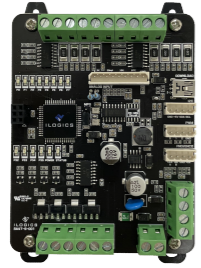


MPINO SERIES MPINO-8A4T-S

사 용 설 명 서

저희 (주)아이로직스 제품을 구입해 주셔서 감사합니다.



사용 전에 안전을 위한 주의사항을 반드시 읽고 사용하십시오.

□ 안전을 위한 주의사항

- ※ ‘안전을 위한 주의사항’은 제품을 안전하고 올바르게 사용하여 사고나 위험을 미리 막기 위한 것이므로 반드시 지켜야 합니다.
 - ※ 주의사항은 ‘경고’와 ‘주의’ 두 가지로 구분되어 있으며, ‘경고’와 ‘주의’의 의미는 다음과 같습니다.
- 지시사항을 위반하였을 때.
- ⚠ **경고** 심각한 상해나 사망이 발생할 가능성이 있는 경우
 - ⚠ **주의** 경미한 상해나 제품 손상이 발생할 가능성이 있는 경우
- ※ 제품과 취급설명서에 표시된 그림기호의 의미는 다음과 같습니다.
- ⚠는 특정조건 하에서 위험이 발생할 우려가 있으므로 주의하라는 기호입니다.

⚠ 경고

- 인명이나 재산상에 영향이 큰 기기(예: 원자력 제어장치, 의료기기, 선박, 차량, 철도, 항공기, 연소장치, 안전장치, 방법/방재장치 등)에 사용할 경우에는 반드시 2중으로 안전장치를 부착한 후 사용해야 합니다. 화재, 인사사고, 재산상의 막대한 손실이 발생할 수 있습니다.
- 자사 수리 기술자 이외에는 제품을 개조하지 마십시오. 감전이나 화재의 우려가 있습니다.

⚠ 주의

- 실외에서 사용하지 마십시오. 제품의 수명이 짧아지는 원인이 되며 감전의 우려가 있습니다. 본 제품은 실내 환경에 적합하도록 제작되었습니다. 실내가 아닌 외부환경으로부터 영향을 받을 수 있는 장소에서 사용할 수 없습니다. (예 : 비, 황사, 먼지, 서리, 햇빛, 결로 등)
- 인화성, 폭발성 가스 환경에서 사용하지 마십시오. 화재 및 폭발의 우려가 있습니다.
- 사용 전압 범위를 초과하여 사용하지 마십시오. 제품이 파손될 수 있습니다.
- 전원의 극성 등 오배선을 하지 마십시오. 제품이 파손될 수 있습니다.
- 진동이나 충격이 많은 곳에서 사용하지 마십시오. 제품이 파손될 수 있습니다.
- 청소 시 물, 유기 용제를 사용하지 마십시오. 감전 및 화재의 우려가 있습니다.

□ 손해배상책임
 (주)아이로직스는 제품을 사용하다 발생하는 인적, 물적 자원에 대해 책임을 지지 않습니다. 충분한 테스트와 안전장치를 사용하여 주시기 바랍니다.

□ 사양서

구 분	개 수	접점명	설 명
전 원	-	전원전압	• DC 12V ~ 24V
디지털 입력	8 포인트 <절연>	P0 ~ P7	• 오퍼레이팅 입력 전압 : DC 0 ~ 40V • HIGH 인식 전압 :DC 12V ~ 24V • 8P / 1COM • NPN 및 PNP 입력가능
트랜지스터 출력 <Sink 출력>	4 포인트 <절연>	P32 ~ P35	• 오퍼레이팅 출력 전압 - 0~30V D.C • 최대 출력 허용전류 : 1A / 1POINT • V+, V- 단자에는 외부전압을 연결
아날로그 입력	6 포인트 <비절연>	AI45(A0) AI46(A1)	• 오퍼레이팅 입력 전압: DC 0~5V, • 분해능 : 10Bit (0~1023) • 입력저항 : 100MΩ (±1%)
		AI47(A2) AI48(A3)	• 오퍼레이팅 입력 전압: 0~20mA, 4~20mA • 분해능 : 0~20mA : 10Bit (0~1,023) 4~20mA : (0~818) • 입력저항 : 250Ω (±1%)
		AI49(A4) AI50(A5)	• 오퍼레이팅 입력 전압: DC 0V ~ 10V • 분해능 : 10Bit (0~1023) • 입력저항 : 200KΩ (±1%)
온도센서 입력	2 포인트 <비절연>	NTEMP 51(A6), 52(A7)	• 오퍼레이팅 온도 입력 : -40℃~120℃ • 온도센서 : NTC 3950K 10KΩ(25℃) • 분해능 : 0.1℃ (0~40℃ 기준)
펄스 입력	2 포인트 <비절연>	고속카운터 24, 25	• 오퍼레이팅 입력 전압 : DC 0 ~ 5V • HIGH 인식 전압 : DC 2V 이상 • 입력가능 주파수 : 최대 50kHz
펄스 출력	3 포인트 <비절연>	PWM 21, 22, 23	• 오퍼레이팅 출력 전압 - LOW(DC 0V), HIGH(DC 5V) • 오퍼레이팅 최대 출력 전류 : 30mA
통신 채널	1채널 <비절연>	I ² C	• I2C 마스터/슬레이브 지원
	1 채널 <비절연>	RS232	• Modbus RTU Master, Slave 라이브러리 지원

□ 메모리 사양서

- 128Kbyte Flash Memory
- 4Kbyte SRAM Data Memory
- 4Kbyte EEPROM Memory

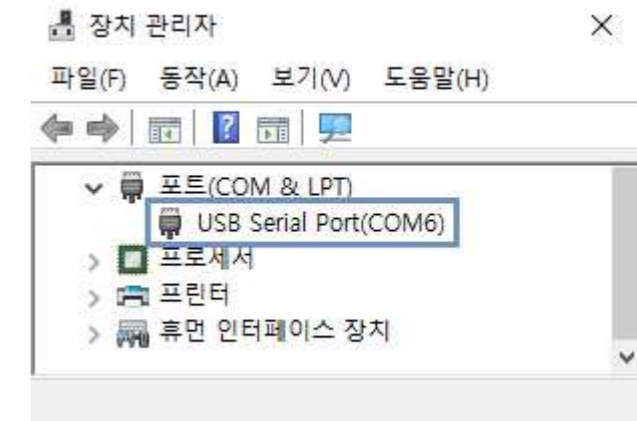
□ 사용방법 [요약]

아이로직스 블로그에서 아두이노 IDE 소프트웨어를 다운로드 받고 설치합니다. (<https://blog.naver.com/ilogics/222367876903>)

위 링크에서 MPINO-8A4T-S 설정 방법을 읽어주시길 바랍니다.

“MP 다운로드 케이블”을 MPINO-8A4T-S”에 연결합니다.

윈도우에서 장치관리자를 실행하여 하기와 같이 “USB Serial Port”가 표시되는지 확인하고 COMx에서 x에 해당하는 포트번호를 확인합니다.



만약, 드라이버가 나타나지 않는다면 (주)아이로직스 홈페이지의 자료실에서 “다운로드 케이블 드라이버” 게시물에서 FTDI 드라이버를 다운로드 받아 설치합니다. (<https://www.ilogics.co.kr/article/자료실/7/18/>)

아이로직스 블로그에서 아두이노 IDE 보드파일을 다운로드 받고 설치합니다. (<https://blog.naver.com/ilogics/222894192984>)

Arduino IDE에서 MPINO-8A4R(T)-S를 선택합니다. (메뉴 -> 툴 -> 보드 -> ILOGICS)

Arduino IDE에서 장치관리자에서 확인한 COM포트를 선택합니다. (메뉴 -> 툴 -> 포트)

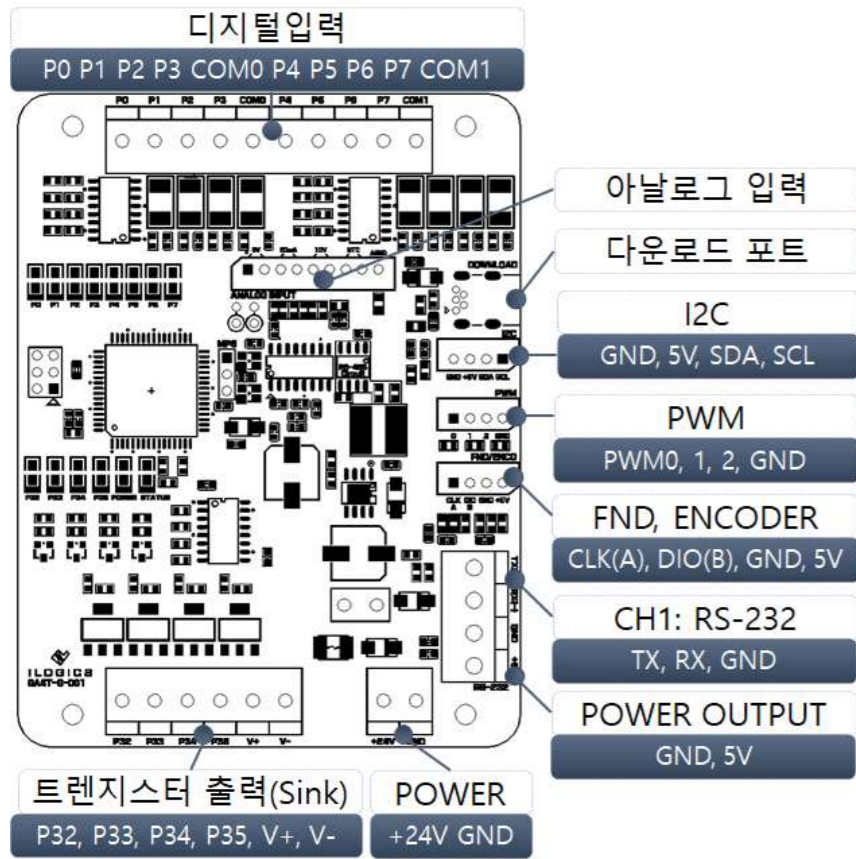
프로그래밍을 하고, 업로드를 합니다.

□ 명령어 설명서

Arduino IDE에서 도움말 -> 참조를 실행하거나 다음 링크에서 확인할 수 있습니다. (<https://www.arduino.cc/reference/ko/>)

EEPROM과 I2C(Wire) 등을 보다 쉽게 사용할 수 있는 라이브러리는 다음 링크에서 확인할 수 있습니다. (<https://www.arduino.cc/reference/en/libraries/>)

기능별 위치



전원

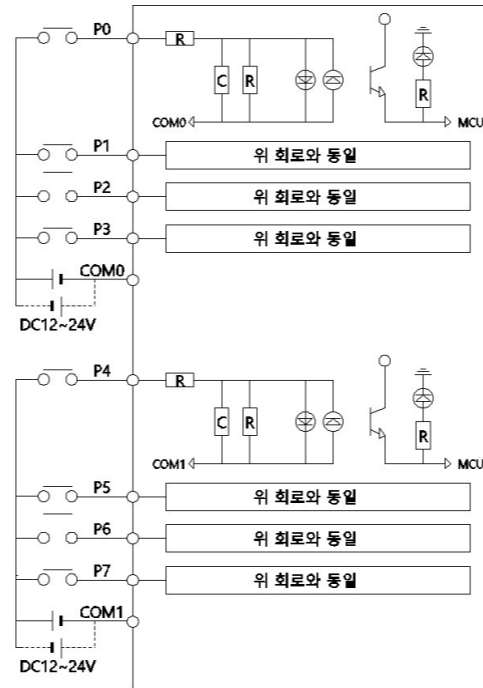
- 전원입력은 DC 12V ~ 24V를 사용할 수 있습니다. DC 12V ~ 24V는 레귤레이터를 통하여 DC 5V로 전환되어 내부회로에 공급됩니다.
- 다운로드 포트에 USB 연결선으로 컴퓨터와 연결하면, USB의 5V 전원을 사용하여 제품이 동작됩니다.
- USB 전원으로 제품을 동작할 경우, USB 전원으로 아날로그 입력의 기준전압으로 사용하게 됩니다. 때문에, 아날로그의 값이 많이 흔들릴 수 있습니다.
- 전원출력부의 +5V 단자는 0.5A 이하의 전원출력으로 사용하실 수 있습니다.

정전유지

- MPINO-8A4T-S 제품은 DC 5V 전원으로 모든 동작이 가능하도록 설계되어 있습니다. DC +5V 단자 대에 배터리를 연결하여, 정전 시에는 배터리의 전원으로 절체 되도록 하여 정전유지가 가능합니다.
- 비휘성 메모리인 EEPROM을 이용하여 메모리를 보존할 수 있습니다. 단, EEPROM은 100,000번 이상 기록(Write)을 할 경우, 해당 섹션의 불량 발생 할 수 있으므로 빠른 속도로 변경되는 데이터를 기록하는 것은 올바르지 않습니다. EEPROM 사용법은 아이로직스 블로그에 포스팅 된 "MPINO STUDIO C코드와 Ladder Logic 활용하기 2편" 게시 글을 참고하시면 됩니다.

디지털 입력

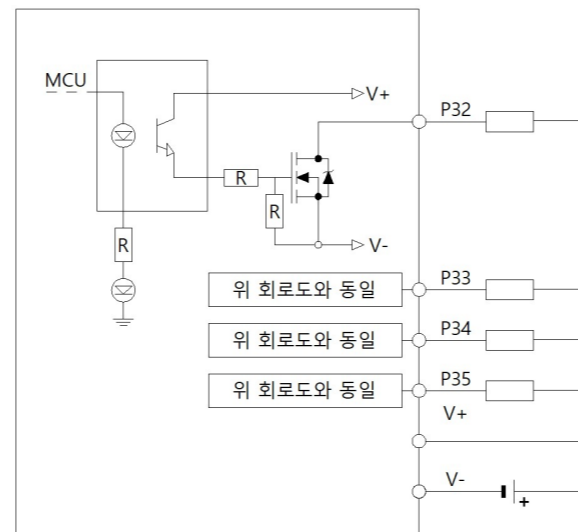
- P(0) ~ P(7)에 12V ~ 24V D.C가 스위치, 센서 등에 의해 입력된다면, COM0은 GND를 연결해야 합니다. 반대로, P(0) ~ P(7)에 GND가 스위치, 센서 등에 의해 입력된다면, COM0은 12V ~ 24V D.C를 연결해야 합니다.



- 관련 명령어
 - `pinMode(Pin, INPUT/OUTPUT)` Pin포트를 INPUT 또는 OUTPUT으로 설정.
 - `digitalRead(Pin)` Pin포트의 입력상태를 "0" 또는 "1"로 반환.
 - `digitalWrite(pin, 0/1)` pin포트의 출력상태를 LOW 또는 HIGH로 변환.

트랜지스터 출력

- 출력접점 메모리 P32 ~ P35의 메모리 상태가 HIGH가 될 때, 각각의 트랜지스터 출력 터미널블럭에 V-(GND)가 출력됩니다. (SINK 출력 : V-(GND)가 출력되는 방식입니다)
- V+ 단자는 DC 5~30V의 P(Plus) 전압을 연결해야 하며, V- 단자는 DC 5~30V의 N(Minus, GND) 전압을 연결해야 합니다.
- 각 포트당 최대 1A의 전류를 사용할 수 있으며, 부하에 걸린 전압이 최대 100V를 넘을 수 없습니다.
- 트랜지스터 출력이 ON되어 있을 때 출력 단자에 P(Plus) 전압을 인가하면, 쇼트 되어 트랜지스터 부품이 소손될 수 있습니다.



디지털 입/출력 예제

```
void setup() {
    pinMode(32, OUTPUT); // R(32)을 출력모드로 설정합니다.
}
void loop() {
    // P(0)가 HIGH 이면, P(32)를 ON 시킵니다.
    if (digitalRead(0) == 1) { digitalWrite(32, HIGH); }
    // P(0)가 HIGH가 아니면, 즉 LOW 이면, P(32)를 OFF 시킵니다.
    else { digitalWrite(32, LOW); }
}
```

1초마다 출력을 ON/OFF 시키는 예제

- `delay(ms)` 명령어를 사용하여 시간지연을 사용할 수 있습니다.

```
void setup() {
    pinMode(32, OUTPUT); // P(32)를 출력모드로 설정합니다.
}
void loop() {
    digitalWrite(32, HIGH); // P(32)를 ON 시킵니다.
    delay(1000);           // 1000ms 동안 기다립니다.
    digitalWrite(32, LOW); // P(32)를 OFF 시킵니다.
    delay(1000);          // 1000ms 동안 기다립니다.
}
```

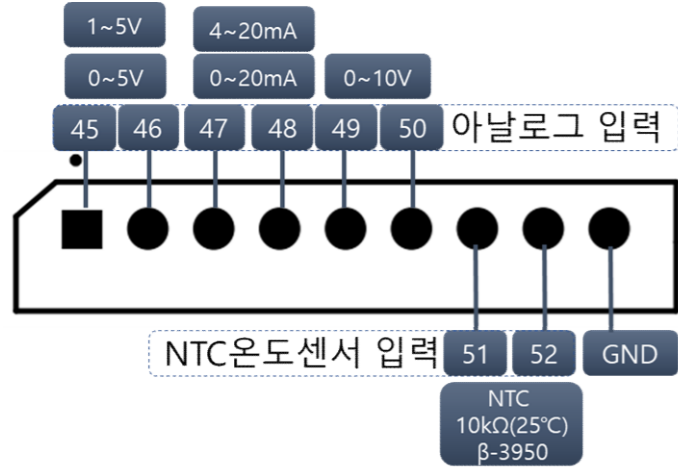
상태 LED

- LED_BUILTIN 변수명 또는 D20핀으로 제품에 삽입되어 있는 STATUS LED를 ON/OFF 시킬 수 있습니다.

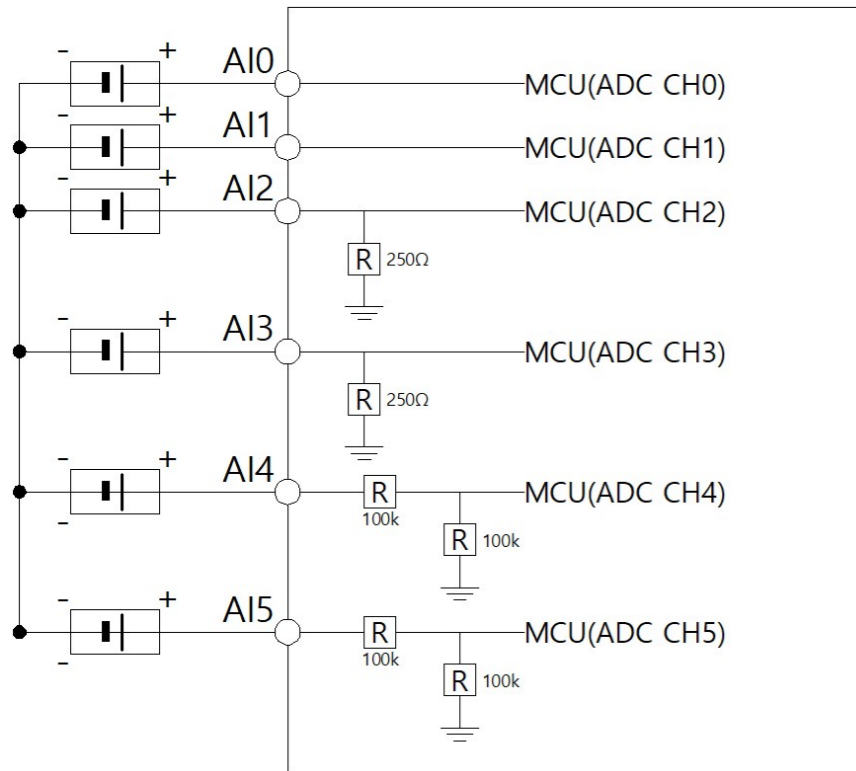
```
void setup() {
    pinMode(LED_BUILTIN, OUTPUT);
    //LED_BUILTIN을 출력모드로 설정
}
void loop() {
    // I(0)가 HIGH 이면, LED_BUILTIN를 ON 시킵니다.
    if (digitalRead(0) == 1) {
        digitalWrite(LED_BUILTIN, HIGH);
    }
    // I(0)가 HIGH가 아니면,
    // 즉 LOW 이면, LED_BUILTIN를 OFF 시킵니다.
    else {
        digitalWrite(LED_BUILTIN, LOW);
    }
}
```

□ 아날로그 입력

☞ 아날로그 입력포트 A0 ~ A5에 입력된 아날로그 신호를 analogRead(pin) 명령어를 사용하여 디지털 값으로 변환하여 사용합니다.



☞ 아날로그 입력포트의 회로는 아래와 같습니다.



☞ 관련 명령어

- **analogReference(EXTERNAL)** : 아날로그 입력의 기준전압을 MCU의 Vref 핀에 연결된 전압으로 설정합니다.
- **analogRead(Pin)** : Pin 포트의 아날로그 신호를 디지털 수치로 변환하여 반환합니다.

□ 아날로그 입력 프로그램 예

☞ A0 포트에 입력된 0V~5V의 전기신호를 0~1023의 디지털 값으로 변환하여 ADC0 변수에 저장하는 프로그램입니다. analogReference(EXTERNAL)로 기준 전압 값을 설정하고, analogRead(pin)으로 아날로그 값을 읽어 옵니다.

```

unsigned int ADC0; // ADC0 변수 생성

void setup() {
  // 아날로그 최댓값 기준을 VREF핀에 입력된 전압으로 설정
  analogReference(EXTERNAL);
}

void loop() {
  // A0에 입력된 0~5V를 0~1023의 디지털 값으로 변환하여 ADC0에 저장
  ADC0 = analogRead(A0);
}
    
```

□ 아날로그 입력 값 스케일 계산

☞ analogRead(pin) 명령어를 이용하여 읽어온 0 ~ 1023 디지털 값을 프로그램에서 로직으로 사용하기 위해 실제 센서의 Range로 디지털 값을 변환하기 위해 스케일 계산법으로 디지털 범위를 재설정하는 방법입니다.

☞ $Scale = (in / 1023) * (Scale_Max - Scale_Min) - Scale_Min$

```

unsigned int ADC0; // ADC0 변수 생성

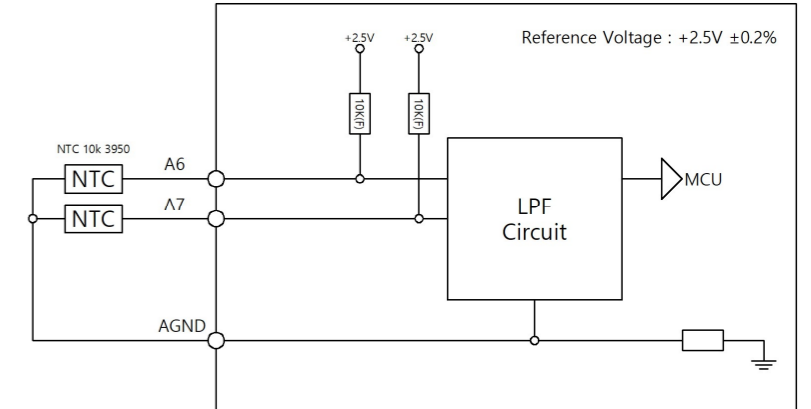
void setup() {
  // 아날로그 최댓값 기준을 VREF핀에 입력된 전압으로 설정
  analogReference(EXTERNAL);
}

void loop() {
  //A0에 입력된 0~5V를 0~1023의 디지털 값으로 변환하여 ADC0에 저장
  ADC0 = analogRead(A0);
  // 스케일공식 = (in/in_Max) * (Scale_Max - Scale_Min) + Scale_Min
  // 0V일 때는 0, 5V일 때는 3000으로 디지털 값 범위를 재설정.
  ADC0 = ((unsigned long)ADC0 * (3000-0)) / 1023 + 0;
}
    
```

□ 온도센서 입력

☞ 온도센서 입력 A6, A7은 써미스터(NTC ,3950, 10KΩ)의 저항 값을 디지털 값(-40℃ ~ 120℃)으로 변환하여 사용합니다.

☞ NTC 온도센서의 연결은 극성이 없습니다.



□ NTC 써미스터를 이용한 온도 값 읽어오는 예제

☞ 써미스터 핀 번호는 A6과 A7핀입니다.

☞ NTC는 25℃일 때, 10kΩ 저항이 되는 제품을 사용해야 합니다.

☞ 하기의 예제는 NTC 10k 3950를 사용하였습니다.. NTC 센서 값이 다르면 3950.0F 값을 변경해야 합니다. 보다 정밀한 측정을 위해서는 보간법을 이용하는 것이 좋습니다.

```

unsigned int Temp // Temp 변수 생성
void setup() {
  // 아날로그 최댓값 기준을 VREF핀에 입력된 전압으로 설정
  analogReference(EXTERNAL);
}

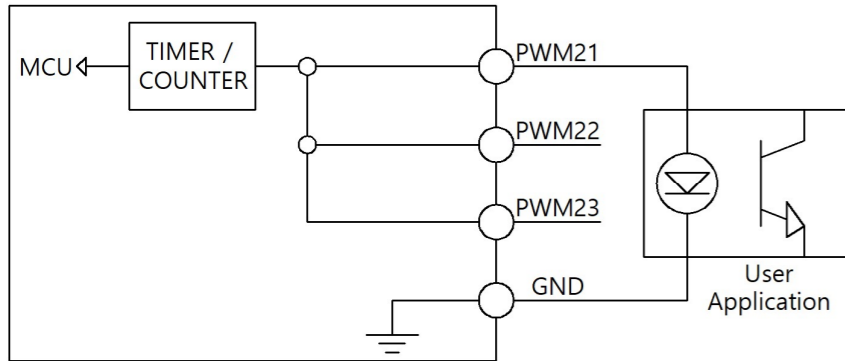
void loop() {
  // 아날로그입력 A6번 포트의 온도 값을 읽어 와서
  // Temp변수에 저장.
  Temp = ntcRead(analogRead(A6));
  // Temp가 251 이면, 25.1도입니다.
}

int ntcRead(unsigned int RawADC)
{
  float v;
  v = (1023.0F / (float)RawADC) - 1.0F;
  v = 10000.0F / v;
  float steinhart;
  steinhart = v / 10000.0F;
  steinhart = log(steinhart);
  steinhart /= 3950.0F;
  steinhart += 1.0F / (25.0F + 273.15F);
  steinhart = 1.0F / steinhart;
  steinhart -= 273.15F;
  return (unsigned int)(steinhart * 10);
}
    
```


□ PWM 고속펄스 출력

☞ PWM은 Pulse Width Modulation 의 약자로서 Width를 조절할 수 있는 펄스라는 의미입니다. PWM은 다른 장비와의 인터페이스로 많이 사용되어 집니다. 많이 사용하는 곳은 모터 드라이버의 속도 및 회전각을 조절하기 위해 사용됩니다.

☞ 21, 22, 23은 MCU의 타이머3 자원을 사용합니다.



☞ 관련 명령어

analogWrite(Pin, Duty) : PWM(Pin)포트에 Duty길이의 펄스를 출력합니다.

Pin : PWM21은 21, PWM22는 22를 사용합니다.

Duty : 0 ~ 255(Default), 0 ~ 65535(Max : 레지스터리 변경 후)

□ PWM 펄스출력 예제

☞ 디지털입력 P(0)가 ON되면, 고속펄스출력 21번 핀에 Duty비가 50%인 연속적인 펄스를 출력하고 P(0)가 OFF되면, 펄스출력을 정지시킵니다.

```
void setup(void) {
}

void loop(void) {
  if (digitalRead(0)==1) { analogWrite(21, 127); }
  else { analogWrite(21, 0); }
}
```

□ PWM의 Duty를 16비트로 변경하는 방법

☞ MCU에 내장되어 있는 타이머 자원의 레지스터리를 수정하여 Duty의 최댓값을 16비트인 65535로 변경할 수 있습니다.

```
void setup(void) {
  //21, 22, 23번 핀의 타이머3의 카운트 최댓값을 65535로 변경
  TCCR3A=0xAA; TCCR3B=0x1A; ICR3 = 65535;
}

void loop(void) {
  if (digitalRead(0)==1) { analogWrite(21, 32767); }
  else { analogWrite(21, 0); }
}
```

☞ 고속카운터는 MCU의 타이머3 자원을 사용합니다.

☞ 타이머3 자원을 이미 사용 중인 경우 인터럽트 기능을 통해 고속카운트 기능을 지원합니다.

☞ 25번 핀은 인터럽트기능을 통해 고속카운트 기능을 지원합니다.

□ 고속카운터 입력 사용방법 (16비트)

```
unsigned int HCNT3;

void setup(void) {
  // 타이머3 자원을 고속카운터 모드로 설정
  TIMSK = 0x00; TCCR3A = 0x00; TCCR3B = 0x07; TCNT3 = 0x00;
}

void loop(void) {
  HCNT3 = TCNT3; // 타이머3 카운트 값을 HCNT3 변수에 저장
}

void hcntReset() {
  // 타이머3 카운트 값을 0으로 리셋
  TCNT3 = 0;
}
```

□ 고속카운터 입력 사용방법 (32비트)

☞ 고속카운터 입력이 16비트를 초과해서 사용해야 할 경우, MCU에 내장된 타이머 레지스터가 16비트이므로 오버플로우 인터럽트를 이용하여 오버플로우 된 값을 변수로 카운트해야 합니다.

```
unsigned long HCNT3;
unsigned int _ofcH3;

void setup(void) {
  // 타이머3 자원을 고속카운터 모드로 설정
  TIMSK = 0x01; TCCR3A = 0x00; TCCR3B = 0x07; TCNT3 = 0x00;
  // 타이머3 카운트 오버플로우 인터럽트 허용
  ETIMSK = (1<<TOIE3);
}

void loop(void) {
  // 타이머3 카운트 값과 오버플로우 카운트 값을 HCNT3 변수에 저장
  HCNT3 = (unsigned long)_ofcH3 << 16 | TCNT3;
}

// 타이머3 카운트 오버플로우 인터럽트
ISR ( TIMER3_OVF_vect ) { TCNT3=0; _ofcH3++; }
```

□ 인터럽트 (attachInterrupt)

☞ 디지털신호의 입력을 받아 빠르게 처리해야 하는 사항이 있을 때 사용합니다.

☞ MPINO-8A4T-S에서는 CLK(24)와 DIO(25) 포트를 통하여 두 개의 인터럽트 핀을 사용할 수 있습니다.

☞ 입력전압은 DC 3V ~ 5V입니다. 과전압 인가시 MCU가 손상될 수 있습니다.

☞ 관련 명령어

attachInterrupt(digitalPinToInterrupt(pin)), ISR, mode);

- pin : 24 또는 25 (CLK 또는 DIO포트)
- ISR : 호출되는 함수명
- mode : LOW, CHANGE, RISING, FALLING
 - LOW : 하강검출 (입력상태가 ON에서 OFF로 될 때)
 - CHANGE : 변경검출 (입력상태가 변경될 때)
 - HIGH : 상승검출 (입력상태가OFF에서 ON으로 될 때)

☞ 인터럽트는 CLK과 DIO 포트를 통하여 사용이 가능합니다.

☞ DIO 입력이 OFF이었다가 ON될 때, countFunc()함수를 호출하는 예제입니다.

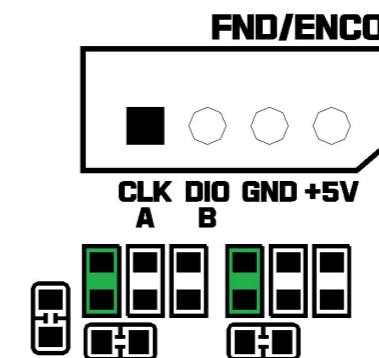
```
const byte pin = 25;
unsigned int count = 0;

void setup() {
  Serial.begin(9600); // 다운로드포트를 9600보레이트로 오픈
  //DIO포트에 상승엿지 입력이 검출되면, countFunc 함수 호출실행
  attachInterrupt(digitalPinToInterrupt(pin), countFunc, RISING);
}

void loop() {
  Serial.println(count);
}

void countFunc() {
  count++;
}
```

□ 엔코더 풀업 저항



☞ 초록색으로 표기된 위치에 풀업 저항을 삽입할 수 있습니다.

☞ 출하시 엔코더 입력단자에는 100kΩ 풀다운 저항이 삽입되어 있습니다.

□ I2C 통신포트

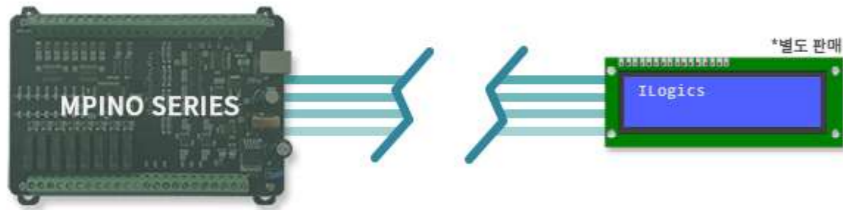
- 총 1개의 I2C 통신포트를 제공합니다.
- 1:N통신이 가능합니다.
- 명령어는 링크를 참조해 주시기 바랍니다.
<https://www.arduino.cc/en/Reference/Wire>

□ RTC (Real Time Clock)

- 정확한 시계기능을 사용하고자 할 경우에는 쇼핑몰 액세서리 카테고리에서 판매중인 DS3231 RTC모듈을 사용하시기 바랍니다.
- I2C 통신포트의 4핀 커넥터에 DS3231 RTC모듈을 연결하여 사용 가능합니다.

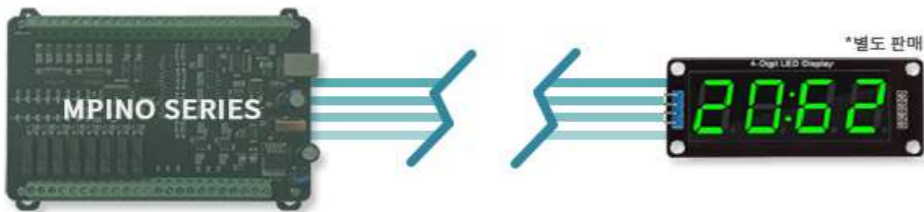
□ 캐릭터LCD 연결

- I2C 통신포트 4Pin 커넥터에 캐릭터 LCD에 연결하여 PLC의 상태값 등을 표시할 수 있습니다.
- 1602(16캐릭터/2줄) 및 2004(20캐릭터/4줄) 캐릭터 LCD를 쇼핑몰 액세서리 카테고리에서 별도 판매하고 있습니다.
- 저렴하게 디스플레이를 구현하고 싶다면 캐릭터 LCD를 연결할 수 있고 고급스럽게 디스플레이를 구현하고 싶다면 터치디스플레이 HMI를 연결할 수 있습니다.



□ 7세그먼트(FND) 연결

- ENC/FND 4핀 커넥터에 7세그먼트(FND)를 연결하여 아두이노PLC의 상태값 등을 표시할 수 있습니다.
- 소수점 표시할 수 있는 7세그먼트와 시간을 표시할 수 있는 7세그먼트를 쇼핑몰 액세서리 카테고리에서 별도 판매하고 있습니다.



□ 디버깅

- Debug는 Serial 함수를 이용해 주세요.

```
unsigned int ADC0; // 아날로그 입력값 변수를 선언
void setup() {
  // 아날로그입력 기준전압을 VREF핀에 연결되어 있는 5V로 설정
  analogReference(EXTERNAL);
  // 업로드 포트를 보레이트가 9600인 시리얼포트로 정의
  Serial.begin(9600);
  for(int k = 32; k <= 35; k++) {
    pinMode(k, OUTPUT);
  }
}
void loop() {
  // 디지털입력이 ON 되면 트랜지스터출력을 ON,
  // OFF일 때 트랜지스터출력을 OFF
  for (int k = 0; k <= 3; k++) {
    if (digitalRead(k)==1) digitalWrite(32+k, HIGH);
    else digitalWrite(32 + k, LOW);
  }for (int k = 4; k <=7; k++) {
    if (digitalRead(k)==1) {
      Serial.print(k);      Serial.println(":on");
    }
  }
  // A0에 입력된 아날로그 신호를 0~1023으로 변환하여 ADC0 변수에 저장
  ADC0 = analogRead(A0);
  // 아날로그입력값 ADC0을 스케일 연산하여 0 ~ 3000으로 범위 변경
  // 스케일공식 = (in/in_Max) * (Scale_Max - Scale_Min) + Scale_Min
  ADC0 = ((unsigned long)ADC0 * (3000-0)) / 1023 + 0;
  Serial.print("Analog Input Value : "); Serial.println(ADC0);
  delay(500);
}
```

□ 시리얼 통신포트

- RS232는 Serial1 함수를 이용해 주세요.
- 모드버스 통신은 Arduino IDE에서 툴 -> 라이브러리 관리..에서 다운로드 받아서 라이브러리를 등록하여 이용하실 수 있습니다.
- 자사 라이브러리 ILIB.h를 통해서 Modbus Master, Slave를 이용하실 수 있습니다.

□ 시리얼 통신포트

- RS-232 1채널의 통신포트를 지원합니다.
- 1:1 통신이 가능합니다.
- Modbus RTU Master 라이브러리를 지원합니다.
- Modbus RTU Slave 라이브러리를 지원합니다.
- IBUS Master / Slave 라이브러리를 지원합니다.
(입/출력 접점이 부족할 경우 사용)

□ RS-232 -> RS-485 or UART

- RS-232 통신을 RS-485 또는 UART 통신으로 변경하고자 할 경우에는 쇼핑몰 액세서리 카테고리에서 별도로 판매중인 컨버터 모듈을 사용해주시기 바랍니다.

□ RS-232 -> RS-485 IC 변경

- RS-232 IC칩을 RS-485 IC칩으로 변경하고자 할 경우에는 쇼핑몰 옵션에서 IC 변경을 체크하여 구매해주시길 바랍니다.

□ MPINO STUDIO

- 저희 ㈜아이로직스에서는 산업에서 사용하기 쉽도록 Arduino 와 Ladder Logic을 모두 사용하여 프로그램 할 수 있는 MPINO STUDIO를 무료로 제공하고 있습니다. (단, MPINO-16A16R, MPINO-8A8R(T)-S, MPINO-8A4R(T)-S 제품은 MPINO STUDIO를 이용하실 수 없습니다.)

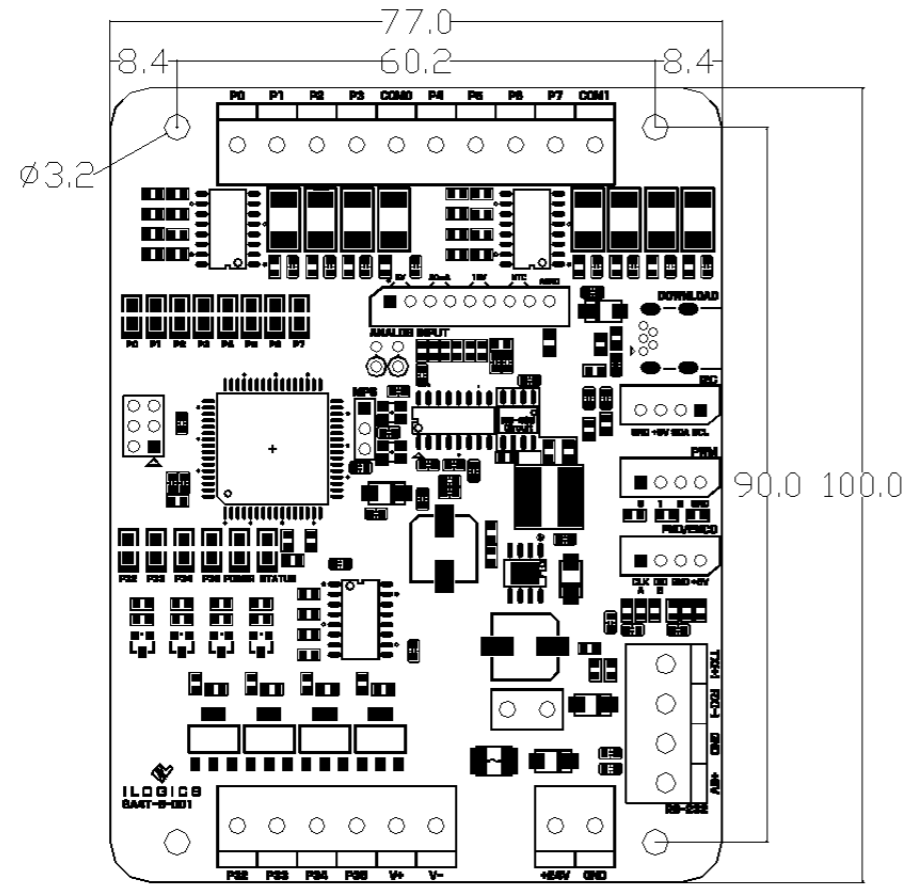
□ MP STUDIO

- 저희 ㈜아이로직스에서는 Ladder Logic만을 사용하여 프로그램 할 수 있는 MP STUDIO를 무료로 제공하고 있습니다. MP STUDIO는 MPS 시리즈 제품군에 사용할 수 있습니다.

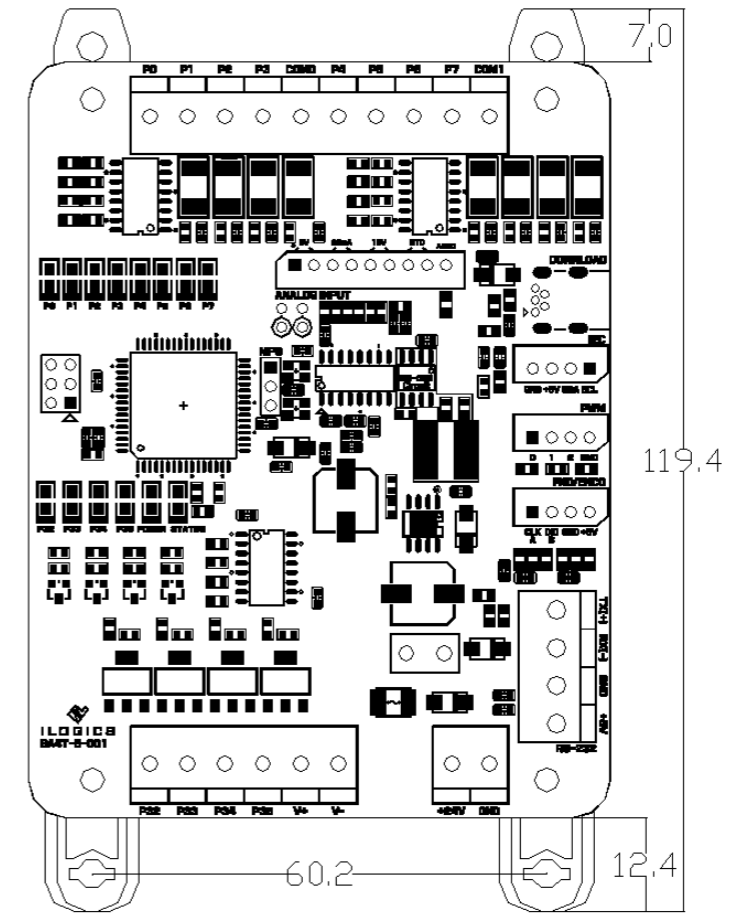
□ 감사드립니다.

- 저희 (주)아이로직스의 제품을 구매해주셔서 감사드립니다.
- 구매는 <https://www.ilogics.co.kr> 쇼핑몰에서 하실 수 있습니다.
- 구매/기술상담은 0507-1426-5027으로 전화 주시기 바랍니다.

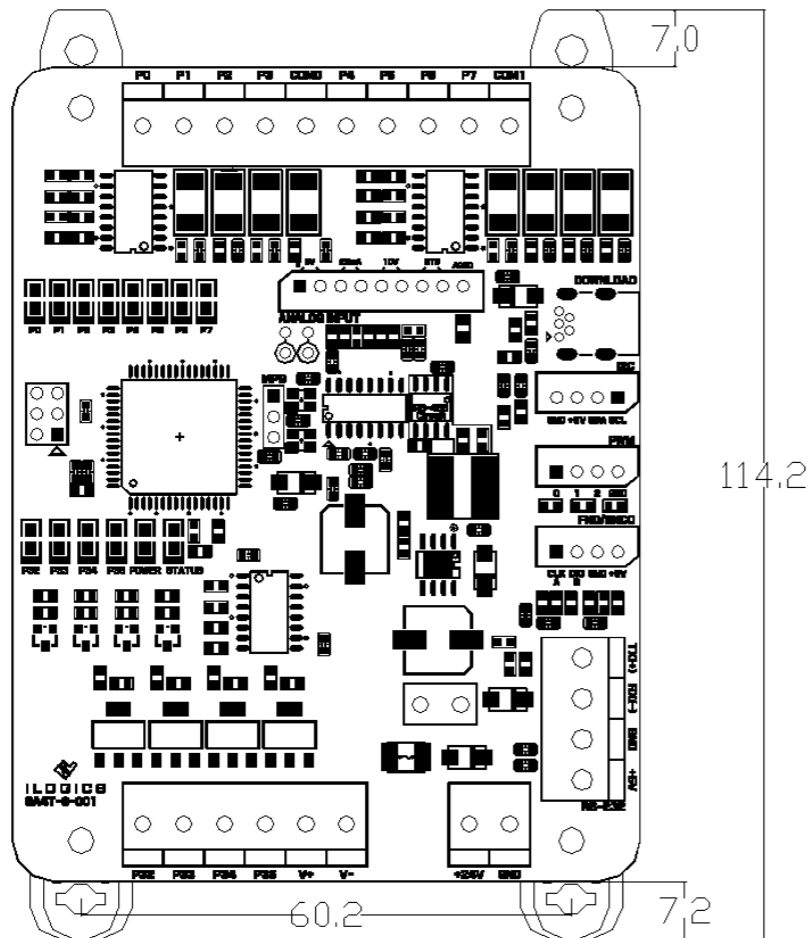
□ DIMENSION (PCB / 단레일 사용하지 않을 경우)



□ DIMENSION (클립 열었을 때 / 단레일 체결 전)



□ DIMENSION (클립 닫았을 때 / 단레일 체결 후)



□ DIMENSION (단레일 : 35mm)

