

moisture sensor B-L072Z-LRWAN1

cmonaton

July 2019

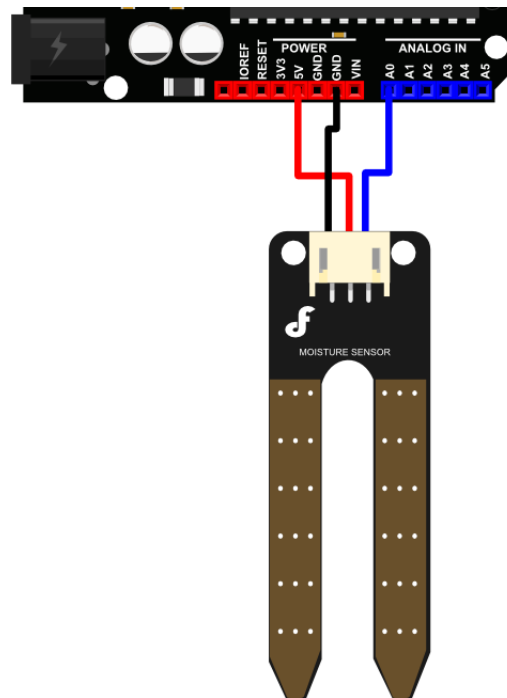
1 Introduction

But de ce tuto : connecter le capteur moisture sensor V2 de DFROBOT à la carte B-L072Z-LRWAN1 et envoyer sa donnée sur lorasever.

Prérequis : Télécharger le code sur la carte et télécharger l'application End_Node cf tuto *B-L072Z-LRWAN1*

2 Connecter le capteur à la carte

Connecter le capteur selon ce lien : https://wiki.dfrobot.com/Moisture_Sensor__SKU_SEN0114_



La carte B-L072Z-LRWAN1 possède les mêmes connecteurs qu'une carte arduino.

3 Lire la valeur du capteur par liaison série

3.0.1 Pour ouvrir les ports ttyACM0 et ttyACM1

Solution temporaire

```
sudo chmod 666 /dev/ttyACM0
```

Il faut le entrez cette commande souvent.

Solution permanente

Créer un fichier dans son home

```
50-myusb.rules
```

l'éditer :

```
KERNEL=="ttyACM[0-9]*",MODE="0666"
```

Puis copiez ce fichier dans /etc/udev/rules.d/ et redémarrer votre PC.

```
sudo cp 50-myusb.rules /etc/udev/rules.d
```

C'est suffisant pour ne plus avoir à réouvrir les ports manuellement. Cependant, n'importe quel dispositif usb connecté au PC a maintenant le droit d'écriture sur le PC.

Pour plus de sécurité ajouter ces lignes dans ce fichier :

```
ACTION=="add", KERNEL=="ttyACM[0-9]*", ATTRS{idVendor}=="xxxx",  
ATTRS{idProduct}=="yyyy", MODE="0666"
```

Pour déterminer idVendor et idProduct des cartes tapez lsusb avant et après avoir connecter la carte.

Dans mon cas avant et après avoir branché une carte externe :

Dans mon cas avant et après avoir branché une carte externe :

```

clement@clement-Latitude-5490:~$ lsusb
Bus 002 Device 002: ID 2109:0812 VIA Labs, Inc. VL812 Hub
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 1bcf:2b96 Sunplus Innovation Technology Inc.
Bus 001 Device 008: ID 045e:077b Microsoft Corp.
Bus 001 Device 010: ID 04d8:ef98 Microchip Technology, Inc.
Bus 001 Device 006: ID 413c:2105 Dell Computer Corp. Model L100 Keyboard
Bus 001 Device 002: ID 2109:2812 VIA Labs, Inc. VL812 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
clement@clement-Latitude-5490:~$ lsusb
Bus 002 Device 002: ID 2109:0812 VIA Labs, Inc. VL812 Hub
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 1bcf:2b96 Sunplus Innovation Technology Inc.
Bus 001 Device 008: ID 045e:077b Microsoft Corp.
Bus 001 Device 006: ID 413c:2105 Dell Computer Corp. Model L100 Keyboard
Bus 001 Device 002: ID 2109:2812 VIA Labs, Inc. VL812 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

```

```

idProduct = ef98
idVendor= 04d8

```

Pour ajouter d'autres appareils, copier coller ces lignes en changeant idProduct et idVendor.

Pour afficher en liaison série : fonction PRINT("");
Dans le main ajoutez

```
PRINTF("%o\n",HW_AdcReadChannel(0));
```

3.0.2 Code

Dans le projet End_Node copiez le main suivant :

```

int main( void )
{
    /* STM32 HAL library initialization*/
    HAL_Init();

    /* Configure the system clock*/
    SystemClock_Config();

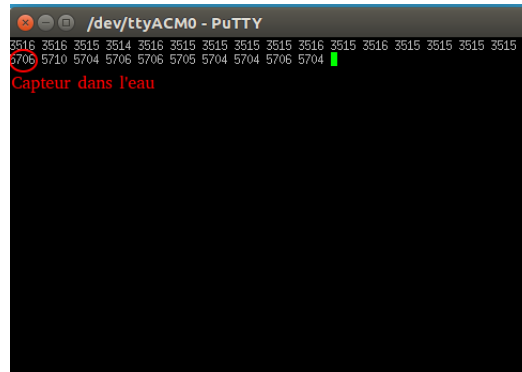
    /* Configure the debug mode*/
    DBG_Init();

    /* Configure the hardware*/
    HW_Init();

    /* USER CODE BEGIN 1 */
    PRINTF("%o ",HW_AdcReadChannel(0));
    /* USER CODE END 1 */
}

```

Le capteur est branché sur la broche 0 de l'ADC donc on choisit canal 0.
Putty, speed : 115200



Appuyer sur Reset pour obtenir la valeur.

4 Envoyer la valeur du capteur sur LoRa server

Remplir le tableau AppData avec la fonction HW_AdcReadChannel(0) :
dans le fichier main.c

```
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = HW_AdcReadChannel(0);  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;  
AppData.Buff[i++] = 5;
```

Et prenez ce main :

```
int main( void )  
{  
    /* STM32 HAL library initialization*/
```

```

HAL_Init();

/* Configure the system clock*/
SystemClock_Config();

/* Configure the debug mode*/
DBG_Init();

/* Configure the hardware*/
HW_Init();

/* USER CODE BEGIN 1 */

/* USER CODE END 1 */

/*Disable Stand-by mode*/
LPM_SetOffMode(LPM_APPLI_Id , LPM_Disable );

PRINTF("VERSION: %X\n\r", VERSION);

/* Configure the Lora Stack*/
LORA_Init( &LoRaMainCallbacks, &LoRaParamInit);

LORA_Join();

LoraStartTx( TX_ON_TIMER) ;

while( 1 )
{
    if (AppProcessRequest==LORA_SET)
    {
        /*reset notification flag*/
        AppProcessRequest=LORA_RESET;
        /*Send*/
        Send( NULL );
    }
    if (LoraMacProcessRequest==LORA_SET)
    {
        /*reset notification flag*/
        LoraMacProcessRequest=LORA_RESET;
        LoRaMacProcess( );
    }
    /*If a flag is set at this point, mcu must not enter low power and must loop*/
    DISABLE_IRQ( );
    /* if an interrupt has occurred after DISABLE_IRQ, it is kept pending
       * and cortex will not enter low power anyway */
}

```

```

        if ((LoraMacProcessRequest!=LORA_SET) && (AppProcessRequest!=LORA_SET))
        {
            #ifndef LOW_POWER_DISABLE
                LPM_EnterLowPower( );
            #endif
        }

        ENABLE_IRQ();

        /* USER CODE BEGIN 2 */
        /* USER CODE END 2 */
    }
}

```

4.1 interface loraserver

<https://lora.campusiot.imag.fr/#/login>

Une fois la carte connectée au server loraserver , on peut voir la payload ou données comme indiqué dans l'image

```

adr: true
applicationID: "139"
applicationName: "xcvxc"
data: "BQUFBQUFBQUABQUFBQUFBQ=="
devEUI: "3131353852378418"
deviceName: "dcc"
fCnt: 10
fPort: 2
▼ rxInfo: [] 1 item
  ▼ 0: {} 5 keys
    gatewayID: "0000024b08060211"
    loRaSNR: 3.5
    ▼ location: {} 3 keys
      altitude: 250
      latitude: 45.19387815352627
      longitude: 5.768014396831585
      name: "KER_109_IM2AG"
      rssi: -107
  ▼ txInfo: {} 2 keys
    dr: 5
    frequency: 868300000

```

Data ou payload

5 Décoder la payload

La payload est codée en base64, pour la décoder en décimal suivre les étapes suivantes :

Décodage base64 →ascii →decimal

Décodeur base64 →ascii : <https://www.opinionatedgeek.com/Codecs/Base64Decoder>
 Décodeur ascii →decimal : <https://www.branah.com/ascii-converter>
 Exemple :
 Capteur sec

```

adr: true
applicationID: "139"
applicationName: "xxxx"
data: "BQUFBQUFBQU1BQUFBQUFBQ=="
devEUI: "3131353852378418"
deviceName: "dcc"
fCnt: 126
fPort: 2
▼ rxInfo: [] 1 item
  ▼ 0: {} 5 keys
    gatewayID: "0000024b08060211"
    loRaSNR: -9
    ▼ location: {} 3 keys
      altitude: 250
      latitude: 45.19387815352627
      longitude: 5.768014396831585
      name: "KER_109_IM2AG"
      rssi: -118
  ▼ txInfo: {} 2 keys
    dr: 5
    frequency: 867100000
  
```

Payload en base64

Valeur en base64 : BQUFBQUFBQU1BQUFBQUFBQ==
 Valeur en décimal : 005005005005005005005005005005005005053005005005005005005
 Valeur capteur : 53

Capteur trempé dans l'eau

3:58:38 PM uplink

```

adr: true
applicationID: "139"
applicationName: "xcvxc"
data: "BQUFBQUFBQX/BQUFBQUFBQ=="
devEUI: "3131353852378418"
deviceName: "dcc"
fCnt: 4
fPort: 2
▼ rxInfo: [] 1 item
▼ 0: {} 5 keys
  gatewayID: "0000024b08060211"
  loRaSNR: -2.5
  ▼ location: {} 3 keys
    altitude: 250
    latitude: 45.19387815352627
    longitude: 5.768014396831585
    name: "KER_109_IM2AG"
    rssi: -117
  ▼ txInfo: {} 2 keys
    dr: 5
    frequency: 867900000

```

Payload base64

Valeur en base64 : BQUFBQUFBQX/BQUFBQUFBQ==
 Valeur en décimal : 005005005005005005005005255005005005005005005
 Valeur de capteur : 255

D'après les caractéristiques du capteur : https://wiki.dfrobot.com/Moisture_Sensor__SKU_SEN0114_

```

# the sensor value description \\
# 0 ~300      dry soil \\
# 300~700     humid soil  \\
# 700~950     in water  \\

```

Les valeurs ne sont pas les mêmes mais la relation de proportionnalité reste vérifiée. Valeur capteur mouillé $\approx 4 * \text{valeurcapteursec}$