실험 결과 보고서

:10주차 Bluetooth 동작 및 기판 납땜

실험날짜: 2024-11-05

제출날짜: 2024-11-10

실험자:3조(강태진, 김기윤, 김도완, 임성표)

목차

1. 실험 목적

• Bluetooth 모듈 (FB755AC) 를 이용한 스마트폰과의 통신
• 기판 납땜을 통해 보드와 모듈 연결

2. 세부 실험 내용

1) 만능 기판 납땜
2) PC의 putty 프로그램과 Bluetooth 모듈 간 통신이 가능하도록 펌웨어 작성
3) Bluetooth의 CONFIG_SELECT 핀에 3v3 준 상태에서 보드를 켠 후 putty에 설정 메뉴가 뜨면 강의 자료 참고하여 설정 변경
- name은 Tue_03 로 설정
4) 안드로이드의 Serial Bluetooth Terminal 어플리케이션을 이용하여 PC의 putty와 통신
- PC의 putty 입력 -> Bluetooth 모듈을 통해 스마트폰의 터미널에 출력
- 스마트폰의 터미널 입력 -> PC의 Putty에 출력

3. 실험 과정

(1) 배경 지식 이해

- 블루투스

1) 근거리 무선통신기술

2) 스마트폰, 무선 이어폰, 웨어러블 기기 등에서 디지털 데이터를 주고받는 기술

3) 2.4MHz ISM 주파수 대역 사용

4) 근거리, 저전력, 높은 신뢰성, 저가의 무선 통신 구현하는 것이 목표

- Identifier

1) SSID (Service Set Identifier)

• 무선랜을 통해 클라이언트가 접속할 때 각 무선랜을 구별하기 위한 고유 식별자

• Wi-Fi의 경우, 각 Wi-Fi 네트워크를 구별하기 위해 사용됨

2) UUID (Universally Unique Identifier)

• 네트워크 상에서 서로 다른 개체들을 구별하기 위한 128비트 고유 식별자

• 블루투스에서는 서비스의 종류를 구분하기 위해 사용됨

(2)코드 작성

1) RCC_Configure

- 라이브러리에서 제공하는 RCC_APB2PeriphClockCmd 함수를 이용하여 RCC에서 Clock을

인가해주기 위해 필요한 GPIOA와 USART, AFIO를 Enable 시켜준다.

```
void RCC_Configure(void)
{
    // TODO: Enable the APB2 peripheral clock using the function 'RCC_APB2PeriphClockCmd'

    /* USART1, USART2 TX/RX port clock enable */

    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);

    /* USART1, USART2 clock enable */
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1,ENABLE);
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_USART2,ENABLE);
    /* Alternate Function IO clock enable */
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_AFIO, ENABLE);
}
```

2) USART(TX,RX)

- GPIO_InitTypeDef 구조체를 사용하여 GPIO 핀 설정을 구성

- USART1 설정:

GPIOA 포트를 사용하여 USART1의 TX 핀과 RX 핀을 설정

TX 핀 (GPIO_Pin_9)은 50MHz의 GPIO 속도와 "Alternative Function Push-Pull" 모드

(GPIO_Mode_AF_PP)로 설정. 이 모드는 USART 데이터를 송신하는 데 사용됨

RX 핀 (GPIO_Pin_10)은 50MHz의 GPIO 속도와 "Input Pull-up" 모드 (GPIO_Mode_IPU)로

설정됨. 이 모드는 USART 데이터를 수신하는 데 사용됨

- USART2 설정:

USART2도 마찬가지로 TX 핀 (GPIO_Pin_2) 및 RX 핀 (GPIO_Pin_3)을 동일한 방식으로 설정

```
void GPIO_Configure(void)
{
    GPIO_InitTypeDef GPIO_InitStructure;

    // TODO: Initialize the GPIO pins using the structure 'GPIO_InitTypeDef' and the function 'GPIO_Init'
    /* USART1 pin setting */

    //TX
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_9;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    //RX
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
    GPIO_Init(GPIOA, &GPIO_InitStructure);


    /* USART2 pin setting */
    //TX
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_2;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    //RX
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_3;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
    GPIO_Init(GPIOA, &GPIO_InitStructure);


}
```

3) USART1_Init

- USART1 모듈 활성화: USART_Cmd(USART1, ENABLE)을 사용하여 USART1 모듈을 활성화

이렇게 하면 USART1을 사용할 수 있음

- USART1 초기화: USART_InitTypeDef 구조체를 사용하여 USART1의 초기화 설정을 구성

다음과 같은 설정을 적용:

통신 속도: 9600

데이터 비트 길이: 8비트

정지 비트: 1비트

패리티 비트: 사용하지 않음

모드: 송신(Tx) 및 수신(Rx) 활성화

- USART1 RX 인터럽트 활성화: USART_ITConfig(USART1, USART_IT_RXNE, ENABLE)를 사용하여

USART1의 수신 데이터 레지스터가 비어있을 때 인터럽트를 활성화함

```
//Week09 참조
void USART1_Init(void)
{
    USART_InitTypeDef USART1_InitStructure;

    // Enable the USART1 peripheral
    USART_Cmd(USART1, ENABLE);

    // TODO: Initialize the USART using the structure 'USART_InitTypeDef' and the function 'USART_Init'

    USART1_InitStructure.USART_BaudRate = 9600;
    USART1_InitStructure.USART_WordLength = USART_WordLength_8b;
    USART1_InitStructure.USART_StopBits =USART_StopBits_1;
    USART1_InitStructure.USART_Parity =USART_Parity_No;
    USART1_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
    USART1_InitStructure.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
    USART_Init(USART1, &USART1_InitStructure);


    // TODO: Enable the USART1 RX interrupts using the function 'USART_ITConfig' and the argument value 'Receive Data register not empty interrupt'.
     USART_ITConfig(USART1, USART_IT_RXNE, ENABLE);


}
```

4) USART2_Init

- USART1_Init 과 마찬가지로 설정

5) NVIC_Configure

- NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2)를 사용하여 NVIC 그룹을 설정

이 그룹 설정은 인터럽트 우선순위 그룹을 구성하는 데 사용됨

- USART1 인터럽트 설정:

NVIC_EnableIRQ(USART1_IRQn)은 USART1 인터럽트를 활성화.

NVIC_InitStructure.NVIC_IRQChannel = USART1_IRQn는 USART1의 인터럽트 채널을 지정

NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0은 USART1 인터럽트의 선점 우선순위를 0으로 설정

NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0은 USART1 인터럽트의 서브 우선순위를

0으로 설정

NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE은 USART1 인터럽트를 활성화

- USART2 인터럽트 설정:

NVIC_EnableIRQ(USART2_IRQn)은 USART2 인터럽트를 활성화

NVIC_InitStructure.NVIC_IRQChannel = USART2_IRQn는 USART2의 인터럽트 채널을 지정

NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 1은 USART2 인터럽트의 선점 우선순위를 1로 설정

NVIC_InitStructure.NVIC_IRQChannelSubPriority = 1은 USART2 인터럽트의 서브 우선순위를

1로 설정

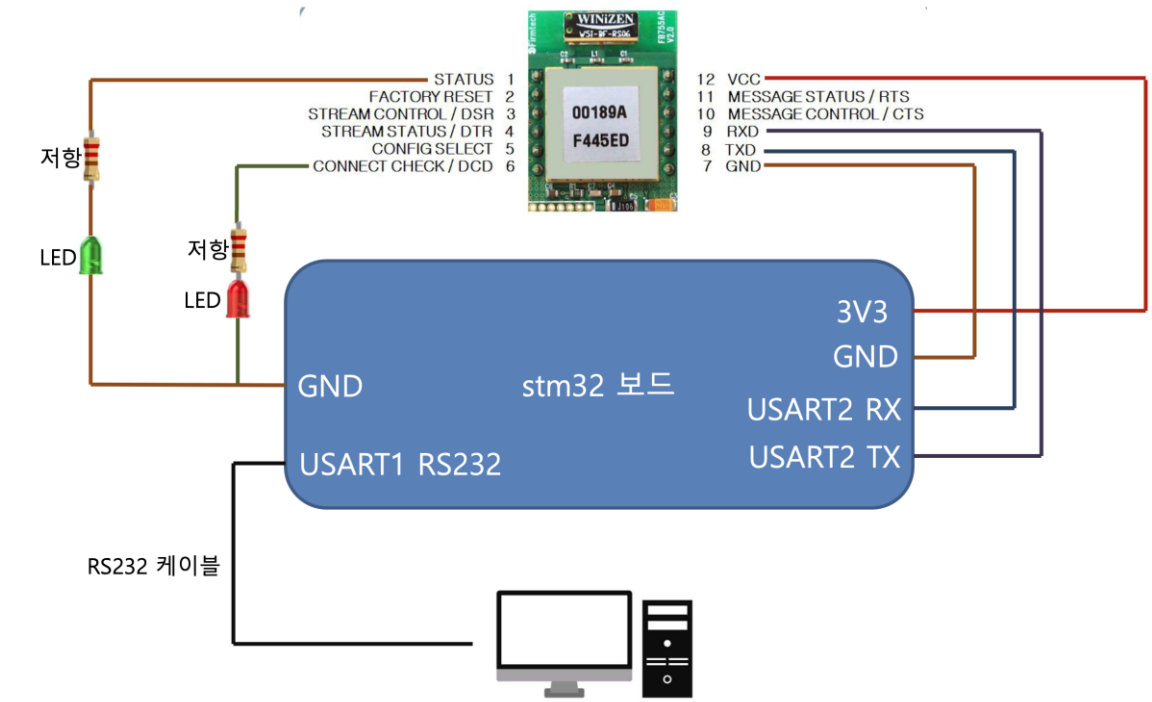
NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE은 USART2 인터럽트를 활성화

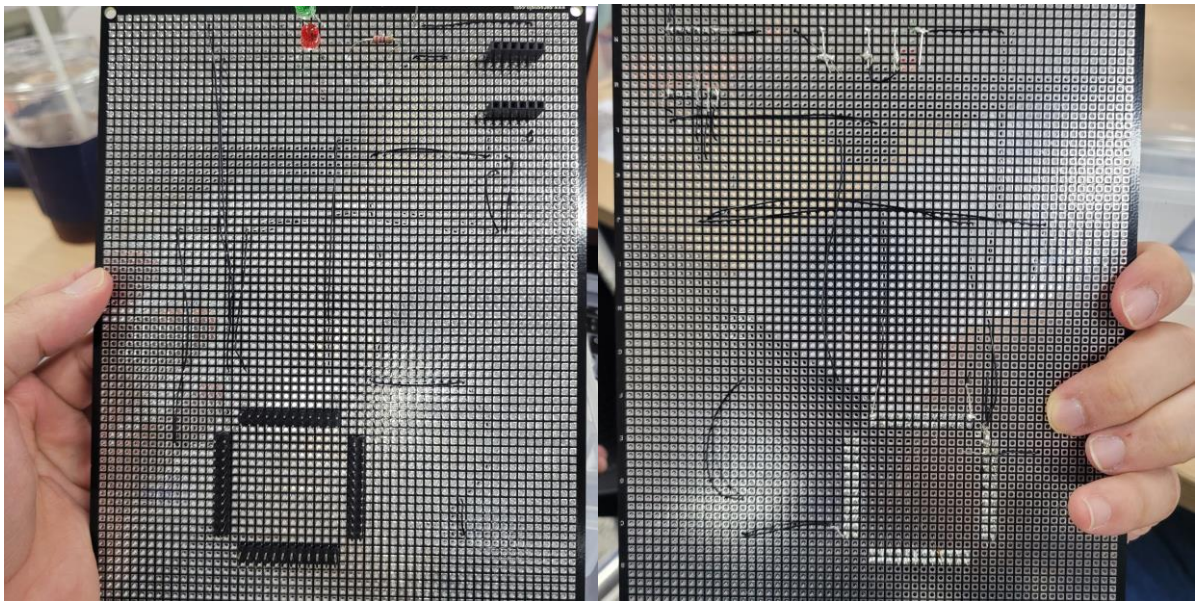
```c
//Week09 슬라이드 참조
void NVIC_Configure(void) {

    NVIC_InitTypeDef NVIC_InitStructure;

    //TODO: fill the arg you want
    //misc.c 참조
    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2);

    // USART1
    // 'NVIC_EnableIRQ' is only required for USART setting
    NVIC_EnableIRQ(USART1_IRQn);
    NVIC_InitStructure.NVIC_IRQChannel = USART1_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0; // TODO
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0; // TODO
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);

    // USART2
    // 'NVIC_EnableIRQ' is only required for USART setting
    NVIC_EnableIRQ(USART2_IRQn);
    NVIC_InitStructure.NVIC_IRQChannel = USART2_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 1; // TODO
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 1; // TODO
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);
}
```

6) USARTn_IRQHandler - USART Handler 함수를 통해서 각 핀에 해당하는 입력을 받으면 Interrupt가 발생하여 동작을 수행한다.

```c
void USART1_IRQHandler() {
    uint16_t word;
    if(USART_GetITStatus(USART1,USART_IT_RXNE)!=RESET)
    {
        // the most recent received data by the USART1 peripheral
        word = USART_ReceiveData(USART1);

        // TODO implement
        //sendDataUART1(word);-----------------------------------------------
        //Week10 슬라이드 14p 참조
        USART_SendData(USART2,word);
        // clear 'Read data register not empty' flag
        USART_ClearITPendingBit(USART1,USART_IT_RXNE);
    }
}

void USART2_IRQHandler() {
    uint16_t word;
    if(USART_GetITStatus(USART2,USART_IT_RXNE)!=RESET){
        // the most recent received data by the USART2 peripheral
        word = USART_ReceiveData(USART2);

        // TODO implement
        //week09 참조
        USART_SendData(USART1,word);

        // clear 'Read data register not empty' flag
        USART_ClearITPendingBit(USART2,USART_IT_RXNE);
    }
}
```

(3)기판 납땜





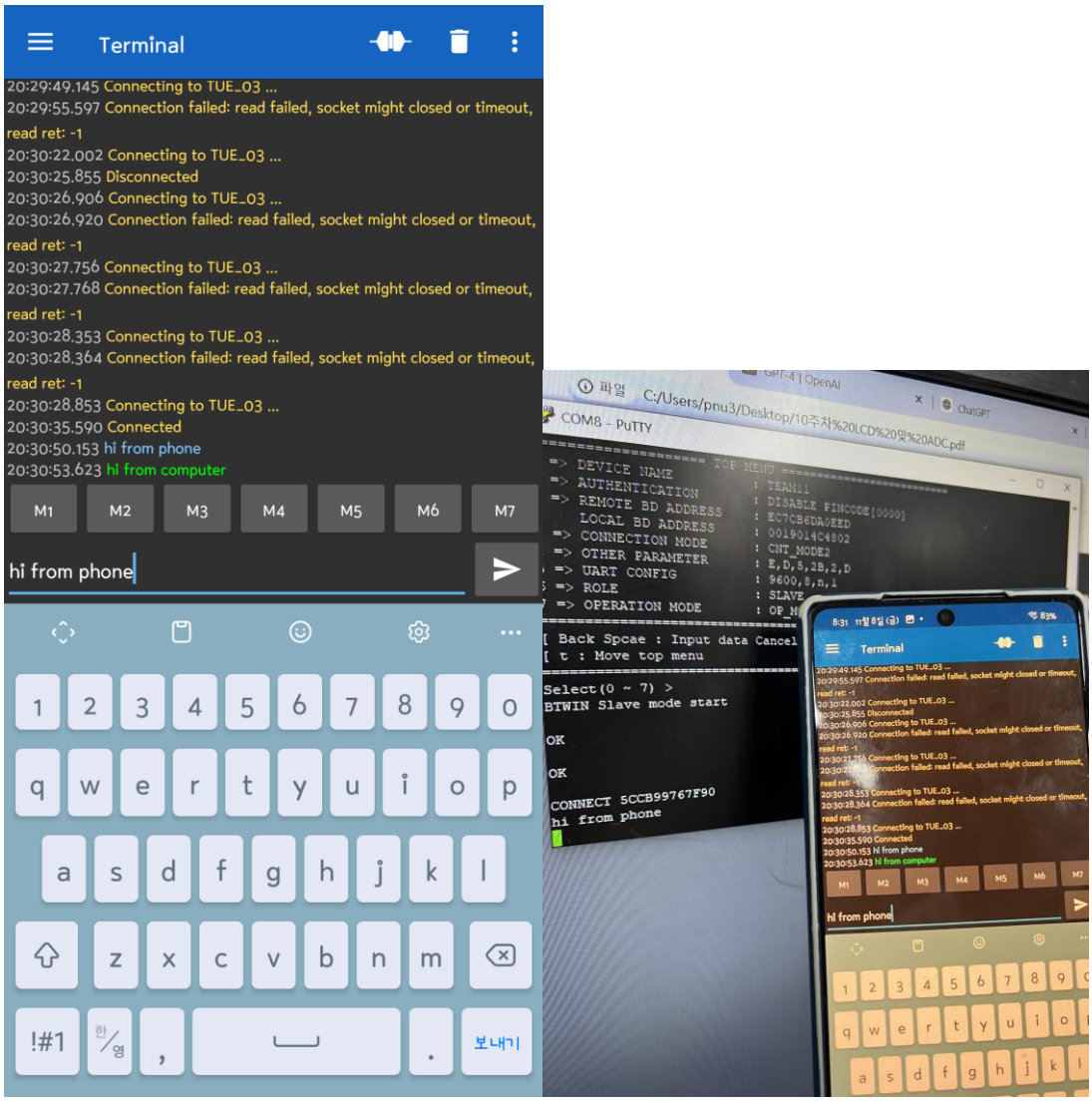
위 회로도를 보고 납땜을 한 앞면과 뒷면

(4)블루투스 설정 부분

Bluetooth의 CONFIG_SELECT 핀에 3.3v 준 상태로 보드를 켠 후 putty에 설정 메뉴가 뜨면 설정

변경

4. 실험 결과



Phone ->PC와 PC -> Phone 데이터 전송이 성공적으로 이루어진 모습

5. 참고 문헌

(강의 자료) 10주차 Bluetooth.pdf