LAB: USART - LED, Bluetooth

Date: 2023-11-16

Author/Partner: NohYunKi / MoonHyeonHo

Demo Video:

1. **LED**: https://youtu.be/bnHJNLVkkmw
2. **Bluetooth**: https://youtu.be/3wsc1a2efRQ

Introduction

```
In this lab, we used USART(Universal synchronous asynchronous receiver transmitter) communication to exchange signal between MCU boards or PCs. By using USART, we performed two assignments. First is turning on and off team partner's LED of MCU board. Second is to control DC Car by using Bluetooth.
```

Requirement

Hardware

- MCU
  - NUCLEO-F411RE
- Actuator/Sensor/Others:
  - DC motor, DC motor driver(L9110s),
  - Bluetooth Module(HC-06)

Software

- Keil uVision, CMSIS, EC_HAL library

Problem 1: Communicate MCU1-MCU2 using RS-232

Circuit Diagram

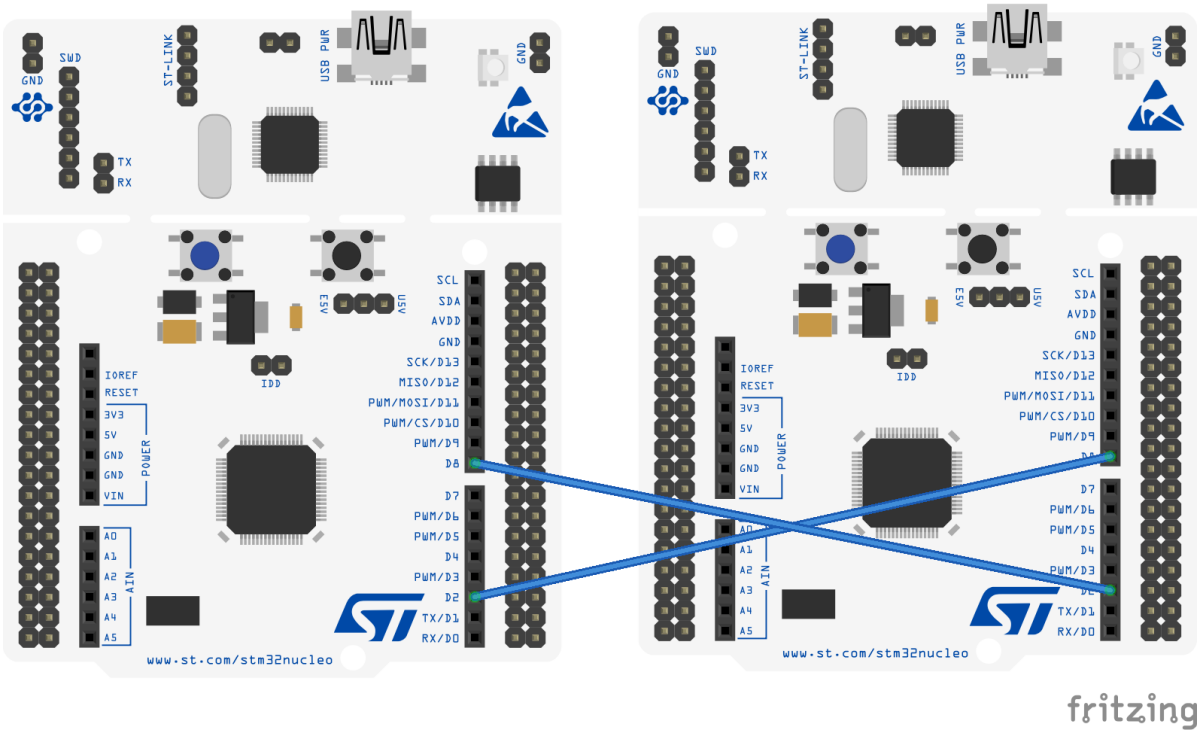

Figure 1. USART1 & USART2 Communication

In the Problem 1, we will turn on and off each other's LED by entering 'H' and 'L' comment. To do this, we connected each PA9 pin and PA10 pin with other partner's PA10 pin and PA9 pin by using USART 1. And to connect each PC with each MCU board, we used USART 2.

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

```
#define LED_PIN    5
#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;

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

```
  USART_setting(USART1, GPIOA,9,GPIOA,10, BAUD_38400); // PA9 - RXD , PA10 - TXD

  // USB serial init
  UART2_init();
  UART2_baud(BAUD_38400);

  // BT serial init
  UART1_init();
  UART1_baud(BAUD_38400);

   GPIO_init(GPIOA, LED_PIN, OUTPUT);    // calls RCC_GPIOA_enable()
   GPIO_otype(GPIOA, LED_PIN, 0);        //  GPIOA LED_PIN Output Type: Output
open drain (1)
   GPIO_ospeed(GPIOA, LED_PIN, EC_FAST);      // GPIOA Speed LED_PIN Medium
speed(01)

}


void main(){
   setup();

   while(1){

            if(BT_Data==0X4C){  //(L)
         GPIO_write(GPIOA,5, 0);
      }else if(BT_Data==0X48){ //(H)
         GPIO_write(GPIOA,5, 1);
         }

    }
}


void USART2_IRQHandler(){                // USART2 RX Interrupt : Recommended
   if(is_USART2_RXNE()){
      PC_Data = USART2_read();      // RX from UART2 (PC)
      USART1_write(&PC_Data,1);      // TX to USART1    (BT)
      printf("MCU_1 sent : %c \r\n",PC_Data); // TX to USART2(PC)
   }
}

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

      printf("MCU_1 received : %c \r\n",BT_Data); // TX to USART2(PC)
   }
}
```

In this code, PA9 and PA10 are for USART 1. And cable which connects each MCU board with each computer is for USART 2. To use each USART Communication, we set both speed of USART 1's speed and USART 2 as 38400. By using USART 2, each MCU board can receive comment from PC. Therefore in the 'USART2_IRQHandler', we can get 'PC_Data'. Also, we can get 'BT_Data' which

is exchanged between MCU board in the 'USART1_IRQHandler' because USART 1 makes us able to exchange signal between each MCU board.

Result

Link: https://youtu.be/bnHJNLVkkmw

Problem 2: Control DC Motor via Bluetooth

In this problem, we operate each DC motor by using Bluetooth. This means we sent and received signal between computer and MCU board wireless.

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

```
/**
******************************************************************************
* @author  SSSLAB
* @Mod     2023-11-15 by NohYunKi
* @brief   Embedded Controller:  LAB_USART_Bluetooth
*
******************************************************************************
*/

#include "stm32f4xx.h"
#include "ecGPIO.h"
#include "ecRCC.h"
#include "ecUART.h"
#include "ecPWM.h"
#include "ecPinNames.h"

#define UP       'W'
#define RIGHT    'D'
#define LEFT     'A'
#define STOP     'S'


PinName_t Front_Right_Pin = PA_1;
PinName_t Front_Left_Pin = PA_0;
```

```
uint8_t btData = 0;
static volatile uint8_t Command = 0;

void setup(void);
void Direction(uint8_t direction);
void Direction_display(uint8_t direction);
void Car_setting(void);

int signal = 0;

int main(void) {
   // Initialiization --------------------------------------------------------
   setup();

   // Inifinite Loop ---------------------------------------------------------
   while (1){
      Direction(btData);
   }
}

void USART1_IRQHandler(){         //USART1 INT
   if(is_USART_RXNE(USART1)){
      Direction_display(btData);
   }
}

// Initialiization
void setup(void)
{
   RCC_PLL_init();
   UART1_init();
   UART1_baud(BAUD_9600);

    // LED setting
   GPIO_setting(GPIOA,LED_PIN,OUTPUT,EC_PUSH_PULL,EC_PU,EC_FAST);

   Car_setting();

   PWM_duty(Front_Right_Pin,1);
   PWM_duty(Front_Left_Pin,1);
}

void Car_setting(void){
   PWM_init(Front_Left_Pin); //Front_Left
   PWM_period_us(Front_Left_Pin,100); // 10kHz
   PWM_init(Front_Right_Pin); //Front_Right
   PWM_period_us(Front_Right_Pin,100); // 10kHz
}



void Direction(uint8_t direction){
          // Up
        if(direction == UP){
```

```
            PWM_duty(Front_Right_Pin,0.2); // 80%
            PWM_duty(Front_Left_Pin,0.2); // 80%
        }
       // Right
        else if(direction == RIGHT){
            PWM_duty(Front_Right_Pin,0.2); // 80%
            PWM_duty(Front_Left_Pin,0.5);  // 50%
        }
       // Left
        else if(direction == LEFT){
            PWM_duty(Front_Right_Pin,0.5);  // 50%
            PWM_duty(Front_Left_Pin,0.2);   // 80%
        }
       // Stop
        else if(direction == STOP){
            PWM_duty(Front_Right_Pin,1);
            PWM_duty(Front_Left_Pin,1);
        }
}

void Direction_display(uint8_t direction){
   btData = USART_read(USART1);

   if(btData == UP){
      USART_write(USART1, (uint8_t*) "UP   ", 5);
   }
   else if(btData == STOP){
      USART_write(USART1, (uint8_t*) "STOP ", 5);
   }
   else if(btData == RIGHT){
      USART_write(USART1, (uint8_t*)"RIGHT", 5);
   }
   else if(btData == LEFT){
      USART_write(USART1, (uint8_t*) "LEFT ", 5);
   }
     else if(btData == 'L'){
       btData = USART_read(USART1);
       if(btData == '0'){
          GPIO_write(GPIOA,5, 0);
          USART_write(USART1, (uint8_t*) "L0   ", 5);
       }
       else if(btData == '1'){
          GPIO_write(GPIOA,5, 1);
          USART_write(USART1, (uint8_t*) "L1   ", 5);
       }
   }

   USART_write(USART1, "\r\n", 1);
}
```

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

- To use USART 1, we set USART 1's environment at the 'void setup(void)'.

```
void setup(void)
{
   RCC_PLL_init();
   UART1_init();
   UART1_baud(BAUD_9600);

   ...

 }
```

Step 2. Check the Bluetooth connection by turning MCU's LED(LD2) On/OFF by sending text of "L0" or "L1" from PC.

- At the 'void Direction_display(uint8_t direction)' function, we show LED output to check whether bluetooth is connected with computer successfully or not.

```
else if(btData == 'L'){
       btData = USART_read(USART1);
       if(btData == '0'){
          GPIO_write(GPIOA,5, 0);
          USART_write(USART1, (uint8_t*) "L0", 2);
       }
       else if(btData == '1'){
          GPIO_write(GPIOA,5, 1);
          USART_write(USART1, (uint8_t*) "L1", 2);
       }
```

Especially to make LED off when 'L0' is entered and to make LED on when 'L1' is entered from computer, we check whether 'L ' is entered or not. If 'L' is entered, PC prepare to receive '0' or '1' additionally. Finally, if '0' is entered after 'L', LED will be off and if '1' is entered after 'L', LED will be on.

Step 3. Run 2 DC motors(Left-wheel, Right-wheel) to steer.

- To change DC motor car's direction, we set each Motors' speed differently.

```
void Direction(uint8_t direction){
          // Up
        if(direction == UP){
            PWM_duty(Front_Right_Pin,0.2); // 80%
            PWM_duty(Front_Left_Pin,0.2); // 80%
        }
       // Right
        else if(direction == RIGHT){
            PWM_duty(Front_Right_Pin,0.5); // 50%
            PWM_duty(Front_Left_Pin,0.2);  // 80%
        }
       // Left
        else if(direction == LEFT){
            PWM_duty(Front_Right_Pin,0.2);  // 80%
            PWM_duty(Front_Left_Pin,0.5);   // 50%
        }
       // Stop
```

```
        else if(direction == STOP){
                        PWM_duty(Front_Right_Pin,1);
            PWM_duty(Front_Left_Pin,1);
        }
}
```

First, to run forward, both motors are set up as 80% speed. Second, to run right, left wheel's speed is set up faster than right wheel's speed. Third, to run left, right wheel's speed is set up faster than left wheel's speed.

Result

Link: https://youtu.be/3wsc1a2efRQ

Reference

- Embeded Controller Gitbook: LAB: USART - LED, Bluetooth