Rapport d'activités des semaines du 9 au 20 mai 2022

TUELEAU Tom

25 mai 2022

Table des matières

1 Introduction

Ce document a pour objectif de faire un état d'avancement du stage. Celui-ci résumera donc le travail fait lors de la fin du mois d'avril et la premier moitié du mois de mai.
Dans un premier temps, je reviendrait sur le montage amplificateur vue lors du dernier rapport et vous montrerait les résultats obtenue.
Une seconde partie présentera les programmes crée afin de récolter et traiter les données des capteurs.
Une troisième partie traitera de l'envoie des données entre les différents élément du système.
Enfin, je conclurait sur le travail effectuer et les difficultés rencontrer.

2 Finalisation du montage

Lors des précédents rapport [1], je vous ai exposer mes recherches et résultats quand au dimensionnement d'un montage amplificateur. Ce besoins c'était fait ressentir quand je m'était rendue compte que le signal émis par le piézo-électrique n'arrivait pas à être capter par l'Arduino. Dans cette partie nous verrons tout d'abords la finalisation du montage et dans un second temps nous verrons l'installation de celui-ci dans le rucher.

2.1 Montage

Lors de cette semaine j'ai pu effectuer le prototype final incluant l'amplification du signal, le piézo-électrique, le capteur de température et d'humidité (Si7021) et l'Arduino. Vous pouvez voire un schéma complet Figure 1.

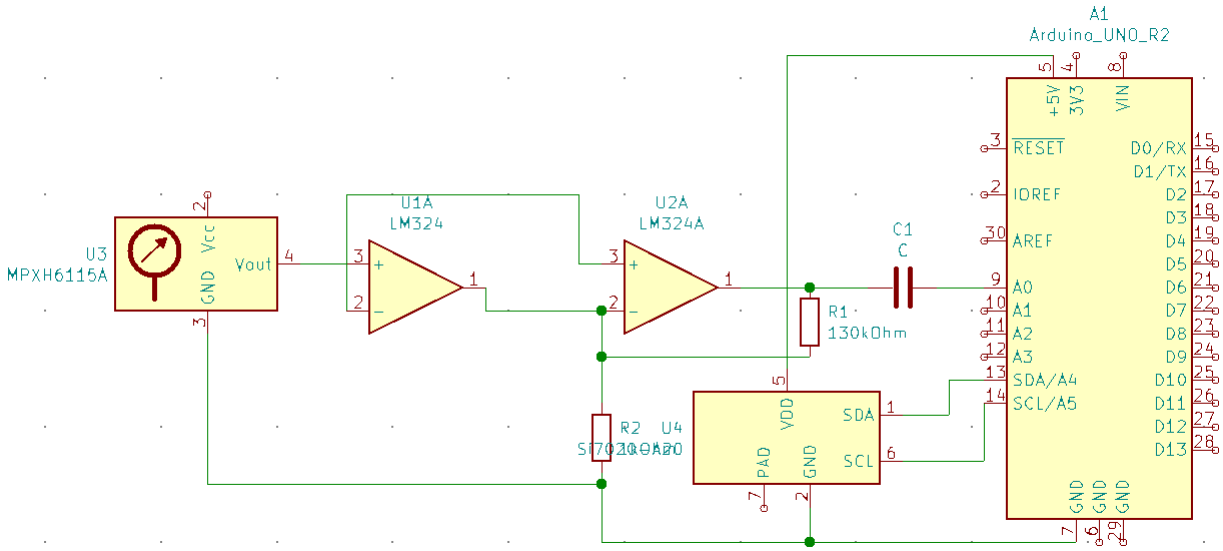

FIGURE 1 – Schéma montage final

Comme nous le verrons lors de la partie programmation, j'arrive à récupérer le signal envoyer par le piézo-électrique avec l'Arduino. Cella vas donc me permettre d'échantillonner celui-ci et de le traiter.

2.2 Installation dans la ruche

L'objectif étant de récolter les données dans la ruche, j'ai du installer les capteurs dans celle-ci. Comme vous pouvez le voire Figure 2, j'ai glisser les différent capteur dans la ruche et leur ai souder des files assez long pour les brancher par la suite sur un microcontrôleur.

1. Voir "Rapport d'activités du 11 et 18 avril 2022", partie 4, page 7

FIGURE 2 – Capteurs dans la ruche

2.3 Résultat

Aprés avoir échantilloner le signal obtenue par le piezo et l'avoire envoyer aux serveur de traitement, j'ai pu obtenire le graphique FIGURE 3 [2].

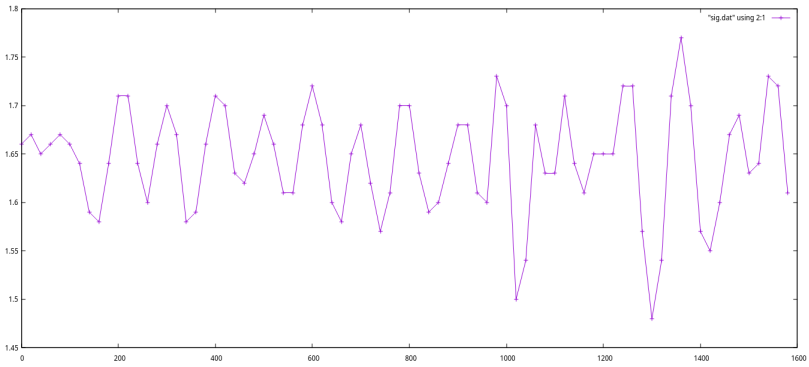

FIGURE 3 – Graph signale piezo

2. Abscisse : temps en ms Ordonné : tension en V

2.4 Amélioration

Maintenant que ce montage est fonctionnel, j'aimerais pouvoir lui apporter des améliorations. La première serai la création d'un PCB pour le montage amplificateur afin de rendre le montage plus professionnel. La seconde serait la création d'un boitier pouvant le contenir lui et l'Arduino.

3 Programmation

Tout aux long des dernier semaine, j'ai eux à programmer plusieurs fonctionnalité. Dans cette partie je vais donc commencer par vous présenter les programme effectuer pour récolter les données des capteurs et effectuer leur traitement. Une deuxième partie abordera la programmation de la mise en communication des différents composant du système.

3.1 Programmation des capteurs

Pour ma premier mission j'avais trois données à récupérer. Celle-ci sont l'humidité, la température et les vibration. Afin de récolter ces données j'ai choisi en tout trois capteur Un capteur piézo-électrique , un capteur de température et d'humidité (Si7021) et un microphone (INMP441). Dans un premier temps je vous parlerait du capteur de température et d'humidité. Une seconde partie abordera la programmation du capteur de vibration. Enfin une dernier partie abordera le traitement des données.

3.1.1 Capteur de température et humidité

Comme indiquer précédemment le capteur utiliser pour ces données est le Si7021. Étant données que le microcontroleur que j'utilise est un arduino, il existe des librairies donnant accès à des fonction permettant d'obtenir les données voulus.

```
1  #include <Adafruit_Si7021.h>
2
3  Adafruit_Si7021 sensor = Adafruit_Si7021();
4
5  void callback(char* topic, byte* payload, unsigned int length) {}
6
7  void setup() {
8    Serial.begin(9600);
9   if (!sensor.begin()) {
10    Serial.println("Did not find Si7021 sensor!");
11    while (true)
12      ;
13   }
14 }
15
16 void loop() {
17   double hum,temp;
18   char Ctemp[4], Chum[4];
19   hum = sensor.readHumidity();
20   temp = sensor.readTemperature();
21   dtostrf(temp,2,2,Ctemp);
22   dtostrf(hum,2,2,Chum);
23  delay(1000);
24 }
25
```

Les fonction principals de ce programme sont , "sensor.begin" qui initialise la liaisons entre l'Arduino et le capteur, readHumidity et readTemperature permet comme leurs nom l'indique de récupérer la température et l'humidité messurer par le capteur. Les valeurs retourner par ces deux fonction sont de type "double". Afin de pouvoire les envoyer, je les convertis en chaine de caractère à l'aide de la fonction "dtostrf".

3.1.2 Capteur de vibration

Après avoir amplifier le signal du piézo-électrique, j'ai pu me consacrer à l'acquisition des données de cellui-ci. Afin de faciliter sont acquisition j'ai créée une fonction "udpSendVibration" prenant la quan-

titer d'échantillon souhaités, le nombre de decimal et la pęriode d'échantillonage (en milli seconde).

```
1
2  void udpSendVibration(int nbEchantillon, int nbVirgule, int periodeEchantillonageMs)
3  {
4
5     int nbChar = nbVirgule+2;
6     char tmp[nbEchantillon][nbChar];
7     char data2[nbEchantillon*nbChar];
8     double centsValeurs[nbEchantillon];
9     int i,y;
10    for(i=0;i<nbEchantillon;i++){
11       centsValeurs[i]=analogRead(piezo);
12       centsValeurs[i]=(centsValeurs[i]*5)/1023;
13       delay(periodeEchantillonageMs);
14    }
15    delay(2000);
16    for(i=0;i<nbEchantillon;i++){
17       dtostrf(centsValeurs[i],nbChar,nbVirgule,tmp[i]);
18       for(y=0;y<nbChar;y++){
19          data2[(nbChar*i)+y]=tmp[i][y];
20          Serial.println(data2[(i*nbChar)+y]);
21       }
22    }
23
24 }
```

3.1.3 Traitement des données

Une fois le signal obtenue il fallait l'annalyser en fréquence. Pour cella j'était dans un premier temps partie pour traiter directement sur l'Arduino en effectuant une transformer de Fourrier du signal. Cependant les capaciter de calcule de l'Arduino étant limiter et ayant observer une certaie lenteur à effectuer cette opération j'ai décider d'effectuer le systéme suivant. J'envoie les échantillons à un ordinateur distant qui effectura le traitement.
J'ai donc par la suite écris un programme utilisant la librairie "fftw". Celle-ci proposant de multiple fonction basser sur l'algorithme FFT (Fast Fourier Transform), permetant d'effectuer une transformer de fourrier sur les signaux numérisés.

```
1
2     unsigned int N = 20;
3     int i;
4     double *out = malloc(sizeof(double) * N);
5     double list[]={3.920000,0.160000,5.630000,4.750000,0.370000,2.010000,
6     5.870000,6.730000,9.870000,4.340000,9.950000,5.460000,8.420000,3.120000,
7     3.670000,4.870000,3.280000,6.050000,4.650000,0.500000};
8     double * values;
9     values = malloc(sizeof(double)*N);
10    fftw_plan p;
11    p = fftw_plan_r2r_1d(N,values,out,FFTW_HC2R,FFTW_MEASURE);
12
13    for(i=0;i<20;i++){
14          values[i]=list[i];
15    }
16
17    fftw_execute(p);
18    for(i=0;i<20;i++){
```

```
19          printf("%f ",out[i]);
20      }
21      printf("\n");
22      fftw_destroy_plan(p);
23      free(out);
```

Dans l'exemple ci-dessus on effecue la transformer de fourirer du signal composer des point de la liste "list".

3.2 Programmation réseaux

Afin d'acheminer les information jusqu'aux different composante (site web, serveur) j'ai eux à programmer plusieurs liaisons réseaux. Je vais donc commencer par vous présenter la topologie de cellui-ci avec les différent éléments le composant ainsi que leur role. Dans un second temps je détaillerait la liaisons MQTT reliant la base de données et l'Arduino. Une dernier partie traitera de la communication des données au serveur.

3.2.1 Topologie du réseaux

Comme dis précédement le système est composer de plusieur éléments. Tout d'abords l'Arduino récolte les données et les envoie aux autre composante. La température et l'humiditer sont envoyer via MQTT vers le serveur tandis que les données du piezo sont envoyer via UDP. Une fois traiter ces dernier sont aussi envoyer aux sur un topic MQTT.

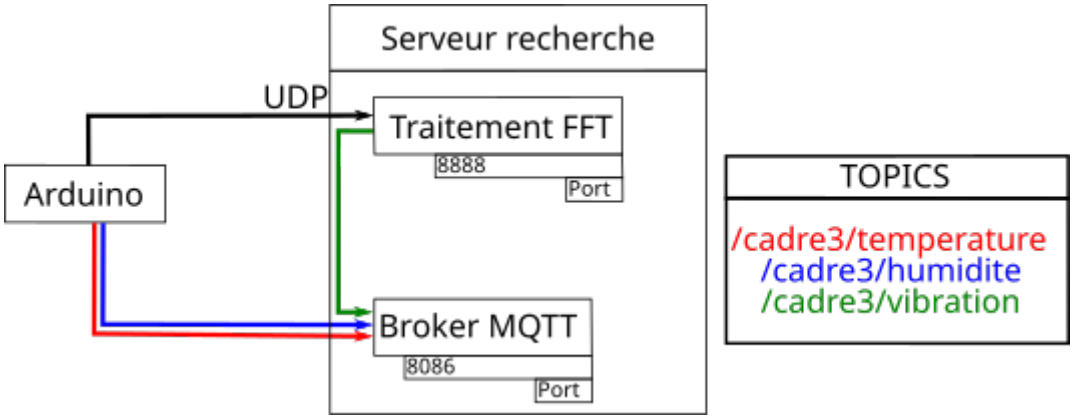

FIGURE 4 – Schema Réseaux

3.2.2 Liaisons MQTT

Afin d'envoyer les données du microcontroleur vers le serveur j'ai choisi le protocle MQTT et ce pour deux raisons.La premier est que le projet seras amener à évoluer et contiendras plusieurs capteur récoltant la même données à des endroits different de la ruche. Ainsi avec le systéme de topics, on pouras identifier facilement ces différent capteur. La deuxième est que la solution envisager pour le site web, et la base de données, est trés facilement couplabe avec ce protocole. Ci-dessous on peut voire le code utiliser pour envoyer sur les topic "cadre3/temperature" et "cadre3/humidite".

```
1       ...
2       if(!client.publish("cadre3/temperature",Ctemp))
3       {
4           Serial.println("error");
5       }
6       if(!client.publish("cadre3/humidite",Chum))
7       {
8           Serial.println("error");
9       }
```

```
10    ...
```

3.2.3 Liaisons UDP

Afin de transmetre les données du piezo aux serveur, j'ai choisis d'effectuer une liaisons UDP. J'ai choisi ce protocole car la liaisons n'as pas besoins de haute fideliter. L'arduino envoie donc sur le serveur les données du piezo. Le programme ci-dessous crées une crées une socket et attends que l'arduino envoie ces données.

```
1
2     ...
3
4     int sockfd, len, n, echantillon, nbchar;
5     char buffer[MAXLINE];
6     struct sockaddr_in servaddr, cliaddr;
7
8     if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 ) {
9        perror("Erreur lors de la creation de la socket");
10       exit(EXIT_FAILURE);
11    }
12
13    memset(&servaddr, 0, sizeof(servaddr));
14    memset(&cliaddr, 0, sizeof(cliaddr));
15
16    servaddr.sin_family = AF_INET;
17    servaddr.sin_addr.s_addr = INADDR_ANY;
18    servaddr.sin_port = htons(PORT);
19
20    if ( bind(sockfd, (const struct sockaddr *)&servaddr,sizeof(servaddr)) < 0 )
21    {
22       perror("bind failed");
23       exit(EXIT_FAILURE);
24    }
25
26    len = sizeof(cliaddr); //len is value/result
27    for(;;){
28       n = recvfrom(sockfd, (char *)buffer, MAXLINE,MSG_WAITALL, ( struct sockaddr *) &cliaddr,&
       len);
29       buffer[n] = '\0';
30       sendto(sockfd, "ACK", strlen("ACK"),MSG_CONFIRM, (const struct sockaddr *)&cliaddr,sizeof(
       cliaddr));
31
32    }
33
34    ...
```

4 Conclusion

En conclusion, maintenent que les données on été récolter et achemineril faut pouvoire les afficher sur un site web. Il me reste aussi à effectuer un montage final plus propre pour la partie electronique du système.

Table des figures