

특정 소출력 무선기기: (다채널 무선송수신 IO보드)

nMJ447RTX_CONT

Users' Guide

Ver 1.3

<이 컨트롤보드와 같이 사용되는 모듈은 아래의 인증을 가진 모듈입니다.>



기기명 : 특정소출력 무선기기(데이터전송용 무선기기)
상호명 : 라디오리써치
모델명 : nMJ447RTX
제조연월 : 2023년 9월 18일
제조자 : 라디오리써치
제조국 : 한 국
인증번호 : R-C-rad-nMJ447RTX



기기명 : 특정소출력 무선기기(데이터전송용 무선기기)
상호명 : 라디오리써치
모델명 : nHT447RTX
제조연월 : 2023년 9월 18일
제조자 : 라디오리써치
제조국 : 한 국
인증번호 : R-C-rad-nMJ447RTX

INDEX

1. 제품의 형상	3
2. 기계적 규격 (nMJ447RTX_CONT)	4
3. 전원공급하기 및 기본동작	5
4. 시험해보기	6
5. 패킷통신	12
5-1. 패킷(U.A.R.T)의 데이터	12
5-2. 패킷의 송수신 방법	12
5-3. 무선전송속도	12
5-4. 패킷전송의 전송시간의 예	13
5-5. 패킷 전송의 방법	14
6. 아두이노를 이용하여 동작시키기	19
7. 수신신호강도(R.S.S.I)	28
8. 주의사항	29

nMJ447RTX_CONT

1. 제품의 형상



[MJ447RTX_CONT]

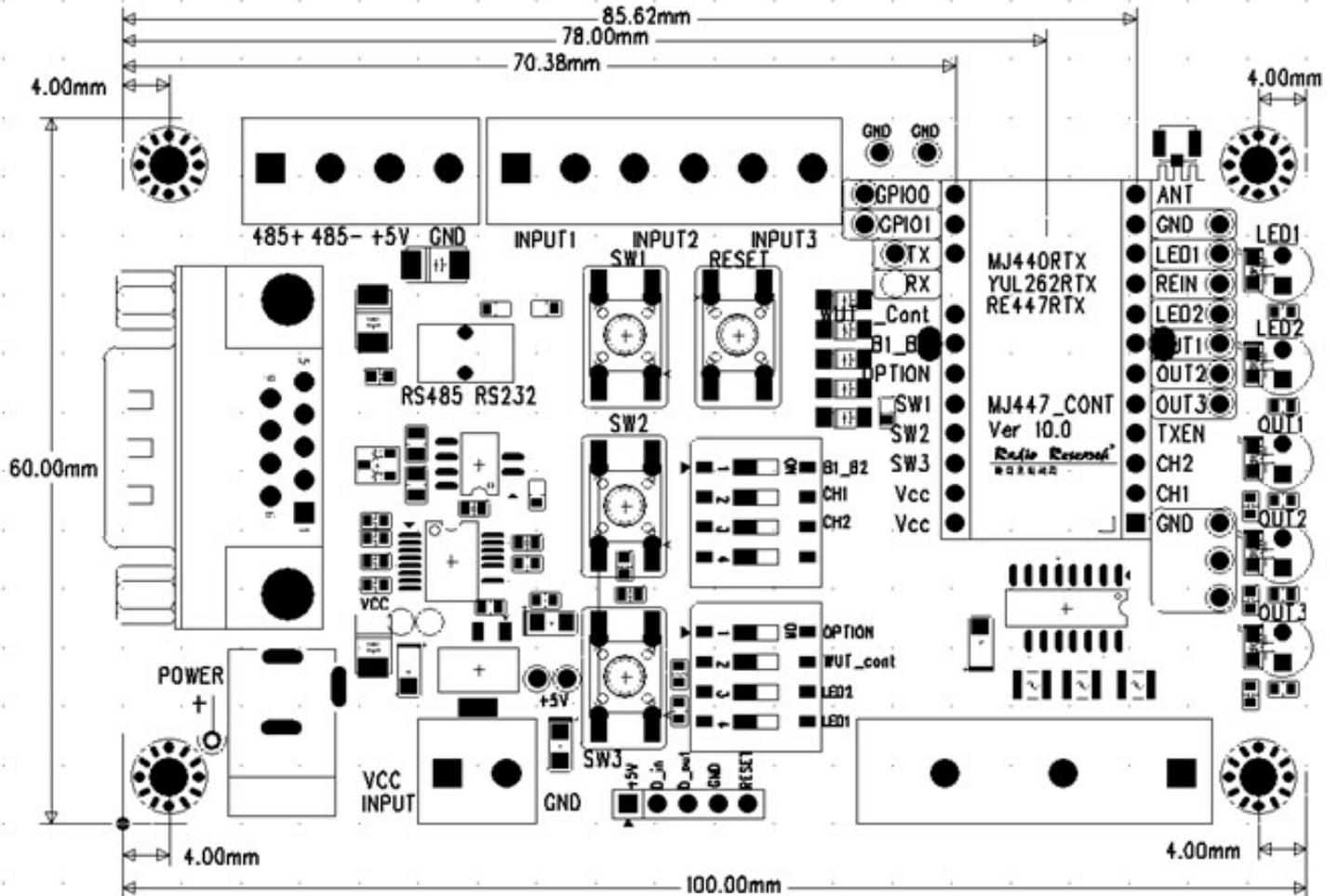


[모듈과 안테나 결합한 사진]

nMJ447RTX_CONT

2. 기계적 규격

1.(nMJ447RTX_CONT)



[입출력보드(IO) 설계지도]

3. 전원공급하기 및 기본동작

3-1. nMJ447RTX를 컨트롤보드(이하 MJ447CONT라 함) 와 결합한다.

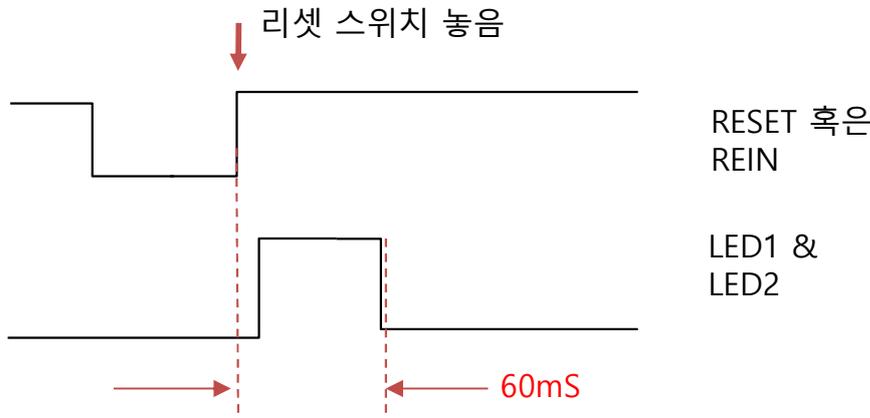


[nMJ447RTX 와 MJ447CONT 가 결합된 사진]

3-2.

좌측하단의 잭이나 전원 공급용 터미널 블록에 전원을 공급한다. 이 때 전원은 +12V 혹은 +24V이다. (릴레이 혹은 아답터 사용을 위해 사용전압을 +12V혹은 +24V로 표기함)

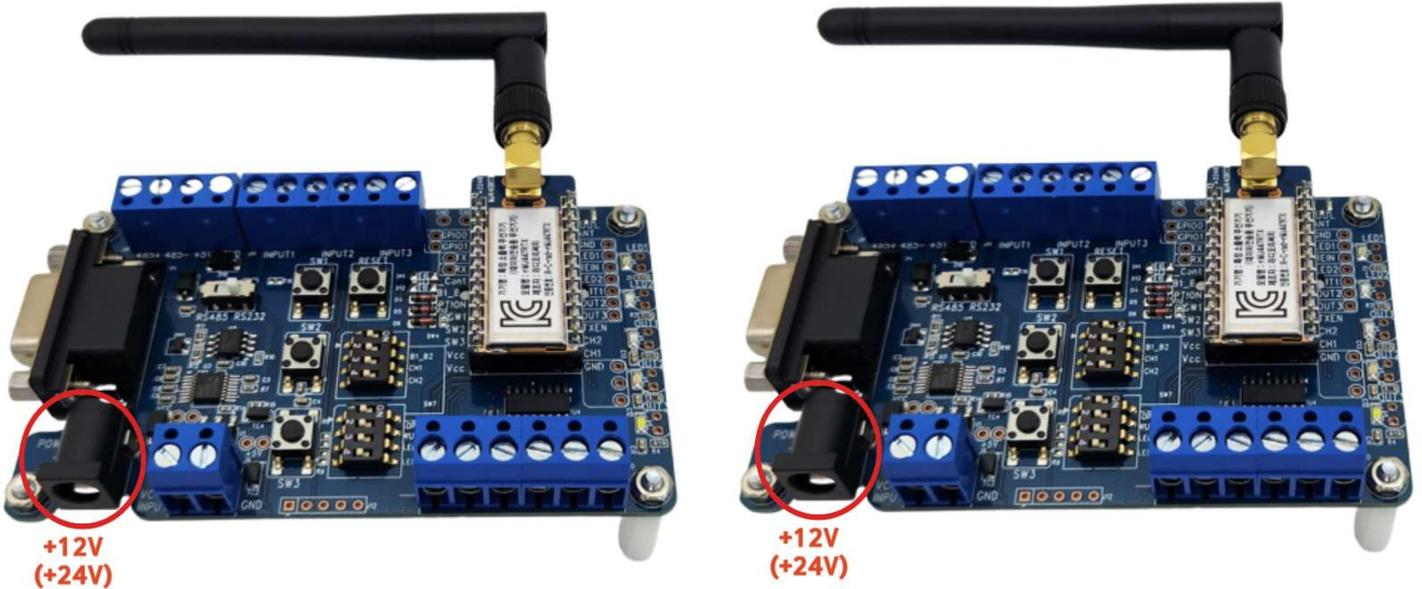
3-3. 보드에 리셋이라 적힌 스위치를 눌러 모듈을 초기화시킨다.



전원을 공급하고, nMJ447RTX 를 초기화하는데 60ms 정도의 시간이 필요함을 확인할 수 있다.

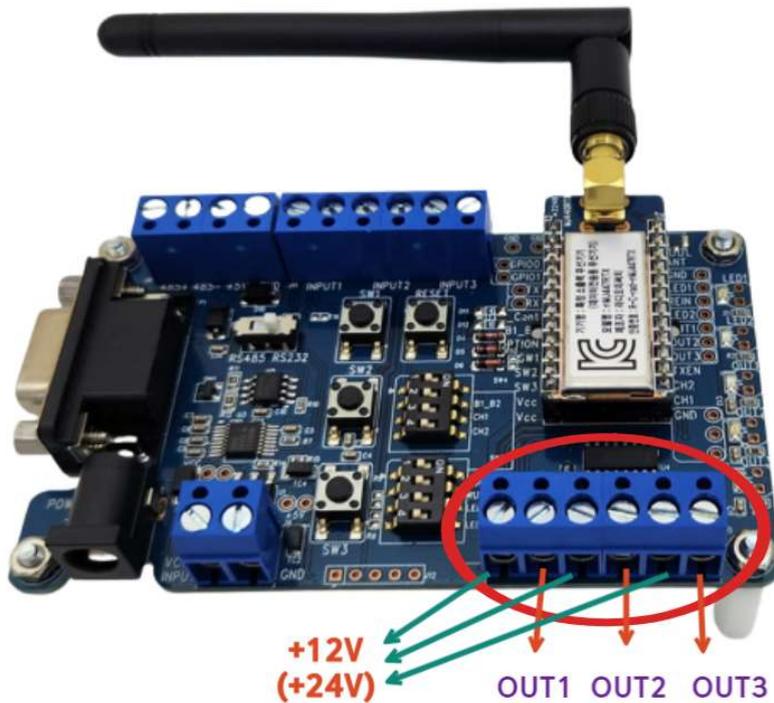
4. 시험해 보기

4-1. 8절의 결합된 세트를 2조 준비한다.



[nMJ447RTX 와 MJ447CONT 가 결합된 2조 준비]

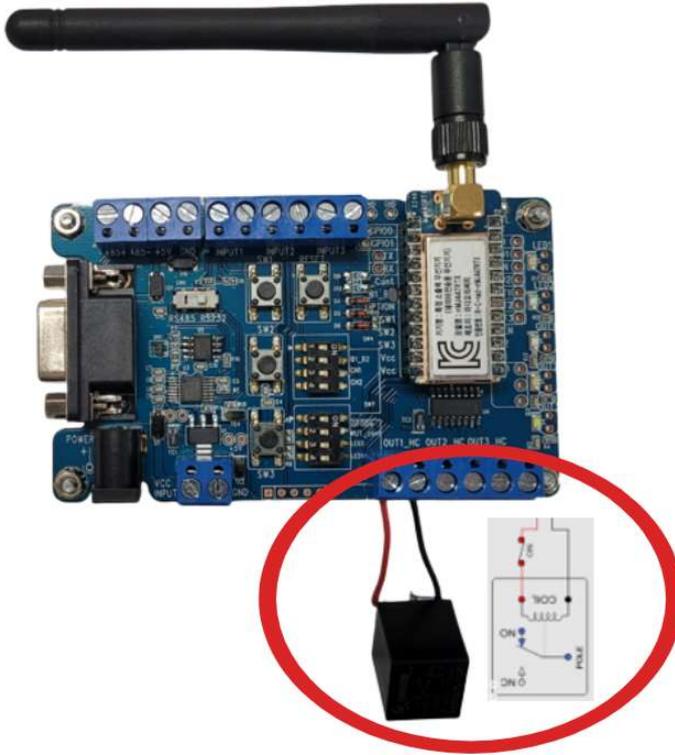
4-2. 송신측에서 SW1,SW2,SW3를 선택하여 누르면 수신측에서 이에 해당하는 LED가 켜지고 하단의 터미널 블록으로 해당 출력이 출력된다. 이 출력은 릴레이를 구동시키기 위한 출력이며, OPEN COLLECTOR 출력이다.



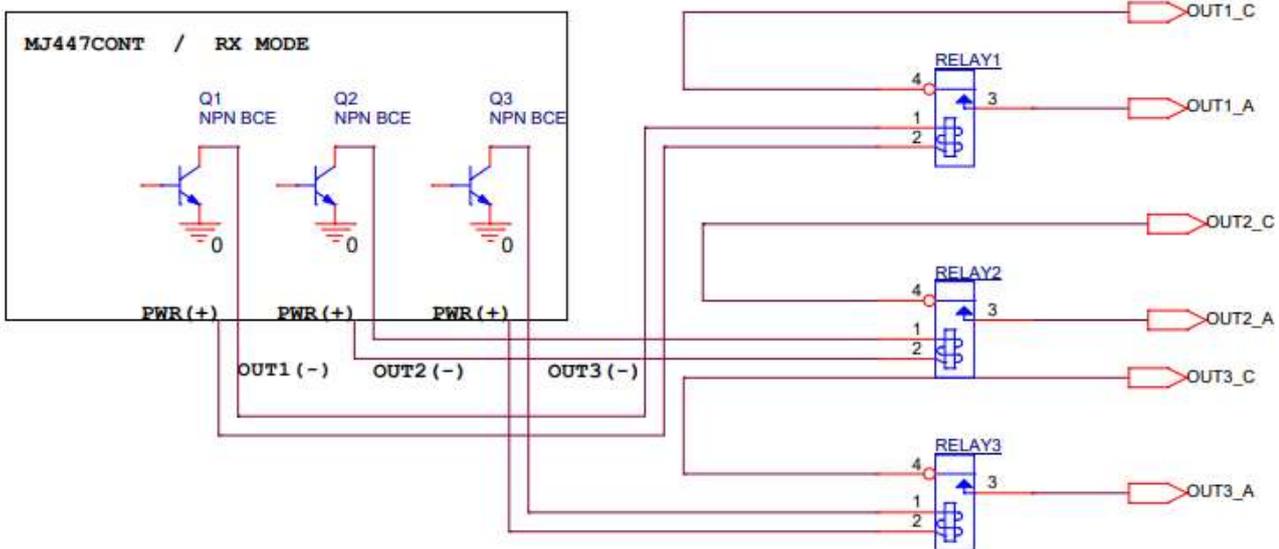
4. 시험해 보기

4-3. 릴레이 접속 방법은 아래와 같다.

릴레이 중 극성이 있는 릴레이 (LED가 내장된 릴레이) 는 특별히 극성에 주의하여야 하며, 극성이 틀리면 콘트롤보드가 타거나, 오동작의 원인이 된다.

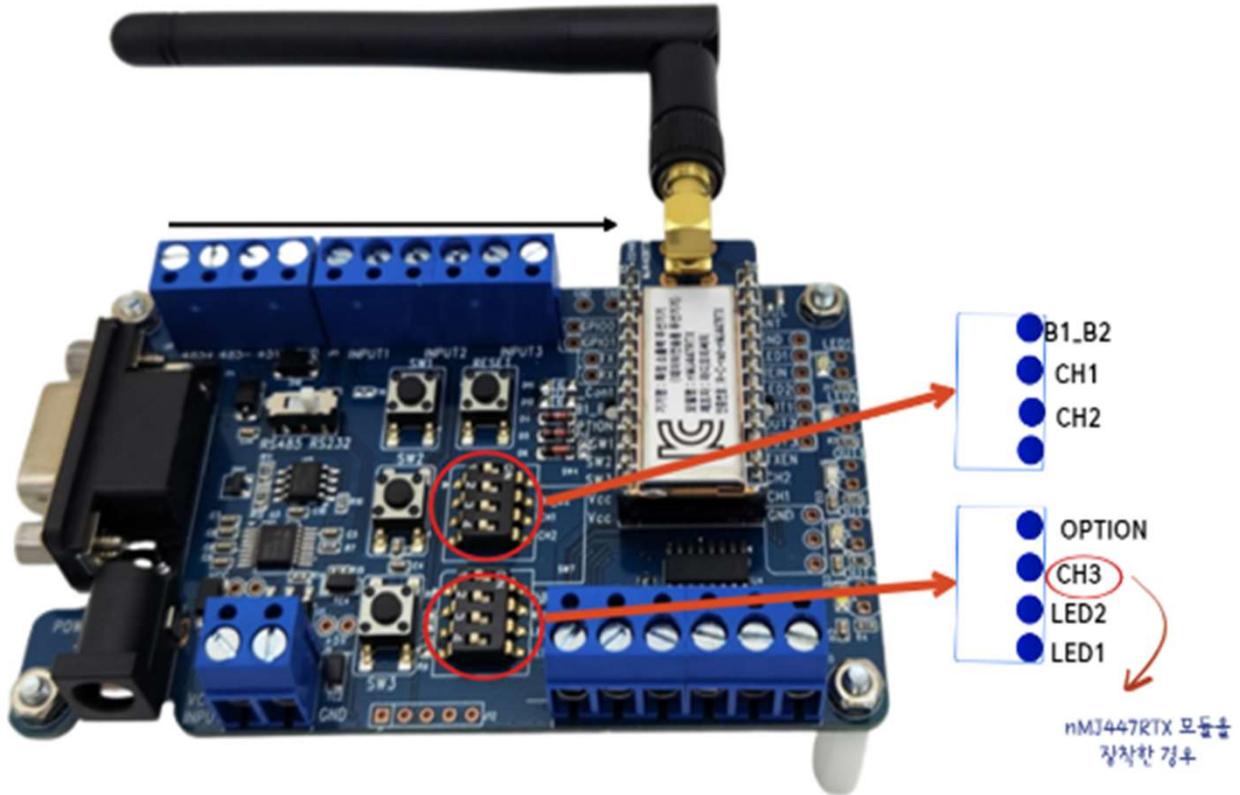


[nMJ447RTX 에 릴레이 연결]



4. 시험해 보기

4-4. 같은 지역에 여러 개의 세트를 장착하는 경우 혹은 무선 채널을 바꾸어 동작시키고자 하는 경우는 DIP스위치를 조작하여 채널을 변경한다.



- B1_B2는 이 무선모듈의 주파수 밴드를 말하며, 이 모듈은 2가지 밴드가 인증되어 있다. 447.2625 ~ 447.5625 MHz밴드는 도난, 화재 경보장치등의 안전시스템용으로 허가된 밴드이며, B1_B2를 ON으로 하여 선택한다.
- 447.8625 ~ 447.9875MHz 밴드는 일반 데이터 전송용 밴드이며 B1_B2 를 OFF로 하여 선택한다. DIP스위치의 조합으로는 인증된 전체 채널을 표현하기 어려우므로 각 밴드당 8채널씩 구현하였다.

아래의 표와 같이 CH1,CH2,CH3의 조합으로 채널을 구성한다.

4. 시험해 보기

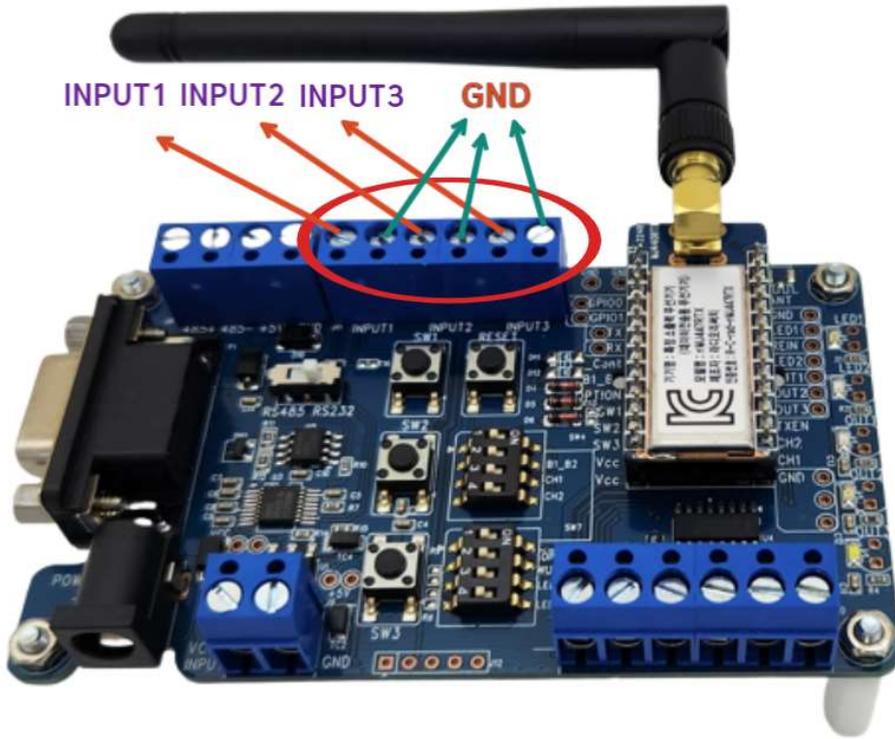
[딥스위치 조작과 주파수표]

B1_B2	CH1	CH2	CH3	해당 주파수
ON	ON	ON	OFF	447.2625MHz
ON	OFF	ON	OFF	447.2875MHz
ON	ON	OFF	OFF	447.3125MHz
ON	OFF	OFF	OFF	447.3375MHz
ON	ON	ON	ON	447.3625MHz
ON	OFF	ON	ON	447.3875MHz
ON	ON	OFF	ON	447.4125MHz
ON	OFF	OFF	ON	447.4375MHz
OFF	ON	ON	OFF	447.8625MHz
OFF	OFF	ON	OFF	447.8875MHz
OFF	ON	OFF	OFF	447.9125MHz
OFF	OFF	OFF	OFF	447.9375MHz
OFF	ON	ON	ON	447.9625MHz
OFF	OFF	ON	ON	447.9875MHz
OFF	ON	OFF	ON	447.8750MHz
OFF	OFF	OFF	ON	447.9000MHz

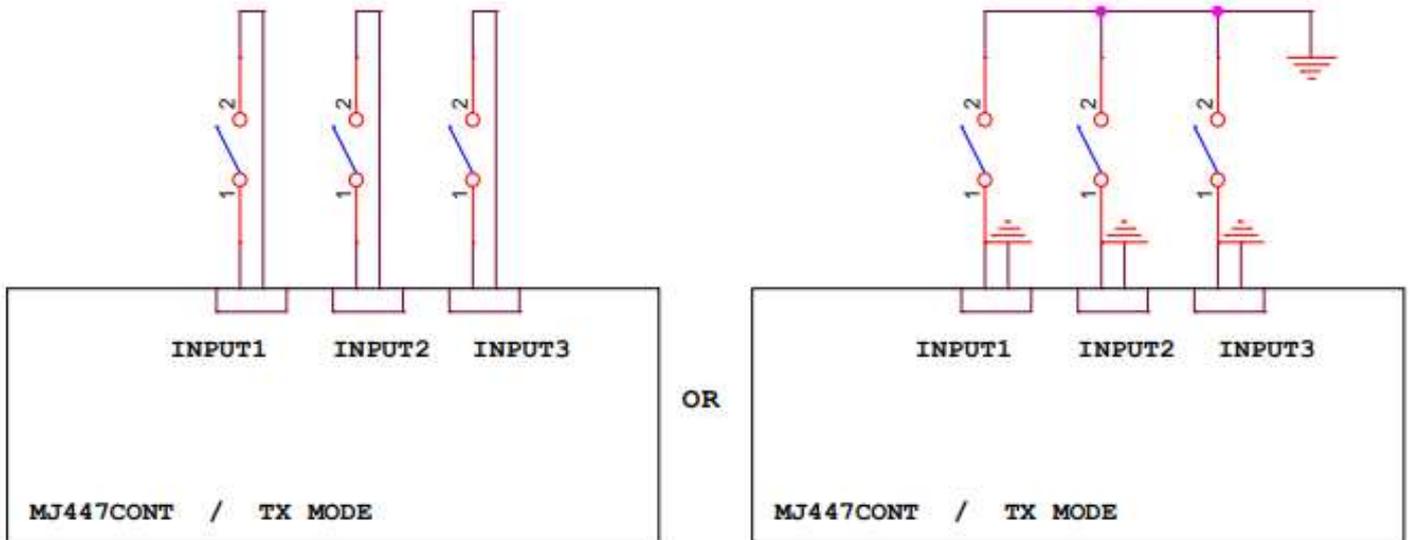
주기 : CH3는 V75(버전)이상에서 추가한 채널이고, 이전 버전과의 호환성을 위하여 위의 표와 같이 채널을 운영하고 있다.

4. 시험해 보기

4-5. 입력 스위치의 연결.



입력 스위치의 연결은 그림과 같다. 입력용 터미널 블록은 컨트롤보드 상의 SW1, SW2, SW3와 각각 병렬로 연결되어 있으며, 각 터미널 블록의 입력 단자는 모듈의 20번, 21번, 22번 핀과 연결되어 있다.



4. 시험해 보기

4-6. 출력의 피드백

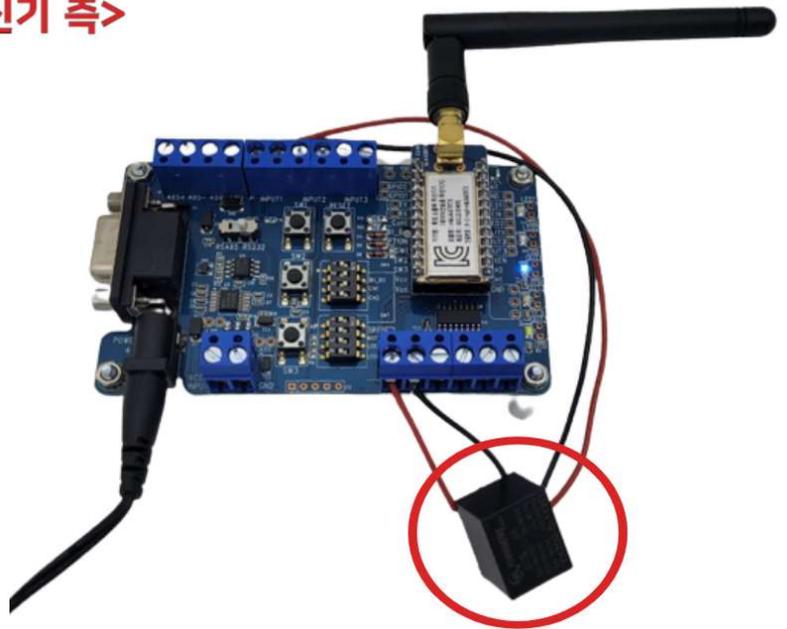
경우에 따라 동작하는 출력 신호를 입력으로 피드백하여 출력의 동작상태를 확인할 필요가 있는 경우에는 아래와 같이 결선한다.

이렇게 결선하면 스위치를 작동시킨 쪽에서, 수신기측 동작 상황을 감시할 수 있다.

<송신기 측>



<수신기 측>



5. 패킷통신

5-1. 패킷(UART) 의 데이터

5.1.1 USART Asynchronous mode

5.1.2 전송속도: 9600bps

5.1.3 데이터 비트: 8비트

스톱비트: 1비트

패러티: NONE

입력전압레벨 : 3.0V

5-2. 패킷의 송수신 방법

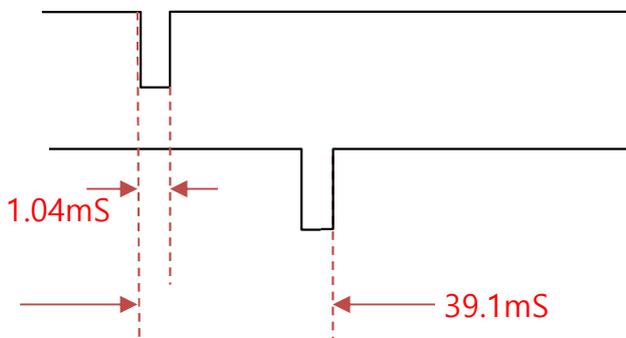
5.1절에 언급한 포맷의 UART데이터를 1회 최대 38바이트까지 전송가능하다. 이 데이터를 모듈의 RX핀(핀16번)에 전송하면 , 모듈은 이 데이터들의 수신이 끝났음을 인지하는대로 자동적으로 무선 송신한다.

5-3. 무선 전송속도

nMJ447RTX 의 패킷 전송속도는 9600bps이고, 이를 무선 전송하는 무선 전송속도는 4800bps이다.

5-4. 패킷 전송의 전송시간의 예

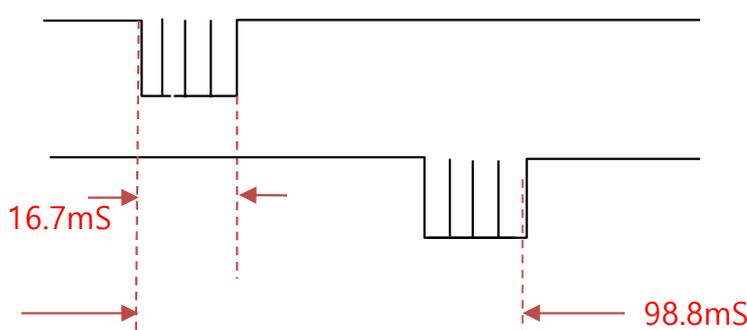
5.4.1 1 바이트 전송



송신모듈측 UART 입력포트 (14번핀)

수신모듈측 UART 출력포트 (13번핀)

5.4.2 16 바이트 전송



송신모듈측 UART 입력포트 (14번핀)

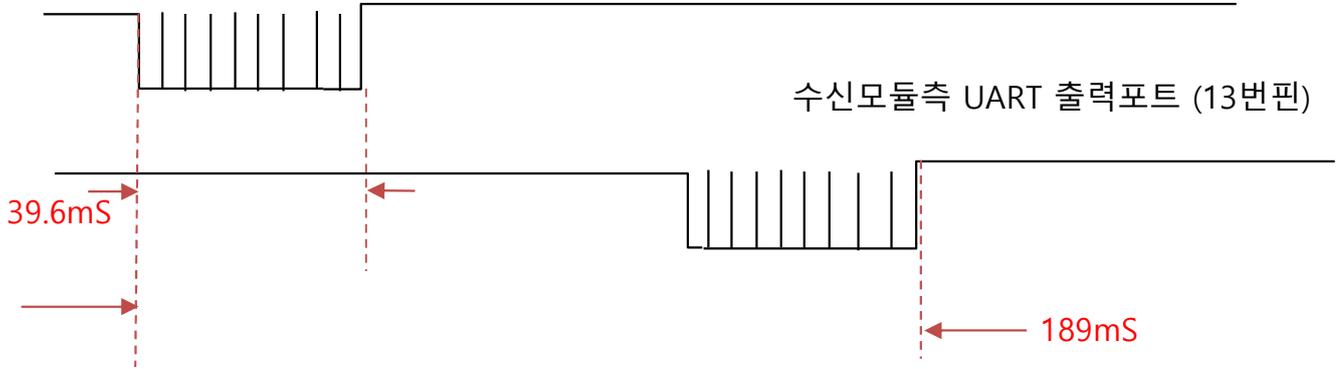
수신모듈측 UART 출력포트 (13번핀)

5. 패킷통신

5-4. 패킷 전송의 전송시간의 예

5.4.3 38 바이트 전송

송신모듈측 UART 입력포트 (14번핀)



5.4.4 전송시간의 어림계산

	1 바이트	16바이트	38바이트
송신측 패킷 전송시간	1.04	16.7	39.6
송신측 딜레이	≒ 10	≒ 10	≒ 10
무선전송시간	18.3mS	43.3mS	80mS(+10mS)
수신측 딜레이	≒ 8.72	≒ 12.1	≒ 9.8
수신측 패킷 전송시간	1.04mS	16.7mS	39.6mS
Total	39.1mS	98.8mS	189mS

① 1바이트 무선 전송시간

◆ Preamble(5바이트) + Sync(2바이트) + 패킷갯수(1바이트) + Payload(1바이트) + CRC(2바이트) = 11바이트

◆ $11 \times 8 / 4800 = 18.3\text{mS}$

② 6바이트 무선 전송시간

◆ Preamble(5바이트) + Sync(2바이트) + 패킷갯수(1바이트) + Payload(16바이트) + CRC(2바이트) = 26바이트

◆ $26 \times 8 / 4800 = 43.3\text{mS}$

③ 38바이트 무선 전송시간

◆ Preamble(5바이트) + Sync(2바이트) + 패킷갯수(1바이트) + Payload(38바이트) + CRC(2바이트) = 48바이트

◆ $48 \times 8 / 4800 = 80\text{mS}$

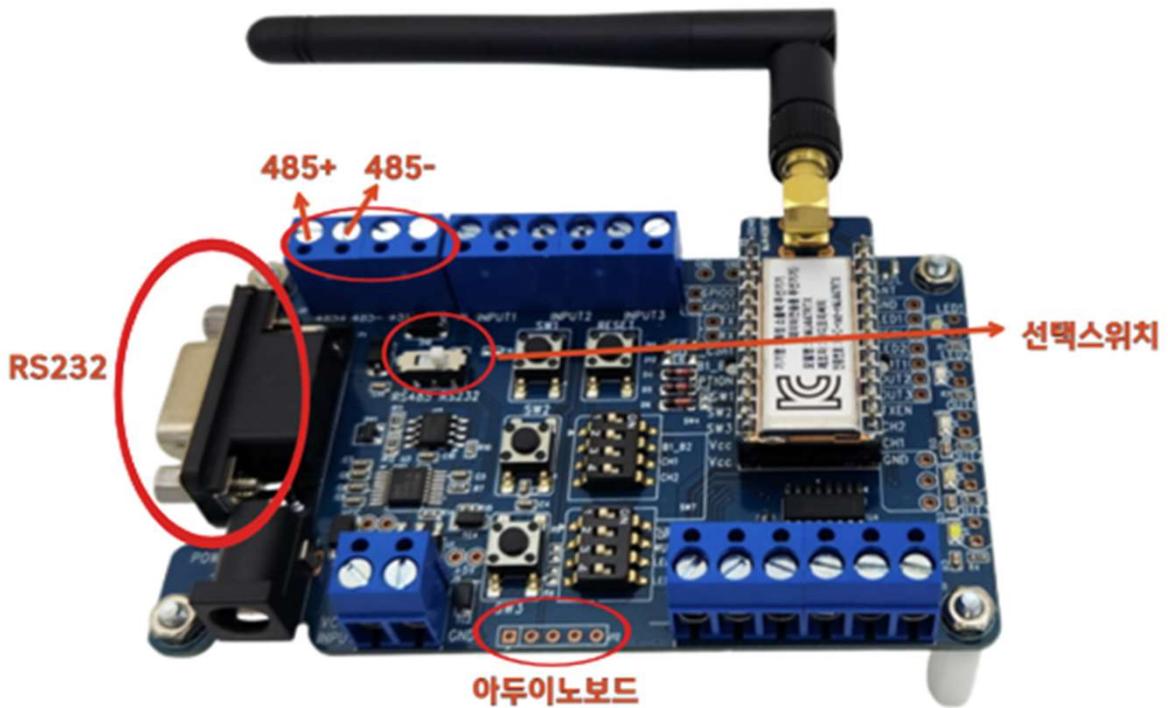
5. 패킷통신

5-5. 패킷 전송의 방법

5.5.1 RS232

5.5.2 RS485

5.5.3 아두이노보드



5.5.4 RS232의 연결

컨트롤보드의 좌측면 9PIN Serial PORT에 RS232 to USB 젠더를 연결하고 PC에 통신 터미널 프로그램을 실행한다.

이 사용자가이드에서는 "Serial Com"이라는 터미널 프로그램을 예로 설명한다. (무상 open되어 있는 프로그램으로 구글에서 검색하여 다운로드 할 수 있음)

수신측에도 같은 방법으로 연결하고 터미널 프로그램을 실행한다.

5. 패킷통신

5-5. 패킷 전송의 방법

5.5.5 터미널프로그램의 실행

① 설치

구글에서 SerialCom을 검색하고 다운로드하여 설치한다.

② 세팅

◆ Serial Port

컴퓨터의 장치관리자에 "포트" 를 클릭하면 RS232 to USB젠더가 연결된 USB Serial Port의 번호를 확인할 수 있다. 이 번호가 10번 이내이면 이대로 선택하여 세트하면되고, 이 번호가 "COM11"이상이면 해당 포트에서 마우스 오른쪽을 클릭하여 **속성>포트설정>고급>COM포트번호>10번이내로 변경>확인** 하여 변경한 후 이 변경된 Serial Port 번호를 등록한다.

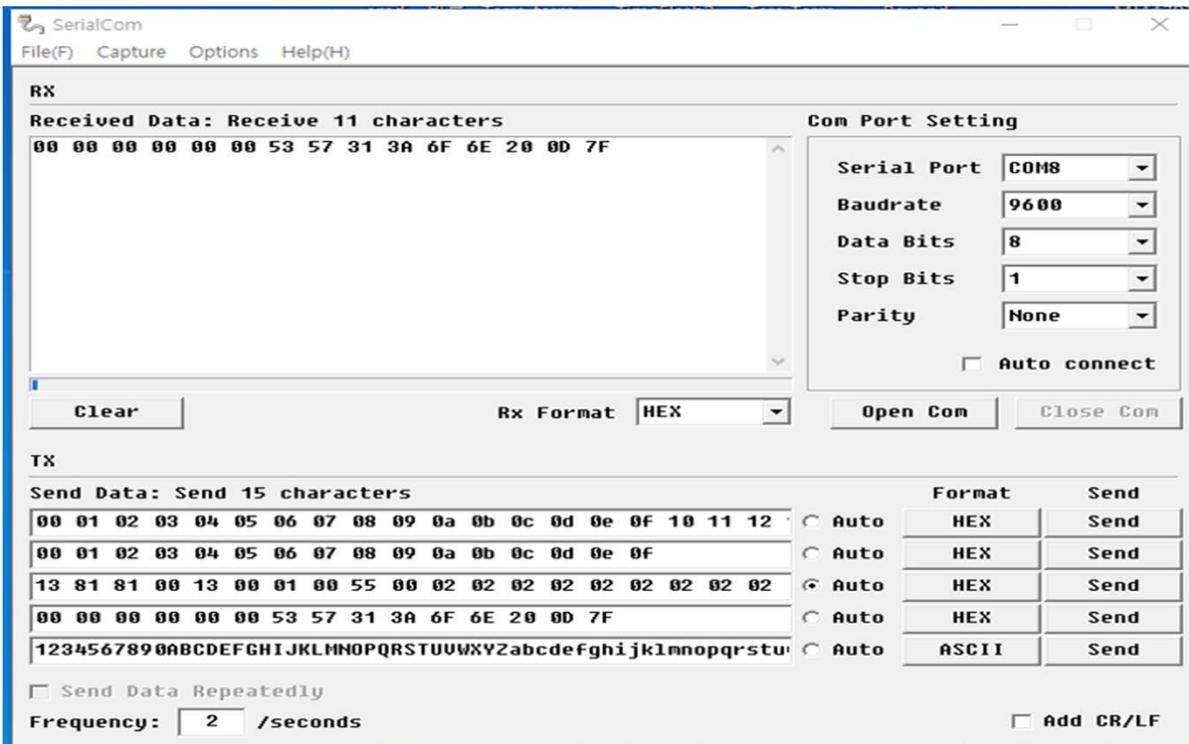
◆ • Baudrate : 9600

• Data Bits : 8

• Stop Bits : 1

• Parity : None

◆ 각 세트할 항목을 선택하였으면 **Close Com** 을 눌러 **Highlight** 되도록 한다.



5. 패킷통신

5-5. 패킷 전송의 방법

5.5.6 스위치의 동작

5.5.6.1 송신측의 스위치 중 SW1을 누르면 수신측의 터미널 프로그램에 15바이트의 숫자열이 출력된다.

이 숫자열은 HEX 값이며 터미널 프로그램의 상단에 기록된 값을 다시 적어보면 00 00 00 00 00 00 53 57 31 3A 6F 6E 20 0D 7F 이고, 이중 마지막 바이트의 0x7F라는 값은 이 패킷을 수신할 때 수신 신호 강도를 나타내는 값(R.S.S.I./Received Signal Strength Indicator)이다. 이 값을 이용하는 방법은 뒤에 아두이노 편에서 다시 설명할 예정이다.

5.5.6.2 위 패킷을 복사하여 아래의 TX 줄에 복사한다.

마지막 바이트를 삭제하고, Format을 **HEX** 로 하고 **Send**를 누른다.

반대편 수신기측의 터미널 프로그램에 15바이트가 출력됨을 확인할 수 있고, 해당 LED및 출력은 약 12초간 지속된다. 마찬가지로 SW1을 땄때에도 패킷이 전송됨을 확인할 수 있고, SW2,SW3에도 동일한 방법으로 적용된다. 이때 송신측의 패킷 데이터를 아래의 항에 예시한다.

또한 아두이노로 동작시키는 방법도 다음절에 예시한다.

5.5.6.3 송신측 HEX DATA로 수신측 출력 스위치 조절하기

1. 아래의 데이터를 HEX값으로 전송한다
2. 4초 주기로 전송한다.
3. 켜(ON)라는 명령어를 10초 이내로 보내지 않으면 수신모듈쪽의 켜짐(ON)이 꺼진다(OFF).

송신모듈측 HEX 데이터	수신모듈측 출력 스위치
00 00 00 00 00 00 53 57 31 3A 6F 6E 20 0D	OUT1 // ON
00 00 00 00 00 00 53 57 31 3A 4F 46 46 0D	OUT1 // OFF
00 00 00 00 00 00 53 57 32 3A 6F 6E 20 0D	OUT2 // ON
00 00 00 00 00 00 53 57 32 3A 4F 46 46 0D	OUT2 // OFF
00 00 00 00 00 00 53 57 33 3A 6F 6E 20 0D	OUT3 // ON
00 00 00 00 00 00 53 57 33 3A 4F 46 46 0D	OUT3 // OFF
00 00 00 00 00 00 53 57 34 3A 6F 6E 20 0D	OUT1 & OUT2 // ON
00 00 00 00 00 00 53 57 34 3A 4F 46 46 0D	OUT1 & OUT2 // OFF
00 00 00 00 00 00 53 57 35 3A 6F 6E 20 0D	OUT1 & OUT3 // ON
00 00 00 00 00 00 53 57 35 3A 4F 46 46 0D	OUT1 & OUT3 // OFF
00 00 00 00 00 00 53 57 36 3A 6F 6E 20 0D	OUT2 & OUT3 // ON
00 00 00 00 00 00 53 57 36 3A 4F 46 46 0D	OUT2 & OUT3 // OFF
00 00 00 00 00 00 53 57 37 3A 6F 6E 20 0D	OUT1 & OUT2 & OUT3 // ON
00 00 00 00 00 00 53 57 37 3A 4F 46 46 0D	OUT1 & OUT2 & OUT3 // OFF

5. 패킷통신

5-5. 패킷 전송의 방법

5.5.7 RS485

5.5.7.1 RS485 to USB 젠더를 좌측상단의 터미널블럭에 연결한다.

5.5.7.2 슬라이드 스위치를 좌측의 RS485로 연결한다.

5.5.7.3 터미널 프로그램을 위의 예와 같은 방법으로 세트하고 동작시킨다.

6. 아두이노를 이용하여 동작시키기

6-1. 연결

하단 중앙의 5핀 콘넥터에 아래와 같이 아두이노보드와 연결한다.



6.1.1 +5V는 아두이노보드의 5V에 연결한다.

6.1.2 D_in은 아두이노보드의 "3"에 연결한다.

아두이노보드상에서 SoftwareSerial을 이용할 예정이다.

6.1.3 D_out은 아두이노보드의 "2"에 연결한다.

6.1.4 GND는 아두이노보드의 GND에 연결한다.

그리고 아두이노 보드를 컴퓨터의 USB Port와 연결하면 MJ447CONT 및 nMJ447RTX는 아두이노의 전원을 이용하게 된다.

6.1.5 OPTION(RESET)은 모듈을 리셋하고자 하는 경우에 필요한데 여기서는 생략하기로 한다.

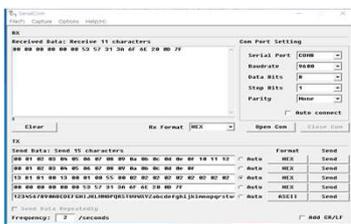
6. 아두이노를 이용하여 동작시키기

6-2. 아두이노의 실행 및 스케치 프로그램

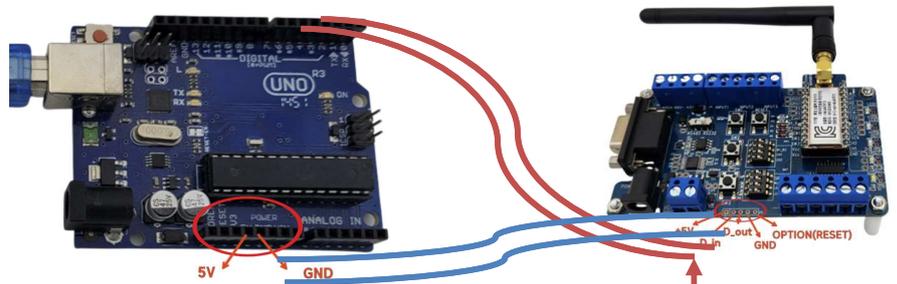
(nMJ447CONT_SW_SEQUENCE_1ST_230911.ino) 스케치 프로그램을 아래에 첨부한다.

이 스케치 프로그램은 HardwareSerial과 SoftwareSerial을 이용하여, 컴퓨터의 터미널 프로그램과 (아두이노에 내장된 터미널 프로그램은 Hexadecimal의 Data를 전송하고 수신하는데 불편하므로 앞에 예로든 터미널 프로그램을 이용함)

아두이노 보드간에는 HardwareSerial을 이용하고, 아두이노보드와 접속한 MJ447CONT 및 nMJ447RTX와의 통신에는 SoftwareSerial을 이용하도록 되어 있다.



HardwareSerial



SoftwareSerial

4초마다 SW3 스위치를 누른다는 명령을 무선 전송하고, 상대방으로부터 받은 ACK신호를 터미널로 출력한다.

이때 ACK 신호뒤에 붙어 있는 R.S.S.I. 값을 이용하여 R.S.S.I. 를 계산하여 출력한다.

또한, 터미널 프로그램 상에서 전송하는 패킷데이터를 무선모뎀으로 전송하는 기능도 포함되어 있으니 확인하기 바란다.

nMJ447RTX_CONT

```

// MJ447CONT_SW_SEQUENCE_1ST_230829
// MJ447CONT_SW_SEQUENCE_1ST_230829_230911

//*****
//*****
//*****
//*****
//*****

#include <TimerOne.h>
#include <avr/wdt.h>
#include <avr/io.h>
#include <SoftwareSerial.h>
//*****
byte serial_input0[64]={};
byte serial_cmd_input[10]={};
byte softwareserial_input1[64]={};
byte serial_cnt1;
//*****
uint8_t timer1_cnt;
uint16_t tmr_1ms_cnt;
uint16_t tmr_1ms_cnt1;
uint16_t tmr_1ms_cnt2;
uint16_t tmr_1ms_cnt3;
byte serial_cmd_cnt;
uint8_t sw_nbr_cnt;
//*****
uint16_t tmr_1ms_cnt4;
uint16_t tmr_u125_4_softwareserial;
//*****
|//*****
const uint8_t SW1_ON_command[14]={0x00,0x00,0x00,0x00,0x00,0x00,0x53,0x57,0x31,0x3A,0x6F,0x6E,0x20,0x0D};
const uint8_t SW2_ON_command[14]={0x00,0x00,0x00,0x00,0x00,0x00,0x53,0x57,0x32,0x3A,0x6F,0x6E,0x20,0x0D};

```

nMJ447RTX_CONT

```
const uint8_t SW3_ON_command[14]={0x00,0x00,0x00,0x00,0x00,0x00,0x53,0x57,0x33,0x3A,0x6F,0x6E,0x20,0x0D};
uint8_t      SW_COMMAND[14]={0x00,0x00,0x00,0x00,0x00,0x00,0x53,0x57,0x31,0x3A,0x6F,0x6E,0x20,0x0D};
uint8_t      SW_NBR[8]={0x30,0x31,0x32,0x33,0x34,0x35,0x36,0x37};
//*****

const uint8_t ALL_OFF_command[14]={0x00,0x00,0x00,0x00,0x00,0x00,0x53,0x57,0x37,0x3A,0x4F,0x46,0x46,0x0D};
//*****
//*****
SoftwareSerial software_uart(2,3);
//*****
#define OUTPUT_4 4
//*****
int  RSSI_value;
int  RSSIcal;
int  RSSI_dbm;
//*****
//*****
//*****
//*****
void setup() {
  wdt_reset();
  Serial.begin(9600);
  software_uart.begin(9600);
  pinMode(OUTPUT_4 OUTPUT);

  Timer1.initialize(125); //타이머 간격 125us
  Timer1.attachInterrupt( timerIsr ); // 타이머 간격대로 timerIsr 함수 실행

  RSSIcal=130;
}
//*****
void loop() {
//*****
  wdt_reset();
```

```

//*****
// Serial uart packet from terminal to modem
//*****
if((serial_cnt0)&&(!Serial.available())) {
    for(int i=0; i<serial_cnt0; i++) {
        software_uart.write(serial_input0[i]);
    }
    serial_cnt0=0;
}
//*****

//*****
// SW1 이 지속적으로 눌러진 경우
//*****
/* if(tmr_1ms_cnt1>4000) {
    tmr_1ms_cnt1=0;
    for(uint8_t i=0; i<14; i++) {
        software_uart.write(SW1_ON_command[i]);
    }
}*/
//*****
// SW2 이 지속적으로 눌러진 경우
//*****
/* if(tmr_1ms_cnt2>4000) {
    tmr_1ms_cnt2=0;
    for(uint8_t i=0; i<14; i++) {
        software_uart.write(SW2_ON_command[i]);
    }
}*/
//*****
// SW3 이 지속적으로 눌러진 경우
//*****
if(tmr_1ms_cnt3>4000) {
    tmr_1ms_cnt3=0;

```

```
for(uint8_t i=0; i<14; i++) {
    software_uart.write(SW3_ON_command[i]);
}
}
//*****
// 모든 출력값 끄고자 할때
//*****
/* if(tmr_1ms_cnt4>4500) {
    tmr_1ms_cnt4=0;
    for(uint8_t i=0; i<14; i++) {
        software_uart.write(ALL_OFF_command[i]);
    }
}*/
//*****
// LED를 순차적으로 켜고자 할때
//*****
/* if(tmr_1ms_cnt>4000) {
    SW_COMMAND[8]=SW_NBR[sw_nbr_cnt%8];
    for(uint8_t i=0; i<14; i++) {
        software_uart.write(SW_COMMAND[i]);
    }
    sw_nbr_cnt++;
}*/
//*****
// R.S.I. Calculation
//*****
if(software_uart.available()) {
    char c = software_uart.read();
    Serial.write(c);
    if (serial_cnt1 < 64) {
        softwareserial_input1[serial_cnt1++] = c;
    }
}
tmr_u125_4_softwareserial=0;
```

```
    }
    if((tmr_u125_4_softwareserial>10)&&(serial_cnt1)) {
        if(((softwareserial_input1[(serial_cnt1-5)]==0x41)&&(softwareserial_input1[(serial_cnt1-4)]==0x43)
            &&(softwareserial_input1[(serial_cnt1-3)]==0x4B)&&(softwareserial_input1[(serial_cnt1-2)]==0x0D)) {
            RSSI_value=softwareserial_input1[(serial_cnt1-1)];
            RSSI_dbm=((RSSI_value/2)-RSSIcal);
            Serial.println(" ");
            Serial.print("RSSI_dbm= ");
            Serial.print(RSSI_dbm);
            Serial.println("(dbm)");
            //Serial.write(softwareserial_input1[(serial_cnt1-1)]);
        }
        if(((softwareserial_input1[(serial_cnt1-5)]==0x6F)&&(softwareserial_input1[(serial_cnt1-4)]==0x6E)
            &&(softwareserial_input1[(serial_cnt1-3)]==0x20)&&(softwareserial_input1[(serial_cnt1-2)]==0x0D)) {
            RSSI_value=softwareserial_input1[(serial_cnt1-1)];
            RSSI_dbm=((RSSI_value/2)-RSSIcal);
            Serial.println(" ");
            Serial.print("RSSI_dbm= ");
            Serial.print(RSSI_dbm);
            Serial.println("(dbm)");
            //Serial.write(softwareserial_input1[(serial_cnt1-1)]);
        }
        serial_cnt1=0;
    }
    //*****
}
//*****
//*****
//*****
//*****
//*****
//*****
void serialEvent() {
    while (Serial.available()) {
```

```
char c = Serial.read();

  if (serial_cnt0 < 64) {
    serial_input0[serial_cnt0++] = c;
  }
}

//*****

void timerISR()
{
  //digitalWrite( OUTPUT_4 !digitalRead(OUTPUT_4)); //Every 125uS

  timer1_cnt++;

  if((timer1_cnt%8)==0) {
    tmr_1ms_cnt++;
    tmr_1ms_cnt1++;
    tmr_1ms_cnt2++;
    tmr_1ms_cnt3++;
  }else if((timer1_cnt%8)==1) {

    tmr_u125_4_softwareserial++;
  }else if((timer1_cnt%8)==2) {

  }else if((timer1_cnt%8)==3) {

  }else if((timer1_cnt%8)==4) {

  }else if((timer1_cnt%8)==5) {

  }else if((timer1_cnt%8)==6) {
```

```
    }else if((timer1_cnt%8)==7) {  
  
    }else {  
  
    }  
}  
//*****  
//*****
```

7. 수신신호강도(R.S.S.I. Received Signal Strength Indicator)

7.1. 패킷을 수신할 때 수신하는 신호의 강도를 측정하는 방법이다. (수신되는 신호의 감도를 측정하는 것은 아니다.)

MJ447CONT에 nMJ447RTX를 연결한 결합체를 2조 준비하고, 송신기의 스위치를 누르면 수신측에 HEX data가 수신되는데, 이때 패킷의

맨 뒤에 붙은 1 byte Data 가 수신된 패킷의 R.S.S.I (수신 신호 세기)이다. 또한 송신측으로는 ACK 신호를 출력하는데 이때 맨 끝에 출력하는 1byte가 수신신호세기 (R.S.S.I)이다.

(예) 00 00 00 00 00 00 53 57 31 3A 4F 46 46 0D xx (15번째 Byte)

(예) 53 57 31 4F 46 46 41 43 4B 0D xx (11번째 Byte)

7.2. 또한 5.4에 언급된 패킷을 전송하여도 같은 결과를 얻을 수 있다.

8. 주의사항

8.1. 이 무선모듈(nMJ447RTX)은 무선으로 정보를 주고 받으며, 이 주고받은 데이터는 암호화되지 않은 일반 신호이다. 따라서 해킹,도난으로부터 취약하다. 해킹,도난으로부터 방지하는 수단은 고객이 별도로 준비하여야 한다.

또한, 의도적인 문제 외에도 무선 채널로 전송되는 정보는 채널상의 여러 종류의 노이즈 상황에 따라, 데이터가 전송이 누락되거나 정보중의 일부가 왜곡되어 달라진 값으로 수신되는 사례도 간혹 있으니, 이에 대한 대책도 별도로 준비해야 한다.

8.2. 이 사용자설명서의 기술된 내용과 펌웨어의 기능은 고객 여러분께 공지없이 수정되거나 업데이트 될 수 있다.

따라서, 이미 구매한 제품과 사용자설명서상의 동작이 완전히 일치하지 않을 수 있다. 각 버전간에는 100%호환을 목표로 하지만, 여러사정으로 호환이 안되거나, 기능이 변경되는 경우가 존재할 수 있음을 공지한다.

8.3 모듈의 그라운드 (핀1과 핀11)는 모듈을 사용하는 시스템의 그라운드와 연결 되어야 하고, 또한 전원의 접지와 연결시켜 동작시키는 것이 기본이다.

8.4 최대 통신거리와 안정적 통신거리

통신거리시험을 하면 무선모듈이 갖는 최대 통신거리는 주변 환경이나 날씨에 따라 달라지게 마련이다. 그런데, 통신에서는 안정적인 통신거리 내에서 제품을 사용하여야 하므로, 최종적인 설치를 하기 전에는 그 지역에서의 최대 통신거리가 어디까지인지 확인하고, 이에 따라 안정적 통신거리를 확보하여야 하고, 세트의 설치 위치나 안테나의 방향을 조정하여 가장 안정적인 통신 상태를 확보하도록 한 후 제품을 고정시킨다.

이 모델의 최대통신거리는 오솔길에서 700m 정도이다.

8.5 7.5 릴레이 장착시 주의사항

첫째, 공급 전원과 동일한 동작 전압을 가진 릴레이를 선택한다.

둘째, 릴레이의 코일쪽 전원은 극성을 가지고 있는 릴레이가 있다. 7절을 참조하여 설치시 주의가 요구된다.

8.6 A/S에 대한 공지

8.6.1 모듈내부에 전원 임펄스 전압에 대한 보호장치가 있음에도 지나친 과전압으로 내부 부품이 완전파손되거나, IC에 화재가 난 상태로 반송되는 경우에는 세트전체가 전원 쇼크를 받은 경우로 수리가 불가능함을 이해하기를 바랍니다.

8.6.2 소비자가 모듈을 구매후, 헤더핀,콘넥터 혹은 내부를 변경한 상태로 반송되는 제품은 공장에서 성능을 확인할 방법이 없습니다. 따라서 수리도 불가능함을 이해하여 주기 바랍니다.

8.6.3 A/S시에는 수리 항목에 따라 수리비가 청구됨을 공지합니다.

8.6.4 A/S 기간은 1년으로 한다.