

SCHNEIDER Electric Industries

SCHNEIDER MODBUS Master Series

MODBUS Serial Master Driver

지원 버전

TOP Design Studio

V1.0 이상



CONTENTS

Touch Operation Panel을 사용해주는 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. 케이블 표** [10 페이지](#)

접속에 필요한 케이블 사양에 대해 설명합니다.
- 6. 지원 어드레스** [12 페이지](#)

본 절을 참조하여 외부 장치와 통신 가능한 어드레스를 확인하십시오.

1. 시스템 구성

본 드라이버는 "MODBUS Organization"의 "MODBUS Protocol" 중 "Serial Master Driver" 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.

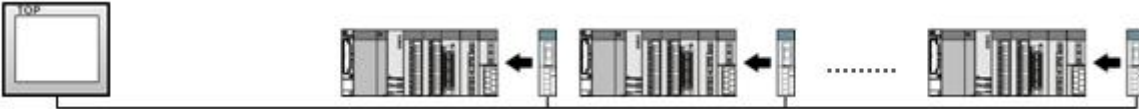
| 시리즈 | CPU | Link I/F | 통신 방식 | 시스템 설정 | 케이블 |
|---------------------|-----|----------|-----------------|-------------------------------------------------------------|--------------------------|
| MODBUS Slave Device | | | RS-232C | 3. TOP 통신 설정 4. 외부 장치 설정 | 5. 케이블 표 |
| | | | RS-422 (4 wire) | | |
| | | | RS-485 (2 wire) | | |

■ 연결 가능 구성

- 1 : 1(TOP 1 대와 외부 장치 1 대) 연결 - RS232C/422/485 통신에서 가능한 구성입니다.

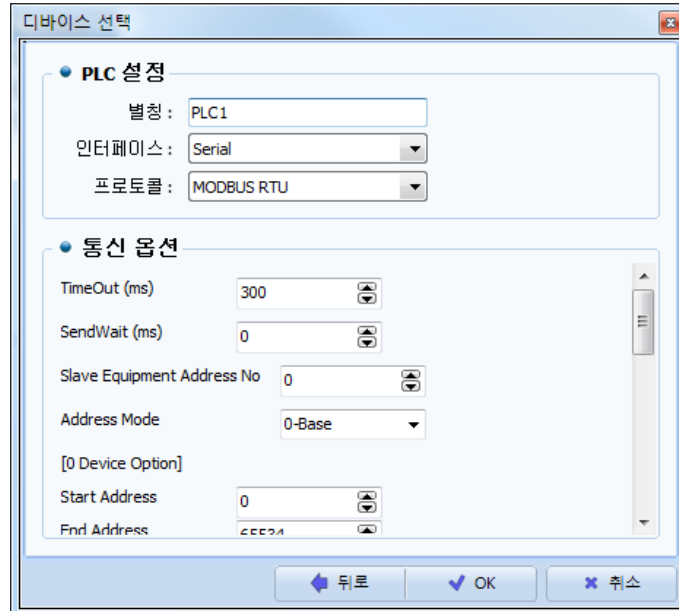
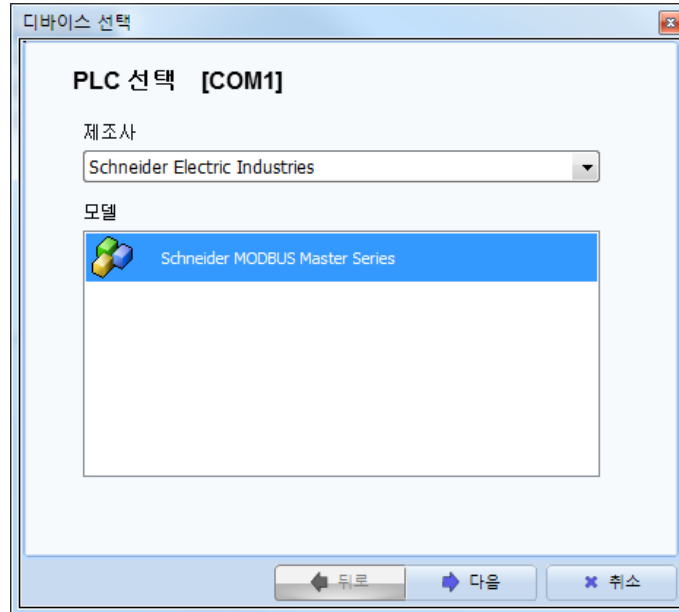


- 1 : N(TOP 1 대와 외부 장치 여러 대) 연결 - RS422/485 통신에서 가능한 구성입니다.



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 선택 합니다. <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>Serial</td> <td>사용자 설정</td> </tr> </tbody> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="2" style="background-color: #e0e0e0;">지원하는 프로토콜</th> </tr> </thead> <tbody> <tr> <td>MODBUS RTU</td> <td>MODBUS ASCII</td> </tr> </tbody> </table> 연결을 원하는 외부 장치가 시스템 구성 가능한 기종인지 1장의 시스템 구성에서 확인 하시기 바랍니다. | 모델 | 인터페이스 | 프로토콜 | Schneider MODBUS Master Series | Serial | 사용자 설정 | 지원하는 프로토콜 | | MODBUS RTU |
| 모델 | 인터페이스 | 프로토콜 | | | | | | | | | |
| Schneider MODBUS Master Series | Serial | 사용자 설정 | | | | | | | | | |
| 지원하는 프로토콜 | | | | | | | | | | | |
| MODBUS RTU | MODBUS ASCII | | | | | | | | | | |

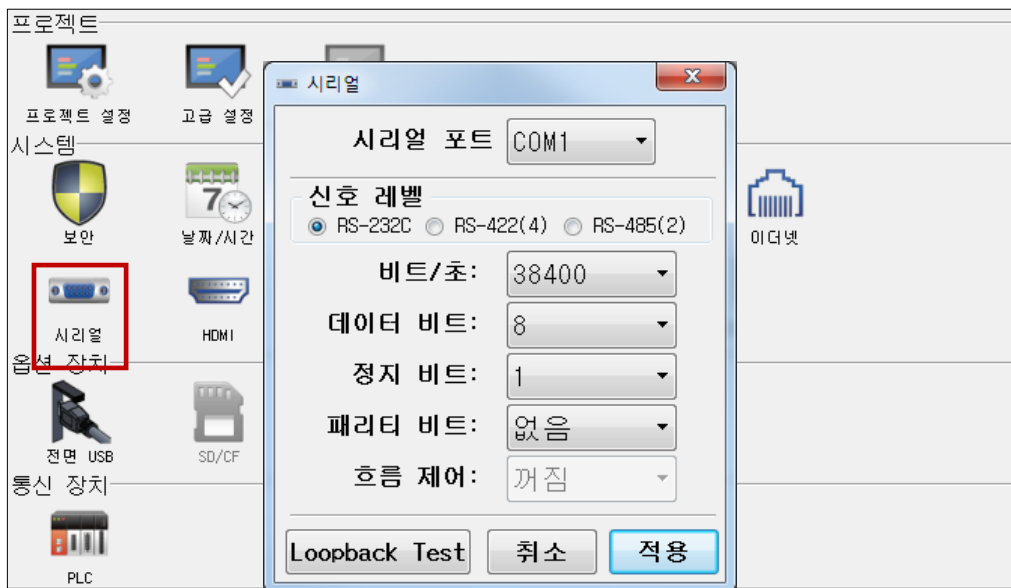
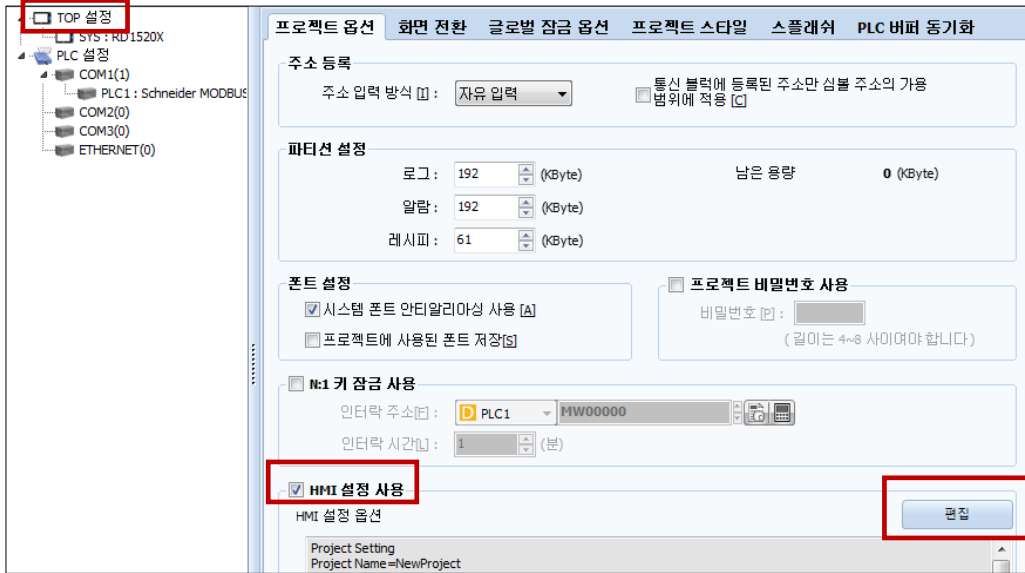
3. TOP 통신 설정

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 > 시리얼]
- TOP 통신 인터페이스를 TOP Design Studio에서 설정합니다.



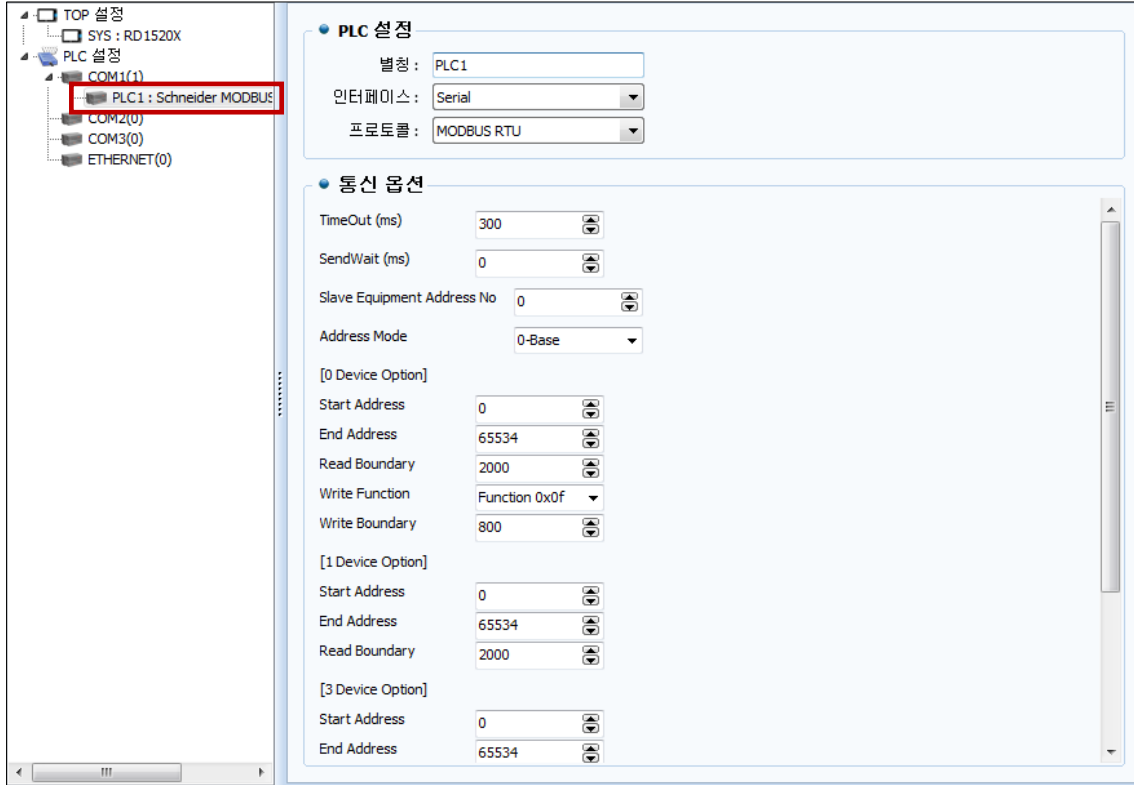
| 항 목 | TOP | 외부 장치 | 비 고 |
|------------|-----------------------|-----------------------|-----|
| 신호 레벨 (포트) | RS-232C RS-422/485 | RS-232C RS-422/485 | |
| 보우레이트 | | 38400 | |
| 데이터 비트 | | 8 | |
| 정지 비트 | | 1 | |
| 패리티 비트 | | 없음 | |

※ 위의 설정 내용은 본 사에서 권장하는 예제입니다.

| 항 목 | 설 명 |
|--------|-------------------------------------------|
| 신호 레벨 | TOP - 외부 장치 간 시리얼 통신 방식을 선택합니다. |
| 보우레이트 | TOP - 외부 장치 간 시리얼 통신 속도를 선택합니다. |
| 데이터 비트 | TOP - 외부 장치 간 시리얼 통신 데이터 비트를 선택합니다. |
| 정지 비트 | TOP - 외부 장치 간 시리얼 통신 정지 비트를 선택합니다. |
| 패리티 비트 | TOP - 외부 장치 간 시리얼 통신 패리티 비트 확인 방식을 선택합니다. |

(2) 통신 옵션 설정

- [프로젝트 > 프로젝트 속성 > PLC 설정 > COM > "PLC1 : Schneider MODBUS Master Series"]
 - Schneider MODBUS Serial Master 통신 드라이버의 옵션을 TOP Design Studio에서 설정합니다.



| 항 목 | 설 정 | 비 고 |
|----------------------------|----------------------------------------------------------------------------------------------------------------|----------------------------------|
| 인터페이스 | "Serial"를 선택합니다. | "2. 외부 장치 선택" 참고 |
| 프로토콜 | TOP - 외부 장치 간 통신 프로토콜을 선택합니다. | |
| TimeOut (ms) | TOP가 외부 장치로부터 응답을 기다리는 시간을 설정합니다. | |
| SendWait (ms) | TOP가 외부 장치로부터 응답 수신 후 다음 명령어 요청 전송 전에 대기 시간을 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) | |
| [0 Device Option] | | |
| 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 설정합니다. Force Single Coil : 05(Hex) / Force Multiple Coils : 0F(Hex) | |
| Write Boundary | Coil 에 대한 연속 쓰기 최대 개수를 설정합니다. | |
| [1 Device Option] | | |
| Start Address | Discrete Input 의 시작 주소를 입력합니다. | |
| End Address | Discrete Input 의 마지막 주소를 입력합니다. | |
| Read Boundary | Discrete Input 에 대한 연속 읽기 최대 개수를 설정합니다. | |
| [3 Device Option] | | |
| Start Address | Input Register 의 시작 주소를 입력합니다. | |
| End Address | Input Register 의 마지막 주소를 입력합니다. | |
| Read Boundary | Input Register 에 대한 연속 읽기 최대 개수를 설정합니다. | |
| [4 Device Option] | | |
| Start Address | Holding Register 의 시작 주소를 입력합니다. | |
| End Address | Holding Register 의 마지막 주소를 입력합니다. | |
| Read Boundary | Holding Register 에 대한 연속 읽기 최대 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 최대 개수를 설정합니다. | |

3.2 TOP 에서 통신 설정

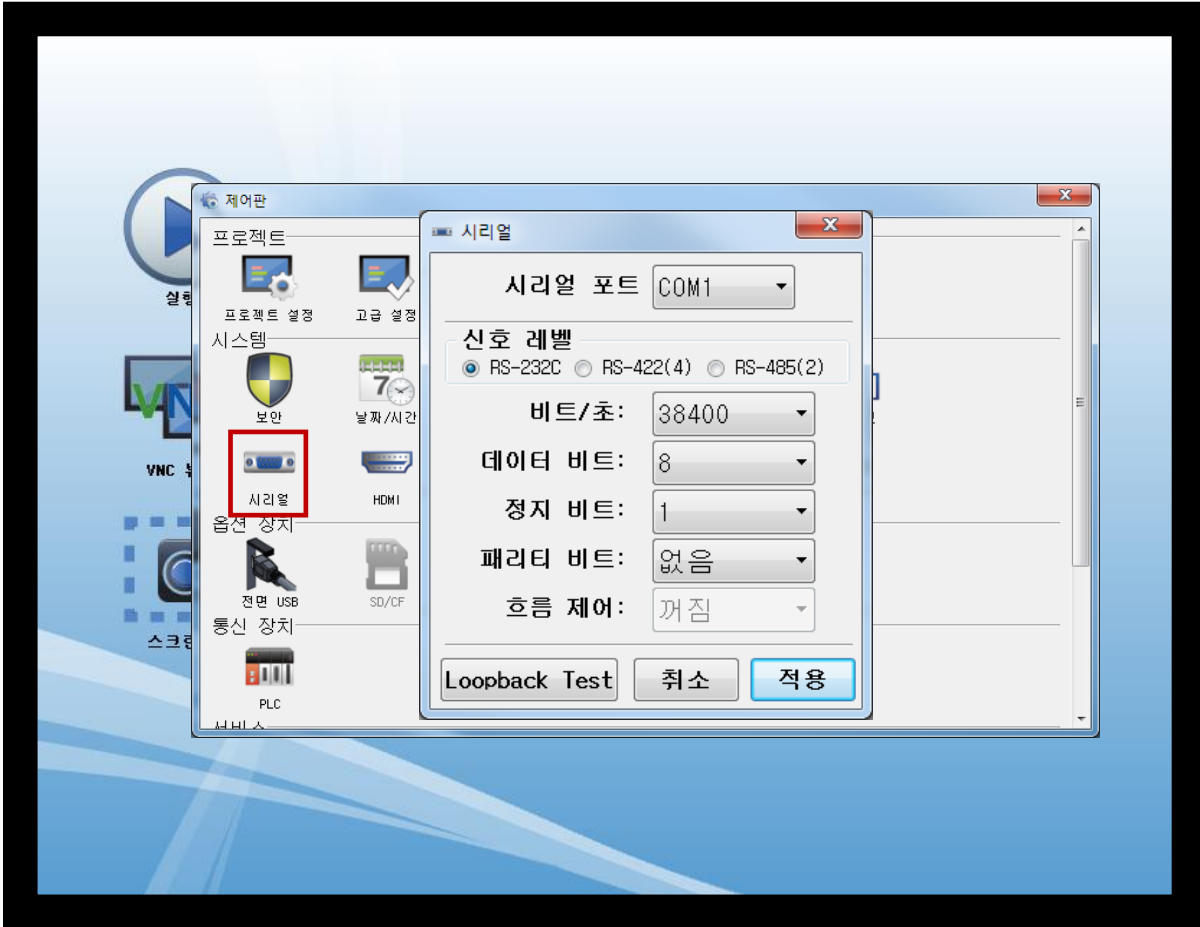
※ “3.1 TOP Design Studio 에서 통신 설정” 항목의 “HMI 설정 사용”을 체크 하지 않은 경우의 설정 방법입니다.

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



(1) 통신 인터페이스 설정

■ [메인 화면 > 제어판 > 시리얼]



| 항 목 | TOP | 외부 장치 | 비 고 |
|------------|-----------------------|-----------------------|-----|
| 신호 레벨 (포트) | RS-232C RS-422/485 | RS-232C RS-422/485 | |
| 보우레이트 | 38400 | | |
| 데이터 비트 | 8 | | |
| 정지 비트 | 1 | | |
| 패리티 비트 | 없음 | | |

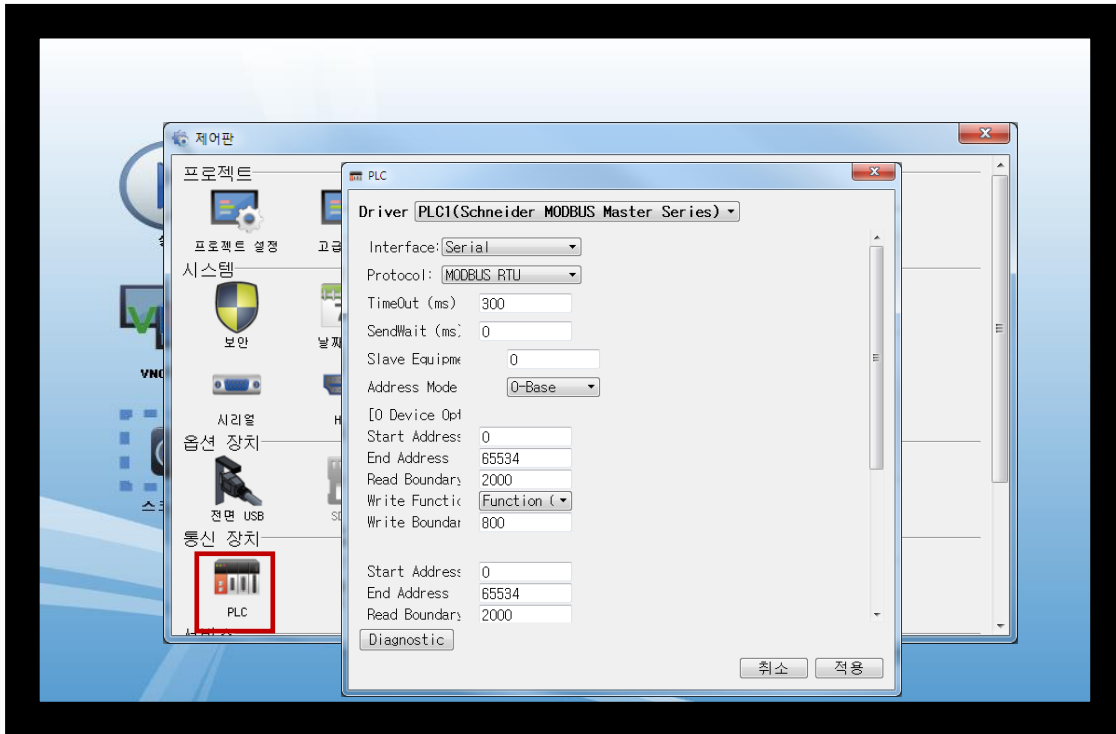
※ 위의 설정 내용은 본사에서 권장하는 설정 예제입니다.

| 항 목 | 설 명 |
|--------|-------------------------------------------|
| 신호 레벨 | TOP – 외부 장치 간 시리얼 통신 방식을 선택합니다. |
| 보우레이트 | TOP – 외부 장치 간 시리얼 통신 속도를 선택합니다. |
| 데이터 비트 | TOP – 외부 장치 간 시리얼 통신 데이터 비트를 선택합니다. |
| 정지 비트 | TOP – 외부 장치 간 시리얼 통신 정지 비트를 선택합니다. |
| 패리티 비트 | TOP – 외부 장치 간 시리얼 통신 패리티 비트 확인 방식을 선택합니다. |



(2) 통신 옵션 설정

■ [메인 화면 > 제어판 > PLC]



| 항 목 | 설 정 | 비 고 |
|----------------------------|----------------------------------------------------------------------------------------------------------------|----------------------------------|
| 인터페이스 | "Serial"를 선택합니다. | "2. 외부 장치 선택" 참고 |
| 프로토콜 | TOP - 외부 장치 간 통신 프로토콜을 선택합니다. | |
| TimeOut (ms) | TOP가 외부 장치로부터 응답을 기다리는 시간을 설정합니다. | |
| SendWait (ms) | TOP가 외부 장치로부터 응답 수신 후 다음 명령어 요청 전송 전에 대기 시간을 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) | |
| [0 Device Option] | | |
| Start Address | Coil 의 시작 주소를 입력합니다. | |
| End Address | Coil 의 마지막 주소를 입력합니다. | |
| Read Boundary | Coil 에 대한 연속 읽기 최대 개수를 설정합니다. | |
| Write Function | Coil 에 대한 쓰기 명령어를 설정합니다. Force Single Coil : 05(Hex) / Force Multiple Coils : 0F(Hex) | |
| Write Boundary | Coil 에 대한 연속 쓰기 최대 개수를 설정합니다. | |
| [1 Device Option] | | |
| Start Address | Discrete Input 의 시작 주소를 입력합니다. | |
| End Address | Discrete Input 의 마지막 주소를 입력합니다. | |
| Read Boundary | Discrete Input 에 대한 연속 읽기 최대 개수를 설정합니다. | |
| [3 Device Option] | | |
| Start Address | Input Register 의 시작 주소를 입력합니다. | |
| End Address | Input Register 의 마지막 주소를 입력합니다. | |
| Read Boundary | Input Register 에 대한 연속 읽기 최대 개수를 설정합니다. | |
| [4 Device Option] | | |
| Start Address | Holding Register 의 시작 주소를 입력합니다. | |
| End Address | Holding Register 의 마지막 주소를 입력합니다. | |
| Read Boundary | Holding Register 에 대한 연속 읽기 최대 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 최대 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 > 시리얼] 에서 사용 하고자 하는 COM 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 | 1. 시스템 구성 | |
| | 접속 케이블 명칭 | OK | NG | | |
| TOP | 버전 정보 | OK | NG | 2. 외부 장치 선택 3. 통신 설정 | |
| | 사용 포트 | OK | NG | | |
| | 드라이버 명칭 | OK | NG | | |
| | 기타 세부 설정 사항 | OK | NG | | |
| | 상대 국번 | 프로젝트 설정 | OK | | NG |
| | | 통신 진단 | OK | | NG |
| | 시리얼 파라미터 | 전송 속도 | OK | | NG |
| | | 데이터 비트 | OK | | NG |
| 정지 비트 | | OK | NG | | |
| 패리티 비트 | | OK | NG | | |
| 외부 장치 | CPU 명칭 | OK | NG | 4. 외부 장치 설정 | |
| | 통신 포트 명칭(모듈 명) | OK | NG | | |
| | 프로토콜(모드) | OK | NG | | |
| | 설정 국번 | OK | NG | | |
| | 기타 세부 설정 사항 | OK | NG | | |
| | 시리얼 파라미터 | 전송 속도 | OK | | NG |
| | | 데이터 비트 | OK | | NG |
| | | 정지 비트 | OK | | NG |
| 패리티 비트 | | OK | NG | | |
| 어드레스 범위 확인 | | OK | NG | 6. 지원 어드레스 (자세한 내용은 PLC 제조사의 매뉴얼을 참고 하시기 바랍니다.) | |

4. 외부 장치 설정

외부 장치의 사용자 매뉴얼을 참조하여 외부기기 I/F에 "MODBUS Serial Slave Driver"를 설정 하십시오.

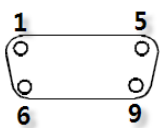
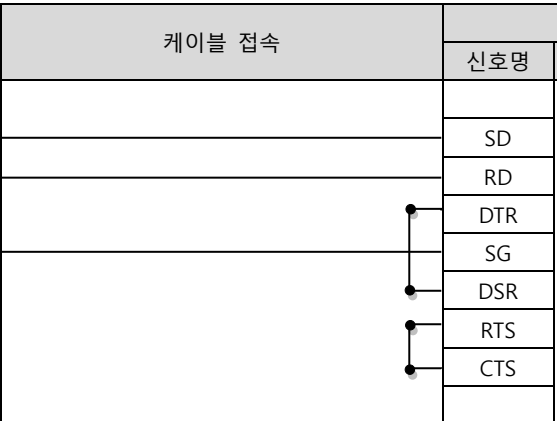


- Protocol Frame 형식 상에서 RTU/ASCII 모드 선정에 주의 하십시오.
- 외부 장치 측 어드레스 맵 내용을 확인하고 그 내용에 따라 통신 어드레스를 사용하십시오.

5. 케이블 표

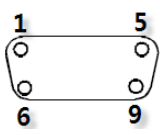
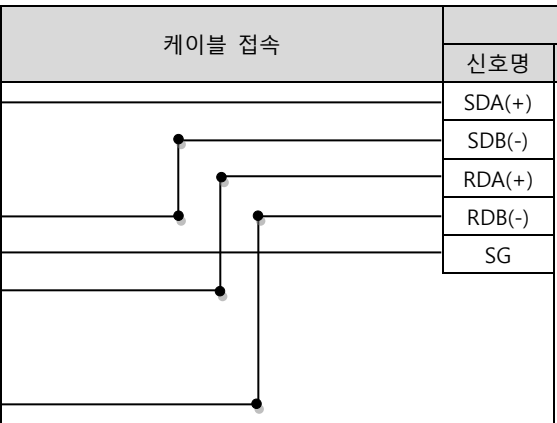
본 Chapter는 TOP와 해당 기기 간 정상 통신을 위한 케이블 다이어그램을 소개 합니다.
 (본 절에서 설명되는 케이블 다이어그램은 외부 장치 제조사의 권장사항과 다를 수 있습니다)

■ RS-232C (1 : 1 연결)

| TOP | | | 케이블 접속 | 외부 장치 | |
|------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|------------------------------------------------------------------------------------|-------|--|
| 핀 배열*주1) | 신호명 | 핀번호 | | 신호명 | |
|  <p>통신 케이블 커넥터 전면 기준, D-SUB 9 Pin male(수, 블록)</p> | CD | 1 |  | | |
| | RD | 2 | | SD | |
| | SD | 3 | | RD | |
| | DTR | 4 | | DTR | |
| | SG | 5 | | SG | |
| | DSR | 6 | | DSR | |
| | RTS | 7 | | RTS | |
| | CTS | 8 | | CTS | |
| | | 9 | | | |

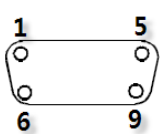
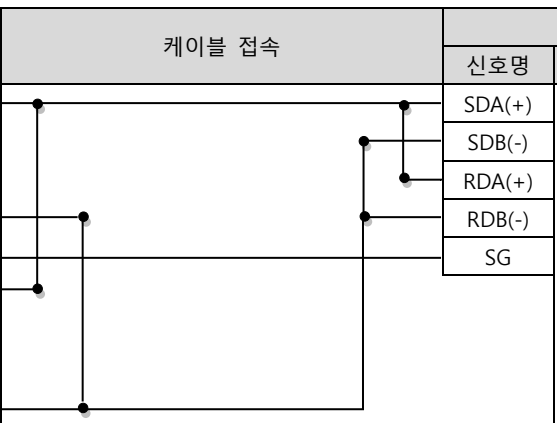
*주1) 핀 배열은 케이블 접속 커넥터의 접속면에서 본 것 입니다.

■ RS-422 (1 : 1 연결)

| TOP | | | 케이블 접속 | 외부 장치 | |
|-------------------------------------------------------------------------------------------------------------------------------------------------|--------|-----|-------------------------------------------------------------------------------------|--------|--|
| 핀 배열*주1) | 신호명 | 핀번호 | | 신호명 | |
|  <p>통신 케이블 커넥터 전면 기준, D-SUB 9 Pin male(수, 블록)</p> | RDA(+) | 1 |  | SDA(+) | |
| | | 2 | | SDB(-) | |
| | | 3 | | RDA(+) | |
| | RDB(-) | 4 | | RDB(-) | |
| | SG | 5 | | SG | |
| | SDA(+) | 6 | | | |
| | | 7 | | | |
| | | 8 | | | |
| | SDB(-) | 9 | | | |

*주1) 핀 배열은 케이블 접속 커넥터의 접속면에서 본 것 입니다.

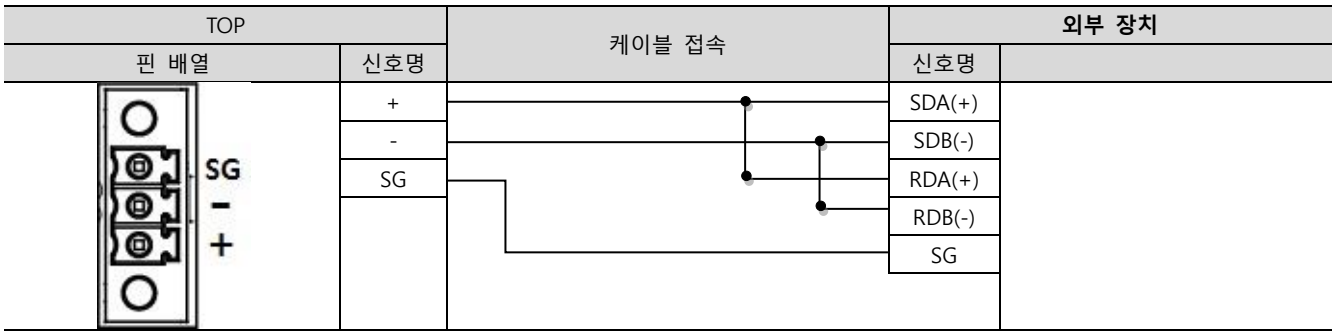
■ RS-485 (1 : 1 연결)

| TOP | | | 케이블 접속 | 외부 장치 | |
|--------------------------------------------------------------------------------------------------------------------------------------------------|--------|-----|--------------------------------------------------------------------------------------|--------|--|
| 핀 배열*주1) | 신호명 | 핀번호 | | 신호명 | |
|  <p>통신 케이블 커넥터 전면 기준, D-SUB 9 Pin male(수, 블록)</p> | RDA(+) | 1 |  | SDA(+) | |
| | | 2 | | SDB(-) | |
| | | 3 | | RDA(+) | |
| | RDB(-) | 4 | | RDB(-) | |
| | SG | 5 | | SG | |
| | SDA(+) | 6 | | | |
| | | 7 | | | |
| | | 8 | | | |
| | SDB(-) | 9 | | | |

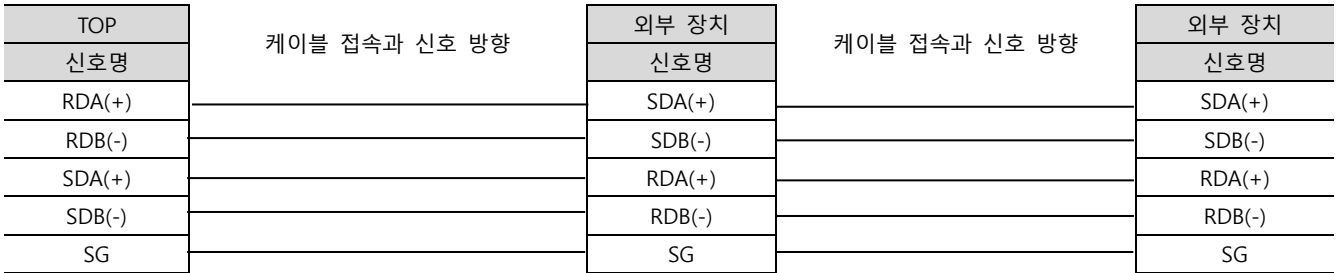
*주1) 핀 배열은 케이블 접속 커넥터의 접속면에서 본 것 입니다.

☞ 다음 페이지에서 계속 됩니다.

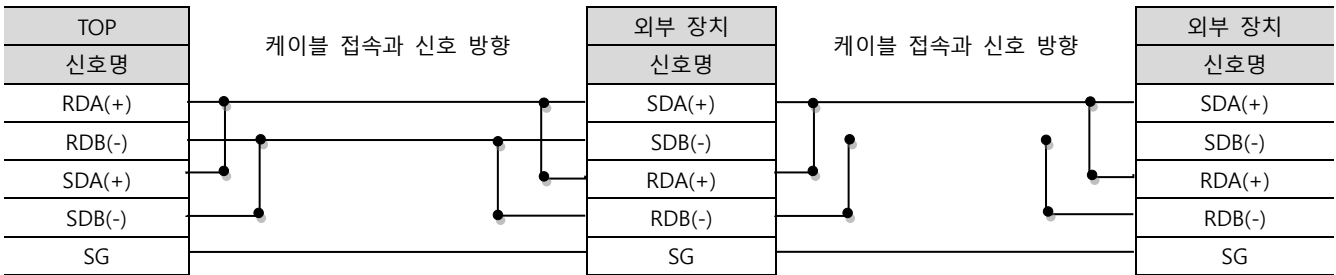
■ RS-485 (1 : 1 연결)



■ RS-422 (1 : N 연결) - 1:1 연결을 참고하여 아래의 방식으로 연결 하십시오.



■ RS-485 (1 : N / N : 1 연결) - 1:1 연결을 참고하여 아래의 방식으로 연결 하십시오.



6. 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.

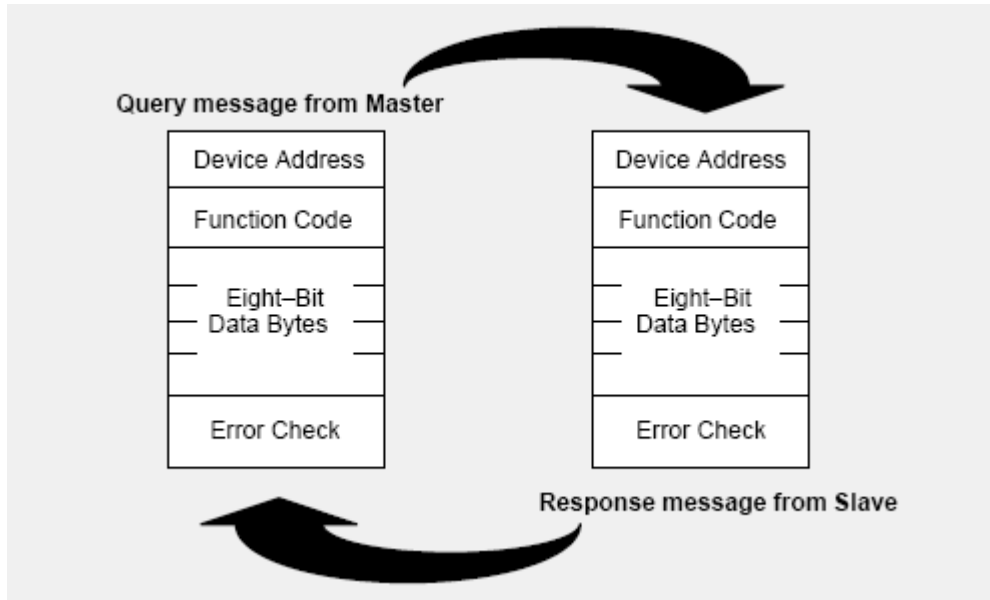
| | Bit Address | Word Address | 32 bits | Remarks |
|------------------|-------------------------|-------------------|---------|---------|
| Coil | Q00001 - Q065536 | Q00001 - Q65521 | L/H | |
| Discrete Input | I00001 - I65536 | I00001 - I65521 | | *주1) |
| Input Register | IW00001.00 - IW65536.15 | IW00001 - IW65536 | | *주1) |
| Holding Register | MW00001.00 - MW65536.15 | MW00001 - MW65536 | | |

*주1) 쓰기 불가능(읽기 전용)

Appendix A. Standard MODBUS Protocol

본 기기의 "MODBUS Serial Master Driver"가 지원하는 MODBUS 프로토콜 명령어 및 디바이스에 대해 설명 합니다.

At the message level, the MODBUS protocol still applies the master-slave principle even though the network communication method is peer-to-peer. If a controller originates a message, it does so as a master device, and expects a response from a slave device. Similarly, when a controller receives a message it constructs a slave response and returns it to the originating controller.



The Query: The function code in the query tells the addressed slave device what kind of action to perform. The data bytes contain any additional information that the slave will need to perform the function. For example, function code 03 will query the slave to read holding registers and respond with their contents. The data field must contain the information telling the slave which register to start at and how many registers to read. The error check field provides a method for the slave to validate the integrity of the message contents.

The Response: If the slave makes a normal response, the function code in the response is an echo of the function code in the query. The data bytes contain the data collected by the slave, such as register values or status. If an error occurs, the function code is modified to indicate that the response is an error response, and the data bytes contain a code that describes the error. The error check field allows the master to confirm that the message contents are valid.

A.1 "0" Device (Coil)

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 | Slave 기구번호 | 명령어 | | 선바다바이시 | | 다바이시편수 | | 체크점비 (CRC) | |
| | H L | H L | H L | L H | L H | | | | |
| Hex | 11 | 01 | 00 | 13 | 00 | 25 | — | — | |

(Slave → Master : 응답 프레임)

| | | | | | | | | | | |
|---------|------------|-----|--------------|-----|----|----|----|------------|-----|---|
| Comment | Slave 기구번호 | 명령어 | 데이터 개수(byte) | 데이터 | | | | 체크점비 (CRC) | | |
| | H L | H L | L | L | L | L | L | L H | L H | |
| Hex | 11 | 01 | 05 | CD | 6B | B2 | 0E | 1B | — | — |

■ Coils 데이터 상태

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

0: OFF / 1:ON

■ ASCII Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | | | | | | |
|---------|--------|------------|-----|-----|----|--------|----|----|----|--------|----|----|------------|---|------|----|----|
| comment | Header | Slave 기구번호 | | 명령어 | | 선바다바이시 | | | | 다바이시편수 | | | 체크점비 (CRC) | | Tail | | |
| | H L | H L | H L | H L | L | L | L | L | L | L | L | L | L | L | CR | LF | |
| ASCII | : | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 3 | 0 | 0 | 2 | 5 | — | — | 0D | 0A |
| Hex | 3A | 31 | 31 | 30 | 31 | 30 | 30 | 31 | 33 | 30 | 30 | 32 | 35 | — | — | 0D | 0A |

(Slave → Master : 응답 프레임)

| | | | | | | | | | | | | | | | | | | | | | |
|---------|--------|------------|-----|-----|----|--------------|----|-----|----|----|----|------------|----|------|----|----|----|---|---|----|----|
| Comment | Header | Slave 기구번호 | | 명령어 | | 데이터 개수(byte) | | 데이터 | | | | 체크점비 (CRC) | | Tail | | | | | | | |
| | H L | H L | H L | L | L | L | L | L | L | L | L | L | L | L | CR | LF | | | | | |
| ASCII | : | 1 | 1 | 0 | 1 | 0 | 5 | C | D | 6 | B | B | 2 | 0 | E | 1 | B | — | — | 0D | 0A |
| Hex | 3A | 31 | 31 | 30 | 31 | 30 | 35 | 43 | 44 | 36 | 42 | 42 | 32 | 30 | 45 | 31 | 42 | — | — | 0D | 0A |



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 | Slave 기판 | 명령어 | | 선두디바이스 | | | | Force data | | | | 체크점비 (CRC) | |
| | | H | L | H | L | H | L | L | L | H | H | L | H |
| Hex | 11 | 05 | 00 | AC | FF | 00 | — | — | — | — | — | — | |

(Slave → Master : 응답 프레임)

| | | | | | | | | | | | | | |
|---------|----------|-----|----|--------|----|----|---|------------|---|---|---|------------|---|
| Comment | Slave 기판 | 명령어 | | 선두디바이스 | | | | Force data | | | | 체크점비 (CRC) | |
| | | H | L | H | L | H | L | L | L | H | H | L | H |
| Hex | 11 | 05 | 00 | AC | FF | 00 | — | — | — | — | — | — | |

■ Force Data

| | High | Low |
|-----------|-----------------|-----------------|
| Force ON | FF _H | 00 _H |
| Force OFF | 00 _H | 00 _H |

■ ASCII Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | | | | | | |
|---------|--------|----------|----|-----|----|--------|----|----|----|------------|----|----|----|------------|---|------|----|
| comment | Header | Slave 기판 | | 명령어 | | 선두디바이스 | | | | Force data | | | | 체크점비 (CRC) | | Tail | |
| | | H | L | H | L | H | - | - | L | H | - | - | L | L | H | CR | LF |
| ASCII | : | 1 | 1 | 0 | 5 | 0 | 0 | 1 | 3 | 0 | 0 | 2 | 5 | — | — | 0D | 0A |
| Hex | 3A | 31 | 31 | 30 | 31 | 30 | 30 | 41 | 43 | 45 | 45 | 30 | 30 | — | — | 0D | 0A |

(Slave → Master : 응답 프레임)

| | | | | | | | | | | | | | | | | | |
|---------|--------|----------|----|-----|----|--------|----|----|----|------------|----|----|----|------------|---|------|----|
| comment | Header | Slave 기판 | | 명령어 | | 선두디바이스 | | | | Force data | | | | 체크점비 (CRC) | | Tail | |
| | | H | L | H | L | H | - | - | L | H | - | - | L | L | H | CR | LF |
| ASCII | : | 1 | 1 | 0 | 5 | 0 | 0 | 1 | 3 | 0 | 0 | 2 | 5 | — | — | 0D | 0A |
| Hex | 3A | 31 | 31 | 30 | 31 | 30 | 30 | 41 | 43 | 45 | 45 | 30 | 30 | — | — | 0D | 0A |

A.2 "1" Device (Discrete Input)

Read Input Status : 02

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

RTU Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | |
|---------|----------|-----|----|--------|----|--------|---|------------|---|
| Comment | Slave 구동 | 명령어 | | 선두디바이스 | | 디바이스점수 | | 체크점비 (CRC) | |
| | | H | L | H | L | L | H | L | H |
| Hex | 11 | 02 | 00 | C4 | 00 | 16 | — | — | — |

(Slave → Master : 응답 프레임)

| | | | | | | | | |
|---------|----------|-----|--------------|-------------|-------------|-------------|------------|---|
| Comment | Slave 구동 | 명령어 | 데이터 개수(byte) | 데이터(Inputs) | | | 체크점비 (CRC) | |
| | | | | 10204~10197 | 10212~10205 | 10218~10213 | L | H |
| Hex | 11 | 02 | 03 | AC | DB | 35 | — | — |

Coils 데이터 상태

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

0: OFF / 1:ON

ASCII Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | | | | | | |
|---------|--------|----------|----|-----|----|--------|----|----|--------|----|----|----|------------|---|------|----|----|
| comment | Header | Slave 구동 | | 명령어 | | 선두디바이스 | | | 디바이스점수 | | | | 체크점비 (CRC) | | Tail | | |
| | | H | L | H | L | H | - | - | L | H | - | - | L | L | H | CR | LF |
| ASCII | : | 1 | 1 | 0 | 2 | 0 | 0 | C | 4 | 0 | 0 | 1 | 6 | — | — | 0D | 0A |
| Hex | 3A | 31 | 31 | 30 | 32 | 30 | 30 | 43 | 34 | 30 | 30 | 31 | 36 | — | — | 0D | 0A |

(Slave → Master : 응답 프레임)

| | | | | | | | | | | | | | | | | | |
|---------|--------|----------|----|-----|----|--------------|-------------|-------------|-------------|-------------|----|----|------------|----|------|----|----|
| Comment | Header | Slave 구동 | | 명령어 | | 데이터 개수(byte) | 데이터(Inputs) | | | | | | 체크점비 (CRC) | | Tail | | |
| | | H | L | H | L | 0 | 3 | 10204~10197 | 10212~10205 | 10218~10213 | L | H | CR | LF | | | |
| ASCII | : | 1 | 1 | 0 | 2 | 0 | 3 | A | C | D | B | 3 | 5 | — | — | 0D | 0A |
| Hex | 3A | 31 | 31 | 30 | 31 | 30 | 35 | 41 | 43 | 44 | 42 | 33 | 35 | — | — | 0D | 0A |

A.3 "3" Device (Input Register)

Read Input Registers : 04

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

■ RTU Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | |
|---------|------------|------|-----|--------|-----|-------------------------|---|------------|--|
| Comment | Slave 기구번호 | 패킷번호 | | 신부디바이스 | | 디바이스주소수 (Word Count) | | 체크점비 (CRC) | |
| | H L | H L | H L | H L | L H | L H | | | |
| Hex | 11 | 04 | 00 | 08 | 00 | 01 | — | — | |

(Slave → Master : 응답 프레임)

| | | | | | | |
|---------|------------|------|--------------|--------------------------|------------|---|
| Comment | Slave 기구번호 | 패킷번호 | 데이터 개수(byte) | 데이터 Register 30009 | 체크점비 (CRC) | |
| | H L | H L | H L | L H | | |
| Hex | 11 | 04 | 02 | 00 | 0A | — |

■ ASCII Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | | | | |
|---------|--------|------------|-----|------|----|--------|----|----|-------------------|----|----|------------|----|------|----|
| comment | Header | Slave 기구번호 | | 패킷번호 | | 신부디바이스 | | | 디바이스주소수 (Word) | | | 체크점비 (CRC) | | Tail | |
| | H L | H L | H L | H L | - | - | L | H | - | - | L | L | H | CR | LF |
| ASCII | : | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 1 | — | — |
| Hex | 3A | 31 | 31 | 30 | 31 | 30 | 30 | 30 | 38 | 30 | 30 | 30 | 31 | 0D | 0A |

(Slave → Master : 응답 프레임)

| | | | | | | | | | | | | | |
|---------|--------|------------|-----|------|----|--------------|----|--------------------------|----|------------|----|------|----|
| Comment | Header | Slave 기구번호 | | 패킷번호 | | 데이터 개수(byte) | | 데이터 Register 40108 | | 체크점비 (CRC) | | Tail | |
| | H L | H L | H L | H L | H | - | - | L | L | H | CR | LF | |
| ASCII | : | 1 | 1 | 0 | 4 | 0 | 2 | 0 | 0 | 0 | A | — | — |
| Hex | 3A | 31 | 31 | 30 | 31 | 30 | 35 | 30 | 30 | 30 | 41 | 0D | 0A |

A.4 "4" Device (Holding Register)

Read Holding Registers : 03

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

■ RTU Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | |
|---------|----------|-----|----|--------|----|--------|---|------------|---|
| Comment | Slave 기번 | 명령어 | | 전나드바이스 | | 디바이스주소 | | 체크섬비 (CRC) | |
| | | H | L | H | L | L | H | L | H |
| Hex | 11 | 03 | 00 | 6B | 00 | 03 | — | — | |

(Slave → Master : 응답 프레임)

| | | | | | | | | | | | | | |
|---------|----------|-----|----------------|----------------|----------------|----|----|----|----|------------|---|---|---|
| Comment | Slave 기번 | 명령어 | 데이터 개수(byte) | 데이터 | | | | | | 체크섬비 (CRC) | | | |
| | | | Register 40108 | Register 40109 | Register 40110 | | | | | | | L | H |
| Hex | 11 | 03 | 06 | 02 | 2B | 00 | 00 | 00 | 64 | — | — | | |

■ ASCII Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | | | | | | |
|---------|--------|----------|----|-----|----|--------|----|----|---------------|----|----|------------|----|------|----|----|----|
| comment | Header | Slave 기번 | | 명령어 | | 전나드바이스 | | | 디바이스주소 (Word) | | | 체크섬비 (CRC) | | Tail | | | |
| | | H | L | H | L | H | - | - | L | H | - | - | L | H | CR | LF | |
| ASCII | : | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 3 | 0 | 0 | 2 | 5 | — | — | 0D | 0A |
| Hex | 3A | 31 | 31 | 30 | 31 | 30 | 30 | 31 | 33 | 30 | 30 | 32 | 35 | — | — | | |

(Slave → Master : 응답 프레임)

| | | | | | | | | | | | | | | | | | | | | | |
|---------|--------|----------|----|-----|----|--------------|----|----------------|----------------|----------------|----|----|----|------------|----|------|----|----|----|----|----|
| Comment | Header | Slave 기번 | | 명령어 | | 데이터 개수(byte) | | 데이터 | | | | | | 체크섬비 (CRC) | | Tail | | | | | |
| | | | | | | | | Register 40108 | Register 40109 | Register 40110 | | | | | | | L | H | | | |
| ASCII | : | 1 | 1 | 0 | 3 | 0 | 6 | 0 | 2 | 2 | B | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 4 | CR | LF |
| Hex | 3A | 31 | 31 | 30 | 31 | 30 | 35 | 30 | 32 | 32 | 42 | 30 | 30 | 30 | 30 | 30 | 30 | 36 | 34 | 0D | 0A |

Preset Single Register : 06

Slave 기기 측의 40002 Register 에 00 03 (hex) 데이터를 입력 하는 예제를 통해 "06"명령어 프레임을 설명 합니다.

■ RTU Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | |
|---------|------------|-----|----|---------|----|-------------|---|------------|---|
| Comment | Slave 기구번호 | 명령어 | | 선택데이터비트 | | Preset data | | 체크섬비 (CRC) | |
| | | H | L | H | L | H | L | L | H |
| Hex | 11 | 06 | 00 | 01 | 00 | 03 | — | — | |

(Slave → Master : 응답 프레임)

| | | | | | | | | | |
|---------|------------|-----|----|---------|----|-------------|---|------------|---|
| Comment | Slave 기구번호 | 명령어 | | 선택데이터비트 | | Preset data | | 체크섬비 (CRC) | |
| | | H | L | H | L | H | L | L | H |
| Hex | 11 | 06 | 00 | 01 | 00 | 03 | — | — | |

■ ASCII Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | | | | | | |
|---------|--------|------------|----|-----|----|---------|----|----|----|-------------|----|----|------------|---|------|----|----|
| comment | Header | Slave 기구번호 | | 명령어 | | 선택데이터비트 | | | | Preset data | | | 체크섬비 (CRC) | | Tail | | |
| | | H | L | H | L | H | - | - | L | H | - | - | L | L | H | CR | LF |
| ASCII | : | 1 | 1 | 0 | 6 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 3 | — | — | 0D | 0A |
| Hex | 3A | 31 | 31 | 30 | 36 | 30 | 30 | 30 | 31 | 30 | 30 | 30 | 33 | — | — | 0D | 0A |

(Slave → Master : 응답 프레임)

| | | | | | | | | | | | | | | | | | |
|---------|--------|------------|----|-----|----|---------|----|----|----|-------------|----|----|------------|---|------|----|----|
| comment | Header | Slave 기구번호 | | 명령어 | | 선택데이터비트 | | | | Preset data | | | 체크섬비 (CRC) | | Tail | | |
| | | H | L | H | L | H | - | - | L | H | - | - | L | L | H | CR | LF |
| ASCII | : | 1 | 1 | 0 | 6 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 3 | — | — | 0D | 0A |
| Hex | 3A | 31 | 31 | 30 | 36 | 30 | 30 | 30 | 31 | 30 | 30 | 30 | 33 | — | — | 0D | 0A |

Preset Multiple Register : 10

Slave 기기 측의 40002 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)

■ RTU Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | | | | | | |
|---------|----------|-----|----|--------|----|-----------------------------------|----|--------------|----|----------------|----|----------------|---|--|--|------------|--|
| Comment | Slave 개수 | 프레임 | | 전투디바이스 | | | | 데이터 개수(Byte) | | | | 데이터 | | | | 체크섬비 (CRC) | |
| | | | | | | Quantity of Register (Word Count) | | | | Register 40002 | | Register 40003 | | | | | |
| Hex | 11 | 10 | 00 | 01 | 00 | 02 | 04 | 00 | 0A | 01 | 02 | — | — | | | | |

(Slave → Master : 응답 프레임)

| | | | | | | | | | |
|---------|----------|-----|----|-----------------------------------|----|----|---|------------|--|
| Comment | Slave 개수 | 프레임 | | 전투디바이스 | | | | 체크섬비 (CRC) | |
| | | | | Quantity of Register (Word Count) | | | | | |
| Hex | 11 | 10 | 00 | 01 | 00 | 02 | — | — | |

■ ASCII Mode

(Master → Slave : 요청 프레임)

| | | | | | | | | | | | | | | | | | | | | | |
|---------|--------|----------|----|-----|----|-----------------------------------|----|----|----|----------------|----|----------------|----|-----|----|----|----|----|----|----|----|
| comment | Header | Slave 개수 | | 프레임 | | 전투디바이스 | | | | 데이터 개수(Byte) | | | | 데이터 | | | | | | | |
| | | | | | | Quantity of Register (Word Count) | | | | Register 40002 | | Register 40003 | | | | | | | | | |
| ASCII | : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | A | 0 | 1 | 0 | 2 |
| Hex | 3A | 31 | 31 | 31 | 30 | 30 | 30 | 41 | 43 | 30 | 30 | 30 | 32 | 30 | 34 | 30 | 41 | 30 | 31 | 30 | 32 |

계속...

| | | |
|------|------------|----|
| Tail | 체크섬비 (CRC) | |
| | CR | LF |
| Hex | 0D | 0A |

(Slave → Master : 응답 프레임)

| | | | | | | | | | | | | | | | |
|---------|--------|----------|----|-----|----|-----------------------------------|----|----|----|--------------|----|----|----|------|----|
| comment | Header | Slave 개수 | | 프레임 | | 전투디바이스 | | | | 데이터 개수(Byte) | | | | Tail | |
| | | | | | | Quantity of Register (Word Count) | | | | 체크섬비 (CRC) | | | | | |
| ASCII | : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | CR | LF |
| Hex | 3A | 31 | 31 | 30 | 31 | 30 | 30 | 30 | 31 | 30 | 30 | 30 | 32 | 0D | 0A |

A.5 LRC/CRC Generation

(1) LRC Generation

The Longitudinal Redundancy Check (LRC) field is one byte, containing an 8-bit binary value. The LRC value is calculated by the transmitting device, which appends the LRC to the message. The receiving device recalculates an LRC during receipt of the message, and compares the calculated value to the actual value it received in the LRC field. If the two values are not equal, an error results.

The LRC is calculated by adding together successive 8-bit bytes in the message, discarding any carries, and then two's complementing the result. The LRC is an 8-bit field, therefore each new addition of a character that would result in a value higher than 255 decimal simply 'rolls over' the field's value through zero. Because there is no ninth bit, the carry is discarded automatically.

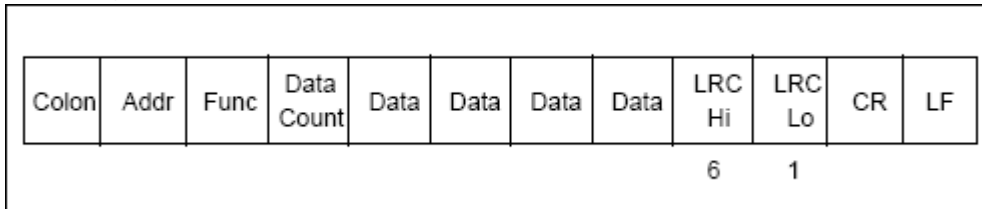
A procedure for generating an LRC is:

1. Add all bytes in the message, excluding the starting 'colon' and ending CRLF. Add them into an 8-bit field, so that carries will be discarded.
2. Subtract the final field value from FF hex (all 1's), to produce the ones-complement.
3. Add 1 to produce the twos-complement.

– Placing the LRC into the Message

When the 8-bit LRC (2 ASCII characters) is transmitted in the message, the high-order character will be transmitted first, followed by the low-order character.

For example, if the LRC value is 61 hex (0110 0001):



– Example

An example of a C language function performing LRC generation is shown below.

The function takes two arguments:

```
unsigned char *auchMsg ;           // A pointer to the message buffer containing
                                   // binary data to be used for generating the LRC
unsigned short usDataLen ;        // The quantity of bytes in the message buffer.
```

The function returns the LRC as a type unsigned char.

– LRC Generation Function

```
static unsigned char LRC(auchMsg, usDataLen)
unsigned char *auchMsg ;          /* message to calculate LRC upon */
unsigned short usDataLen ;       /* quantity of bytes in message */
{
    unsigned char uchLRC = 0 ;    /* LRC char initialized */
    while (usDataLen--)          /* pass through message buffer */
        uchLRC += *auchMsg++;    /* add buffer byte without carry */
    return ((unsigned char)-((char)uchLRC)); /* return twos complement */
}
```

(2) CRC Generation

The Cyclical Redundancy Check (CRC) field is two bytes, containing a 16-bit binary value. The CRC value is calculated by the transmitting device, which appends the CRC to the message. The receiving device recalculates a CRC during receipt of the message, and compares the calculated value to the actual value it received in the CRC field. If the two values are not equal, an error results.

The CRC is started by first preloading a 16-bit register to all 1's. Then a process begins of applying successive 8-bit bytes of the message to the current contents of the register. Only the eight bits of data in each character are used for generating the CRC. Start and stop bits, and the parity bit, do not apply to the CRC.

During generation of the CRC, each 8-bit character is exclusive ORed with the register contents. Then the result is shifted in the direction of the least significant bit (LSB), with a zero filled into the most significant bit (MSB) position. The LSB is extracted and examined. If the LSB was a 1, the register is then exclusive ORed with a preset, fixed value. If the LSB was a 0, no exclusive OR takes place.

This process is repeated until eight shifts have been performed. After the last (eighth) shift, the next 8-bit character is exclusive ORed with the register's current value, and the process repeats for eight more shifts as described above. The final contents of the register, after all the characters of the message have been applied, is the CRC value.

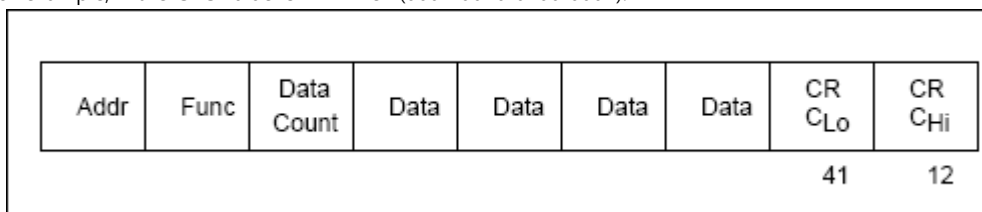
A procedure for generating a CRC is:

1. Load a 16-bit register with FFFF hex (all 1's). Call this the CRC register.
2. Exclusive OR the first 8-bit byte of the message with the low-order byte of the 16-bit CRC register, putting the result in the CRC register.
3. Shift the CRC register one bit to the right (toward the LSB), zero-filling the MSB. Extract and examine the LSB.
4. (If the LSB was 0): Repeat Step 3 (another shift). (If the LSB was 1): Exclusive OR the CRC register with the polynomial value A001 hex (1010 0000 0000 0001).
5. Repeat Steps 3 and 4 until 8 shifts have been performed. When this is done, a complete 8-bit byte will have been processed.
6. Repeat Steps 2 through 5 for the next 8-bit byte of the message. Continue doing this until all bytes have been processed.
7. The final contents of the CRC register is the CRC value.
8. When the CRC is placed into the message, its upper and lower bytes must be swapped as described below.

– Placing the CRC into the Message

When the 16-bit CRC (two 8-bit bytes) is transmitted in the message, the low-order byte will be transmitted first, followed by the high-order byte.

For example, if the CRC value is 1241 hex (0001 0010 0100 0001):



– Example

An example of a C language function performing CRC generation is shown on the following pages. All of the possible CRC values are preloaded into two arrays, which are simply indexed as the function increments through the message buffer.

One array contains all of the 256 possible CRC values for the high byte of the 16-bit CRC field, and the other array contains all of the values for the low byte. Indexing the CRC in this way provides faster execution than would be achieved by calculating a new CRC value with each new character from the message buffer.

Note This function performs the swapping of the high/low CRC bytes internally. The bytes are already swapped in the CRC value that is returned from the function. Therefore the CRC value returned from the function can be directly placed into the message for transmission.

The function takes two arguments:

```
unsigned char *puchMsg ;           //A pointer to the message buffer containing
                                   //binary data to be used for generating the CRC
unsigned short usDataLen ;        //The quantity of bytes in the message buffer.
```

The function returns the CRC as a type unsigned short.

