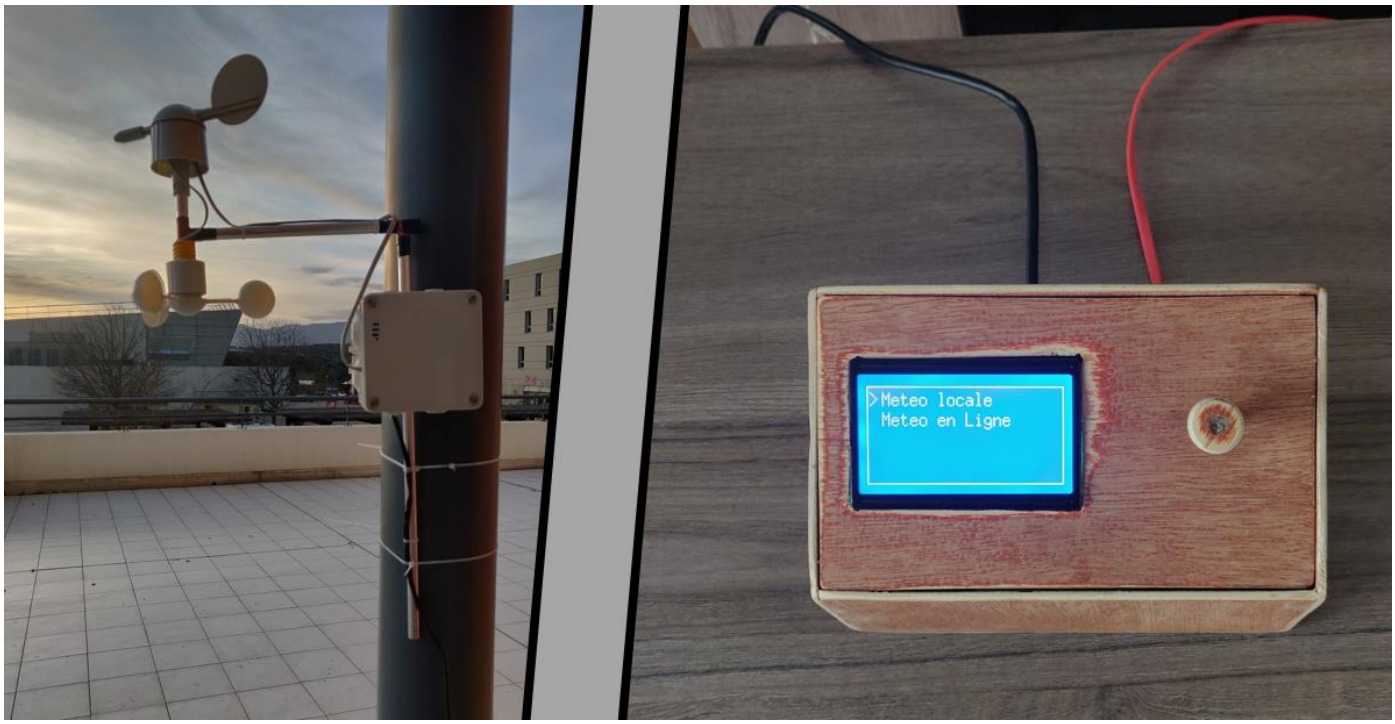
PROJETS ARDUINO– PeiP2

Année scolaire 2018-2019

Météo 2000



Etudiants : Victor Girard & Benjamin Ferrer

Encadrants : M. Pascal Masson & M. Nassim Abderrahmane

SOMMAIRE

I. Introduction

I.1 Le projet

L'objectif de ce projet était simple ; nous voulions réaliser une station météo qui nous permette de mesurer les données météorologiques locales, mais aussi, grâce à une connexion internet nous permettre de récupérer la météo de n'importe quelle ville du monde, via le code postal et le pays. De plus il était important pour nous de faire une station météo à la fois simple d'utilisation et potentiellement praticable par le plus grand nombre. Pour la météo locale nous voulions pouvoir mesurer les points suivants :

- La température.
- L'humidité.
- La pression.
- Le vent (vitesse/direction).

Et en plus de ces mesures nous voulions que le boitier extérieur et le boîtier intérieur communiquent entre eux via une connexion sans fil (Bluetooth ou RF).

Pour la météo en ligne nous voulions les points suivants :

- Accès à la météo de n'importe quelle ville.
- Météo actualisée en direct.
- Affichage de certaines informations.

Notre objectif principal c'est avant tout de faire un projet modulable.

Ce que nous avons finalement fait se rapproche fortement de ce que nous voulions faire, les différences notables sont notamment les différentes communications entre les cartes Arduino et entre l'Arduino du boitier intérieur et l'ESP. Par souci de fiabilité du Bluetooth, connexion très peu stable et facilement perturbable nous avons choisi une connexion filaire. De même pour la communication entre l'esp et l'Arduino du boitier intérieur, comme prévu à l'origine.

En ce qui concerne le boitier extérieur, par souci de matériel nous avons été limités à la mesure du vent (vitesse + direction), et aux mesures de températures et d'humidité. Cependant grâce à notre Shield Arduino (cf. matériel) notre projet reste modulable et nous pouvons rajouter sans souci des modules tels qu'un capteur de pluie notamment.

Pour la communication en wifi via l'ESP la seule chose que nous n'avons pas réussi à faire parfaitement est le choix de n'importe quelle ville dans le monde pour l'affichage de la météo, nous avons dû revoir nos idées et avons finalement trouvé un compromis : nous pouvons choisir la météo parmi une liste de villes prédéfinie. Cette liste reste cependant extensible, dans notre code nous n'en avons mis qu'une dizaine, extensible avec pour seule limite la place disponible sur la carte.

Nous avons aussi décidé de n'afficher que les informations que nous sommes en mesure de mesurer en locale, sachant que cette fois si aussi, nous pourrions afficher toutes les informations que l'on reçoit mais nous n'avons pas trouvé cela intéressant pour un utilisateur lambda.

I.2 Plannings

Planning initial :

SEANCES	1	2	3	4	5	6	7	8	9
Meteo en reseau									
Anémomètre									
Capt. Température/humidité									
Communication entre les bords									
Interface									
Boitier									
Pluviomètre									
Girouette									
Mise en commun des instrument extérieur									
Rotary encoder									
Présentation oral									

Ferrer Benjamin	
Girard Victor	

Planning final :

SEANCES	1	2	3	4	5	6	7	8	9
Meteo en reseau									
Anémomètre									
Capt. Température/humidité									
Communication entre les bords									
Interface									
Boitier									
Pluviomètre									
Girouette									
Mise en commun des instrument ex.									
Rotary encoder									
Présentation oral									

Comme on peut le voir très clairement, notre planning initial a été quelque peu chamboulé; nous avons dû, pour le boitier intérieur essentiellement, faire face à de nombreux problèmes notamment pour l'interface et la communication entre nos différents modules. Nous reviendrons sur ceux-ci plus en détail par la suite.

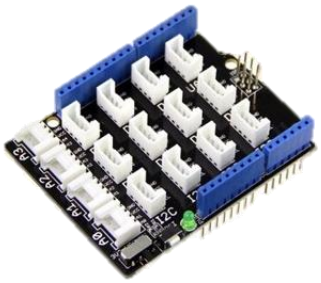
II. Matériel utilisé

II.1 Boitier extérieur

- Anémomètre : Fonctionne grâce à la fonction attachInterrupt(digitalPinToInterrupt,ISR, mode), celle-ci renvoi combien de fois par seconde un pin passe de l'état LOW à HIGH, grâce au données du constructeur nous savons qu'un certain nombre de passages 1/0 tous les certains temps correspondent à une vitesse de X mètres par seconde.

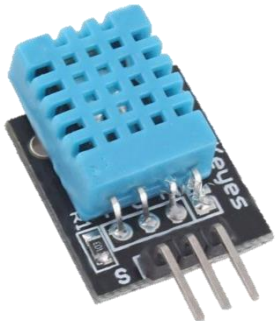
- Girouette : Renvoie huit valeurs analogiques bien précises, chacune de ces valeurs analogiques encadre les 8 points cardinaux (N, NE, E, …), celle-ci renvoie toujours les mêmes valeurs en fonction de la position par rapport a l'axe dans laquelle nous l'avons initialisé, dans notre cas le mat de support doit être plein ouest pour que les positions affichées à l'écran soient justes.

- Platine Grove : Cette petite platine Shield est destinée à venir s'enficher sur une plateforme Arduino. Elle dispose de connecteurs vous permettant de lui raccorder des modules d'extension au format "Grove". Dans notre projet elle nous permet de récuperer la data envoyer par la girouette et l'anémomètre.

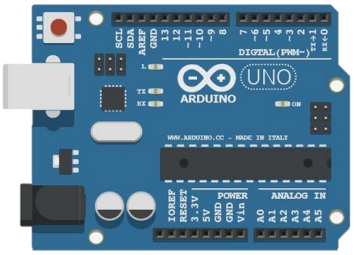


- Module RJ11 : Permet de faire le lien entre les modules Anémomètre et girouette, qui envoient les données à la carte Arduino grâce à un câble RJ11 (câble téléphonique), et la Platine Grove (ci-dessus).

- DHT11 : Capteur humidité & température, permet, via 1 seul pin de data, de renvoyer soit la température, soit l'humidité. Le choix de l'une ou l'autre dépend uniquement du code, qui crée des ''events '' de requêtes.

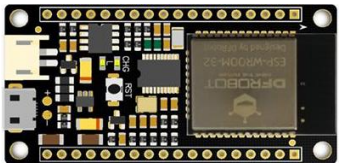


- Carte Arduino : Permet de récuperer les informations de mesure de tous les instruments dont nous disposons, et de les envoyées sous une forme bien précise vers la carte Arduino du boitier intérieur, via un câble RJ45 (câble Ethernet).

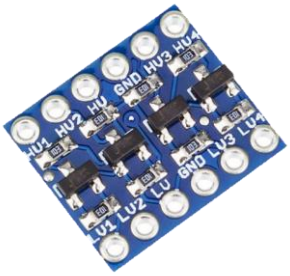


II.2 Boitier intérieur

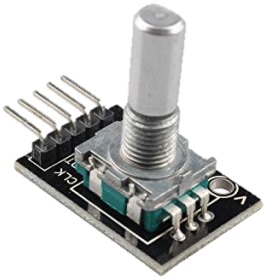
- ESP : carte Arduino compatible, équivalent d'une carte UNO, l'intérêt c'est qu'elle possède beaucoup plus de fonctionnalités telles que le Bluetooth et le wifi notamment. Dans le projet nous n'avons utilisé que la partie Wifi de celle-ci.



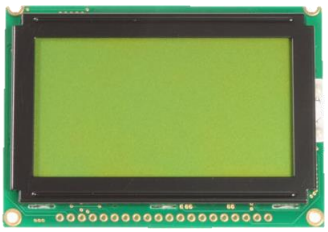
- Level shifter : Petit composant permettant de convertir du 5V en 3.3V et inversement. Nous l'utilisons pour la communication ESP Arduino permettant principalement de passer du TX Arduino qui envoie des pics de 5V vers l'ESP qui ne supporte que des pics de 3.3V max, et SANS perte d'information.



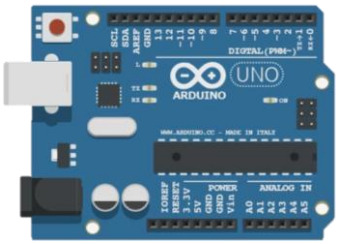
- Encodeur rotatif : il possède 3 bornes, 1 commune liée à 2 autres bornes (A et B) pour déterminer le sens de rotation. Lorsqu'on tourne l'encodeur rotatif, la borne "A" et la borne "B" vont toutes les deux passer de 0V à 5V et inversement. Mais il existe un décalage. Si on tourne dans un sens, c'est d'abord la borne "B" qui passe de 0V à 5V, si on tourne dans l'autre sens, c'est la borne "A" qui passe d'abord à 0V. Le système électronique interprète cela comme tourner vers la gauche ou tourner vers la droite. Il possède aussi un bouton poussoir, activé lorsqu'on clic sur la tige de l'encodeur.



- Ecran GLCD 128x64 : Il s'agit d'un écran LCD graphique, permettant d'afficher à peu près n'importe quoi (monochrome). Nous l'utilisons en SPI ce qui nous permet de limiter le nombre de fils connecté sur l'Arduino (seulement 3 pour l'affichage + 2 pour l'alimentation.

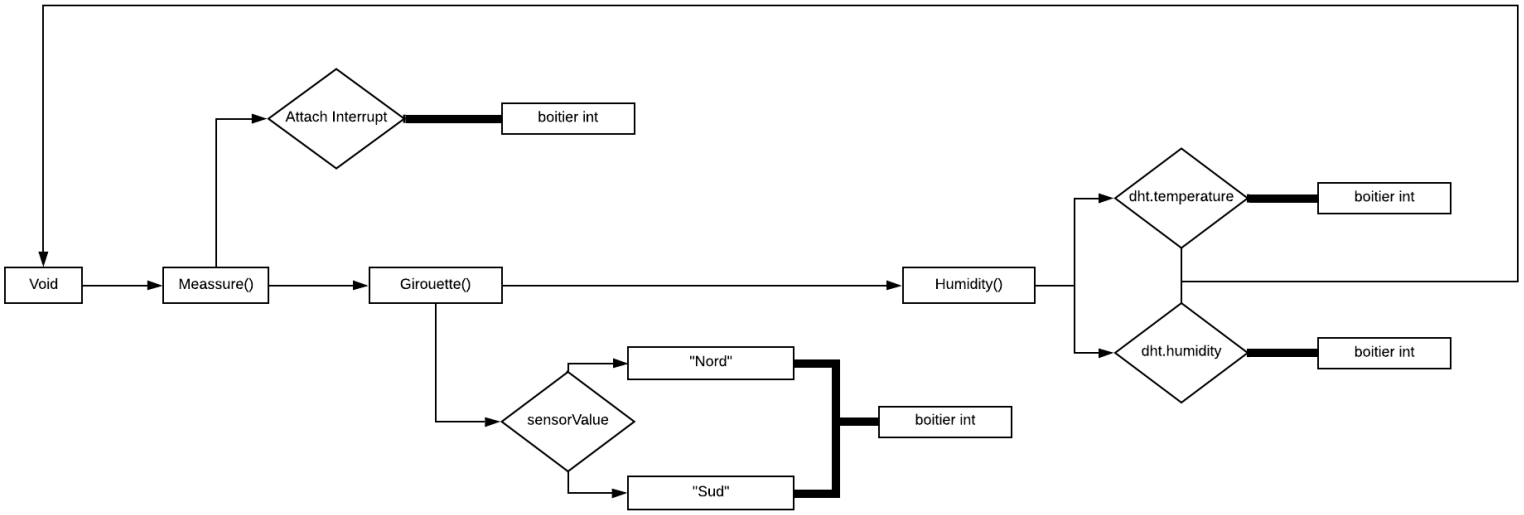


- Carte Arduino : celle-ci s'occupe essentiellement de l'affichage, elle reçoit les infos (sous forme de chaines de caractères bien spécifiques), les traites et les affiches sur l'écran. C'est elle qui gère aussi la disposition de l'arborescence du menu ainsi que l'interaction de l'utilisateur avec les contrôles (encodeur).



III. Fonctionnement

III.1 Mesures locales



L'algorithme du boitier se décompose de la façon suivante :

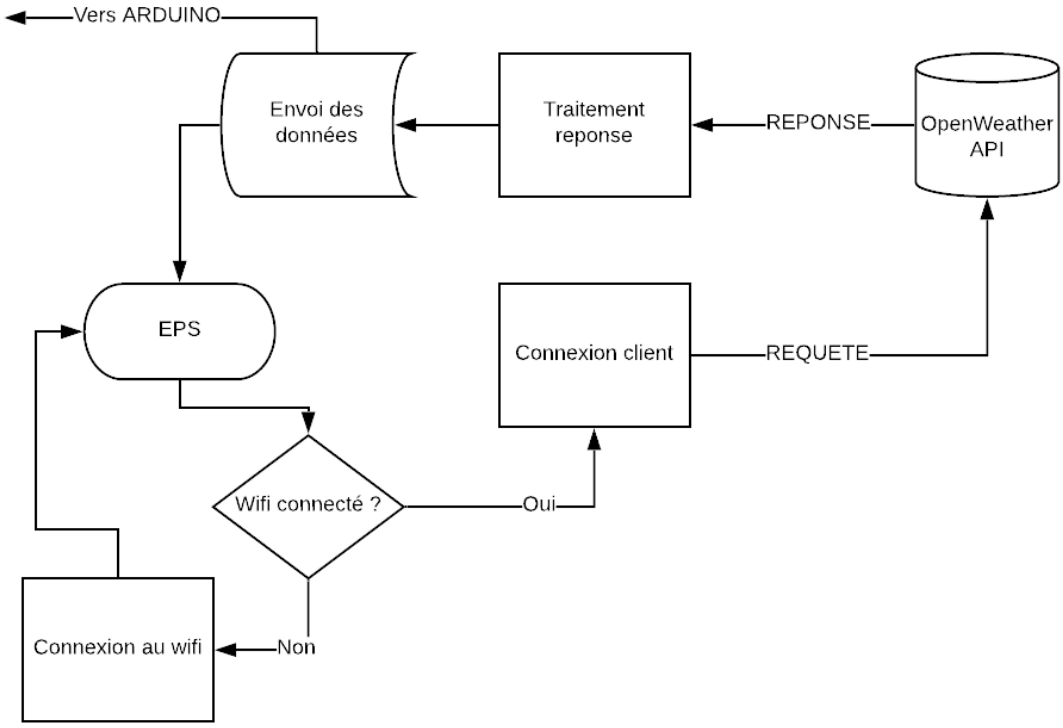
Dans le void loop() le programme commence par l'appel de la fonction Meassure() qui est la fonction permettant de calculer de la vitesse du vent grâce à l'anémomètre. La fonction attachInterrupt va compter le nombre de fois ou le pin va passer de l'état LOW à l'état HIGH et va renvoyer la vitesse angulaire de l'anémomètre et ensuite à l'aide d'une formule mathématique on passe d'une vitesse angulaire à une vitesse classique et on envoie le résultat au boitier extérieur.

Ensuite le programme passe à la fonction "Girouette" qui a la fonction faisant fonctionner la girouette. La girouette va se positionner vers un des 8 points cardinaux (Nord, Nord-Est, …, Nord-Ouest) et va envoyer un signal. A l'aide d'un switch case, on envoie en premier caractère quelles types de données c'est. À l'aide SoftwareSerial on envoie tout au boitier extérieur.

Enfin le programme se termine par la fonction Humidity() qui appelle d'abord dht.temperature elle va traiter ce que le capteur reçoit et l'envoie au boitier extérieur
Ensuite la fonction Humidity() appelle ensuite le dht.humidity elle va traiter ce que le capteur reçoit et l'envoie au boitier extérieur.

Enfin après tout cela le programme se refait en boucle pour actualiser les données et avoir les données les plus proche de la réalité.

III.2 Meteo en ligne

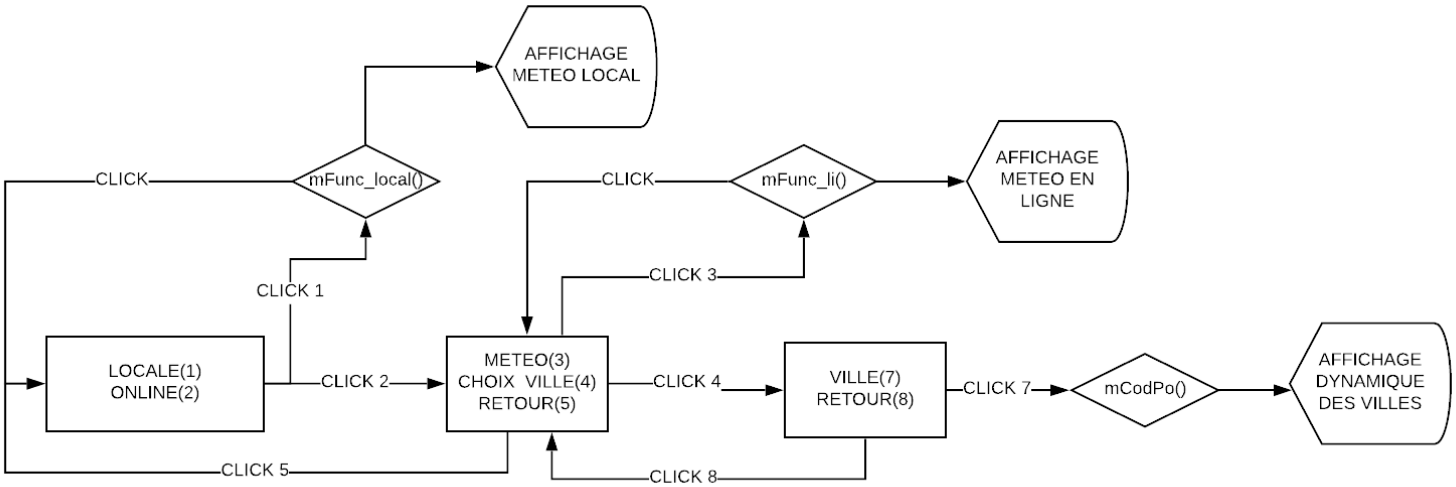


L'ESP commence par vérifier s'il est bien connecté au wifi, si ce n'est pas le cas il redémarre et retente de ce connecté à un réseau wifi renseigné. S'il est connecté il commence par se connecter à un client web via l'adresse IP du serveur, dans notre cas nous voulons joindre le serveur API d'OpenWeatherMap. Une fois connecté On envoie une requête sous forme d'URL vers le serveur. Celui-ci nous renvoi du texte en JSON, seulement si la requête est bien formée, c'est-à-dire qu'on a bien dans l'URL, le code postal d'une ville et le pays de celle-ci, ainsi que la bonne clé d'api qui fonctionne comme un mot de passe pour nous authentifier sur le serveur.

Une fois qu'on a récupéré une repose, il faut découper le gros bloc de texte en plusieurs petits blocs ; on récupère donc les blocs correspondant à la ville, au vent, à la température et à l'humidité, et on stocke chacune de ces valeurs dans des variables.

Pour l'envoie vers la carte Arduino du boitier intérieur on recrée de nouvelles chaines de caractère, avec un caractère d'identification (ex : '$' pour la température), puis la chaine de caractère de la valeur de l'info qu'on veut envoyer (avec String(valeur)), puis on fini par un caractère de fin (ici le '#'). Nous avons décidé de prendre des caractères spéciaux et non des lettres pour être sûr d'éviter toute erreur. On réitère l'opération pour chacune des infos, avec des caractères d'identification diffèrent mais le même caractère final, et on envoie tout vers l'Arduino.

III.3 Menus et affichage



Pour les menus nous utilisons une librairie qui nous permet de créer des menus en arbre et d'avoir plusieurs éléments sur chaque page du menu avec lesquels il est possible d'interagir. Lorsqu'on allume la station on arrive sur le menu d'accueil, celui-ci comprend deux éléments cliquables : soit la meteo locale soit la meteo en ligne.

Si on choisit la meteo locale, cela nous emmène vers un nouvel affichage et lance la fonction qui s'occupe de recuperer les infos des mesures externes, et de les afficher sur le nouvel écran, la fonction actualise l'affichage dès qu'il reçoit de nouvelles infos (toutes les 10 ms). La fonction détecte aussi le moindre clic et si elle en détecte un elle renvoi l'utilisateur sur le menu précèdent.

Si on choisit la meteo en ligne on arrive sur un menu à 3 éléments :

- Accès meteo : celui-ci emmène vers un nouveau menu qui affiche la meteo en ligne, et lance en même temps la fonction qui gère la réception, le traitement et l'affichage des informations envoyées par l'ESP sur l'écran. Cette fonction détecte le moindre mouvement de l'encodeur, si on le tourne à gauche ou à droite, l'affichage défile vers le haut ou le bas, et si on clic sur le bouton on est renvoyé au menu précédent.

- CHANGER : celui-ci emmène vers un nouveau menu a deux options, si on clique sur la ligne « meteo de », la fonction de cette ligne bloque la navigation sur l'écran et enclenche le défilement des villes sur la même ligne, si on tourne à gauche ou à droite on navigue parmi la liste de ville disponibles pour la meteo. Au second clic on valide la ville que l'on a d'afficher sur l'écran, la fonction réactive alors la navigation sur l'écran. En dessous de cette ligne on a aussi un élément appeler « Retour » qui permet de revenir à la page menue précédente.

- Retour : comme son nom l'indique, permet de retourner au menu précédent.

IV. Problèmes

IV.1 Lié à la girouette

Il s'est avéré que nous avions quelques problèmes avec la girouette car étant dépendante de son état initial de ce fait si on place la girouette dans un autre état que celui auxquelles on l'a configuré cela va chambouler les traitements de données et de ce fait renvoyer des valeurs fausses, comme par exemple, la girouette pointer vers le Nord et va indiquer le Sud car la girouette a été déréglée de 180 degrés à l'état initial.

La solution est simple, nous avons choisit de ne pas faire de pied pour le mat de la girouette afin qu'on le fixe dans une direction bien précise (plein ouest).

IV.2 Lié à la communication ESP/Arduino

Afin de faire communiqué en filaire nos deux cartes nous devions réussir à convertir le 5V envoyé par l'Arduino en 3.3V, pour ne pas griller l'ESP. Sur les conseilles de notre enseignant nous avons donc réalisé un pont diviseur de tension permettant de convertir le 5V en 3.3V grâce a 2 résistances de 1kΩ et 2kΩ cependant bien que cela ne semble fonctionner de prime abord, pour des raisons encore aujourd'hui floues cela ne fonctionnais pas, probablement de la perte de signal lorsque celui-ci passait au travers du pont. Nous sommes donc restés longtemps avec deux cartes dans l'incapacité de communiqué, fortement problématique puisqu'il s'agissait là du cœur du projet.

La communication entre ESP et Arduino étant peu courante dans le monde Arduino nous avons eu du mal à résoudre le problème. Mais après de longues recherches nous avons trouvé LA solution, grâce au digital level shifter (ci-dessus) nous sommes enfin parvenus à résoudre le problème. Celui-ci s'intercale au milieu des fils de communication et gère la conversion sans perte des informations de 5V a 3.3V et inversement.

V. Conclusion

Nous l'avons souligné notre projet est assez modulable et il est aisé de rajouter des modules comme une pluviomètre ou un capteur de pression sans pour autant passer des heures à bricoler/coder.

De plus nous avons la possibilité de travailler un peu plus l'esthétique des 2 boîtes en faisant aussi en sorte de choisir une boite qui soit à la fois étanche et surtout qui laisse passer les ondes Bluetooth pour ajouter une connexion entre les 2 bords et éviter qu'elle soit filaire.

Surtout Nous pourrions travailler l'aspect ergonomique qui n'est pas un aspect à négliger. Pour le côté ergonomique, nous avons la possibilité de changer d'écran pour un écran couleur et ainsi pouvoir dessiner des petits dessins comme des nuages ou un soleil lorsqu'il fait beau ou non ou alors la possibilité de se déplacer entre les différents menus de façon tactile et de simplifier les menus pour les rendre plus attractives.

Remerciements :

Nos premiers remerciements sont envoyés à nos encadrants M.Pascal MASSON pour l'apport des connaissances de base en Arduino et l'apport des matériaux essentielles à notre projet. Et aussi à M. Nassim ABDERRAHMANE pour l'aide apportée au début de notre projet et le soutien durant notre projet. Et enfin à Ines GMATI pour son aide sur les instruments extérieur.

Bibliographie

http://www.zonnepanelen.wouterlood.com/an-128x64-graphic-lcd-display-with-st7920-controller-for-the-arduino-displaying-temperature-and-relative-humidity/

https://openweathermap.org/

https://www.lextronic.fr/temperature-meteo/19999-capteur-anenometre.html

https://www.youtube.com/watch?v=Ru0aI_2J2dI&feature=youtu.be