# Lab Worksheet 2

**CSE360: Computer Interfacing**
**Department of Computer Science and Engineering**

## Lab 01: Introduction to UART protocol using a DHT22 sensor on STM32.

### I. Topic Overview

This lab worksheet is to introduce the UART protocol on an STM32 microcontroller, using it to receive data from a DHT22 sensor. This involves setting up the UART communication on the STM32, interfacing with the DHT22 sensor, and writing the necessary code to read and display the temperature and humidity data.

### II. Learning Outcome

After this lab, students will be able to:

1. Understand the basic principles of UART communication and its configuration on an STM32 microcontroller.
2. Interface a DHT22 sensor with an STM32 and implement the protocol to read data from the sensor.
3. Process the sensor data to obtain meaningful temperature and humidity readings.
4. Transmit the sensor data over UART to a terminal or PC and display it in a human-readable format.

### III. Materials

- STM32 development board (e.g., STM32F446RE)
- DHT22 Sensor
- Jumper wires
- Breadboard

## IV. STM32F446RE Pins

The board consists of 64 pins out of which 38 pins are used and depicted in the following diagram. The relevant ports, pins and registers for today's lab are described later.

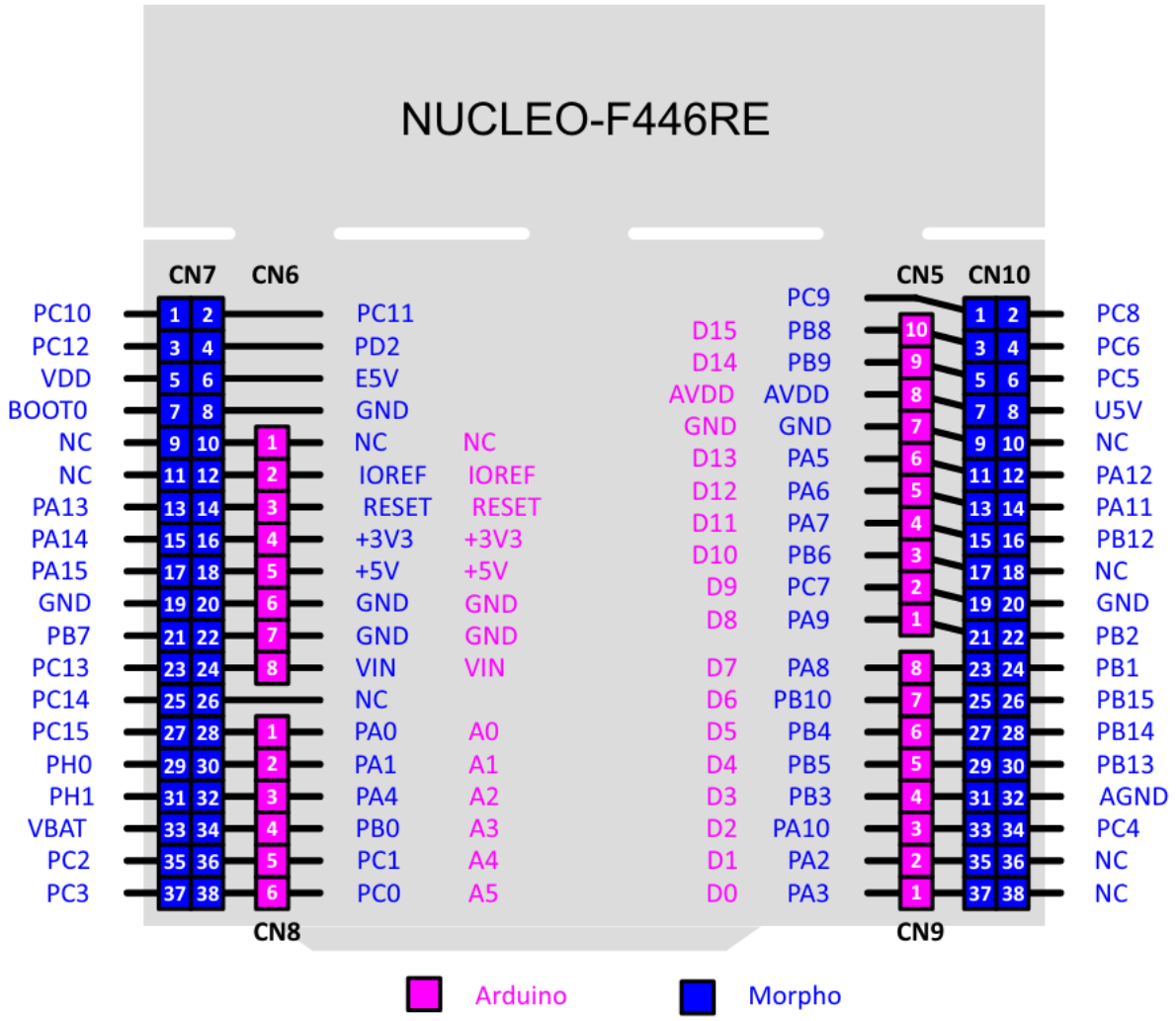



**Figure 3: Pin diagram of STM32 development board**

**Power:** 5 watt through the CN1 ST-Link USB mini B connector demonstrated in Fig. 1.

**Bus:** APB1 and APB2 peripheral bus, AHB port bus.

**Ports:** 8 GPIO ports (A-H) each with 16 pins. We will be using the ones depicted in Fig. 3 annotated as PortNamePinNumber. For example, port A pin 0 is annotated as PA0. These GPIO pins can be used for all types of signals.

**Ground and Voltage Source:** one 3.3V, one 5V and six GND.

## V. Registers:

1. **32-bit AFR register:** Alternate function register's first bit**.**
2. **32-bit CR1 register:** To enable data communication through UART protocol activate 13th pin (UE)

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | UE | M | WAKE | PCE | PS | PEIE | TXEIE | TCIE | RXNEIE | IDLEIE | TE | RE | RWU | SBK |
| | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

3. **32-bit BRR register:** To set up the baud rate of UART.

**Formula of baud rate:**

$$\text{Tx/ Rx baud} = \frac{f_{CK}}{(16*\text{USARTDIV})}$$

*legend: $f_{CK}$ - Input clock to the peripheral (PCLK1 for USART2, 3, 4, 5 or PCLK2 for USART1)*

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| DIV_Mantissa[11:0] | | | | | | | | | | | | DIV_Fraction[3:0] | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

4. **32-bit SR Register:** To check if the RX and TX buffer is empty or not.

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | CTS | LBD | TXE | TC | RXNE | IDLE | ORE | NE | FE | PE |
| | | | | | | rc_w0 | rc_w0 | r | rc_w0 | rc_w0 | r | r | r | r | r |

**VI. Experiment: Reading temperature and humidity from the environment using a DHT22 sensor.**

1. **Description the components:**

a) Breadboard

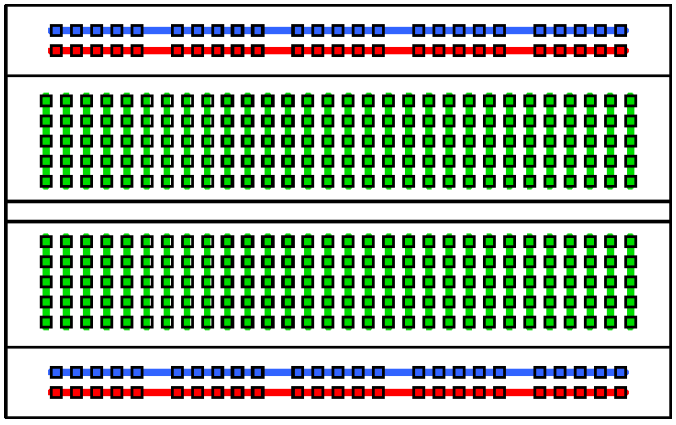


**Figure 9: Breadboard internal connection**

b) Jumper wires

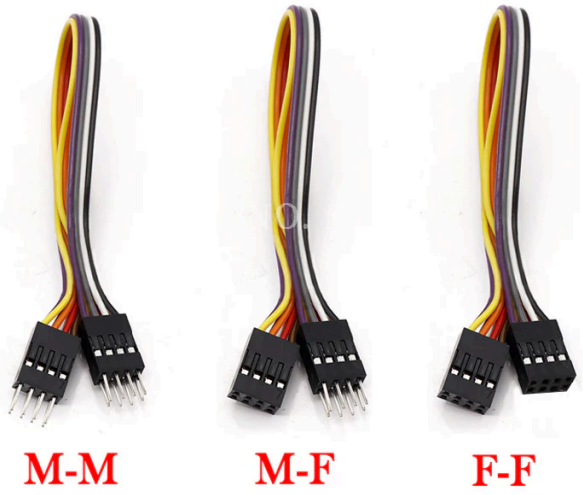



**Figure 10: Jumper wire types**

c) DHT22 Sensor:

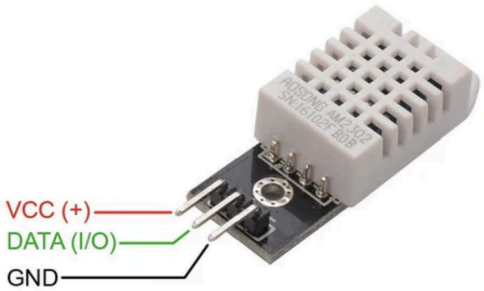



**Figure 11: DHT22 Sensor pins**

2. **Setting up the circuit:**

- Insert the DHT22 into the breadboard.

- Connect one end of a male to male jumper wire with the VCC pin of the DHT22 sensor and other end to the STM32 microcontroller board's VCC/ 5V pin.
- Connect one end of a male to male jumper wire with the GND pin of the DHT22 sensor and other end to the STM32 microcontroller board's GND pin.
- Connect one end of a male to male jumper wire with the DATA/OUT pin of the DHT22 sensor and other end to the STM32 microcontroller board's PA10 pin.
- Connect the STM32 board using the USB mini B connector with the computer.

3. **Circuit diagram:**

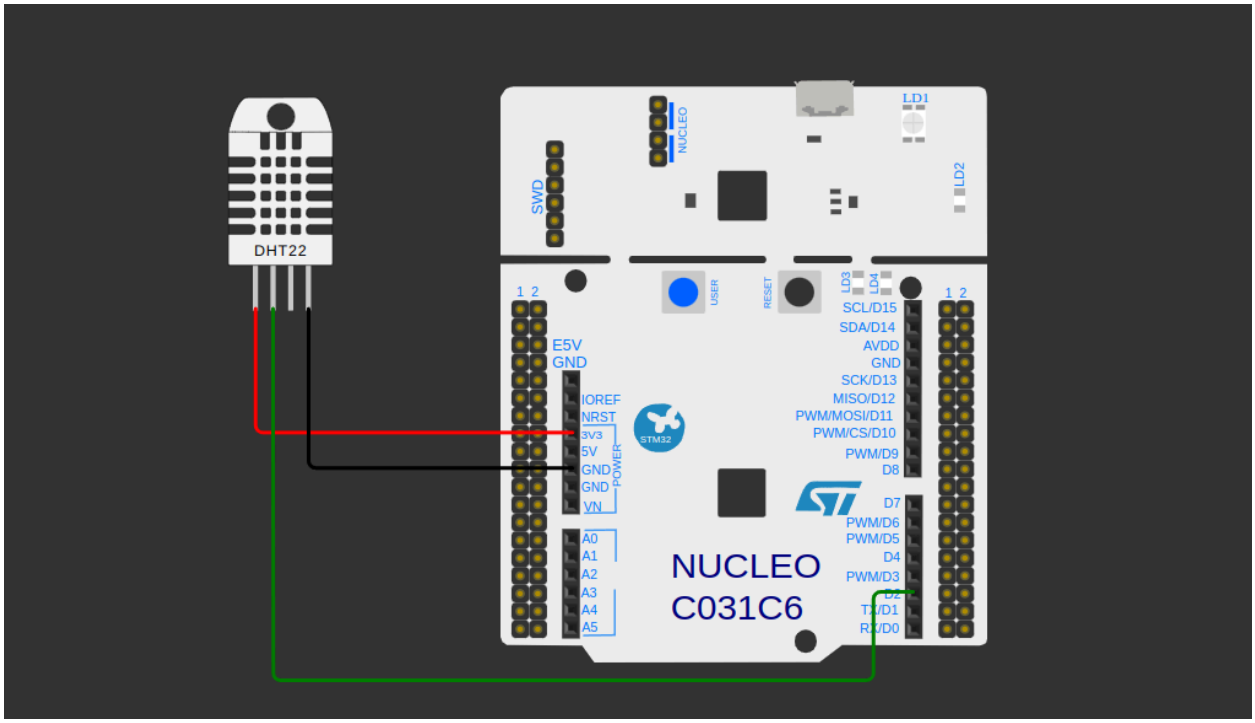


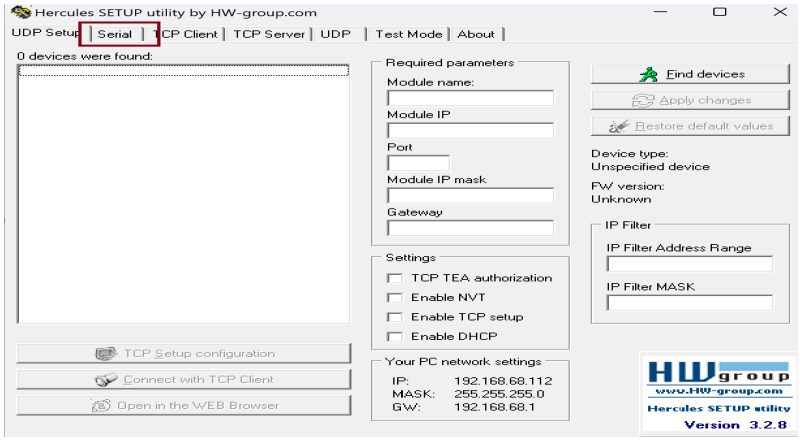

**Figure 12: Circuit diagram of a DHT22 sensor connected to PA10 pin of the STM32 board**

**VII. Install hercules:** It's a serial monitor. You need to install it to see the output.

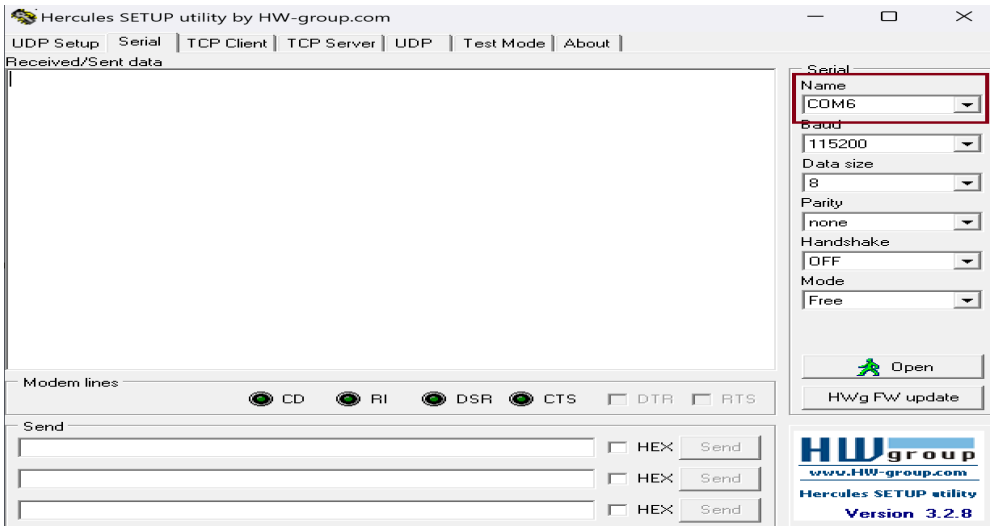
**Installation link:** [https://www.hw-group.com/product-version/hercules](https://www.hw-group.com/product-version/hercules) **[Install the latest version]**

**To see the output:**

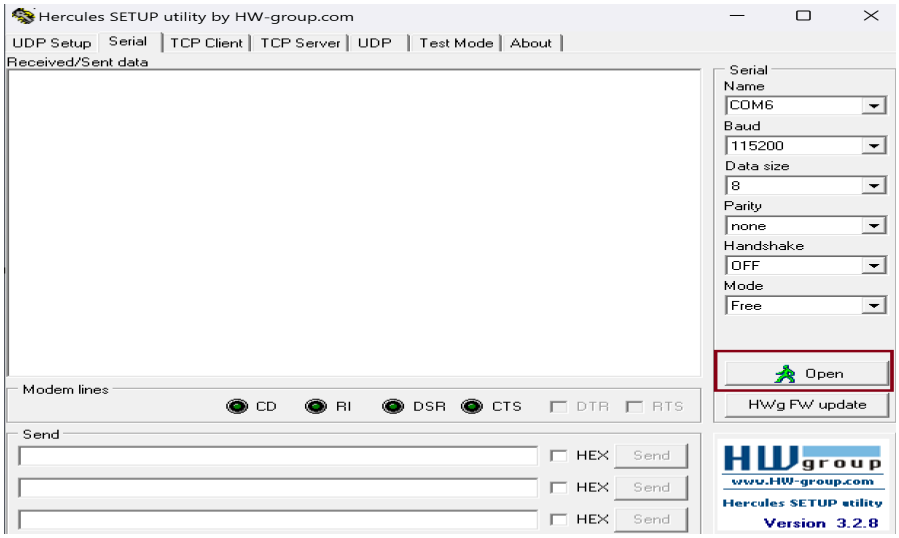
1.




2. Check the COM of your PC. *[to check COM: i. Open Device Manager, ii. Click on View in the menu bar and select Show hidden devices, iii) Locate Ports (COM & LPT) in the list, iv) Check for the com ports by expanding the same]*




3. Open the serial monitor




**VIII. Code:**

Open the Keil µVision5 IDE and follow the steps.

i) From the project tab in the menu bar, open a new project.

ii) In the select device for target window, select STM32F446RETx under STMicroelectronics and click ok.

iii) In the managed run-time environment window, select Core under CMSIS and Startup under Devices and click ok.

iv) To create files, right click on the Source Group 1 tab under Target 1 and select Add new item. A pop up window will appear for file type. Create the following files with the mentioned type and name. Paste the following codes in their respective files.

v) From the options for target menu (magic icon), go to debug window, select ST-Link Debugger

i) File name: uart.h, file type: header file (.h)

```
#ifndef INC_UART_H
#define INC_UART_H
#include "stm32f446xx.h"
#include "stm32f446xx.h"
#include <stdio.h>
```

```
void UART_Init();
void UART_SendChar(uint8_t ch);
uint8_t UART_ReceiveChar();
void UartSendBuffer(char buffer[]);
void UartReceiveBuffer(char buffer[], uint8_t len);

#endif
```

ii) File name: uart.c, file type: C file (.c)

```c
#include "uart.h"
void UART_Init()
{

    // Enabling Clock for GPIOA
    RCC->AHB1ENR |= (1U << 0U);

    // Configuring RX and TX Pin
    GPIOA->MODER |= (2U << 4U) | (2U << 6U);   // Setting Mode Alternate Function
    GPIOA->OSPEEDR |= (3U << 4U) | (3U << 6U); // Configuring speed High
    GPIOA->AFR[0] |= (7 << 8U) | (7 << 12U);   // Setting Alternate Function
number 7

    // Enabling Clock for UART
    RCC->APB1ENR |= (1U << 17U);

    // Disabling UART for Configuring
    USART2->CR1 &= ~(1U << 13U);

    // Enabling TX & RX
    USART2->CR1 |= (1U << 3U) | (1U << 2U);

    // Configuring Baud Rate
    uint32_t uart_div = 16000000UL / 115200U;

    USART2->BRR = ((uart_div / 16) << 4U) | ((uart_div % 16) << 0U);

    // Enabling UART
    USART2->CR1 |= (1U << 13U);
}

void UART_SendChar(uint8_t ch)
{
    while (!(USART2->SR & (1 << 7))) // holding until Tx Buffer is empty /// 1 =
empty, 0 = not empty
        ;
    USART2->DR = ch;
}
```

```c
uint8_t UART_ReceiveChar()
{
    while (!(USART2->SR & (1 << 5))) // holding Rx Buffer is not empty /// 0 = empty, 1 = not empty
        ;
    return USART2->DR;
}


void UartSendBuffer(char buffer[])
{
     int i = 0;
    while (1)
    {
         UART_SendChar(buffer[i]);
         i++;
         if (buffer[i] == '\0')
             break;
    }
}
void UartReceiveBuffer(char buffer[], uint8_t len)
{
     int i = 0;
     while (len--)
     {
         buffer[i] = UART_ReceiveChar();
         if ((buffer[i - 1] == '\r') || (buffer[i - 1] == '\n'))
         {
             break;
         }
          i += 1;
     }
}
```

iii) File name: main.c, file type: C file (.c)

```
#include "dht22.h"
#include "uart.h"
DHT dht22 = {GPIOA, 10};

int main()
{
   UART_Init();

   DHT_begin(dht22);

   char buffer[50]; // Character's array of length 50

   while (1)
   {

      float data = DHT_ReadTemperature(dht22, false);
      sprintf(buffer, "Tempareture: %.2fC\n", data);
      UartSendBuffer(buffer);


      float humidity = DHT_ReadHumidity(dht22);
      sprintf(buffer, "Humidity: %.2f%%\n", humidity);
      UartSendBuffer(buffer);
   }
}
```

**Driver code files:** *[Thanks to Asraful Islam Taj for his special contribution]*

**DHT22 Sensor Library Code from Github**

After going to the github link there will be two files dht22.c and dht22.h create two files with the same name into your keil project and copy those code from that library use these 3 functions

**1. DHT_begin(DHT Config)**

This function initialize all the necessary peripherals for the sensor to work

**2. DHT_ReadTemperature(DHT Config, bool isFahrenheit);**

This function returns the temperature of the dht 22 sensor you can provide true isFahrenheit parameter to get temperature in Fahrenheit and false for celsius

**3. float DHT_ReadHumidity(DHT Config);**

This function reads and returns the humidity from the sensor

Now if we notice all these functions takes a parameter called Config that is a Struct of DHT type

Now according to our choice we can create one like this

```
DHT dht22 = {GPIOA, 10};
```

```
void DHT_begin(DHT Config);
float DHT_ReadTemperature(DHT Config, bool isFahrenheit);
float DHT_ReadHumidity(DHT Config);
```

After creating all the files, save all files > batch build > download.

**VIII. Lab evaluation:** Show “Hello world” message using UART protocol in hercules serial monitor.

**IX. References:**


1. https://www.st.com/en/microcontrollers-microprocessors/stm32f446re.html
2. https://www.electronicsplanet.ch/en/electronics/resistor-color-code/resistor-color-code.php
3. https://pmdway.com/products/2-54mm-0-1-pitch-dual-row-jumper-cables-various-types-5-pack
4. http://designbuildcode.weebly.com/breadboard-circuits.html