MPAINO SERIES MPAINO-48A48T

사용설명서 [MPINO STUDIO]

저희 ㈜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

1. 인명이나 재산상에 영향이 큰 기기(예: 원자력 제어장치, 의료기기, 선박, 차량, 철도, 항공기, 연소장치, 안전장치, 방범/방재장치 등)에 사용할 경우에는 반드시 2중으로 안전장치를 부착한 후 사용해야 합니다.
화재, 인사사고, 재산상의 막대한 손실이 발생할 수 있습니다.

2. 자사 수리 기술자 이외에는 제품을 개조하지 마십시오.
감전이나 화재의 우려가 있습니다.

⚠ 주의

1. 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)

2. 인화성, 폭발성 가스 환경에서 사용하지 마십시오.
화재 및 폭발의 우려가 있습니다.

3. 사용 전압 범위를 초과하여 사용하지 마십시오.
제품이 파손될 수 있습니다.

4. 전원의 극성 등 오배선을 하지 마십시오.
제품이 파손될 수 있습니다.

5. 진동이나 충격이 많은 곳에서 사용하지 마십시오.
제품이 파손될 수 있습니다.

6. 청소 시 물, 유기 용제를 사용하지 마십시오.
감전 및 화재의 우려가 있습니다.

❒ 손해배상책임

㈜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
보 드	-	-		• MPAINO-48A48R(T) • (Customized MEGA2560 Board)
전 원	-	전원전압		• DC 12V ~ 24V (DC24V 0.75A 이상)
디지털 입력	48 포인트 〈 절연 〉	0	D0 ~ D3 / COM0 D4 ~ D7 / COM1	• 오퍼레이팅 입력 전압 : DC 0 ~ 80V • HIGH 인식 전압 : DC 5V 이상 • 4P / 1COM • 1COM당 NPN 및 PNP 선택 입력가능
		1	D8 ~ D11 / COM0 D12 ~ D15 / COM1	
		2	D16 ~ D19 / COM0 D20 ~ D23 / COM1	
		3	D24 ~ D27 / COM0 D28 ~ D31 / COM1	
		4	D32 ~ D35 / COM0 D36 ~ D39 / COM1	
		5	D40 ~ D43 / COM0 D44 ~ D47 / COM1	
트랜지스터 출력 〈SINK〉	48 포인트 〈 절연 〉	0	D64 ~ D71 / COM2	• SINK 출력 - 모듈의 N24에 연결된 GND가 출력됨 • 오퍼레이팅 연결 전압 - DC 0 ~ 100V 1A이하 부하전압 • 8P / 1COM • 최대 출력 허용전류 - 1A / 1POINT - 8A / 1COM
		1	D72 ~ D79 / COM2	
		2	D80 ~ D87 / COM2	
		3	D88 ~ D95 / COM2	
		4	D96 ~D103 / COM2	
		5	D104 ~ D111/COM2	
통신 채널	1채널 〈비절연〉	I²C		• I2C (WIRE 라이브러리 사용)
	3 채널 〈비절연〉	RS232 RS485, UART		• Serial1 : RS232 • Serial2 : RS485 • Serial3 : UART (TX: D14, RX: D15)

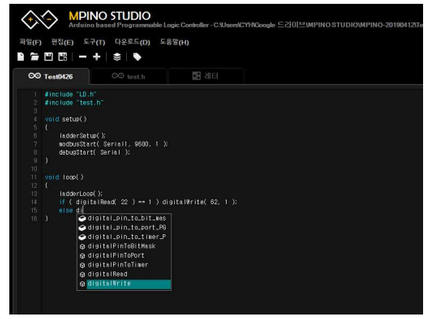
❒ 메모리 사양서

- 256Kbyte Flash Memory
- 8Kbyte Data Memory
- 4Kbyte EEPROM Memory

❒ MPINO STUDIO 사용방법 [요약]

✎ MPINO STUDIO 설치

- **아이로직스 쇼핑몰 자료실** 에서 "MPINO STUDIO"를 다운로드 받고, 설치합니다.
(https://www.ilogics.co.kr/article/자료실/7/24/)
- MPINO STUDIO의 자세한 설명은 "MPINO STUDIO 사용설명서"를 참조 바랍니다.



[아두이노 C언어]

[LADDER LOGIC]

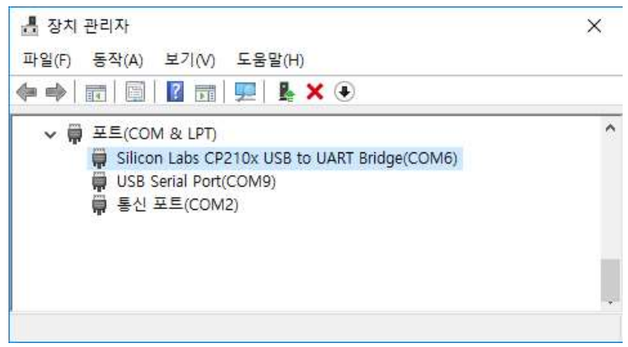
✎ 컴퓨터의 USB포트와 제품(MPAINO-48A48T)에 다운로드 포트를 연결합니다.

✎ 아이로직스 자료실에서 **MPINO STUDIO 사용설명서**를 다운로드 & 설치합니다.
(https://www.ilogics.co.kr/article/자료실/7/24/)

✎ CP2102 USB 드라이버를 설치합니다.
(https://ilogics.synology.me/Drivers/CP210x_Windows_Drivers.zip)
"CP2102 드라이버가 기설치되어 있다면 설치하지 않으셔도 됩니다."
"CP2102 드라이버는 Silicon Labs 반도체 회사에서 드라이버를 제공합니다."

✎ "MPINO STUDIO"를 실행하고 도구창에서
"도구 -> 디바이스 -> MPAINO-48A48R(T) -> MPAINO-48A48R(T)" 을 선택합니다.

✎ 다운로드 포트를 장치관리자에서 확인합니다.
- 윈도우의 장치관리자에서 아래 그림처럼 선택된 COM포트를 확인합니다.



✎ "도구 -> 포트"에서 위에서 확인한 COM포트를 선택합니다.

✎ "도구 -> 하드웨어"에서 LADDER LOGIC에서 사용할 포트번호를 바인딩 합니다.
바인딩을 모두 한 저장파일이 MPINO STUDIO/Example 폴더에 있습니다.

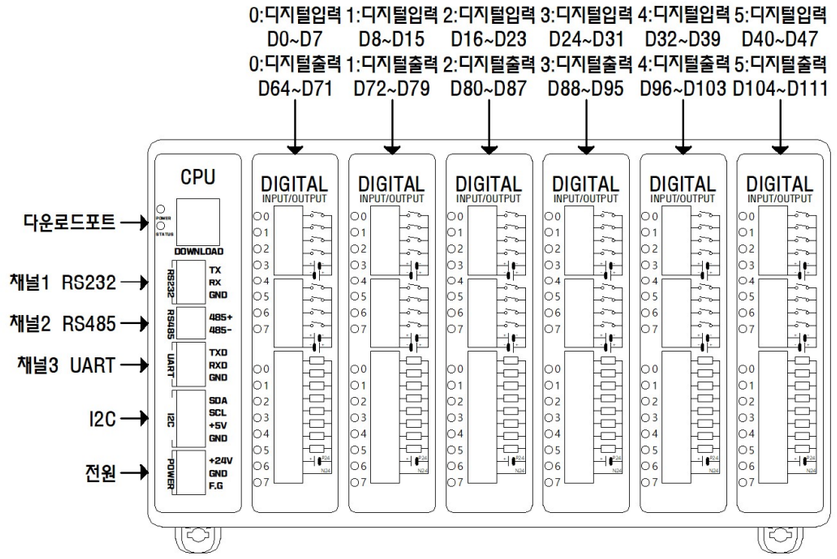
✎ 아두이노 프로그램 작성 창 및 래더 작성 창에서 프로그램을 작성하고 다운로드를 합니다.

❒ 아두이노 명령어

✎ 아두이노 홈페이지(https://www.arduino.cc/)에서 아두이노의 다양한 명령어 및 사용법을 확인하실 수 있습니다. (https://www.arduino.cc/reference/en/)

✎ EEPROM과 I2C(Wire) 등을 보다 쉽게 사용할 수 있는 라이브러리는 다음 링크에서 확인할 수 있습니다.(https://www.arduino.cc/en/Reference/Libraries)

❒ 기능별 위치



❒ 전원

✎ 전원입력은 DC 12V~24V를 사용할 수 있습니다. DC 12V~24V는 Regulator를 통하여 DC 5V로 전환되어 내부회로에 전원을 공급합니다.

✎ 다운로드 포트에 USB 연결선으로 컴퓨터와 연결하면, 컴퓨터의 5V 전원을 사용하여 제품이 동작됩니다.

✎ I2C 터미널블럭의 +5V 단자는 최대 1A의 DC 5V 전원을 출력할 수 있습니다.

❒ 정전유지

✎ 제품은 DC 5V전원으로 모든 동작이 가능하도록 설계되어 있습니다. 따라서 DOWNLOAD (USB-B TYPE) 커넥터에 배터리 등을 연결하여, 정전 시에도 제품동작 및 메모리를 유지하게 할 수 있습니다.

✎ 배터리 연결이 불가할 경우와 장기간 정전 시에도 데이터의 보존을 원할 경우에는 MCU에 내장된 EEPROM을 사용해주시기 바랍니다. 비활성 메모리인 EEPROM을 이용하여 메모리를 보존할 수 있습니다. 단, EEPROM은 100,000번 이상 기록(Write)을 할 경우, 해당 섹션의 불량이 발생할 수 있으므로 수시로 변경되는 데이터를 기록하는 것은 올바르지 않습니다.

❒ LADDER LOGIC 메모리

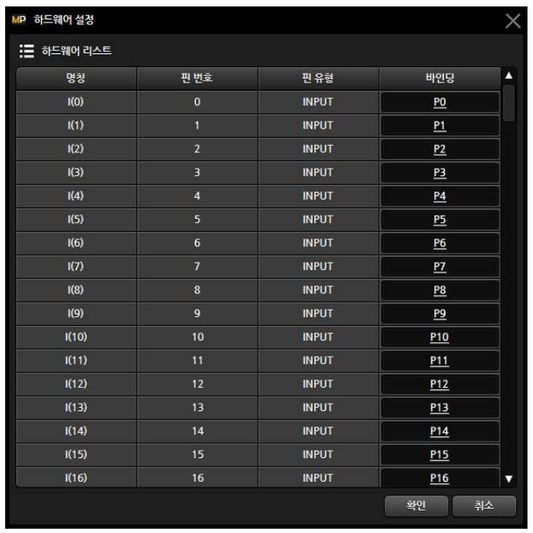
✎ LADDER LOGIC 접점명은 P, M, D, C, T, R 메모리가 있으며, 메모리마다의 최대 사용크기는 “도구->메모리 구성”에서 변경할 수 있습니다. (LADDER LOGIC에서 메모리를 너무 높이면 아두이노 프로그램 작성 창에서 변수를 많이 사용할 수 없습니다)

메모리	형식	특징
P	비트(Px) 워드(WPx)	제품의 입/출력포트 상태
M	비트(Mx) 워드(WMx)	제품의 내부메모리로서 주로 비트를 사용
D	비트(Dx.b) 워드(Dx)	제품의 내부메모리로서 주로 워드를 사용 Unsigned Int (16비트)로서 0~65535
C	비트(Cx.b) 워드(Cx)	제품의 내부메모리로서 주로 카운터 펑션블럭에 사용 (CTU, CTD 펑션블럭) Unsigned Int (16비트)로서 0~65535
T	비트(Tx.b) 워드(Tx)	제품의 내부메모리로서 주로 타이머 펑션블럭에 사용 (TON, TOFF, TMON... 펑션블럭) Unsigned Int (16비트)로서 0~65535
R	실수(Rx)	32비트 부동 소수점
@	특수메모리	@10 (10ms 마다 한 스캔동안 ON) @100 (100ms 마다 한 스캔동안 ON) @1000 (1초마다 한 스캔동안 ON) @F10 (10ms ON/ 10ms OFF 토글 ON/OFF) @F100 (100ms ON/ 100ms OFF 토글 ON/OFF) @F1000 (1초 ON/ 1초 OFF 토글 ON/OFF) @ON (항상 ON) @OFF (항상 OFF) @BEGIN (전원 투입시 최초 한 스캔동안 ON)

x는 10진수 0~9,10,11,12.... , b는 비트의 위치 0~F (16진수)

❒ 하드웨어 설정

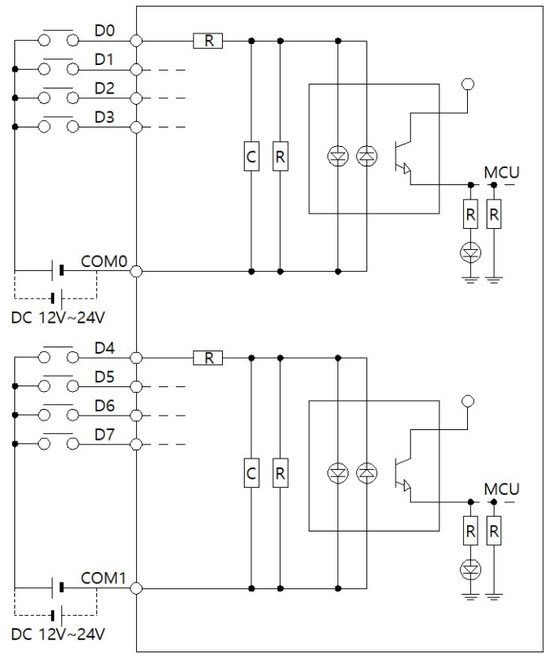
✎ LADDER LOGIC을 사용하기 위해서는 “도구 -> 하드웨어”에서 래더에서 사용할 포트 번호를 바인딩 해야 합니다.



❒ 디지털 입력

✎ 디지털입력 포트는 D0~D47 핀 번호를 사용하며, 포토커플러를 사용하여 다양한 전압을 입력 받을 수 있고 양방향(NPN, PNP) 입력을 지원합니다. 또한, 외부와 내부의 회로를 분리하여 노이즈에 강하며, 채터링 방지가 됩니다.

✎ 디지털입력 핀에 DC 5V~24V가 스위치, 센서 등에 의해 입력된다면, COM에는 GND를 연결해야 합니다. 반대로, 디지털입력 핀에 GND가 스위치, 센서 등에 의해 입력된다면, COM에는 DC 5V~24V를 연결해야 합니다.



※ 모든 모듈(D8 ~ D47)은 위에서 표시한 0번 모듈 D0 ~ D7의 회로도와 동일합니다. (포트번호만 다릅니다)

※ 필수설치파일을 설치하지 않으면 제어 할 수 없습니다.

✎ 관련 명령어

digitalRead(Pin) Pin포트의 입력상태를 “0” 또는 “1”로 반환.

digitalWrite(pin, 0/1) pin포트의 출력상태를 LOW 또는 HIGH로 변환.

❒ 디지털 입/출력 예제

✎ 디지털입력 D0에 따라 릴레이출력 D32를 ON/OFF하는 예제입니다.

```
void setup() {
}
void loop() {
    // D0이 HIGH 이면, D64 트렌지스터를를 ON 시킵니다.
  if (digitalRead(0) == 1) { digitalWrite(64, HIGH); }
  // D0이 HIGH가 아니면, 즉 LOW 이면, D64 릴레이를 OFF 시킵니다
  else { digitalWrite(64, LOW); }
}
```

❒ 트랜지스터 출력

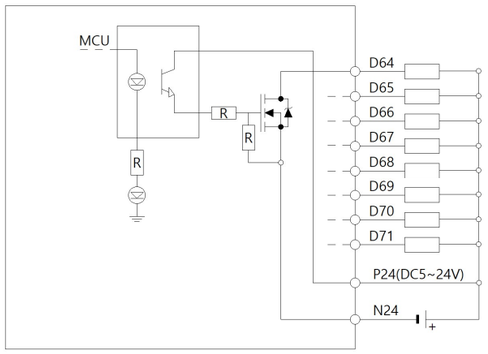
✎ 출력접점 메모리 D64 ~ D111의 메모리 상태가 HIGH가 될 때, 각각의 트랜지스터 출력 터미널블럭에 N24가 연결되어 GND(N24)가 출력됩니다. (SINK 출력 : GND가 출력되는 방식입니다)

✎ P24 터미널블럭은 DC 5~24V의 전원을 연결해야 하며, N24는 P24의 GND를 연결해야 합니다.

✎ 각 포트당 최대 1A의 전류를 사용할 수 있으며, 부하에 걸린 전압이 최대 100V를 넘을 수 없습니다.

✎ MPAINO-48A48R(T) 제품은 pinMode(pin) 함수로 입력/출력 모드를 설정하지 않고 사용해주시기 바랍니다. 함수를 사용하지 않아도 자동으로 설정됩니다.

✎ 필수설치파일을 설치하지 않으면 제어 할 수 없습니다.

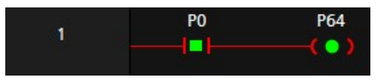


✎ 모든 모듈(D72 ~ D111)은 위에서 표시한 0번 모듈 D64 ~ D71의 회로도와 동일합니다. (포트번호만 다릅니다)

❒ 디지털 입력 및 디지털 출력 간단 예제

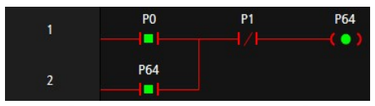
✎ 디지털 입력 I(0):P0 포트에 전기신호가 입력되면 I(0):P0 접점이 On되고, O(32):P64 출력접점이 On되어 O(32):P64 포트와 COM0 포트가 물리적으로 연결됩니다. 반대로 I(0):P0 접점이 OFF되면, O(32):P64 접점도 OFF됩니다.

[P0 OFF -> P64 OFF]

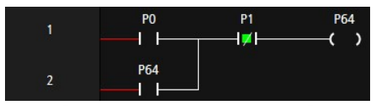


[P0 ON -> P64 ON]

✎ 디지털입력 I(0):P0 포트가 ON되면, 출력 O(32):P64가 ON되며 보조접점 O(32):P64에 의해 자기유지 되어 O(32):P64는 계속 ON됩니다. 이후, 입력 I(1):P1 포트가 ON되면, O(32):P64는 자기유지가 풀리며 OFF 됩니다.



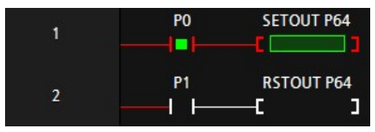
[P0 ON -> P64 ON]



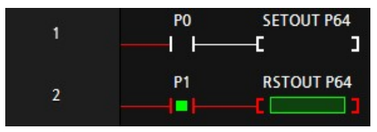
[P1 ON -> P64 OFF]

✎ 디지털입력 I(0):P0 점점이 ON되면, O(32):P64 출력접점이 ON되고,

✎ 디지털입력 I(1):P1 접점이 ON되면, O(32):P64 출력접점이 OFF됩니다.



[P0 ON -> P64 ON]



[P1 ON -> P64 OFF]

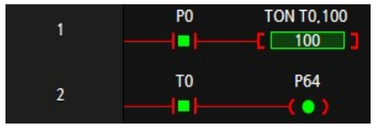
❒ 타이머 펑션블럭

✎ 타이머 펑션블럭은 TON, TOFF, TMON 3가지 유형을 제공합니다.

펑션블럭		사용법	예시
TON	On Delay Timer	TON Timer,Time(10ms)	TON T0,100 (1초 On Delay)
TOFF	Off Delay Timer	TOFF Timer,Time(10ms)	TOFF T1,200 (2초 Off Delay)
TMON	Monostable Timer	TMON Timer,Time(10ms)	TMON T2,300 (3초 동안 ON)
TAON	On Delay Timer	TAON Timer,Time(100ms)	TAON T0,10 (1초 On Delay)
TAOFF	Off Delay Timer	TAOFF Timer,Time(100ms)	TAOFF T1,20 (2초 Off Delay)
TAMON	Monostable Timer	TAMON Timer,Time(100ms)	TAMON T2,30 (3초 동안 ON)

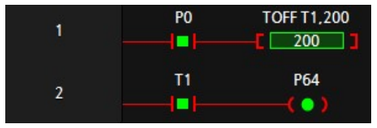
✎ TON 타이머는 ON을 Delay 해주는 타이머입니다. TON 펑션블럭의 실행조건이 ON 된 이후, 타이머의 설정 값까지 T0 보조접점을 OFF 시켰다가 설정 값에 도달했을 때 ON 시킵니다.

[P0 ON -> T0 타이머 작동]

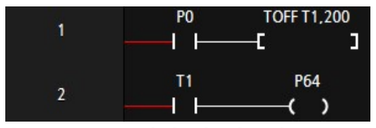


[타이머 설정 값 도달 -> T0 ON]

✎ TOFF 타이머는 OFF를 Delay 해주는 타이머입니다. TOFF 펑션블럭의 실행조건이 ON되었을 때, 타이머의 보조접점을 ON 시켜주었다가 TOFF 펑션블럭의 실행조건이 OFF되어도 바로 보조접점을 OFF 시키지 않고 타이머의 설정시간 후에 OFF시킵니다.

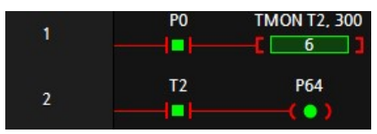


[P0 ON -> T1 ON]

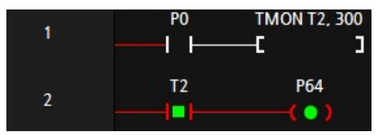


[P0 OFF-> 설정 값 미도달-> T1 ON]

✎ TMON 타이머는 실행조건이 ON되면, 실행조건이 계속 ON이 되거나 OFF되더라도 보조접점이 설정시간 동안 ON 시켰다가 설정시간에 도달한 후에 OFF시킵니다.



[P0 ON -> T2 ON]



[P0 OFF-> 설정 값 미도달-> T2 ON]
[P0 OFF -> 설정 값 도달-> T2 OFF]

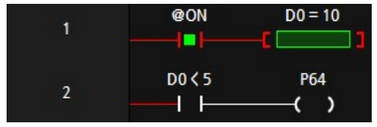
[P0 ON -> 설정 값 미도달-> T2 ON]
[P0 ON -> 설정 값 도달-> T2 OFF]

❒ 비교접점

✎ NO(Normal Open)접점에서 "비교명1 연산자 비교명2"를 입력하여 비교접점을 사용할 수 있습니다.

비교식	설 명
D0 == D1	D0과 D1이 같으면
D0 != D1	D0과 D1이 같지 않으면
D0 > D1	D0이 D1보다 크면
D0 < D1	D0이 D1보다 작으면
D0 >= D1	D0이 D1보다 크거나 같으면
D0 <= D1	D0이 D1보다 작거나 같으면

✎ 부호 있는 대소 관계를 비교하기 위해서는 DW(D0) == DW(D1)와 같이 형변환을 사용해야 합니다.



[D0 < 5 대소 관계 불일치]

[D0 < 20 대소 관계 일치]

❒ 이동 펑션블럭

✎ LADDER LOGIC의 MOV 펑션블럭을 사용할 수 있고 아래와 같이 C언어 형식처럼 사용할 수도 있습니다.

[D1 메모리 값을 D0에 복사]

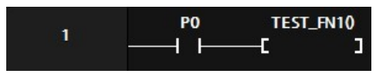
[D1*20/3.14산술연산 값을 D0에 저장]

❒ 아두이노 함수 실행

✎ 아두이노 프로그램 창에서 생성한 함수를 LADDER LOGIC에서 호출할 수 있습니다. 단, 래더에서 대문자만 사용이 가능하므로 함수명은 대문자로 만들어야 합니다.

```
14  void TEST_FN1()
15  {
16      // 실행코드
17  }
```

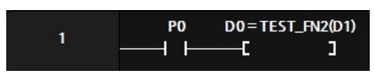
[아두이노 void 함수]



[래더에서 void 함수 호출]

```
19  int TEST_FN2( int In )
20  {
21      // 실행코드
22  }
```

[아두이노 int 함수]



[래더에서 int 함수 호출]

❒ 아두이노 프로그램에서 래더 메모리 사용

✎ LADDER LOGIC에서 사용흔 메모리와 같은 이름으로 아두이노에서 사용이 가능합니다. Ex) 아두이노에서 D0 = 10; 코딩을 하면 D0에 10이 저장됩니다.

❒ 형변환

✎ 워드 및 더블워드를 Singed 와 Unsigned로 다양하게 형변환이 가능합니다.

✎ 펑션블럭에서 DW(D0) = DW(D2) + 1 하게 되면 D2,D3의 값에 1을 더해서 D0,D1에 저장합니다.

❒ I2C 통신

✎ I2C 통신은 SDA(데이터), SCL(클럭) 2가닥의 통신선으로 구성되며, 1:N 통신이 가능합니다.

✎ I2C 통신은 최대 10m이내에 사용할 수 있으며, 전송선로가 길거나 노이즈가 많다면 통신 속도를 줄이는 것이 안정성에 도움이 됩니다.

✎ 통신 속도는 100kHz로 기본 설정돼 있으며, Wire.setClock(ColocFrequence) 명령어로 변경할 수 있습니다.

✎ #include "Wire.h" 를 아두이노 프로그램 창 상단에 기입해야 합니다.

✎ I2C 통신 및 시리얼 통신은 아두이노 프로그램 창에서 사용할 수 있습니다. LADDER LOGIC에서 사용할 수 없습니다.

✎ 아두이노의 사용명령어는 아두이노 홈페이지의 Wire Library를 참조해 주시기 바랍니다.

```
#include "LD.h"
#include "Wire.h"

void setup(void) {
        ladderSetup();
        Wire.begin();  // I2C를 마스터 모드로 시작합니다.
        Serial.begin( 9600 ); // 0번 채널(다운로드포트)를 Open 합니다.
}

void loop(void) {
        ladderLoop();
        // 슬레이브 Address를 1로 저장합니다.
        Wire.beginTransmission( 1 );
        Wire.write( 0x30 ); // 전송할 Data를 0x30으로 저장합니다.
        // I2C시작, Address 전송, 0x30과 0x31전송,
            I2C재시작을 순서대로 실행합니다.
        Wire.endTransmission( false );
        // 슬레이브 Address가 1인 디바이스에서 10 바이트를 읽어오고
           I2C를 정지시킵니다.
        Wire.requestFrom( 1, 10, true );
        // 읽은 바이트가 0이 될 때까지 루프를 형성
        while ( Wire.available() )
        {
                // 한 바이트를 읽어서iRxData에 저장
                byte iRxData = Wire.read();
                // iRxData를 시리얼 모니터로 전송
                Serial.println( iRxData );
        }
}
```

❒ 시리얼 통신

✎ 다운로드 채널은 Serial로 정의되어 있으며, 다운로드 및 Serial 키워드를 사용하여 디버깅에 사용합니다.

✎ 채널1은 Serial1로 정의되어 있으며, RS232 통신으로 사용할 수 있습니다.
(1:1 통신, 약 10m이내 통신거리)

✎ 채널2는 Serial2로 정의되어 있으며, RS485 통신으로 사용할 수 있습니다.
(1:N 통신, 약 1km 이내 통신거리)

✎ 채널3은 Serial3로 정의되어 있으며, UART 통신으로 사용할 수 있습니다.
(1:1 통신, 약 1m이내 통신거리)

✎ 관련 명령어
- **Serial.begin()** : 시리얼 포트를 Open합니다.
- **Serial.write(byte)** : 1개의 Byte를 전송합니다.
- **Serial.write(array, length)** : Array에서 Length만큼 전송합니다.
- **Serial.available()** : 수신된 Data(Byte)의 개수를 반환합니다.
- **Serial.Read()** : 수신된 1개의 Byte를 읽어 옵니다.

❒ Modbus RTU Slave

✎ 산업에서 범용적으로 사용하는 산업범용 프로토콜입니다.
✎ 통신영역은 LADDER LOGIC의 메모리를 참조합니다.
✎ MPINO STUDIO 사용설명서에서 자세한 사용방법을 확인하실 수 있습니다.
✎ 관련 명령어
- **modbusStart(Serial, BaudRate, SlaveAddress)** : Serial 포트를 BaudRate와 SlaveAddress로 modbus RTU slave로 지정.
- **modbusStop()** : Modus RTU Slave를 모두 해제 합니다.

```
#include "LD.h"

void setup(void) {
        ladderSetup();
        // Serial1 채널을 9600 보레이트와 1 슬레이브 어드레스로
           modbus RTU slave를 시작합니다.
        modbusStart( Serial1, 9600, 1 );
}

void loop(void) {
        D0 = 1234; //D0레지스터리에 1234값을 저장
        // D0는 0x0000 시작어드레스
}
```

❒ 디버깅

✎ 아두이노 프로그램의 setup() 함수에 debugStart(Serial)을 사용하면 LADDER LOGIC에서 모니터링을 사용하여 디버깅이 가능합니다.
✎ 아두이노 프로그램에서 print 및 println 등의 명령어를 사용하면 “도구->시리얼 모니터”에서 디버깅이 가능합니다.
✎ 아두이노 프로그램에서 Serial 포트로 디버깅을 하고 LADDER LOGIC에서도 Serial 포트로 모니터링 하면 충돌이 일어나기 때문에 debugStart(Serial1) 과 같이 모니터링에서 사용할 포트번호를 변경할 수 있습니다.

```
void setup() {
  Serial.begin(9600); // 업로드 포트를 9600 보레이트로 정의
}
void loop() {
  // 디지털입력이 ON 되면 D64를 ON, OFF일 때 D64를 OFF
  for (int k = 0; k < 8; k++) {
    if (digitalRead(2 + k)==1) digitalWrite(64 + k, 1);
    else digitalWrite(64 + k, 0);
  }
  // 디지털입력 D2가 HIGH일 경우 디버깅 포트로 “D2 HIGH"를 전송
  // 디지털입력 D2가 LOW일 경우 디버깅 포트로 “D2 LOW”를 전송
  if (digitalRead(2) == 1) Serial.println(“D2 HIGH");
  else if (digitalRead(2) == 0) Serial.println(“D2 LOW");
  delay(500);
}
```

❒ Arduino IDE

✎ 본 제품은 아두이노 플랫폼과 호환됩니다. Arduino IDE에서도 프로그램이 가능합니다.

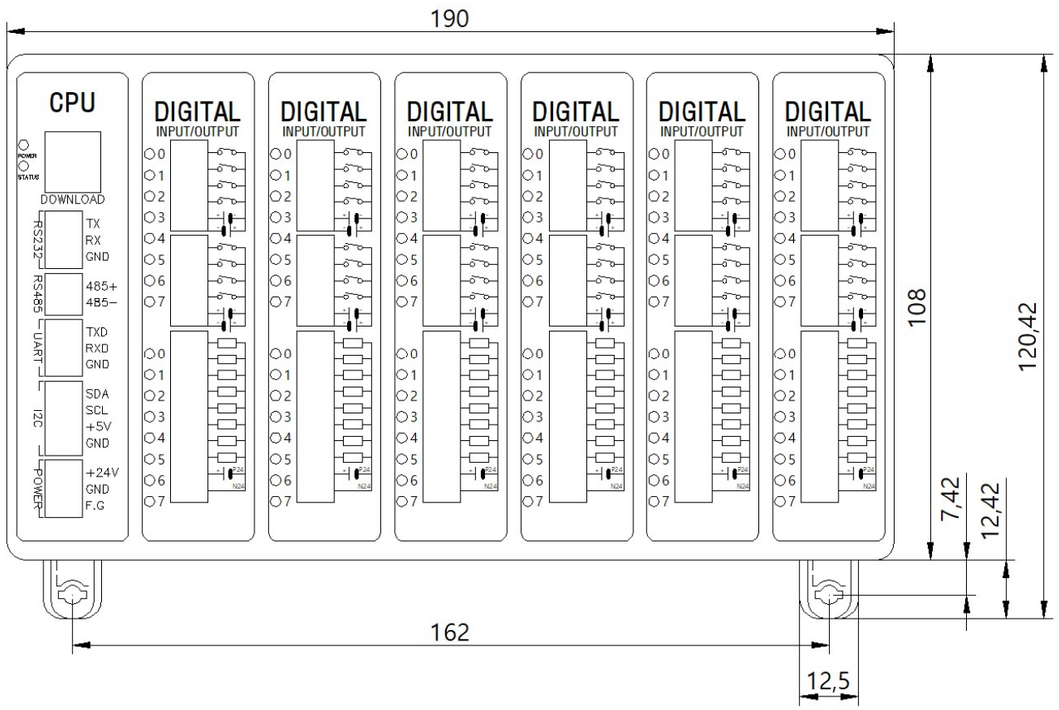
❒ MP STUDIO

✎ 저희 ㈜아이로직스에서는 Ladder Logic만을 사용하여 프로그램 할 수 있는 MP STUDIO를 무료로 제공하고 있습니다. MP STUDIO는 MPS 및 MPA 시리즈 제품군에 사용할 수 있습니다. MP STUDIO는 MPINO STUDIO보다 래더로직의 펑션블럭과 기능이 훨씬 많습니다. 래더로직을 중심적으로 사용하실려면 MPS 및 MPA 제품군을 검토해주시기 바랍니다.

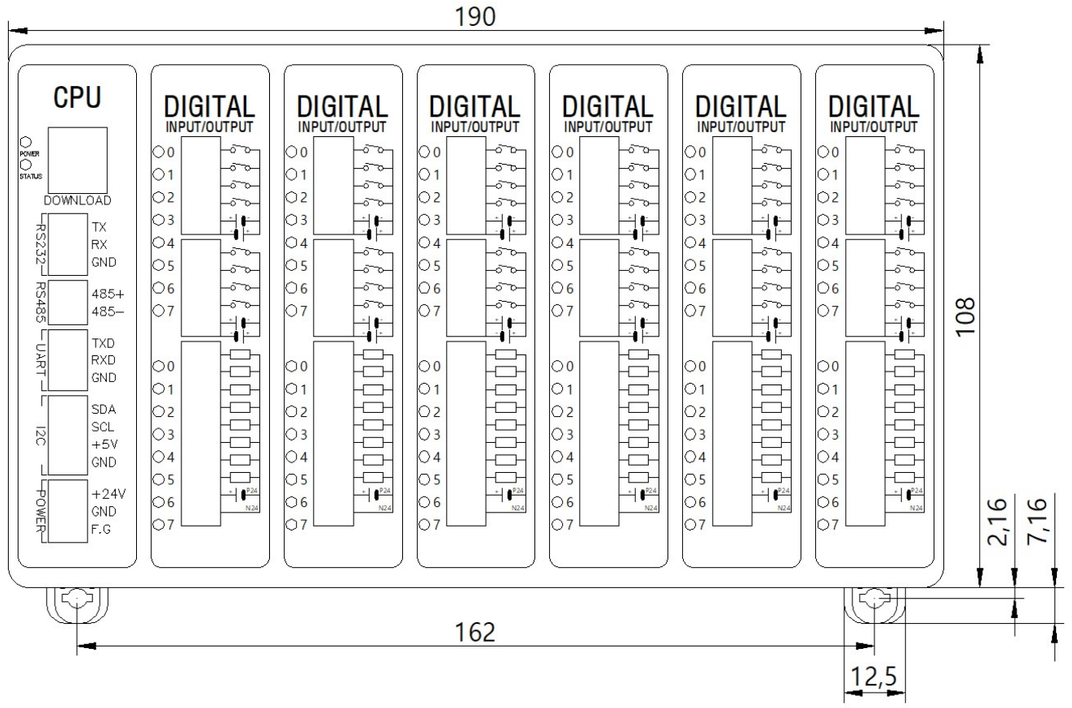
❒ 감사드립니다.

✎ 저희 (주)아이로직스의 제품을 구매해주셔서 감사드립니다.
✎ 구매는 https://www.ilogics.co.kr 쇼핑몰에서 하실 수 있습니다.
✎ 구매/기술 상담은 0507-1362-5020으로 전화 주시기 바랍니다.
(상담시간은 오전10시~오후5시 입니다)

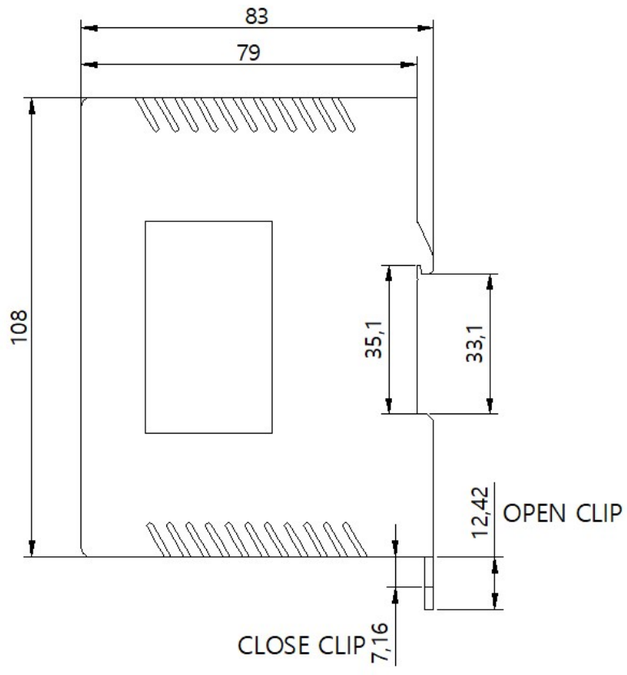
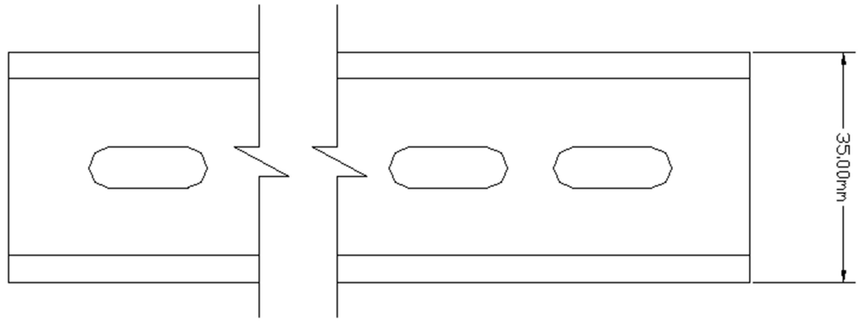
❒ DIMENSION (클립 열었을 경우)



❒ DIMENSION (클립 닫았을 경우)



❒ DIMENSION (클립 열었을 때 / 딘 레일 체결 후)



❒ DIMENSION (딘 레일 : 35mm)