

Chapter 215. IR - 적외선 리모컨

1. 적외선

흔히 IR이라 불리는 적외선(Infrared, IR)은 가정에서 리모컨으로 흔히 사용되고 있으며 10미터 이내의 근거리에서 전자 기기를 제어하기 위한 간단한 방법 중 하나이다. 아두이노와 함께 사용하는 경우에도 싼 가격에 간단하게 무선 조정 장치를 만들 수 있다. 그림 1은 파장에 따라 전자기파를 분류해 놓은 것이다.

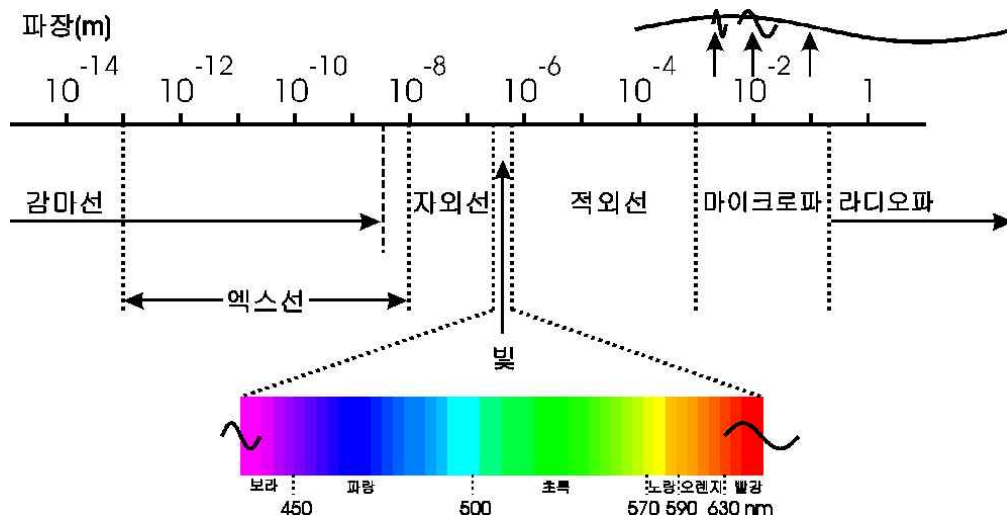


그림 1. 파장에 따른 빛

그림 1에서 볼 수 있듯이 적외선은 가시광선의 붉은색 영역 바로 옆에 위치하여 가시광선과 그 특성이 비슷하지만 파장이 약간 더 길다. 또한 전자레인지에서 사용하는 마이크로파에 비해서는 파장이 짧다. 적외선은 사람의 눈으로는 확인할 수 없지만 스마트폰의 카메라로는 확인할 수 있다. 옆에 리모컨과 스마트폰이 있다면 버튼을 눌렀을 때 리모컨의 LED로 빛이 나오고 있음을 확인해 보자. 최근 판매되는 리모컨 중에는 버튼을 눌렀을 때 LED가 깜빡거리는 것을 확인할 수 있는 리모컨이 있지만 이는 동작 상태를 알려주기 위한 것이며 실제 정보가 전달되는 것은 눈에 보이지 않는 적외선에 의한 것이다.

리모컨의 버튼을 누르면 리모컨은 초당 38,000회 깜빡거리면서 데이터를 전송한다. 적외선은 자연환경에서도 흔히 볼 수 있지만 38KHz 주파수 대역의 적외선은 거의 없으므로 데이터 전달을 위해 흔히 사용되고 있다. 대부분의 리모컨은 38KHz 대역을 사용하지만 일부 다른 주파

수 대역을 사용하는 경우도 있다.

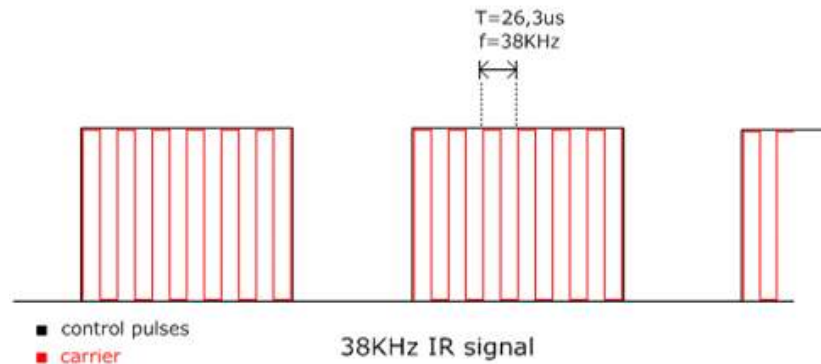


그림 2. 리모컨에서 38KHz로 변조된 신호

적외선 신호를 수신하는 장치에서는 그림 2의 신호를 그림 3과 같이 펄스열로 바꾸는 역할을 하며 마이크로컨트롤러에서 펄스열로부터 전달된 데이터를 알아낸다.

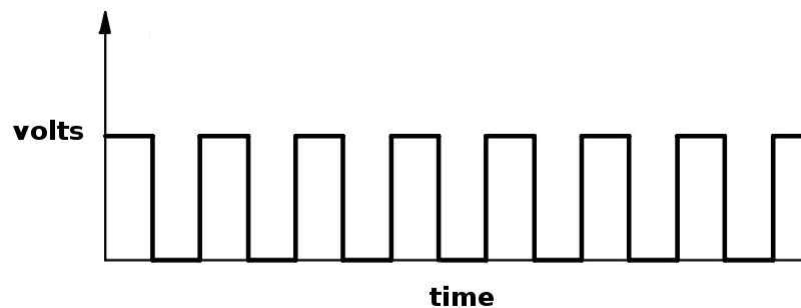


그림 3. 적외선 수신 장치에서 복조된 신호

2. 적외선 통신을 위한 준비

적외선 통신에서 적외선 신호를 보내기 위해서는 적외선 LED가 흔히 사용된다. 적외선 LED는 일반 LED와 동일한 형태를 가지므로 외형만으로는 구별할 수 없다. 일반 LED와 마찬가지로 다리가 긴 쪽이 (+), 짧은 쪽이 (-)이다.



그림 4. 적외선 LED

적외선 수신기인 PL-IRM0101은 3개의 핀을 가지며 2번과 3번 핀을 GND와 VCC에 연결하면 1번 핀으로 수신된 데이터를 확인할 수 있다.



그림 5. PL-IRM0101 적외선 수신기

아두이노에서 적외선 송수신을 위해서는 IRremote 라이브러리를 사용할 수 있다. IRremote 라이브러리를 다운 받아¹⁾ 아두이노 스케치 디렉터리 아래 libraries 디렉터리에 'IRremote' 디렉터리를 만들어서 설치하자. 이로써 모든 준비는 끝났다. 이제 본격적으로 적외선 통신을 해보자.

현재 아두이노의 최신 버전은 1.6.0이다. 아두이노 최신 버전에서는 여러 가지 라이브러리를 제공하고 있으며 이 중 하나가 아두이노 로봇을 위한 RobotIRremote 라이브러리이다. RobotIRremo 라이브러리는 IRremote 라이브러리를 바탕으로 만들어져 있지만 아두이노 로봇 전용으로 만들어져 있어 이 장에서 설명하는 내용과 충돌이 발생하여 컴파일이 정상적으로 이루어지지 않는다. 이 장의 내용을 실험해보고 싶다면 아두이노

1) <https://github.com/shirriff/Arduino-IRremote>

설치 디렉터리 아래 'libraries\RobotIRremote' 디렉터리를 제거하거나 다른 곳으로 옮겨 놓은 후 컴파일 하여야 정상적으로 동작하는 것을 확인할 수 있다.

3. IRremote 라이브러리 사용

먼저 적외선 수신기를 그림 6과 같이 아두이노 우노에 연결하자.

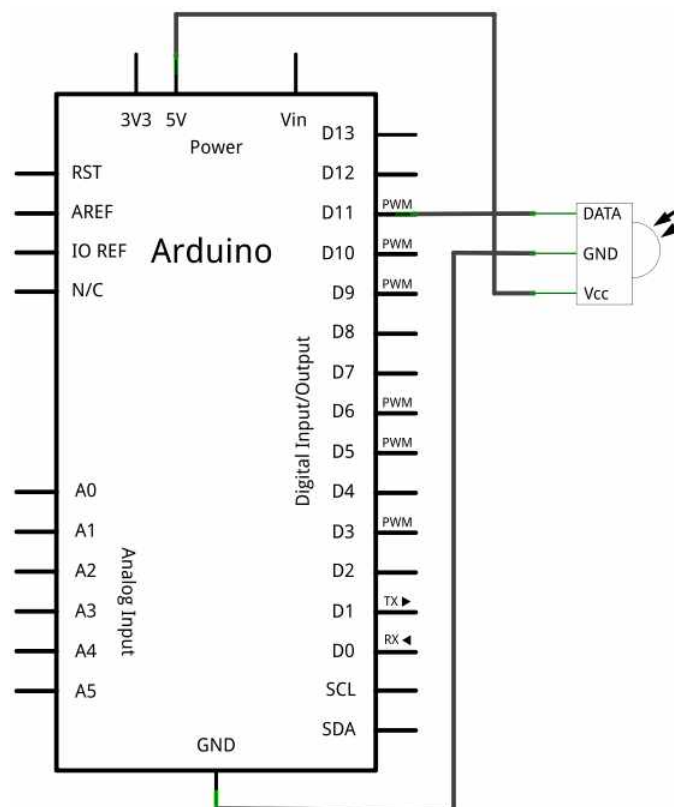


그림 6. 적외선 수신기 연결 회로도

스케치 1을 업로드 하고 시리얼 모니터를 실행시킨다. 리모컨 버튼을 수신기 쪽으로 향하고 버튼을 눌러 시리얼 모니터로 출력되는 값을 확인해 보자. 실험에 사용한 리모컨은 그림 7의 리모컨이며, 주변에서 흔히 볼 수 있는 일반 TV 리모컨을 사용하여도 무방하다. 단, 리모컨의 종류에 따라 시리얼 모니터로 출력되는 값은 달라진다.



그림 7. 리모컨

스케치 1. 리모컨 테스트

```
#include <IRremote.h>
#include <IRremoteInt.h>

int RECV_PIN = 11;                // 적외선 수신기의 DATA 핀 연결

IRrecv irrecv(RECV_PIN);          // 적외선 수신기 초기화
decode_results results;            // 수신 데이터

void setup()
{
  Serial.begin(115200);             // 시리얼 통신 초기화
  irrecv.enableIRIn();              // 적외선 수신기 시작
}

void loop()
{
  if(irrecv.decode(&results)) {    // 디코딩 결과
    Serial.print("Received data : 0x");
    Serial.println(results.value, HEX); // 수신 데이터 출력
  }
}
```

```

    irrecv.resume();                // 다음 데이터 수신
}

delay(200);
}

```

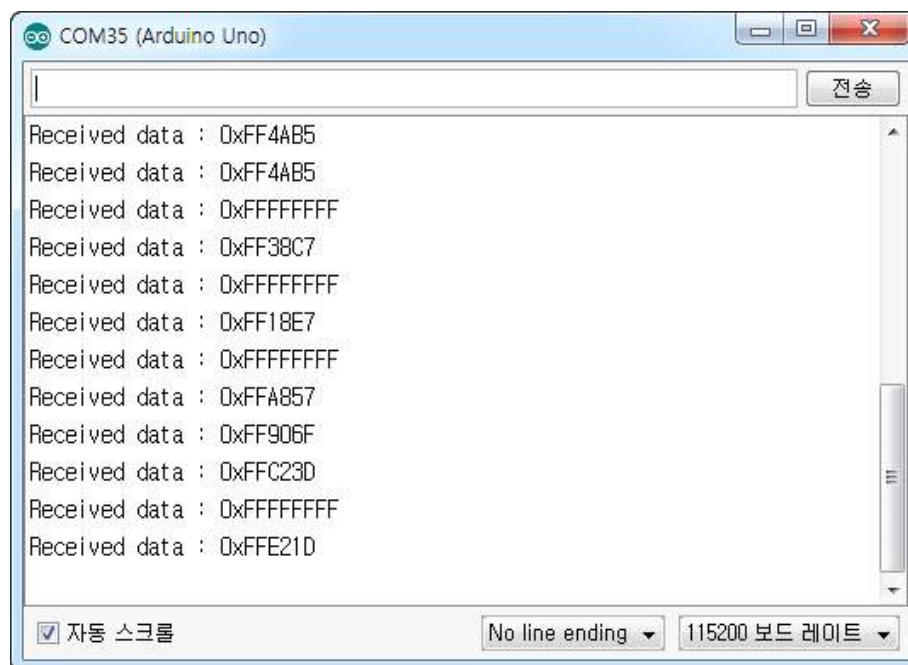


그림 8. 스케치 1 실행 결과

스케치 1에서 볼 수 있듯이 IRremote 라이브러리를 사용하여 리모컨의 데이터를 수신하는 과정은 간단하다. 먼저 적외선 수신기의 데이터 핀이 연결된 핀을 지정하여 객체를 생성하고, enableIRIn 멤버함수로 초기화를 수행하면 준비는 끝난다. 실제 데이터 수신은 decode와 resume 멤버함수를 통해 가능하다. 수신되어 해석된 데이터는 전용 클래스인 decode_results 클래스에 저장된다. decode_results 클래스에는 여러 가지 멤버변수들이 있지만 이 중 수신된 데이터가 저장되는 변수는 value 멤버변수로 unsigned long의 4바이트 크기를 가지는 변수이다. 즉, 일반적으로 리모컨의 버튼 하나를 누르면 4바이트의 데이터가 전송된다. 리모컨 데이터는 리모컨 데이터의 시작을 알려주는 리드(lead) 코드, 2바이트의 커스텀(custom) 코드 그리고 2바이트의 데이터 코드로 이루어진다. 리드 코드는 리모컨 데이터의 시작을 나타내는 신호로, 데이터를 나타내기 위해 사용되는 펄스폭보다 긴 폭을 가지는 ON/OFF 신호로 구성된다. 리드 코드 이후의 4바이트 데이터는 펄스 거리 인코딩 (Pulse

Distance Encoding) 방식이 흔히 사용된다. 펄스 거리 인코딩은 상향 에지 사이의 시간을 통해 논리 0과 논리 1을 구별하는 방법을 말한다.

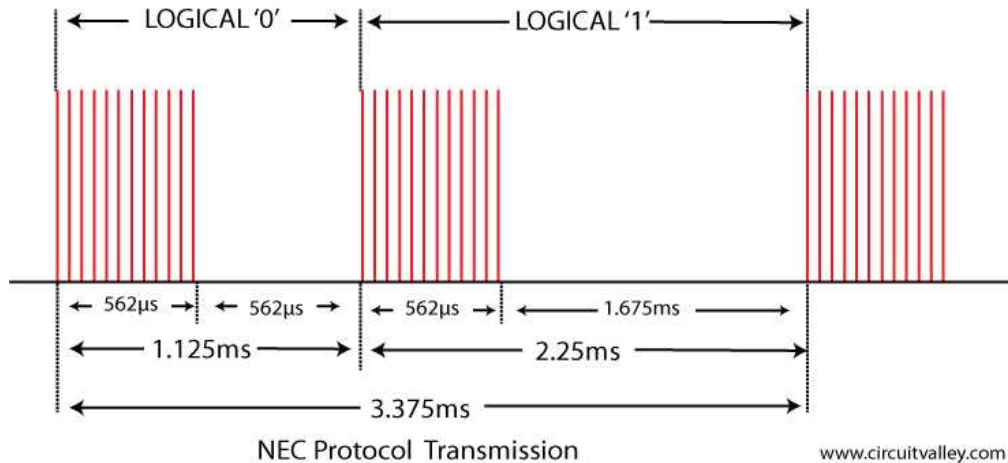


그림 9. 펄스 거리 인코딩 (NEC 프로토콜의 예)

제조 회사에 따라 리드 코드와 데이터의 길이 및 인코딩 방법에 차이가 있을 수 있으므로 사용하고자 하는 리모컨에서 사용하는 방식을 알고 있어야 한다. 하지만 IRremote 라이브러리에서는 다양한 방식으로 표현되는 리모컨 데이터에서 리드 코드를 인식하고 4바이트의 데이터를 찾아내어 반환하므로 데이터의 의미에만 신경을 쓰면 된다. 위에서 설명한 리드 코드 및 데이터 길이는 리모컨에서 흔히 사용되는 NEC의 방식에 해당한다.

수신된 4바이트 중 커스텀 코드 2바이트는 제조사 및 제품의 종류에 따라 결정되며, 데이터 코드는 해당 제품의 리모컨 키 값을 담고 있다. 데이터 코드는 8비트만 있고 뒤에 8비트는 반전된 값이다. 그림 8에서 마지막에 수신된 4바이트 데이터는 0x00FFE21D의 4 바이트로 커스텀 코드는 0x00FF로 별다른 의미가 없으며, 데이터 코드는 0xE21D로 $0xE2 + 0x1D = 0xFF$ 로 반전된 데이터로 이루어져 있음을 확인할 수 있다. 0xFFFFFFFF 값은 이전에 눌러진 버튼이 계속해서 눌러져 있는 상태를 나타낸다.

적외선 리모컨 데이터를 수신하여 해석하는 방법을 살펴보았으니 이제 적외선 데이터를 송신하는 방법에 대해 알아보자. 적외선 리모컨 데이터 송수신을 위해서는 두 개의 아두이노를 사용한다. 데이터 수신 장치는 그림 6의 회로를 사용하며, 데이터 송신 장치를 위해서는 그림 10과 같이 버튼과 적외선 LED를 연결하여 사용한다.

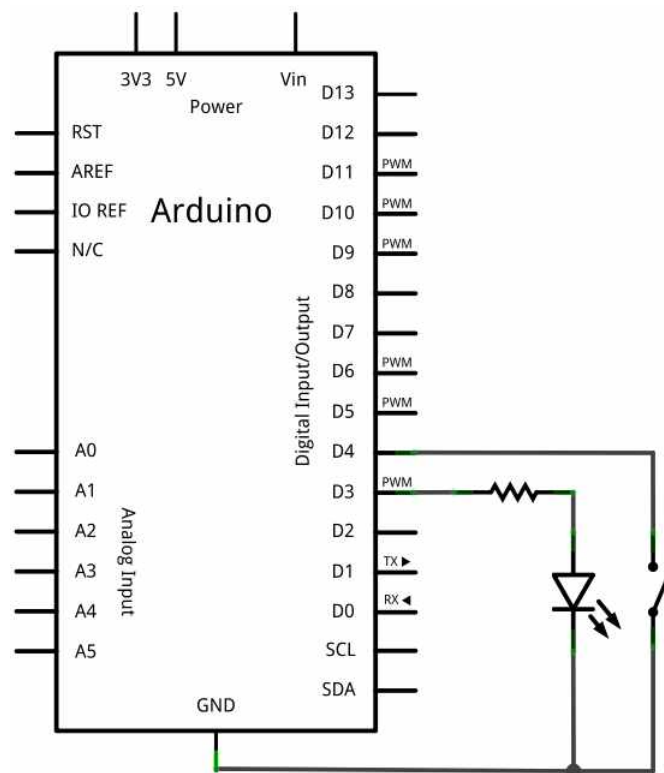


그림 10. 적외선 송신기 연결 회로도

적외선 LED를 연결할 때 주의할 점은 PWM 출력이 가능한 핀에 연결하여야 한다는 점으로, IRremote 라이브러리에서는 디지털 3번 핀을 사용하므로 그림 10에서도 디지털 3번 핀에 적외선 LED가 연결되어 있다. 디지털 4번 핀의 버튼을 누르면 적외선 LED로 데이터를 전송하며 송신된 데이터는 수신 장치에서 수신하여 시리얼 모니터로 출력한다. 적외선 데이터 송신을 위해서는 먼저 전용 클래스인 IRsend 클래스의 객체를 생성하여야 한다. NEC 방식에서는 4바이트의 데이터를 전송하므로 sendNEC 멤버함수를 사용하면 된다. 스케치 2는 내부 풀업 저항을 사용하여 연결된 버튼이 눌려진 경우 4바이트의 데이터를 적외선 통신을 통해 전송하는 예이다.

스케치 2. 적외선 송신

```
#include <IRremote.h>
#include <IRremoteInt.h>

IRsend irsend;
```

```
int button_pin = 4;

void setup()
{
  Serial.begin(115200);           // 시리얼 통신 초기화
  pinMode(button_pin, INPUT_PULLUP); // 내부 풀업 저항을 사용한 버튼 연결
}

void loop()
{
  if(!digitalRead(button_pin)){    // 버튼이 눌려진 경우
    Serial.println("** Data Sending...");
    irsend.sendNEC(0x18E758A7, 32); // NEC 방식의 4바이트 데이터 전송
    delay(200);
  }
}
```

이외에도 다양한 멤버함수들이 준비되어 있으므로 사용하고자 하는 전송 방식에 맞는 멤버함수를 사용하면 된다.

```
void sendNEC(unsigned long data, int nbits);
void sendSony(unsigned long data, int nbits);
void sendRaw(unsigned int buf[], int len, int hz);
void sendRC5(unsigned long data, int nbits);
void sendRC6(unsigned long data, int nbits);
void sendDISH(unsigned long data, int nbits);
void sendSharp(unsigned int address, unsigned int command);
void sendSharpRaw(unsigned long data, int nbits);
void sendPanasonic(unsigned int address, unsigned long data);
void sendJVC(unsigned long data, int nbits, int repeat);
```

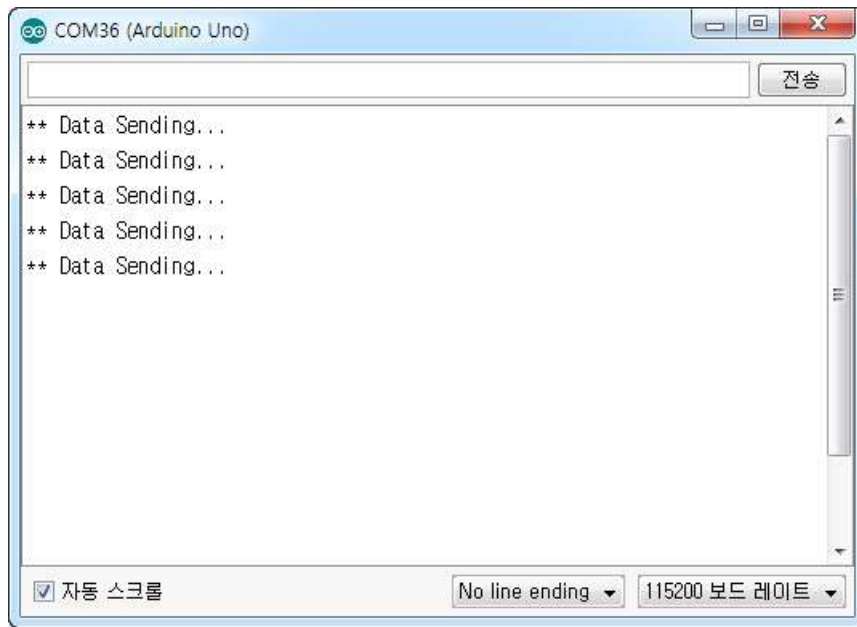


그림 11. 스케치 2 실행 결과 - 송신 장치

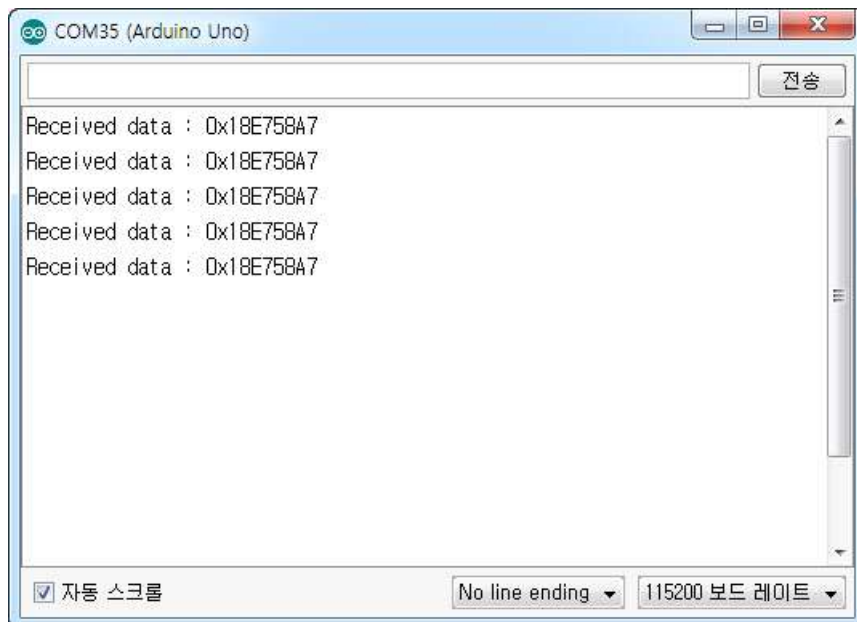


그림 12. 스케치 2 실행 결과 - 수신 장치