NS-RX231를 위한 소프트웨어가이드

<IR 리모컨 송신>

목차

1 프로젝트 가져오기

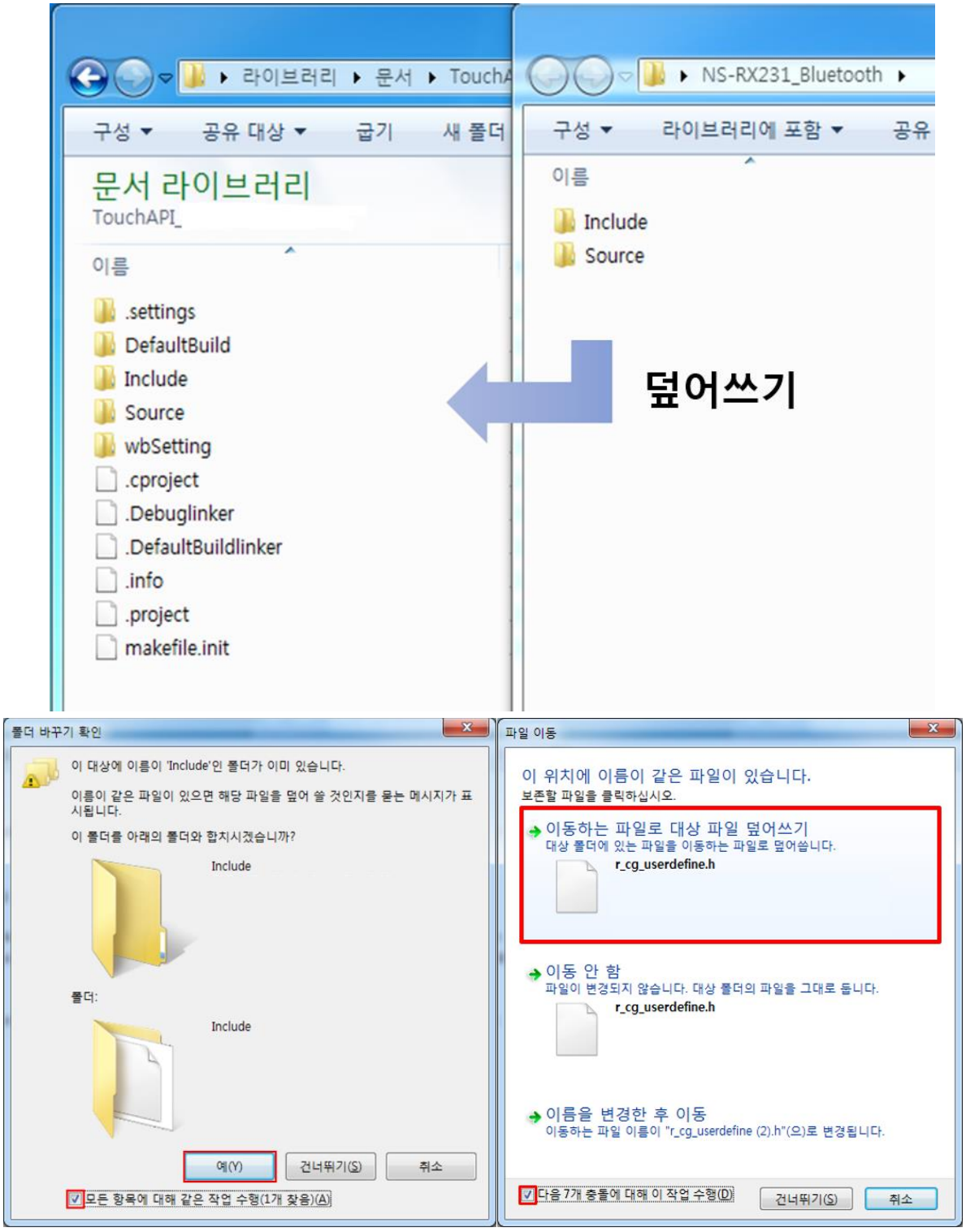

그림 1-1 소스파일 덮어쓰기

첨부된 소스파일을 Workbench6 First step guide 마법사로 만든 프로젝트에 덮어쓰기 한 후, e2studio에서 실행합니다.

2 개요

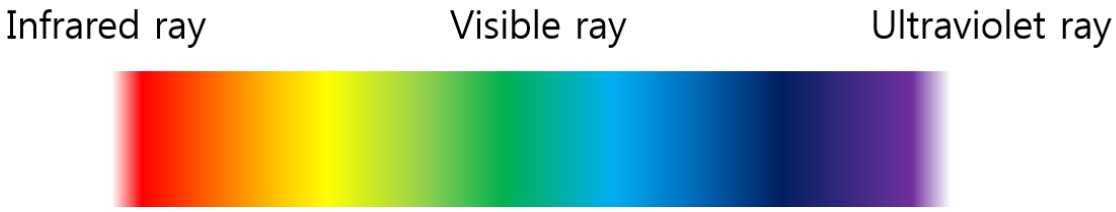

그림 2-1 전자기파 파장

적외선은 사람이 볼 수 있는 파장인 가시광선보다 파장이 긴 전자파입니다. 빨간색을 표현하는 파장보다 바깥쪽에 위치하고 있기 때문에 적외(赤外)라 합니다.

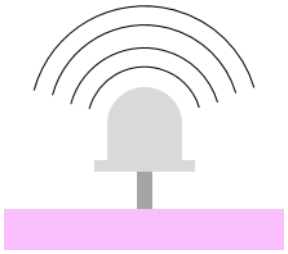

그림 2-2 적외선 발광소자 IR LED

IR LED는 일반 LED와 같이 전류가 흐르면 빛을 내지만, IR LED는 가시 광선이 아닌 적외선을 출력하기 때문에 사람의 눈에는 보이지 않는다는 점이 다릅니다.

적외선을 이용하여 무선 통신을 실현하고 있는 좋은 예가 적외선 리모컨입니다.

적외선 통신은 인체에 미치는 영향이 적은 무선 통신이며, 비교적 쉽게 구현할 수 있기 때문에 소용량의 데이터 통신에 많이 사용되고 있습니다.

Carrier frequency	30 kHz	TSOP4130
	33 kHz	TSOP4133
	36 kHz	TSOP4136
	38 kHz	TSOP4138
	40 kHz	TSOP4140
	56 kHz	TSOP4156

그림 2-3 캐리어 주파수와 수신 장치의 관계

NS-RX231에 장착되어 있는 적외선 수신소자(TSOP4138)는 38kHz의 캐리어 주파수에 대응하는 특성을 가지고 있기 때문에, 타이머는 38kHz를 생성하기 위해 약13.15us마다 인터럽트를 발생시켜 파형을 만들게 됩니다.

3 프로젝트

3.1 신호 포맷

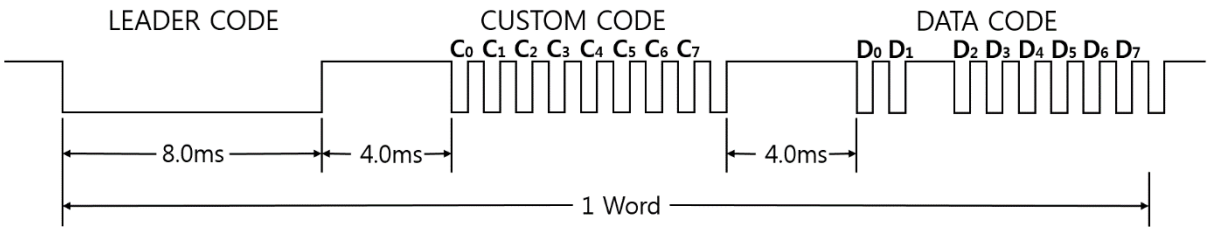

그림 3-1 리모컨 신호 포맷 예

우선, 간단한 신호 포맷을 예로 들어 프로젝트를 제작해보았습니다.

신호 포맷은, 통신의 시작을 알리는 리더코드 이후에 리모컨기기의 구분을 위한 사용자 정의 (Custom) 코드, 데이터를 전달하기 위한 데이터 코드로 구성되어있습니다.

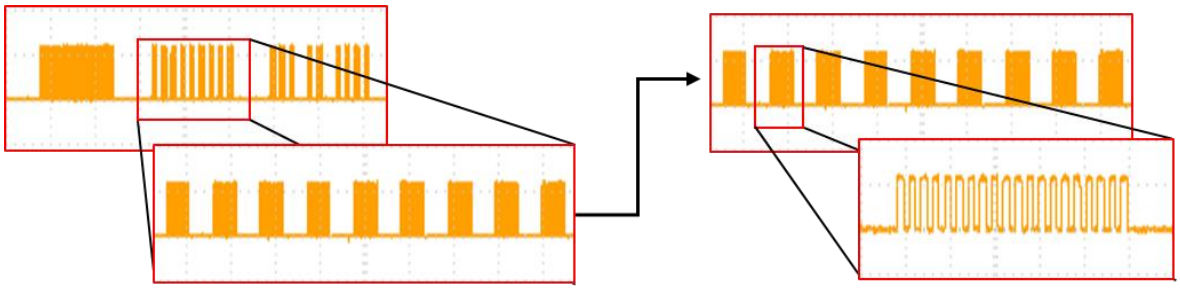

그림 3-2 적외선통신 송신시 파형

그리고 비트 0과 1의 차이는 다음과 같습니다.

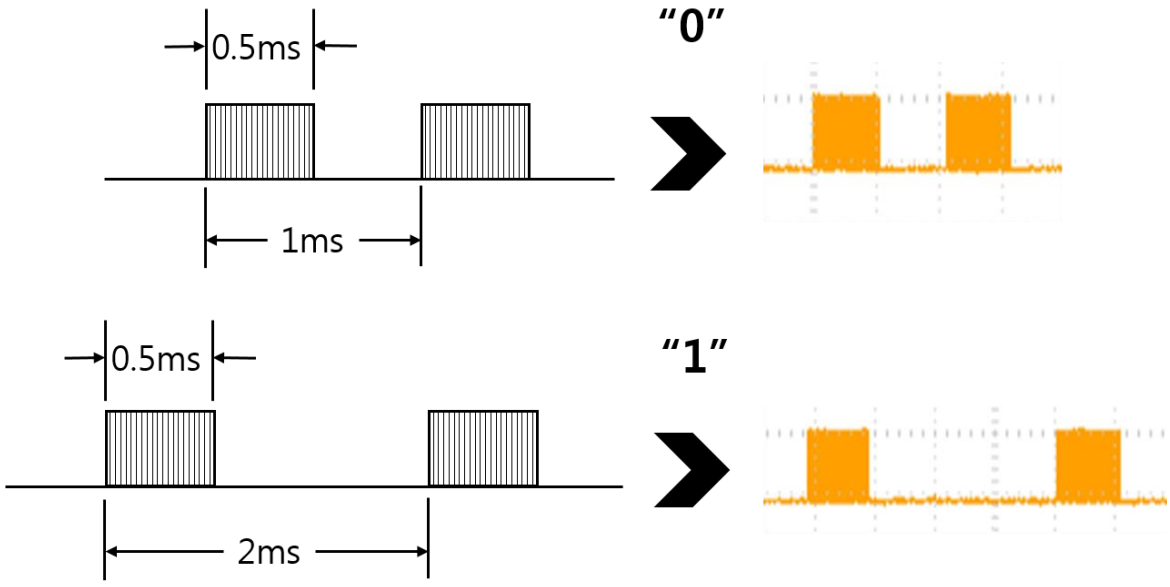

그림 3-3 적외선 통신 송신시 0과 1의 구분

3.2 소스코드

r_main.c

```
if (_0_SUCCESS == R_Set_Cap_Touch_Result_Create( method ))
{
    ts_result = R_Get_Cap_Touch_Result( method );
    if(1 == ts_result.button[1] >> 0)
    {
        PORTB.PODR.BYTE = 0x02;
        sendArr = IR_send(232,40);
    }
    else if(1 == ts_result.button[1] >> 6)
    {
        PORTB.PODR.BYTE = 0x08;
        sendArr = IR_send(232,24);
    }
    else if(1 == ts_result.button[1] >> 7)
    {
        PORTB.PODR.BYTE = 0x20;
        sendArr = IR_send(232,168);
    }
}
```

r_cg_tmr_user.c

```
if(sendArr[37] >= 1){
    sendArr[cnt]--;
    if(flg == 0){
        PORTA.PODR.BIT.B4 ^= 1;
    }

    if(sendArr[cnt] < 1){
        flg ^= 1;
        cnt++;
        if(cnt == 38){
            cnt = 0;
        }
    }
}
```

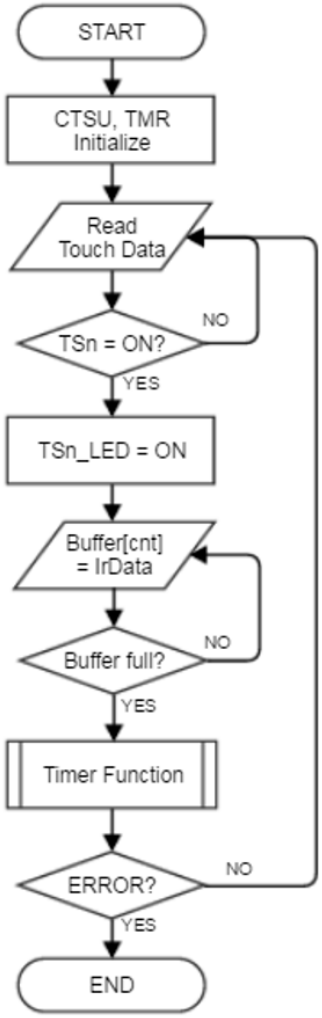

그림 3-4 소스코드 및 순서도

타이머인터럽트는 약 13us마다 실행되어 조건 식을 처리합니다. (38kHz의 한 주기는 26us) sendArr배열의 마지막 요소가 1이상일 때 데이터를 처리하기 시작하며, flg값이 0의 경우 포트의 출력을 반전시켜 38kHz의 파형을 생성하지만, 1의 경우 배열의 요소의 지정된 만큼 대기합니다.

동작 예로는 다음과 같습니다. 8ms의 리더코드를 생성하기 위해 sendArr배열의 0번째 요소를 참조하면 616이 되고, 포트 출력반전을 616번(13us * 616 = 8008us) 반복하여 38kHz의 파형을 만들게 됩니다. 출력을 반전시킬 때마다 배열의 요소를 1씩 감산하고, 값이 0이되면 flg값을 반전시킨 이후 sendArr배열의 다음 요소를 참조하도록 변수 cnt값을 1증가 시킵니다.

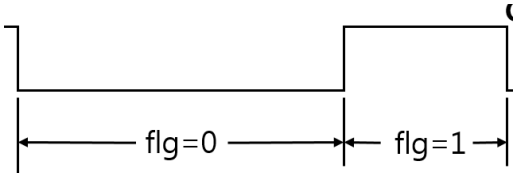

그림 3-5 신호 파형과 flag의 관계

4 디버깅

그림 4-1 NS-RX231을 전원 어댑터와 E1디버거를 연결한 모습.

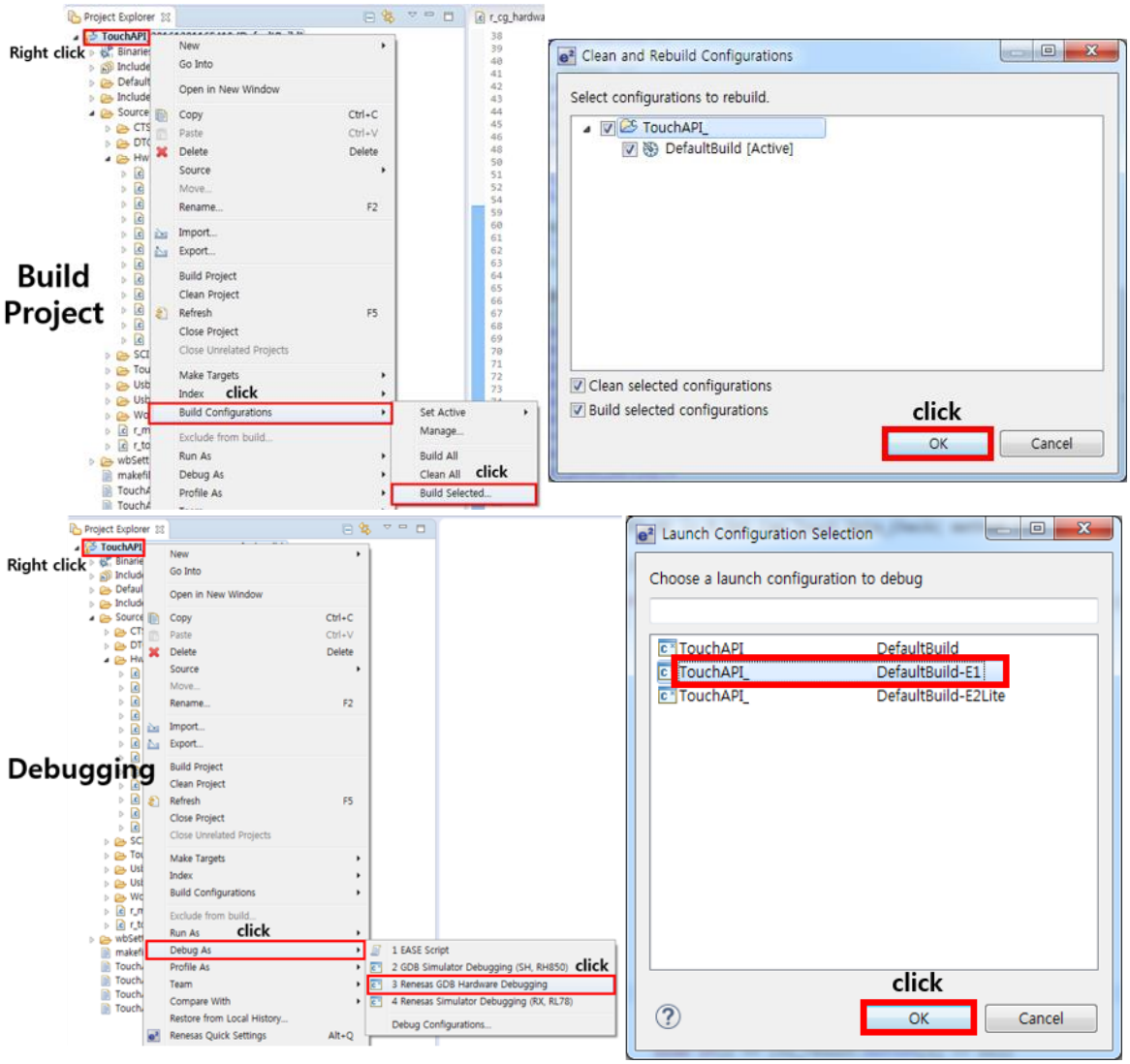

그림 4-2 프로젝트 빌드 및 디버깅

5 실행

위 소스코드를 보드에 업로드 한 후, TS16을 터치 해 봅시다.

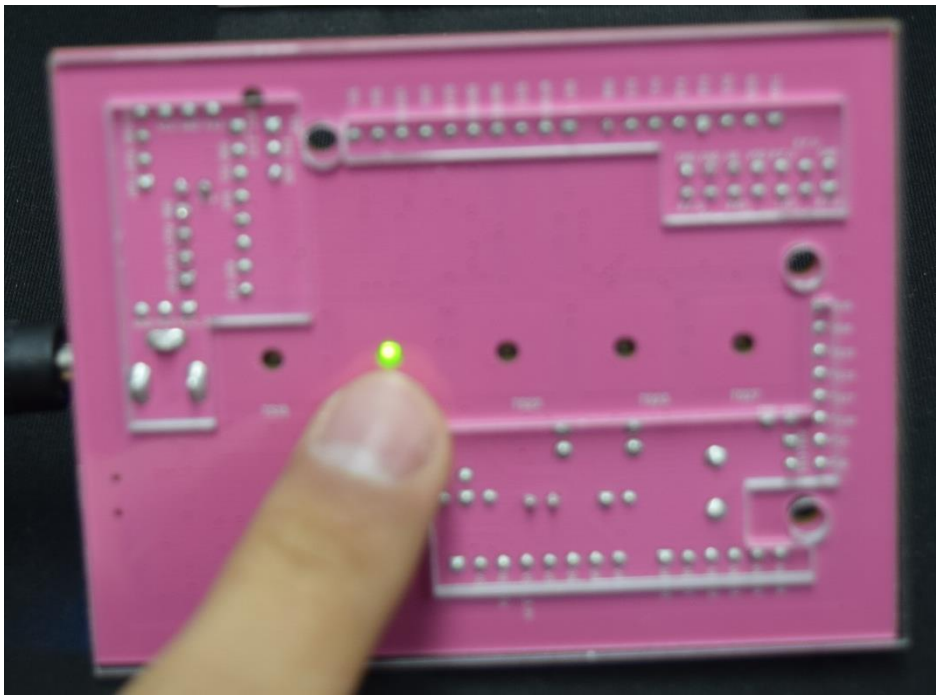

그림 5-1 TS16번 터치

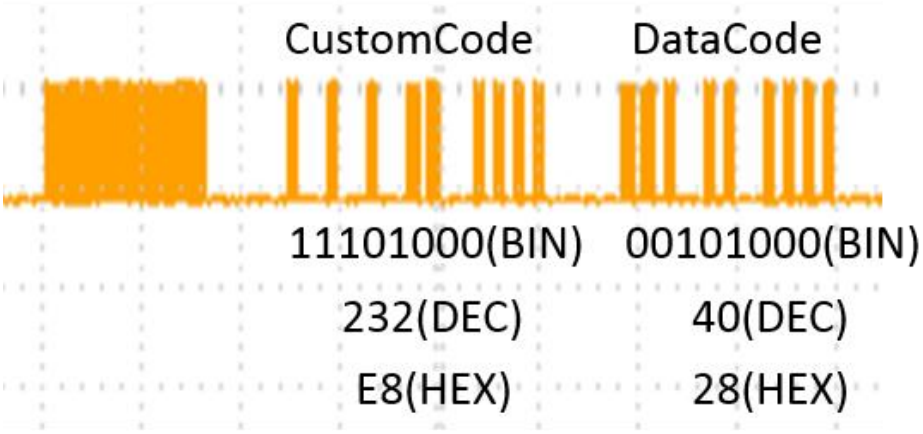

그림 5-2 오실로스코프에서 관측된 파형

오실로스코프에서 위 사진과 같은 파형이 관측되었습니다.

6 회로도

아래 사진은 적외선 리모컨 송신부의 회로도 입니다.

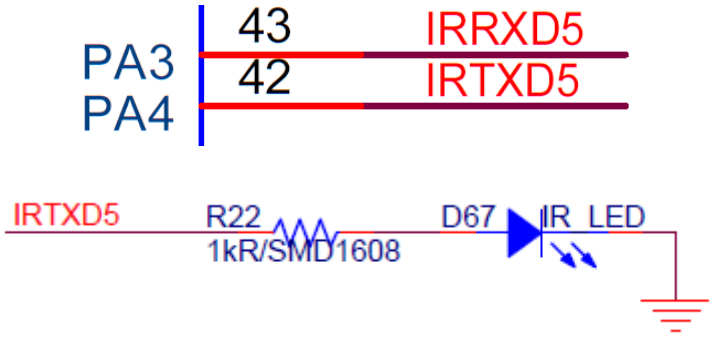

그림 6-1 NS-RX231의 적외선 송신부 회로도

7 추가

만약 리더코드의 길이를 다르게 하고 싶거나, 데이터코드의 오차를 줄이기 위해 데이터코드를 보내고 반전된 데이터코드를 보내 검사하고 싶거나, 자신이 만든 다른 프로토콜로 하고 싶다면, r_detectIR.c 소스폴더에서 IR_send부분을 수정하면 됩니다.

returnArr[38]은 함수에서 반환해줄 배열이며, 38은 배열의 크기를 지정합니다.

배열의 내용은 다음과 같습니다

리더코드 8ms는 returnArr[0] = 616
리더코드 4ms는 returnArr[1] = 308

커스텀코드 C0 returnArr[2] = 38
커스텀코드 C0 returnArr[3] = 1이면 116, 0이면 38

…

커스텀코드 C7 returnArr[17] = 1이면 116, 0이면 388

끝을 알리는 짧은 파형 = 38
대기시간 4ms = 308

1과 0을 구분하는 한 비트는 짧은 파형(0.5ms)을 내보내는 첫 번째 인수와 1과 0을 구분하는 (0.5ms or 1.5ms) 두 번째 인수를 한 쌍으로 이루어져 있고, 총 8개의 비트가 1바이트를 이루기 때문에 16개의 인수가 필요합니다. 그리고 한 바이트의 끝을 알리는 종료코드도 두 개의 인수가 한 쌍을 이루기 때문에 한 바이트의 데이터코드를 추가하고 싶다면 총 18개의 배열 요소가 추가로 필요하게 됩니다.

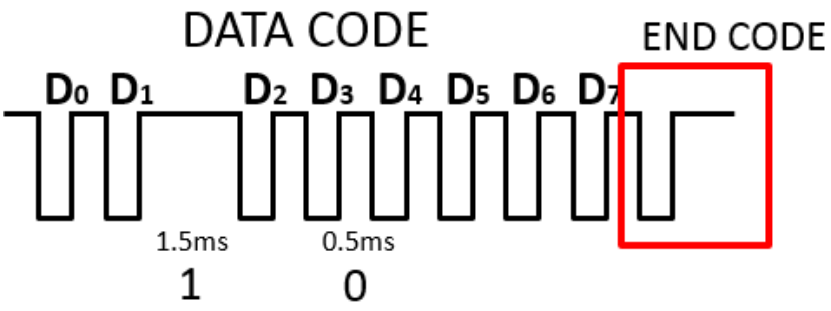

그림 7-1 1바이트 데이터코드 예

※ 추가하는 각각의 배열 요소 값은 반드시 짝수로 해야 합니다.