

# I<sup>2</sup>C 특징

## a. 다수의 장치간의 연결이 가능.

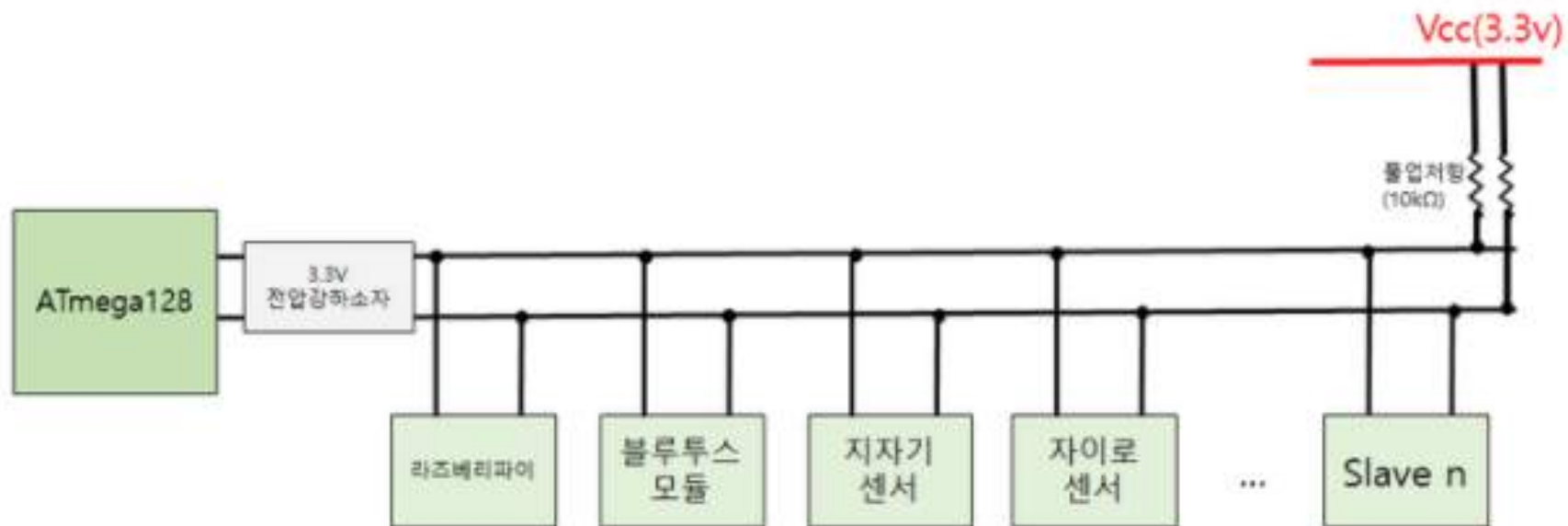
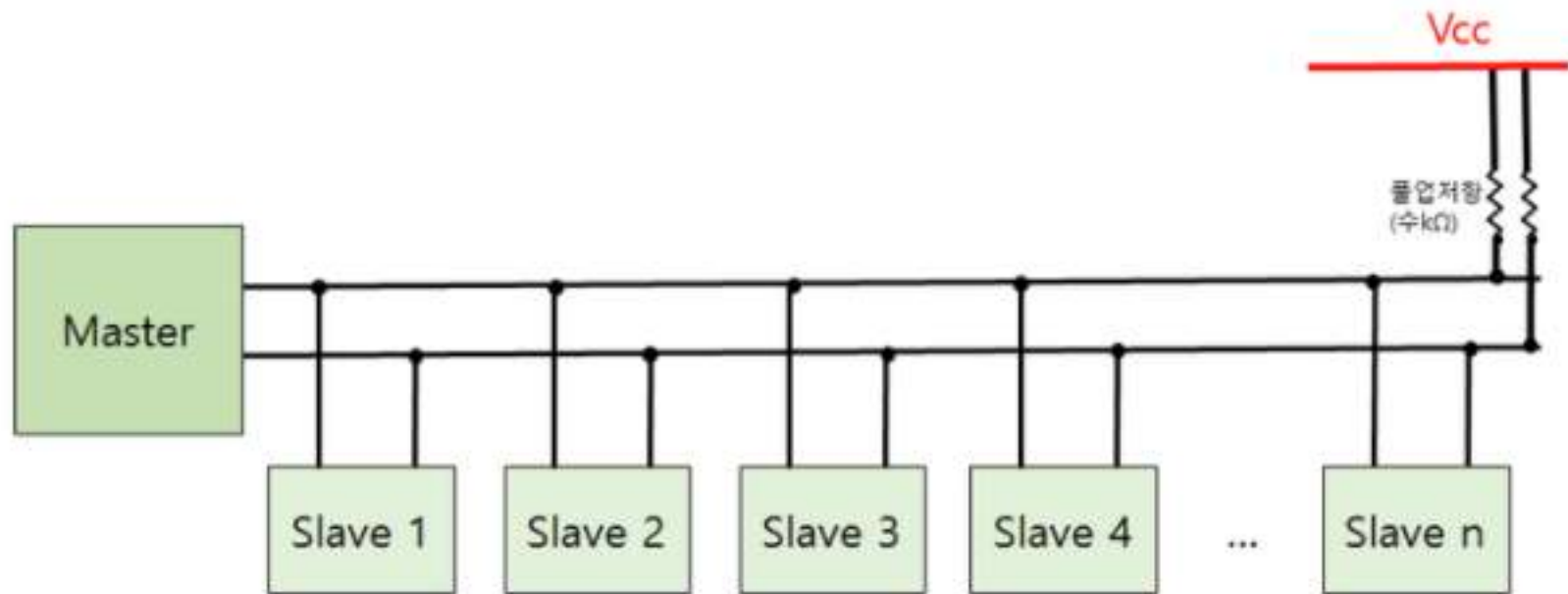
- 버스의 용량(capacitor)이 400pF을 넘지 않는 한에서 장치를 다량 연결 가능
- 다수의 마스터가 버스에 연결되어 있어도 장치간 충돌 검출 가능
- 버스의 연결된 장치들이 고유의 주소값을 가지고 master/slave 관계를 가진다.
- 각 (슬레이브)장치는 고유의 주소값(7bit)을 가진다.

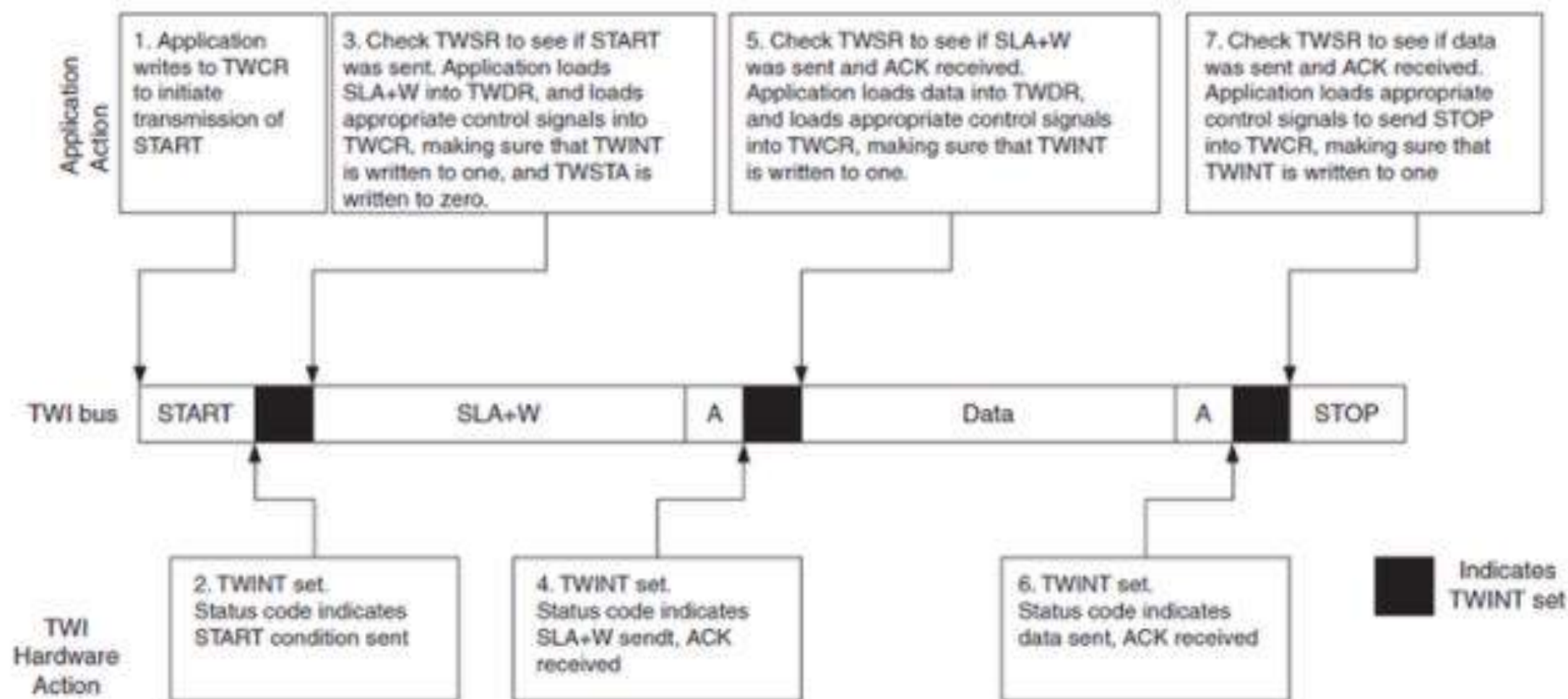
## b. 세 가지 전송 속도를 지원.

- 표준 모드(Standard Mode) : 100 Kbps
- 고속 모드(Fast Mode) : 400 Kbps
- 초고속 모드(High Speed Mode) : 3.4Mbps

## c. 통신 방법

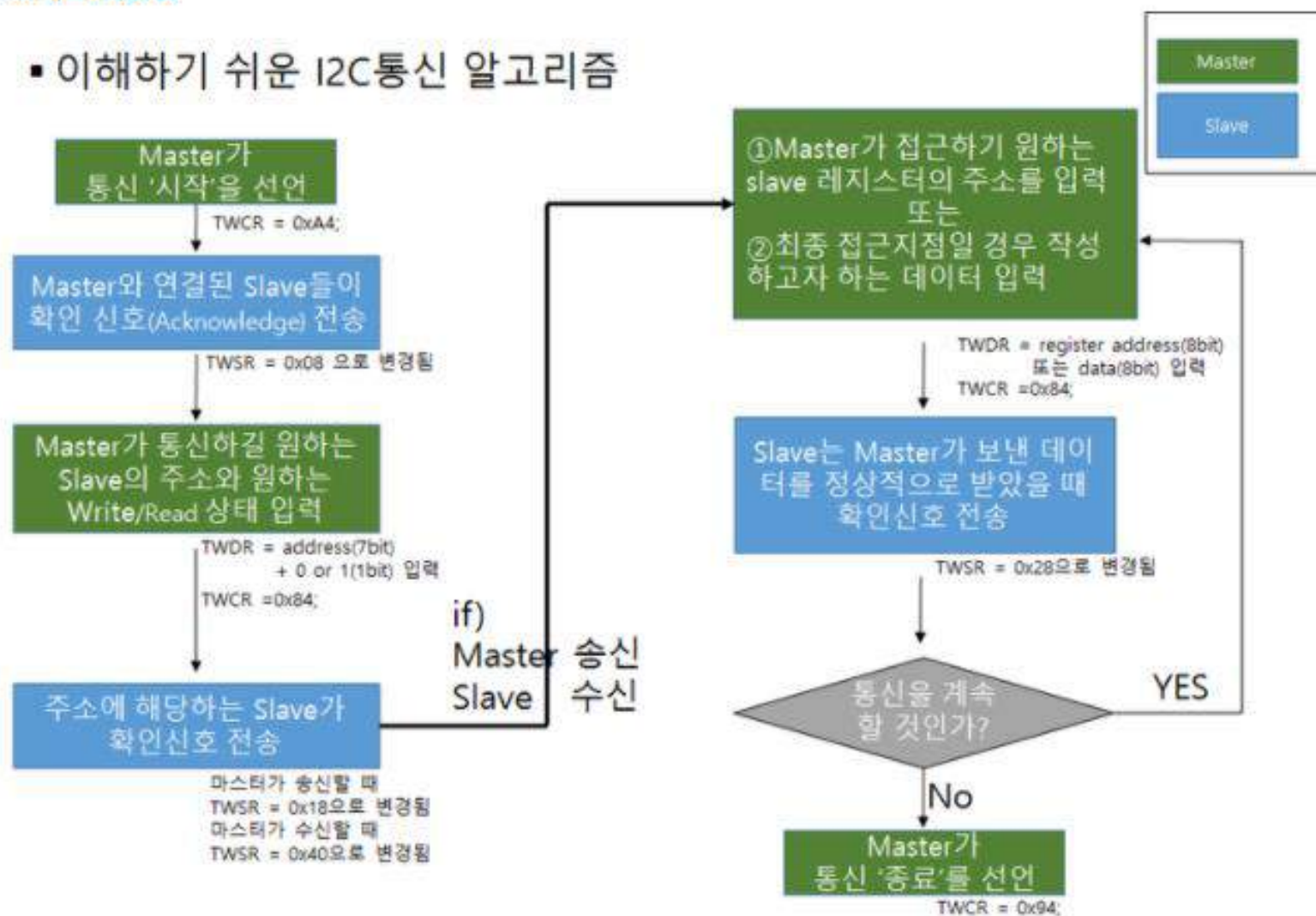
- '마스터'가 통신의 시작 및 통신의 종료를 선언하여 통신이 이루어진다.
- '마스터'가 특정 주소를 가진 슬레이브와 연결하여 송수신을 수행한다.
- '마스터'가 클럭신호 SCL을 발생하여 통신을 중계한다.





## a. 마스터송신

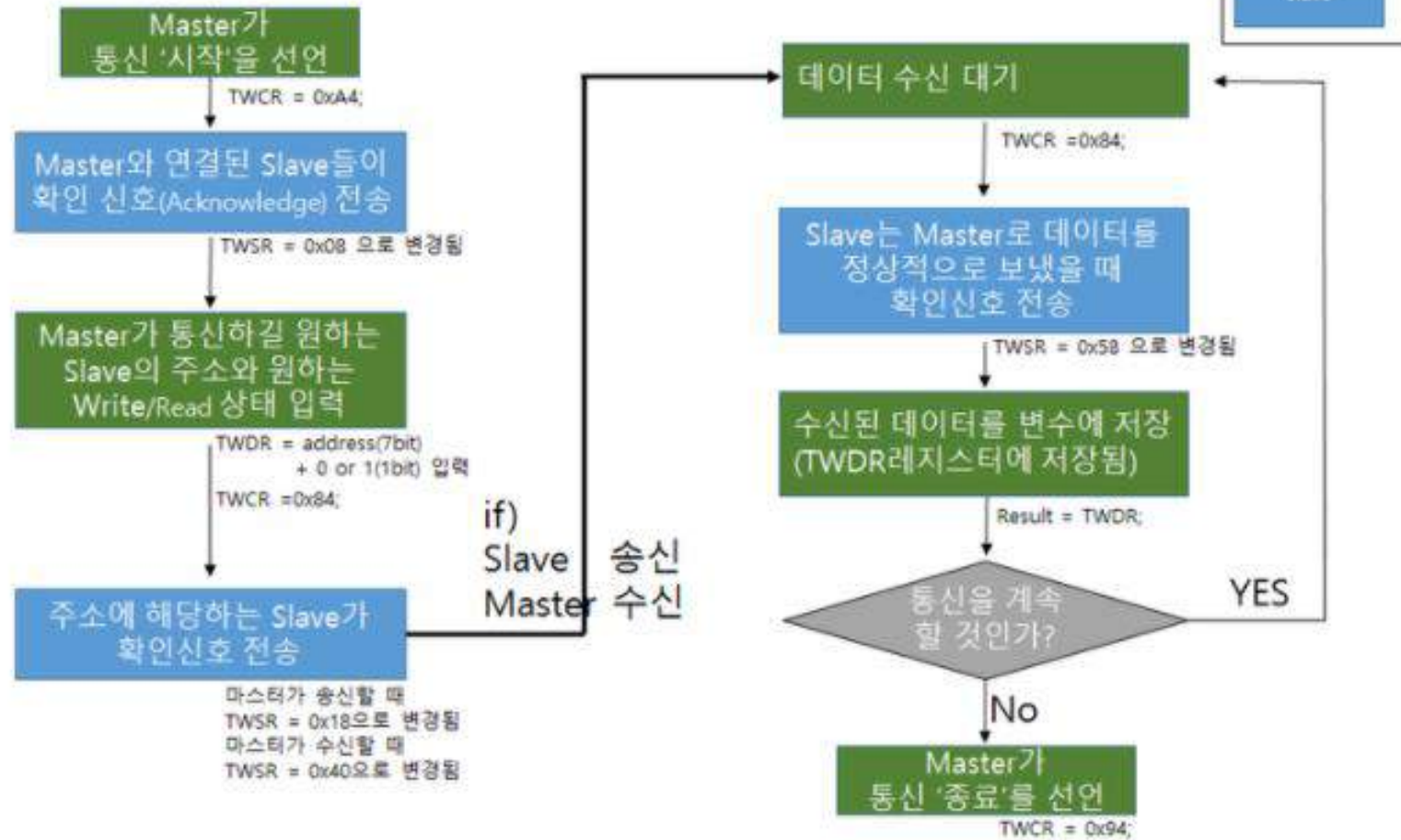
### ■ 이해하기 쉬운 I2C통신 알고리즘





## b. 마스터수신

### ■ 이해하기 쉬운 I2C통신 알고리즘



```
void I2C_init(void)
{
    TWBR = 12;      // I2C 전송속도 400KHz로 세팅 (시스템클럭이 16MHz인 경우)
    TWSR = 0x00;
}
```

```
void I2C_start(void)
{
    TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);
}
```

```
void I2C_stop(void)
{
    TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO);
}
```

```

void I2Cwrite(unsigned char address, unsigned char data)
{
    I2C_start()                // START condition 송신 (TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);)
    // START condition 신호가 전송완료까지 대기
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x08);
    TWDR = 0xd0 | 0x00;        // 부품의 I2C 주소(datasheet: DS1307은 0xd0 )
    TWCR = 0x84;               // 마스터 송신 모드
    // 전송 완료 시 슬레이브에서 ACK 대기
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x18);
    TWDR = address;            // Write address (datasheet참조해야 함)
    TWCR = 0x84;               // 마스터 송신 모드
    // 전송 완료 시 슬레이브에서 ACK 대기
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x28);
    TWDR = data;               // write data
    TWCR = 0x84;               // 마스터 송신 모드
    // 전송 완료 시 슬레이브에서 ACK 대기
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x28);
    I2C_stop();                // STOP condition (TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO);)
    delay(1);
}

```



```
unsigned char I2Cread(unsigned char address)
```

```
{
```

```
    unsigned char data = 0;
```

```
    I2C_start(); // START condition 송신
```

```
    // START condition 신호가 전송완료까지 대기
```

```
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x08);
```

```
    TWDR = DS1307|WMODE;          // DS1307 address & write mode
```

```
    TWCR = 0x84;                  // 마스터 송신 모드
```

```
    // 전송 완료시 슬레이브에서 ACK 대기
```

```
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x18);
```

```
    TWDR = address;              // Write address (읽고자 하는 주소)
```

```
    TWCR = 0x84;                  // 마스터 송신 모드
```

```
    // 전송 완료시 슬레이브에서 ACK 대기
```

```
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x28);
```

```
    TWCR = 0xA4;                  // RESTART
```

```
}
```

```
    // 상태코드가 0x10 확인
```

```
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x10);
```

```
    TWDR = DS1307|RMODE;          // DS1307 address & read mode
```

```
    TWCR = 0x84;                  // 마스터 송신 모드
```

```
    // 전송 완료시 슬레이브에서 ACK 대기
```

```
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x40);
```

```
    TWCR = 0x84;                  // 마스터 수신대기 모드
```

```
    // data 수신 대기
```

```
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x58);
```

```
    data = TWDR;                  // data 수신
```

```
    TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO); // STOP condition
```

```
    return data;
```

예제: GY-86

(출처: <https://blog.komastar.kr/13>)

- I2C(TWI)통신으로 GY-86모듈 내에 있는 MPU6050의 가속도 XYZ 출력(0x3B - 0x40)과 자이로 XYZ 출력(0x43 - 0x48) 레지스터를 읽기

## Description

10DOF modules (3축 자이로 + 3축 가속도 + 3축 지자계 + 대기압력)

Sensor : MPU6050 + HMC5883L + MS5611

gyroscope range: + 250 500 1000 2000 °/ s

Acceleration range:  $\pm 2 \pm 4 \pm 8 \pm 16$  g

HMC5883L: Measuring range:  $\pm 1.3-8$  Gauss.

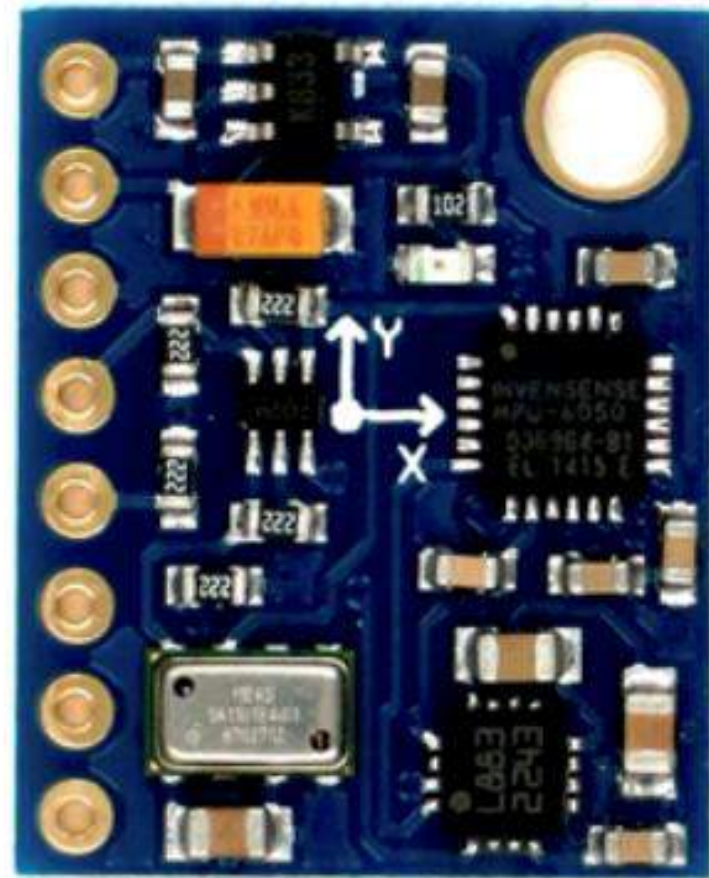
Power supply :3-5v

(fully compatible with 3v-5V systems, including LLC circuit)

Interface : I2C

Weight: 1.46g

Size: 22 x 17mm



DATASHEET 다운로드 !!!  
(MPU6050 + HMC5883L +  
MS5611)

MPU6050 datasheet의 register map (7페이지)를 보면 아래와 같음 (원하는 데이터의 주소를 확인할 것)

[illegible]

```

#include <stdlib.h>
#include <delay.h>
#include <mega128.h>
#include <math.h>
#include <stdio.h>
#include <string.h>
typedef unsigned char byte;

byte MPU6050_read(byte addr);
void MPU6050_write(byte addr, char data);
void getRawData();

unsigned int gx = 0, gy = 0, gz = 0, ax = 0, ay = 0, az = 0;
byte buffer[12];

void main(void)
{
    TWBR = 0x12;
    TWCR = (1 << TWEN);    //init i2c

    MPU6050_write(0x6B, 0x00);
    MPU6050_write(0x6C, 0x00);

    getRawData();

    while(1)
    {
        getRawData();
    }
}

```

```

byte MPU6050_read(byte addr)
{
    byte dat;

    TWCR = 0xA4;
    while(((TWCR & 0x80) == 0x00 || ((TWSR & 0xF8) != 0x08)));

    TWDR = 0xD0;
    TWCR = 0x84;

    while(((TWCR & 0x80) == 0x00 || ((TWSR & 0xF8) != 0x18)));
    TWDR = addr;
    TWCR = 0x84;

    while(((TWCR & 0x80) == 0x00 || ((TWSR & 0xF8) != 0x28)));

    TWCR = 0xA4;

    while(((TWCR & 0x80) == 0x00 || ((TWSR & 0xF8) != 0x10)));

    TWDR = 0xD1;
    TWCR = 0x84;

    while(((TWCR & 0x80) == 0x00 || ((TWSR & 0xF8) != 0x40)));
    TWCR = 0x84;
    while(((TWCR & 0x80) == 0x00 || ((TWSR & 0xF8) != 0x58)));
    dat = TWDR;
    TWCR = 0x94;
    return dat;
}

```

```
void MPU6050_write(byte addr, char data)
```

```
{  
  
    TWCR = 0xA4;  
    while(((TWCR & 0x80) == 0x00 || ((TWSR & 0xF8) != 0x08)));  
  
    TWDR = 0xD0;  
    TWCR = 0x84;  
  
    while(((TWCR & 0x80) == 0x00 || ((TWSR & 0xF8) != 0x18)));  
    TWDR = addr; // addr = 0x43  
    TWCR = 0x84;  
  
    while(((TWCR & 0x80) == 0x00 || ((TWSR & 0xF8) != 0x28)));  
    TWDR = data;  
    TWCR = 0x84;  
  
  
    while(((TWCR & 0x80) == 0x00 || ((TWSR & 0xF8) != 0x28)));  
    TWCR = 0x94;  
  
  
    delay_us(50);  
  
}
```

```
void getRawData()
```

```
{  
  
    buffer[0] = MPU6050_read(0x3B);  
    buffer[1] = MPU6050_read(0x3C);  
    buffer[2] = MPU6050_read(0x3D);  
    buffer[3] = MPU6050_read(0x3E);  
    buffer[4] = MPU6050_read(0x3F);  
    buffer[5] = MPU6050_read(0x40);  
    buffer[6] = MPU6050_read(0x43);  
    buffer[7] = MPU6050_read(0x44);  
    buffer[8] = MPU6050_read(0x45);  
    buffer[9] = MPU6050_read(0x46);  
    buffer[10] = MPU6050_read(0x47);  
    buffer[11] = MPU6050_read(0x48);  
  
    ax = (int)buffer[0] << 8 | (int)buffer[1];  
    ay = (int)buffer[2] << 8 | (int)buffer[3];  
    az = (int)buffer[4] << 8 | (int)buffer[5];  
  
    gx = (int)buffer[6] << 8 | (int)buffer[7];  
    gy = (int)buffer[8] << 8 | (int)buffer[9];  
    gz = (int)buffer[10] << 8 | (int)buffer[11];  
  
}
```