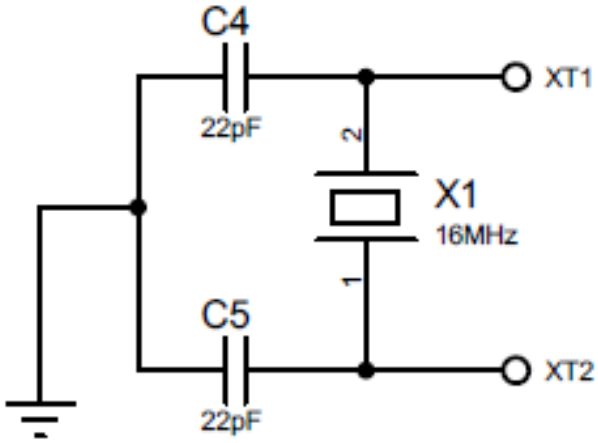
Schéma réalisé sur le logiciel Proteus

- **Schéma structurel (Part I)**



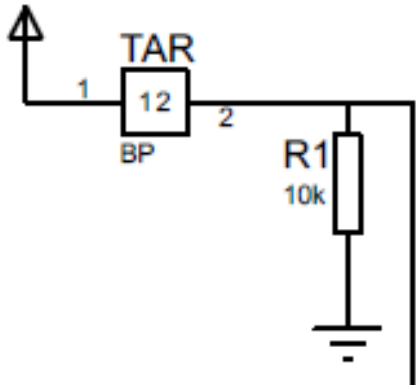
Cette partie du schéma ne sert qu'à vérifier le bon déroulement de l'alimentation.
Si la carte est mise sous tension et que la LED ne s'allume pas cela signifie que la carte présente un défaut et il y a de forte chance que le système ne soit pas alimenté.

- **Schéma structurel (Part II)**



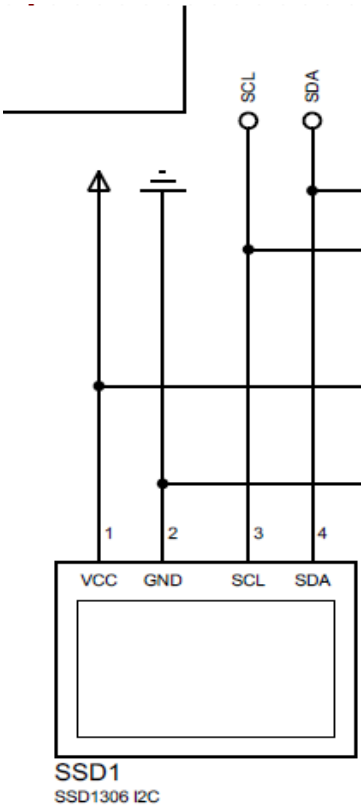
Le quartz de fréquence 16 Mhz, est en parallèle avec 2 condensateurs de même capacités (22pF) pour stabilisé la fréquence.

- **Schéma structurel (Part III)**



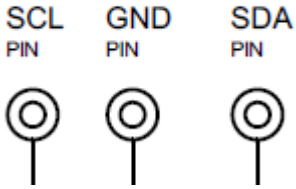
Le bouton TAR permet d'effectuer un tarage, pour connaître le poids net de la ruche sans son récipient. Au repos il y a un niveau logique 0, et quand on appuie sur le bouton on obtient un niveau logique 1.

- **Schéma structurel (Part IV)**



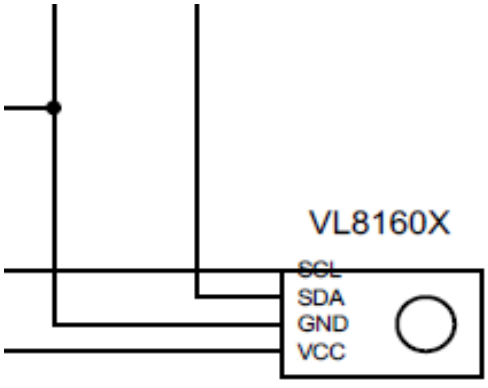
Il s'agit de l'afficheur qui communique sur le bus I2C, son adresse correspond à 0x3C (voir les analyses de trames).

- **Schéma structurel (Part IV)**



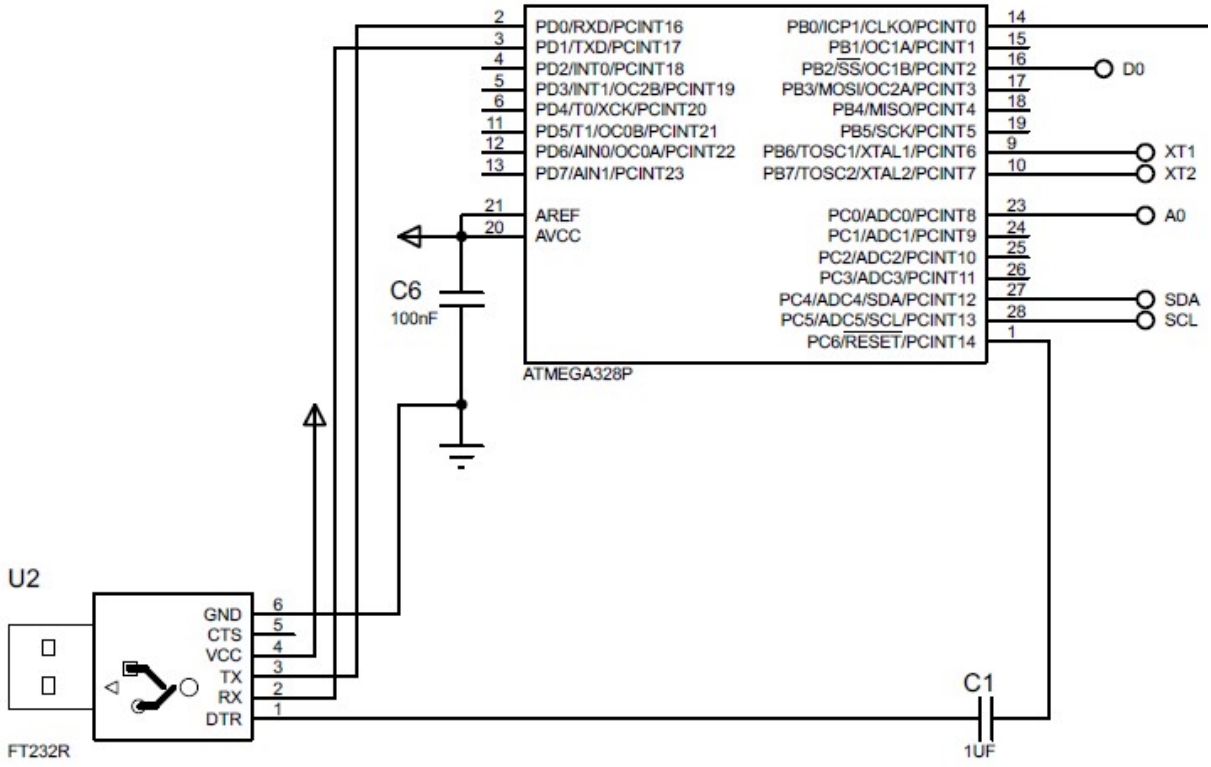
Sur le circuit sur trouve 3 pointes de touches, la masse, et les lignes SCL et SDA.

- **Schéma structurel (Part V)**



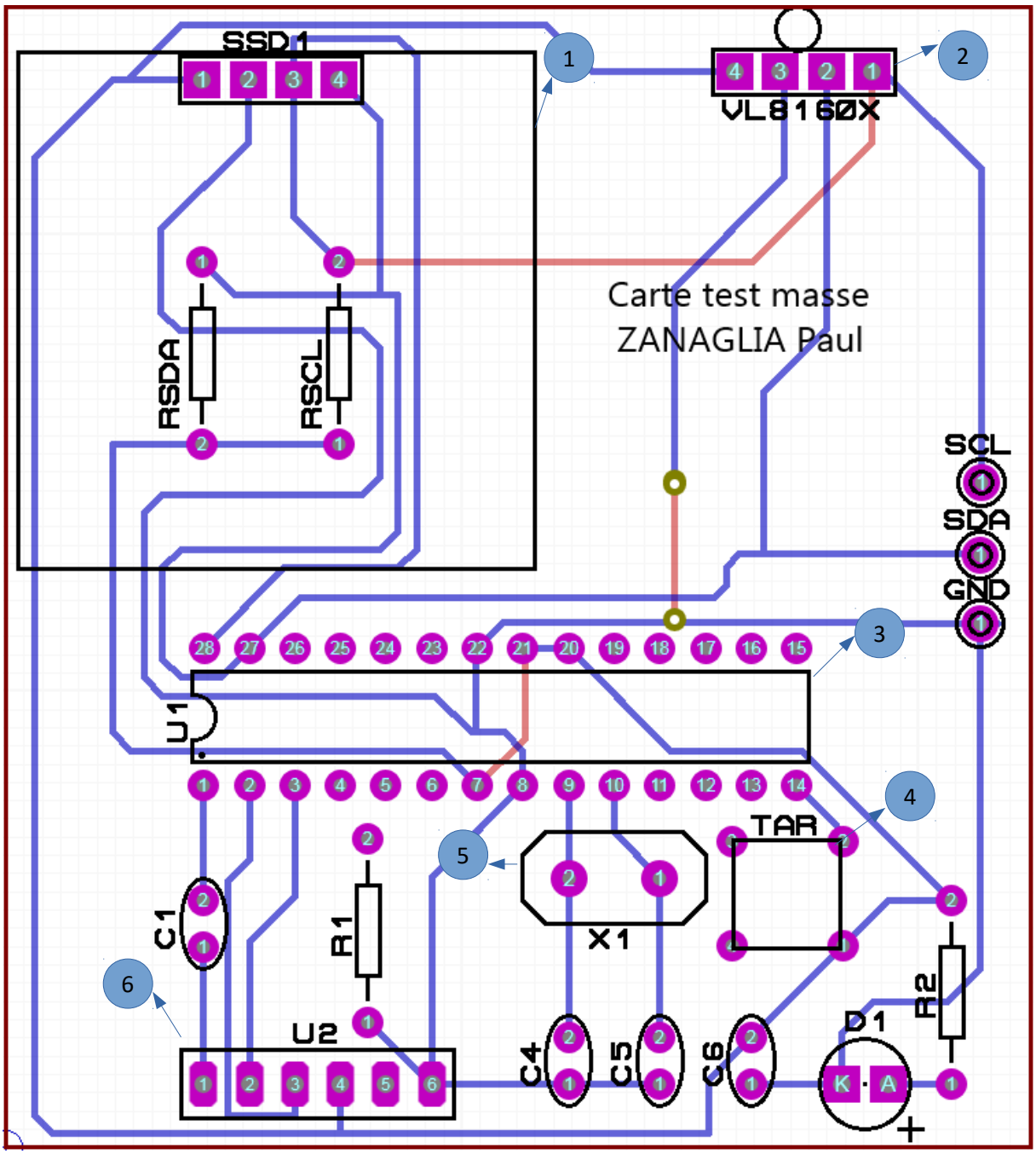
Il s'agit du capteur de distance (VL8160X). Il communique en I2C avec l'adresse 0x29 (pour plus d'information voir l'étude du capteur).

- Schéma structurel (Part VI)



U2 est une puce FTDI qui permet le débogage de la carte. Elle permet d'envoyer un programme au système et d'alimenter le circuit.
L'ATMEGA est quant à lui le microcontrôleur.

- Typon simplifié du système de la masse



Typon réalisé sur le logiciel Proteus

1. Afficheur
2. Capteur de distance
3. Microcontrôleur
4. FTDI
5. Quartz
6. Bouton Poussoir « TAR »

- Programmation

Le programme a été réaliser grâce au logiciel Arduino et est codé en C. (Voir annexe)

Le programme fait appelle à divers librairies comme celle de l'afficheur (Adafruit_SSD1306.h) et du capteur(VL6180X.h) pour simplifier la programmation et le paramétrage du système.
Les fonctions avec les commentaires du types « Fonction « X » » avec X étant le registre appelé (ex : //Fonction "SYSRANGE__MAX_CONVERGENCE_TIME") sont des registres dans le capteur de distance.

Voici un exemple de transmission :

```
//Fonction "RESULT__RANGE_VAL"
Wire.beginTransmission(adrVL6180X);
Wire.write(0x00);  // reg octet de poids fort
Wire.write(0x62);  // reg octet de poids faible
Wire.endTransmission();
```

Dans ce cas-ci on appelle le registre «RESULT__RANGE_VAL».
On fait appelle à Wire.beginTransmission pour débuter la transmission avec l'adresse du capteur (0x29).
Wire.write permet de sélectionner la fonction que l'on souhaite utiliser dans un registre.
Wire.endTransmission met fin à la demande

```
//Affecte la valeur dans le tableau
Wire.requestFrom(adrVL6180X, 1 );
tabMes[indiceTab] = Wire.read();
Wire.endTransmission();
```

Maintenant qu'on est dans le registre, Wire.requestFrom indique qu'on veut faire une lecture sur l'adresse du VL6180X d'1 octet.
Puis Wire.read donne la valeur (qu'on affecte a une variable)
Et on met fin à la demande.

Nous avons utilisé dans cette exemple l'acquisition de la valeur de la distance. Cette donnée étant trop fluctuante, nous avons intégré un moyenneur pour rendre la donnée plus stable.

```
for(indiceTab = 0; indiceTab <=9; indiceTab++){
//Fonction "RESULT__RANGE_VAL"
Wire.beginTransmission(adrVL6180X);
Wire.write(0x00);  // reg octet de poids fort
Wire.write(0x62);  // reg octet de poids faible
Wire.endTransmission();

//Affecte la valeur dans le tableau
Wire.requestFrom(adrVL6180X, 1 );
tabMes[indiceTab] = Wire.read();
Wire.endTransmission();

//Fonction "SYSTEM__INTERRUPT_CLEAR"
Wire.beginTransmission(adrVL6180X);
Wire.write(0x00);
Wire.write(0x15);
Wire.write(0x00);
Wire.endTransmission();

somme=somme+tabMes[indiceTab];
delay(50);


}
```

Pour réaliser ce moyenneur, j'ai créé une sorte de tableau qui va déposer dans 10 cellules (de 0 à 9) une nouvelle valeur d'acquisition toute les 50 ms.
Puis il nous suffit de faire la moyenne de ces valeurs.

```
somme=somme/10;
```

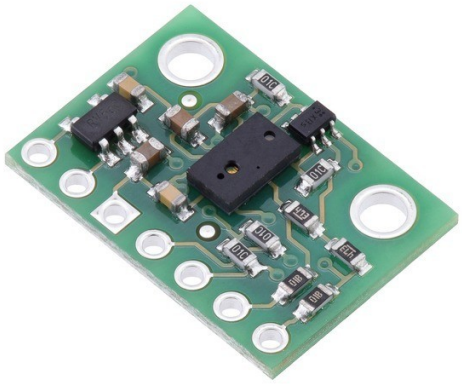
```
float masse= -0.4762*somme; //constante equation changable
masse=masse+46.6667;        //constante equation changable
```

Voici la manière dont on converti la distance en masse (voir courbe d'équation à l'étude du capteur).
Les lignes ayant « constante équation changeable » Possède des valeurs qui peuvent être calibrées.

```
if (somme>102 & masse<0){ //constante equation changable
    masse=0;

      alert++;
      }

  else { alert=0; }

 if (alert>=5) {
     display.clearDisplay();
     display.setCursor(0,32);
     display.setTextSize(2);
     display.print("Absence de la ruche");
     }
```

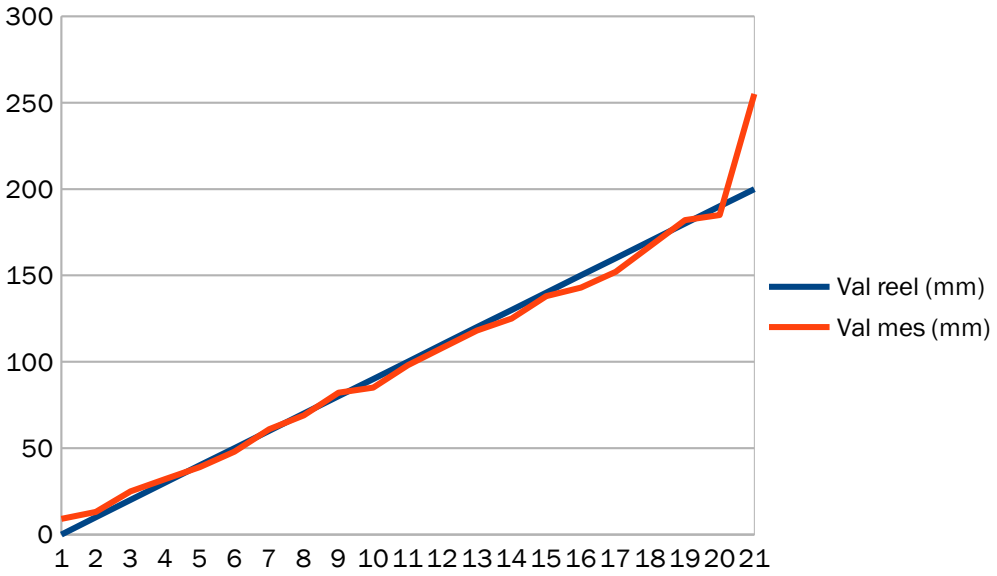
Voici ce qui permet d'alerter l'utilisateur qu'il y a un soucis avec la ruche.
Pour être sûr qu'il y a un problème, on vérifie si la somme des distances est supérieur à 102 et si la masse inférieur à 0 alors on incrémente la variable alerte.
Si celle ci vaut 5 ou plus alors on avertit l'utilisateur. Sinon la variable retourne à 0.

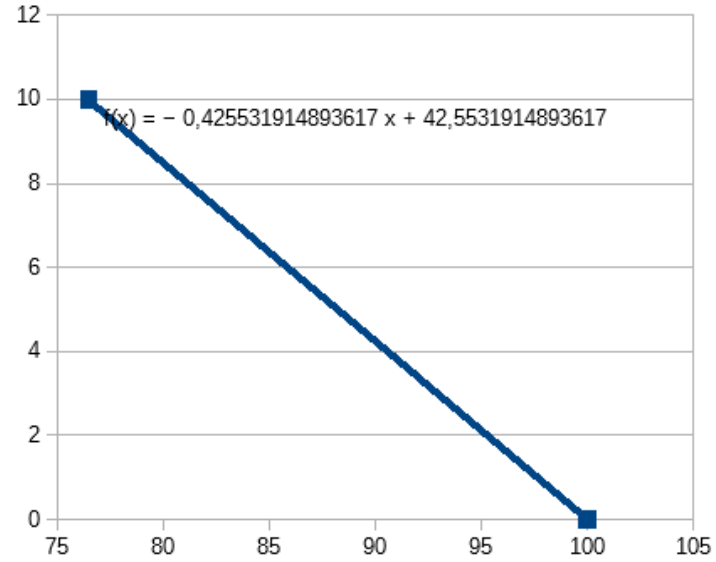
Étude du capteur optique de distance



Le VL6180X est un capteur optique de distance qui réalise une mesure de celle-ci en envoyant un rayon infrarouge. La distance est déterminée en fonction du temps de retour du rayon. Le constructeur assure que jusqu'à 100 mm le capteur devrait faire une mesure précise et au dessus ne promet rien.

Voici un graphique des mesures réalisé avec le capteur.
L'abscisse représente le numéro du test réalisé (soit 21 test) on était fait.
Et l'ordonné la distance en mm.
La courbe bleu correspond aux valeurs réelles contre l'orange aux valeurs mesurées par le capteur.





Voici la courbe et l'équation qui permet de déterminer la masse à partir de la distance.
L'abscisse représente la distance en mm et l'ordonné le poids en kg.
Cette mesure peut être modifié et adapté, ici un poids de 10kg a servi de référence pour un point en plus du point de référence pour 0kg.

Pour plus d'information sur le capteur (comme les registres), se référer à la documentation technique du composant.

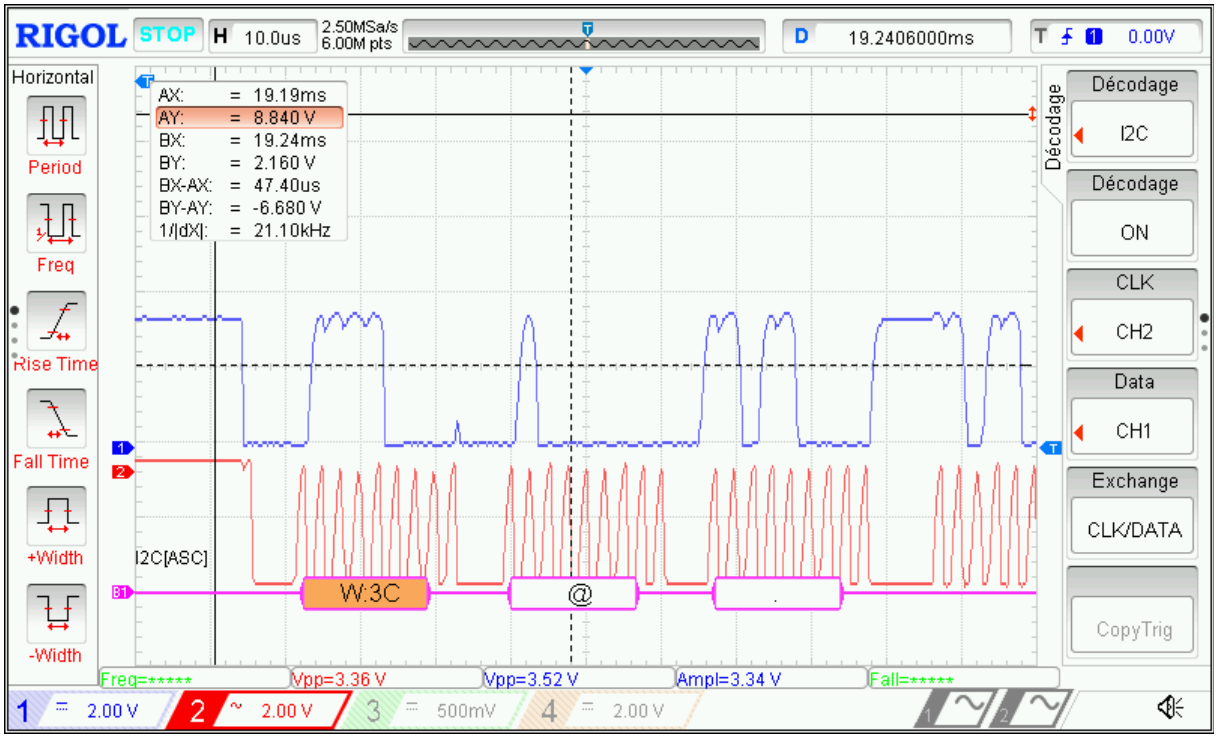
Étude de la balance

Cette balance est une maquette pour concevoir le système de la masse.

Elle est équipé de 4 ressorts, chacun ayant pour valeur 1,304 N/mm et donc pour un total de 4*1,304 = 5,216 N/mm.

Un changement de 1 mm corresponds à une variation de la masse d'environ 0,48 Kg soit 480g.
Il s'agit de la valeur minimal pour varier la valeur de la distance du capteur pour une balance équipée de 4 ressorts ayant un coefficient d'amortissement de 5,216 N/mm.

Mesures



Il s'agit d'une trame à l'oscilloscope montrant une trame typique de l'afficheur sur le bus i2c qu'on peu reconnaître à l'adresse.

Cette trame représente une trame de donnée typique du capteur de distance. On le reconnaît à son adresse 0x27.
0x062 (donc en décimal 098) est un appel du registre d'acquisition de la mesure de la distance.
Et la valeur 255 qui représente la valeur de la distance
(Attention 255 est une valeur donnée lorsque il y a un soucis avec le capteur).

Conclusion

Le projet aura débuté le 25/01/18 et s'achevera le 08/06/18.
Pour répondre à la demande du client (réaliser une ruche connecté) nous avons dû réaliser les différentes fonctionnalités de ce système individuellement pour mieux l'integrer par la suite et faciliter la conception de ce projet.
Pour ma part je devais principalement réaliser le système d'acquisition de la masse d'une ruche, avec un capteur de distance optique. La carte de test fût réalisée completement, et sans emcombre. Néanmoins le plus grand challenge était de s'approprier la manière de programmer le capteur. En effet, le VL6180X est un composant fonctionnant en I2C et possédant énormement de registre, il aura fallu s'interroger sur la façon dont le capteur est programmer et quel registre nous étais le plus utile pour simplifier le programme.
Au final, après tests, reflexions et analyse avec les professeurs nous avons décider de ne pas toucher les registres d'initialisations et de paramétrages du capteur pour éviter d'avoir des soucis avec celui-ci. De ce fait je n'ai programmé principalement que par moi même les demandes des valeurs à la lecture du capteur.
Par la suite la mise en commun a était très enrichissant car il a fallu regrouper l'ensemble des travaux de l'ensemble du groupe et s'enrichir des connaissances que les autres avait développer.
Cependant par manque de temps, nous ne pourrons peut-être pas finir ce projet, car nous devons finir et affiner la programation qui est encore très instable et rempli de bugs (en plus des problèmes materiels.
Ce projet fût une aubeine pour entrevoir ne serait-ce qu'un peu le monde de l'entreprise et de la conception, c'est quelque chose qui m'a beaucoup plu de par le concept, le travail de groupe et la recherche de solutions pour une problématique donné.

ANNEXE - Système masse avec capteur

Programme complet

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <VL6180X.h>
#include <VirtualWire.h>


#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);

#if (SSD1306_LCDHEIGHT != 64)
#error("Height incorrect, please fix Adafruit_SSD1306.h!");
#endif

//Variables Paul
int masse;
const int nbMes = 10;
int tabMes[nbMes];
int indiceTab=0 ;
int somme;
char alert;
byte adrVL6180X=0x29;
int period;
int period_reg;

//Variables Romain
int potPin = A0;
int potValue = 0;  //déclaration de variables
float masse2;

//Variables Axel
int nbrabeilles;
int capt;
byte adrPCF8574=0x20;
byte adrHIH8120=0x27;
int msbhi;
int lsbhi;
int msblo;
int lsblo;
int humidity;
int temp;

bool  bpreset = 14;
float vide;

VL6180X sensor;

void setup() {
  Serial.begin(9600);
  Wire.begin();
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);

  pinMode (bpreset , INPUT);

  //rafraichissement écran avec le logo du constructeur Adafruit
  display.display();

  delay(2000);

  sensor.init();
  sensor.configureDefault();

  //Fonction "SYSRANGE__MAX_CONVERGENCE_TIME"
  Wire.beginTransmission(adrVL6180X);
  Wire.write(0x00);  // reg octet de poids fort
  Wire.write(0x1C); // reg octet de poids faible
  Wire.write(30);  // temps maximum pour lancer la mesure en ms
  Wire.endTransmission();

  //Fonction "SYSALS__INTEGRATION_PERIOD"
  Wire.beginTransmission(adrVL6180X);
  Wire.write(0x00);  // reg octet de poids fort
  Wire.write(0x40); // reg octet de poids faible
  Wire.write(0x00);
  Wire.write(50);  //"Integration period" pour ALS mode 50 ms
  Wire.endTransmission();

  sensor.setTimeout(500);

  // Arrete le mode continu s'il est déjà activé
  sensor.stopContinuous();
  // Dans le cas où stopContinuous() a déclanché un single-shot

  delay(300);

  // Commence le mode à intervalle continu avec une période de 100 ms
  // Comment les mesures de la distance en continu avec la periode donnée en ms
```

```
float vide= -0.4762*vide; //constante equation changable
vide=vide+46.6667;
 }

display.clearDisplay(); // Efface des éléments de l'affichage
display.setTextColor(WHITE); //Choix de la couleur, dans notre cas WHITE c

if (sensor.timeoutOccurred()) {
    display.setCursor(0,0);
    display.print("Erreur    systeme");
     }
else {
    for(indiceTab = 0; indiceTab <=9; indiceTab++){
    //Fonction "RESULT__RANGE_VAL"
    Wire.beginTransmission(adrVL6180X);
    Wire.write(0x00);  // reg octet de poids fort
    Wire.write(0x62);  // reg octet de poids faible
    Wire.endTransmission();

    //Affecte la valeur dans le tableau
    Wire.requestFrom(adrVL6180X, 1 );
    tabMes[indiceTab] = Wire.read();
    Wire.endTransmission();

    //Fonction "SYSTEM__INTERRUPT_CLEAR"
    Wire.beginTransmission(adrVL6180X);
    Wire.write(0x00);
    Wire.write(0x15);
    Wire.write(0x00);
    Wire.endTransmission();

    somme=somme+tabMes[indiceTab];
    delay(50);


    }

   somme=somme/10;
   float masse= -0.4762*somme; //constante equation changable
   masse=masse+46.6667;        //constante equation changable
   masse=masse-vide;

  if (somme>102 & masse<0){ //constante equation changable
      masse=0;
                                        alert++;
                                        }

                                  else { alert=0; }

                                  display.setTextSize(2);

                                  //masse mesurée par le capteur
                                  display.setCursor(0,0);
                                  display.print(masse);
                                  display.print("Kg");


                                  display.setCursor(0,15);
                                  display.print(somme);
                                  display.print("mm");

                                 if (alert>=5) {
                                      display.clearDisplay();
                                      display.setCursor(0,32);
                                      display.setTextSize(2);
                                      display.print("Absence de la ruche");
                                      }

                                 }

                              }
```