임베디드시스템 설계 및 실험 보고서

[002 분반 - 2 조 - 9 주차]

조원	202055531 김후겸 202055584 이태경 202155540 김채현 202255535 김진우

실험날짜	2024-10-30

1. 실험 주제

- Bluetooth 동작 및 납땜

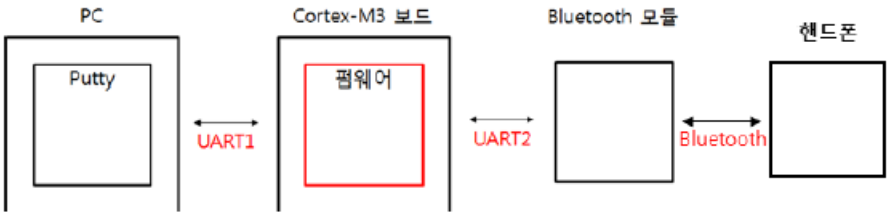
2. 실험 목적

- Bluetooth 모듈 (FB755AC) 를 이용한 스마트폰과의 통신레지스터와 주소 제어를 통한 임베디드 펌웨어 개발 이해
- 기판 납땜을 통해 보드와 모듈 연결

3. 세부 실험 목적

1. UART1 을 통해 Putty 의 데이터를 수신하면 바로 UART2 를 통해 Bluetooth 모듈로 전송

2. UART2 를 통해 Bluetooth 모듈의 데이터를 수신하면 바로 UART1 을 통해 Putty 로 전송



3. 블루투스 모듈의 CONFIG SELECT 에 3v3 을 입력한 상태로 보드 전원을 껐다 켜면 Putty 설정 모드가 시작

4. 블루투스의 설정을 변경하고 AT 명령어를 이용해 스마트폰과 블루투스를 연결하여 PC 와 통신

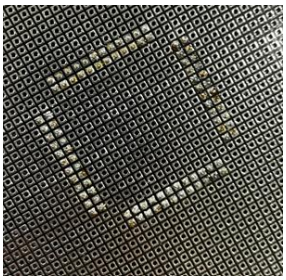
5. 스마트폰 또는 PC 에 입력한 값이 PC 또는 스마트폰에 출력

4. 실험 장비

- Bluetooth 모듈 (FB755AC)
- 만능기판, 납땜기 등

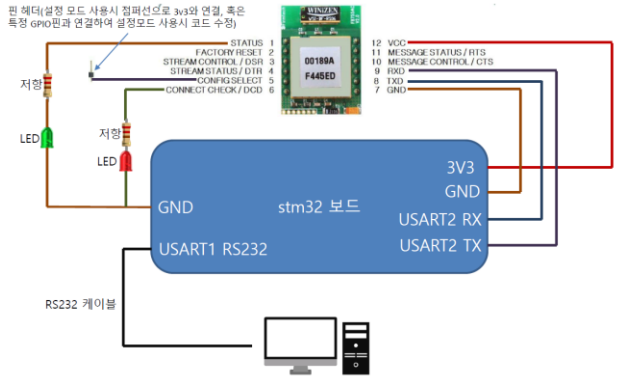
5. 실험 과정

1. stm32 보드의 핀 모양을 만능기판에 그대로 복사해서 납땜을 진행한다.



2. 블루투스 모듈과 블루투스 모듈의 동작 유무를 확인할 수 있는 LED 도 만능기판에 납땜을 해준다.

3. 아래의 사진과 같이 연결되도록 만능기판에 납땜을 해준다.



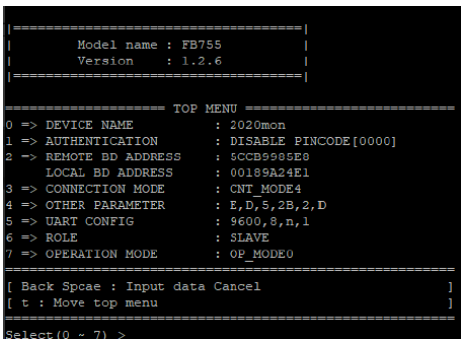
4. USART1 RS232 는 RS232 케이블로 컴퓨터와 연결한다.

5. 작성한 코드를 업로드를 한 후 보드를 껐다 키면 putty 에서 설정모드가 작동한다.

6. selection 핀을 3.3v 와 연결하고 configuration mode 설정을 한다.

이때 selection pin 은 나중에 연결을 해제해야 한다.

블루투스의 이름(WED_02), Pincode(비밀번호), Connection mode 4 slave, Uart config 를 설정한다.



이 과정의 경우 사진을 찍지 못해 ppt 에 제공된 사진으로 대체

selection 의 3.3v 입력을 해제하고 보드 전원을 껐다 키면 "AT+BTSCAN" 커맨드를 입력해 연결 대기 돌입한다. (이때, putty 창에는 글자가 출력이 안될 수도 있지만, enter 를 했을 때 OK 라고 나오면 성공한 것이다.

7. 스마트폰에 "Serial Bluetooth Terminal"을 설치한 후 블루투스 연결해 입출력이 잘 되는지 확인한다.

[코드 설명] – 지난주와 거의 비슷한 맥락이다

```
void RCC_Configure(void)
{
    // TODO: Enable the APB2 peripheral clock using the function 'RCC_APB2PeriphClockCmd'
    /* USART1 Tx(PA9)/Rx(PA10), USART2 Tx(PA2)/Rx(PA3) port clock enable */
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);

    /* USART1, USART2 clock enable */
    /* USART1 */
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1, ENABLE);
    /* USART2 */
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_USART2, ENABLE);

    /* Alternate Function IO clock enable */
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_AFIO, ENABLE);
}
```

RCC 설정 코드- USART1, 2 모두 통신 가능하게 해야 한다.

```c
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(PA9)
    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(PA10)
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    /* USART2 pin setting */
    //Tx(PA2)
    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(PA3)
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_3;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
    GPIO_Init(GPIOA, &GPIO_InitStructure);
}
```

GPIO 설정 코드 - USART1, 2 에 대한 RX, TX 를 가능하게 하는 코드이다.

```c
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_HardwareFlowControl = USART_HardwareFlowControl_None;
    USART1_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
    USART1_InitStructure.USART_Parity = USART_Parity_No;
    USART1_InitStructure.USART_StopBits = USART_StopBits_1;
    USART1_InitStructure.USART_WordLength = USART_WordLength_8b;
    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);
}
```

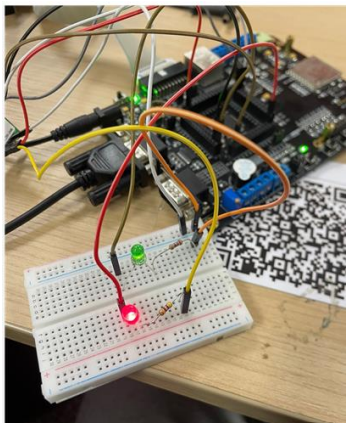
USART 초기화 코드 - USART 에 대한 초기 설정을 하는 코드이다.(USART1,2 동일한 흐름을 가진다)

```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

    USART_SendData(USART2, word);

        // clear 'Read data register not empty' flag
        USART_ClearITPendingBit(USART1,USART_IT_RXNE);
    }
}
```

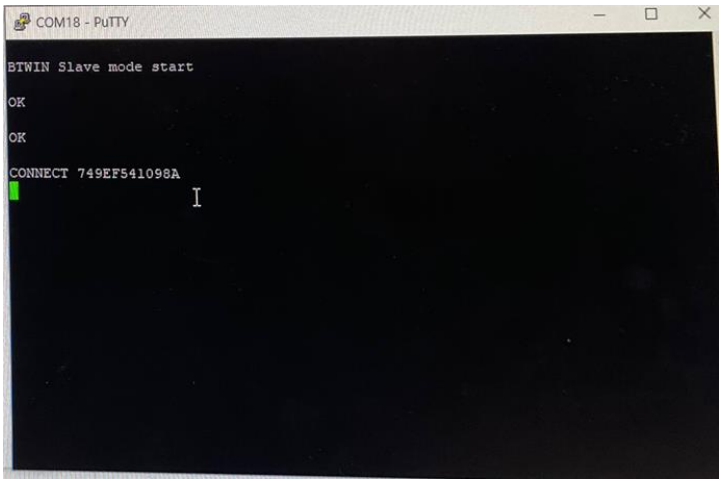
USART1 인터럽트 핸들러 코드 - USART 에 대한 인터럽트를 handling 하는 코드이다.
USART1 으로 데이터를 수신하고 USART2 로 데이터를 송신하는 코드이다.

USART2 에 대한 코드는 이와는 반대로 구성하면 된다.

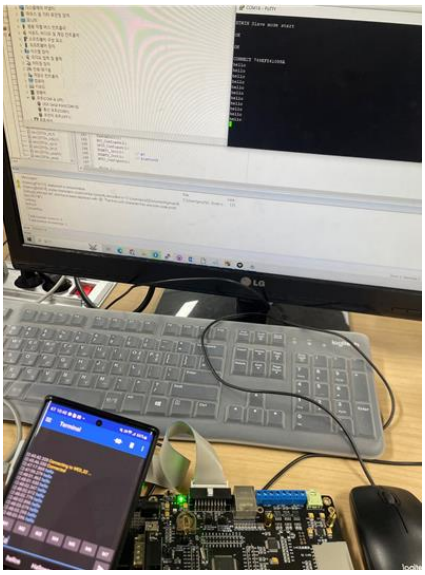
6. 실험 결과



블루투스 모듈이 정상 작동하는지 확인(납땜 전에 진행)



제대로 연결이 된 것을 확인할 수 있다.



스마트폰과 연결이 잘 된 것을 확인할 수 있다

7. 분석 및 결론

이번 실험에서는 USART1 과 USART2 를 통해 PC 와 스마트폰을 Bluetooth 통신하는 과정을 진행했다. 블루투스를 연결하는 것은 빠르게 진행되었으나, 납땜하는 과정이 낯설어 시간이 많이 걸렸다. 처음에는 납땜이 완성되기를 계속 기다렸는데, 비효율적이라고 생각해 납땜은 ppt 에 제공된 사진대로 진행하고, 코드가 제대로 작성되었는지를 확인하기 위해 브래드보드와 케이블을 사용해 블루투스 모듈을 따로 연결하여 putty 를 실행시켜보았다. 납땜의 경우 잘못 납땜한 부분을 사전에 방지하기 위해 계속 전류가 흐르는지 확인하며 진행하였다. 이전의 실험에 비해 시간이 많이 들었지만 이번 실험을 통해 추후 진행될 텀 프로젝트에서 납땜 및 블루투스 모듈 사용을 할 수 있을 것 같다.