

## [2조] 7주차 결과보고서

---



Subject	임베디드 시스템 설계 및 실험
Professor	김원석
Major	정보컴퓨터공학부
Date	2022.10.31
Team Member	202055600 정홍빈
	201924617 께앗띠퐁 유옢
	201824534 윤상호
	201824636 이강우



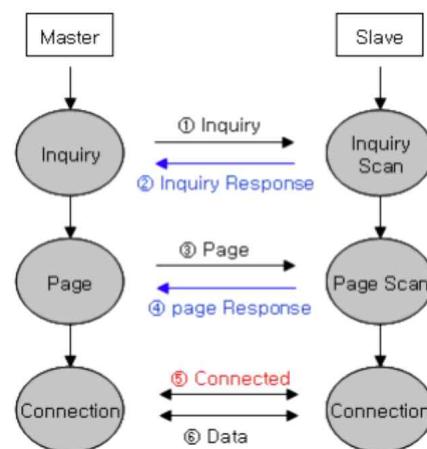
## 1. 실험 목적

블루투스 통신 기술을 활용한 핸드폰과의 양방향 통신과 납땜 기술 익히기

## 2. 실험 원리 및 이론

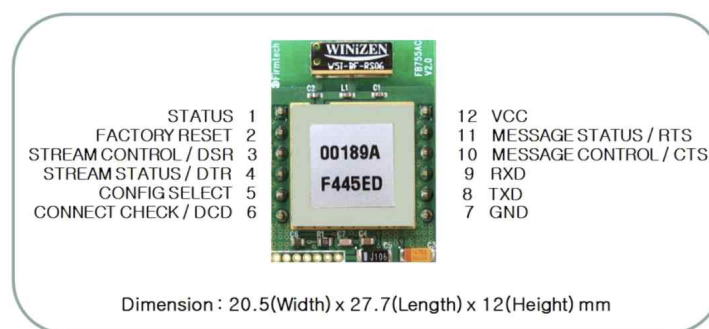
### 1) 블루투스

근거리 무선통신 기술, 스마트폰, 무선이어폰, 웨어러블 기기에서 데이터를 주고받는 기술.



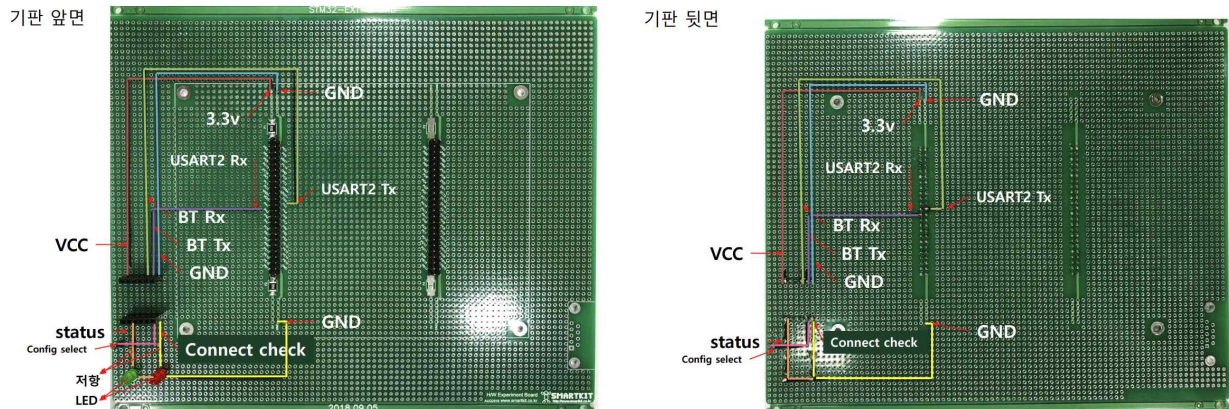
[그림-1] 블루투스 통신 과정

MASTER와 SLAVE로 동작하며 MASTER가 Inquiry 나 Page 요청을 하면 Slave가 각각의 요구사항을 Scan 하는 방식으로 통신한다.



[그림-2] 블루투스 모듈

실험은 FB755AC 모듈을 사용하며 Bluetooth v2.1을 지원하고 최대 1:7 연결이 가능하며 AT 명령어를 이용해 조작할 수 있다. 고유의 용도가 존재하는 핀이 있고, 이 핀에 각각의 용도에 맞는 연결을 하면 블루투스 모듈이 동작한다.

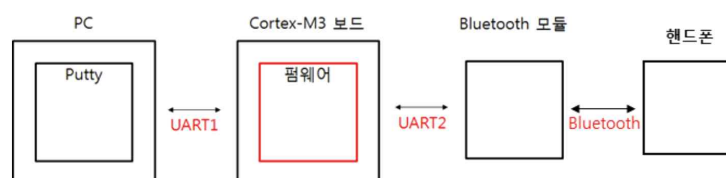


[그림-3] 납땜이 완료된 만능기판

블루투스 모듈과 STM32보드가 장착될 만능기판으로 인두와 실 납을 이용해 제작한다. 주의사항으로는 전선을 느슨하거나 사선으로 납땜하지 말고, 수직으로 팽팽하게 해주어야 한다.

### 3. 실습 진행 과정

이번 실험은 남땜과 코드를 짜는 분야가 나누어진다. 코드 쪽 실험과정은 [그림-4]와 같이 이루어진다.



[그림-4] 실험 과정

저번 실험에서 사용했었던 UART1 통신을 이용해 PC의 Putty 데이터를 전송받으면 블루투스 모듈로 UART2로 전송하고 반대로 핸드폰에서 모듈로 전송받으면 역으로 PC의 Putty로 데이터를 전송받는 양방향 통신이 이루어진다.

### [Source Code]

```
void RCC_Configure(void)
{
    // TODO: Enable the APB2 peripheral clock using the function 'RCC_APB2PeriphClockCmd'
    // USART2_TX : PA2, USART2_RX : PA3
    /* UART TX/RX port clock enable */
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);
    /* USART1 clock enable */
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1, ENABLE);
    /* USART2 clock enable */
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_USART2, ENABLE);
    /* Alternate Function IO clock enable */
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_AFIO, ENABLE);
}
```

RCC\_Configure함수를 이용해 실험에서 사용하는 TX, RX, USART1, USART2의 Clock을 활성화한다.

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

    // TODO: Initialize the GPIO pins using the structure 'GPIO_InitTypeDef'
    // and the function 'GPIO_Init'

    /* UART pin setting */
    //TX 9 mode 01 open-drain
    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 10 mode 00 push-pull
    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 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_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
    GPIO_Init(GPIOA, &GPIO_InitStructure);
}
```

GPIO Configuration을 통해 사용할 핀 (PortA 2번, 3번, 9번, 10번)과 상응하는 상태를 설정한다.

```

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);
}

void USART2_Init(void){
    USART_InitTypeDef USART2_InitStructure;

    // Enable the USART2 peripheral
    USART_Cmd(USART2, ENABLE);

    // TODO: Initialize the USART using the structure 'USART_InitTypeDef' and the function 'USART_Init'
    USART2_InitStructure.USART_BaudRate = 9600;
    USART2_InitStructure.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
    USART2_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
    USART2_InitStructure.USART_Parity = USART_Parity_No;
    USART2_InitStructure.USART_StopBits = USART_StopBits_1;
    USART2_InitStructure.USART_WordLength = USART_WordLength_8b;
    USART_Init(USART2, &USART2_InitStructure);

    // TODO: Enable the USART2 RX interrupts using the function 'USART_ITConfig' and the argument value 'Receive Data register not empty interrupt'
    USART_ITConfig(USART2, USART_IT_RXNE, ENABLE);
}

```

USART1과 2를 BaudRate 9,600과 MODE, Parity bit, Stopbit, WordLength등을 저번실험과 마찬가지로 초기화해준다.

```

void sendDataUART1(uint16_t data) {
    USART_SendData(USART1, data);
}

void sendDataUART2(uint16_t data) {
    USART_SendData(USART2, data);
}

```

일련의 양방향 통신 과정이 빠르게 이루어지려면 UART의 SendData() 함수의 TC 데이터를 기다리는 While 문을 삭제해준다.

```

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);

        sendDataUART2(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);

        sendDataUART1(word);

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

```

각각의 IRQHandler 함수를 이용해 인터럽트를 통해 Data가 들어오면 양방향 통신을 구현하기 위해 [그림-4]와 같이 근접한 USART로 바로 전송할 수 있게 함수를 작성한다.

```

void NVIC_Configure(void) {

    NVIC_InitTypeDef NVIC_InitStructure;

    // TODO: fill the arg you want
    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_0);

    // TODO: Initialize the NVIC using the structure 'NVIC_InitTypeDef' and the function 'NVIC_Init'

    // UART1
    // '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 = 4; // TODO
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);

    // UART2
    // 'NVIC_EnableIRQ' is only required for USART setting

    NVIC_EnableIRQ(USART2_IRQn);
    NVIC_InitStructure.NVIC_IRQChannel = USART2_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0; // TODO
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 4; // TODO
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);

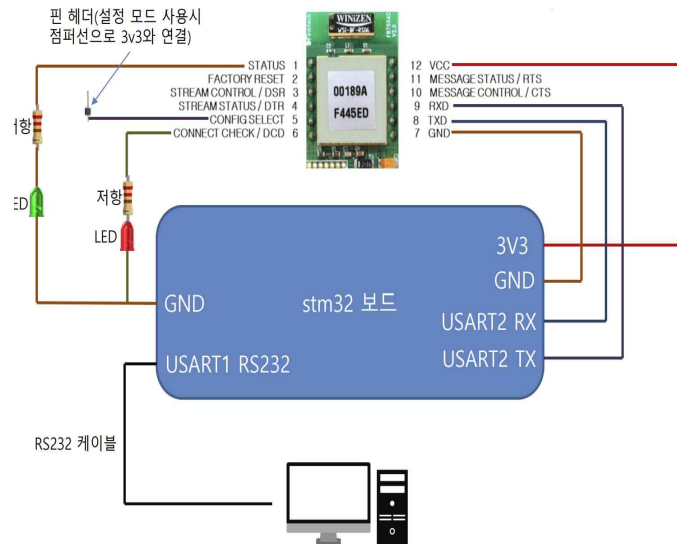
}

```

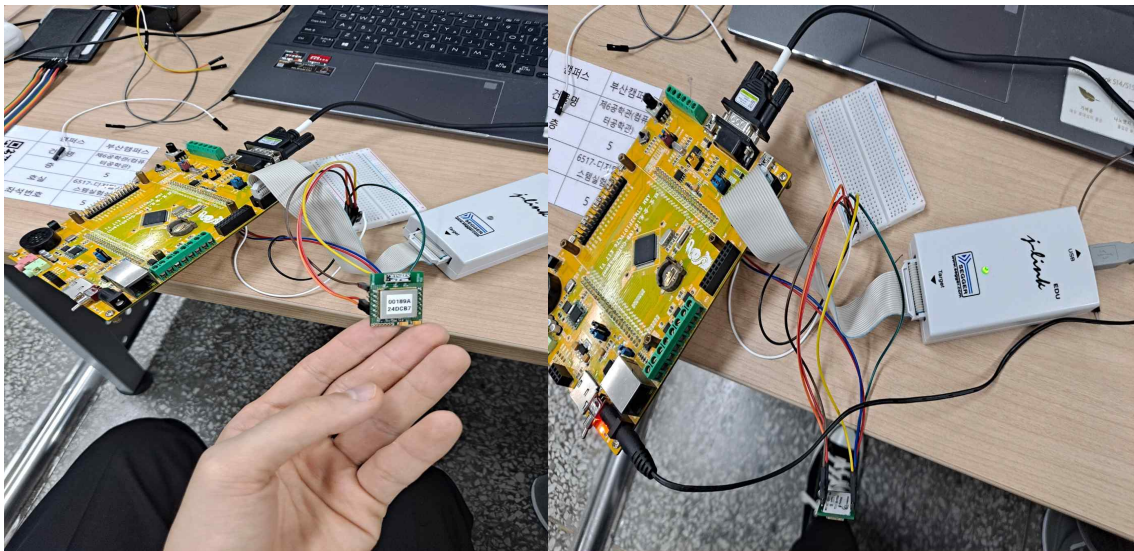
NVIC configure은 우선순위를 설정해주는 함수로, 이번 실험에서 UART의 우선순위는 지정할 필요가 없으므로, 같은 우선순위로 설정해주었다.



### [블루투스 모듈 설정]



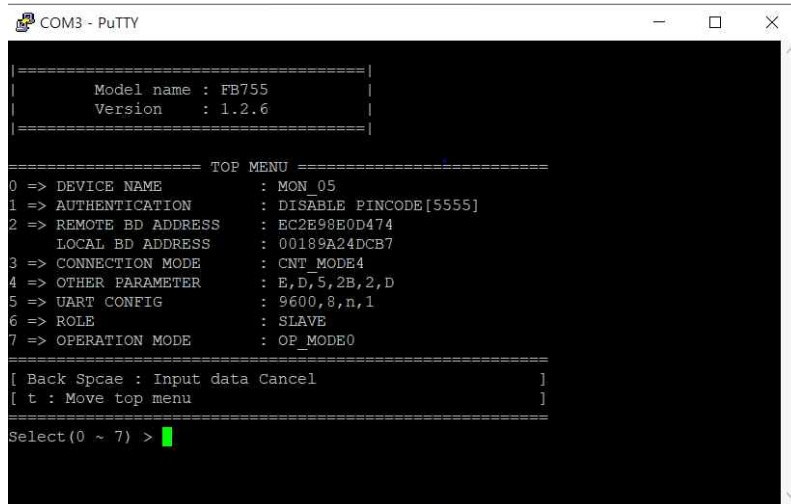
[그림-5] 블루투스 모듈 연결(이론)



[그림-6] 블루투스 모듈 STM32보드 연결 (실습)

[그림-5]와 같이 STM32보드와 블루투스 모듈을 연결해 준다. 실습에서는 STM32보드에 연결할 전선이 부족해 브레드보드를 이용하였다. 첫 시도에 연결은 잘했지만 블루투스 모듈이 불량품이라 동작하지 않았다. 조교님이 정상 제품으로 교환하고 연결하니 정상적으로 동작하였다.

#### 4. 실험결과

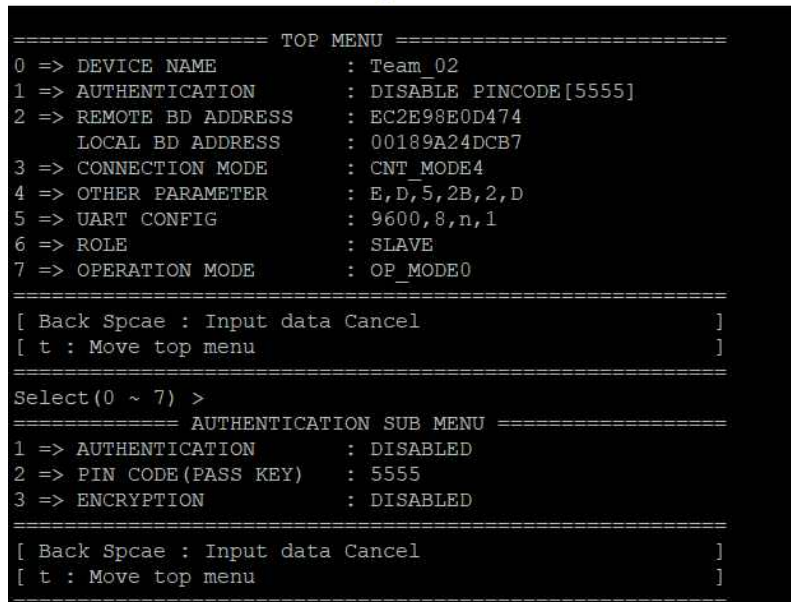


```

=====
|          Model name : FB755          |
|          Version   : 1.2.6          |
|=====|
===== TOP MENU =====
0 => DEVICE NAME       : MON_05
1 => AUTHENTICATION    : DISABLE PINCODE[5555]
2 => REMOTE BD ADDRESS : EC2E98E0D474
      LOCAL BD ADDRESS : 00189A24DCB7
3 => CONNECTION MODE   : CNT_MODE4
4 => OTHER PARAMETER    : E,D,5,2B,2,D
5 => UART CONFIG       : 9600,8,n,1
6 => ROLE              : SLAVE
7 => OPERATION MODE     : OP_MODE0
=====
[ Back Spcae : Input data Cancel      ]
[ t : Move top menu                    ]
=====
Select(0 ~ 7) > █
  
```

[그림-7] Putty 초기화면

Config SELECT 에 3v3을 연결 후 전원을 껐다 키면 [그림-7]과 같이 화면이 출력된다.



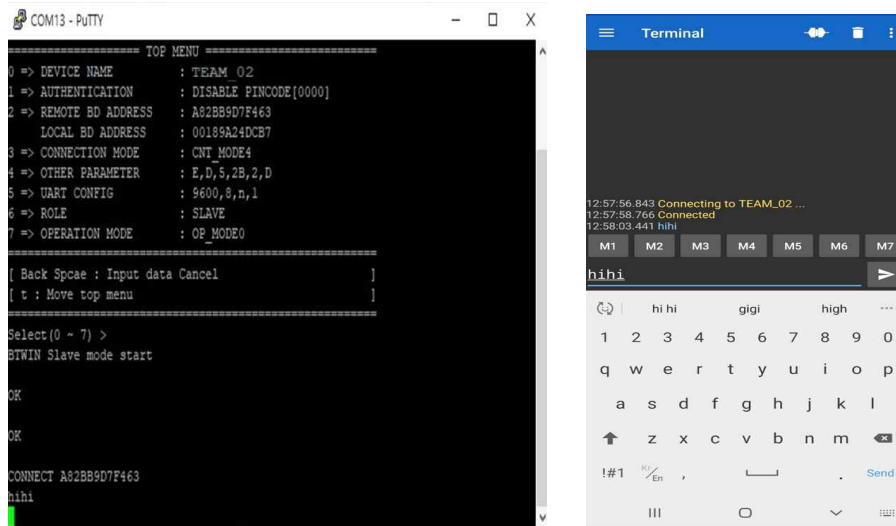
```

===== TOP MENU =====
0 => DEVICE NAME       : Team_02
1 => AUTHENTICATION    : DISABLE PINCODE[5555]
2 => REMOTE BD ADDRESS : EC2E98E0D474
      LOCAL BD ADDRESS : 00189A24DCB7
3 => CONNECTION MODE   : CNT_MODE4
4 => OTHER PARAMETER    : E,D,5,2B,2,D
5 => UART CONFIG       : 9600,8,n,1
6 => ROLE              : SLAVE
7 => OPERATION MODE     : OP_MODE0
=====
[ Back Spcae : Input data Cancel      ]
[ t : Move top menu                    ]
=====
Select(0 ~ 7) >
===== AUTHENTICATION SUB MENU =====
1 => AUTHENTICATION    : DISABLED
2 => PIN CODE(PASS KEY) : 5555
3 => ENCRYPTION        : DISABLED
=====
[ Back Spcae : Input data Cancel      ]
[ t : Move top menu                    ]
=====
  
```

[그림-8] Putty 초기 설정 완료

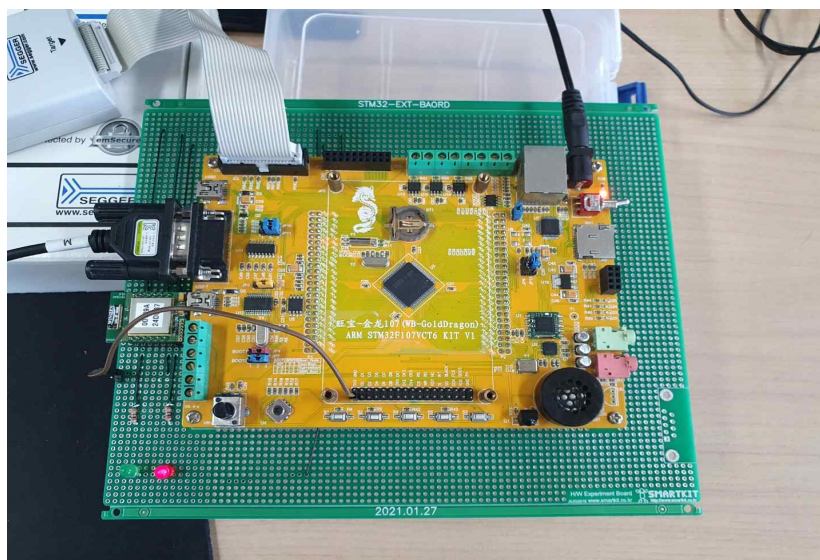
Device name, 블루투스 연결 비밀번호에 쓰일 Pincode, Connection mode와 Uart config를 PPT에 나와 있는 대로 설정한다.





[그림-9] 블루투스 모듈 연결 후 양방향 통신

AT- BTSCAN모드로 돌입하면 스마트폰과 블루투스 연결 후 휴대폰과 Putty로 서로 통신이 가능하다.



[그림-10] 기판과 결합한 블루투스 모듈과 STM32보드

### [결론]

이번 실험에서 코드를 작성하는 데는 저번 실험의 코드를 일부 쓰기도 하고 개념 자체가 어렵지 않아 그렇게 오래 걸리지 않았는데, 블루투스 모듈과 STM32보드를 연결하는 데 모듈의 문제가 있어 해결하는 데 시간이 걸렸고, 컴퓨터 전공자에게는 생소할 수 있는 납땜 과정이 있어 납땀하는 데 애를 먹었다. 휴대전화나 노트북을 이용해 블루투스 이어폰이나 블루투스 마우스를 연결하는 것을 당연하게만 생각해 왔는데, 직접 블루투스와 접속을 하고 통신해보니 실제 생활에도 응용할 수 있겠다는 생각이 들었다. 팀 프로젝트 과제에서 유용하게 쓸 수 있다고 본다.