18 SEPTEMBRE 2023

Compte Rendu : Réalisation d'un système d'alarme domestique

BENSEBAH Kheireddine

BEHILLIL Sofiane

COMPTE RENDU CONCEPTION SYSTEME D'ALARME

SAE

Table des matières

Introduction :

Lors de ce projet en séance d'apprentissage et d'évaluation (SAE), nous avions pour objectif la conception et la mise en œuvre d'un système d'alarme domestique à l'aide de capteurs, d'un système de communication et d'une carte Arduino. Ce projet nous laisse donc une certaine liberté vis-à-vis de la conception de notre système de sécurité alliant créativité, la technologie et la sécurité, offrant ainsi une opportunité d'appliquer les connaissances acquises lors de nos années passées en génie électrique et informatique industrielle (GEII).

L'objectif de ce projet était de développer un système d'alarme efficace et adaptable pour une utilisation domestique. Pour atteindre cet objectif, plusieurs étapes cruciales ont été définit, ce compte rendu témoigne et retrace notre parcours de création de ce système d'alarme domestique.

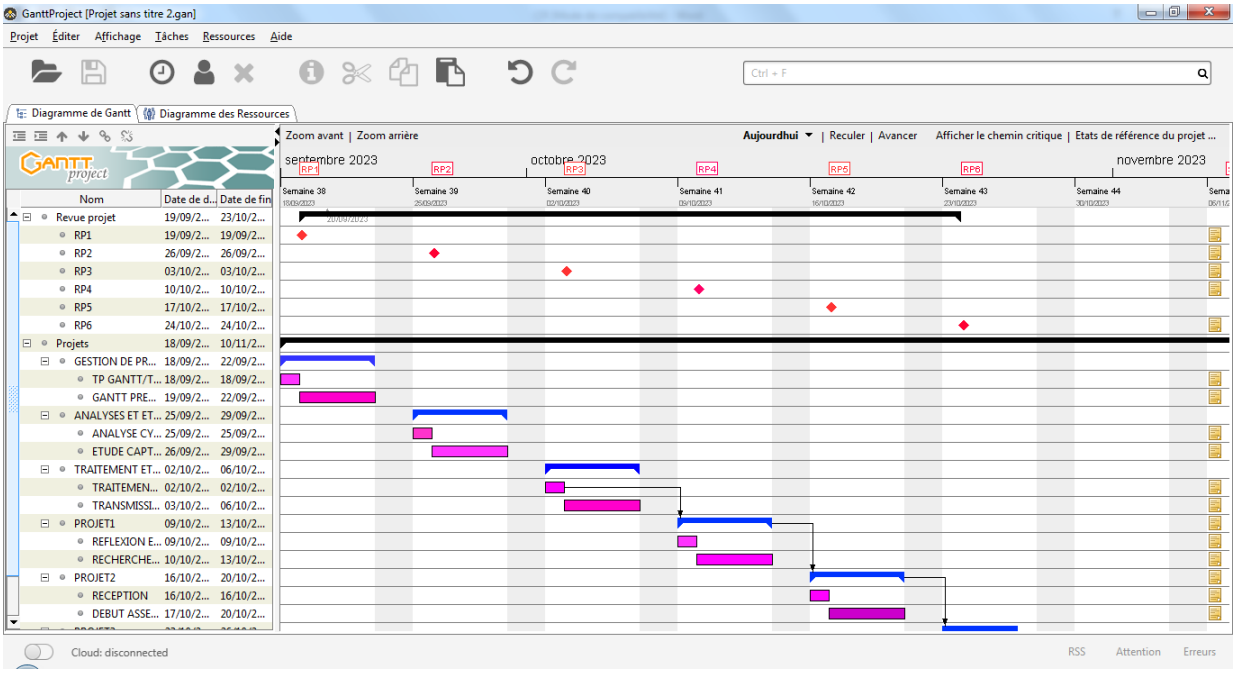
I) Planning et Outil Gantt:

Afin d'assurer une progression structurée tout au long de la SAE, nous avons élaboré un planning détaillé. Pour cela, nous avons utilisé l'outil de planification Gantt qui est couramment utilisé en gestion de projet, il permet de représenter visuellement les différentes activités (tâches) et durées qui constituent un projet. Cet outil a été essentiel pour la répartition efficace du travail au sein de l'équipe, permettant ainsi de respecter les délais requis et le bon déroulement de ce projet.

A) Gantt prévisionnel

Pour commencer, nous avons donc mis en place un planning prévisionnel, prenant en considération notre emploi du temps respectif pour ce projet. Ce planning a été structuré de manière à intégrer les dates cruciales associées à nos petites présentations orales périodiques, permettant ainsi un suivi régulier de chaque amélioration et avancement de nos tâches.

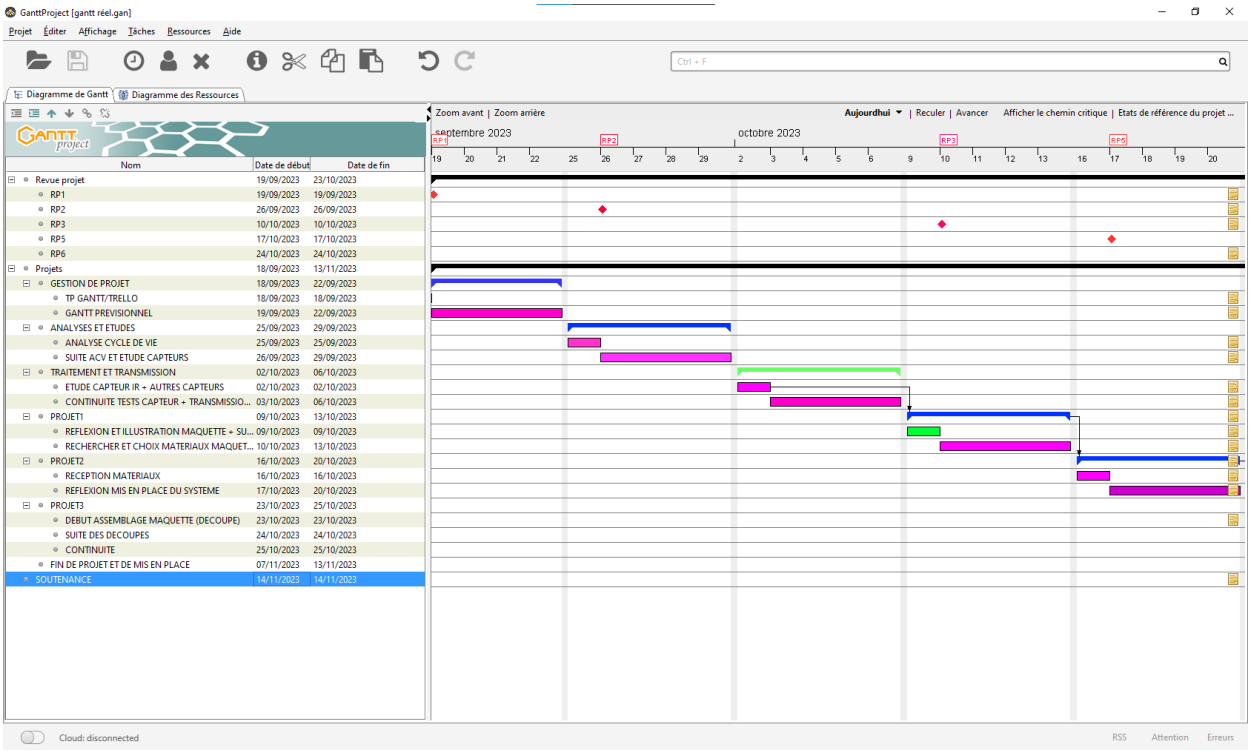
Ensuite, nous avons établi ce planning prévisionnel en soulignant les grandes étapes de ce projet en débutant par la création de ce planning préalablement élaboré. En enchaînant les différentes étapes, notamment par exemple avec une analyse approfondie des différents capteurs à utiliser, ainsi qu'une évaluation du cycle de vie des matériaux envisagés.



Gantt prévisionnel

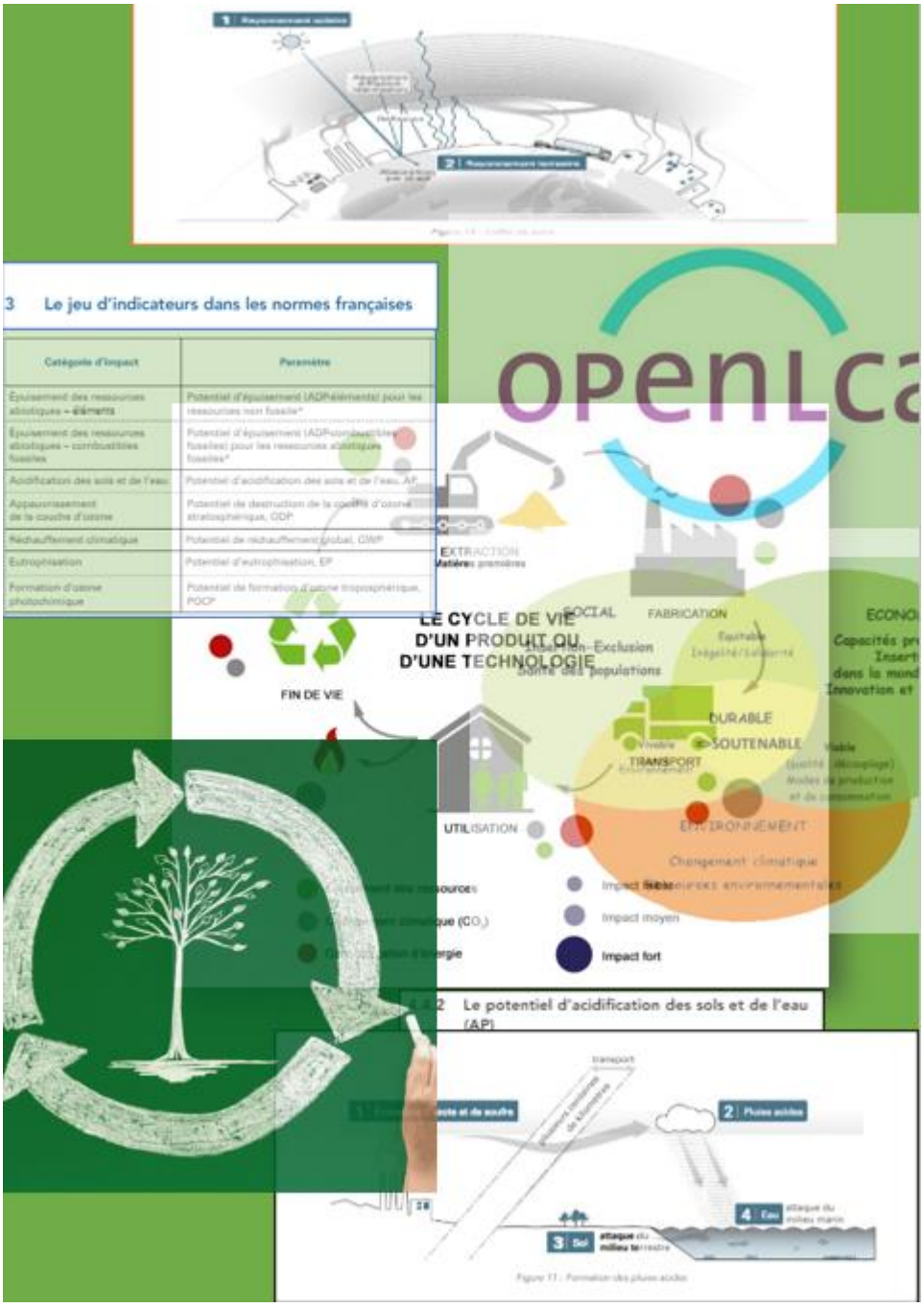
B) Gantt réel

Suite à la mise en œuvre du projet, voici désormais le planning réel, mettant en lumière les différents écarts entre notre planning prévisionnel et ce qui a été réellement réalisé. Ces écarts sont principalement attribuables à une mauvaise gestion du temps lors de l'accomplissement des premiers objectifs, qui se sont avérés peut-être plus long que prévu. Notamment lors de l'analyse de cycle de vie, l'étude du premier capteur de température. Cependant, cet écart a très vite été rattrapé lors du début de fabrication de la maquette. Cette expérience nous enseigne l'importance de mieux évalué la durée nécessaire aux différentes tâches.



Gantt réel après SAE

II) Analyse cycle de vie :



Affiche d'introduction ACV

Dans cette seconde partie de notre compte rendu, nous avons étudié l'impact environnemental que pourrait avoir notre projet vis-à-vis de la planète. En effet, en tant que concepteurs, il est impératif d'évaluer les implications environnementales de notre projet de système d'alarme, en fonction de nos choix de matériaux et de technologie, car nous avons un impact sur l'environnement et nous devons intégrer les principes de l'Analyse de Cycle de Vie (ACV) dans notre projet.

A) OpenLCA

Pour mieux comprendre et appréhender l'ACV au sein de notre projet de système d'alarme, nous avons utilisé l'outil OpenLCA. Un logiciel qui permet de modéliser et de comprendre les impacts environnementaux de manière pratique. Nous avons donc utilisé un guide qui propose une création de modèles utilisant l'exemple de bouteilles comme cas pratique.

Grâce à ce logiciel, nous avons pu comprendre grâce aux étapes détaillées de l'exemple fournit les impacts que peuvent avoir ces différents objets du quotidien, dans l'exemple utilisé la bouteille que nous avions conçue était représentée par plusieurs processus distincts, c'est-à-dire chaque étape de production de notre bouteille.

B) Maquette et ACV

Dès la conception de notre maquette pour concevoir notre maison et y intégrer notre système d'alarme, nous avons pris la décision de choisir du bois. Tout comme avec l'exemple de la bouteille, ce bois a dû être extrait, transformé, transporté, etc. Son utilisation était donc primordiale, car parmi les différentes étapes du cycle de vie d'un objet, sa fin de vie est très importante, parce qu'ici, on récupèrera ainsi du bois qui a déjà eu une vie passée. Cette approche au niveau de l'ACV au sein de notre projet permet de quantifier grâce à l'outil OpenLCA les avantages environnementaux du choix du bois dans notre projet.

C) Capteurs et ACV

En parallèle, nous avons intégré différents capteurs dans notre maquette dont deux petits capteurs de distance dont nous détaillerons l'utilisation un peu plus loin dans ce compte rendu. L'importance de comprendre donc ici l'empreinte écologique de notre système de sécurité influx sur la décision des différents capteurs à choisir. Par exemple pour notre projet, nous avons pris la décision de ne prendre que deux capteurs de distance, ce choix peut être justifié par rapport aux coûts des différents matériaux, notamment en ces temps où la technologie devient de plus en plus chère, en analysant le cycle de vie de nos capteurs, on peut savoir très facilement qu'en diminuant le nombre de capteurs utilisé entraîne une diminution du nombre de processus de fabrication requis. De plus, moins de capteurs signifie potentiellement moins de consommation d'énergie pendant la phase d'utilisation du système.

D) Conclusion

L'analyse de cycle de vie, se présente comme une méthode évaluant les impacts environnementaux d'un produit tout au long de son cycle de vie, depuis l'extraction des matières premières jusqu'à la gestion des déchets. L'ACV permet une prise de décision éclairée dans la conception, la production et la gestion d'un produit en visant à réduire son impact environnemental.

III) Choix et déploiement des capteurs

Pour cette troisième partie, cette étape cruciale de notre projet explore le processus méticuleux qui a guidé la sélection et le déploiement de nos capteurs. Comme dit précédemment, une certaine liberté nous est attribuée vis-à-vis de nos décisions sur le type de sécurité voulu pour notre maison. En effet, en tant que concepteurs d'un système de sécurité pour notre maison, il est impératif de choisir de quels composants sera équipé notre maison, nos choix devront donc être justifiés, car il faut veiller à choisir des capteurs ayant un rôle stratégique dans la sécurité globale du système.

Capteur de température LM335

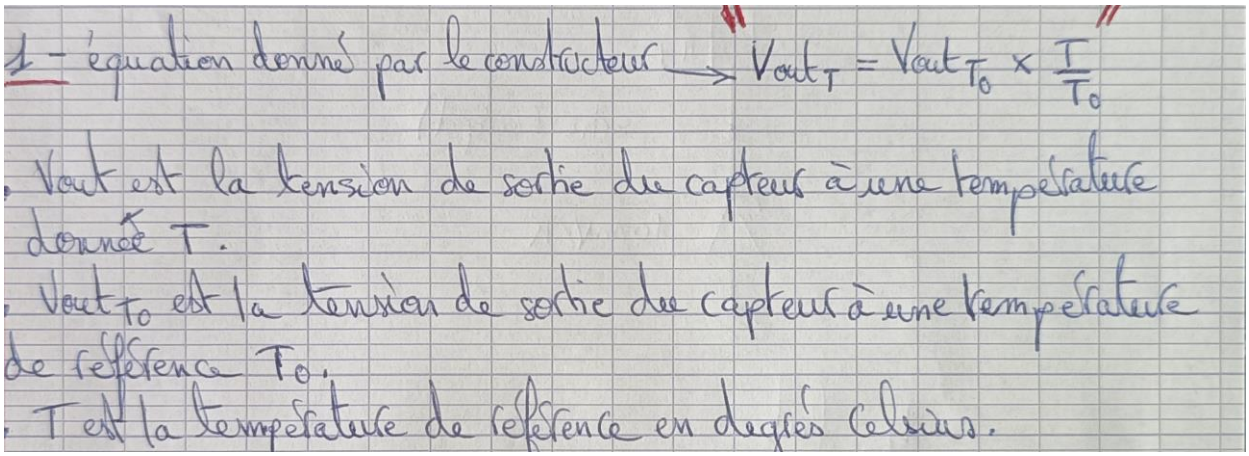
A) Capteur de température

Dans cette démarche méthodique, nous avons entamé notre exploration par le test du capteur de température, une des nombreuses composantes proposées pour sécuriser notre environnement. Nous avons donc suivi un processus étape par étape afin de s'assurer du bon fonctionnement de ce capteur.

1) Théorie

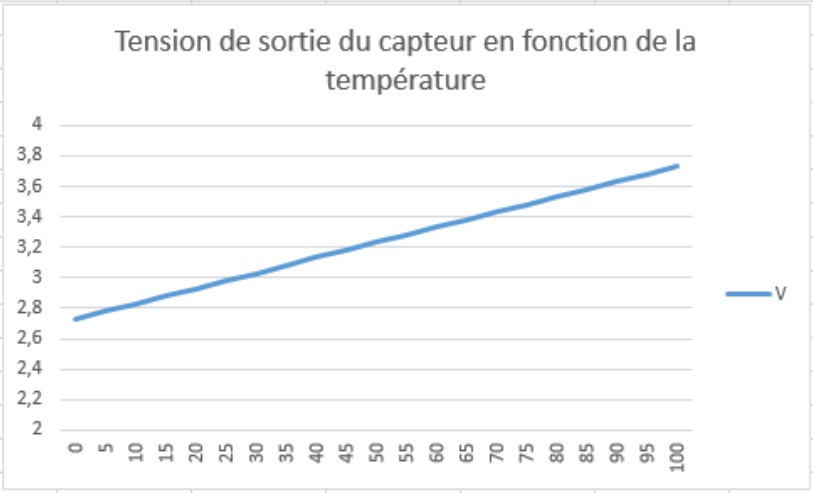
Notre première étape consistait à étudier le fonctionnement théorique du capteur de température LM335 à l'aide de la datasheet fournit par le constructeur. En obtenant l'équation donnée par celui-ci,

nous pouvons fournir précisément l'évolution la courbe théorique de la température en fonction de la tension de sortie :



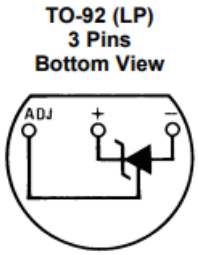
Formule théorique de la courbe de température

	A	B
1	°C	V
2	0	2,7315
3	5	2,7815
4	10	2,8315
5	15	2,8815
6	20	2,9315
7	25	2,9815
8	30	3,0315
9	35	3,0815
10	40	3,1315
11	45	3,1815
12	50	3,2315
13	55	3,2815
14	60	3,3315
15	65	3,3815
16	70	3,4315
17	75	3,4815
18	80	3,5315
19	85	3,5815
20	90	3,6315
21	95	3,6815
22	100	3,7315
23		



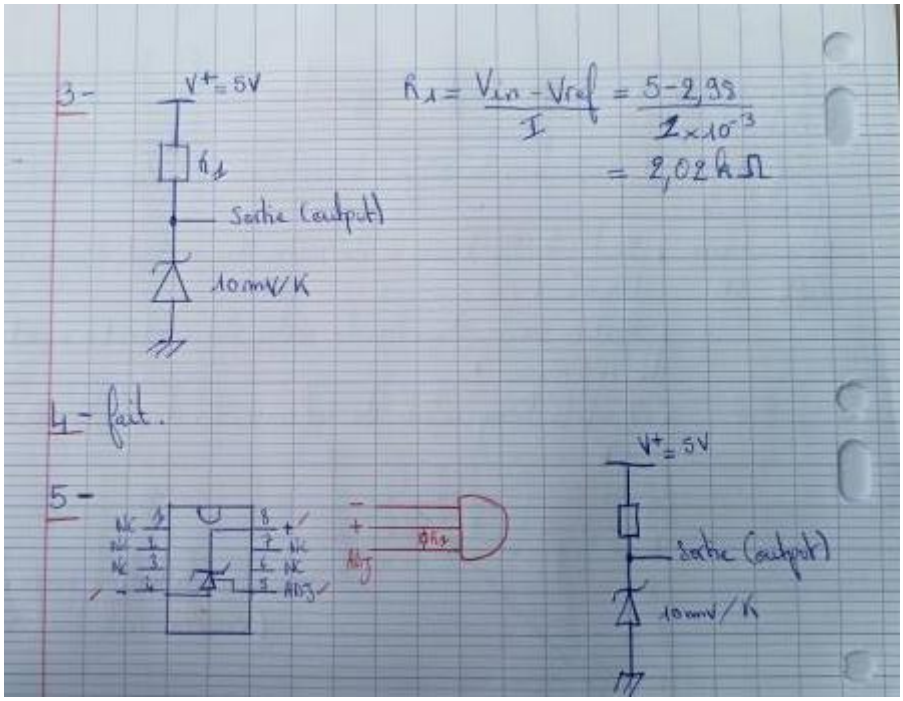
Courbe de la tension de sortie en fonction de la température (obtenue à partir de la formule)

Pour vérifier la précision des mesures données par le constructeur, on s'assure que le LM335 est correctement câblé avant de commencer toute mesure.

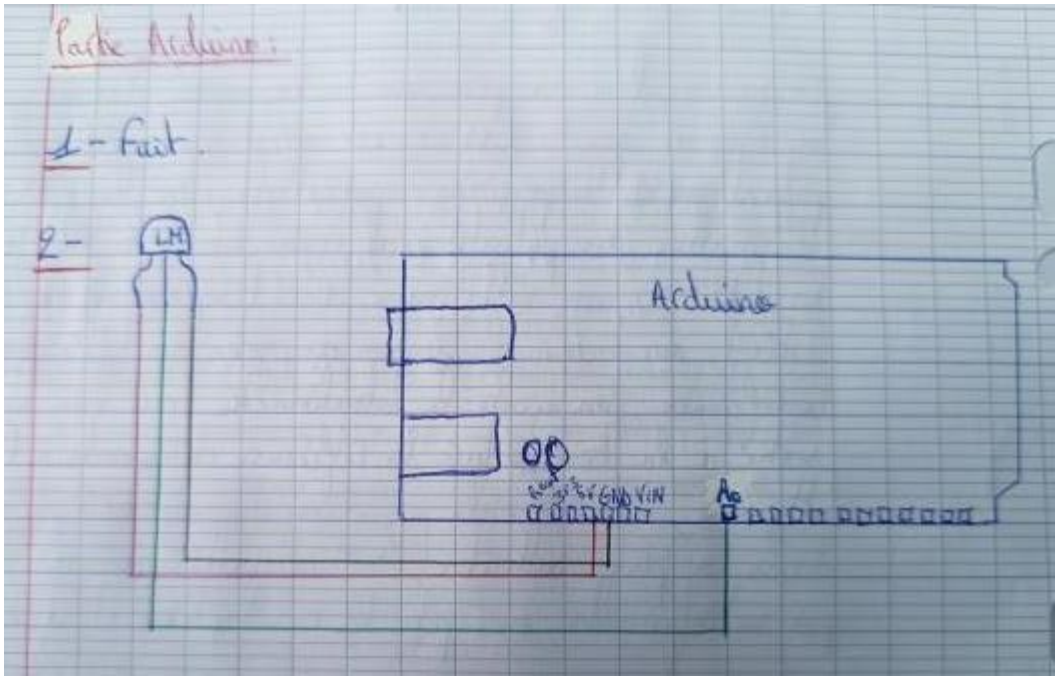


Brochage du LM335

On fera donc attention aux broches de ce composant en le câblant comme indiqué.



Câblage du capteur de température



Câblage du LM335 avec l'Arduino

2) Pratique

Une fois le câblage réalisé, on prévoit un protocole d'utilisation du capteur et un programme en conséquence qui permet de relever la température grâce au capteur brancher à notre Arduino.

```
#include <stdio.h>
#include <stdlib.h>

float val = A0;
float P;
float T,K,C;

void setup ()
{
  Serial.begin(9600);

}

void loop()
{
  P = analogRead(val);
  T = P * 0.00448;
  //Serial.println(T);
  K = T * 100;
  C = K - 273;

  Serial.print(C);
  Serial.println("°C");
}
```

Code de test pour le capteur de température

En ce qui concerne le protocole, on placera notre capteur dans de la glace pilée afin de relever la tension en sortie du capteur (donc pour 0°C) et par la suite, on prendra de l'eau bouillante et on placera le capteur dans cette eau dans le but de mesurer la tension en sortie pour 100°C. Cette méthode permettra de calibrer le capteur de température pour des conditions extrêmes d'utilisation, par ailleurs, on pourra confirmer nos mesures prévues théoriquement.

3) Conclusion

Pour conclure, notre câblage et nos tests pratiques ont confirmé la validité du capteur de température LM335. Les résultats obtenus et comparés correspondent à nos attentes, toute fois, il est important de noter que la résistance du capteur est limitée aux températures extrêmes, car au bout de quelques utilisations celui-ci ne fonctionnait plus, effectivement il n'a pas résisté très longtemps à l'eau, de la glace pilée ou à l'extrême chaleur auquel il était confronté. Ce processus méthodique contribue à garantir la fiabilité des capteurs choisis pour notre système de sécurité.

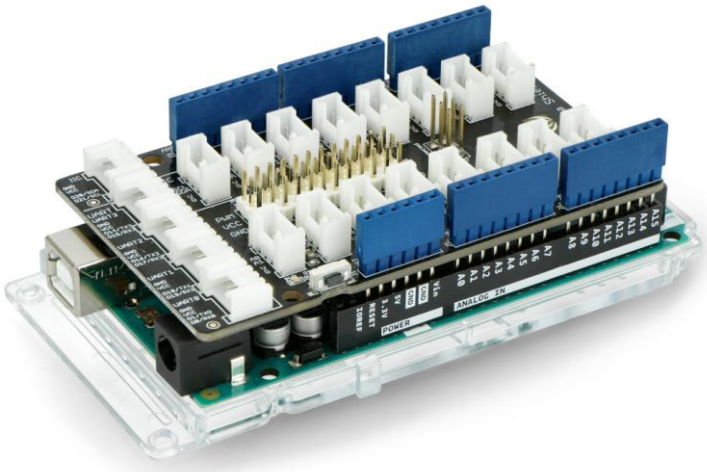
B) Capteur de mouvement

Poursuivant notre exploration méthodique pour trouver le capteur adéquat destiné à renforcer la sécurité de notre environnement domestique, nous focalisons maintenant notre attention sur le capteur de mouvement.

Capteur de mouvement

1) Théorie

Pour commencer, il est important de notifier que nous avons utilisé ce capteur en conjonction avec le module Grove Megashield qui est un module adapté à l'Arduino Mega que nous avons utilisé, celui-ci est simplement composé de plusieurs broches en plus de l'Arduino Uno que nous avions l'habitude d'utiliser.



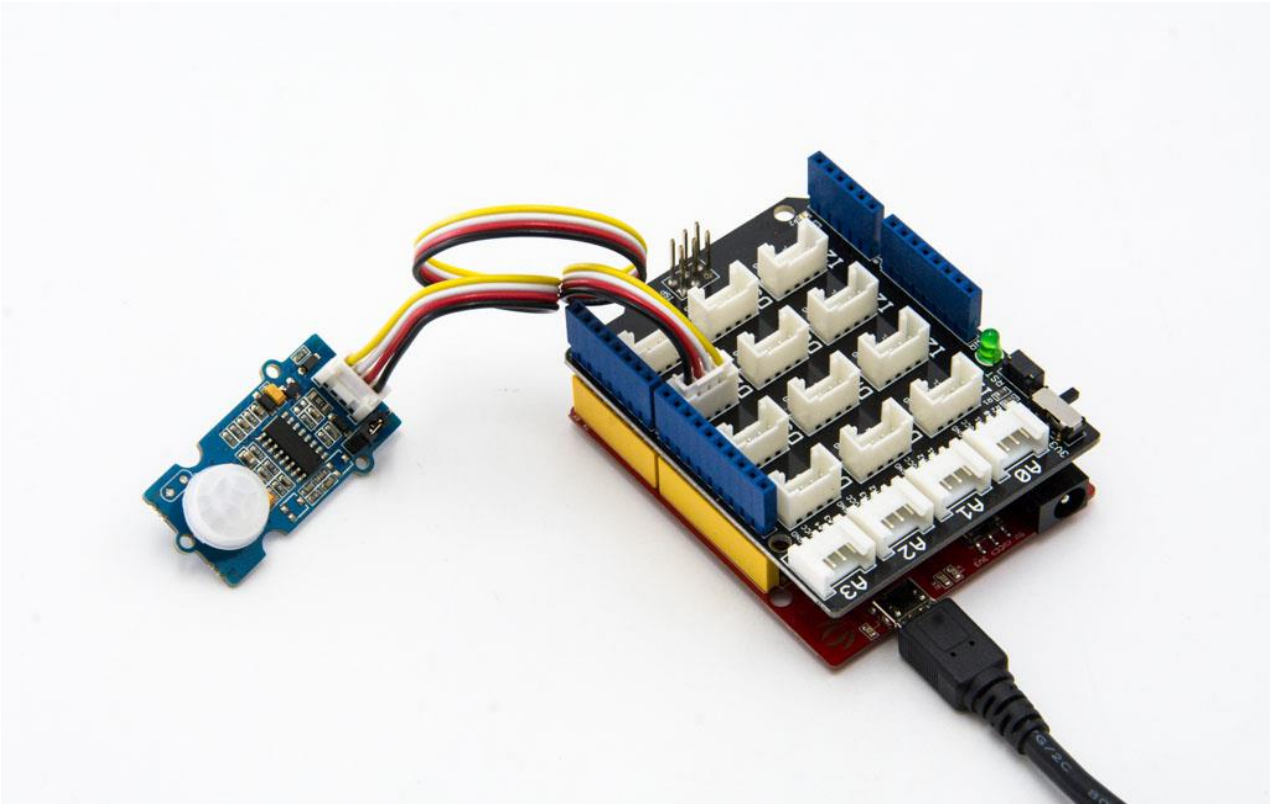
Arduino Mega et son module Grove mega shield

Pour comprendre le fonctionnement théorique de ce capteur, nous nous sommes appuyés sur des ressources en ligne telle que le site « **Seeed Studio ».** Cette étape nous a permis de cerner les fonctionnalités du capteur de mouvement, offrant ainsi une base solide pour nos tests pratiques.

D'après nos recherches, le capteur de mouvement fonctionne donc comme suit : Il repose sur le principe de détection de chaleur, il est équipé d'un matériau qui permet de générer un signal électrique en réponse aux variations de température dans son champ de vision. Le capteur étant sensible aux émissions de chaleur, il détecte grâce à un capteur infrarouge les êtres vivants qui émettent de la chaleur (ou tout simplement les changements de température, donc attention à garder le capteur sur un endroit stable, car tout mouvement du capteur entraîne une détection). Ce changement de chaleur perçu par le capteur entraîne alors une génération d'un signal qui est converti en signal électrique qui annonce donc la présence d'un mouvement.

En résumer, le capteur de mouvement est un capteur sensible aux émissions infrarouges associés aux mouvements dans son champ de vision. Il convertit ces changements de température en signaux électriques, fournissant ainsi une méthode efficace pour la détection de mouvement.

En ce qui concerne le câblage prévu, on utilisera uniquement le câble fournit avec le module Grove PIR Motion Sensor, qui est une nappe qui relie directement les broches d'alimentation et de sortie au module Grove Megashield de l'Arduino que nous utilisons. Le capteur de mouvement sera donc relié via ce câble fournit à la broche digitale de notre choix :



Câblage Arduino Mega avec le capteur de mouvement

2) Pratique

Une fois le câblage réalisé, l'Arduino alimenté via le câble branché à notre ordinateur, on se servira du code de test du capteur de mouvement que l'on a récupéré lors de notre documentation :

```
/*
macro definitions of PIR motion sensor pin
Use pin 2 to receive the signal from the module
*/
#define PIR_MOTION_SENSOR 2


void setup()
{
    pinMode(PIR_MOTION_SENSOR, INPUT);
    Serial.begin(9600);

}

void loop()
{
    // If it detects moving people
    // To know more about why digital numbers are used as boolean, check
https://www.techtarget.com/whatis/definition/Boolean#:~:text=The%20Boolean%20d
ata,1%20or%200
    if(digitalRead(PIR_MOTION_SENSOR))
        Serial.println("Hi,people is coming");
    else
        Serial.println("Watching");

 delay(200);
}
```

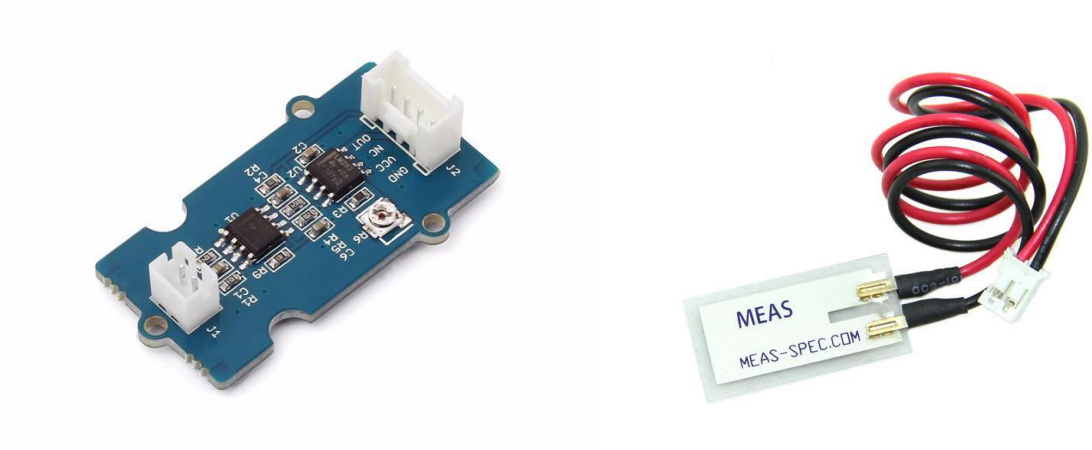
Code de test du capteur de mouvement

3) Conclusion

Nos tests pratiques et notre compréhension du capteur de mouvement a validé le fonctionnement de celui-ci. Les résultats obtenus et comparés aux attentes démontrent la fiabilité de ce composant.

C) Capteur de vibration

Dans la continuité de notre démarche visant à identifier les capteurs adaptés au renforcement du système de sécurité de notre environnement, nous orientons maintenant notre attention vers le capteur de vibration.



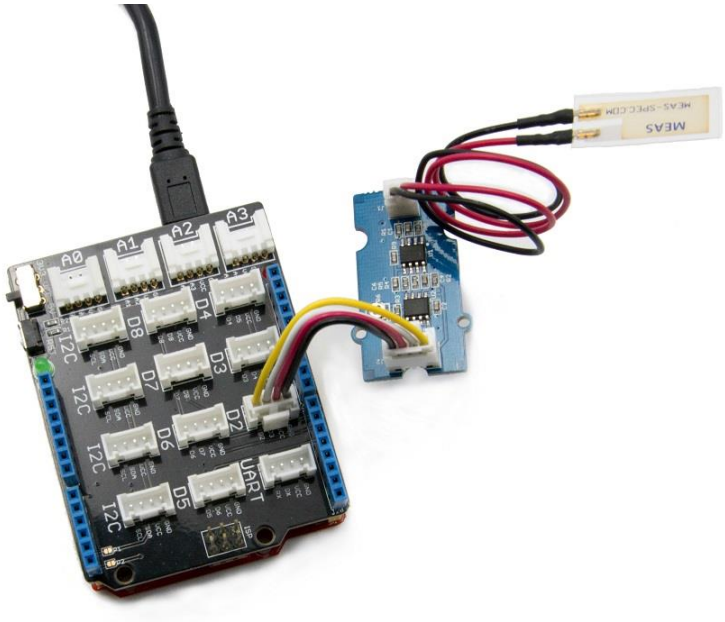
Capteur de vibration

1) Théorie

Le capteur de vibration est tout aussi un élément crucial pour détecter des mouvements indésirables tels que des tentatives d'effractions. Avant de procéder aux tests pratiques, une compréhension et des recherches approfondies étaient nécessaires.

Après quelques recherches, nous pouvons dire que le capteur fonctionne de cette façon : Il est composé d'un matériau sensible aux vibrations/pression qui converti les variations perçues en signaux électriques. La charge électrique convertie en signal électrique est ensuite convertie puis amplifiée pour une utilisation semblable à notre projet pour un système de sécurité par exemple.

Tout comme le capteur de mouvement, il se branche simplement via des câbles fournit par le constructeur, on relie donc la nappe du capteur doté du matériau sensible aux vibrations à la carte électronique associé, ensuite, on relie l'autre partie de la carte électronique à notre Arduino Grove Megashield.



Câblage du capteur de vibration

Le capteur sera donc alimenté en 5V et la sortie du capteur sera relié à une entrée digitale de notre choix.

2) Pratique

Pour notre expérimentation pratique, nous opterons pour une approche semblable au capteur précédent. On réalisera donc le câblage en utilisant le matériel fourni avec le capteur, puis on met en œuvre le code de test obtenu au préalable.

```
const int ledPin=13;
void setup() {
    Serial.begin(9600);
    pinMode(ledPin,OUTPUT);
}

void loop() {
    int sensorState = digitalRead(2);
    Serial.println(sensorState);
    delay(100);
    if(sensorState == HIGH)
    {
        digitalWrite(ledPin,HIGH);
    }
    else
    {
        digitalWrite(ledPin,LOW);
    }
}
```

Code de test du capteur de vibration

3) Conclusion

Le capteur de vibration joue aussi un rôle crucial dans la détection d'intrusion pour notre système de sécurité, cependant les tests et résultats obtenus ne sont pas concluants malgré diverses tentatives de réglage de la sensibilité du capteur les données obtenues concernant la détection de vibration restent très aléatoires.

D) Capteur de sons

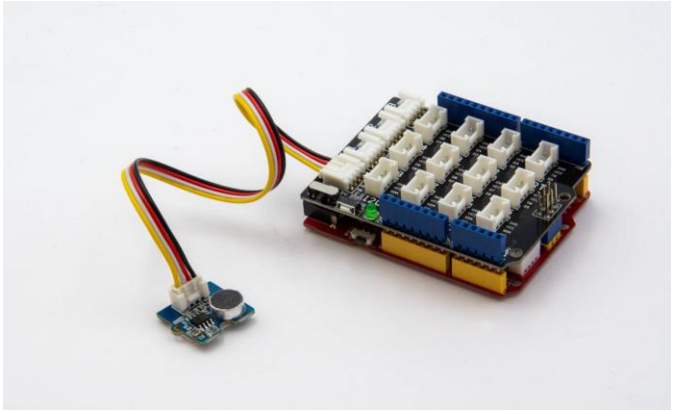
Pour suivre dans notre exploration sur les tests des capteurs adaptés à la sécurité de notre environnement, nous avons aussi testé un capteur essentiel dans le domaine de la sécurité qui est le capteur sonore.

Capteur de sons

1) Théorie

Le capteur de sons fonctionne sur le principe de variation des variations acoustiques en signaux électriques : Le capteur est équipé d'un microphone sensible aux ondes sonores, lorsqu'une onde sonore frappe la membrane du microphone, elle génère des variations de pression qui font vibrer la membrane. Ces sons sont alors convertis en signaux électriques, puis ce même signal est amplifié pour pouvoir être exploitable.

En ce qui concerne la partie câblage, on alimente et réceptionne la sortie du capteur de sons via le câble fourni par le constructeur en le reliant à notre module Grove Megashield.



Câblage du capteur de sons avec l'Arduino Mega et son module Grove

2) Pratique

Pour la partie pratique, on aura donc câblé comme l'image ci-dessus le capteur de sons et on utilisera le code de test adapté :

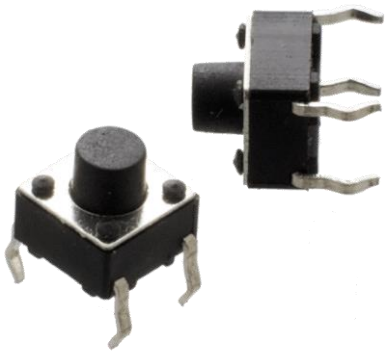
```
// test code for Grove - Sound Sensor
// loovee @ 2016-8-30

const int pinAdc = A0;

void setup()
{
    Serial.begin(115200);
    //Serial.println("Grove - Sound Sensor Test...");
}

void loop()
{
    long sum = 0;
    for(int i=0; i<32; i++)
    {
        sum += analogRead(pinAdc);
    }

    sum >>= 5;

    Serial.println(sum);
    delay(10);
}
```

3) Conclusion

Le capteur de sons joue un rôle vital dans la détection d'activités suspectes, nos tests et nos pratiques se sont révélés concluant, le capteur de sons est très fiable et constitue un des éléments qui pourrait faire partie de notre système de sécurité.

E) Bouton poussoir

Afin de ne pas nous attarder sur tous les capteurs que nous avons pu tester, car la liste reste longue (avec le capteur de température, capteur IR, capteur de mouvement, capteur de vibrations, capteur sonore et capteur de qualité de l'air…), notre exploration des capteurs pour renforcer la sécurité de notre domicile nous a également conduits à considérer le bouton poussoir comme une option intrigante et surprenante pour notre système de sécurité. Son utilisation ingénieuse en tant que capteur offre des possibilités intéressantes.



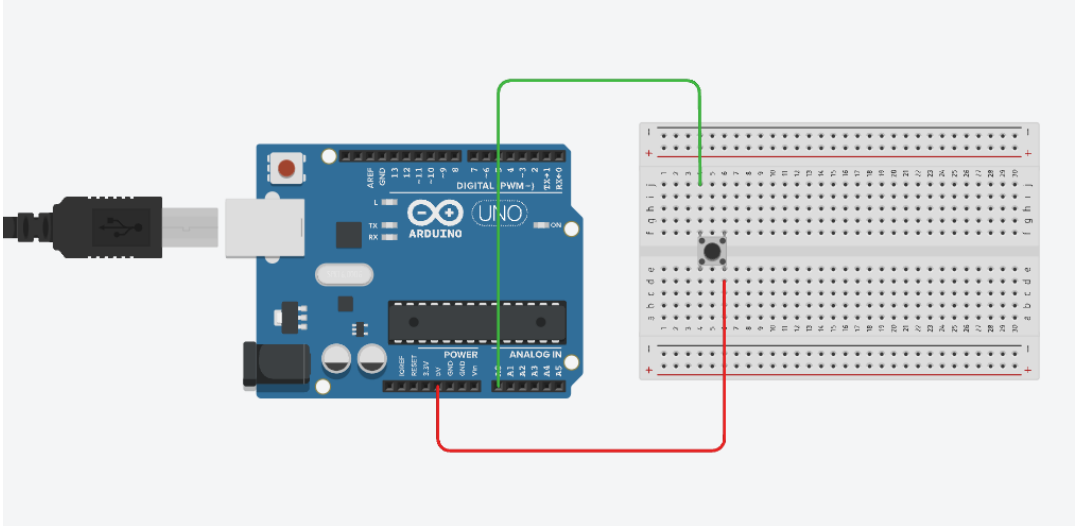
Bouton poussoir

1) Théorie

Le bouton poussoir est généralement utilisé de la plus simple des façons pour des interactions manuelles et cet objet peut être repensé dans notre contexte de système de sécurité. Il est composé de deux paires de broches reliées en diagonales et lorsqu'on appuie sur le bouton, ces broches se connectent. En tant que capteur, ce bouton poussoir pourrait être utilisé pour détecter une variation de son état électrique, passant de l'état ouvert à l'état fermé. Et donc cette variation peut être utilisée comme un signal de détection.

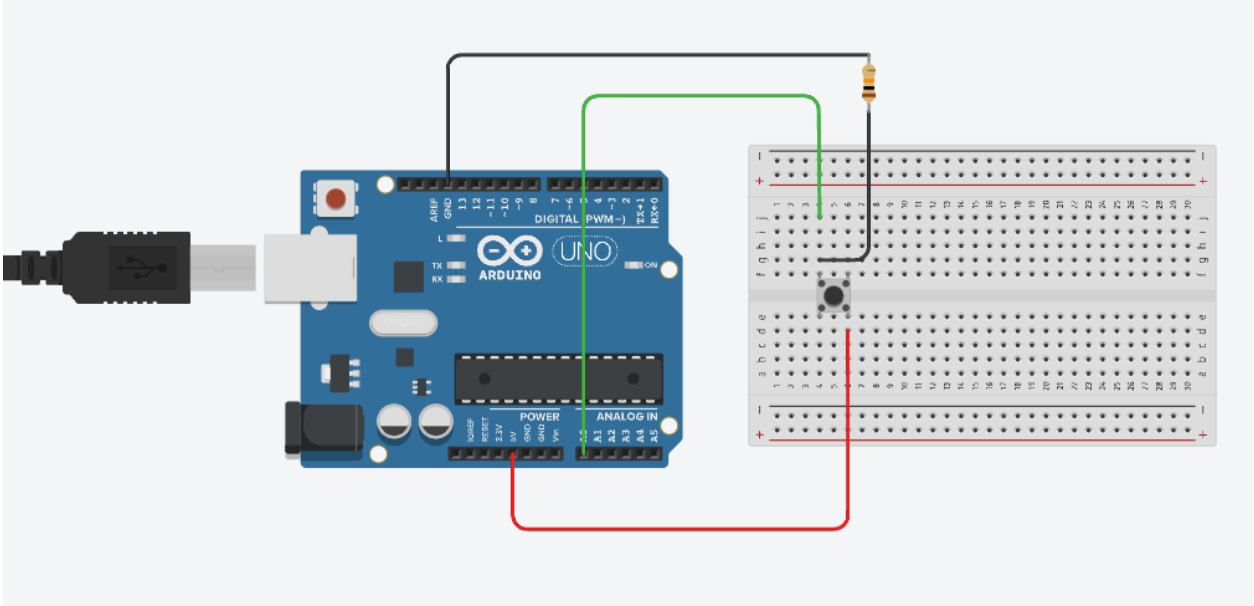
2) Pratique

En ce qui concerne la partie pratique, nous avons simplement alimenté le bouton poussoir en 5V d'un côté et de l'autre la sortie vers notre entrée analogique, mais aussi une résistance sur la masse qui sert de résistance de pull down, c'est-à-dire que si on laisse le câblage de notre bouton poussoir comme suit :



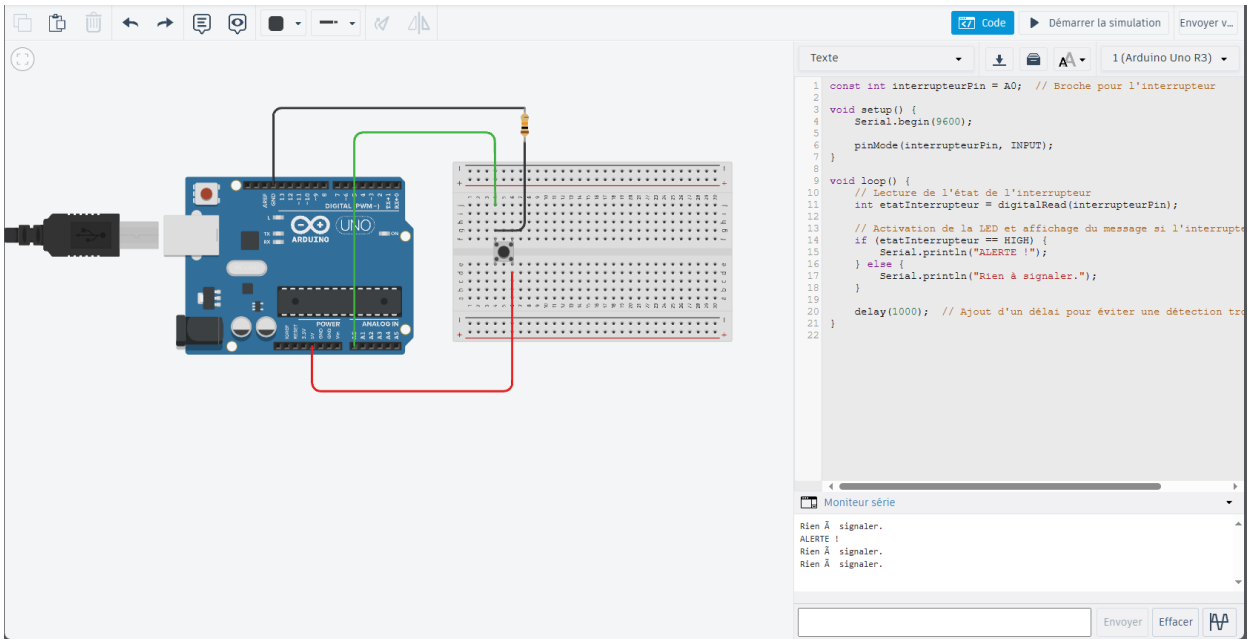
Câblage du bouton poussoir à ne pas reproduire

Un seul état seulement sera perçu, c'est-à-dire la partie ou le 5V est injecté quand on appuie sur le bouton, cependant on veut aussi que le circuit repasse à un état bas lorsqu'on n'appuie pas sur le bouton, pour cela, on ajoute donc la résistance à la masse qui sert à remettre à cet état bas le système.



Câblage du bouton poussoir

Pour pouvoir utiliser ce code, on programmera nous-mêmes un simple code qui détectera la variation d'état lorsqu'on appuie sur le bouton, cela aura pour effet d'avertir d'une intrusion.



Test du bon fonctionnement du bouton poussoir

```
const int interrupteurPin = A0;  // Broche pour l'interrupteur

void setup() {
    Serial.begin(9600);

    pinMode(interrupteurPin, INPUT);
}

void loop() {
    // Lecture de l'état de l'interrupteur
    int etatInterrupteur = digitalRead(interrupteurPin);

    // Affichage du message si l'interrupteur est activé
    if (etatInterrupteur == HIGH) {
        Serial.println("ALERTE !");
    } else {
        Serial.println("Rien à signaler.");
    }

    delay(1000);  // Ajout d'un délai pour éviter une détection trop fréquente
}
```

Code de test du bouton poussoir

3) Conclusion

Le bouton poussoir, souvent négligé pour des utilisations simples, a démontré sa polyvalence en tant que capteur pour notre système de sécurité. Sa simplicité et sa facilité d'utilisation en font une option surprenante, mais efficace.

F) Conclusion

Pour conclure sur le choix et le déploiement de nos capteurs, qui est une étape importante qui nous a amenés à explorer les diverses solutions de sécurisation de notre système, chacun offre une

fonctionnalité différente et exploite une vision de sécurité spécifique pour renforcer la sécurité de notre système. Notre approche méthodique combinant théorie et pratique-nous a permis de déterminer quelles seront les composantes de notre système de sécurité.

Après des tests approfondis, deux capteurs auront retenu notre attention, ils se sont distingués comme les choix les plus évidents, les plus pertinents et efficaces pour maintenir en sécurité notre domicile : le capteur de mouvement grâce à son principe de détection infrarouge qui en fait un élément fiable et précis, mais aussi par son large porté qui couvrira le moindre recoin de notre maison. Et ensuite le bouton poussoir, bien que souvent perçu comme simple, il s'est révélé être une option des plus efficaces en tant que capteur. Il est basique et facile d'utilisation pour détecter des variations.

Le choix de ces capteurs est motivé par la simplicité d'utilisation de ceux-ci, mais aussi pour garantir une mise en œuvre fluide et une gestion aisée, en optant pour une mise en place moins complexe, on garantit une meilleure fiabilité de détection d'intrusion et on évite donc les faux positifs.

Enfin, l'analyse de cycle de vie de notre dispositif a influencé notre choix en faveur de ces capteurs grâce à leur faible consommation d'énergie qui contribue à la durabilité environnementale.

Ainsi, la combinaison de deux capteurs de mouvement et d'un bouton poussoir nous a paru comme un choix évident comme solution idéale pour notre système de sécurité.

Voici le bon de commande associé à nos choix :

BON DE COMMANDE N° 1
Date d'émission : 26/10/2023

Code client : 5818914918
BEHILLIL SOFIANE
36 rue du Pelvoux ZI de la Petite Montagne Nord
91080 Courcouronnes
France
0766538795

GEII Group
36 rue du Pelvoux ZI de la Petite Montagne Nord
91080 Courcouronnes
France
Tél : 0606060606
Email : geii@gmail.com
352310767
Evry B 352 310 767

Bon de commande pour capteurs

Adresse de livraison

Destinataire : BEHILLIL SOFIANE
Adresse : 36 rue du Pelvoux ZI de la Petite Montagne Nord
Code postal : 91080 Ville : Courcouronnes
Pays : France
Téléphone : 0766538795 Email : behillils41@gmail.com
Date de livraison : 27/10/2023

Référence	Désignation	Quantité	Unité	Prix U HT	Montant (HT)
101020353	Grove - PIR Motion Sensor	2.00	U	8.00	16.00
COM-00097	SparkFun COM-00097 (Mini Pushbutton Switch)	1.00	U	0.33	0.33
				Total HT	16.33
				Net à payer (€)	**16.33**

TVA non applicable, art. 293 B du CGI

Réglement
- Date limite : 26/10/2023 (À la réception)

Vous disposez d'un délai de rétractation de 14 jours.

GEII Group 36 rue du Pelvoux ZI de la Petite Montagne Nord 91080 Courcouronnes Tél : 0606060606 Page 1/1

Bon de commande des capteurs

IV) Communication sans fil

A) Communication FM

Dans le cadre de notre projet de mis en place d'un système de sécurité pour notre domicile, la communication sans fil est un des besoins nécessaires au bon fonctionnement de notre application. En effet, il est important de pouvoir assurer une communication sans fil afin de gérer les différentes alertes associées aux capteurs. C'est pour cela que nous avons opté pour le Grove - 433MHz Simple RF Link Kit, qui propose une communication en modulation de fréquence (FM).

Kit de communication sans fil FM

1) Grove - 433MHz Simple RF Link Kit

Le module de communication FM offre une solution simple et efficace de communication sans fil entre les capteurs et notre Arduino, voici pourquoi nous avons ce module :

- La fréquence de 433MHz : Sa fréquence de 433MHz offre une portée plus que suffisante et adéquate pour notre environnement tout en évitant les interférences.

- Facilité d'utilisation : Ce kit Grove est conçu pour sa simplicité d'utilisation avec l'Arduino. L'émetteur et le récepteur sont simples à câbler et à implémenter à notre système.

- Compatibilité Arduino : Un des derniers points et pas des moindres, c'est que nous pouvons donc utiliser ce kit avec notre Arduino et son Grove Megashield qui simplifie son câblage.

La communication FM en lui-même présente aussi quelques points avantageux pour notre système de sécurité :

- Stabilité de la transmission : La modulation en fréquence offre une stabilité vis-à-vis de la transmission entre l'émetteur et le récepteur. On ne devrait donc pas avoir de problème de communication entre l'émetteur et le récepteur.
- Consommation d'énergie : Dans le cadre de l'analyse de cycle de vie, l'impact environnementale de notre système de communication sans fil est aussi une des contraintes à respecter, la faible consommation d'énergie de ce mode de communication est une des raisons pour laquelle nous avons choisi ce mode de communication.

2) Pratique

Comme dit précédemment le kit Grove - 433MHz Simple RF Link Kit est simple d'utilisation, pour câbler celui-ci, on utilise simplement tout comme nos capteurs les câbles fournis par le constructeur que l'on relie d'une part à notre Arduino et de l'autre à la carte d'émission et de même pour la réception.

Pour tester notre communication tout comme pour l'utilisation de nos capteurs, on utilisera le code de test fourni d'une part pour l'émission et de l'autre pour la réception :

```
#include <VirtualWire.h>

int RF_TX_PIN = 10;

void setup()
{
  vw_set_tx_pin(RF_TX_PIN); // Setup transmit pin
  vw_setup(2000); // Transmission speed in bits per second.
}

void loop()
{
  const char *msg = "hello";
  vw_send((uint8_t *)msg, strlen(msg));  // Send 'hello' every 400ms.
  delay(400);
}
```

Code de test d'émission

```
#include <VirtualWire.h>

int RF_RX_PIN = 2;

void setup()
{
  Serial.begin(9600);
  Serial.println("setup");
  vw_set_rx_pin(RF_RX_PIN);  // Setup receive pin.
  vw_setup(2000); // Transmission speed in bits per second.
  vw_rx_start(); // Start the PLL receiver.
}

void loop()
{
  uint8_t buf[VW_MAX_MESSAGE_LEN];
  uint8_t buflen = VW_MAX_MESSAGE_LEN;
  if(vw_get_message(buf, &buflen)) // non-blocking I/O
  {
    int i;
    // Message with a good checksum received, dump HEX
    Serial.println("Got: ");
    for(i = 0; i < buflen; ++i)
    {
      //Serial.print(buf[i], HEX);
      Serial.print(" ");
      Serial.println((char)buf[i]);
    }
    Serial.println("");
  }
}
```

Code de test de réception

3) Conclusion

Le choix de la communication est donc aussi une étape importante dans le cadre de notre projet de mis en place d'un système de sécurité domestique. Le module est de ce fait flexible et assurera notre communication nécessaire pour une installation facile tout en garantissant une fiabilité et une stabilité à notre système.

V) Implantation du système

Pour terminer, une des dernières étapes fondamentales de notre projet, l'implantation de notre système de sécurité, nous avons donc réussi à concrétiser nos efforts en intégrant les capteurs soigneusement sélectionnés dans une maquette que nous avons pu créer et qui représente notre environnement domestique. Cette étape marque le début d'une nouvelle étape où la théorie, c'est concrétisé.

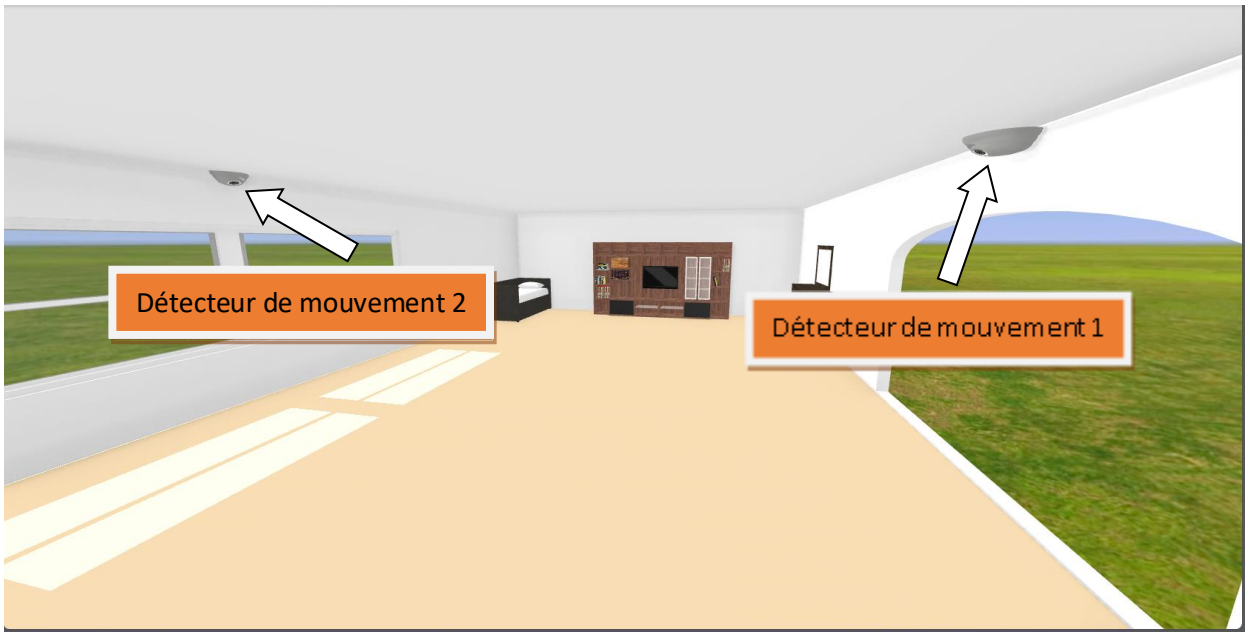
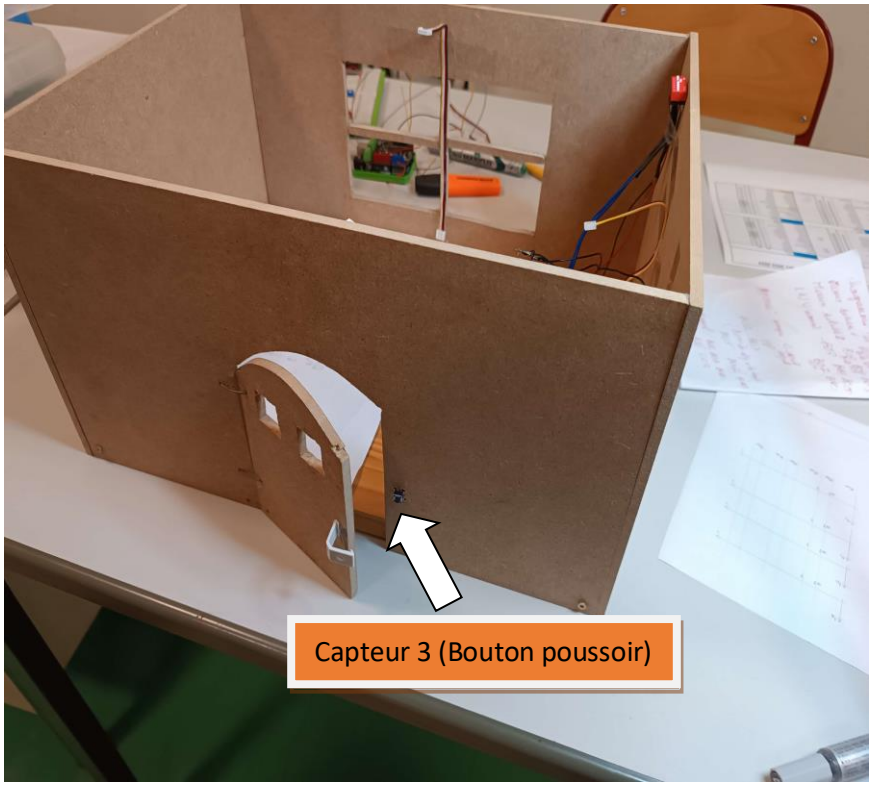
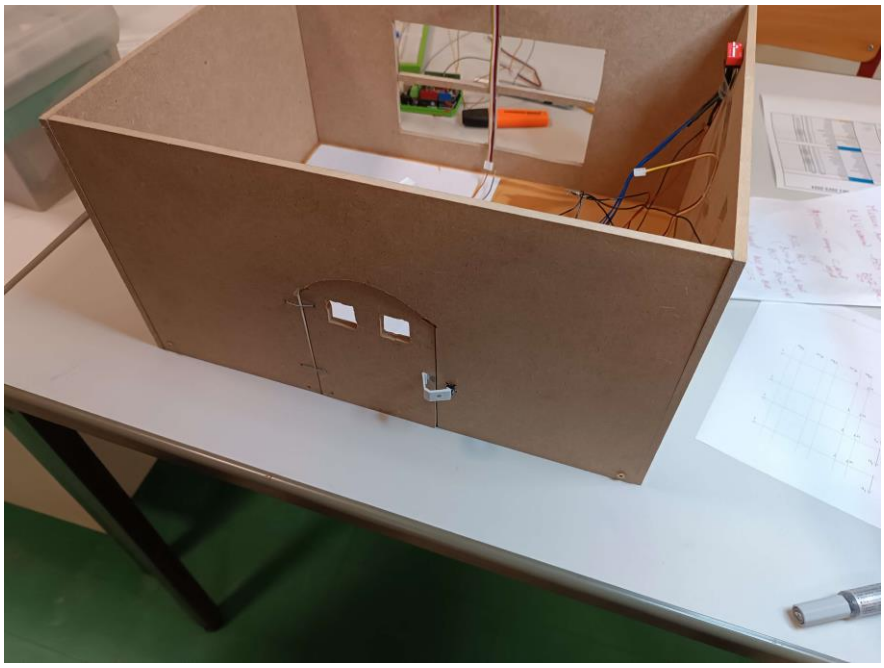


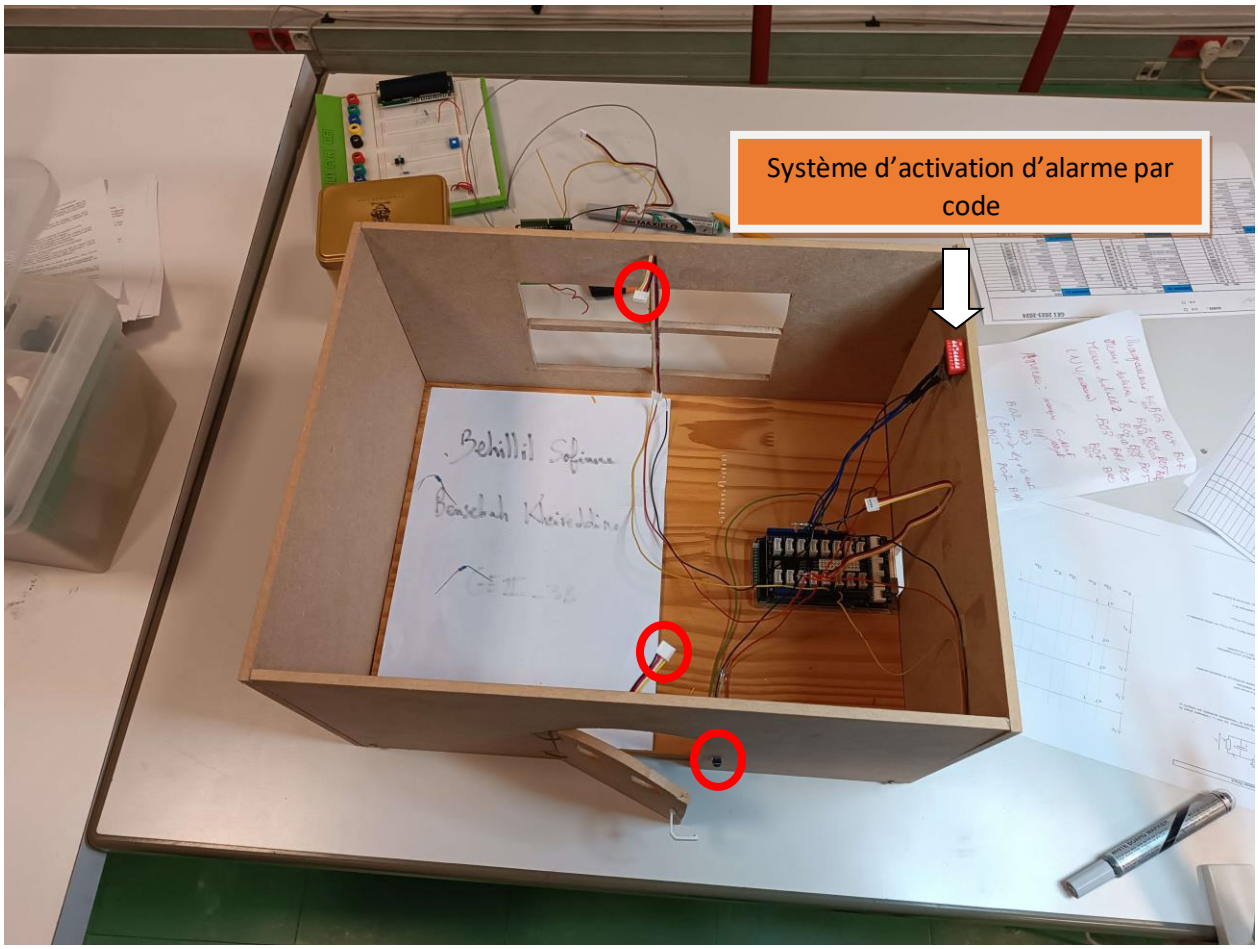
Schéma 3D de la conception de notre maison



Maquette de notre maison avec la porte ouverte



Maquette de notre maison avec la porte fermé



Vue de haut de notre maquette et de ses capteurs

Nous avons donc pris soin de placer les différents capteurs à des endroits stratégiques de sorte à recouvrir la plus grande partie des entrées par lesquels une intrusion peut avoir lieu. Nous avons donc en cas d'effraction par la fenêtre un premier capteur de mouvement qui s'activera lors d'un passage par la fenêtre lorsque l'alarme est en état dit « actif » c'est-à-dire lorsque via le système de code que nous avons mis en place l'alarme est actif. Ensuite, un deuxième capteur de mouvement au niveau de la porte est placé, si un mouvement est détecté proche de celui-ci, une alerte est aussi envoyée. Et pour terminer, nous avons utilisé le bouton poussoir au niveau de la porte, lorsque celle-ci est ouverte, lorsque l'alarme est activée une alerte est aussi envoyée, c'est donc un système autant efficace en cas d'effraction que pour les oublies de fermer la porte derrière soi quand on sort.

VI) Conclusion

Pour conclure sur ce compte rendu, durant cette SAE, nous avons pu mettre en place un système de sécurité, concrétisé nos connaissances acquises pour choisir et intégré ce système mis en place. Au-delà de la technique, ce projet incarne notre volonté de réussite et notre vision d'une maison intelligente.

CODE :

```
#include <VirtualWire.h>

// Broches pour les capteurs
const int brocheCapteurMouvement1 = 6;
const int brocheCapteurMouvement2 = 8;
const int brocheInterrupteurPorte = A2;

// Broche pour l'émetteur RF
int RF_TX_PIN = 10;

// Variables qui seront associés aux capteurs
int valeurCapteurMouvement1;
int valeurCapteurMouvement2;
int valeurInterrupteurPorte;

bool alarmeActive = false; // Ajout d'une variable pour suivre l'état de
l'alarme

void setup() {
    // Définition les broches A3, A4, A5, A6 en tant qu'entrées pour le
système de code
    pinMode(A3, INPUT);
    pinMode(A4, INPUT);
    pinMode(A5, INPUT);
    pinMode(A6, INPUT);

    Serial.begin(9600);

    // Configuration des broches des capteurs en tant qu'entrées
    pinMode(brocheCapteurMouvement1, INPUT);
    pinMode(brocheCapteurMouvement2, INPUT);
    pinMode(brocheInterrupteurPorte, INPUT);

    // Configuration de la broche de l'émetteur VirtualWire
    vw_set_tx_pin(RF_TX_PIN);
    vw_setup(2000);
}
```

```
void loop() {
    // Lecture des valeurs des capteurs
    valeurCapteurMouvement1 = digitalRead(brocheCapteurMouvement1);
    valeurCapteurMouvement2 = digitalRead(brocheCapteurMouvement2);
    valeurInterrupteurPorte = digitalRead(brocheInterrupteurPorte);

    // Lire l'état des broches A3, A4, A5 et A6
    int etatA3 = digitalRead(A3);
    int etatA4 = digitalRead(A4);
    int etatA5 = digitalRead(A5);
    int etatA6 = digitalRead(A6);

    // Envoi de l'état de l'alarme
    if (etatA3 == HIGH && etatA4 == LOW && etatA5 == HIGH && etatA6 == LOW) {
        const char *msg = "Alarme ON";
        vw_send((uint8_t *)msg, strlen(msg));
        vw_wait_tx();
        Serial.println("Alarme ON");
        // Activer les capteurs de mouvement et le capteur de porte si
l'alarme est en ON
        alarmeActive = true;
    } else {
        const char *msg = "Alarme OFF";
        vw_send((uint8_t *)msg, strlen(msg));
        vw_wait_tx();
        Serial.println("Alarme OFF");

        // Désactiver les capteurs de mouvement et le capteur de porte si
l'alarme est en OFF
        alarmeActive = false;
    }
```

```
    // Vérification si l'un des capteurs de mouvement est actif on active
l'alarme
    if (alarmeActive && (valeurCapteurMouvement1 == HIGH ||
valeurCapteurMouvement2 == HIGH)) {
        Serial.println("Intrusion détectée !");

        // Envoi du message avec VirtualWire
        const char *msg = "Intrusion détectée !";
        vw_send((uint8_t *)msg, strlen(msg));
        vw_wait_tx();
    } else if (alarmeActive && valeurInterrupteurPorte == LOW) { //Vérification
si le capteur de porte est actif on active l'alarme
        Serial.println("Intrusion par la porte détectée !");

        // Envoi du message avec VirtualWire
        const char *msg = "Intrusion par la porte détectée !";
        vw_send((uint8_t *)msg, strlen(msg));
        vw_wait_tx();
    } else {
        Serial.println("RAS!");  // RAS si aucun des capteurs est actif

        // Envoi du message "RAS!" avec VirtualWire
        const char *msg = "RAS!";
        vw_send((uint8_t *)msg, strlen(msg));
        vw_wait_tx();
    }

    delay(500);
}
```

Code final d'émission en cas d'alerte en cas d'intrusion

```
#include <VirtualWire.h>

// Broche pour le récepteur RF
int RF_RX_PIN = 2;

void setup() {
    Serial.begin(9600);

    // Configuration de la broche du récepteur VirtualWire
    vw_set_rx_pin(RF_RX_PIN);
    vw_setup(2000); // Vitesse de transmission en bits par seconde.
    vw_rx_start();
}

void loop() {
    uint8_t buf[VW_MAX_MESSAGE_LEN];
    uint8_t buflen = VW_MAX_MESSAGE_LEN;

    if (vw_get_message(buf, &buflen)) {
        // Message reçu

        // Convertir le tableau d'octets en une chaîne de caractères
        buf[buflen] = '\0';
        String message = String((char*)buf);

        // On fait afficher le message sur le moniteur série
        Serial.println("Message reçu : " + message);

        // Faire quelque chose en fonction du contenu du message
        if (message == "Intrusion détectée !") {
            Serial.println("Intrusion détectée !");
        } else if (message == "Intrusion par la porte détectée !") {
            Serial.println("Intrusion par la porte détectée !");
        } else if (message == "RAS!") {
            Serial.println("RAS!");
        } else if (message == "Alarme ON") {
            Serial.println("Alarme activée");
        } else if (message == "Alarme OFF") {
            Serial.println("Alarme désactivée");
        }
    }}
```

Code final de réception en cas d'alerte en cas d'intrusion