

MODBUS Organization

MODBUS Master Series

MODBUS-TCP Client (Master) Driver

지원 버전

TOP Design Studio

V1.4.5 이상



CONTENTS

Touch Operation Panel을 사용해주시는 고객님께 감사 드립니다.

- 1. 시스템 구성** [2 페이지](#)
연결 가능한 기기 및 네트워크 구성에 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** [12 페이지](#)
외부 장치의 통신 설정 방법에 대해서 설명합니다.
- 5. 지원 어드레스** [13 페이지](#)
본 절을 참고하여 외부 장치와 통신 가능한 데이터 주소를 확인 하십시오.

1. 시스템 구성

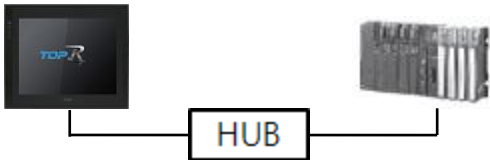
본 드라이버는 "MODBUS Organization"의 "MODBUS Protocol" 중 "MODBUS-TCP Client (Master)" 입니다.
 외부 장치(MODBUS Slave Protocol 지원)에 따라서 드라이버의 "명령어 코드", "프로토콜 프레임 형식" 등을 별도 설정 해야 할 수 있습니다. 이 경우 통신 방식에 따른 세부 설정 사항을 외부 장치 측에 맞추어 설정 해주십시오.
 본 드라이버가 지원하는 외부 장치와의 시스템 구성은 아래와 같습니다.

| 시리즈 | CPU | Link I/F | 통신 방식 | 시스템 설정 | 케이블 |
|-----|-----|----------|-------------------------|---|-----------------|
| | | | Ethernet (TCP / UDP) | 3. TOP 통신 설정 4. 외부 장치 설정 | 트위스트 페어 케이블*주1) |

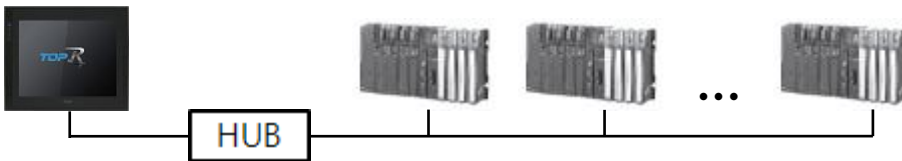
- *주1) 트위스트 페어 케이블
- STP(실드 트위스트 페어 케이블) 혹은 UTP(비실드 트위스트 페어 케이블) 카테고리 3, 4, 5 를 의미 합니다.
 - 네트 워크 구성에 따라 허브, 트랜시버 등의 구성기기에 접속 가능하며 이 경우 다이렉트 케이블을 사용 하십시오.

■ 연결 가능 구성

- 1 : 1 연결

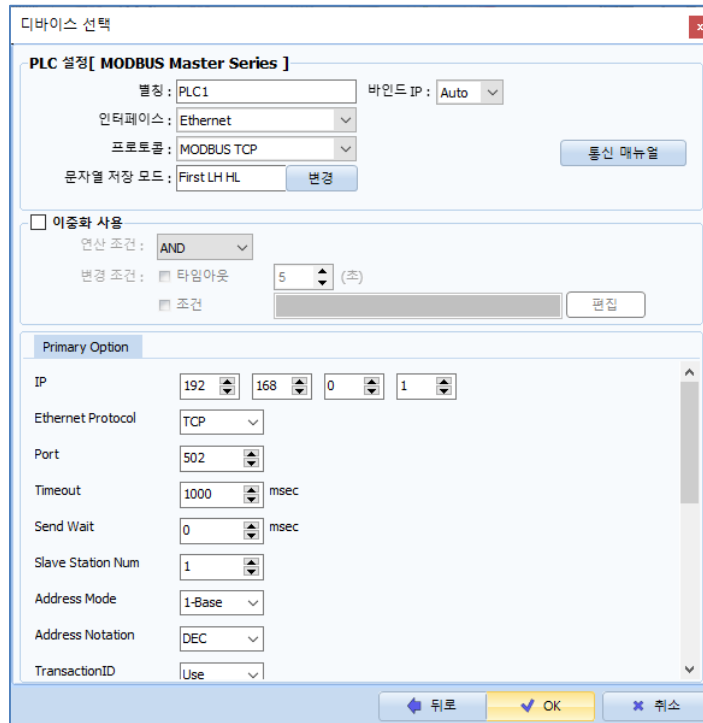
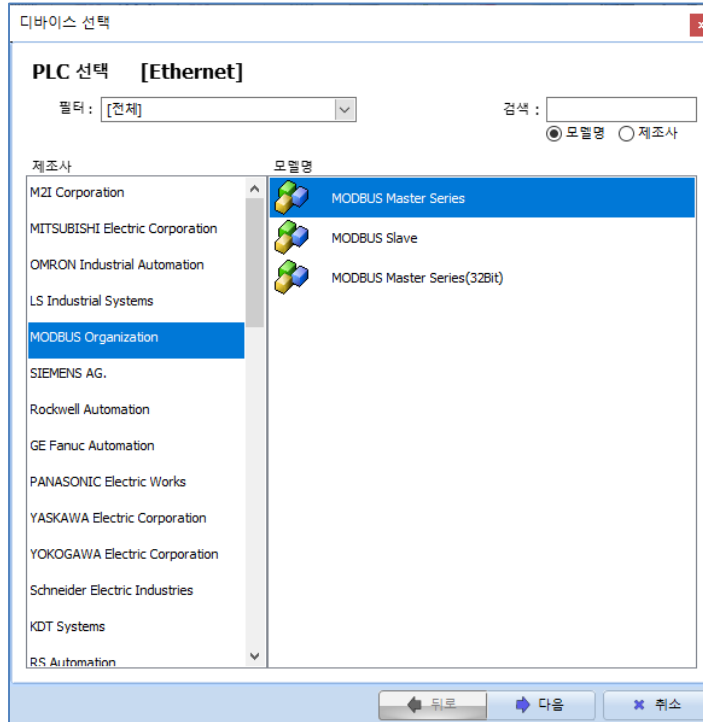


- 1 : N 연결



2. 외부 장치 선택

■ TOP 모델 및 포트 선택 후 외부 장치를 선택합니다.



| 설정 사항 | | 내용 | | | | | |
|----------------------|----------|---|----|-------|------|----------------------|----------|
| TOP | 모델 | TOP 모델을 선택합니다. | | | | | |
| 외부 장치 | 제조사 | TOP와 연결할 외부 장치의 제조사를 선택합니다. "MODBUS Organization"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>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 | MODBUS Master Series | Ethernet |
| 모델 | 인터페이스 | 프로토콜 | | | | | |
| MODBUS Master Series | Ethernet | MODBUS TCP | | | | | |

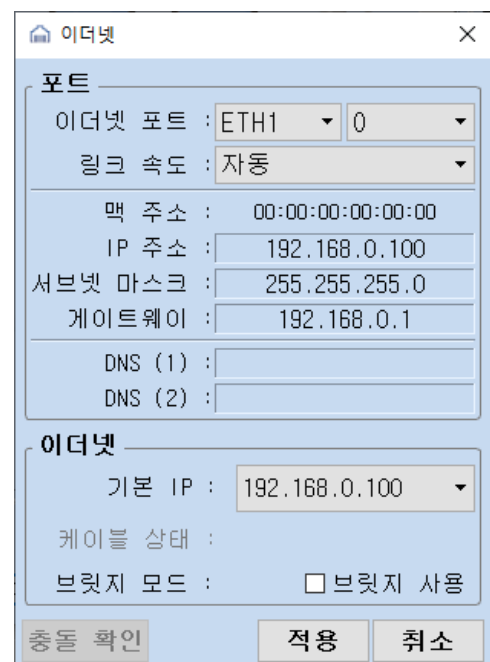
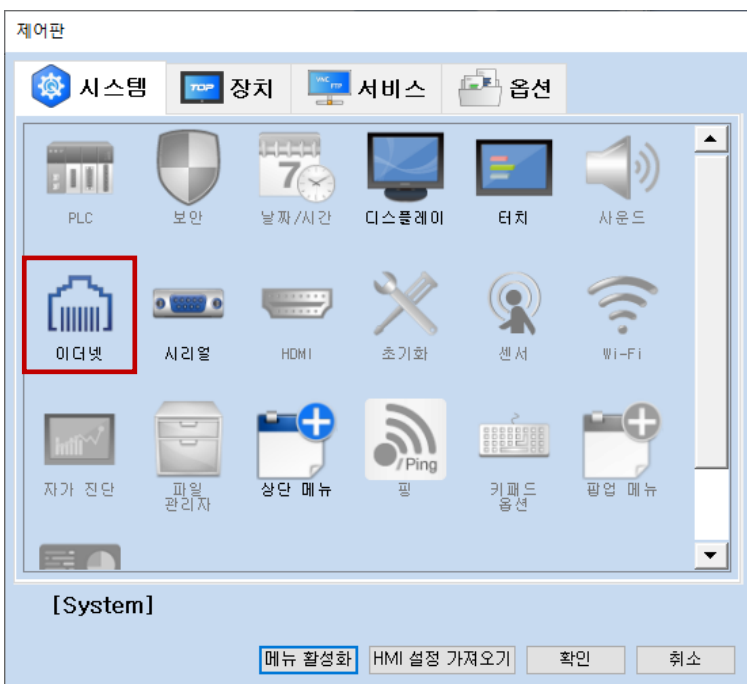
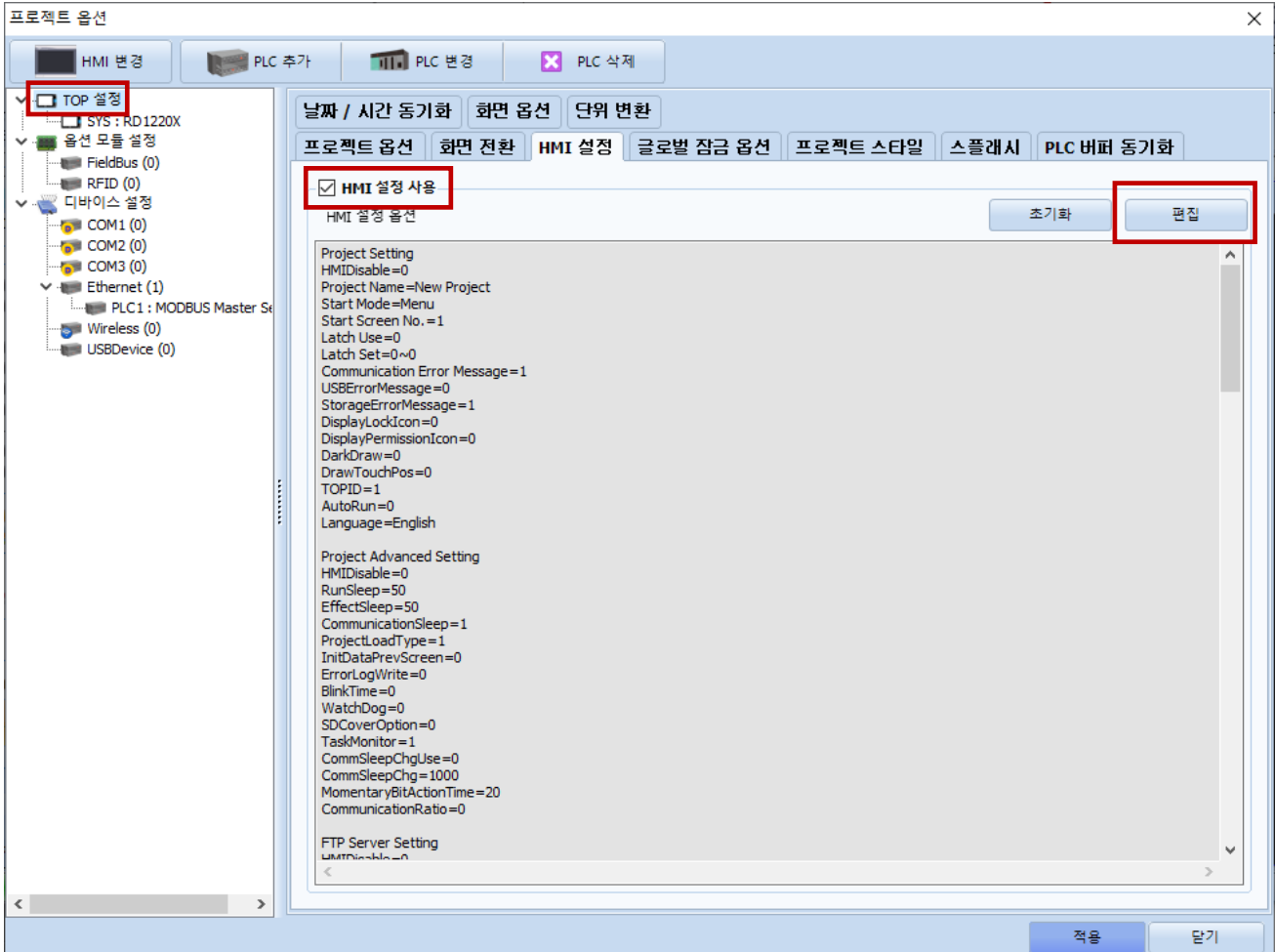
3. TOP 통신 설정

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] → [속성] → [TOP 설정] → [HMI 설정] → [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.

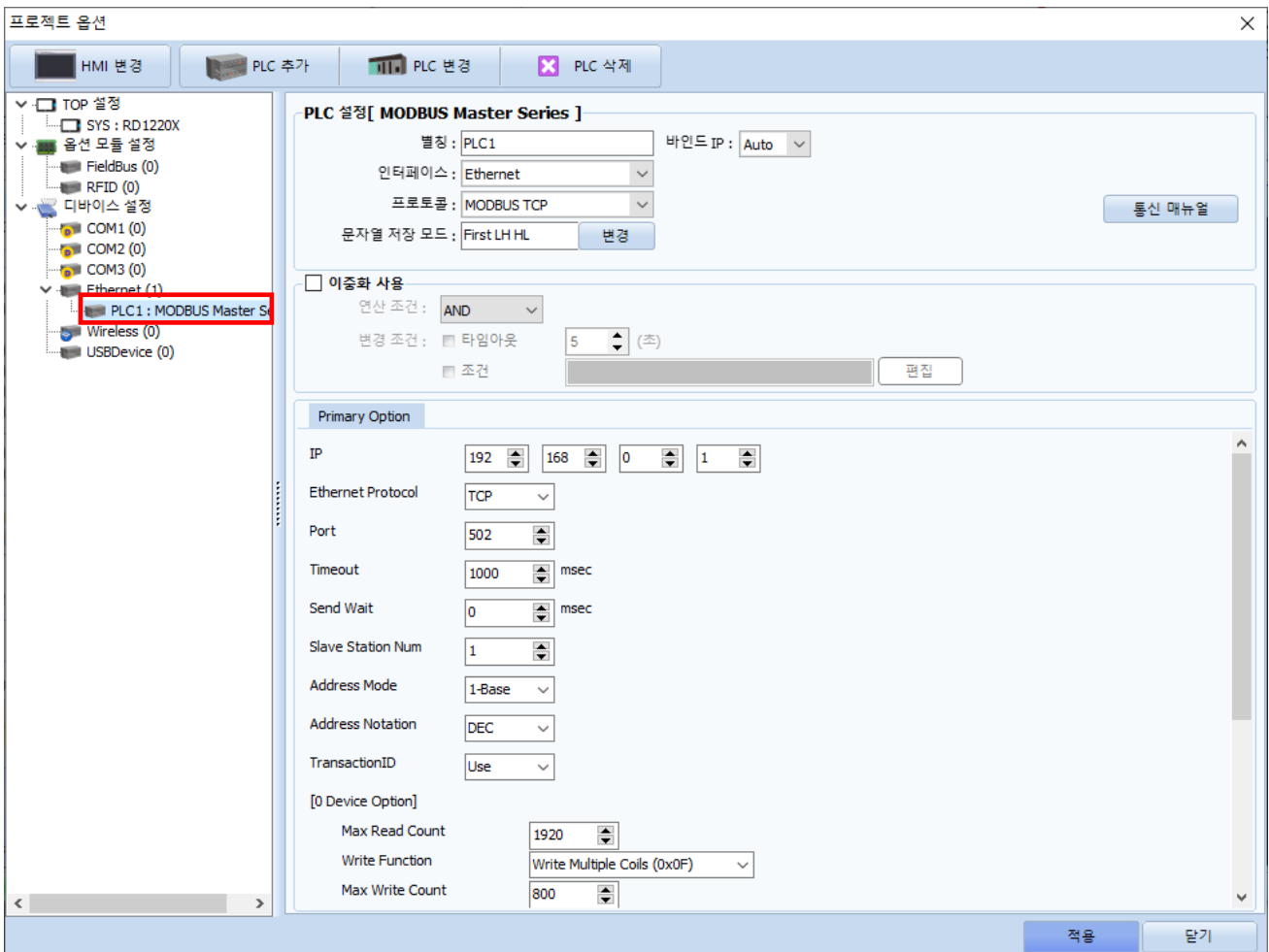
※ 위의 설정 내용은 본 사에서 권장하는 예제입니다.

| 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 : MODBUS Master Series]

- MODBUS-TCP Client (Master) 통신 드라이버의 옵션을 TOP Design Studio에서 설정합니다.



| 항 목 | 설 정 | 비 고 | |
|--------------------------|--|---|--|
| 인터페이스 | "Ethernet"을 선택합니다. | "2. 외부 장치 선택" 참고 | |
| 프로토콜 | "MODBUS TCP"를 선택합니다. | | |
| 문자열 저장모드 | 문자열 저장 방식을 선택합니다. | | |
| 이중화 | 이중화 사용 | 이중화 설정 사용 유무를 체크합니다. | |
| | 연산 조건 | 변경 조건에 대한 연산 조건을 설정합니다. AND : 체크된 변경 조건이 모두 만족하면 Primary ↔ Secondary 변경 OR : 체크된 변경 조건 중 하나라도 만족하면 Primary ↔ Secondary 변경 | |
| | 변경 조건 | Primary ↔ Secondary 변경 조건을 설정합니다. | |
| IP | 외부 장치의 IP 주소를 입력 합니다. | | |
| 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. | | |
| Slave Station Num | 외부 장치의 국번을 입력합니다. | | |
| Address Mode | 주소 입력 방식을 선택합니다. "1-base"로 설정 할 경우 통신 할 때 작화에 입력된 주소에 1을 뺀 값으로 요청하게 됩니다. 예) 40001 등록 시 Holding Register의 0000(hex)을 요청 | | |
| Address Notation | 주소 표기 방식을 선택합니다. | | |
| Transaction ID | 모드버스 TCP 패킷의 Transaction ID 적용 유무를 설정합니다. 미사용 시 0으로 고정 | | |
| [0 Device Option] | Coil | | |
| Max Read Count | Coil 읽기 요청 시 1회에 요청할 수 있는 최대 개수를 설정합니다. | *주1) *주2) | |
| Write Function | Coil 쓰기 요청 명령어를 설정합니다. 0x05 : Force Single Coil (1 비트 단위로 쓰기. 비트 단위 동작만 사용 가능) 0x0F : Force Multiple Coils (16 비트 단위로 쓰기) Auto : 데이터 수에 따라 0x05 또는 0x0F로 요청 | *주3) | |
| Max Write Count | Coil 쓰기 요청 시 1회에 요청 할 수 있는 최대 개수를 설정합니다. | *주2) | |
| Read Bit Unit | Coil 읽기 요청 시 요청하는 비트 수를 설정합니다. 설정 값이 16이고 화면에 이어지는 주소가 등록된 경우 1회에 최대 "Max Read Count" 수만큼 데이터를 요청합니다. | | |
| [1 Device Option] | Discrete Input | | |
| Max Read Count | Discrete Input 읽기 요청 시 1회에 요청할 수 있는 최대 개수를 설정합니다. | *주1) *주2) | |
| Read Bit Unit | Discrete Input 읽기 요청 시 요청하는 비트 수를 설정합니다. 설정 값이 16이고 화면에 이어지는 주소가 등록된 경우 1회에 최대 "Max Read Count" 수만큼 데이터를 요청합니다. | | |
| [3 Device Option] | Input Register | | |
| Max Read Count | Input Register 읽기 요청 시 1회에 요청할 수 있는 최대 개수를 설정합니다. | *주1) *주2) | |
| [4 Device Option] | Holding Register | | |
| Max Read Count | Holding Register 읽기 요청 시 1회에 요청할 수 있는 최대 개수를 설정합니다. | *주1) | |
| Write Function | Holding Register 쓰기 요청 명령어를 설정합니다. 0x06 : Preset Single Register (1개 쓰기) 0x10 : Preset Multiple Registers (n개 쓰기) Auto : 데이터 수에 따라 0x06 또는 0x10으로 요청 | *주3) | |
| Max Write Count | 명령어 0x10으로 Holding Register 데이터 쓰기 요청 시 1회에 요청할 수 있는 최대 개수를 설정합니다. | *주2) | |

***주 1)**

- 각 디바이스의 Max Read Count 는 화면에 등록된 주소들이 연속되지 않을 때 통신을 여러 번 하지 않고 1 회에 요청 할 주소 범위로도 사용됩니다.

예 1) 화면에 숫자 오브젝트로 400001, 400002, 400003, 400004, 400005, 400120 을 등록하고 4 디바이스의 Max Read Count 를 120 으로 설정 할 경우 400001 부터 400120 까지 연속된 주소로 가정하여 400001 부터 120 워드로 하여 한 번의 요청으로 데이터를 읽어옵니다.

예 2) 화면에 숫자 오브젝트로 400001, 400002, 400003, 400004, 400005, 400120 을 등록하고 4 디바이스의 Max Read Count 를 3 으로 설정 할 경우 400001 부터 400003 까지 3 워드, 400004 부터 400005 까지 2 워드, 400120 1 워드로 하여 세 번의 요청으로 데이터를 읽어옵니다.

예 3) 화면에 숫자 오브젝트로 400001, 400010, 400011, 400021, 400031, 400041 을 등록하고 4 디바이스의 Max Read Count 를 10 으로 설정 할 경우 400001 부터 400010 까지 10 워드, 400011 1 워드, 400021 1 워드 400031 1 워드, 400041 1 워드로 하여 다섯 번의 요청으로 데이터를 읽어옵니다.

- Max Read Count 를 0 으로 설정 할 경우 연속된 주소에 한해서 최대 120 워드만큼 요청합니다.

***주 2)**

- 외부 장치의 매뉴얼을 참고하여 등록된 주소부터 한번에 몇 개의 데이터가 읽기/쓰기가 가능한지 확인 하십시오.

외부 장치가 지원하는 범위보다 크게 설정 할 경우 통신이 정상적으로 이루어지지 않습니다.

예) 외부 장치의 Holding Register(4 디바이스)가 1 회 통신에 최대 10 개 워드만 응답 가능한 경우 TOP 의 통신 설정 중 4 디바이스의 Max Read Count 를 외부 장치의 사양에 맞게 10 으로 설정하십시오.

***주 3)**

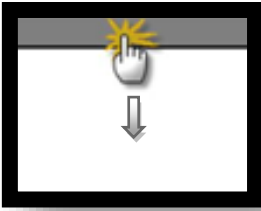
- 외부 장치의 매뉴얼을 참고하여 지원하는 쓰기 명령어에 맞게 설정하십시오.

지원하지 않는 쓰기 명령어 설정 시 데이터 쓰기 동작이 이루어지지 않습니다.

3.2 TOP 에서 통신 설정

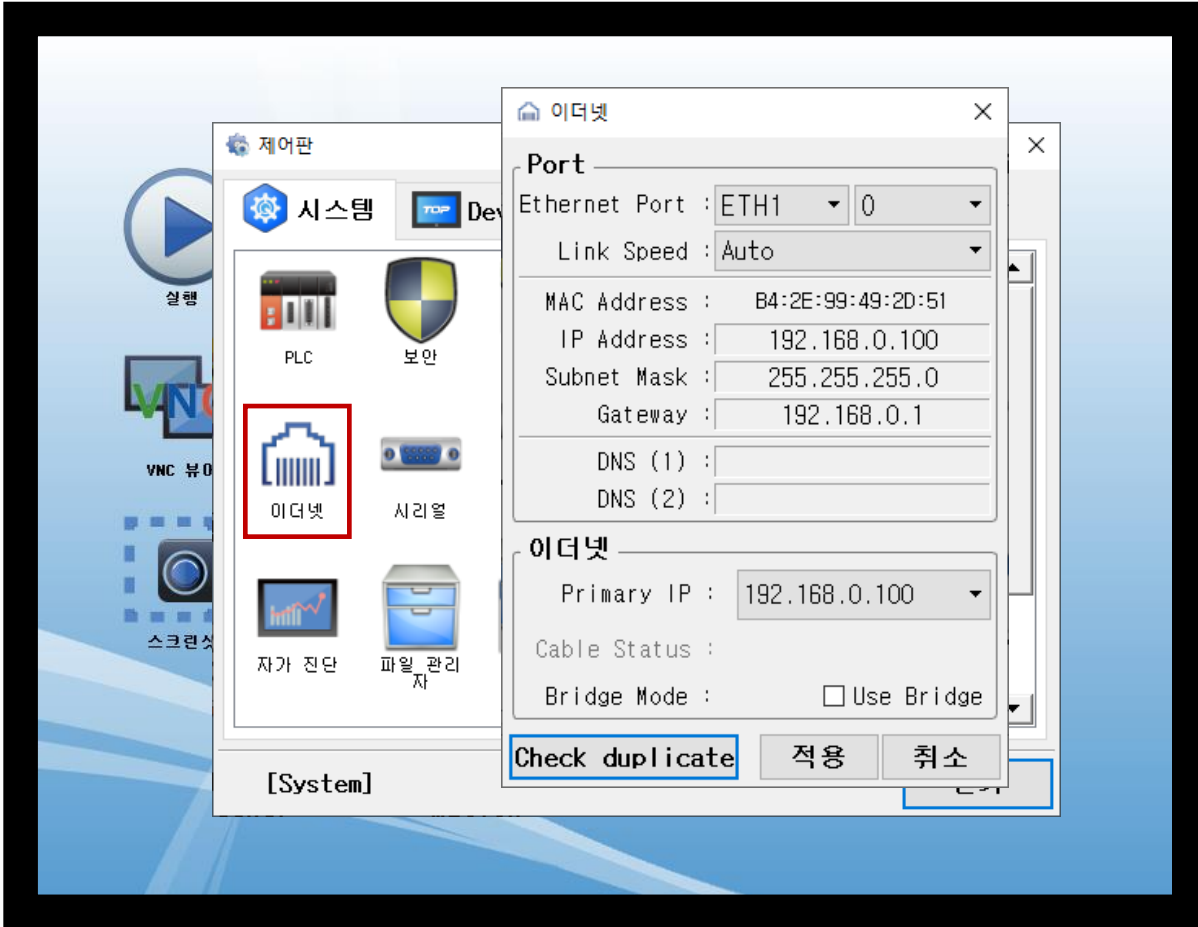
※ “3.1 TOP Design Studio 에서 통신 설정” 항목의 “HMI 설정 사용”을 체크 하지 않은 경우의 설정 방법입니다.

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



(1) 통신 인터페이스 설정

■ [제어판] → [이더넷]



| 항 목 | TOP | 외부 장치 | 비 고 |
|---------|---------------|---------------|----------|
| IP 주소 | 192.168.0.100 | 192.168.0.51 | *주1) 주2) |
| 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.

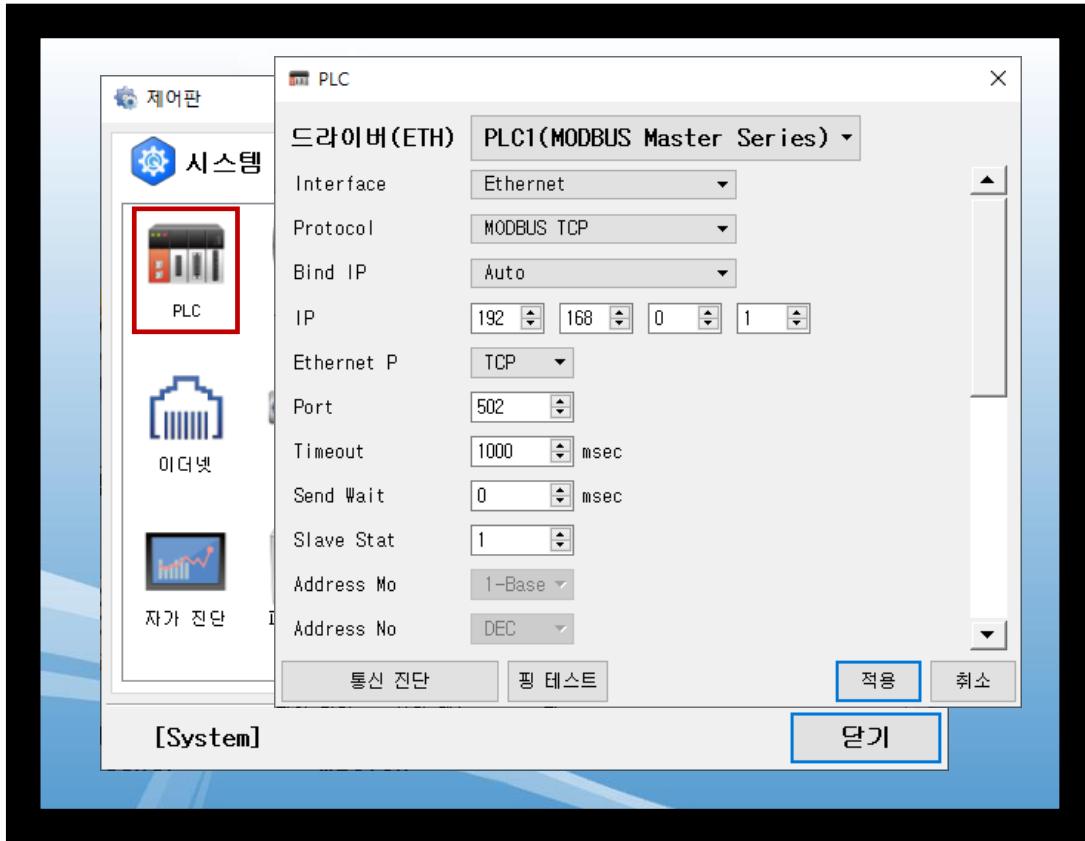
※ 위의 설정 내용은 본 사에서 권장하는 예제입니다.

| 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

■ [제어판] → [PLC]



| 항 목 | 설 정 | 비 고 |
|--------------------------|--|----------------------------------|
| 인터페이스 | "Ethernet"을 선택합니다. | "2. 외부 장치 선택" 참고 |
| 프로토콜 | "MODBUS TCP"를 선택합니다. | |
| IP | 외부 장치의 IP 주소를 입력 합니다. | |
| 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. | |
| Slave Station Num | 외부 장치의 국번을 입력합니다. | |
| Address Mode | 주소 입력 방식을 선택합니다. 1-base : 장비의 메모리 주소가 1부터 시작. 등록된 주소-1로 데이터 요청. 0-base : 장비의 메모리 주소가 0부터 시작. 등록된 주소로 데이터 요청 | |
| Address Notation | 주소 표기 방식을 선택합니다. | |
| Transaction ID | 모드버스 TCP 패킷의 Transaction ID 적용 유무를 설정합니다. 미사용 시 0으로 고정 | |
| [0 Device Option] | Coil | |
| Max Read Count | Coil 읽기 요청 시 1회에 요청할 수 있는 최대 개수를 설정합니다. | *주1) *주2) |
| Write Function | Coil 쓰기 요청 명령어를 설정합니다. 0x05 : Force Single Coil (1 비트 단위로 쓰기. 비트 단위 동작만 사용 가능) 0x0F : Force Multiple Coils (16 비트 단위로 쓰기) Auto : 데이터 수에 따라 0x05 또는 0x0F로 요청 | *주3) |
| Max Write Count | Coil 쓰기 요청 시 1회에 요청 할 수 있는 최대 개수를 설정합니다. | *주2) |
| Read Bit Unit | Coil 읽기 요청 시 요청하는 비트 수를 설정합니다. 설정 값이 16이고 화면에 이어지는 주소가 등록된 경우 1회에 최대 "Max Read Count" 수만큼 데이터를 요청합니다. | |
| [1 Device Option] | Discrete Input | |
| Max Read Count | Discrete Input 읽기 요청 시 1회에 요청할 수 있는 최대 개수를 설정합니다. | *주1) *주2) |
| Read Bit Unit | Discrete Input 읽기 요청 시 요청하는 비트 수를 설정합니다. | |

| | | |
|--------------------------|---|-----------|
| | 설정 값이 16이고 화면에 이어지는 주소가 등록된 경우 1회에 최대 "Max Read Count" 수만큼 데이터를 요청합니다. | |
| [3 Device Option] | Input Register | |
| Max Read Count | Input Register 읽기 요청 시 1회에 요청할 수 있는 최대 개수를 설정합니다. | *주1) *주2) |
| [4 Device Option] | Holding Register | |
| Max Read Count | Holding Register 읽기 요청 시 1회에 요청할 수 있는 최대 개수를 설정합니다. | *주1) |
| Write Function | Holding Register 쓰기 요청 명령어를 설정합니다. 0x06 : Preset Single Register (1개 쓰기) 0x10 : Preset Multiple Registers (n개 쓰기) Auto : 데이터 수에 따라 0x06 또는 0x10으로 요청 | *주3) |
| Max Write Count | 명령어 0x10으로 Holding Register 데이터 쓰기 요청 시 1회에 요청할 수 있는 최대 개수를 설정합니다. | *주2) |

***주 1)**

- 각 디바이스의 Max Read Count 는 화면에 등록된 주소들이 연속되지 않을 때 통신을 여러 번 하지 않고 1 회에 요청 할 주소 범위로도 사용됩니다.

예 1) 화면에 숫자 오브젝트로 400001, 400002, 400003, 400004, 400005, 400120 을 등록하고 4 디바이스의 Max Read Count 를 120 으로 설정 할 경우 400001 부터 400120 까지 연속된 주소로 가정하여 400001 부터 120 워드로 하여 한 번의 요청으로 데이터를 읽어옵니다.

예 2) 화면에 숫자 오브젝트로 400001, 400002, 400003, 400004, 400005, 400120 을 등록하고 4 디바이스의 Max Read Count 를 3 으로 설정 할 경우 400001 부터 400003 까지 3 워드, 400004 부터 400005 까지 2 워드, 400120 1 워드로 하여 세 번의 요청으로 데이터를 읽어옵니다.

예 3) 화면에 숫자 오브젝트로 400001, 400010, 400011, 400021, 400031, 400041 을 등록하고 4 디바이스의 Max Read Count 를 10 으로 설정 할 경우 400001 부터 400010 까지 10 워드, 400011 1 워드, 400021 1 워드 400031 1 워드, 400041 1 워드로 하여 다섯 번의 요청으로 데이터를 읽어옵니다.

- Max Read Count 를 0 으로 설정 할 경우 연속된 주소에 한해서 최대 120 워드만큼 요청합니다.

***주 2)**

- 외부 장치의 매뉴얼을 참고하여 등록된 주소부터 한번에 몇 개의 데이터가 읽기/쓰기가 가능한지 확인 하십시오.
외부 장치가 지원하는 범위보다 크게 설정 할 경우 통신이 정상적으로 이루어지지 않습니다.

예) 외부 장치의 Holding Register(4 디바이스)가 1 회 통신에 최대 10 개 워드만 응답 가능한 경우 TOP 의 통신 설정 중 4 디바이스의 Max Read Count 를 외부 장치의 사양에 맞게 10 으로 설정하십시오.

***주 3)**

- 외부 장치의 매뉴얼을 참고하여 지원하는 쓰기 명령어에 맞게 설정하십시오.
지원하지 않는 쓰기 명령어 설정 시 데이터 쓰기 동작이 이루어지지 않습니다.

3.3 통신 진단

■ TOP - 외부 장치 간 인터페이스 설정 상태를 확인

- TOP 화면 상단을 터치하여 아래로 드래그. 팝업 창의 "EXIT"를 터치하여 시스템 화면으로 이동한다
- [제어판] → [이더넷] 에서 연결된 포트 설정이 올바르게 되어있는지 확인한다

■ 포트 통신 이상 유무 진단

- [제어판] → [PLC] 에서 "통신 진단"을 터치한다.
- 통신 연결 여부를 확인한다.

| | |
|-----------|---|
| 통신 진단, 성공 | 통신 설정 정상 |
| 에러 메시지 | 통신 설정 비정상 |
| | - 케이블 및 TOP, 외부 장치의 설정 상태 확인한다. (통신 진단 시트 참고) |

■ 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 | 1. 시스템 구성 | |
| | 케이블 | OK | NG | | |
| TOP | 버전 정보 | OK | NG | 2. 외부 장치 선택 3. TOP 통신 설정 | |
| | 통신 포트 | OK | NG | | |
| | 통신 드라이버 및 프로토콜 | 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 | 4. 외부 장치 설정 | |
| | 통신 포트 | OK | NG | | |
| | 프로토콜 | OK | NG | | |
| | 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 | 5. 지원 어드레스 (자세한 내용은 PLC 제조사의 매뉴얼을 참고 하시기 바랍니다.) | | |

4. 외부 장치 설정

제조사 사용자 매뉴얼을 참고하여 모드버스 슬레이브(서버)로 설정 하십시오.



- 외부 장치의 메모리 주소를 확인하십시오.

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.

| | 비트 | 워드 | 비고 |
|------------------|-----------------------|-----------------|------|
| Coil | 000001 – 065536 | 000001 – 065521 | |
| 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) 워기 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?

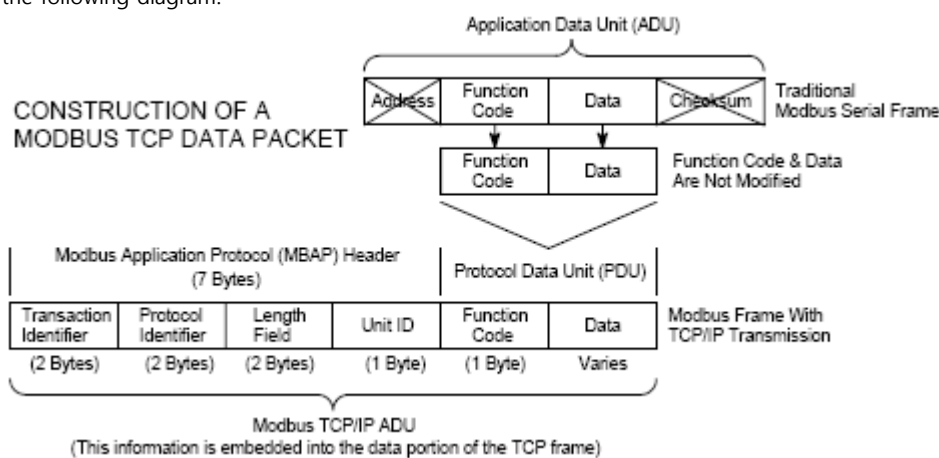
MODBUS TCP/IP (also MODBUS-TCP) is simply the MODBUS RTU protocol with a TCP interface that runs on Ethernet. The MODBUS messaging structure is the application protocol that defines the rules for organizing and interpreting the data independent of the data transmission medium.

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 프레임 설명 합니다.

■ RTU Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | |
|---------|------------------------|----|---------------------|----|--------------|----|--------------------|-----|---------|----|---------|----|
| Comment | Transaction Identifier | | Protocol Identifier | | Length Field | | Unit ID (Slave #코) | 코번호 | 장치-디바이스 | | 디바이스-주소 | |
| | 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 #코) | 코번호 | 데이터-길이(바이트) | | 데이터 | | | |
| | H | L | H | L | H | L | | | | | | | | |
| 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 프레임 설명 합니다.

■ RTU Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | |
|---------|------------------------|----|---------------------|----|--------------|----|--------------------|-----|---------|----|------------|----|
| Comment | Transaction Identifier | | Protocol Identifier | | Length Field | | Unit ID (Slave #코) | 코번호 | 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 _H | 00 _H |
| Force OFF | 00 _H | 00 _H |

(Slave → Master : 응답 프레임)

| | | | | | | | | | | | | |
|---------|------------------------|----|---------------------|----|--------------|----|--------------------|-----|---------|----|------------|----|
| Comment | Transaction Identifier | | Protocol Identifier | | Length Field | | Unit ID (Slave #코) | 코번호 | 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 프레임 설명합니다.

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | |
|---------|------------------------|----|---------------------|----|--------------|----|---------------------|------|---------|----|---------|----|
| Comment | Transaction Identifier | | Protocol Identifier | | Length Field | | Unit ID (Slave 17번) | 코일번호 | 진짜 디바이스 | | 디바이스 점수 | |
| | 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 17번) | 코일번호 | 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 #) | 패킷번호 | 전송 데이터 길이 | | 데이터 개수 (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 #) | 패킷번호 | 데이터 개수 (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 프레임에 설명 합니다.

(Master → Slave : 요청 프레임)

| Comment | Transaction Identifier | | Protocol Identifier | | Length Field | | Unit ID (Slave F#) | 주소 | 장치번호 | | 디바이스번호 | |
|---------|------------------------|----|---------------------|----|--------------|----|--------------------|----|------|----|--------|----|
| | 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 F#) | 주소 | 데이터 길이(byte) | 데이터 | | | | | | | |
|---------|------------------------|----|---------------------|----|--------------|----|--------------------|----|----------------|----------------|----|----------------|----|----|----|---|--|
| | H | L | H | L | H | L | | | Register 40108 | Register 40109 | | Register 40110 | | | | | |
| | H | L | H | L | H | L | | | H | L | H | L | H | L | H | L | |
| Hex | 00 | 05 | 00 | 00 | 00 | 09 | 11 | 03 | 06 | 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 프레임에 설명 합니다.

(Master → Slave : 요청 프레임)

| Comment | Transaction Identifier | | Protocol Identifier | | Length Field | | Unit ID (Slave F#) | 주소 | 장치번호 | | 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 F#) | 주소 | 장치번호 | | 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_H)

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | | | | | | |
|---------|------------------------|----|---------------------|----|--------------|----|--------------------|------|--------|----|-----------------------------------|--------------|----------------|----------------|----|----|----|
| Comment | Transaction Identifier | | Protocol Identifier | | Length Field | | Unit ID (Slave ID) | 패킷주소 | 전나디값이신 | | Quantity of Register (Word Count) | 패킷의 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 | 전나디값이신 | | 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 |