
〈 DTPM11 〉


〈 DTPM81 〉


〈 DTPM151 〉

- 비접촉 온도 측정
- 방사율 조절 가능
- IR refresh rate : 2Hz
- High Accuracy
- Digital Resolution : 0.1℃
- Digital Interface : SPI
- 아두이노, 라즈베리파이 코드 제공

▶ 특징

- 대상 온도와 센서 온도를 동시 측정.
- 측정 온도 구간 : 특성표 참고(2 page)
- 동작 온도 구간 : -20 ℃ ~ 70℃
- 파장대역 : 특성표 참고(2 page)
- FOV : 특성표 참고(2 page)
- 정확도 : ±2%
- 입력 전압 : 3.3V
- 통신 인터페이스 : SPI
- 방사율 변경 가능 : 0.10 ~ 1(default:0.97)

▶ 응용분야

- 과열방지 시스템
- 산업용 온도 측정 장치
- 체온 측정을 통한 인체 감지
- 가전기기
- 지능형 온도 제어 시스템

▶ Ordering Guide

DTPM△△△

DS ratio	
11	1:1
81	8:1
151	15:1

Absolute Maximum Ratings

- Supply voltage : 3.5V
- Operating Temperature Range : -20℃ ~ 70℃
- Storage Temperature Range : -40℃ ~ 85℃

위 조건을 넘어서게 되면 제품의 수명을 보장할 수 없습니다.
반드시 아래 Electrical Requirements 를 지켜주세요.

Electrical Requirements

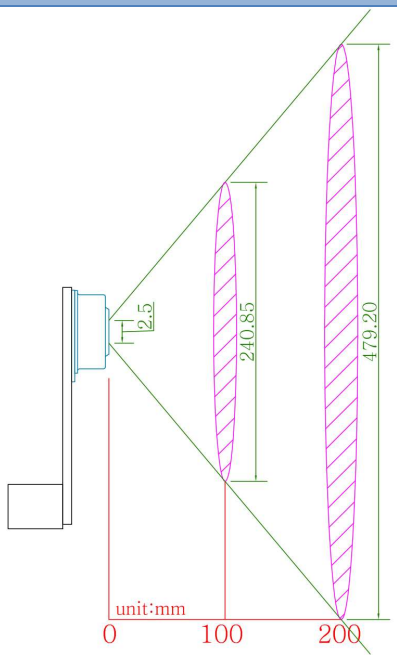
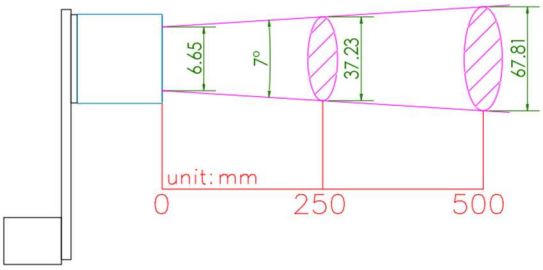
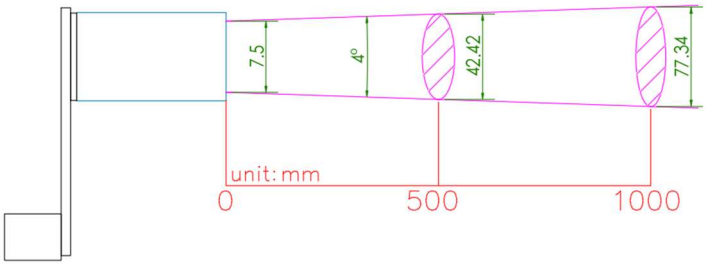
Parameter		Symbol	Conditions	min	Typ	Max	Unit
공급전압		Vcc	Measured versus GND	3.1	3.3	3.5	V
방사율(Emission Coefficient)		ε		0.1	0.97	1	
공급 전류			Full ambient temp. range, Typical value, no output load		10.96	12	mA
SPI	Clock					1	MHz
	INPUT High Level			3.1		3.5	V
	INPUT Low Level					0.9	V
	OUTPUT High Level			Vcc-0.3		Vcc	V
	OUTPUT LOW Level			Vss		Vss+0.3	V

Operational Characteristics

- if not otherwise noted, 25℃ ambient temperature, 3.3V supply voltage and object with ε =0.97 were applied

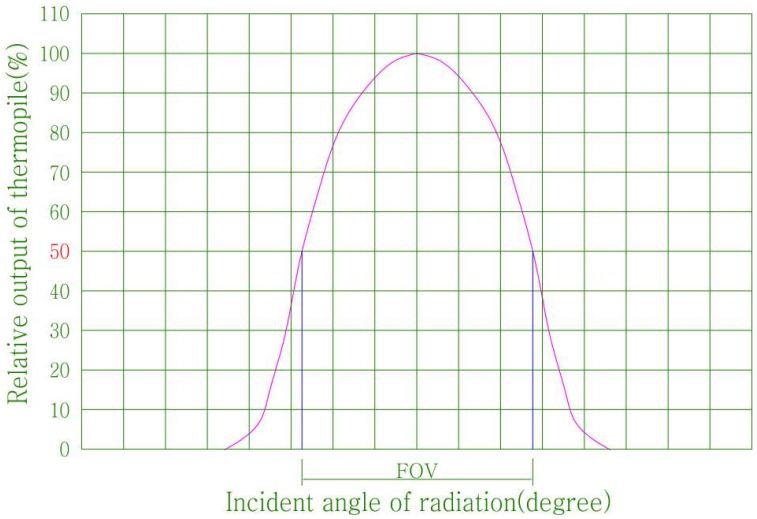
Parameter		min	Typ	Max	Unit
측정각도(FOV)	DTPM11 DTPM81 DTPM151		100 7 4		°
온도측정범위	DTPM11 DTPM81 DTPM151	-20 -20 -20		100 200 270	℃
측정 파장 대역	전모델	5.5		14	㎛
동작온도(주변온도)		-20		70	℃
온도측정 시간			0.5		sec
정확도			±2		%
Digital Resolution			0.1		℃
Standard Start-UP Time			3		sec
Stabilization Time			1		min

Optical field of view (FOV)

모델명	측정각도	FOV
DTPM11	100°	 ※ 측정직경 계산식 : 2 x tan(100 °/ 2) x 거리(mm) + 2.5 (mm)
DTPM81	7°	 ※ 측정직경 계산식 : 2 x tan(7 °/ 2) x 거리(mm) + 6.65 (mm)
DTPM151	4°	 ※ 측정직경 계산식 : 2 x tan(4 °/ 2) x 거리(mm) + 7.5 (mm)

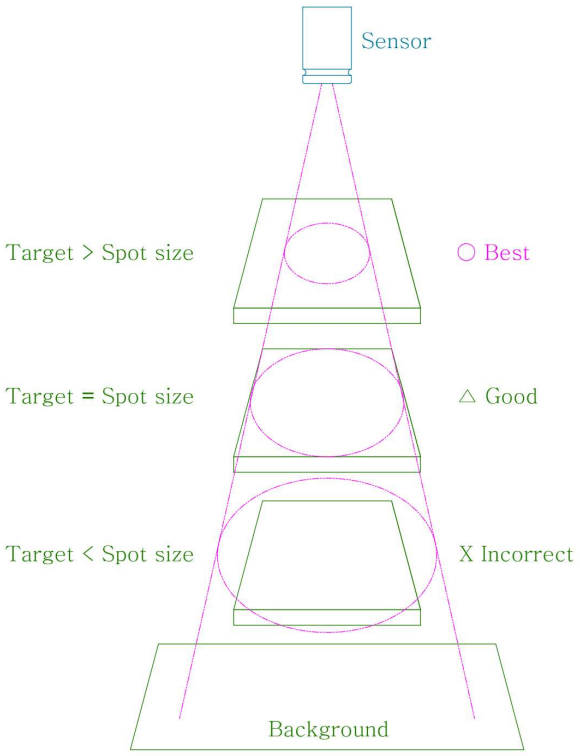
※ 측정하고자 하는 물체의 크기는 위 계산식의 spot size보다 충분히 더 커야 측정이 용이합니다.
다음 페이지 Distance and spot size 그림을 참고하십시오.

The optical chart below indicates the nominal target spot diameter at any given distance from the sensing head and assumes 50% energy.



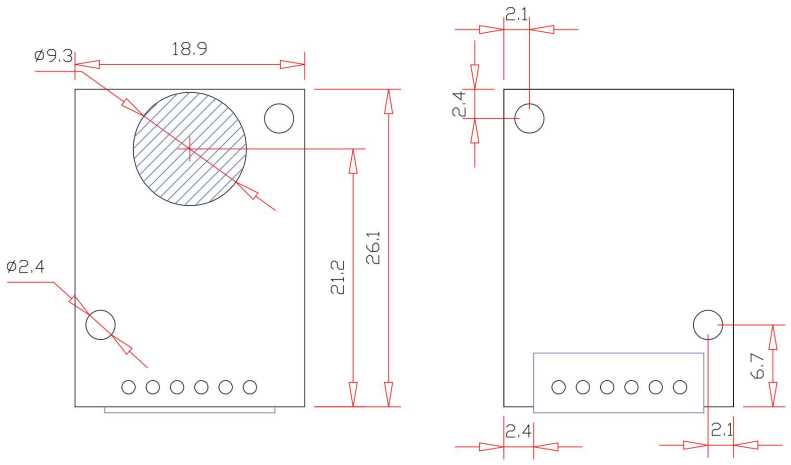
DISTANCE AND SPOT SIZE

Spot Size는 아래 그림에서와 같이 측정하고자 하는 대상의 면적보다 반드시 작아야 합니다.

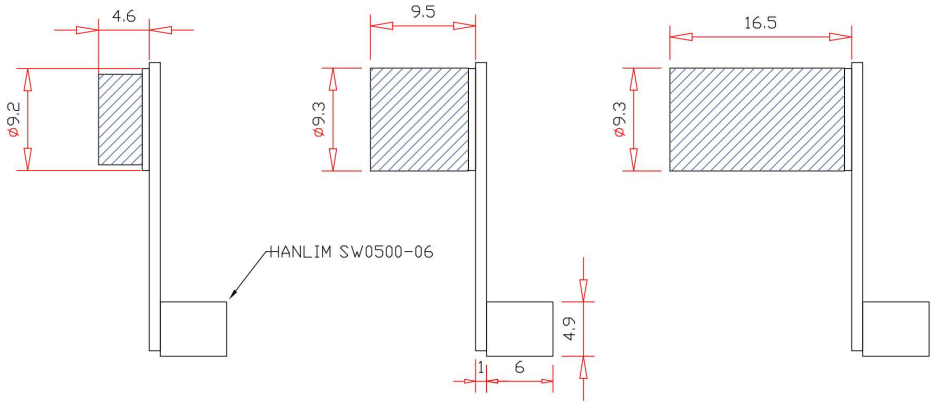


Mechanical Dimensions

단위 : mm



< TOP View > < BOTTOM View >



< DTPM11 > < DTPM81 > < DTPM151 >

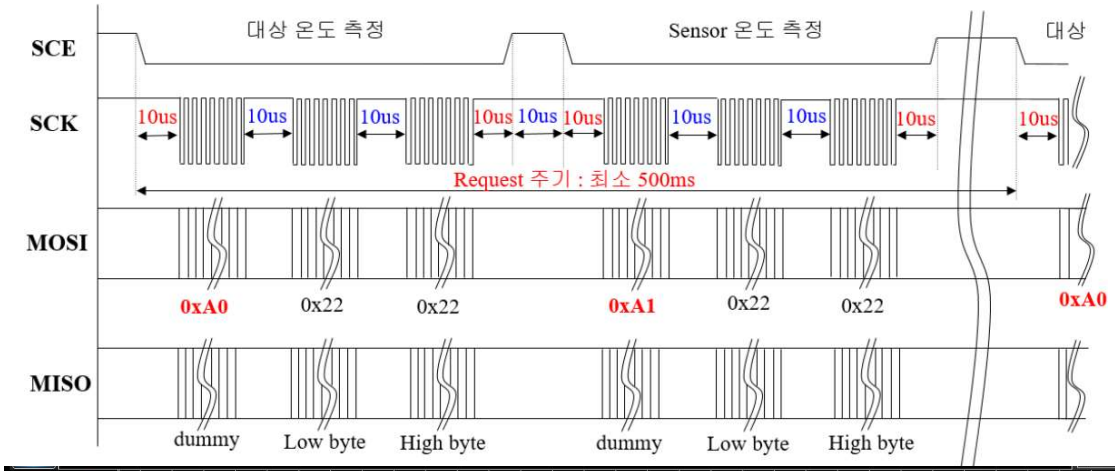
Pin Assignment

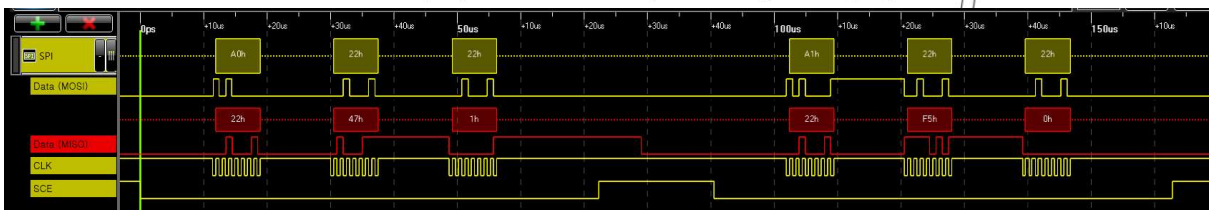
number	Name	Description	Type
1	GND	Ground	Ground
2	SCE	ENABLE	Input
3	SCK	CLOCK	Input
4	SDI	MOSI	Input
5	SDO	MISO	Output
6	VCC	3.3V	Supply

- **Connector : HANLIM SW0500-06(반대편 Connector : HANLIM CH0500-06)**

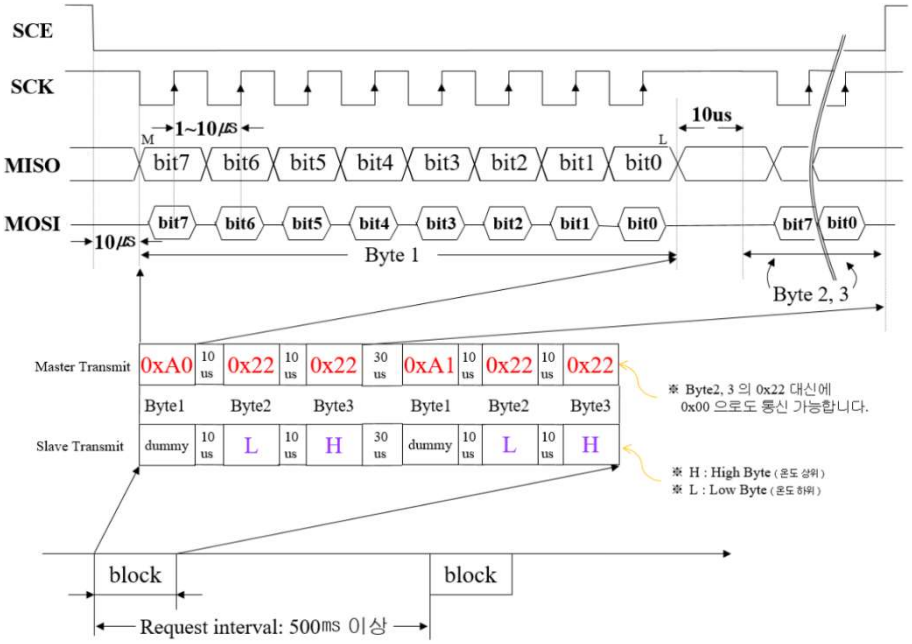
SPI Communication and Timings

- SPI Data Mode : Mode3 (SCK rising edge data sampling, SCK idle High)
- SPI Clock : Max 1MHz
- SPI bit order : MSB First
- Object(대상) & Sensor(센서) 온도 읽기 통신 파형/ 오실로스코프 캡쳐 화면

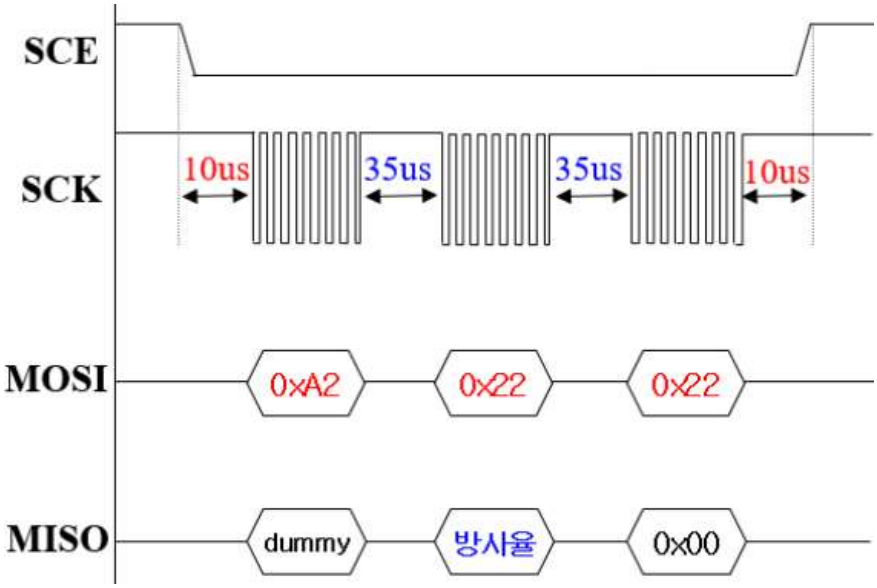




- 1byte 확대 그림

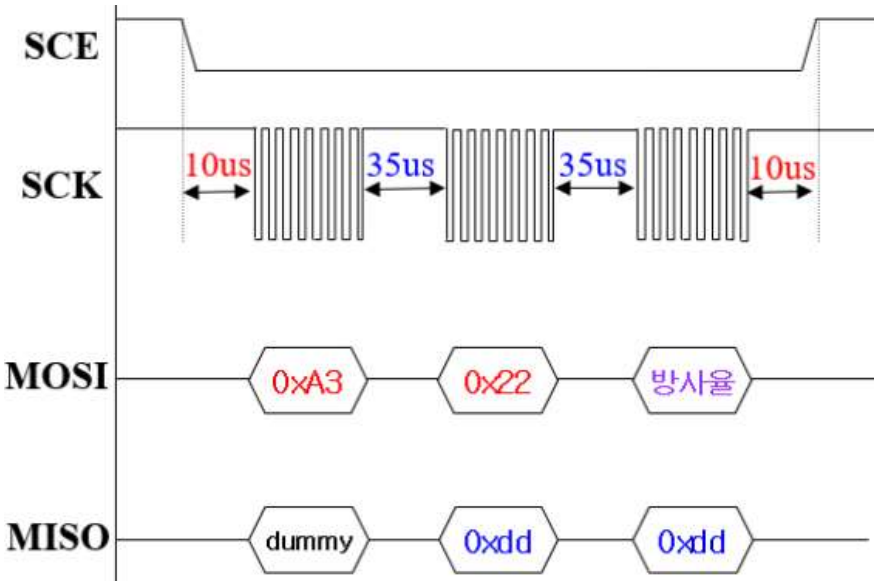


- **방사율 READ**



예) 읽어온 데이터가 97 일 경우 : 0.97 의 방사율을 의미합니다.

- **방사율 WRITE**



※ 방사율(default 0.97) 변경은 0.1~ 1.0 까지만 가능합니다.
방사율 수치에 100을 곱한 값을 전송하면 됩니다.
예) 0.97 로 변경하고자 하는 경우 : 0.97 * 100 = 97

※ 방사율은 한번 write 할 경우 전원이 리셋 돼도 유지가 됩니다.
※ 메인문의 반복문이나 전원 켤 때마다 명령을 수행할 필요가 전혀 없습니다.
※ 반드시 SCK의 byte 통신간 최소 35us 의 지연시간이 필요합니다.

▶ 온도 계산 방법

※ 온도 데이터에서 10을 나눠주면 됩니다.

- **영상온도 계산**

object	0x6D	0x01

sensor	0xFA	0x00

*** object 온도 계산 :** 상위 Byte(0x01) + 하위 Byte(0x6D) = 0x016D (Hex)
=> 365(dec) 이며 이 값을 10으로 나누면 36.5 도입니다.
*** sensor 온도 계산 :** 상위 Byte(0x00) + 하위 Byte(0xFA) = 0x00FA
=>250 (dec) 즉, 25.0 도입니다.

- **영하온도 계산(영하(0도 미만)일 때는 2의 보수 값으로 전송됩니다.)**

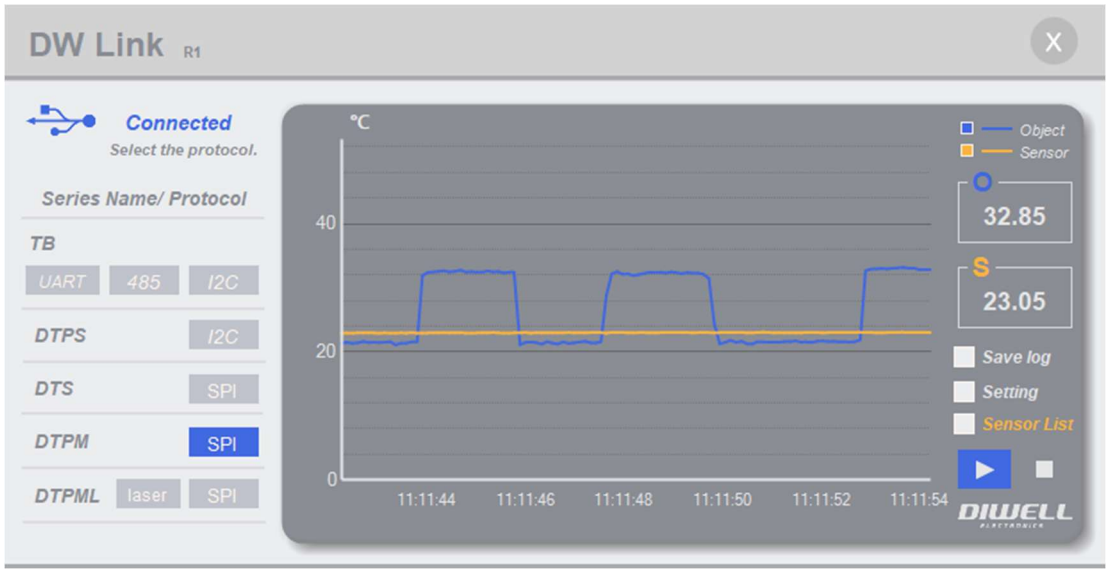
object	0xF1	0xFF

sensor	0x9C	0xFF

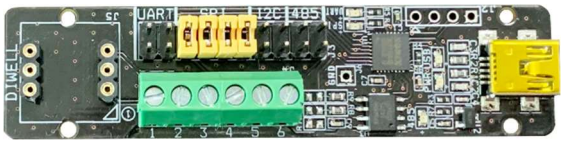
*** objcect 온도 계산 :** 상위 Byte(0xFF) + 하위 Byte(0xF1) = 0xFFF1 (Hex)
0xFFF1 의 2의 보수 = 0x000F = 15 즉, -1.5 도를 의미합니다.
*** sensor 온도 계산 :** 상위 Byte(0xFF) + 하위 Byte(0x9C) = 0xFF9C => 즉, -10.0도입니다.

▶ 윈도우 실행 프로그램 제공

온도 센서는 사용환경에서 측정해 보는 것이 가장 중요합니다.
하지만 응용분야에 적용 가능 여부만을 판단하기 위해 개발에 소요되는 시간/ 비용은 무시 못합니다.
이런 경우 DW-LINK 통신 보드를 구매하시면 간단히 PC 와 연결하여 온도 측정 및 기록이 가능합니다.
DW-LINK 관련 자세한 사항은 별도의 문서로 제공됩니다. Windows 10 전용입니다.
쇼핑몰 상세 페이지에서 다운받으세요.



< PC 화면 >



< DW-LINK >

▶ Appendix - A (방사율표)

방사율이란 물체가 외부 적외선 에너지를 흡수, 투과 및 반사하는 비율을 말하는데, 이론적으로 외부에너지를 흡수만 하고 반사하지 않는 물체를 흑체라 하여 이때의 방사율은 "1"입니다. 하지만, 일반적으로 물체의 표면상태(광택, 거침, 산화여부 등) 에 따라 흡수, 반사하는 에너지량이 변합니다. 재질에 따른 방사율 값은 하단의 "방사율표" 를 참고하여 변경하면 됩니다. 단, "방사율표"의 값은 절대적인 값이 아니며 표면 상태와 그 외 복합적인 환경 요인에 따라 오차가 있을 수 있으니, 이점 참고하십시오.

대상	*방사율*	대상	*방사율*	대상	*방사율*
산화아연	**0.1**	에나멜	**0.9**	구리(연마된)	**0.5**
아연도금철	**0.3**	페인트	**0.95**	구리(산화된)	**0.8**
주석도금철	**0.1**	라 카	**0.9**	니켈(순수)	**0.1**
금(연마된)	**0.1**	고무(smooth)	**0.9**	니켈(산화된)	**0.4~0.5**
은(연마된)	**0.1**	고무(Rough)	**0.98**	니켈크롬	**0.7**
크롬(연마된)	**0.1**	플라스틱	**0.8~0.95**	니켈크롬(산화된)	**0.95**
붉은 벽돌	**0.75~0.9**	플라스틱필름	**0.5~0.95**	직물	**0.9**
흙	**0.92~0.96**	주철(연마)	**0.2**	피부	**0.98**
석면	**0.95**	Steel	**0.6**	가죽	**0.75~0.8**
콘크리트	**0.7**	산화 Steel	**0.9**	얼음	**0.96~0.98**
대리석	**0.9**	목재	**0.8~0.9**	모래	**0.9**
모르타르	**0.89~0.91**	스테인레스(연마된)	**0.1**	아스팔트	**0.9~0.98**
석고	**0.85**	스테인레스(기타)	**0.2~0.6**	유리	**0.8~0.9**
시멘트	**0.96**	알루미늄(연마된)	**0.1**	물	**0.8~0.9**
규토(정제된)	**0.4**	알루미늄(합금)	**0.1~0.25**	종이	**0.9**
세라믹	**0.90~0.94**	황동(연마된)	**0.1**	실리콘	**0.7**
석 영	**0.9**	황동(거친)	**0.2**	주철(부식된)	**0.95**
석 탄	**0.75**	황동(산화된)	**0.6**	Mild Steel	**0.3~0.5**
Fe(부식된)	**0.7~0.85**				

※ 측정하고자 하는 대상의 재질이 *코팅*이 돼 있거나 *반짝이는 재질(동판, 알루미늄 등)*이라면 방사율을 수정 적용해도 온도 변화가 적을 수 있습니다. 이 때에는 방사율 조정을 하지 마시고, 측정 물체 표면에 "방사율 테이프" 또는 "흑색 무광의 스프레이" 를 칠하면 됩니다. 단, 측정 표면의 온도에 적합한 내열성을 가지는지 확인하십시오.

▶ Appendix - B (Example Code - Arduino UNO)

하단 코드는 아두이노 UNO 코드 예제 입니다. 예제 코드는 쇼핑몰에서 다운로드 가능합니다.

```
/************************************************************************
*
* Project Name : (DTPM 시리즈) SPI Master Code
* Version : 1.1 (2020.01.02)
* SYSTEM CLOCK : 16Mhz
* BOARD : Arduino UNO. 5V operation

 PORT Description
1. ChipSelectPin : 10
2. MOSI(Master Output) : 11
3. MISO(Master Input) : 12
4. SCK : 13
   온도센서모듈 입력전원은 3.3V 로 하셔야 하며 포트 연결 방법은 회로도를 참고하십시오.
   온도센서 통신포트의 논리 레벨은 3.3V 이기 때문에 반드시 회로도를 참고하시기 바랍니다.

 Revision history.

1. 2016.5.4   : First version is released.
2. 2020.1.10 : 데이터 시트 통신 지연시간 변경에 따른 코드 수정.
************************************************************************/


#include<SPI.h>

#define OBJECT  0xA0                    // 대상 온도 커맨드
#define SENSOR  0xA1                    // 센서 온도 커맨드

const int chipSelectPin   = 10;
unsigned char T_high_byte;
unsigned char T_low_byte;
unsigned char Timer1_Flag=0;
```

```
int iOBJECT, iSENSOR;      // 부호 2byte 온도 저장 변수

void setup()
{
  /* Setting SCE & SPI */
  digitalWrite(chipSelectPin , HIGH);          // SCE High Level
  pinMode(chipSelectPin , OUTPUT);             // SCE OUTPUT Mode
  SPI.setDataMode(SPI_MODE3);                  // SPI Mode
  SPI.setClockDivider(SPI_CLOCK_DIV16);        // 16MHz/16 = 1MHz
  SPI.setBitOrder(MSBFIRST);                   // MSB First
  SPI.begin();                                 // Initialize SPI

  delay(500);                                  // Sensor initialization time
  Timer1_Init();                               // Timer1 setup : 500ms(2Hz) interval
  Serial.begin(9600);
  interrupts();                                // enable all interrupts
}
int SPI_COMMAND (unsigned char cCMD)
{
    digitalWrite(chipSelectPin , LOW);         // SCE Low Level
    delayMicroseconds(10);                     // delay(10us)
    SPI.transfer(cCMD);                        // transfer 1st Byte
    delayMicroseconds(10);                     // delay(10us)
    T_low_byte = SPI.transfer(0x22);           // transfer 2nd Byte
    delayMicroseconds(10);                     // delay(10us)
    T_high_byte = SPI.transfer(0x22);          // transfer 3rd Byte
    delayMicroseconds(10);                     // delay(10us)
    digitalWrite(chipSelectPin , HIGH);        // SCE High Level

    return (T_high_byte<<8 | T_low_byte);      // 온도값 return
}
ISR(TIMER1_OVF_vect)                           // interrupt service routine (Timer1 overflow)
{
  TCNT1 = 34286;                               // preload timer : 이 값을 바꾸지 마세요.
  Timer1_Flag = 1;                             // Timer 1 Set Flag
}
```

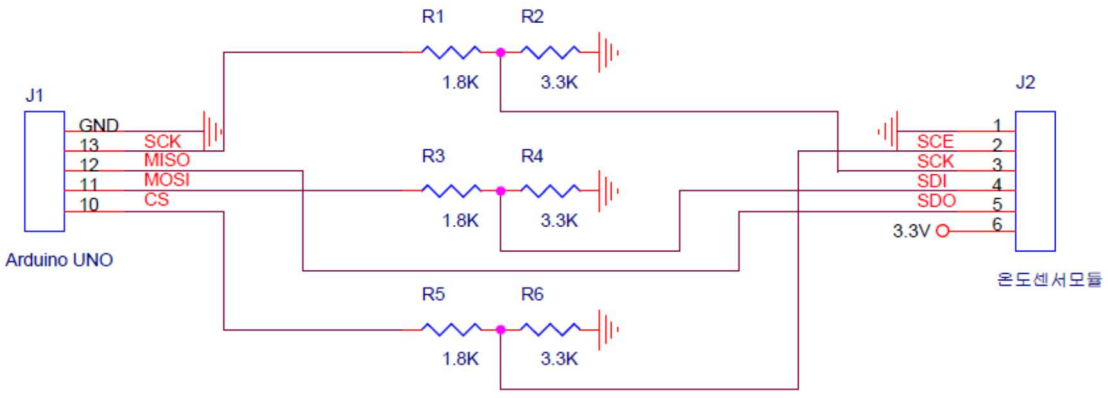
```
void Timer1_Init(void)
{
   TCCR1A = 0;
   TCCR1B = 0;
   TCNT1 = 34286;                          // preload timer 65536-16MHz/256/2Hz
   TCCR1B |= (1 << CS12);                  // 256 prescaler
   TIMSK1 |= (1 << TOIE1);                 // enable timer overflow interrupt
}
void loop()
{
   if(Timer1_Flag)                         // 500ms 마다 반복 실행(Timer 1 Flag check)
   {
      iOBJECT= SPI_COMMAND(OBJECT);        // 대상 온도 Read
      delayMicroseconds(10);               // delay(10us)
      iSENSOR = SPI_COMMAND(SENSOR);       // 센서 온도 Read

      Serial.print("Object Temp : ");      // 하이퍼터미널 출력
      Serial.print(float(iOBJECT)/10,1);
      Serial.print("       Sensor Temp : ");
      Serial.println(float(iSENSOR)/10,1);

   }
}
```

※ 아두이노 UNO 와 DTPM 연결 회로도.

MISO 포트를 제외한 나머지 포트는 5V 로 동작하는 아두이노와 연결시 반드시 아래 회로도 대로 연결해야 합니다. 아두이노가 아닌 3.3V 로 동작하는 그 외 MCU 와 연결할 경우는 저항 없이 바로 연결하면 됩니다.



▶ Appendix - C (Example Code - 라즈베리파이)

하단 코드는 라즈베리파이 코드 예제 입니다. 예제 코드는 쇼핑몰에서 다운로드 가능합니다.

```
/*
*       description : DTPM series Example Code.
 *            Version : 1.0
 *            Board : Raspberry 2 Model B V1.1
 *            Support Module : DTPM11, DTPM81, DTPM151
 *       Revision history.
 *       1.0 2020.01.02 : First version is released.
 */

#include <stdint.h>
#include <stdio.h>
#include <string.h>
#include "wiringPi.h"
#include "wiringPiSPI.h"

#define LED             21              // LED PORT NUM 21
#define SCE             22              // SCE PORT NUM 22
#define spi_chn0        0               // SPI Channel 0
#define SPEED_1MHz      1000000         // SPI Speed : 1MHz
#define SPI_MODE3       3               // SPI MODE
#define OBJECT  0xA0                    // COMMAND(Read Object Temp.)
#define SENSOR  0xA1                    // COMMAND(Read Sensor Temp.)

int16_t iSensor, iObject;


int16_t SPI_COMMAND(uint8_t ADR){
        uint8_t Data_Buf[3];
        Data_Buf[0] = ADR;
        Data_Buf[1] = 0x22;
        Data_Buf[2] = 0x22;
```

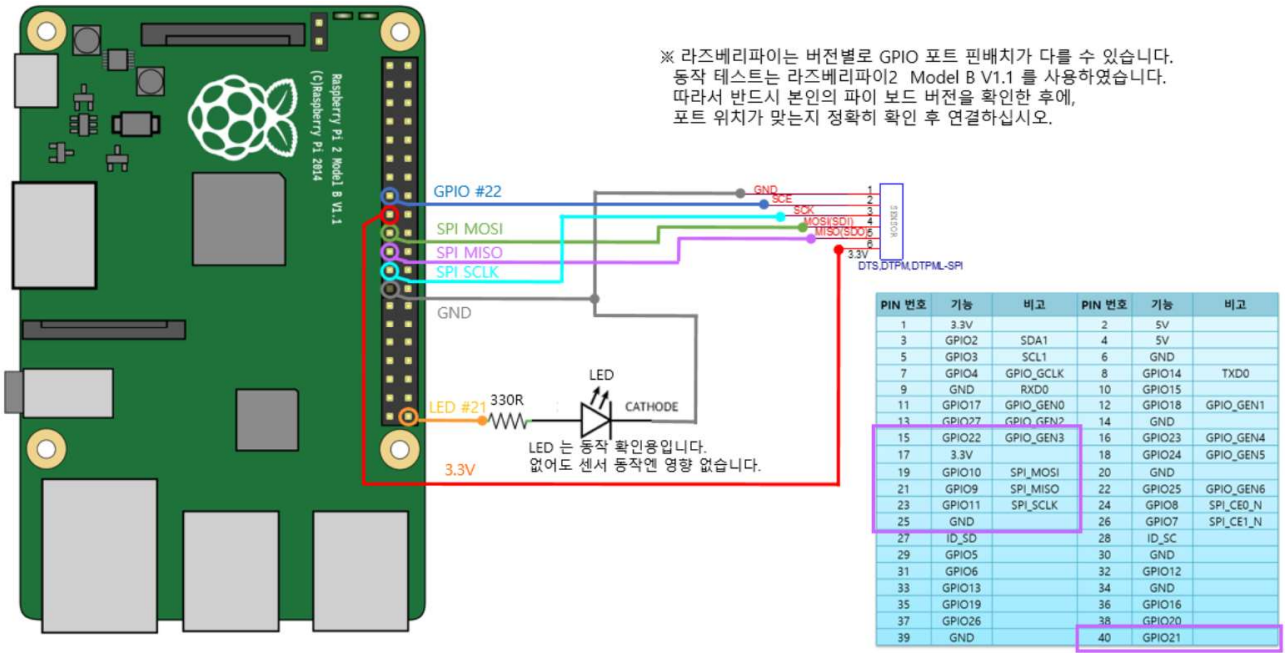
```
        digitalWrite(SCE, 0);                       // SCE LOW
        delayMicroseconds(10);                      // delay 10us

        wiringPiSPIDataRW (spi_chn0, Data_Buf, 1);  // transfer 1st byte.
        delayMicroseconds(10);                      // delay 10us
        wiringPiSPIDataRW (spi_chn0, Data_Buf+1, 1); // transfer 2nd byte
        delayMicroseconds(10);                      // delay 10us
        wiringPiSPIDataRW (spi_chn0, Data_Buf+2, 1); // transfer 3rd byte
        delayMicroseconds(10);                      // delay 10us

        digitalWrite(SCE, 1);                       // SCE HIGH
        return (Data_Buf[2]*256+Data_Buf[1]);       // High + Lo byte
}

int main(void){
        wiringPiSetup();                            // Wiring Pi setup
        if(wiringPiSetupGpio() == -1)
        return 1;
        pinMode(LED, OUTPUT);                       // LED Port Output (not necessary)
        pinMode(SCE, OUTPUT);                       // SCE Port Output
        digitalWrite(SCE,1);                        // SCE high

        wiringPiSPISetupMode(spi_chn0, SPEED_1MHz, SPI_MODE3); //SPI0, 1MHz, SPI Mode3 Setting
        delay(500);                                 // wait 500ms
        while(1){
                iSensor = SPI_COMMAND(SENSOR);      // Read Sensor temp.
                digitalWrite(LED, 1);               // LED ON(not necessary)
                delayMicroseconds(10);              // delay 10us
                iObject = SPI_COMMAND(OBJECT);      // Read Object temp.
                digitalWrite(LED,0);                // LED OFF(not necessary)
                delay(500);                         // Wait 500ms
                printf("Sensor : %5.1f   , Object : %5.1f ₩n", (double)iSensor/10, (double)iObject/10);
        }
        return 0;
}
```



PIN 번호	기능	비고	PIN 번호	기능	비고
1	3.3V		2	5V	
3	GPIO2	SDA1	4	5V	
5	GPIO3	SCL1	6	GND	
7	GPIO4	GPIO_GCLK	8	GPIO14	TXD0
9	GND	RXD0	10	GPIO15	
11	GPIO17	GPIO_GEN0	12	GPIO18	GPIO_GEN1
13	GPIO27	GPIO_GEN2	14	GND	
15	GPIO22	GPIO_GEN3	16	GPIO23	GPIO_GEN4
17	3.3V		18	GPIO24	GPIO_GEN5
19	GPIO10	SPI_MOSI	20	GND	
21	GPIO9	SPI_MISO	22	GPIO25	GPIO_GEN6
23	GPIO11	SPI_SCLK	24	GPIO8	SPI_CE0_N
25	GND		26	GPIO7	SPI_CE1_N
27	ID_SD		28	ID_SC	
29	GPIO5		30	GND	
31	GPIO6		32	GPIO12	
33	GPIO13		34	GND	
35	GPIO19		36	GPIO16	
37	GPIO26		38	GPIO20	
39	GND		40	GPIO21	

Additional Information

- manufacturer : Diwell Electronics Co., Ltd. <(주)디웰전자>
- Homepage : www.diwell.com
- shopping mall : www.diwellshop.com
- Phone : +82-70-8235-0820
- Fax : +82-31-429-0821
- Quotation request : sale01@diwell.com, sale02@diwell.com
- Technical support : tech01@diwell.com, dsjeong@diwell.com
- 본 문서의 내용은 사전 통보 없이 변경될 수 있습니다.
- 쇼핑몰 내 제품 상세 페이지에서 최신 데이터시트가 제공됩니다.

Revision History

Version	Date	Description
1.0	2014-06-10	First version is released.
1.3	2014-09-11	Page11. FOV 그림 오류 수정
1.4	2016-08-31	입력 전압 3.3V 고정. DW Testboard 신제품 내용 추가 Arduino 코드 / 회로도 추가
2.0	2020-01-02	**1. DW Testboard 단종. 신규 DW-LINK 내용 추가** **2. SPI 통신 스펙 변경** **2.1 온도읽기 시퀀스 byte간 지연시간 축소 10ms → 10us** **2.2 방사율 R/W 시 byte 간 지연시간 수정 →35us** **3. PCB Rev2 용 dimension 수정 및 그 외 표기 오류 정정** **4. 라즈베리파이용 코드 추가**
2.1	2025-11-04	**FOV change**