Project_4.0_UART

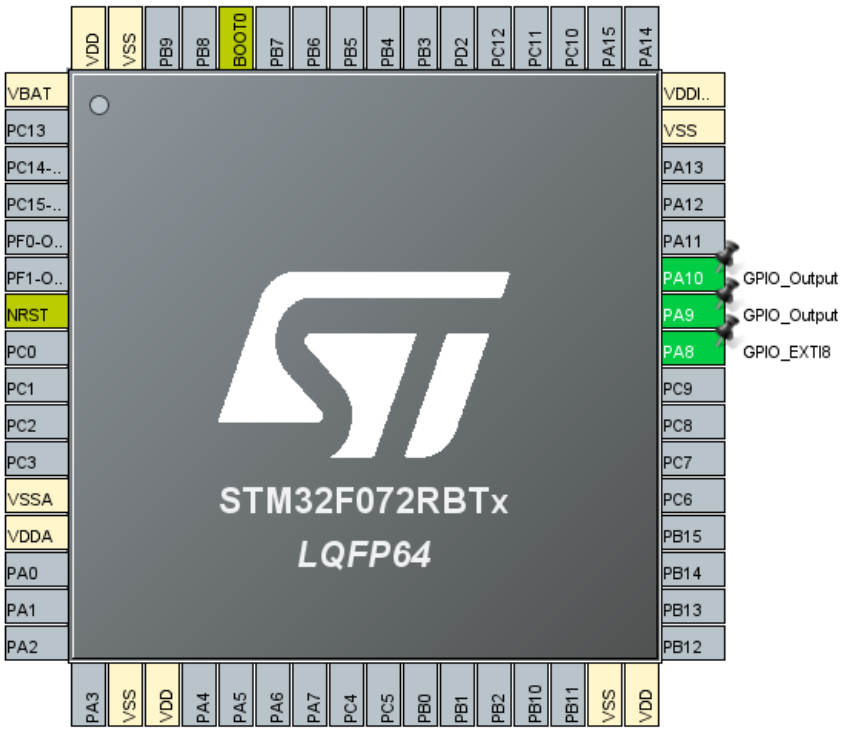
In this project, we will communicate between the computer and the MCU using UART. We will configure a GPIO interrupt and send data to the computer when the interrupt occurs. We will also send data from the computer to the MCU to toggle the LED on and off.

STEP 1 : Open STM32CubeIDE and Create New Project.

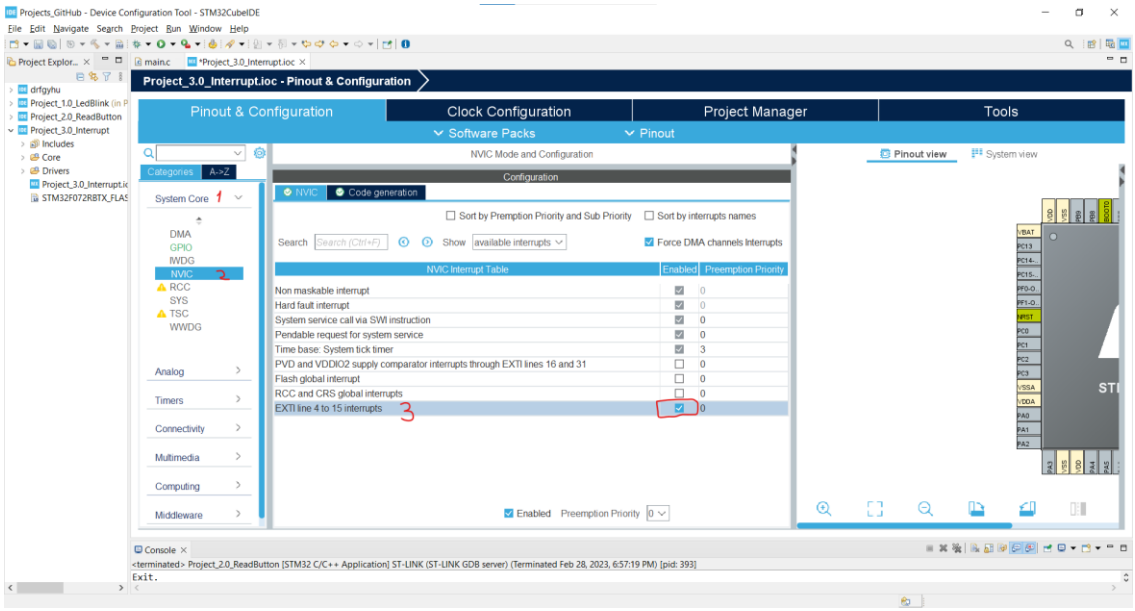
STEP 2 : Select Target microcontroller and double Click. My MCU is STM32F072RBT6.

STEP3 : Enter the project name and finish .

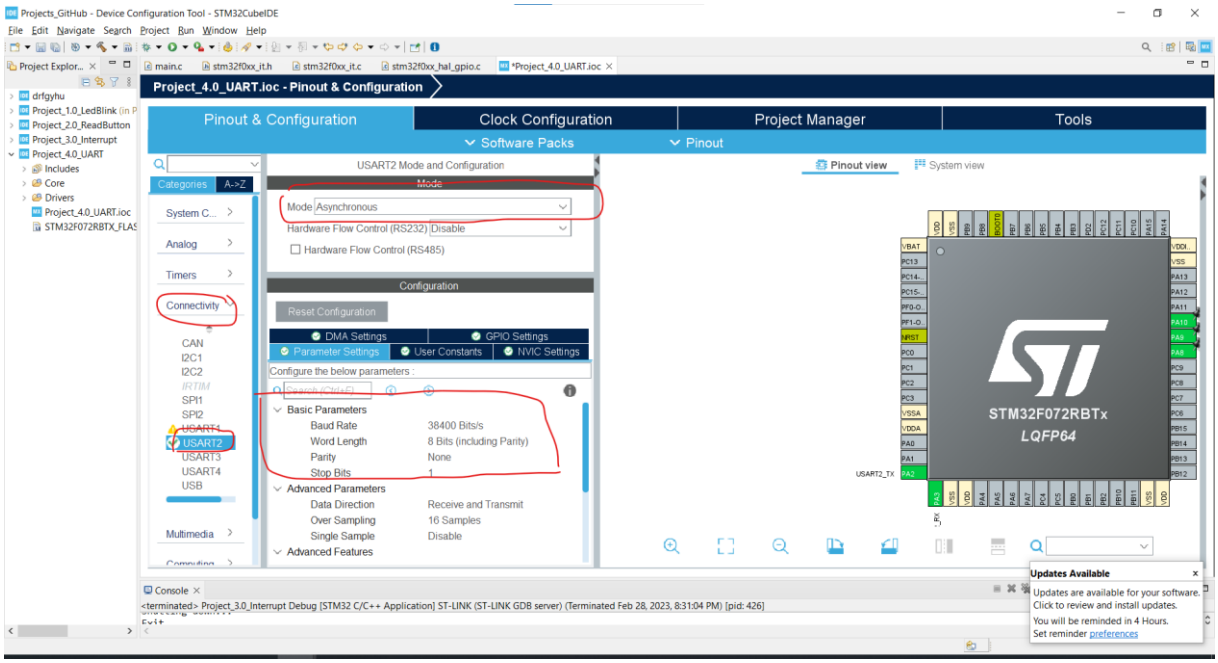
STEP 4 : Click the pin you want to set as input for interrupt and select GPIO_EXTIx/ GPIO_output. For example, in this project, I choose the A9/A10 pin for output and A8 pin for input.



STEP 5: Open The NVIC Tab And Enable The EXTI line8 Interrupt.



STEP 6: Enable USART2 Module and chose paramater settings.



STEP 6 : Set the RCC External Clock Source and then CTRL + S to generate the project code. And we open our main.c file in the project files.

Then write code:

Firstly, When we press the button and create an interrupt, let's write the code needed to send data from the mcu to the computer.

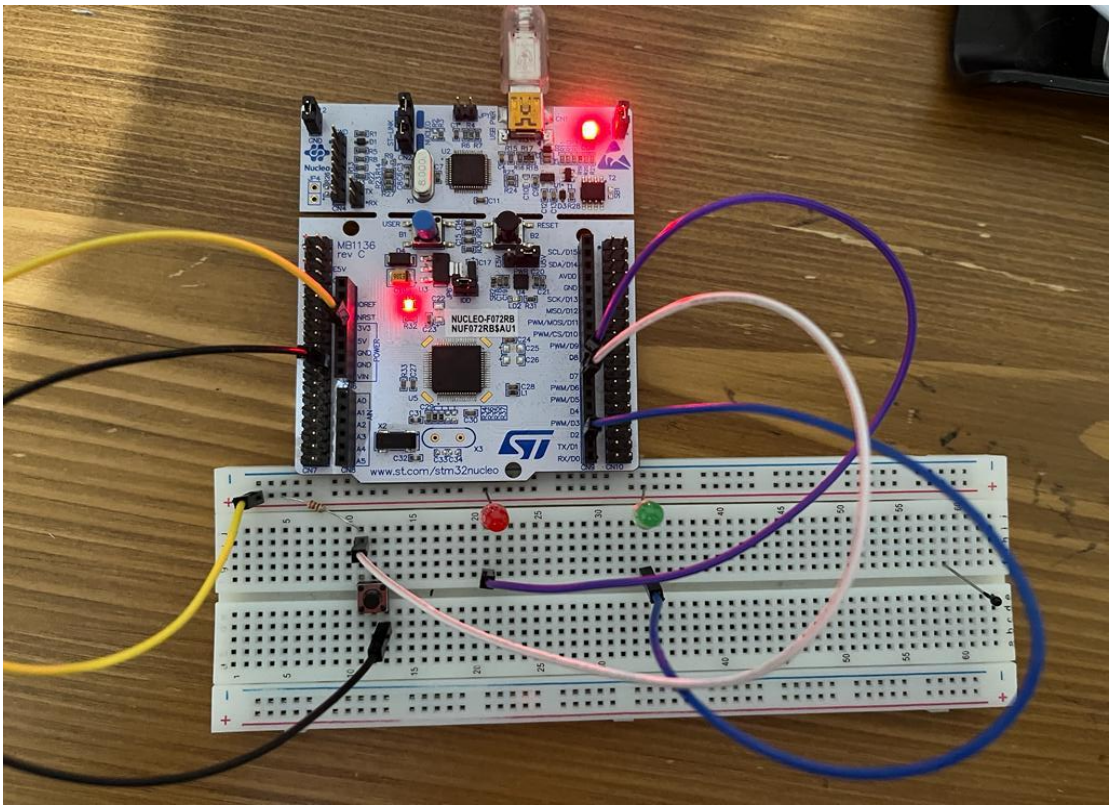
```
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

uint8_t MSG[] = "Button Pressed\r\n";                          //The message that will be sent when we press the button.

void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)                 // EXTI Line[4:15] External Interrupt  CallBackFunction
{
    if (GPIO_Pin == GPIO_PIN_8)                                // If The Interrupt source Is EXTI Line8 (A8 Pin)
    {
        HAL_UART_Transmit(&huart2, MSG, sizeof(MSG), 100);     //Send the MSG with UART
    }

}
/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
```

The code required to send data from the computer to the MCU and toggle the LEDs.

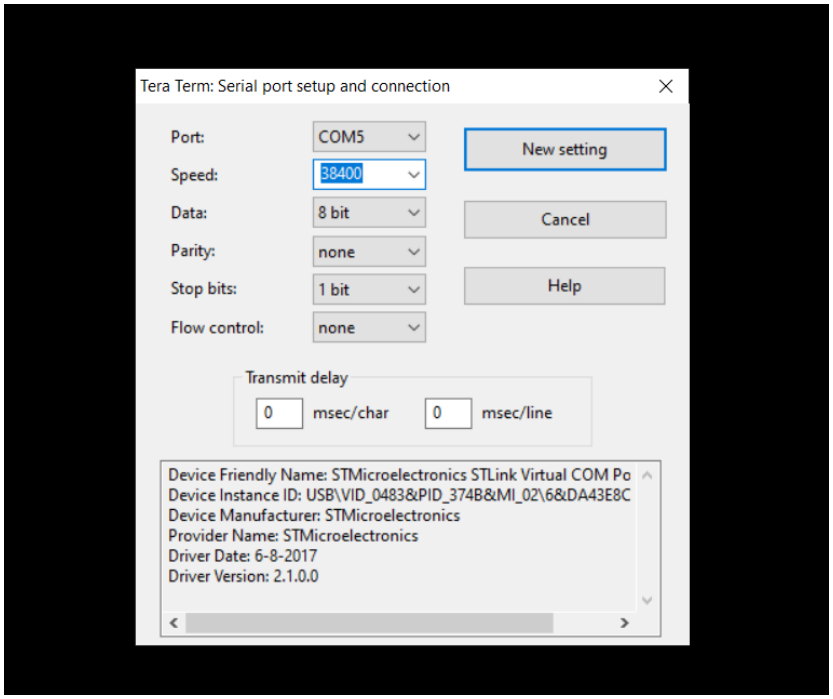
```
/* USER CODE BEGIN WHILE */
  while (1) {
      /* USER CODE END WHILE */

      /* USER CODE BEGIN 3 */
      if (HAL_UART_Receive(&huart2, &RX1_Char, 1, 10) == HAL_OK) { // We are listening. We are checking if there is any incoming data.
          if (RX1_Char == '1') {                                    //  If there is incoming data, we enter the function and check the incoming data
              HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_6);               //  If the incoming data is "1", we toggle PA6; if it is "2", we toggle PA10.
              RX1_Char = 0;
              HAL_UART_Receive(&huart2, &RX1_Char, 1, 10);

          }
          if (RX1_Char == '2') {
              HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_10);
              RX1_Char = 0;
              HAL_UART_Receive(&huart2, &RX1_Char, 1, 10);

          }
          HAL_Delay(100);                                       // We are creating a small delay to prevent data from overlapping
      }
  }
```

STEP 7 : We press RUN to compile the code and upload it to the board.

STEP 8 : Now it's time to connect the led and button to the board.



To verify that we can communicate with the MCU, we are using a terminal emulator program. I use TeraTerm. We configure the settings of this program according to the settings we used in the USART module of our MCU.



Everything is ready. When we press the button, a signal will be generated and data will be sent to the computer via UART. The terminal will display the message "Button pressed". At the same time, we can send data from the terminal screen to the MCU. If the data sent is "1" or "2", the corresponding LED will toggle.