Produit : Station météorologique

L'objectif de cette activité est d'ajouter à une station météorologique la possibilité de communiquer à distance les informations récoltées.

Table des matières

1. Découverte du produit et de la problématique technique

Le système d'étude est une station météo utilisée par un agriculteur maraîcher pour suivre les conditions météorologiques de ses cultures.

Dans une démarche personnelle de transition agroécologique, l'agriculteur souhaite réduire l'impact environnemental de son exploitation, en pratiquant un usage raisonné de l'eau et des traitements phytosanitaires. Pour cela, il a besoin d'une meilleure maîtrise des conditions environnementales locales, afin de prendre ses décisions sur la base de données objectives.

L'objectif est d'optimiser les arrosages et les traitements, en fonction :

- de la quantité de pluie reçue (mesurée par un pluviomètre), pour déclencher l'arrosage uniquement lorsque cela est nécessaire,

- et des conditions de vent (mesurées par un anémomètre et une girouette), afin d'éviter la dérive des produits lors des pulvérisations (engrais foliaires, insecticides naturels).

La station météo utilisée est connectée à un microcontrôleur intégré au système, qui permet la lecture locale des données météo via une liaison série sur écran LCD.

Cependant, la station est installée en bordure de champ, sur un poteau ou une serre, dans une zone non accessible sans déplacement. L'agriculteur souhaite pouvoir stocker et consulter ces données à distance, sans avoir à se rendre physiquement à la station.

Problématique technique

> Comment assurer la transmission sans fil des données mesurées par la station météo, afin qu'elles puissent être consultées et stockées à distance par l'agriculteur ?

Contrainte : la solution actuelle n'intègre aucun système de transmission sans fil. Les données actuellement affichées sur l'écran LCD n'ont pas une mise en forme facilitant leurs stockages.

a. Caractéristiques de la station météo

- capteurs intégrés :
  - Pluviomètre (interrupteur à bascule à chaque volume de pluie),
  - Anémomètre (générateur d'impulsions), Girouette (sortie tension continue selon la direction),
  - Microcontrôleur intégré au système : carte Arduino (entrée analogique et numérique disponibles),
- alimentation **:** par batterie 9 V elle-même alimentée par un panneau solaire (non étudiée),
- affichage local des données **:** par liaison série sur écran LCD,
- station installée en bordure de champ, à plusieurs dizaines de mètres des zones fréquentées,

b. Cahier des charges

- distance minimale de transmission **:** ≥ 15 mètres en champ libre.
- données transmises **:** Vitesse du vent, direction du vent, pluviométrie (valeurs issues des capteurs).
- trame transmise **:** Format ASCII. Doit inclure un identifiant de station pour distinguer la source des données. Les données seront séparées par un “;” (format CSV). Il ne doit pas y avoir d’espace. Ce format permet d’exploiter facilement les données météorologiques avec un tableur.
  Exemple : S1;12.4;225;3.2 (identifiant ; vitesse du vent en km/h ; direction du vent (en degré par rapport au Nord) ; pluviométrie en mm depuis minuit).
- fréquence d’émission **:** 1 trame toutes les 1 secondes pendant l’expérimentation, puis 1 trame toutes les 5 minutes en fonctionnement normal.
- coût du module ajouté **:** ≤ 25 €.
- simplicité de mise en œuvre (branchement + code) : doit permettre une intégration fonctionnelle au système actuel en utilisant une liaison série UART.

c. Schéma structurel existant

Sur le schéma fourni :
- la carte arduino UNO (sans shield grove)
- la direction du vent est récupérée sur l’entrée analogique A0
- la vitesse du vent est récupérée sur l’entrée digitale 3
- la quantité de pluie est récupérée sur l’entrée digitale 2

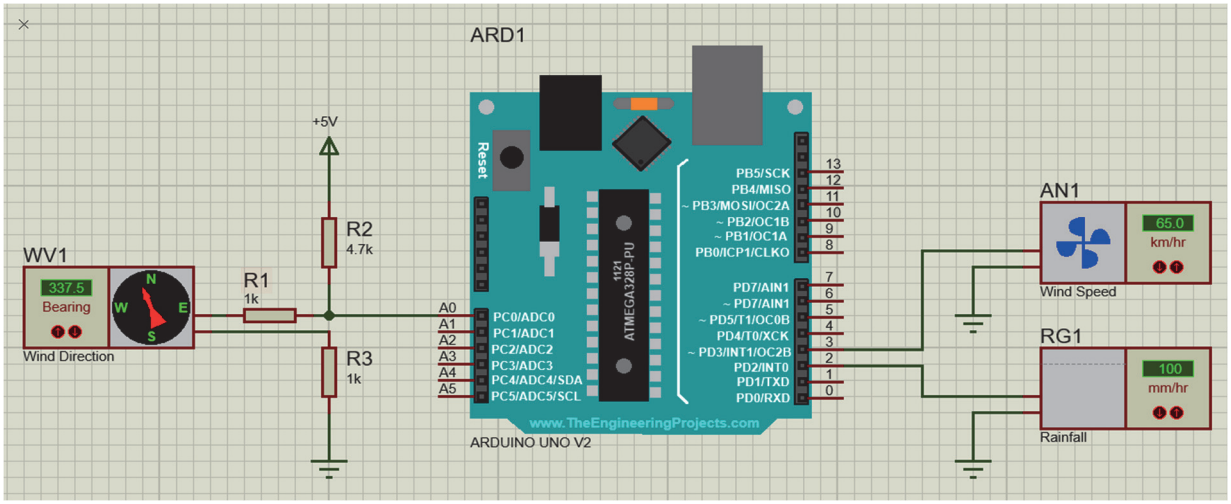


Figure 1: Schéma structurel du système actuel

L’écran d’affichage actuel utilisant la liaison série n’est pas câblé.

d. Extrait du programme existant

Dans un souci de simplification les éléments suivants ne seront pas traités dans le programme fourni ni dans les attendus de l'activité :

- La calibration de la station météo.
- La remise à zéro journalière de la quantité de pluie totale.

```
#include "SparkFun_Weather_Meter_Kit_Arduino_Library.h"
#include <SoftwareSerial.h>

// Définition des différentes broches de l'arduino utilisés
 int windDirectionPin = A0;
 int windSpeedPin = 3;
 int rainfallPin = 2;
 int Rx = 4;
 int Tx = 5;
// Création d'une instance du Kit météo
SFEWeatherMeterKit weatherMeterKit(windDirectionPin, windSpeedPin,
rainfallPin);
// Création d'une instance de la liaison série
SoftwareSerial liaison(Rx, Tx); // RX, TX

void setup()
{
    Serial.begin(115200); // Initialisation de la liaison série USB
    weatherMeterKit.begin();  // Démarrage du kit de mesure de la météo
(Sparkfun)
    liaison.begin(9600);  // Initialisation de la liaison série UART
}

void loop()
{
    float Vitesse_vent=weatherMeterKit.getWindSpeed();
    float Direction_vent=weatherMeterKit.getWindDirection();
    float Quantite_pluie=weatherMeterKit.getTotalRainfall();
    // Début : Partie à MODIFIER ----------------------------------------
    liaison.print(F("Vitesse du vent: "));
    liaison.print(Vitesse_vent,1);
    liaison.println(F(" km/h"));
    liaison.print(F("Direction du vent: "));
    liaison.print(Direction_vent,1);
    liaison.println(F(" degre par rapport au Nord"));
    liaison.print(F("Quantité de pluie depuis 24h : "));
    liaison.print(Quantite_pluie,1);
    liaison.println(F(" mm"));
    //fonctions à rajouter :
```

```
            //Convertir les trois variables flottantes en chaînes de caractères
            //Construction de la trame ASCII
            //Transmission de la trame
    delay(300000); // Rafraichissement de l'affichage toutes les 5 minutes
    // Fin : Partie à MODIFIER ----------------------------------------
}
```

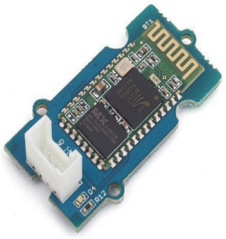
Le programme fonctionne actuellement pour afficher les informations sur un écran LCD connecté sur une liaison série.

2. Conception

a. Les modules de communication à disposition

- Module Bluetooth série HC-05 (format Grove)

Description : Module de communication sans fil Bluetooth classique (S2P), utilisé pour des échanges simples de données série entre microcontrôleurs et périphériques (ordinateur, smartphone…). Fonctionne en mode esclave ou maître configuré. Interface UART standard, compatible avec la plupart des microcontrôleurs.



Caractéristiques :

- Interface UART (TX/RX 3,3 V niveau logique) – compatible 5 V via Grove.
- Débit : 9600 bauds par défaut (modulable jusqu'à 1382400 bauds).
- Distance maximale : environ 10 m (champ libre).
- Tension d'alimentation : 3,6 à 6 V (via interface Grove).
- Consommation : ~30 mA en transmission.
- LED intégrée indiquant l'état de la liaison.
- Dimensions : 42 × 16 mm environ.
- Prix : 21,5€

Module XBee (format Grove)

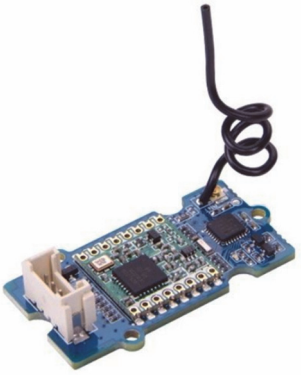
Description : Module de communication radio point à point ou multipoint (réseau mesh possible) fonctionnant sur la bande 2,4 GHz. Utilisé pour des communications transparentes UART ou des applications ZigBee simples. Facilement utilisable en remplaçant un câble série.

Caractéristiques :

- Interface UART (TX/RX 3,3 V niveau logique) – compatible 5 V via Grove.
- Protocole : XBee série 1 (ZigBee, API ou mode transparent).
- Distance maximale : jusqu'à 30 m en intérieur / 100 m en extérieur.
- Tension d'alimentation : 3,0 à 3,6 V (convertie via carte Grove).
- Consommation : 50 mA en émission, 10 µA en veille.

- Antennes intégrée ou externe selon modèle.
- Dimensions : 27 × 24 mm (hors breakout Grove).
- Prix : Environ 40€ + 20€ pour le module grove

Module LoRa UART 433 MHz (format Grove)



Description : Module de communication longue portée utilisant la technologie LoRa (modulation Spread Spectrum) avec interface UART. Permet des communications point à point sur de très longues distances avec faible consommation.

Caractéristiques :

- Interface UART (TX/RX, 3,3 V niveau logique).
- Fréquence : 433 MHz.
- Portée : jusqu'à 1 km en champ libre (antennes visibles).
- Débit : 1200 à 9600 bauds (configurable).
- Tension d'alimentation : 3,3 à 5 V (via Grove).
- Consommation : 20 à 40 mA en émission, <10 mA en veille.
- Trame transmise de manière transparente (comme un câble série).
- Dimensions : 35 × 20 mm environ.
- Prix : 22,5€

b. Fonctions Arduino utiles

`dtostrf()`

Fonction permettant de convertir une valeur réelle **(**float**)** en une chaîne de caractères **(**char[]**)**, pour l'insérer ensuite dans une trame textuelle ASCII.

Syntaxe :

```
dtostrf(valeur, largeur_totale, nb_decimales, buffer);
```

Paramètres :

- valeur : le nombre réel (type float) à convertir
- largeur_totale : nombre total de caractères dans la chaîne finale (espaces inclus)
- nb_decimales : nombre de chiffres après la virgule
- buffer : tableau char[] qui contiendra le texte généré

Exemple d'utilisation :

```
char texte[6];
float mesure = 23.72;
dtostrf(mesure, 0, 1, texte);  // → texte = "23.7"
```

💡 Astuce : utiliser 0 en largeur pour ne pas ajouter d'espaces inutiles.

`sprintf()`

Fonction permettant de construire une chaîne de caractères complète à partir de plusieurs variables, avec un format personnalisé. C'est comme écrire une ligne formatée dans un tableau.

Syntaxe :

```
sprintf(buffer, "format", val1, val2, ...);
```

Paramètres :

- buffer : tableau char[] où sera stockée la chaîne finale
- "format" : texte avec des marqueurs (%s, %d, etc.), le texte restera tel quel et les marqueurs seront remplacés par les valeurs indiquées après
- val1, val2, … : les variables à insérer dans le format

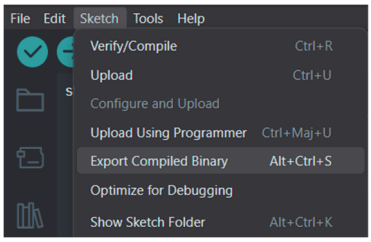
Principaux formats :

Code	Signification
%s	Chaîne de caractères (char[])
%d	Entier (int)
%f	Float (⚠ non supporté sur Arduino Uno → utiliser dtostrf() à la place pour convertir en chaîne de caractères %s)

Exemple d'utilisation :

```
char trame[32];
char tempStr[6] = "23.7";
char humidStr[6] = "45.2";
sprintf(trame, "S1;%s;%s", tempStr, humidStr);
                                    //trame= "S1;23.7;45.2"
```

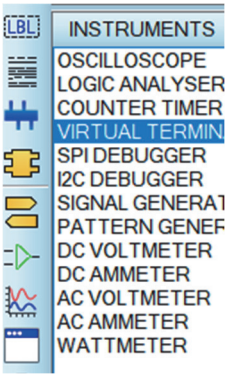
c. Compilation et export des binaires



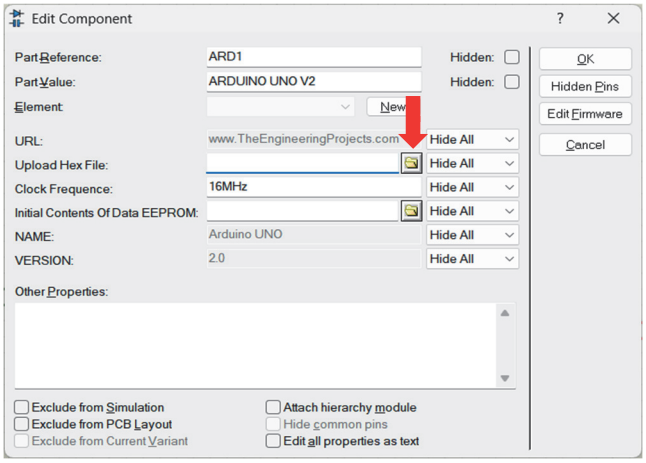
Les binaires se trouveront dans le dossier « build » qui aura été créé automatiquement là où se trouve le programme Arduino.

d. Simulation

Sous ISIS, Le Terminal virtuel est présent dans ”Virtual Instruments Mode”.

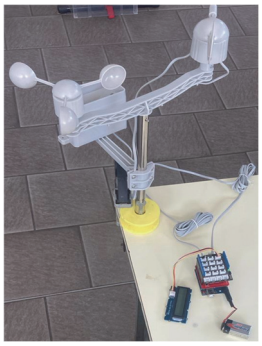
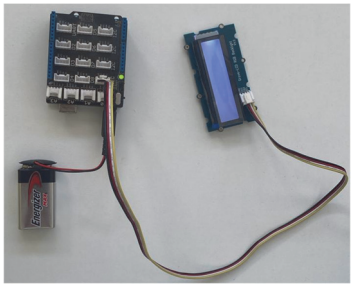


L'intégration du code source de la carte de développement Arduino sous ISIS se fait en double-cliquant sur la carte puis en cliquant sur l'icône de sélection de fichier. Il ne reste ensuite qu'à sélectionner le fichier HEX généré sous l'IDE d'Arduino.



3. Expérimentation

Pour effectuer l'expérimentation, le matériel suivant est à disposition :

<table>
<tr><td>Une alimentation à tension variable
  </td><td>Un ventilateur
  </td></tr>
<tr><td>Anémomètre étalonné
  </td><td>Un télémètre laser
  </td></tr>
<tr><td>Partie émetteur Station météo, un arduino Uno équipé du “SparkFun Weather Shield” et d’un “Grove Base Shield”, un écran LCD
  </td><td>Partie récepteur - Aruduino Uno équipé d’un “Grove Base Shield” et écran LCD
  </td></tr>
<tr><td colspan="2">Modules de communication à distance : Les 3 modules décrits dans la partie 2.a</td></tr>
</table>