

# SCHNEIDER Electric Industries

## SCHNEIDER MODBUS Master Series

### MODBUS-TCP Client (Master) Driver

지원 버전

TOP Design Studio

V1.0 이상



## CONTENTS

Touch Operation Panel을 사용해주시는 고객님께 감사 드립니다

### 1. 시스템 구성 [2 페이지](#)

접속에 필요한 기기, 각 기기의 설정, 케이블, 구성 가능한 시스템에 대해 설명합니다.

### 2. 외부 장치 선택 [3 페이지](#)

TOP 기종과 외부 장치를 선택합니다.

### 3. TOP 통신 설정 [4 페이지](#)

TOP 통신 설정 방법에 대해서 설명합니다.

### 4. 외부 장치 설정 [9 페이지](#)

외부 장치의 통신 설정 방법에 대해서 설명합니다.

### 5. 지원 어드레스 [10 페이지](#)

본 절을 참조하여 외부 장치와 통신 가능한 어드레스를 확인하십시오.

# 1. 시스템 구성

본 드라이버는 "MODBUS Organization"의 "MODBUS Protocol" 중 "MODBUS-TCP Client (Master)" 입니다.  
 외부 장치(MODBUS Slave Protocol 지원)에 따라서 드라이버의 "명령어 코드", "프로토콜 프레임 형식" 등을 별도 설정 해야 할 수 있습니다. 이 경우 통신 방식에 따른 세부 설정 사항을 외부 장치 측에 맞추어 설정 해주십시오.  
 본 드라이버가 지원하는 외부 장치와의 시스템 구성은 아래와 같습니다.

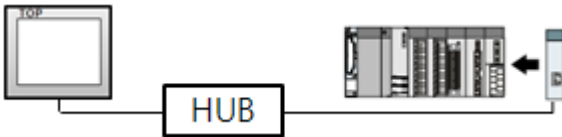
시리즈	CPU	Link I/F	통신 방식	시스템 설정	케이블
MODBUS Slave/Server Device			Ethernet (TCP / UDP)	<a href="#">3. TOP 통신 설정</a> <a href="#">4. 외부 장치 설정</a>	트위스트 페어 케이블*주1)

\*주1) 트위스트 페어 케이블

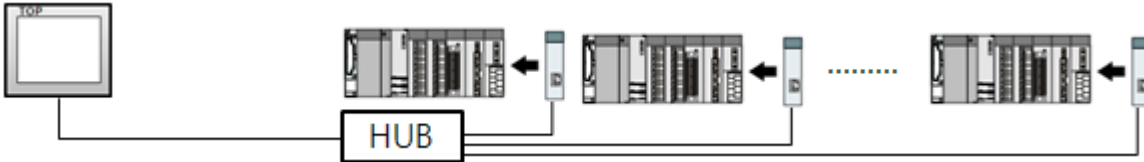
- STP(실드 트위스트 페어 케이블) 혹은 UTP(비실드 트위스트 페어 케이블) 카테고리 3, 4, 5 를 의미 합니다.
- 네트 워크 구성에 따라 허브, 트랜시버 등의 구성기기에 접속 가능하며 이 경우 다이렉트 케이블을 사용 하십시오.

## ■ 연결 가능 구성

• 1 : 1 연결(TOP 1 대와 외부 장치 1 대) 연결

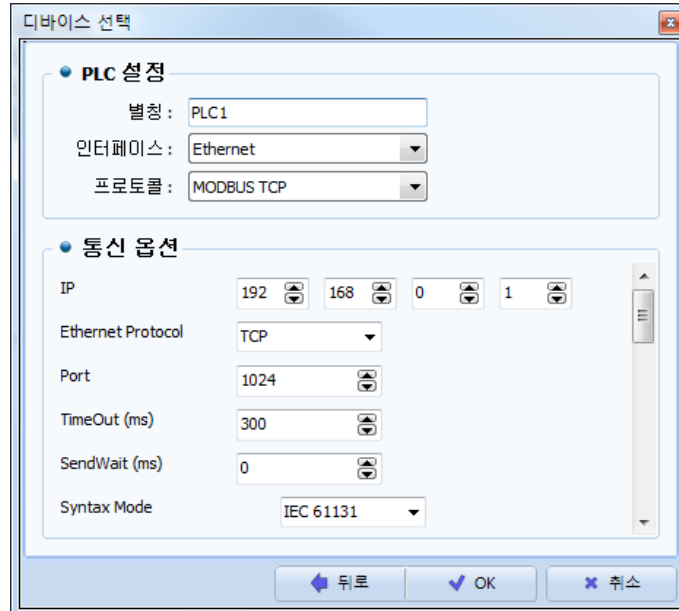
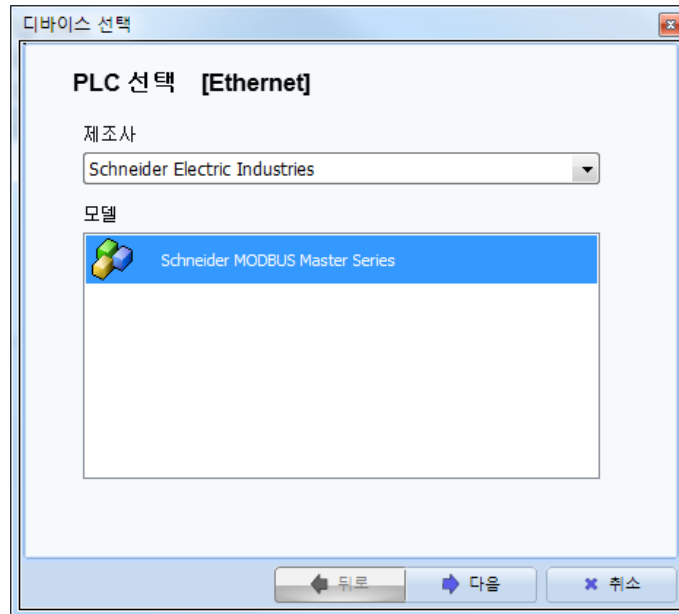


• 1 : N 연결(TOP 1 대와 외부 장치 여러 대) 연결



## 2. 외부 장치 선택

■ TOP 모델 및 포트 선택 후 외부 장치를 선택합니다.



설정 사항		내용					
TOP	모델	TOP 디스플레이와 프로세스를 확인하여 터치 모델을 선택합니다.					
외부 장치	제조사	TOP와 연결할 외부 장치의 제조사를 선택합니다. "Schneider Electric Industries"를 선택 하십시오.					
	PLC	TOP와 연결할 외부 장치를 선택합니다. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="background-color: black; color: white;">모델</th> <th style="background-color: black; color: white;">인터페이스</th> <th style="background-color: black; color: white;">프로토콜</th> </tr> </thead> <tbody> <tr> <td>Schneider MODBUS Master Series</td> <td>Ethernet</td> <td>MODBUS TCP</td> </tr> </tbody> </table> <p>연결을 원하는 외부 장치가 시스템 구성 가능한 기종인지 1장의 시스템 구성에서 확인 하시기 바랍니다.</p>	모델	인터페이스	프로토콜	Schneider MODBUS Master Series	Ethernet
모델	인터페이스	프로토콜					
Schneider MODBUS Master Series	Ethernet	MODBUS TCP					

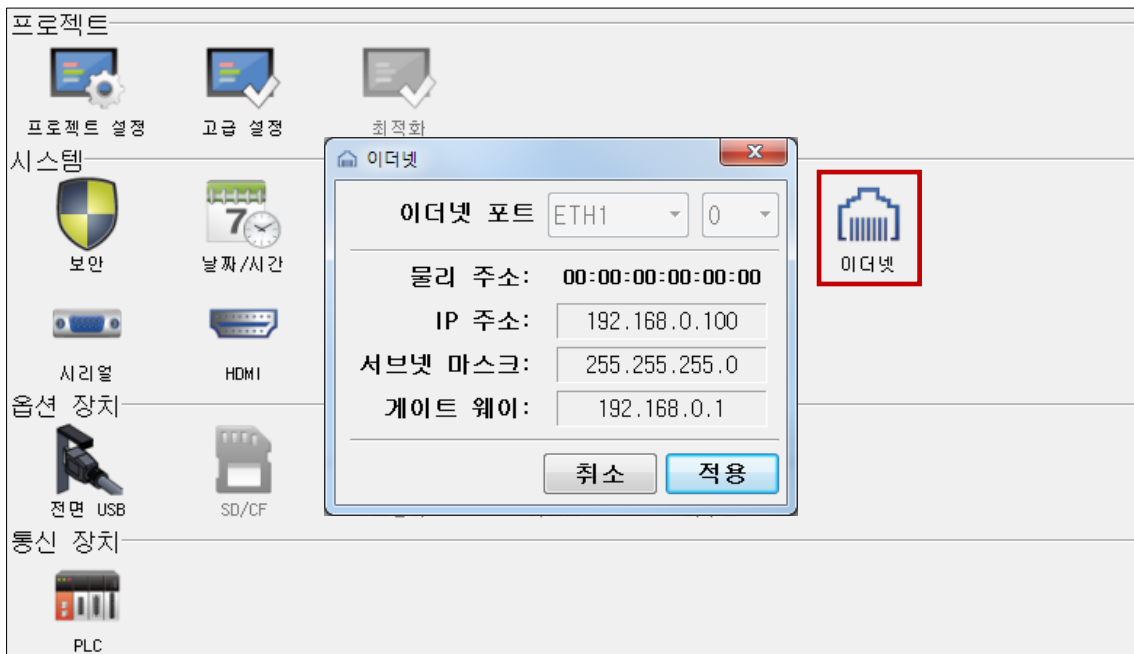
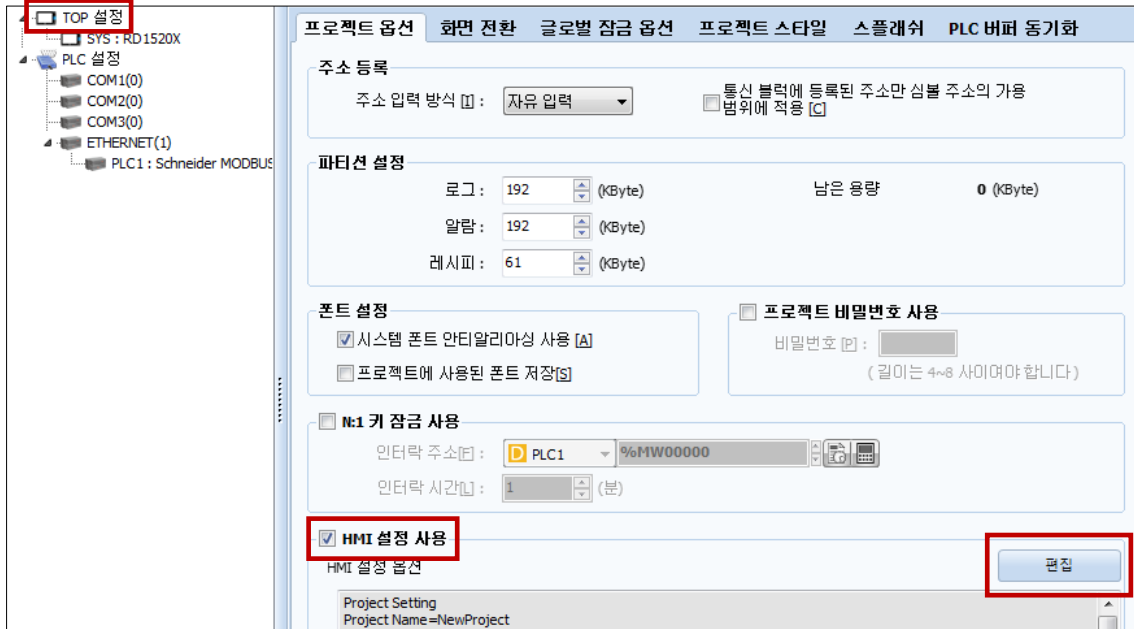
### 3. TOP 통신 설정

통신 설정은 TOP Design Studio 혹은 TOP 메인 메뉴에서 설정 가능 합니다. 통신 설정은 외부 장치와 동일하게 설정해야 합니다.

#### 3.1 TOP Design Studio 에서 통신 설정

##### (1) 통신 인터페이스 설정

- [ 프로젝트 > 프로젝트 속성 > TOP 설정 ] → [ 프로젝트 옵션 > “HMI 설정 사용” 체크 > 편집 > 이더넷 ]  
 – TOP 통신 인터페이스를 TOP Design Studio에서 설정합니다.



항 목	TOP	외부 장치	비 고
IP 주소*주1)주2)	192.168.0.100	192.168.0.51	
서브넷 마스크	255.255.255.0	255.255.255.0	
게이트 웨이	192.168.0.1	192.168.0.1	

\*주1) TOP와 외부 장치의 네트워크 주소 (IP 앞 세자리 192.168.0.0)는 일치해야 합니다.

\*주2) 동일 네트워크 상에서 중복된 IP 주소를 사용하지 마십시오.

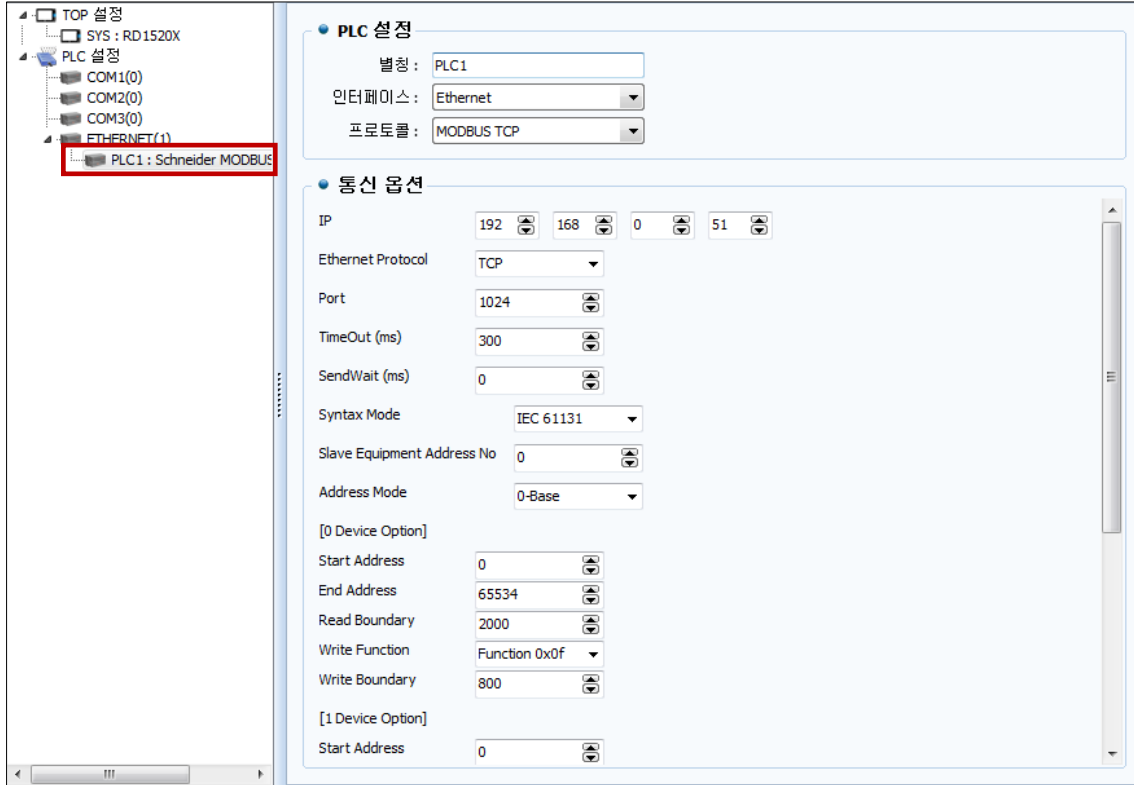
※ 위의 설정 내용은 본 사에서 권장하는 예제입니다.

항 목	설 명
IP 주소	네트워크 상에서 TOP가 사용 할 IP 주소를 설정합니다.
서브넷 마스크	네트워크의 서브넷 마스크를 입력합니다.
게이트 웨이	네트워크의 게이트 웨이를 입력합니다.



(2) 통신 옵션 설정

- [ 프로젝트 > 프로젝트 속성 > PLC 설정 > ETHERNET > "PLC1 : Schneider MODBUS Master Series" ]  
 - Schneider MODBUS-TCP Client (Master) 통신 드라이버의 옵션을 TOP Design Studio에서 설정합니다.



항 목	설 정	비 고
인터페이스	"Ethernet"을 선택합니다.	<a href="#">"2. 외부 장치 선택" 참고</a>
프로토콜	"MODBUS TCP"를 선택합니다.	
IP	외부 장치의 IP 주소를 입력합니다.	
Ethernet Protocol	TOP - 외부 장치 간 이더넷 프로토콜을 선택합니다.	
Port	외부 장치의 이더넷 통신 포트 번호를 입력합니다.	
TimeOut (ms)	TOP이 외부 장치로부터 응답을 기다리는 시간을 설정합니다.	
SendWait (ms)	TOP이 외부 장치로부터 응답 수신 후 다음 명령어 요청 전송 간에 대기 시간을 설정합니다.	
Syntax Mode	디바이스의 표기 방식을 설정합니다.	
Slave Equipment Address No	외부 장치(Slave)의 국번을 입력합니다.	
Address Mode	주소 방식을 선택합니다. (1-base : "address-1" 연산 / 0-base : 연산 없음)	
<b>[0 Device Option]</b>		
Start Address	Coil 의 시작 주소를 입력합니다.	
End Address	Coil 의 마지막 주소를 입력합니다.	
Read Boundary	Coil 에 대한 연속 읽기 개수를 설정합니다.	
Write Function	Coil 에 대한 쓰기 명령어를 설정 합니다. Force Single Coil : 05(Hex) / Force Multiple Coils : 0F(Hex)	
Write Boundary	Coil 에 대한 연속 쓰기 개수를 설정합니다.	
<b>[1 Device Option]</b>		
Start Address	Discrete Input 의 시작 주소를 입력합니다.	
End Address	Discrete Input 의 마지막 주소를 입력합니다.	
Read Boundary	Discrete Input 에 대한 연속 읽기 개수를 설정합니다.	
<b>[3 Device Option]</b>		
Start Address	Input Register 의 시작 주소를 입력합니다.	
End Address	Input Register 의 마지막 주소를 입력합니다.	
Read Boundary	Input Register 에 대한 연속 읽기 개수를 설정합니다.	
<b>[4 Device Option]</b>		
Start Address	Holding Register 의 시작 주소를 입력합니다.	
End Address	Holding Register 의 마지막 주소를 입력합니다.	
Read Boundary	Holding Register 에 대한 연속 읽기 개수를 설정합니다.	
Write Function	Holding Register 에 대한 쓰기 명령어를 설정합니다. Preset Single Register : 06(Hex) / Preset Multiple Registers : 10(Hex)	
Write Boundary	Holding Register 에 대한 연속 쓰기 개수를 설정합니다.	

### 3.2 TOP 에서 통신 설정

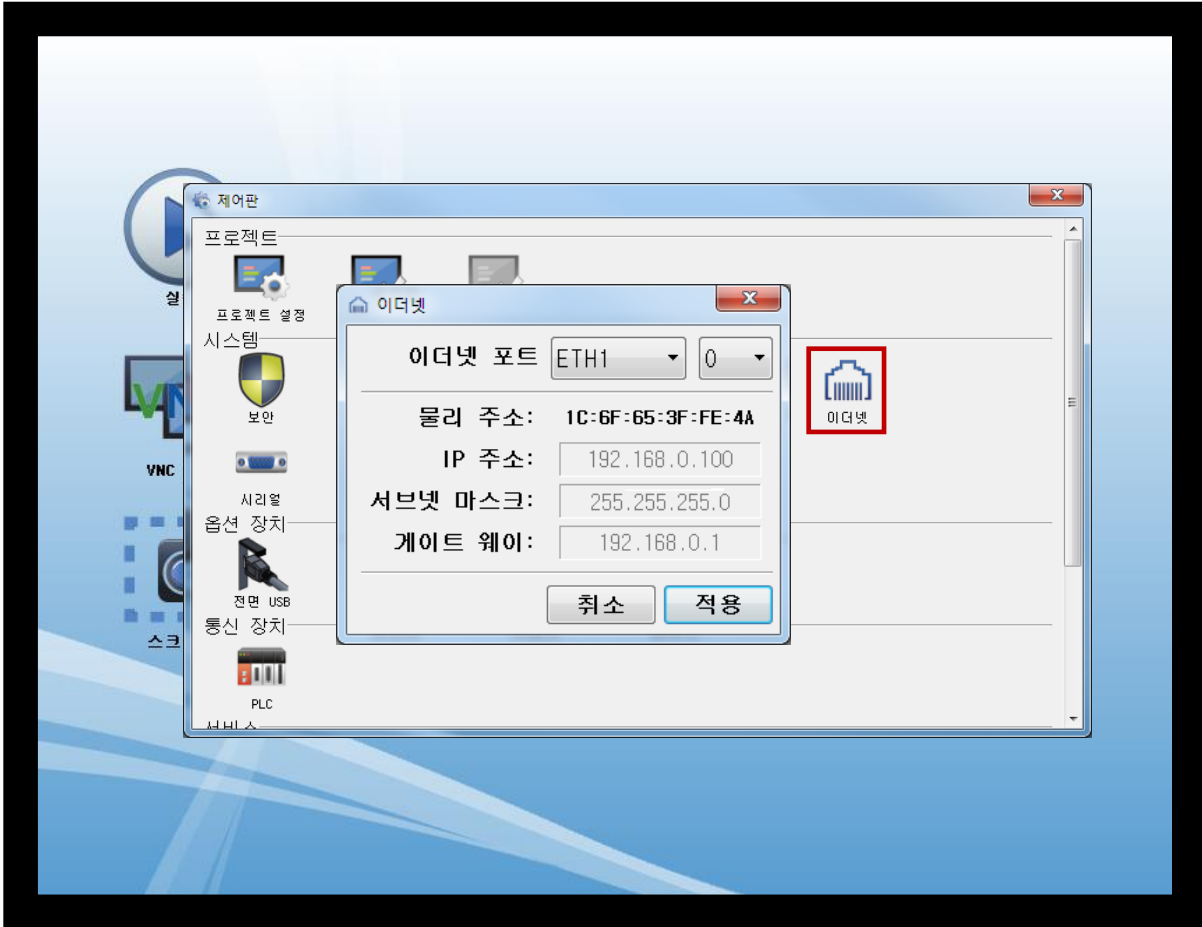
※ “3.1 TOP Design Studio 에서 통신 설정” 항목의 “HMI 설정 사용”을 체크 하지 않은 경우의 설정 방법입니다.

■ TOP 화면 상단을 터치하여 아래로 드래그 합니다. 팝업 창의 “EXIT”를 터치하여 메인 화면으로 이동합니다.



#### (1) 통신 인터페이스 설정

■ [ 메인 화면 > 제어판 > 이더넷 ]



항 목	TOP	외부 장치	비 고
IP 주소*주1)주2)	192.168.0.100	192.168.0.51	
서브넷 마스크	255.255.255.0	255.255.255.0	
게이트 웨이	192.168.0.1	192.168.0.1	

\*주1) TOP와 외부 장치의 네트워크 주소 (IP 앞 세자리 192 . 168 . 0 . 0)는 일치해야 합니다.

\*주2) 동일 네트워크 상에서 중복된 IP 주소를 사용하지 마십시오.

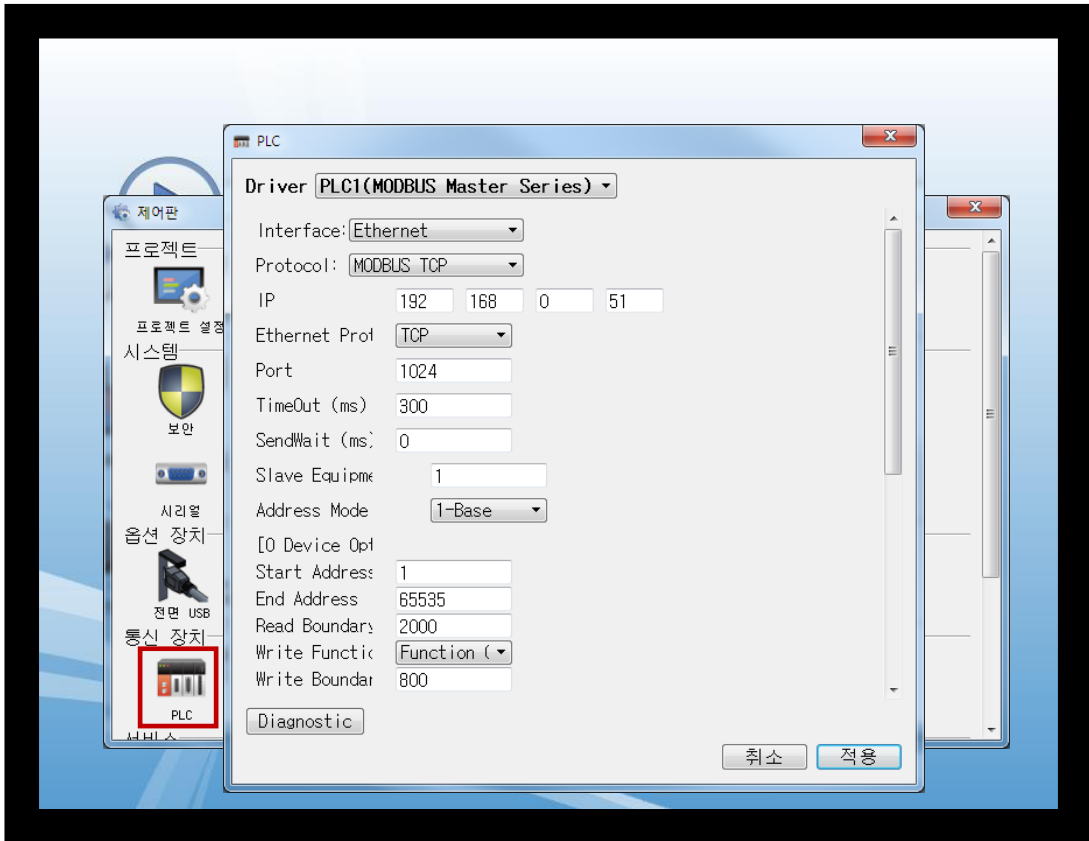
※ 위의 설정 내용은 본 사에서 권장하는 예제입니다.

항 목	설 명
IP 주소	네트워크 상에서 TOP가 사용 할 IP 주소를 설정합니다.
서브넷 마스크	네트워크의 서브넷 마스크를 입력합니다.
게이트 웨이	네트워크의 게이트 웨이를 입력합니다.



(2) 통신 옵션 설정

■ [ 메인 화면 > 제어판 > PLC ]



항 목	설 정	비 고
인터페이스	"Ethernet"을 선택합니다.	<a href="#">"2. 외부 장치 선택" 참고</a>
프로토콜	"MODBUS TCP"를 선택합니다.	
IP	외부 장치의 IP 주소를 입력합니다.	
Ethernet Protocol	TOP - 외부 장치 간 이더넷 프로토콜을 선택합니다.	
Port	외부 장치의 이더넷 통신 포트 번호를 입력합니다.	
TimeOut (ms)	TOP가 외부 장치로부터 응답을 기다리는 시간을 설정합니다.	
SendWait (ms)	TOP가 외부 장치로부터 응답 수신 후 다음 명령어 요청 전송 간에 대기 시간을 설정합니다.	
Syntax Mode	디바이스의 표기 방식을 설정합니다.	
Slave Equipment Address No	외부 장치(Slave)의 국번을 입력합니다.	
Address Mode	주소 방식을 선택합니다. (1-base : "address-1" 연산 / 0-base : 연산 없음)	
<b>[0 Device Option]</b>		
Start Address	Coil 의 시작 주소를 입력합니다.	
End Address	Coil 의 마지막 주소를 입력합니다.	
Read Boundary	Coil 에 대한 연속 읽기 개수를 설정합니다.	
Write Function	Coil 에 대한 쓰기 명령어를 설정 합니다. Force Single Coil : 05(Hex) / Force Multiple Coils : 0F(Hex)	
Write Boundary	Coil 에 대한 연속 쓰기 개수를 설정합니다.	
<b>[1 Device Option]</b>		
Start Address	Discrete Input 의 시작 주소를 입력합니다.	
End Address	Discrete Input 의 마지막 주소를 입력합니다.	
Read Boundary	Discrete Input 에 대한 연속 읽기 개수를 설정합니다.	
<b>[3 Device Option]</b>		
Start Address	Input Register 의 시작 주소를 입력합니다.	
End Address	Input Register 의 마지막 주소를 입력합니다.	
Read Boundary	Input Register 에 대한 연속 읽기 개수를 설정합니다.	
<b>[4 Device Option]</b>		
Start Address	Holding Register 의 시작 주소를 입력합니다.	
End Address	Holding Register 의 마지막 주소를 입력합니다.	
Read Boundary	Holding Register 에 대한 연속 읽기 개수를 설정합니다.	
Write Function	Holding Register 에 대한 쓰기 명령어를 설정합니다. Preset Single Register : 06(Hex) / Preset Multiple Registers : 10(Hex)	
Write Boundary	Holding Register 에 대한 연속 쓰기 개수를 설정합니다.	

### 3.3 통신 진단

■ TOP - 외부 장치 간 인터페이스 설정 상태를 확인

- TOP 화면 상단을 터치하여 아래로 드래그. 팝업 창의 "EXIT"를 터치하여 메인 화면으로 이동한다
- [ 제어판 > 이더넷 ] 에서 사용 하고자 하는 ETH 포트 설정이 외부 장치의 설정 내용과 같은지 확인한다

■ 포트 통신 이상 유무 진단

- [ 제어판 > PLC ] 에서 "통신 진단"을 터치한다.
- 화면 상에 Diagnostics 다이얼로그 박스가 팝업 되며 진단 상태를 판단한다.

OK	통신 설정 정상
Time Out Error	통신 설정 비정상 - 케이블 및 TOP, 외부 장치의 설정 상태 확인한다. (참조 : 통신 진단 시트 )

■ 통신 진단 시트

- 외부 단말기와 통신 연결에 문제가 있을 경우 아래 시트의 설정 내용을 확인 바랍니다.

항목	내용	확인		참 고	
시스템 구성	시스템 연결 방법	OK	NG	<a href="#">1. 시스템 구성</a>	
	접속 케이블 명칭	OK	NG		
TOP	버전 정보	OK	NG	<a href="#">2. 외부 장치 선택</a> <a href="#">3. 통신 설정</a>	
	사용 포트	OK	NG		
	드라이버 명칭	OK	NG		
	기타 세부 설정 사항	OK	NG		
	상대 국번	프로젝트 설정	OK		NG
		통신 진단	OK		NG
	이더넷 포트 설정	IP 주소	OK		NG
		서브넷 마스크	OK		NG
게이트 웨이		OK	NG		
외부 장치	CPU 명칭	OK	NG	<a href="#">4. 외부 장치 설정</a>	
	통신 포트 명칭(모듈 명)	OK	NG		
	프로토콜(모드)	OK	NG		
	설정 국번	OK	NG		
	기타 세부 설정 사항	OK	NG		
	이더넷 포트 설정	IP 주소	OK		NG
		서브넷 마스크	OK		NG
		게이트 웨이	OK		NG
어드레스 범위 확인	OK	NG	<a href="#">5. 지원 어드레스</a> (자세한 내용은 PLC 제조사의 매뉴얼을 참고 하시기 바랍니다.)		



## 4. 외부 장치 설정

---

외부 장치의 사용자 매뉴얼을 참조하여 외부기기 I/F에 “MODBUS-TCP Slave (Server) Driver”를 설정 하십시오.



- 동일 네트워크 상에서 IP 어드레스를 중복 사용하지 마십시오.
  - 외부 장치 측 어드레스 맵 내용을 확인하고 그 내용에 따라 통신 어드레스를 사용하십시오.
-

## 5. 지원 어드레스

TOP에서 사용 가능한 디바이스는 아래와 같습니다.

CPU 모듈 시리즈/타입에 따라 디바이스 범위(어드레스) 차이가 있을 수 있습니다. TOP 시리즈는 외부 장치 시리즈가 사용하는 최대 어드레스 범위를 지원합니다. 사용하고자 하는 장치가 지원하는 어드레스 범위를 벗어 나지 않도록 각 CPU 모듈 사용자 매뉴얼을 참조/주의 하십시오.

### ■ Syntax Mode : IEC 61131

	Bit Address	Word Address	32 bits	Remarks
Coil	%Q00001 - %Q65536	%Q00001 - %Q65521	L/H	
Discrete Input	%I00001 - %I65536	%I00001 - %I65521		*주1)
Input Register	%IW00001.00 - %IW65536.15	%IW00001 - %IW65536		*주1)
Holding Register	%MW00001.00 - %MW65536.15	%MW00001 - %MW65536		

\*주1) 쓰기 불가능(읽기 전용)

### ■ Syntax Mode : MODBUS

	Bit Address	Word Address	32 bits	Remarks
Coil	000001 - 065536	000001 - 065521	L/H	
Discrete Input	100001 - 165536	100001 - 165521		*주1)
Input Register	300001.00 - 365536.15	300001 - 365536		*주1)
Holding Register	400001.00 - 465536.15	400001 - 465536		

\*주1) 쓰기 불가능(읽기 전용)

## Appendix A. MODBUS TCP/IP ADU Frame(Data Frame)

본 기기의 "MODBUS TCP Client(Master) Driver"가 지원하는 MODBUS 프로토콜 명령어 및 디바이스에 대해 설명 합니다.

### WHAT IS MODBUS?

The MODBUS protocol was developed in 1979 by Modicon, Incorporated, for industrial automation systems and Modicon programmable controllers. It has since become an industry standard method for the transfer of discrete/analog I/O information and register data between industrial control and monitoring devices. MODBUS is now a widely-accepted, open, public-domain protocol that requires a license, but does not require royalty payment to its owner.

MODBUS devices communicate using a master-slave (client-server) technique in which only one device (the Client(Master)) can initiate

transactions (called queries). The other devices (slaves/servers) respond by supplying the requested data to the master, or by taking the action requested in the query. A slave is any peripheral device (I/O transducer, valve, network drive, or other measuring device) which processes information and sends its output to the master using MODBUS. The Acromag I/O Modules form slave/server devices, while a typical master device is a host computer running appropriate application software. Other devices may function as both clients (masters) and servers (slaves).

Masters can address individual slaves, or can initiate a broadcast message to all slaves. Slaves return a response to all queries addressed to them individually, but do not respond to broadcast queries. Slaves do not initiate messages on their own, they only respond to queries from the master.

A master's query will consist of a slave address (or broadcast address), a function code defining the requested action, any required data, and an error checking field. A slave's response consists of fields confirming the action taken, any data to be returned, and an error checking field. Note that the query and response both include a device address, a function code, plus applicable data, and an error checking field. If no error occurs, the slave's response contains the data as requested. If an error occurs in the query received, or if the slave is unable to perform the action requested, the slave will return an exception message as its response (see MODBUS Exceptions). The error check field of the slave's message frame allows the master to confirm that the contents of the message are valid. Traditional MODBUS messages are transmitted serially and parity checking is also applied to each transmitted character in its data frame.

At this point, It's important to make the distinction that MODBUS itself is an application protocol, as it defines rules for organizing and interpreting data, but remains simply a messaging structure, independent of the underlying physical layer. As it happens to be easy to understand, freely available, and accessible to anyone, it is thus widely supported by many manufacturers.

다음 페이지에 계속 됩니다.

## WHAT IS MODBUS TCP/IP?

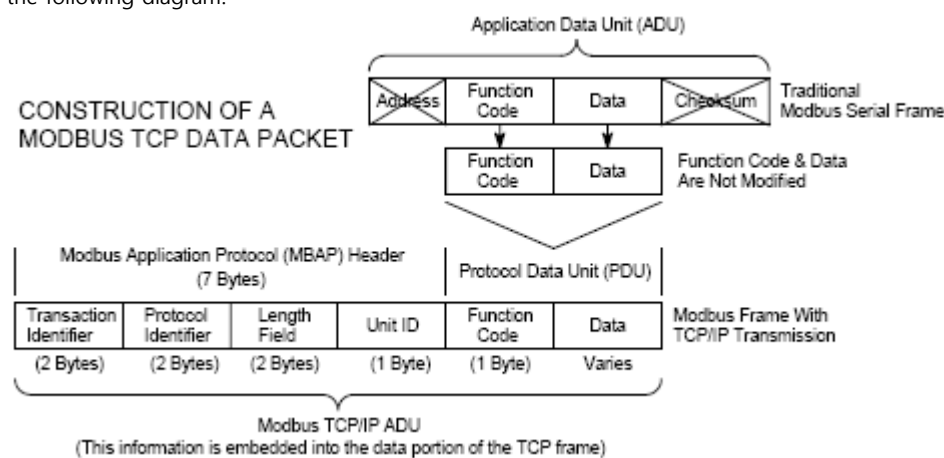
MODBUS TCP/IP (also MODBUS-TCP) is simply the MODBUS RTU protocol with a TCP interface that runs on Ethernet. The MODBUS messaging structure is the application protocol that defines the rules for organizing and interpreting the data independent of the data transmission medium.

TCP/IP refers to the Transmission Control Protocol and Internet Protocol, which provides the transmission medium for MODBUS TCP/IP messaging.

Simply stated, TCP/IP allows blocks of binary data to be exchanged between computers. It is also a world-wide standard that serves as the foundation for the World Wide Web. The primary function of TCP is to ensure that all packets of data are received correctly, while IP makes sure that messages are correctly addressed and routed. Note that the TCP/IP combination is merely a transport protocol, and does not define what the data means or how the data is to be interpreted (this is the job of the application protocol, MODBUS in this case).

So in summary, MODBUS TCP/IP uses TCP/IP and Ethernet to carry the data of the MODBUS message structure between compatible devices. That is, MODBUS TCP/IP combines a physical network (Ethernet), with a networking standard (TCP/IP), and a standard method of representing data (MODBUS as the application protocol). Essentially, the MODBUS TCP/IP message is simply a MODBUS communication encapsulated in an Ethernet TCP/IP wrapper.

In practice, MODBUS TCP embeds a standard MODBUS data frame into a TCP frame, without the MODBUS checksum, as shown in the following diagram.



The MODBUS commands and user data are themselves encapsulated into the data container of a TCP/IP telegram without being modified in any way. However, the MODBUS error checking field (checksum) is not used, as the standard Ethernet TCP/IP link layer checksum methods are instead used to guaranty data integrity. Further, the MODBUS frame address field is supplanted by the unit identifier in MODBUS TCP/IP, and becomes part of the MODBUS Application Protocol (MBAP) header (more on this later).

From the figure, we see that the function code and data fields are absorbed in their original form. Thus, a Modbus TCP/IP Application Data Unit (ADU) takes the form of a 7 byte header (transaction identifier + protocol identifier + length field + unit identifier), and the protocol data unit (function code + data). The MBAP header is 7 bytes long and includes the following fields:

- **Transaction/invocation Identifier (2 Bytes):** This identification field is used for transaction pairing when multiple messages are sent along the same TCP connection by a client without waiting for a prior response.
- **Protocol Identifier (2 bytes):** This field is always 0 for MODBUS services and other values are reserved for future extensions.
- **Length (2 bytes):** This field is a byte count of the remaining fields and includes the unit identifier byte, function code byte, and the data fields.
- **Unit Identifier (1 byte):** This field is used to identify a remote server located on a non TCP/IP network (for serial bridging). In a typical MODBUS TCP/IP server application, the unit ID is set to 00 or FF, ignored by the server, and simply echoed back in the response.

The complete MODBUS TCP/IP Application Data Unit is embedded into the data field of a standard TCP frame and sent via TCP to well-known system port 502, which is specifically reserved for MODBUS applications. MODBUS TCP/IP clients and servers listen and receive MODBUS data via port 502.

We can see that the operation of MODBUS over Ethernet is nearly transparent to the MODBUS register/command structure. Thus, if you are already familiar with the operation of traditional MODBUS, then you are already very with the operation of MODBUS TCP/IP.

## A.1 "0" Device (Coil)

### (1) Read Single Coil : 01

MASTER 기기에서 Slave 기기 측(국번:17번)의 "000020-000056 Coil" 데이터를 읽어 오는 예제를 통해 "01"명령어 프레임에 설명 합니다.

■ RTU Mode

( Master → Slave : 요청 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #표)	국번	장치-디바이스		디바이스주소	
	H	L	H	L	H	L			H	L	H	L
Hex	00	01	00	00	00	06	11	01	00	13	00	25

( Slave → Master : 응답 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #표)	국번	데이터 개수(byte)	데이터				
	H	L	H	L	H	L				Coils 27~20	Coils 35~28	Coils 43~36	Coils 51~44	Coils 56~52
Hex	00	01	00	00	00	08	11	01	05	CD	6B	B2	0E	1B

■ Coils 데이터 상태

Coils on/off	27	26	25	24	23	22	21	20
Coils on/off	1	1	0	0	1	1	0	1
Coils on/off	35	34	33	32	31	30	29	28
Coils on/off	0	1	1	0	1	0	1	1
Coils on/off	43	42	41	40	39	38	37	36
Coils on/off	1	0	1	1	0	0	1	0
Coils on/off	51	50	49	48	47	46	45	44
Coils on/off	0	0	0	0	1	1	1	0
Coils on/off	59	58	57	56	55	54	53	52
Coils on/off	-	-	-	1	1	0	1	1

0: OFF /

### (2) Force Single Coil : 05

MASTER 기기에서 Slave 기기 측의 Coil 000173 에 FORCE "ON" 하는 예제를 통해 "05"명령어 프레임에 설명 합니다.

■ RTU Mode

( Master → Slave : 요청 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #표)	국번	장치-디바이스		Force data	
	H	L	H	L	H	L			H	L	H	L
Hex	00	02	00	00	00	06	11	05	00	AC	FF	00

■ Force Data

	High	Low
Force ON	FF <sub>H</sub>	00 <sub>H</sub>
Force OFF	00 <sub>H</sub>	00 <sub>H</sub>

( Slave → Master : 응답 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #표)	국번	장치-디바이스		Force data	
	H	L	H	L	H	L			H	L	H	L
Hex	00	02	00	00	00	06	11	05	00	AC	FF	00

## A.2 "1" Device (Discrete Input)

### (1) Read Input Status : 02

MASTER 기기에서 Slave 기기 측(국번:17번)의 "100197~100218 Input" 데이터를 읽어 오는 예제를 통해 "02"명령어 프레임에 설명합니다.

( Master → Slave : 요청 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #)	명령어	선택디바이스		디바이스주소	
	H	L	H	L	H	L			H	L	H	L
Hex	00	03	00	00	00	06	11	02	00	C4	00	16

( Slave → Master : 응답 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #)	명령어	데이터 개 수(byte)	데이터(Inputs)
	H	L	H	L	H	L				
Hex	00	03	00	00	00	06	11	02	03	AC DB 35

■ Coils 데이터 상태

Coils	204	203	202	201	200	199	198	197
on/off	1	0	1	0	1	1	0	0
Coils	212	211	210	209	208	207	206	205
on/off	1	1	0	1	1	0	1	1
Coils	220	219	218	217	216	215	214	213
on/off	-	-	1	1	0	1	0	1

0: OFF / 1:ON

### A.3 "3" Device (Input Register)

#### (1) Read Input Registers : 04

MASTER 기기에서 Slave 기기 측(국번:17번)의 "30009 Register" 데이터를 읽어 오는 예제를 통해 "03"명령어 프레임을 설명 합니다.

( Master → Slave : 요청 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #표)	패킷주소	신뢰도값		디바이스편수 (Word Count)	
	H	L	H	L	H	L			H	L	H	L
Hex	00	04	00	00	00	06	11	04	00	08	00	01

( Slave → Master : 응답 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #표)	패킷주소	데이터 개 수(Byte)	데이터 Register 30009	
	H	L	H	L	H	L				H	L
Hex	00	04	00	00	00	05	11	04	02	00	0A

## A.4 "4" Device (Holding Register)

### (1) Read Holding Registers : 03

MASTER 기기에서 Slave 기기 측(국번:17)의 "400108 - 400110 Register" 데이터를 읽어 오는 예제를 통해 "03"명령어 프레임의 설명 합니다.

( Master → Slave : 요청 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #표)	패킷번호	전나디바이스		디바이스주소	
	H	L	H	L	H	L			H	L	H	L
Hex	00	05	00	00	00	06	11	03	00	6B	00	03

( Slave → Master : 응답 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #표)	패킷번호	패킷크기(byte)	데이터					
	H	L	H	L	H	L				Register 40108	Register 40109		Register 40110		
Hex	00	05	00	00	00	09	11	03	06	H	L	H	L	H	L
										02	2B	00	00	00	64

### (2) Preset Single Register : 06

Slave 기기 측의 400002 Register 에 00 03 (hex) 데이터를 입력 하는 예제를 통해 "06"명령어 프레임의 설명 합니다.

( Master → Slave : 요청 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #표)	패킷번호	전나디바이스		Preset data	
	H	L	H	L	H	L			H	L	H	L
Hex	00	06	00	00	00	06	11	06	00	01	00	03

( Slave → Master : 응답 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave #표)	패킷번호	전나디바이스		Preset data	
	H	L	H	L	H	L			H	L	H	L
Hex	00	06	00	00	00	06	11	06	00	01	00	03



**(3) Preset Multiple Register : 10**

Slave 기기 측의 400002 Register 에 "00 0A (hex)", "01 02 (hex)" 연속한 두 개의 데이터를 입력 하는 예제를 통해 "10"명령어 프레임임을 설명 합니다. (Error Code : 90<sub>H</sub>)

( Master → Slave : 요청 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave ID)	패킷주소	전나디값이시		Quantity of Register (Word Count)	데이터 개수(Byte)	데이터				
	H	L	H	L	H	L			H	L	H	L	Register 40003				
Hex	00	07	00	00	00	0B	11	10	00	01	00	02	04	Register 40002			
													H	L	H	L	
													00	0A	01	02	

( Slave → Master : 응답 프레임 )

Comment	Transaction Identifier		Protocol Identifier		Length Field		Unit ID (Slave ID)	패킷주소	전나디값이시		Quantity of Register (Word Count)	
	H	L	H	L	H	L			H	L	H	L
Hex	00	07	00	00	00	06	11	10	00	01	00	02