LAB: USART - LED, Bluetooth

Date : 2023.11.10

Author/Partner: Duwon Yang/ Sunwoo Kim

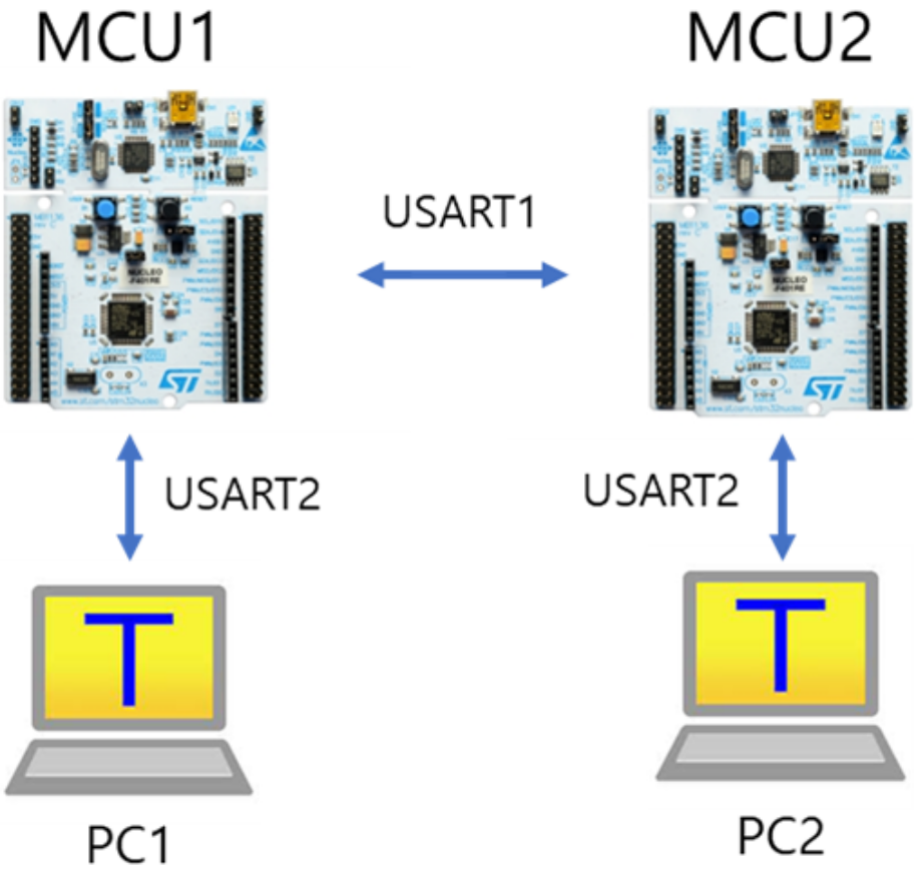
Github : LINK

Demo Video : LINK1 LINK2

Introduction

In this lab, We configure and use 'USART(Universal synchronous asynchronous receiver transmitter)' of MCU. Then, we communicate between my PC and MCU and to another MCU with wired serial communication.

- **Mission 1** : Control LED(LD2) of each other MCU.
- **Mission 2** : Run DC motors with Bluetooth



Requirement

Hardware

- MCU
  - NUCLEO-F411RE

- Actuator/Sensor/Others:
  - DC motor, DC motor driver(L9110s)
  - Bluetooth Module(HC-06)

Software

- Keli uVision, CMSIS, EC_HAL library

Problem 1:Create HAL library

Create HAL library

Declare and Define the following functions in library. and update header files located in the directory `EC\lib\`.

ecUSART.h

```
// Configuration UART 1, 2 using default pins
void UART1_init(void);
void UART2_init(void);
void UART1_baud(uint32_t baud);
void UART2_baud(uint32_t baud);

// USART write & read
void USART1_write(uint8_t* buffer, uint32_t nBytes);
void USART2_write(uint8_t* buffer, uint32_t nBytes);
uint8_t USART1_read(void);
uint8_t USART2_read(void);

// RX Interrupt Flag USART1,2
uint32_t is_USART1_RXNE(void);
uint32_t is_USART2_RXNE(void);
```

General USART setup

```
// General Setting
void setup(){
  RCC_PLL_init();

  // BT serial : specific RX/TX pins
  USART_setting(USART1, PA_9,PA_10, BAUD_9600);     // PA9 - RXD , PA10 - TXD
}
```

Problem2: Communicate MCU1-MCU2 using RS-232

Procedure

1. Create a new project under the directory `\repos\EC\LAB\LAB_USART_LED`

- The project name is **"LAB_USART_LED"**
- Create a new source file named as **"LAB_USART_LED.c"**

2. Include updated library in `\repos\EC\lib\` to project.

- **ecGPIO.h, ecGPIO.c**
- **ecRCC.h, ecRCC.c**
- **ecTIM.h, ecTIM.c**
- **ecPWM.h, ecPWM.c**
- **ecSysTick.h, ecSysTick.c**
- **ecUATT.h, ecUART.c**

3. Connect each MCUs to each PC with **USART2** via USB cable (ST-Link)

- MCU-PC1, MCU2-PC2

4. Connect MCU1 to MCU2 with **USART1**

- connect RX/TX pins externally as
    - MCU1_TX to MCU2_RXD
    - MCU1_RX - MCU2_TX

  Connect the same GND for MCU1-MCU2

5. Send a message from PC_1 by typing keys on Teraterm. It should send that message from MCU_1 to MCU_2

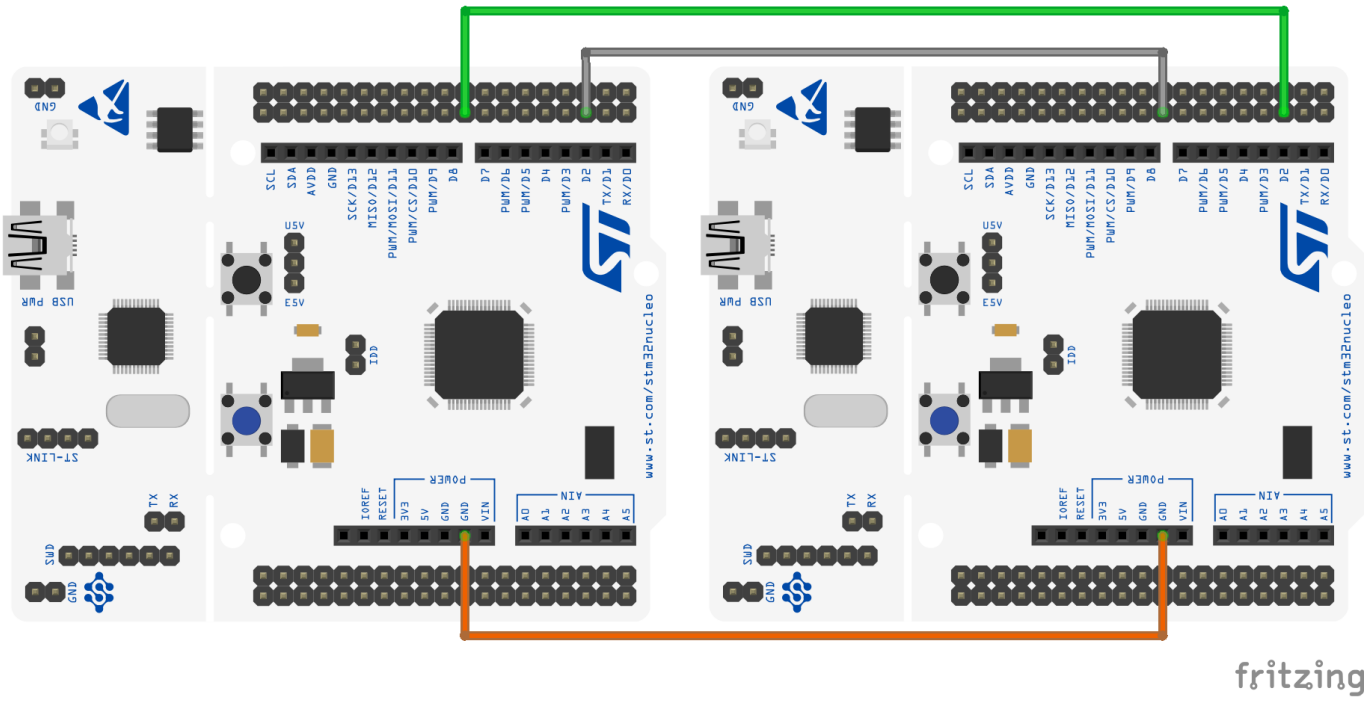
6. The received message by MCU_2 should be displayed on PC_2.

7. Turn other MCU's LED(LD2) On/OFF by sending text:

- "**L**"for Turn OFF
- "**H**"for Turn ON

Configuration

Type	Port - Pin	Configuration
System Clock		PLL 84MHz
USART2 : USB cable (ST-Link)		No Parity, 8-bit Data, 1-bit Stop bit, 38400 baud-rate
USART1 : MCU1 - MCU2	TXD: PA9 RXD: PA10	No Parity, 8-bit Data, 1-bit Stop bit, 38400 baud-rate
Digital Out: LD2	PA5	



Code

setup function and global parameter is defined below.

Set UART1/UART2 Baudrate as 9600 and initilize LED(PA_5) as OUTPUT MODE.

Define PC_Data, BT_Data for reading and save USART1/USART2 data.

```
#include "stm32f4xx.h"
#include "ecGPIO.h"
#include "ecRCC.h"
#include "ecUART.h"
#include "ecSysTick.h"
```

```
static volatile uint8_t PC_Data = 0;
static volatile uint8_t BT_Data = 0;
uint8_t PC_string[]="Loop:\r\n";

void setup(void){
   RCC_PLL_init();
   SysTick_init();

   // USART2: USB serial init
   UART2_init();
   UART2_baud(BAUD_9600);

   // USART1: BT serial init
   UART1_init();
   UART1_baud(BAUD_9600);

    GPIO_init(PA_5, OUTPUT);
}
```

USART1/USART2 Intterupt Request Handler is set.

```
void USART2_IRQHandler(){                // USART2 RX Interrupt : Recommended
   if(is_USART2_RXNE()){
      PC_Data = USART2_read();      // RX from UART2 (PC)
      USART2_write(&PC_Data,1);      // TX to USART2    (PC)    Echo of keyboard
typing
      USART1_write(&PC_Data,1);      // TX to USART2    (PC)    Echo of keyboard
typing

   }
}


void USART1_IRQHandler(){                // USART2 RX Interrupt : Recommended
   if(is_USART1_RXNE()){
      BT_Data = USART1_read();      // RX from UART1 (BT)

      printf("RX: %c \r\n",BT_Data); // TX to USART2(PC)
   }
}
```

In the main function, LED is ON when received data(BT_Data) equals to 'H', and OFF when received data(BT_Data) equals to 'L'.

the received data is displayed on the screen per 2000ms through the tera-term program.

```
int main(void){
    setup();
    printf("MCU Initialized\r\n");
```

```
    while(1){
        // USART Receive: Use Interrupt only
        // USART Transmit:  Interrupt or Polling
        USART2_write(PC_string, 7);
        if(BT_Data == 'H'){
            GPIO_write(PA_5, HIGH);
        }
else if(BT_Data == 'L'){
            GPIO_write(PA_5, LOW);
}
        delay_ms(2000);
    }
}
```

Result

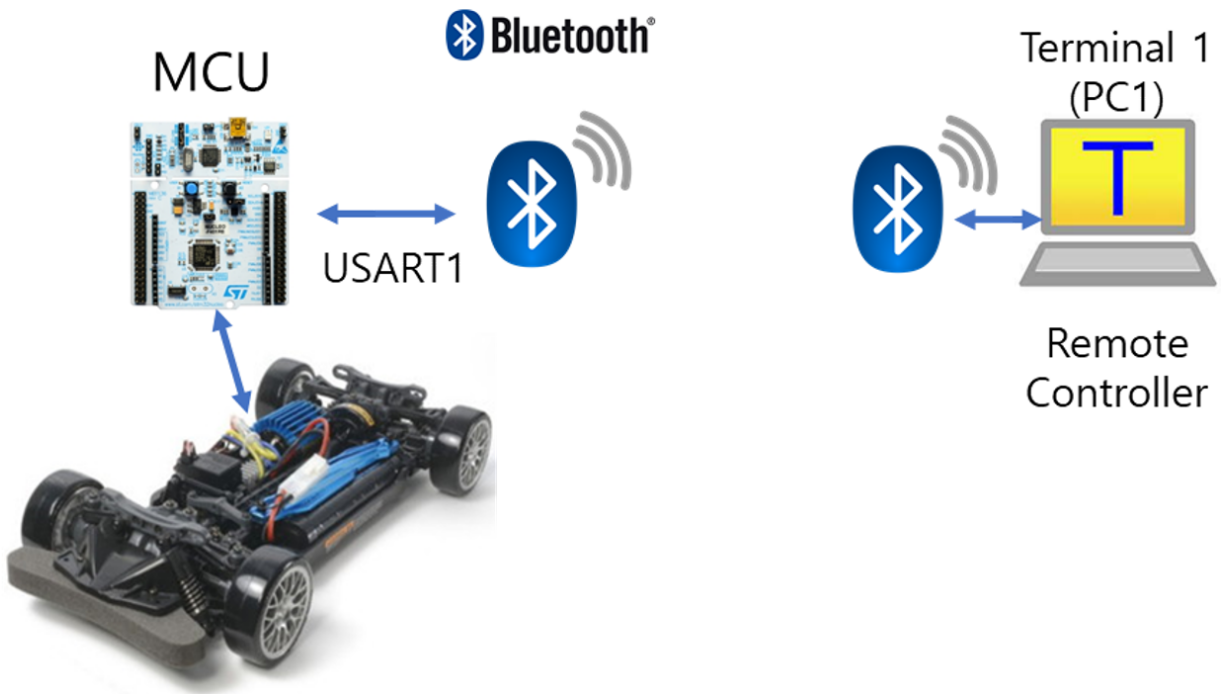
In this experiment, the objective is to effectively utilize USART2 communication between the PC and MCU, and USART1 communication between MCUs. The design involves transferring a key typed on PC1 to MCU1 via USART1 communication. This key is then exchanged between MCU1 and MCU2 using USART2 communication, and subsequently transferred from MCU2 to PC2 via USART1 communication. Based on this key, the program performs the task of turning an LED on and off.

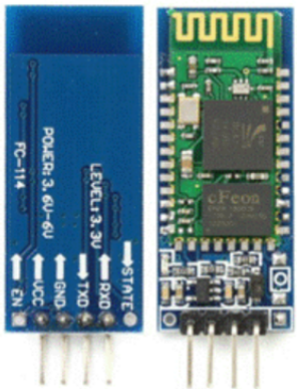
"L" for Turn OFF / "H" for Turn ON

demo video : LINK

Problem 3: Control DC Motor via Bluetooth

Bluetooth





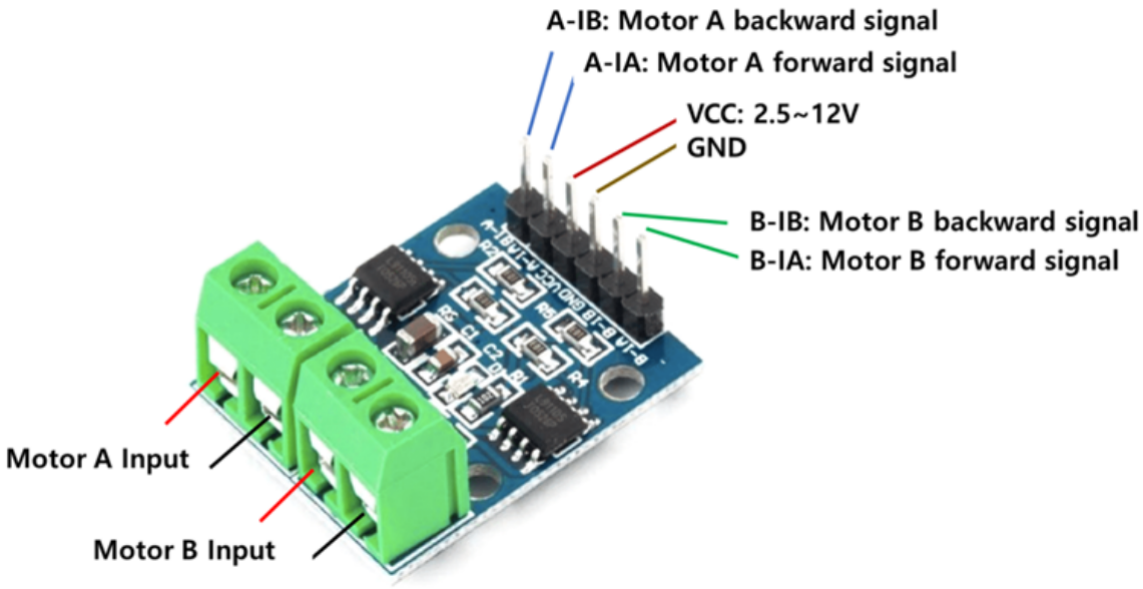
Bluetooth Module (HC-06)	STM32F411RE
RxD	PA_9(UART1_TX)
TxD	PA_10(UART1_RX)
GND	GND
VCC	5V

DC Motor Driver

Connect DC motor driver(L9110s) module pins to MCU as shown below.

DO NOT use MCU's VCC to motor driver. You should use external voltage source.

- A- IA: PWM pin (0~100% duty) for Motor A
- B- IB: Direction Pin (Digital Out H or L) for Motor B



Procedure

1. Create a new project under the directory `\repos\EC\LAB\LAB_USART\Bluetooth`

- The project name is **"LAB_USART_Bluetooth".**
- Create a new source files named as **"LAB_USART_Bluetooth.c"**

2. Include your updated library in `\repos\EC\lib\` to your project.

- **ecGPIO.h, ecGPIO.c**
- **ecRCC.h, ecRCC.c**
- **ecTIM.h, ecTIM.c**
- **ecPWM.h, ecPWM.c**

- **ecSysTick.h, ecSysTick.c**
- **ecUATT.h, ecUART.c**

3. Connect the MCU to PC via Bluetooth. Use USART 1

- connect RX/TX pins as
    - MCU TXD - BLUE RXD
    - MCU RXD - BLUE TXD

4. Check the Bluetooth connection by turning MCU's LED(LD2) On/OFF by sending text of **"1"** or **"0"** from PC.

5. Run 2 DC motors(Left-wheel, Right-wheel) to steer.

- Turn Left: MotorA / MotorB = (50 / 80%) duty
- Turn Right: MotorA / MotorB = (80 / 50%) duty
- Go Striaght : MotorA / MotorB = (80 / 80%) duty
- STOP : MotorA / MotorB = (0 / 0%) duty

use the key inputs as your preference for Left, Right, Straight.

Configuration

Type	Port - Pin	Configuration
System Clock		PLL 84MHz
USART1 : MCU - Bluetooth	TXD: PA9 RXD: PA10	No Parity, 8-bit Data, 1-bit Stop bit, 9600 baud-rate
Digital Out: LD2	PA5	
PWM (Motor A)	TIM2-Ch1	PWM period (2kHz~10kHz)
PWM (Motor B)	TIM2-Ch2	

Code

setup function and global parameter is defined below.

Set UART1/UART2 Baudrate as 9600 and initilize LED(PA_5) as OUTPUT MODE.

Set DIR_LEFT/DIR_RIGHT pins and PWM_LEFT/PWM_RIGHT pins.

Define PC_Data, BT_Data for reading and save USART1/USART2 data.

```
#include "stm32f411xe.h"
// #include "ecSTM32F411.h"
#include "ecPinNames.h"
#include "ecGPIO.h"
#include "ecSysTick.h"
#include "ecRCC.h"
#include "ecTIM.h"
#include "ecEXTI.h"
#include "ecPWM.h"   // ecPWM2.h
#include "ecUART.h"


// Definition Button Pin & PWM Port, Pin
#define BUTTON_PIN 13
#define PWM_PIN PA_0
#define DIR_LEFT PC_2
#define DIR_RIGHT PC_3
#define PWM_LEFT PA_0
#define PWM_RIGHT PA_1
#define STOP 1

void setup(void);

float period = 50;
static volatile uint8_t PC_Data = 0;
static volatile uint8_t BT_Data = 0;

void setup(void){
   RCC_PLL_init();
   SysTick_init();

   // USART2: USB serial init
   UART2_init();
   UART2_baud(BAUD_9600);

   // USART1: BT serial init
   UART1_init();
   UART1_baud(BAUD_9600);

    GPIO_init(PA_5, OUTPUT);
}

void setup(void) {
    RCC_PLL_init();

        // LED setup
        GPIO_init(PA_5, OUTPUT);

    // BT serial init
    UART1_init();
    UART1_baud(BAUD_9600);
```

```
    // DIR1 SETUP
    GPIO_init(DIR_LEFT, OUTPUT);
    GPIO_otype(DIR_LEFT, EC_PUSH_PULL);
        GPIO_write(DIR_LEFT, 1);

    // DIR2 SETUP
    GPIO_init(DIR_RIGHT, OUTPUT);
    GPIO_otype(DIR_RIGHT, EC_PUSH_PULL);
        GPIO_write(DIR_RIGHT, 1);

    // PWM1
    PWM_init(PWM_LEFT);
    PWM_period_us(PWM_LEFT, period);

    // PWM2
    PWM_init(PWM_RIGHT);
    PWM_period_us(PWM_RIGHT, period);

}
```

USART1 Intterupt Request Handler is set. According to input spelling, We can confirm whether the connection is good. if the input is '0', LED is OFF, and '1' LED is ON. Moreover, we can set RIGHT and LEFT PWM to drive a RC car. It works 'GO','STOP','RIGHT','LEFT' for 'w','s','d','a'.

```
void USART1_IRQHandler(){                       // USART2 RX Interrupt :
Recommended
    if(is_USART1_RXNE()){
        BT_Data = USART1_read();                // RX from UART1 (BT)
        USART1_write(&BT_Data, 1);
        if(BT_Data == 's' || BT_Data == 'S'){
            PWM_duty(PWM_LEFT, 1);
            PWM_duty(PWM_RIGHT, 1);
        }
        else if(BT_Data == 'd' || BT_Data == 'D'){
            PWM_duty(PWM_LEFT, 0.5);
            PWM_duty(PWM_RIGHT, 0.2);
        }
        else if(BT_Data == 'a' || BT_Data == 'A'){
            PWM_duty(PWM_LEFT, 0.2);
            PWM_duty(PWM_RIGHT, 0.5);
        }
        else if(BT_Data == 'w' || BT_Data == 'W'){
            PWM_duty(PWM_LEFT, 0.2);
            PWM_duty(PWM_RIGHT, 0.2);
        }
                else if(BT_Data == '1') GPIO_write(LED_PIN, 1);
                else if(BT_Data == '0') LED_OFF();
    }
}
```

In the main function, PWM for LEFT, RIGHT DC Motor is set to work 'STOP'.

```
int main(void) {
    // Initialization ---------------------------------------------------
      setup();

    /*Initial Duty Set*/
    PWM_duty(PWM_LEFT, STOP);
    PWM_duty(PWM_RIGHT, STOP);
    while(1){

    }
}
```

Results

In this lab, we created a program to remotely control two DC motors using PWM via Bluetooth USART communication for an RC car. To verify the successful connection of the communication, we configured the system such that when '0' is entered, an LED turns off, and when '1' is entered, the LED turns on. As a result, we developed and tested a program where entering 'w' makes the car move forward, 's' makes it stop, and 'a' and 'd' make it turn left and right, respectively.

demo video : LINK

Reference

https://ykkim.gitbook.io/ec/ec-course/lab/lab-report-template

https://ykkim.gitbook.io/ec/ec-course/lab/lab-usart-led-bluetooth