



UNIVERSITÉ
TOULOUSE III
PAUL SABATIER



Master 1 EEA - SME

UE Réalisations systèmes et microsystèmes

Rapport TP de base

Réalisé par:

CHALLALI Ramzi
MESSAOUDI Sara

Année universitaire : 2023-2024

Table des matières

I. Introduction.....	3
II. Manipulation.....	3
1. Matériel utilisé.....	3
2. Projet N°1: STM32 + DHT22 + LCD.....	5
2.1. Matériel utilisé :.....	5
2.2. Schéma de câblage:.....	5
2.3. Configuration et initialisation des pins:.....	6
2.4. Fonctionnement du capteur DHT22:.....	6
2.5. Le code :.....	8
2.6. Visualisation de la trame:.....	8
2.7. Visualisation du résultat:.....	9
3. Projet N°2: STM32 + SHT31 + LCD.....	9
2.1. Matériel utilisé :.....	9
2.2. Schéma de câblage:.....	10
2.3. Configuration et initialisation des pins.....	10
2.4. Spécifications du SHT31:.....	11
2.5. Fonctionnement du capteur SHT31:.....	12
2.6. Le code :.....	12
2.7. Visualisation de la trame:.....	13
III. Conclusion:.....	14

I. Introduction

L'objectif de ces TP est de se familiariser avec la famille de cartes STM32, ainsi qu'avec le processeur ARM, en utilisant des capteurs pour répondre à des spécifications précises et acquérir une meilleure compréhension des outils de développement associés à la carte NUCLEO STM32 fournie.

Cette expérience nous permettra d'acquérir une expérience pratique avec des logiciels et du matériel couramment utilisés dans l'industrie. Le projet sera divisé en deux parties : la première partie portera sur le fonctionnement et la mise en œuvre du capteur de température et d'humidité SHT31, ainsi que sur son affichage sur un écran LCD. La seconde partie se concentrera sur la lecture des données du capteur DHT22 et leur affichage sur le LCD. L'objectif principal est de comparer les protocoles de communication utilisés pour chaque capteur.

II. Manipulation

1. Matériel utilisé

- Carte Nucleo STM32- L476RG:

Cette carte est dotée d'une capacité de stockage interne permettant à l'utilisateur d'y installer un programme informatique. Ce programme consiste en une série d'instructions destinées à être exécutées par le microcontrôleur qui équipe la carte, un STM32. Elle est idéale pour les applications embarquées en raison de sa faible consommation d'énergie et de ses multiples fonctionnalités. En plus de sa mémoire interne, la carte Nucleo STM32L476RG est pourvue d'une gamme de périphériques intégrés incluant des interfaces de communication et des convertisseurs analogiques-numériques. Ces caractéristiques offrent aux développeurs la possibilité de concevoir des applications complexes et performantes.

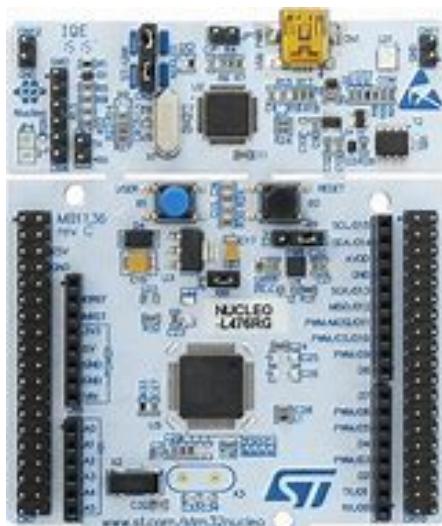


Figure 1: Carte STM32L476RG

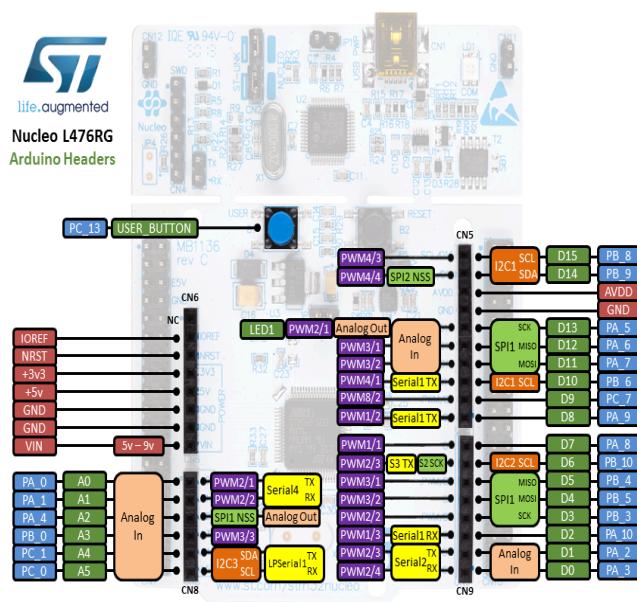


Figure 2: Pinout de la carte

- **Module Grove Base Shield:**

Le module Grove Base Shield simplifie la connexion de modules électroniques aux cartes de développement. Il utilise des connecteurs standardisés pour les modules Grove, permettant ainsi un prototypage rapide sans câblage compliqué.



Figure 3: Module Grove Base Shield

- **Capteur SHT31:**

Le SHT31 est un capteur de température et d'humidité de haute précision fabriqué par Sensirion. Il offre une fiabilité exceptionnelle avec une précision de $\pm 2\%$ pour l'humidité relative et de $\pm 0,3^{\circ}\text{C}$ pour la température, ce qui le rend adapté à de nombreuses applications. Ce capteur est doté d'une interface I2C, offrant une lecture simplifiée et une intégration aisée dans les systèmes électroniques. Il est compatible avec les tensions de 3V ou 5V, ce qui lui permet d'être alimenté et utilisé avec une variété de microcontrôleurs et de micro-ordinateurs.

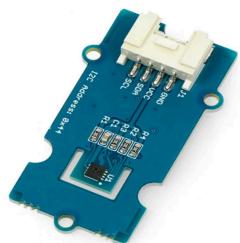


Figure 4: Capteur SHT31

- **Capteur DHT22:**

Également appelé AM2302, le capteur DHT22 est un dispositif commun utilisé pour mesurer la température et l'humidité relative. Il utilise une connexion à un microcontrôleur, souvent via une interface OneWire, pour communiquer les données. Bien qu'il soit largement utilisé en raison de sa disponibilité et de son prix abordable, sa précision peut être limitée dans des conditions extrêmes.

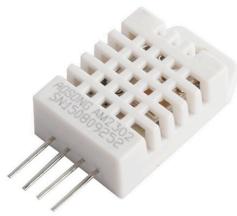


Figure 5: Capteur AM2302

2. Projet N°1: STM32 + DHT22 + LCD

2.1. Matériel utilisé :

Carte Nucleo STM32L476RG

Capteur DHT22

Ecran LCD JHD1802M1

Shield Base

Un Picoscope

2.2. Schéma de câblage:

À l'aide du logiciel Fritzing, nous avons élaboré notre schéma de câblage suivant :

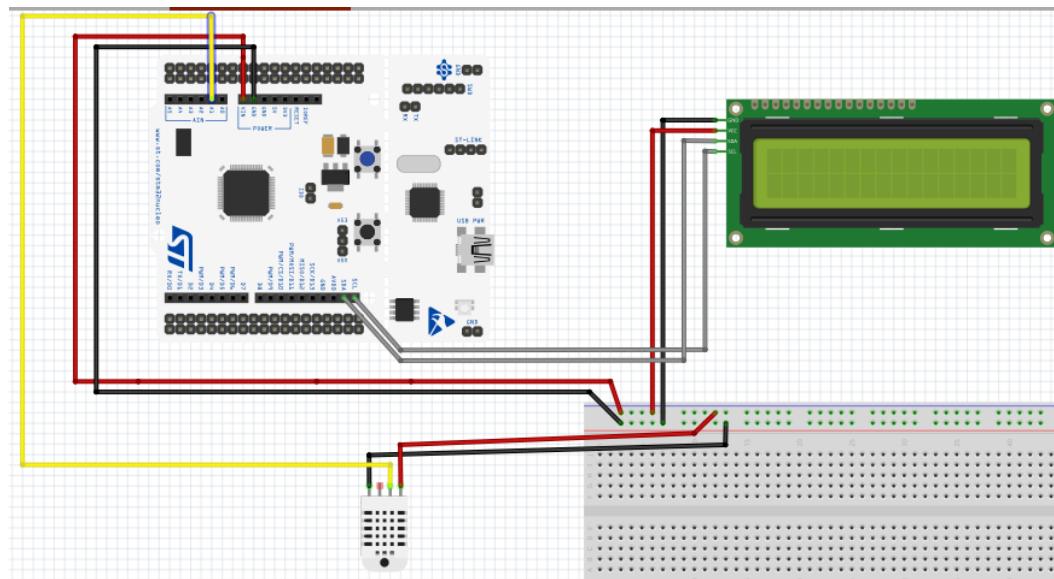


Figure 6: Schéma de câblage du DHT22

2.3. Configuration et initialisation des pins:

Nous avons utilisé le logiciel STM32CubeMX pour configurer les broches nécessaires à notre projet. Nous avons connecté les broches PB8 et PB9 relativement au SCL et SDA du LCD à pour configurer l'I2C, puis nous avons défini la broche PA1 en tant que GPIO_ANALOG où nous avons connecté le pin de données du DHT22.

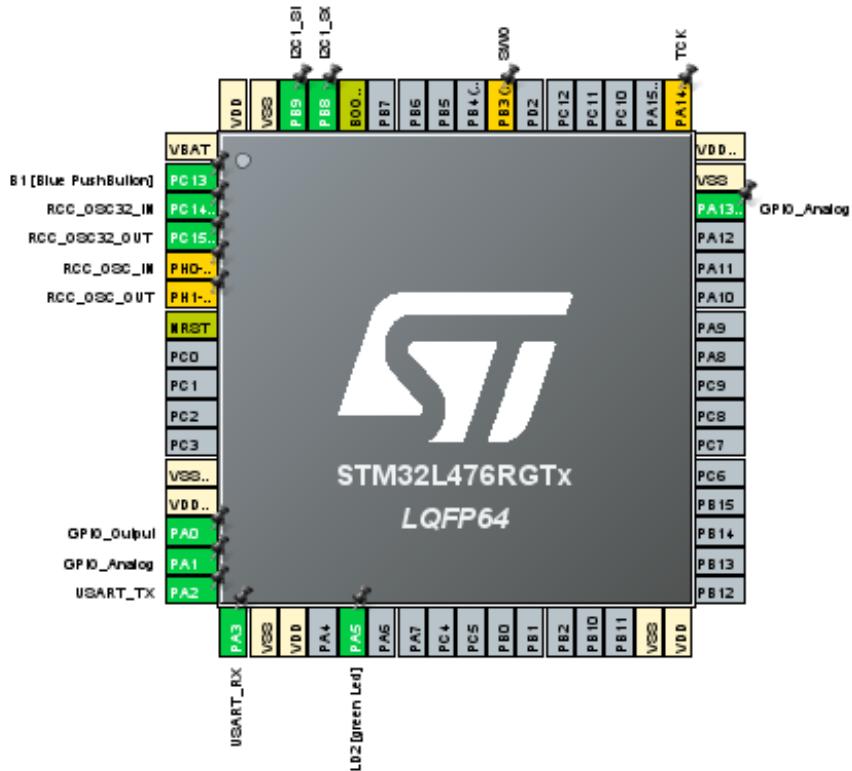


Figure 7: Configuration du STM32

2.4. Fonctionnement du capteur DHT22:

1. Initialisation :

- Le microcontrôleur envoie une impulsion de démarrage au capteur en tirant la ligne de données vers le bas pendant une durée minimale spécifiée (généralement 1 à 10 millisecondes).
- Après cette impulsion de démarrage, le microcontrôleur relâche la ligne de données, permettant au capteur de prendre le contrôle de la ligne.

2. Réponse d'acquittement :

- Le capteur DHT22 répond à l'impulsion de démarrage en tirant la ligne de données vers le bas pendant une durée spécifique (environ 20 à 40 microsecondes), puis la libère pour permettre au microcontrôleur de prendre le contrôle.

3. Transmission de données :

- Le capteur envoie les données en commençant par le bit de départ, suivi des données de température et d'humidité, bit par bit.

- Chaque bit de données est représenté par la durée pendant laquelle la ligne de données reste à l'état bas. Une courte durée représente un bit "0", tandis qu'une durée plus longue représente un bit "1".
- Les données sont envoyées dans un format spécifique, généralement 16 bits pour la température, 16 bits pour l'humidité et 8 bits pour le checksum.

4. Checksum :

- Après avoir reçu toutes les données, le microcontrôleur calcule la somme de contrôle (checksum) à partir des données reçues pour vérifier l'intégrité des données.
- Le checksum est comparé à celui envoyé par le capteur pour vérifier s'il y a eu des erreurs de transmission.

5. Fin de la communication :

- Une fois toutes les données transmises et vérifiées, la communication entre le microcontrôleur et le capteur DHT22 est terminée.

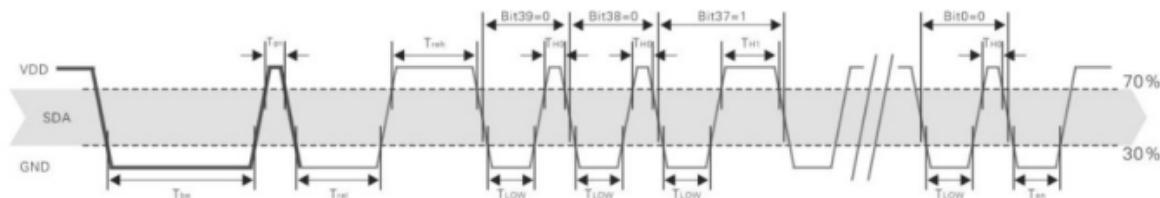


Figure 8: Temporisation de la communication Single-bus

Model	DHT22
Power supply	3.3-6V DC
Output signal	digital signal via single-bus
Sensing element	Polymer capacitor
Operating range	humidity 0-100%RH; temperature -40~80Celsius
Accuracy	humidity +2%RH(Max +5%RH); temperature <+0.5Celsius
Resolution or sensitivity	humidity 0.1%RH; temperature 0.1Celsius
Repeatability	humidity +1%RH; temperature +0.2Celsius
Humidity hysteresis	+0.3%RH
Long-term Stability	+0.5%RH/year
Sensing period	Average: 2s
Interchangeability	fully interchangeable
Dimensions	small size 14*18*5.5mm; big size 22*28*5mm

Figure 9: Spécificités techniques du DHT22

2.5. Le code :

```
/*début de la communication avec le capteur*/  
  
HAL_Delay(1000);  
Data_Output(GPIOA, GPIO_PIN_1); //info vers le capteur  
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_RESET); // remise à l'état bas  
DWT_Delay_us(1200); //signal de commande > 1ms  
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_SET); // état haut  
DWT_Delay_us(30); //signal de commande  
Data_Input(GPIOA, GPIO_PIN_1); //info vers le microcontrôleur
```

Figure 10: Code de la communication avec le capteur

```
/*début de la réception de données*/  
  
while(!(HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_1))){  
  
for (k=0;k<1000;k++){  
{  
if (HAL_GPIO_ReadPin (GPIOA, GPIO_PIN_1) == GPIO_PIN_RESET) {  
break;}}}  
while(!(HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_1)));  
DWT_Delay_us(40);  
  
Read_data(&dataH1);  
Read_data(&dataH2);  
Read_data(&dataT1);  
Read_data(&dataT2);  
Read_data(&SUM);  
  
check = dataH1 + dataH2 + dataT1 + dataT2; //pour vérifier la lecture dans le IDE  
  
if(check == (SUM)){  
RH = (dataH1<<8) | dataH2;  
TEMP = (dataT1<<8) | dataT2; }  
  
Humidite = RH / 10.0;  
Temperature = TEMP / 10.0;  
  
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_SET); //pour la prochaine lecture
```

Figure 11: Code de la réception de données

Remarque: L'intégralité de notre code est disponible sur notre dépôt Github

2.6. Visualisation de la trame:

En utilisant un PicoScope, nous avons observé la trame de réponse du capteur DHT22. Après l'avoir convertie d'abord en binaire puis en hexadécimal, nous obtenons les valeurs de température et d'humidité.

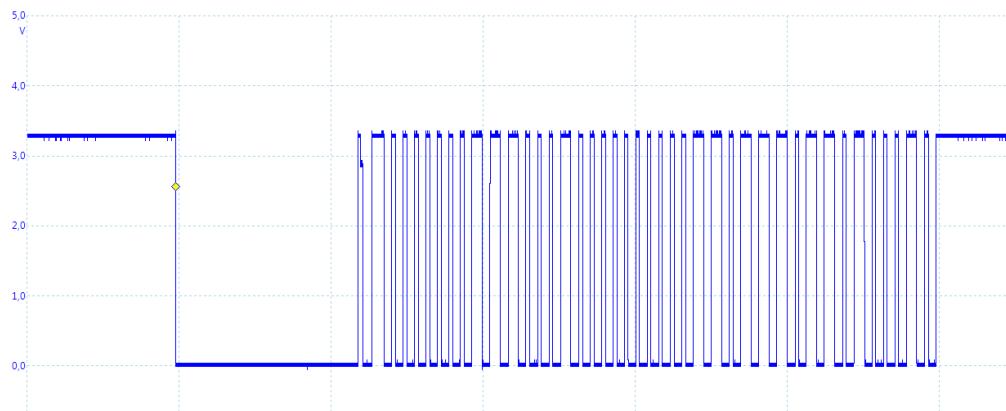


Figure 12: Trame du capteur DHT22

2.7. Visualisation du résultat:

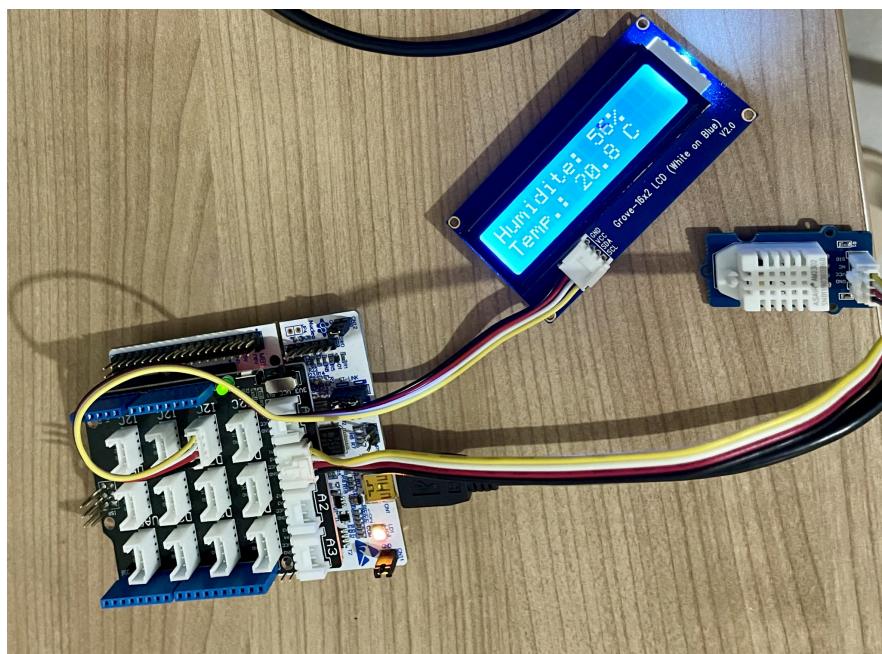


Figure 13: Affichage du résultat sur l'écran LCD

3. Projet N°2: STM32 + SHT31 + LCD

2.1. Matériel utilisé :

Carte Nucleo STM32L476RG

Capteur SHT31

Ecran LCD JHD1802M1

Shield Base

Un Picoscope

3.2. Schéma de câblage:

À l'aide du logiciel Fritzing, nous avons élaboré notre schéma de câblage suivant :

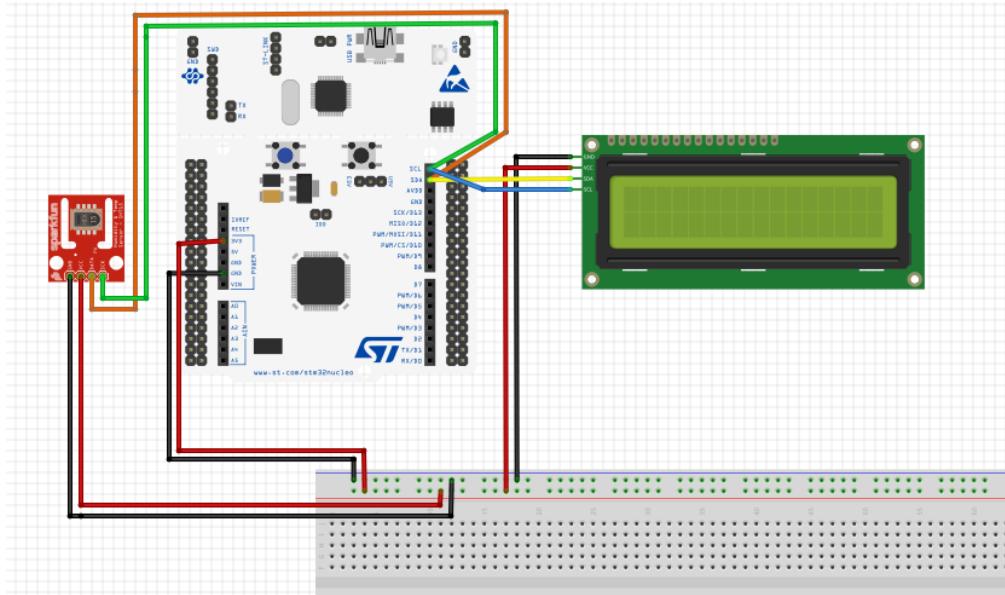


Figure 14: Schéma de câblage du SHT31

3.3. Configuration et initialisation des pins

Pour intégrer ce capteur, nous avons choisi d'utiliser les mêmes broches que celles déjà attribuées à l'afficheur LCD, à savoir les broches PB9 et PB8 pour SDA et SCL respectivement.

Cette décision découle du fait que le capteur et l'afficheur partagent le même bus de données I2C, mais avec des adresses différentes. Ainsi, en utilisant ces broches, nous simplifions la gestion des connexions matérielles et évitons la nécessité d'affecter des broches supplémentaires pour le capteur. Cela contribue à une meilleure efficacité dans le développement et l'utilisation des ressources disponibles

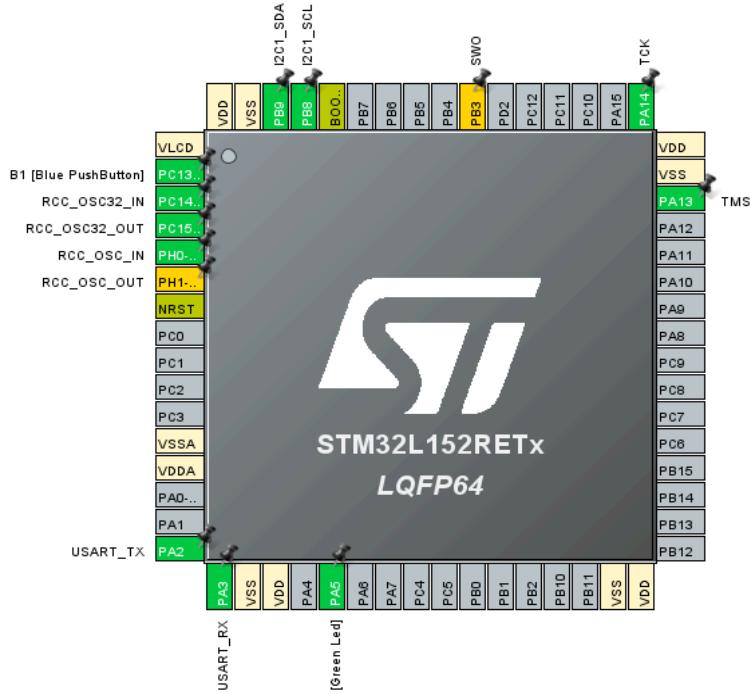


Figure 15: Configuration du STM32

3.4. Spécifications du SHT31:

Le capteur SHT 31 est renommé pour sa fiabilité, sa précision et sa réactivité rapide. Il offre une précision typique de $\pm 2\%$ RH pour l'humidité relative et de $\pm 0.3^{\circ}\text{C}$ pour les mesures de température. Il est compatible avec les tensions de 3.3 Volts et 5 Volts, éliminant ainsi le besoin d'un décaleur de niveau de tension.

La communication s'effectue via le bus série I²C, avec une prise en charge de vitesses allant jusqu'à 1 MHz. De plus, une bibliothèque hautement abstraite est mise à disposition pour simplifier son utilisation.

Specifications #	
Parameter	Value
Input voltage (VCC)	3.3 volts or 5 volts
I/O Logic Level	3.3 volts or 5 volts based on VCC
Operating Current	100 μA
Operating Temperature	-40–125 $^{\circ}\text{C}$
Temperature Sensor Range	-40–125 $^{\circ}\text{C}$, with $\pm 0.3^{\circ}\text{C}$ accuracy
Humidity Sensor Range	0% - 100%(Relative Humidity), with $\pm 2\%$ accuracy
Sensor Chip	SHT31(Datasheet)
Port	I ² C
Weight	4 g (for breakout board), 9 g for whole package each piece
Dimensions	40(length)×20(width) mm

Figure 16: Caractéristiques du SHT31

3.5. Fonctionnement du capteur SHT31:

- 1. Initialisation du capteur :** Lorsque le microcontrôleur démarre ou lorsqu'une nouvelle session de mesure est nécessaire, des commandes d'initialisation sont envoyées au capteur via le bus I2C. Ces commandes incluent généralement des instructions pour activer le capteur, configurer les paramètres de mesure, et démarrer le processus de capture de données. Par exemple, dans le code fourni, les commandes d'initialisation sont envoyées via HAL_I2C_Master_Transmit() pour démarrer une mesure de température et d'humidité.
- 2. Acquisition des données :** Une fois que le capteur est correctement initialisé, il commence à mesurer la température et l'humidité de l'environnement. Ces mesures sont effectuées à l'aide de composants internes sensibles aux variations de température et d'humidité. Une fois que la mesure est terminée, les données sont stockées dans les registres internes du capteur, prêtes à être lues par le microcontrôleur via le bus I2C. Dans le code fourni, les données sont reçues du capteur via HAL_I2C_Master_Receive().
- 3. Traitement des données :** Les données brutes reçues du capteur nécessitent souvent un traitement avant d'être utilisées. Ce traitement peut inclure des opérations telles que la conversion des données brutes en unités utilisables, le calcul de valeurs moyennes ou la vérification de la qualité des mesures. Par exemple, dans le code fourni, les valeurs brutes de température et d'humidité sont converties en degrés Celsius et en pourcentage respectivement, puis affichées sur un écran LCD.
- 4. Boucle principale :** Après l'initialisation du capteur, l'acquisition des données et le traitement initial, le microcontrôleur entre dans une boucle principale où il continue à surveiller l'environnement et à traiter les données de manière continue. Dans cette boucle, le microcontrôleur peut également effectuer d'autres tâches telles que la communication avec d'autres périphériques ou la prise de décisions en fonction des données capturées. La boucle principale garantit que le microcontrôleur reste opérationnel et réactif aux changements environnementaux.

3.6. Le code :

```
while (1)
{
    buf[0] = CAPTEUR_CMD_MSB;
    buf[1] = CAPTEUR_CMD LSB;
    ret = HAL_I2C_Master_Transmit( &hi2c1, CAPTEUR_ADDRS, buf, 2, HAL_MAX_DELAY);
    if ( ret != HAL_OK)
    {
        strcpy((char*)buf, "erreur_T!!\r\n");
    }
    else
    {

        ret = HAL_I2C_Master_Receive( &hi2c1, CAPTEUR_ADDRS, buf, 6, HAL_MAX_DELAY);
        if ( ret != HAL_OK)
        {
            strcpy((char*)buf, "erreur_R!!\r\n");
        }
    }
}
```

```

else
{
    valeur = buf[1] | buf[0] << 8;

    temp = -45 + 175 * ( float)valeur / 65535;

    partieEntiere = (int) temp;
    partieDecimal = temp;
    partieDecimal *= 100;
    partieDecimal = partieDecimal - (partieEntiere * 100);

    valeur = buf[4] | buf[3] << 8;

    umid = 100*( float)valeur / 65535;

    sprintf( (char*)buf, "%u.%u C ; %u ", (unsigned int) partieEntiere,
            (unsigned int) partieDecimal,(unsigned int) umid );
}

```

Figure 17: Code du SHT31

3.7. Visualisation de la trame:

La récupération de l'adresse du capteur est essentielle pour établir la communication avec celui-ci et lui envoyer des commandes spécifiques. Dans notre cas, l'utilisation du Picoscope permet de visualiser cette adresse, qui est affichée en hexadécimal, équivalente à 0x44. Cette adresse est unique pour chaque capteur et permet au microcontrôleur de cibler spécifiquement le capteur avec lequel il souhaite communiquer sur le bus I2C.

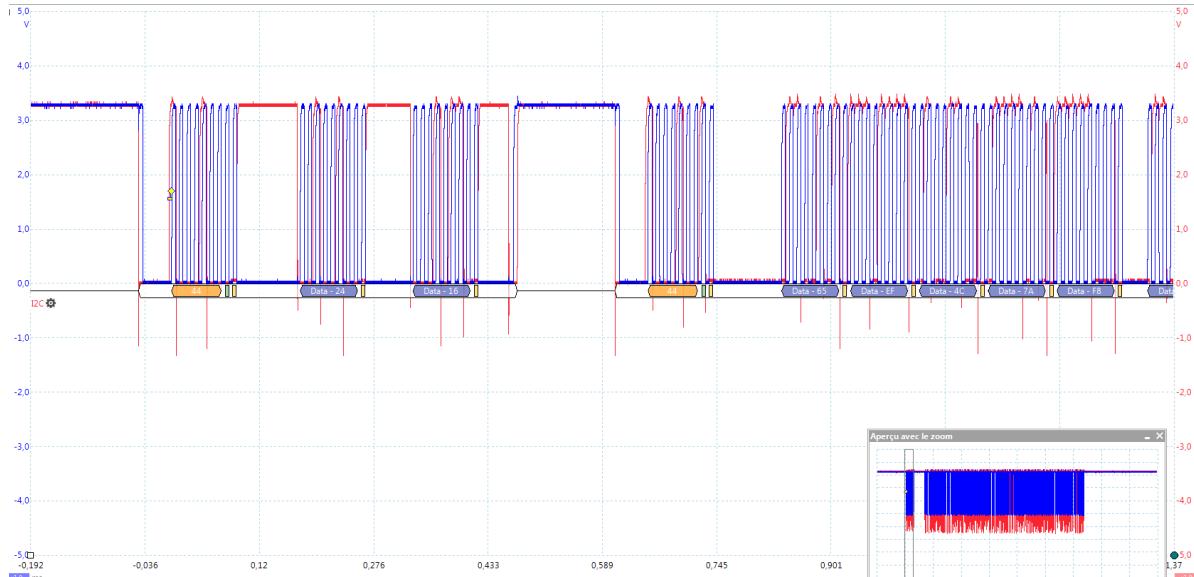


Figure 18: Trame du capteur DHT22

III. Conclusion:

Dans cette première partie du projet, nous avons utilisé les capteurs SHT31 et DHT22. Ce qui nous a permis de renforcer notre expertise dans l'utilisation des cartes STM32 et de mieux comprendre le fonctionnement des différents composants ainsi que différents protocoles de communication.

Cette expérience nous a non seulement permis de vérifier les fonctionnalités des capteurs, mais également d'approfondir notre compréhension des spécifications techniques et des protocoles de communication associés.