

2022년 IoT기반 스마트 솔루션 개발자 양성과정



Firmware [펌웨어]

22-IR Remote

담당 교수 : 유근택

010-5486-5376

rgt3340@naver.com



충북대학교 공동훈련센터

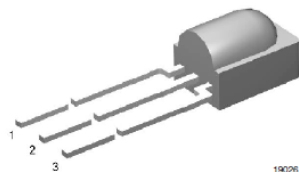
TSOP38x IR Receiver

TSOP382.., TSOP384..

Vishay Semiconductors



IR Receiver Modules for Remote Control Systems



19020

MECHANICAL DATA

Pinning:

1 = OUT, 2 = GND, 3 = V_S

FEATURES

- Very low supply current
- Photo detector and preamplifier in one package
- Internal filter for PCM frequency
- Improved shielding against EMI
- Supply voltage: 2.5 V to 5.5 V
- Improved immunity against ambient light
- Insensitive to supply voltage ripple and noise
- Component in accordance to RoHS 2002/95/EC and WEEE 2002/96/EC



RoHS
COMPLIANT

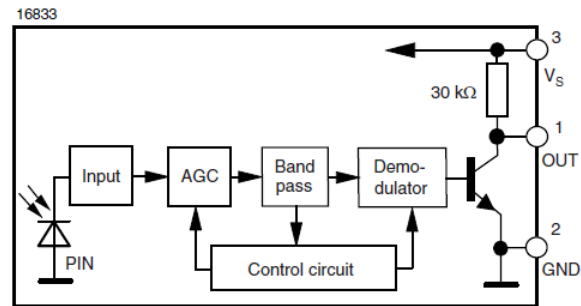
DESCRIPTION

The TSOP382.., TSOP384.. series are miniaturized receivers for infrared remote control systems. A PIN diode and a preamplifier are assembled on a lead frame, the epoxy package acts as an IR filter.

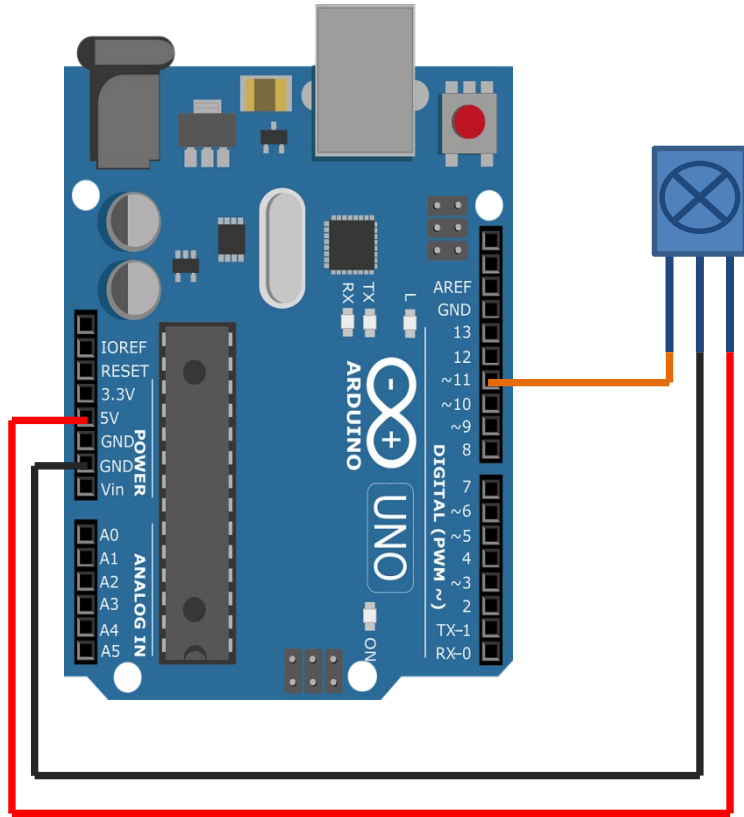
The demodulated output signal can be directly decoded by a microprocessor. The TSOP382.. is compatible with all common IR remote control data formats. The TSOP384.. is optimized to suppress almost all spurious pulses from energy saving fluorescent lamps but will also suppress some data signals.

This component has not been qualified according to automotive specifications.

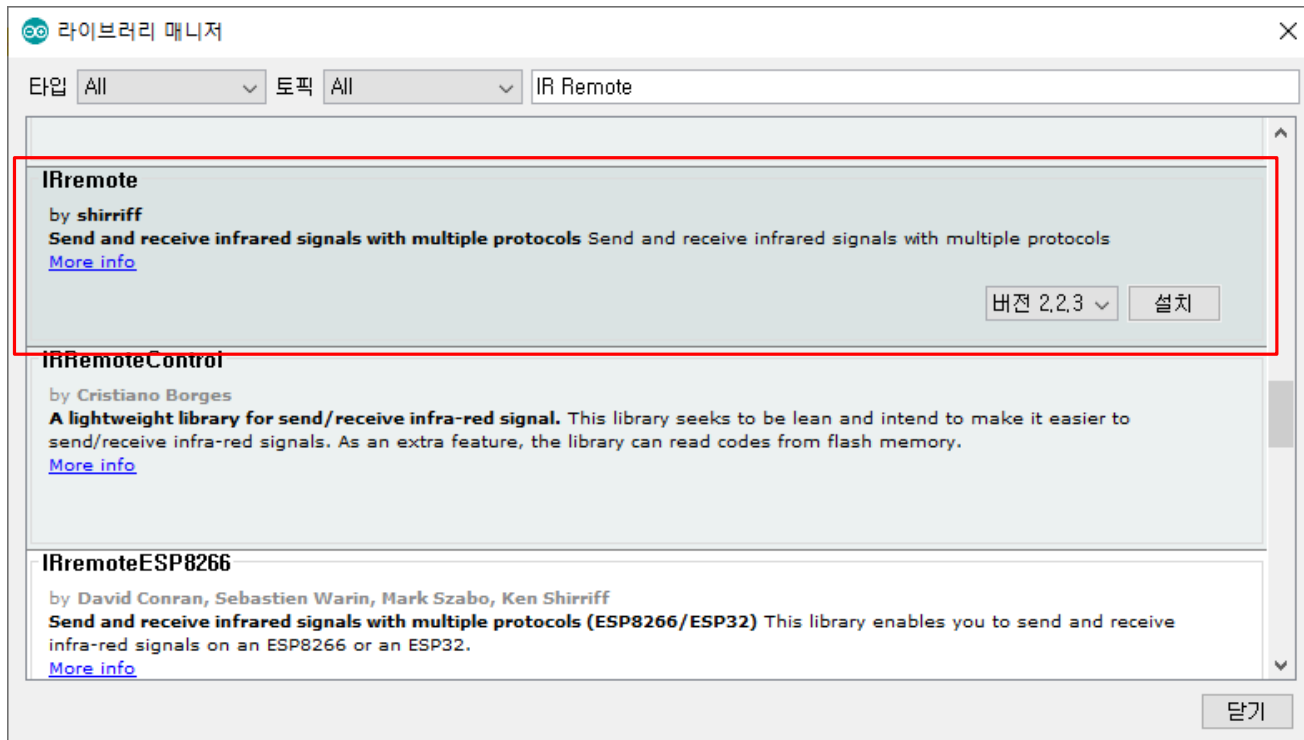
BLOCK DIAGRAM



Wiring















Library Manager



https://github.com/z3t0/Arduino-IRremote

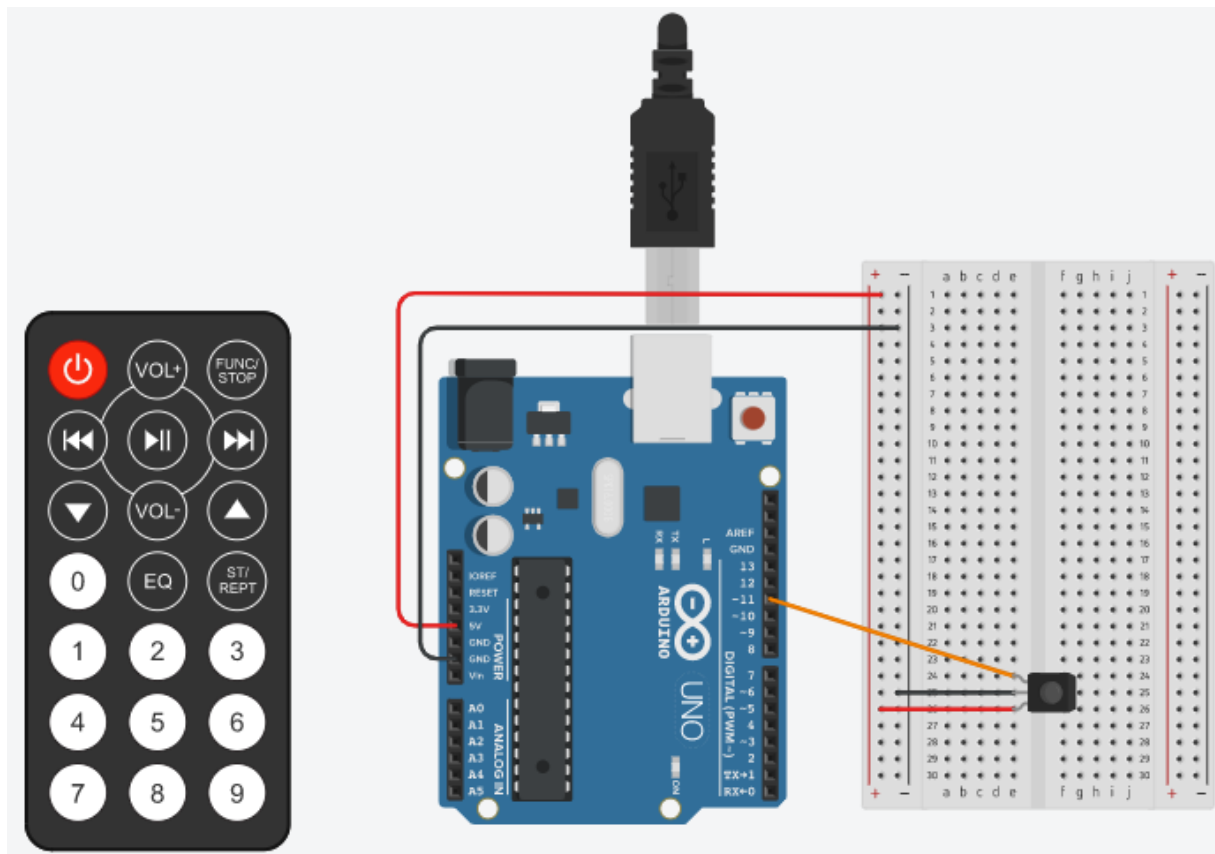
Branch: master ▾ New pull request Find file Clone or download ▾

 z3t0 Merge pull request #688 from Kruemmelspalter/patch-1 ... Latest commit 6c0a603 3 days ago

 examples	Add SAMSUNG to IRRecord.ino	3 days ago
 .gitignore	added sublime workspace to gitignore	4 years ago
 .travis.yml	Add Lego Power Functions tests	4 years ago
 Contributing.md	added ISSUE_TEMPLATE	4 years ago
 Contributors.md	Update Contributors.md (#488)	3 years ago
 IRremote.cpp	Cleaned up ESP32 integration, reverted ESP32 ifdefs on irreceive exam...	3 years ago
 IRremote.h	Add Lego Power Functions send protocol	4 years ago
 IRremoteInt.h	Move board specific configuration info to new file boarddefs.h.	4 years ago
 ISSUE_TEMPLATE.md	Update ISSUE_TEMPLATE.md	3 years ago
 LICENSE.txt	Initial commit from Irremote.zip	11 years ago
 README.md	Supported board order	3 years ago



IR_Remote-1



IR_Remote-1

```
#include <IRremote.h>

#define TSOP38x 11

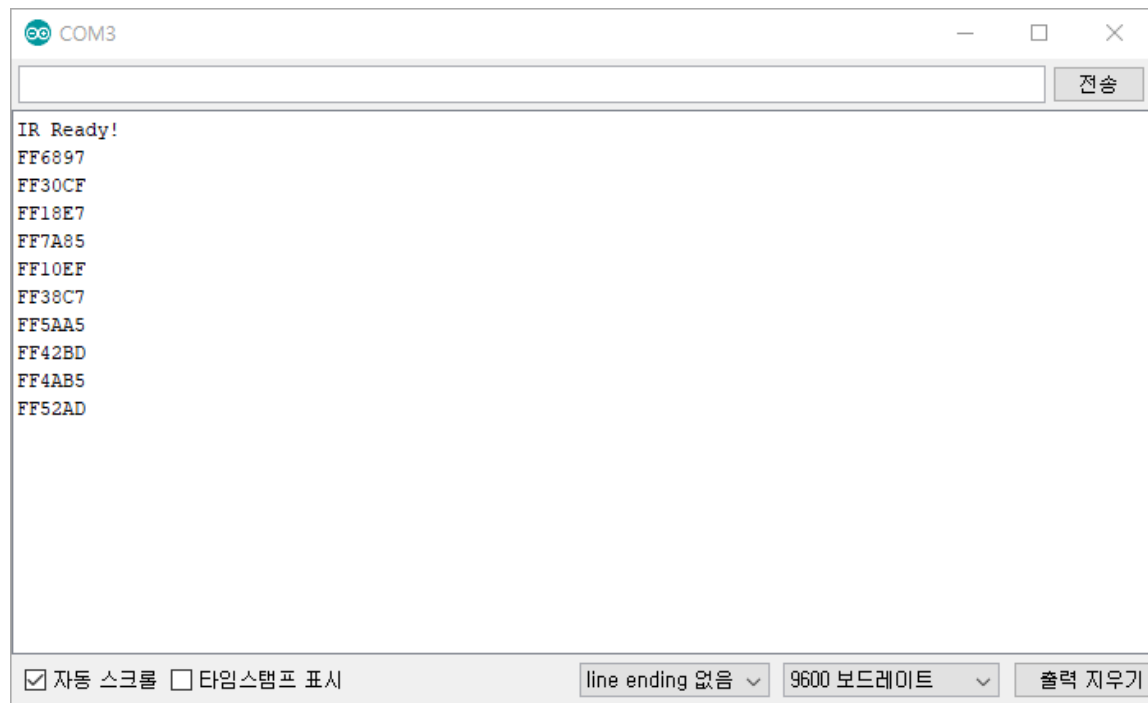
IRrecv IR(TSOP38x);      // 적외선 센서가 연결된 디지털 핀 번호 매핑
decode_results result;    // 수신된 적외선 신호를 저장할 변수

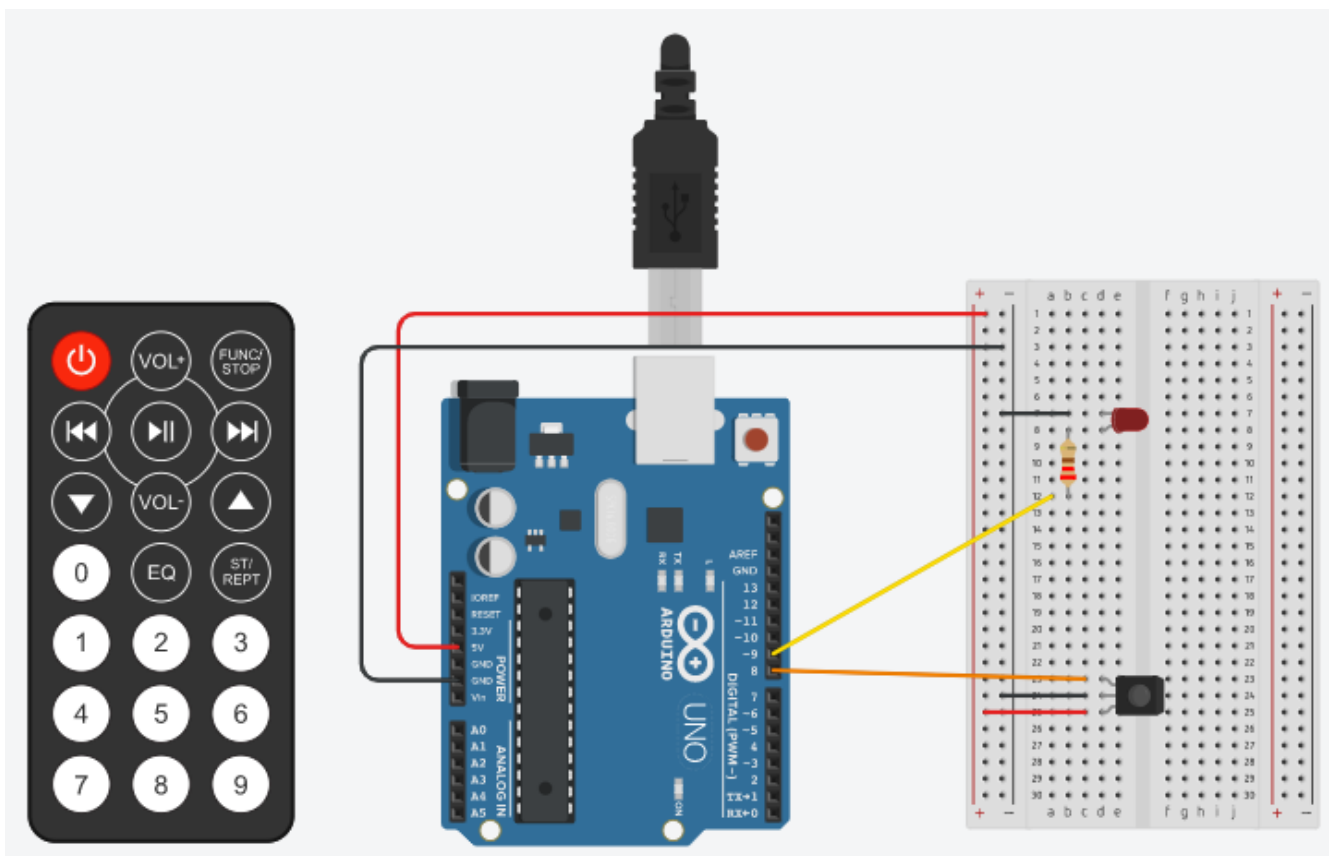
void setup() {
  IR.enableIRIn();        // 적외선 센서 활성화
  Serial.begin(9600);
  Serial.println("IR Ready!");
}

void loop() {
  if (IR.decode(&result)){
    Serial.println(result.value,HEX);
    IR.resume();
  }
  delay(250);
}
```



IR_Remote-1 : Serial Monitor





IR_Remote-2

```
#include <IRremote.h>

IRrecv irrecv(8);           // 적외선 센서가 연결된 디지털 핀 번호 매핑
decode_results results;      // 수신된 적외선 신호를 저장할 변수

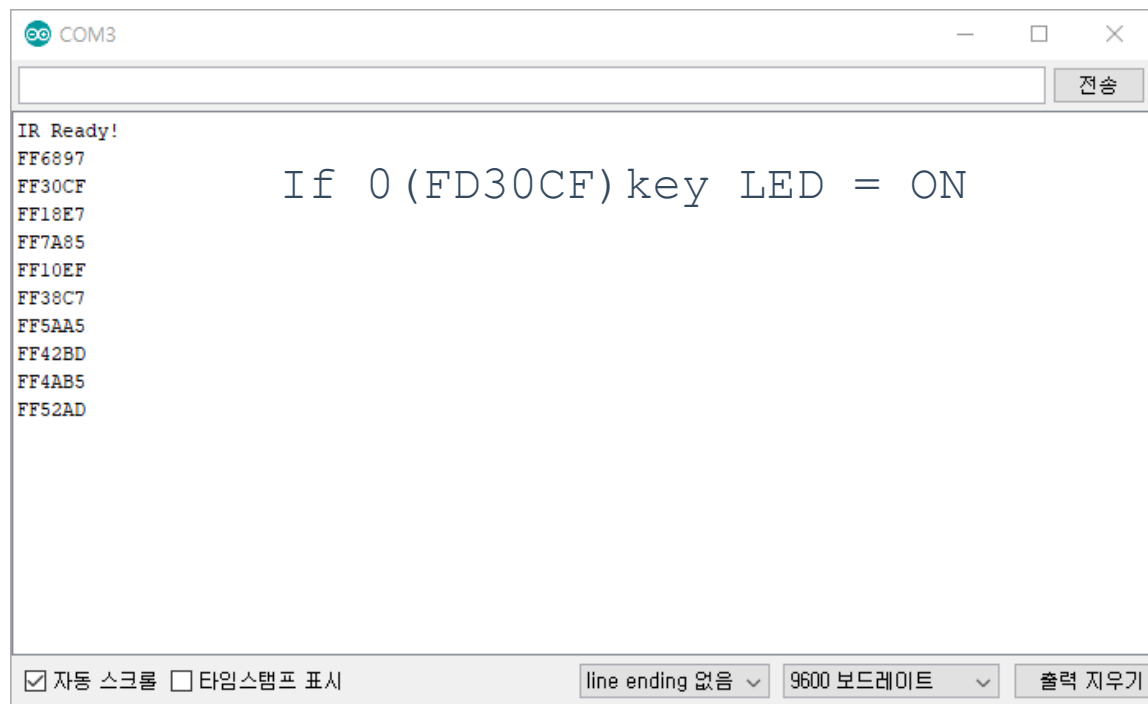
void setup() {
  irrecv.enableIRIn();       // 적외선 센서 활성화
  pinMode(9, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  if(irrecv.decode(&results)) { // 적외선 신호를 해석
    Serial.println(results.value, HEX); // 적외선 신호 값을 출력

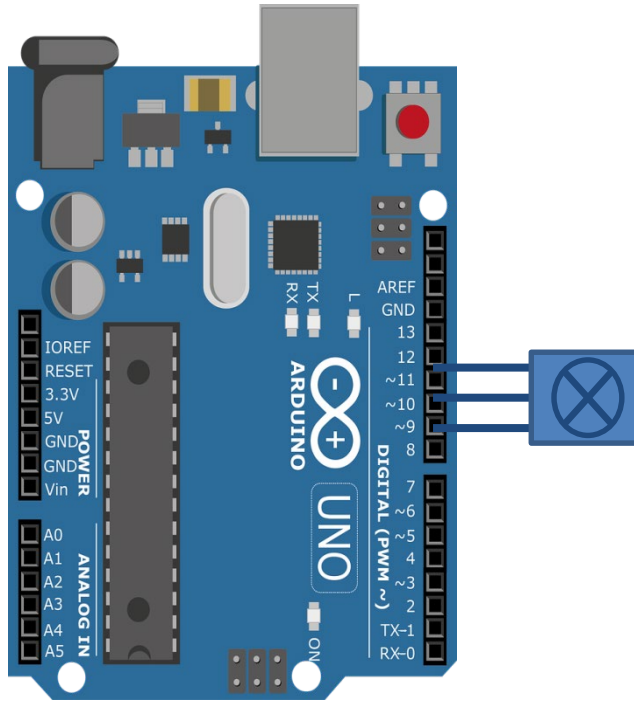
    if(results.value == 0xFD30CF) { // 숫자 0 버튼이 눌리면 LED를 켜
      digitalWrite(9, HIGH);       // 다른 버튼이 눌리면 LED를 끄
    }
    else {
      digitalWrite(9, LOW);
    }
    delay(30);
    irrecv.resume();              // 다음 신호를 받을 수 있도록 초기화
  }
}
```



IR_Remote-2 : Serial monitor



IR_Remote-3 Wiring



IR_Remote-3 : Info

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

#define TSOP38x 11
IRrecv IR( TSOP38x );
decode_results result;

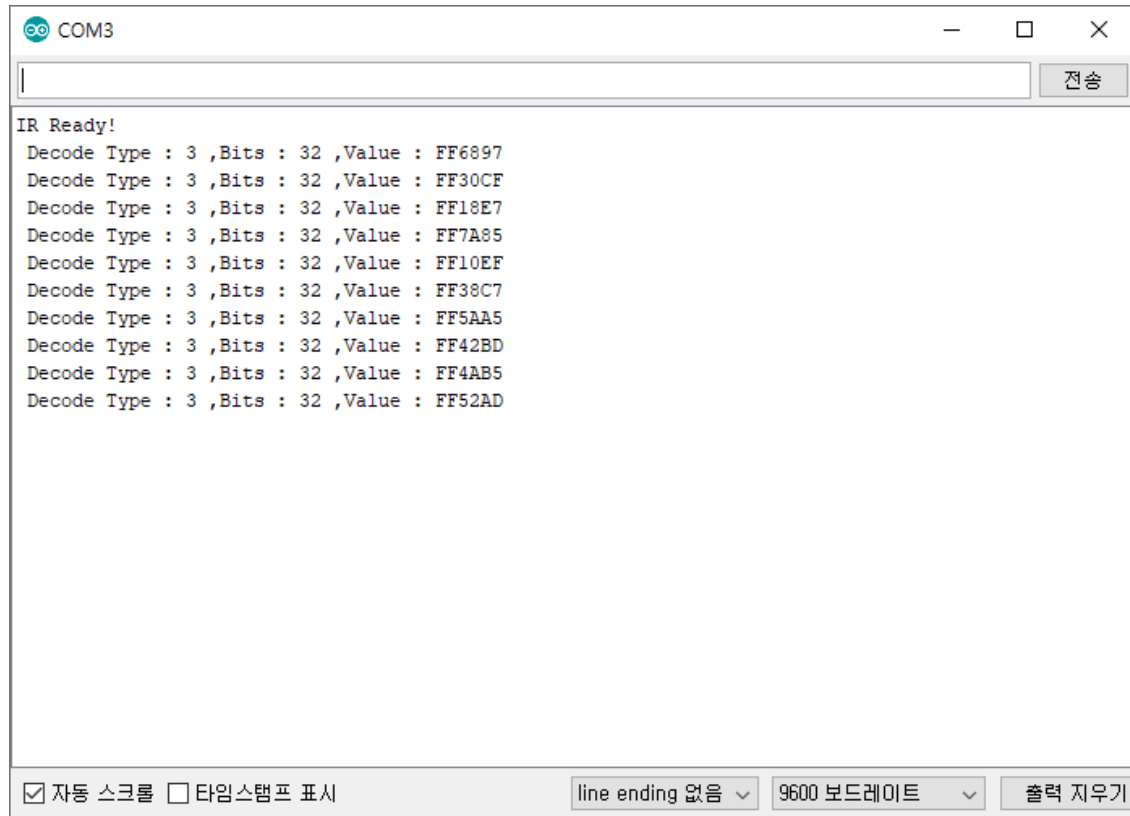
void setup( ) {
  pinMode(9,OUTPUT);
  digitalWrite(9,HIGH);
  pinMode(10,OUTPUT);
  digitalWrite(10,LOW);

  Serial.begin(9600);
  Serial.println("IR Ready!");
  IR.enableIRIn( );
}
```

```
void loop( ) {
  if ( IR.decode( &result ) ){
    Serial.print(" Decode Type : ");
    Serial.print( result.decode_type );
    Serial.print(" ,Bits : ");
    Serial.print( result.bits );
    Serial.print(" ,Value : ");
    Serial.println( result.value,HEX );
    IR.resume( );
  }
  delay(250);
}
```



IR_Remote-3 : Serial Monitor



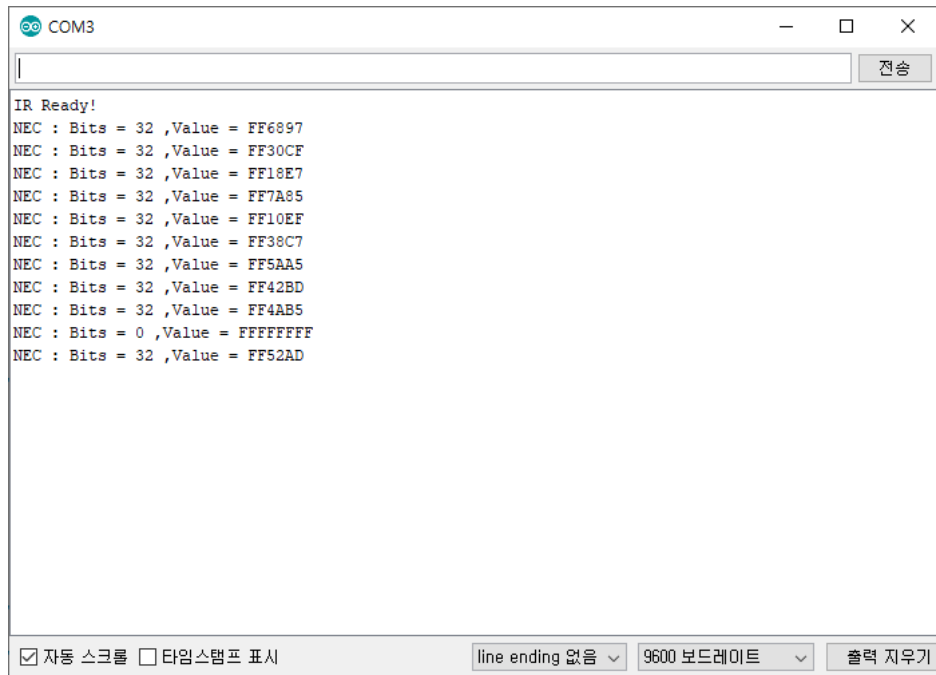
IR_Remote-4 : decode_type

```
void loop( ) {  
  if (IR.decode( &result )){  
    encoding( result.decode_type );  
    Serial.print("Bits = ");  
    Serial.print( result.bits );  
    Serial.print(" ,Value = ");  
    Serial.println( result.value,HEX );  
    IR.resume( );  
  }  
  delay(250);  
}
```

```
void encoding(int decode_types){  
  switch (decode_types) {  
    case UNKNOWN:  Serial.print("UNKNOWN : ");      break ;  
    case NEC:      Serial.print("NEC : ");           break ;  
    case SONY:     Serial.print("SONY : ");          break ;  
    case RC5:      Serial.print("RC5 : ");           break ;  
    case RC6:      Serial.print("RC6 : ");           break ;  
    case DISH:     Serial.print("DISH : ");          break ;  
    case SHARP:    Serial.print("SHARP : ");         break ;  
    case JVC:      Serial.print("JVC : ");           break ;  
    case SANYO:    Serial.print("SANYO : ");         break ;  
    case MITSUBISHI: Serial.print("MITSUBISHI : ");  break ;  
    case SAMSUNG:  Serial.print("SAMSUNG : ");       break ;  
    case LG:       Serial.print("LG : ");            break ;  
    case WHYENTER: Serial.print("WHYENTER : ");     break ;  
    case AIWA_RC_T501: Serial.print("AIWA_RC_T501 : "); break ;  
    case PANASONIC: Serial.print("PANASONIC : ");   break ;  
    case DENON:    Serial.print("Denon : ");        break ;  
  }  
}
```



IR_Remote-4 : Serial Monitor



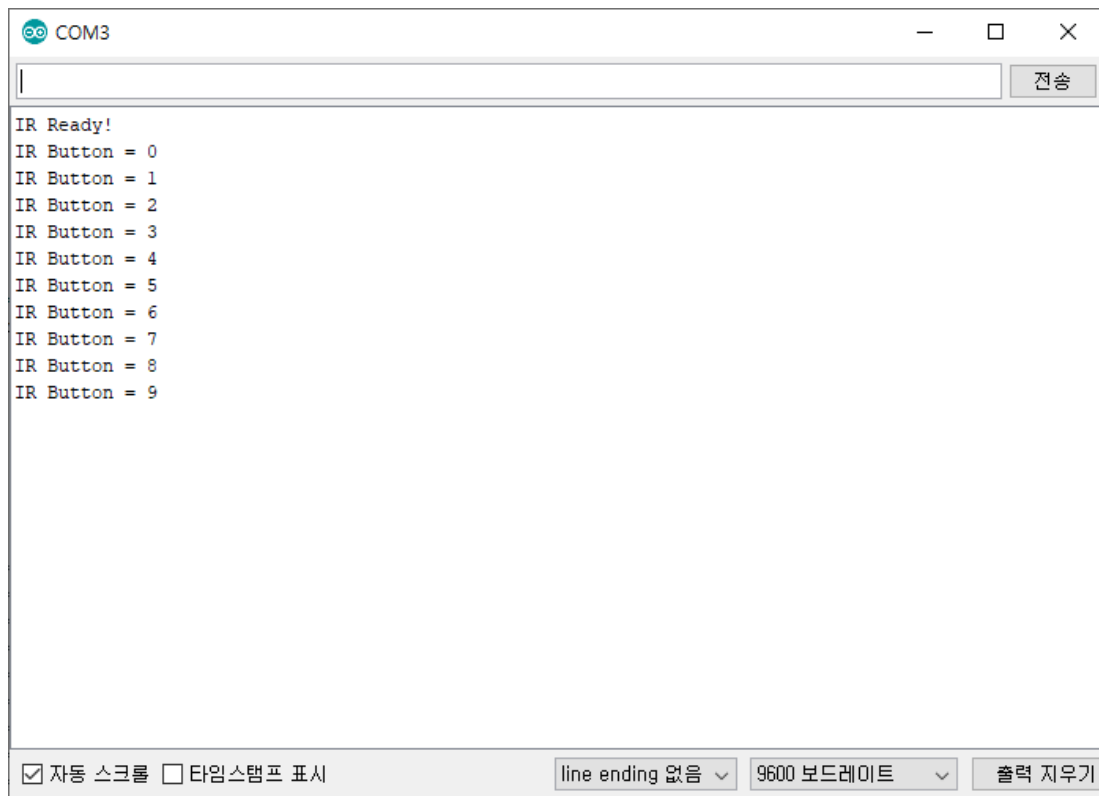
IR_Remote-5 : Buttons

```
void loop( ) {  
  if (IR.decode( &result )){  
    Serial.print("IR Button = ");  
    Serial.println( encoding(result.value) );  
    IR.resume( );  
  }  
  delay(250);  
}
```

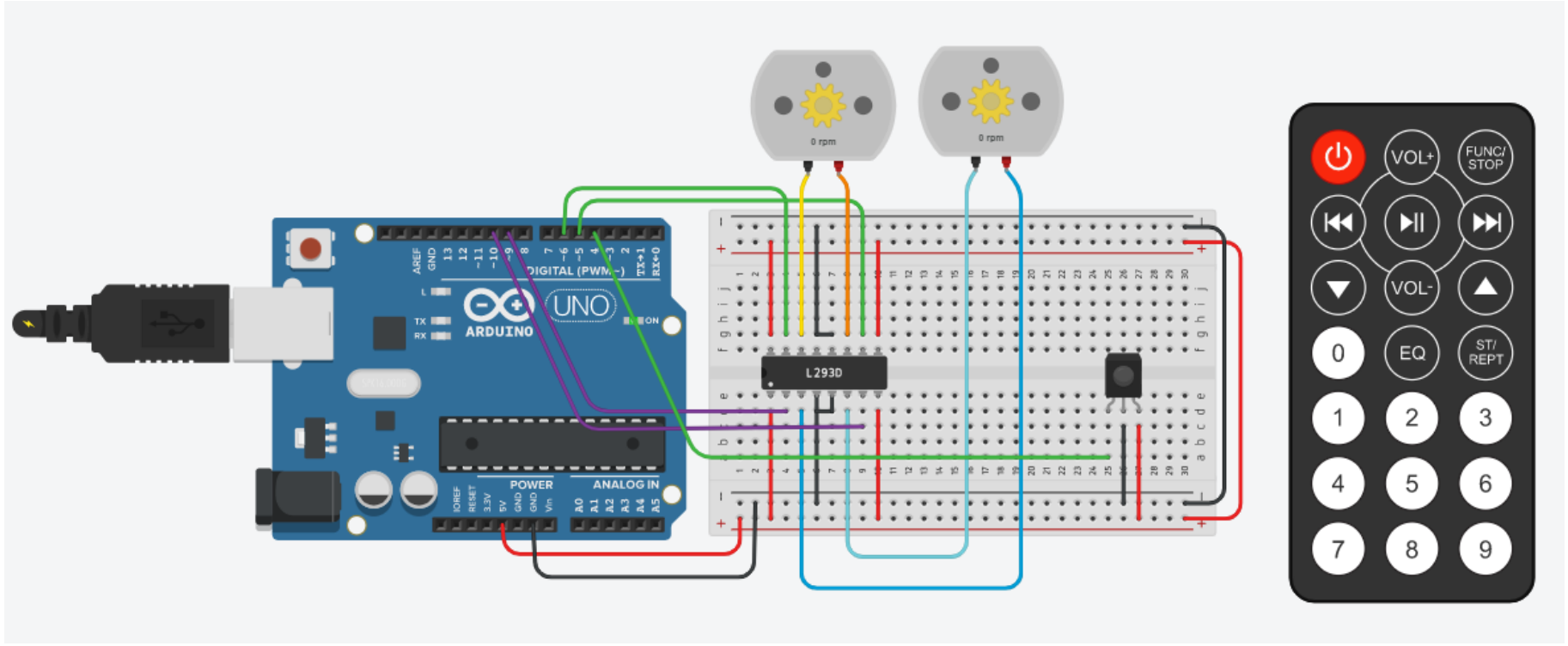
```
int encoding( long val ){  
  int ButtonNo;  
  switch (val) {  
    case 0xff6897: ButtonNo=0; break;  
    case 0xff30cf: ButtonNo=1; break;  
    case 0xff18e7: ButtonNo=2; break;  
    case 0xff7a85: ButtonNo=3; break;  
    case 0xff10ef: ButtonNo=4; break;  
    case 0xff38c7: ButtonNo=5; break;  
    case 0xff5aa5: ButtonNo=6; break;  
    case 0xff42bd: ButtonNo=7; break;  
    case 0xff4ab5: ButtonNo=8; break;  
    case 0xff52ad: ButtonNo=9; break;  
    default : ButtonNo=-1; break;  
  }  
  return ButtonNo;  
}
```



IR_Remote-5 : Serial Monitor



IR_Remote-5 Wiring



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

```
const int MOTOR_PIN_A = 5; // 왼쪽 DC모터의 빨간색 단자
const int MOTOR_PIN_B = 6; // 왼쪽 DC모터의 검은색 단자
const int MOTOR_PIN_C = 9; // 오른쪽 DC모터의 빨간색 단자
const int MOTOR_PIN_D = 10; // 오른쪽 DC모터의 검은색 단자
const int REMOTE_PIN = 4; // 적외선 리모컨 제어 단자
```

```
IRrecv irrecv(REMOTE_PIN); // 적외선 센서에 연결된 핀 번호 매핑
decode_results results; // 수신된 적외선 신호를 저장할 변수
```

```
void setup() {
```

```
    // 적외선 센서 활성화
    irrecv.enableIRIn();
```

```
    // DC모터와 연결된 디지털 핀을 출력 모드로 설정
```

```
    pinMode(MOTOR_PIN_A, OUTPUT);
    pinMode(MOTOR_PIN_B, OUTPUT);
    pinMode(MOTOR_PIN_C, OUTPUT);
    pinMode(MOTOR_PIN_D, OUTPUT);
```

```
    Serial.begin(9600);
}
```

```
void loop() {
```

```
    if(irrecv.decode(&results)) { // 적외선 신호를 해석
        Serial.println(results.value, HEX); // 적외선 신호 값을 출력
```

```
    // 적외선 신호 값에 따라 모터 제어
```

```
    if(results.value == 0xFD8877) // 2번 버튼을 눌러 전진
        moveForward();
```

```
    else if(results.value == 0xFD9867) // 8번 버튼을 눌러 후진
        moveBackward();
```

```
    else if(results.value == 0xFD28D7) // 4번 버튼을 눌러 좌회전
        turnLeft();
```

```
    else if(results.value == 0xFD6897) // 6번 버튼을 눌러 우회전
        turnRight();
```

```
    else // 기타 버튼을 눌러 멈춤
        stopMoving();
```

```
    delay(30);
    irrecv.resume(); // 다음 신호를 받기 위해 초기화
```

```
    }
}
```



```
void moveForward() {
    analogWrite(MOTOR_PIN_A, 0);    // 왼쪽 DC모터를 역방향으로 회전
    analogWrite(MOTOR_PIN_B, 255);
    analogWrite(MOTOR_PIN_C, 255);  // 오른쪽 DC모터를 정방향으로 회전
    analogWrite(MOTOR_PIN_D, 0);
}
```

```
void moveBackward() {
    analogWrite(MOTOR_PIN_A, 255);  // 왼쪽 DC모터를 정방향으로 회전
    analogWrite(MOTOR_PIN_B, 0);
    analogWrite(MOTOR_PIN_C, 0);    // 오른쪽 DC모터를 역방향으로 회전
    analogWrite(MOTOR_PIN_D, 255);
}
```

```
void turnLeft() {
    analogWrite(MOTOR_PIN_A, 255);  // 왼쪽 DC모터를 정방향으로 회전
    analogWrite(MOTOR_PIN_B, 0);
    analogWrite(MOTOR_PIN_C, 255);  // 오른쪽 DC모터를 정방향으로 회전
    analogWrite(MOTOR_PIN_D, 0);
}
```

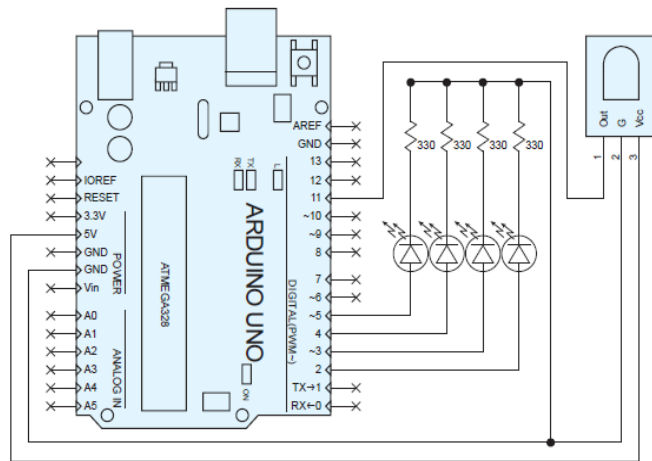
```
void turnRight() {
    analogWrite(MOTOR_PIN_A, 0);    // 왼쪽 DC모터를 역방향으로 회전
    analogWrite(MOTOR_PIN_B, 255);
    analogWrite(MOTOR_PIN_C, 0);    // 오른쪽 DC모터를 역방향으로 회전
    analogWrite(MOTOR_PIN_D, 255);
}
```

```
void stopMoving() {
    analogWrite(MOTOR_PIN_A, 0);    // 모든 DC모터의 속도를 0으로 설정
    analogWrite(MOTOR_PIN_B, 0);
    analogWrite(MOTOR_PIN_C, 0);
    analogWrite(MOTOR_PIN_D, 0);
}
```



적외선 리모컨을 이용한 LED 제어

- 실습목표
 - 1. 적외선 리모컨의 신호를 수신한다.
 - 2. 수신된 코드에 따라 LED를 제어한다.
 - 3. 메시지를 시리얼 통신으로 전송한다.
- HARDWEAR
 - 1. IR Receiver의 Vcc와 G를 Arduino의 5V와 GND에 연결한다.
 - 2. IR Receiver Out은 디지털입출력 11번 핀에 연결한다.
 - 3. Arduino의 디지털 입출력 1~4번핀에 4개의 LED의 Anode 핀을 연결한다.
 - 4. 각 LED의 Cathode에는 220 Ω 저항을 연결하여 GND에 연결한다.



Commends

- `Serial.begin(전송속도)`
 - 시리얼 통신 포트를 컴퓨터와 연결한다.
 - 전송속도는 bps(bits per sec)로 일반적으로 9600으로 설정한다. 19200, 57600, 115200 등의 값을 설정할 수 있다.
- `Serial.print(전송내용)`
 - 괄호 안의 내용을 시리얼 통신으로 전송한다. 따옴표로 구분된 부분은 텍스트를 직접 전송하고 따옴표 없이 변수를 써주면 변수의 값이 전송된다.
- `Serial.println(전송내용)`
 - 'Serial.print'와 같으나 전송뒤 줄바꿈을 한다.
- `IRrecv 리모컨이름(리모컨 수신 핀번호)`
 - 리모컨 이름으로 리모컨 수신핀을 설정한다.
- `리모컨이름.enableIRIn()`
 - 리모컨 수신시 타이머 인터럽트를 활성화한다.
- `decode_result` 실습결과
 - 입력된 신호의 결과를 '실습결과' 변수에 입력한다.
- `리모컨이름.decode(&실습결과)`
 - 리모컨 수신에 이루어졌을 때 그 값을 회신한다.
- `실습결과.decode_type`
 - 수신된 리모컨 코드의 타입이 저장된다(NEC, SONY, RC5, RC6 등).
- `실습결과.value`
 - 수신된 리모컨의 코드가 저장된다.
- `실습결과.bits`
 - 수신된 리모컨 코드의 비트수가 저장된다.
- `실습결과.rawbuf`
 - 수신된 적외선 펄스의 시간이 배열형태로 저장된다.
- `실습결과.rawlen`
 - 수신된 적외선 펄스의 시간이 배열의 개수가 저장된다.
- `리모컨이름.resume()`
 - 수신 후에 다른 신호를 수신하기 위하여 준비한다.
- `리모컨이름.blink13(true)`
 - Arduino에 내장된 LED(13번핀)을 리모컨 수신할 때마다 점멸시킨다.



Program 구성

- **Sketch** 구성

- 1. 리모컨 라이브러리를 불러온다.
- 2. 디지털입력 11번 핀을 리모컨 수신핀으로 설정한다.
- 3. 'irrecv'라는 이름으로 리모컨이름을 설정한다.
- 4. 리모컨신호 수신 결과를 'results'라는 이름으로 설정한다.
- 5. 디지털 입출력핀 2, 3, 4, 5를 LED 출력으로 사용하기위해 출력모드로 설정한다.
- 6. 각 LED의 점등과 소등 신호를 미리 설정해 준다.
- 7. 적외선 리모컨 신호가 수신되었을 때 신호에 맞춰 LED를 동작시킨다.

- **결과**

- 1. 리모컨 키를 누를 때 마다 해당 LED가 점등/소등 한다.
- 2. 현재 동작 상태에 대한 메시지가 시리얼 통신으로 전송된다.

