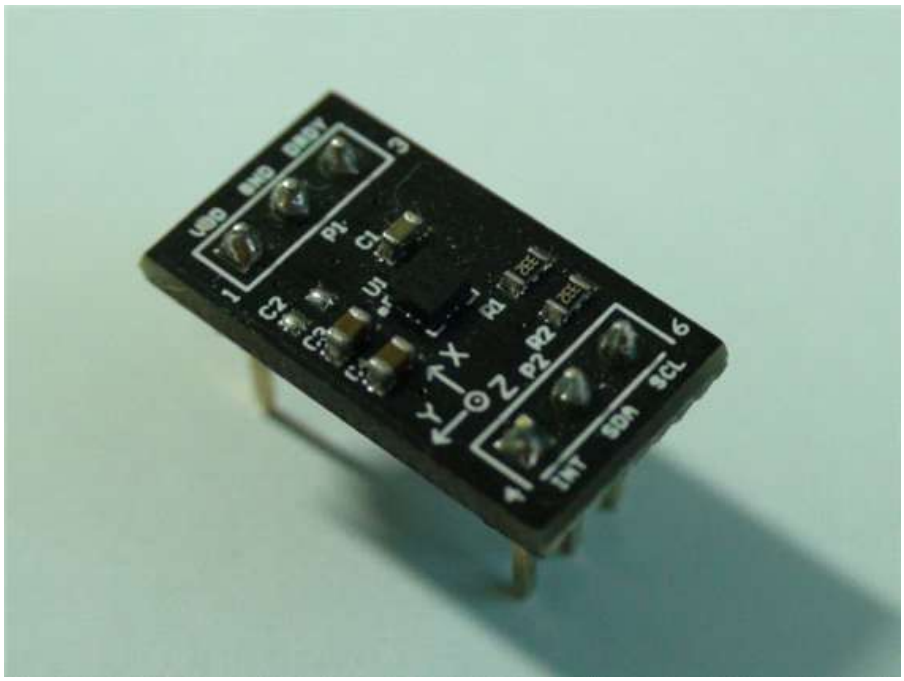
P0-MAI-00-01
Example code
(Digital 3-Axis Magnetometer Module)



1. 코드 내용

- 본 코드는 PIC 기반으로 작성된 코드.
- 본 코드가 작성된 PIC에는 I2C 모듈이 내장 되어 있음.
- I2C 모듈 셋팅 과정은 생략. (통신 속도 400KHz)
- P0-MAI-00-01 센서의 INIT-SETTING, READ, WRITE 코드.
- P0-MAI-00-01 센서의 센서 값 계산도 포함된 코드

2. READ

```
void Mag_Read(unsigned char byte, unsigned char addr)
{
    I2C1CONbits.SEN=1;                  //  S ( Start Condition )
    I2C1_Done();                        // 응답 대기

    I2C1TRN= 0x1C;                      // SAD+W ( slave address + write )
    I2C1_Done();                        // 응답 대기

    if(I2C1STATbits.ACKSTAT == 1)       // A ( ACK – SDA_Low) 응답 확인
    return 0;                           // Error 처리

    I2C1TRN= addr;                      // RAD ( Read-in address ) 읽을 레지스터 주소
    I2C1_Done();

    if(I2C1STATbits.ACKSTAT == 1)       // A ( ACK – SDA_Low) 응답 확인
    return 0;                           // Error 처리

    I2C1CONbits.RSEN=1;                 // Sr ( Restart Condition )
    I2C1_Done();                        // 응답 대기

    I2C1TRN= 0x1D;                      // SAD+R ( slave address + read )
    I2C1_Done();                        // 응답 대기

    if(I2C1STATbits.ACKSTAT == 1)       // A ( ACK – SDA_Low) 응답 확인
    return 0;                           // Error 처리

    for(rxcnt=0; rxcnt < byte; rxcnt++)
    {
        I2C1CONbits.RCEN=1;             // I2C 수신 동작
        I2C1_Done();                    // 응답 대기

        buf[rxcnt] = I2C1RCV;           // 데이터 수신
        I2C1CONbits.ACKDT=0;            // A ( ACK – SDA_Low )
        I2C1CONbits.ACKEN=1;            // ACK 전송
        I2C1_Done();                    // 응답 대기
    }

    I2C1CONbits.ACKDT=1;                // N ( NACK – SDA_Hgih )
    I2C1CONbits.ACKEN=1;                // NACK 전송
    I2C1_Done();                        // 응답 대기

    I2C1CONbits.PEN=1;                  // P ( Stop Condition )
    I2C1_Done();                        // 응답 대기
    return 1;                           // 정상 처리
}
```

3. WRITE

```
void Mag_Write(unsigned char byte, unsigned char addr, unsigned data1, unsigned data2)
{
    I2C1CONbits.SEN=1;                          //  S ( Start Condition )
    I2C1_Done();                                // 응답 대기

    I2C1TRN= 0x1C;                              // SAD+W ( slave address + write )
    I2C1_Done();                                // 응답 대기

    if(I2C1STATbits.ACKSTAT == 1)               // A ( ACK – SDA_Low) 응답 확인
    return 0;                                   // Error 처리

    I2C1TRN= addr;                              // WAD ( Write-in address ) 쓰기 할 레지스터 주소
    I2C1_Done();

    if(I2C1STATbits.ACKSTAT == 1)               // A ( ACK – SDA_Low) 응답 확인
    return 0;                                   // Error 처리

    switch(byte)
    {
        case 1:
            I2C1IRN=data1;                      // WDA1 ( Write-in data 1 )
            I2C1_Done();                        // 응답 대기
            if(I2C1STATbits.ACKSTAT == 1)       // A ( ACK – SDA_Low) 응답 확인
            return 0;                           // Error 처리
            break;

        case 2:
            I2C1IRN=data1;                      // WDA1 ( Write-in data 1 )
            I2C1_Done();                        // 응답 대기
            if(I2C1STATbits.ACKSTAT == 1)       // A ( ACK – SDA_Low) 응답 확인
            return 0;                           // Error 처리
            I2C1IRN=data2;                      // WDA2 ( Write-in data 2 )
            I2C1_Done();                        // 응답 대기
            if(I2C1STATbits.ACKSTAT == 1)       // A ( ACK – SDA_Low) 응답 확인
            return 0;                           // Error 처리
            break;

        default:
            break;
    }
    I2C1CONbits.PEN=1;                          // P ( Stop Condition )
    I2C1_Done();                                // 응답 대기
    return 1;                                   // 정상 처리
}
```

4. INIT-SETTING

```
/// CNTL1 ///
Mag_Write(0x01, 0x1B, 0x82, 0x00);
/// CNTL2 ///
Mag_Write(0x01, 0x1C, 0x80, 0x00);
/// CNTL4 ///
Mag_Write(0x02, 0x5C, 0x7E, 0xA0);
```

5. Magnetic data & Temperature

```
while(1)
{
    Mag_Write(0x01, 0x1D, 0x40, 0x00);
    Delay_us(500);                          // 딜레이 500 uS

    Mag_Read(0x06, 0x10);                   // 지자기 3축에 대한 데이터 수신
    mag_x=buf[1];                           // 지자기 X축 상위 바이트
    mag_x=mag_x<<8 | buf[0];                // 지자기 X축
    mag_y=buf[3];                           // 지자기 Y축 상위 바이트
    mag_y=mag_y<<8 | buf[2];                // 지자기 Y축
    mag_z=buf[5];                           // 지자기 Z축 상위 바이트
    mag_z=mag_z<<8 | buf[4];                // 지자기 Z축

    Mag_Read(0x06, 0x96);                   // 지자기 3축에 대한 감도 수신
    sens_x=buf[1];                          // 지자기 X축 상위 바이트
    sens_x=sens_x<<8 | buf[0];              // 지자기 X축
    sens_y=buf[3];                          // 지자기 Y축 상위 바이트
    sens_y=sens_y<<8 | buf[2];              // 지자기 Y축
    sens_z=buf[5];                          // 지자기 Z축 상위 바이트
    sens_z=sens_z<<8 | buf[4];              // 지자기 Z축

    MAG_X=(double)mag_x/(double)sens_x;     // X축 절대 자장
    MAG_Y=(double)mag_y/(double)sens_y;     // X축 절대 자장
    MAG_Z=(double)mag_z/(double)sens_z;     // X축 절대 자장

    Mag_Read(0x02, 0x60);                   // 센서 모듈 온도 수신
    temp=buf[1];                            // 온도 상위 바이트
    temp=temp<<8 | buf[0];                  // 센서 모듈 온도
    MAG_TEMP=(double)temp-1900;             // 센서 온도 계산
    MAG_TEMP= 25-(MAG_TEMP*196/1000);       // 센서 별 온도의 차이가 있음
}
```

6. Notes

- 지자기센서 chip 매뉴얼 참조
(http://www.aichi-mi.com/3_products/110127ami306e.pdf)

7. Contact

- 판매사 : mySen(마이센)
- 주소 : 경기도 고양시 덕양구 항공대학로 76, 301호
(화전동중소벤처육성지원센터)
- 전화 : 070-7010-6989
- Fax. : 070-7614-3989
- Homepage : www.mysen.kr