# FnIO M - Series:

# *M9222*

***M9222 (PROFIBUS Network Adapter(Single Type))***

Date: 2020.07.14

# Table of Contents

## History

| REV. | PAGES | REMARKS | DATE | Editor |
|---|---|---|---|---|
| 1.00 | | Preliminary | Nov 25, 2019 | JY BAE |
| 1.03 | 34 | Modify Firmware Revision | 2020/10/29 | CW SEO |
| 1.04 | | Remove Description pages of Hot Swap Function, Use in Hazardous Envirnments and Caution(Before using the unit) | 2020/12/7 | SJ LIM |
| | | | | |
| | | | | |

# 1. ENVIRONMENT SPECIFICATION

| **Environmental specification** | |
|---|---|
| Operating Temperature | -25℃~60℃ |
| UL Temperature | -20℃~60℃ |
| Storage Temperature | -40℃~85℃ |
| Relative Humidity | 5% ~ 90% non-condensing |
| Mounting | DIN rail |
| | |
| **General specification** | |
| Shock Operating | IEC 60068-2-27 |
| Vibration Resistance | Based on IEC 60068-2-6<br>DNVGL-CG-0039 : Vibration Class B, 4g |
| Industrial Emissions | EN 61000-6-4/A11 : 2011 |
| Industrial Immunity | EN 61000-6-2 : 2005 |
| Installation Position | Vertical and horizontal installation is available. |
| Product Certifications | CE, UL, FCC, ATEX |

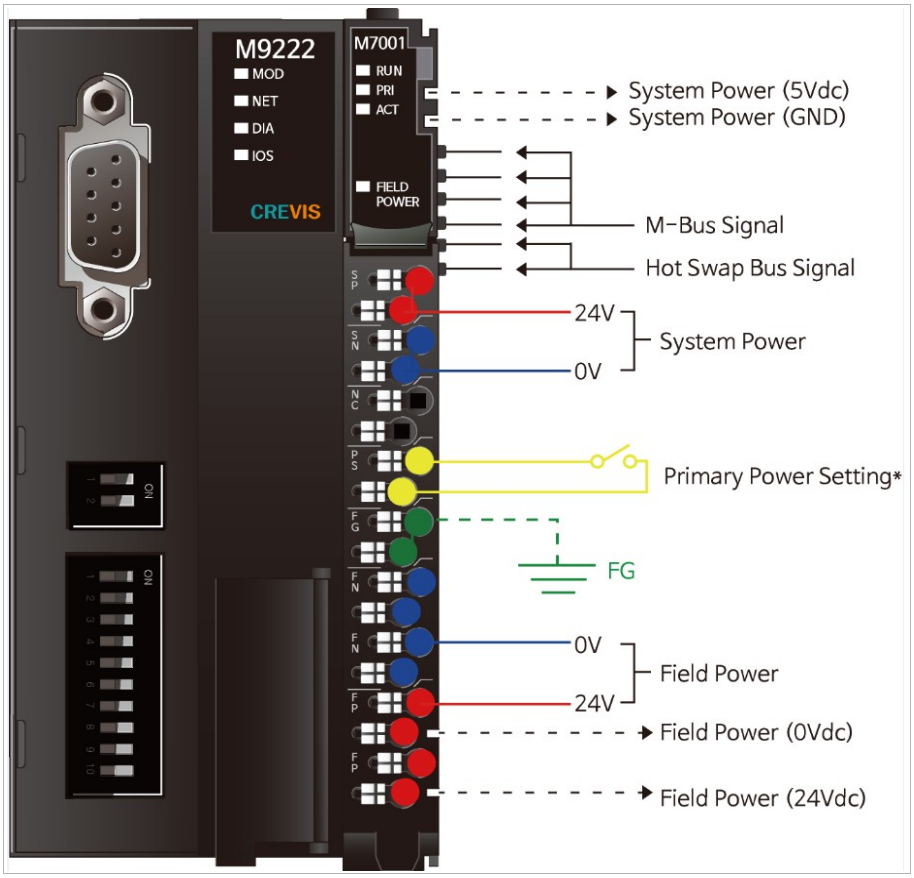
# 2. M9222 (Profibus NETWORK ADAPTER)

## 2.1. M9222 Specification

| Items | Specification |
|---|---|
| **Communication Interface Specification** | |
| Repeater Control Signal | RS-485 Signal |
| Freeze mode | Support |
| Sync mode | Support |
| Auto baudrate | Support |
| Station type | Slave |
| FMS support | Not supported |
| Max. Network Node | 125 Nodes |
| Max. Expansion Slot | 63 Slots |
| I/O Data Size | Input : 244bytes / Output : 244bytes |
| Indicator | 4 LEDs<br>1 Green/Red, Module Status (MOD)<br>1 Green, Receive Data (NET)<br>1 Green, Transmit Data (DIA)<br>1 Green/Red, Expansion I/O Module Status (IOS) |
| Communication Rate | 9.6K ~ 12M (1.2Km~100m) |
| Communication Speed | 9.6 ~ 12000 Kbps (Auto baudrate selection) |
| Bus Connection | 9 Pin D-Sub Connector |
| Module Location | Starter module left side of M-Series system |
| Field Power Detection | About 15Vdc |
| **General specification (Supplied by M7001)** | |
| UL System Power | Supply voltage : 24Vdc nominal, Class 2 |
| System Power | Supply voltage : 24Vdc nominal<br>Supply voltage range : 15~28.8Vdc<br>Protection : Output current limit, Reverse polarity protection |
| Power Dissipation | 80mA typical @ 24Vdc |
| Current for I/O Module | 2.0A @ 5Vdc<br>(If except for NA, current for I/O module is about 1.7A) |
| Isolation | System power to internal logic : Non-isolation<br>System power I/O driver : Isolation |
| UL Field Power | Supply voltage : 24Vdc nominal, Class 2 |
| Field Power | Supply voltage : 24Vdc typical (Max. 28.8Vdc)<br>* Field Power Range is different depending on IO Module series.<br>Refer to IO Module`s Specification. |
| Max. Current Field Power Contact | DC 10A Max |
| Single Wire | 0.205㎟ - 1.3㎟ (24-16 AWG) |
| Torque | 0.8Nm(7 lb-in) |
| Weight | 179g |
| Module Size | 54mm x 110mm x 75mm |
| Environment Condition | Refer to ‘1. Environment Specification’ |

* Class 2, adjacent to voltage rating (30Vmax)

## 2.2. M9222 Wiring Diagram




*** Primary Power Setting (P.S pin)**
- Short the P.S pin to set one of the two M7001 as the primary power.

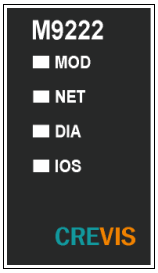
| Pin No. | | Signal Description |
|---|---|---|
| 0 | SP | System Power, 24V |
| 1 | SP | System Power, 24V |
| 2 | SN | System Power, 0V(GND) |
| 3 | SN | System Power, 0V(GND) |
| 4 | NC | ---------- |
| 5 | NC | ---------- |
| 6 | PS | Primary Power Setting |
| 7 | PS | Primary Power Setting |
| 8 | FG | F.G |
| 9 | FG | F.G |
| 10 | FN | Field Power 0V (GND) |
| 11 | FN | Field Power 0V (GND) |
| 12 | FN | Field Power 0V (GND) |
| 13 | FN | Field Power 0V (GND) |
| 14 | FP | Field Power 24V |
| 15 | FP | Field Power 24V |
| 16 | FP | Field Power 24V |
| 17 | FP | Field Power 24V |

| Series No | Through Air | Over Surface | CTI |
|---|---|---|---|
| RTB18C | 1.5mm | 1.5mm | 175≤CTI≤400 |

Spacings : The follwing minimum spcing in inches(millimeters) shall be maintained between uninsulated live parts of opposite polarity; and beween an uninsulated live part and a grounded Part including any mounting surface or exposed metal part.

## 2.3. M9222 LED Indicator

### 2.3.1. LED Indicator




| LED No. | LED Function / Description | LED Color |
|---|---|---|
| RUN | M-Bus Status | Green/Red |
| NET | Network Status | Green/Red |
| DIA | Diagnostic Status | Red |
| IOS | Expansion Module Status | Green/Red |

### 2.3.2. MOD (Module Status LED)

| Status | LED | To indicate |
|---|---|---|
| Not Powered | OFF | Power is not supplied to the unit. |
| Device Operational | Green | The unit is operating in normal condition. |
| Device in Standby | Flashing Green | The EEPROM parameter is not initialized yet.<br>Serial Number is zero value (0x00000000) |
| Minor Fault | Flashing Red | The unit has occurred recoverable fault in self-testing.<br>- EEPROM checksum fault. |
| Unrecoverable Fault | Red | The device has an unrecoverable fault.<br>- Memory error or CPU watchdog error. |

### 2.3.3. NET (Network Status LED)

| Status | LED | To indicate |
|---|---|---|
| Not Powered<br>Not On-line | OFF | Device is not on-line or may not be powered |
| On-line,<br>Not connected | Flashing<br>Green | Device is on-line but has no connections in the established state.<br>- Not allocated to a master |
| On-line,<br>Connected | Green | Device is on-line and allocated to a master |
| Connection Time-out | Flashing Red | One or more I/O connections are in the time-out state. |
| Critical Communication Failure | Red | Failed communication |

### 2.3.4. DIA(Diagnostic Status LED)

| Status | LED | To indicate |
|---|---|---|
| Hardware Error | Flashing Red | Device has hardware checking error.<br>(with MOD led is red.) |
| Expansion Module Error | Flashing Red | Device has expansion module error.<br>(with IOS led is red.) |
| IO Configuration Error | Flashing Red | Failed to initialize expansion module<br>- Overflow Input/Output size. (244bytes / 244bytes)<br>- Overflow Configuration data size. (244bytes / 244bytes)<br>- Too many expansion module. (Max 63 slot)<br>- Mismatch vendor code between adapter and expansion module. |

### 2.3.5. IOS LED (Extension Module Status LED)

| Status | LED | To indicate |
|---|---|---|
| Not Powered | OFF | Device may not be powered. |
| On-line,<br>Do not Exchanging I/O | Flashing<br>Green | I/O Communication is normal but does not exchanging I/O data.<br>(Passed the expansion module configuration) |

| | | |
|---|---|---|
| Connection,<br>Run Exchanging I/O | Green | Exchanging I/O data. |
| Connection Fault during Exchanging I/O | Red | One or more expansion module occurred in fault state.<br>- Changed expansion module configuration.<br>- Communication failure.<br>- Mismatch vendor code between adapter and expansion module. |
| Expansion Configuration Failed | Flashing Red | Failed to initialize expansion module.<br>- Detect invalid expansion module ID.<br>- Overflow Input/Output size. (244bytes / 244bytes)<br>- Too many expansion module.<br>- Initial protocol failure.<br>- If Hotswap function is enable, configured module is incorrect. |

## 2.4. M7001 LED Indicator

### 2.4.1. LED Indicator


| ED No. | LED Function / Description | LED Color |
|---|---|---|
| RUN | M-Bus Status | Green |
| PRI | Primary Status | Green |
| ACT | Active | Green |
| Field Power | Field Power Enable | Green |

### 2.4.2. RUN(RUN Status LED)

| Status | LED | To indicate |
|---|---|---|
| Main Power Module | Green On | When the Power Module is operating in main operation. |
| Substitution Power Module | Green Off | Standby with Substitution Power Module. |

### 2.4.3. PRI(Primary Status LED)

| Status | LED | To indicate |
|---|---|---|
| Main Power Module | Green On | When the Power Module is operating in main operation. |
| Substitution Power Module | Green Off | Standby with Substitution Power Module. |

### 2.4.4. ACT(Active Status LED)

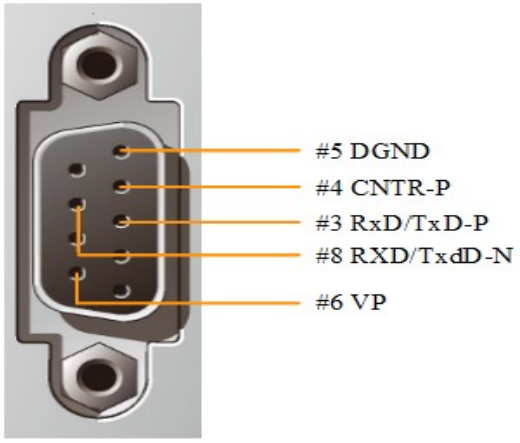
| Status | LED | To indicate |
|---|---|---|
| Main Power Module | Green On | When the Power Module is operating in main operation. |
| Substitution Power Module | Green Off | Standby with Substitution Power Module. |

### 2.4.5. Field Power LED (Field Power Status LED)

| Status | LED | To indicate |
|---|---|---|
| No field power | OFF | Not supplied 24Vdc field power. |
| Supplied field power | Green | Supplied 24Vdc field power. |

## 2.5. M9222 Electrical Interface

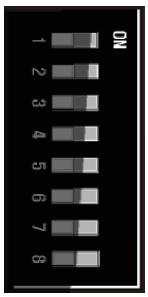
### 2.5.1. PROFIBUS Connector




| Pin No. | Description |
|---|---|
| 3 | RxD / TxD-P |
| 4 | CNTR-P |
| 5 | DGND |
| 6 | VP |
| 8 | RXD / TxdD-N |

### 2.5.2. Dip Switch

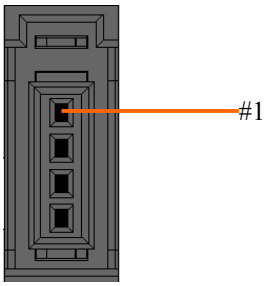

| Terminating Resistance | 1 | 2 |
|---|---|---|
| Applied | On | On |
| Not applied | Off | Off |




| Node ID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | On | Off | Off | Off | Off | Off | Off | Off |
| 2 | Off | On | Off | Off | Off | Off | Off | Off |
| ~ | | | | | | | | |
| 125 | On | Off | On | On | On | On | On | Off |

1) If switch value is 0, change the node ID is 1.

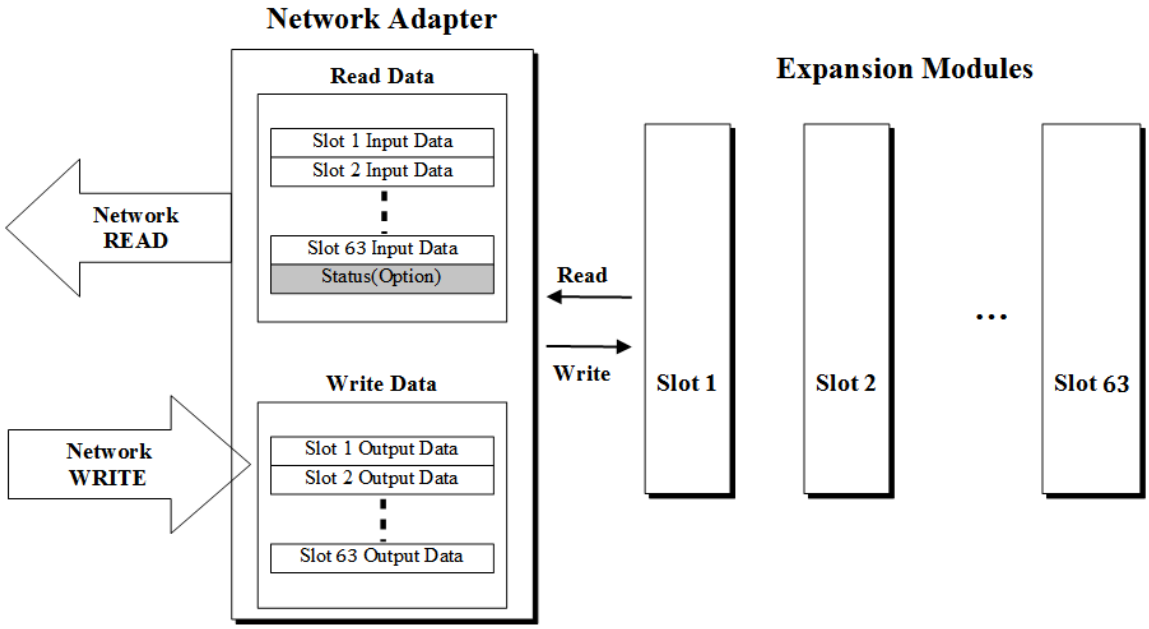
### 2.5.3. RS232 Port for MODBUS/RTU, Touch Panel or IO-Guide




| Pin# | Signal Name | Description |
|---|---|---|
| 1 | Reserved | ---- |
| 2 | TXD | RS232 TXD |
| 3 | RXD | RS232 RXD |
| 4 | GND | RS232 GND |

## 2.6. Process Image Map

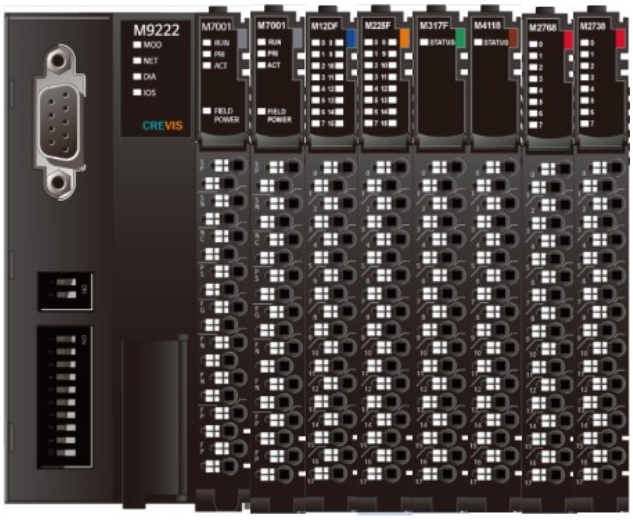
An expansion module may have 3 types of data as I/O data, configuration parameter and memory register.
The data exchange between network adapter and expansion modules is done via an I/O process image data by M-Series protocol. The following figure shows the data flow of process image between network adapter and expansion modules.



### 2.6.1. Example of Input and Output Process Image Map

Input image data depends on slot position and expansion slot data type. Input process image data is only ordered by expansion slot position

- **For example slot configuration**




| Slot No. | Module Description |
|---|---|
| #0 | MODBUS/TCP Adapter |
| #1 | Power Input |
| #2 | Power Input |
| #3 | 16-discrete input |
| #4 | 16-discrete output |
| #5 | 16-analog input |
| #6 | 8-analog output |
| #7 | 8-discrete output |
| #8 | 8-discrete output |

- **Input Process Image**

| Byte | Bit 7 | Bit 6 | Bti 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0 | Power module (Slot#1) | | | | | | | |
| 1 | Power module (Slot#2) | | | | | | | |
| 2 | Analog Input Ch0 low byte (Slot#5) | | | | | | | |
| 3 | Analog Input Ch0 high byte (Slot#5) | | | | | | | |
| 4 | Analog Input Ch1 low byte (Slot#5) | | | | | | | |
| 5 | Analog Input Ch1 high byte (Slot#5) | | | | | | | |
| ... | ... | | | | | | | |
| ... | ... | | | | | | | |
| 30 | Analog Input Ch14 low byte (Slot#5) | | | | | | | |
| 31 | Analog Input Ch14 high byte (Slot#5) | | | | | | | |
| 32 | Analog Input Ch15 low byte (Slot#5) | | | | | | | |
| 33 | Analog Input Ch15 high byte (Slot#5) | | | | | | | |

- **Output Process Image**

| Byte | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0 | Discrete Output 8 pts (Slot#1) | | | | | | | |
| 1 | Discrete Output 8 pts (Slot#2) | | | | | | | |
| 2 | Analog Output Ch0 low byte (Slot#6) | | | | | | | |
| 3 | Analog Output Ch0 high byte (Slot#6) | | | | | | | |
| 4 | Analog Output Ch1 low byte (Slot#6) | | | | | | | |
| 5 | Analog Output Ch2 high byte (Slot#6) | | | | | | | |
| ... | ... | | | | | | | |
| ... | ... | | | | | | | |
| 14 | Analog Output Ch6 low byte (Slot#6) | | | | | | | |
| 15 | Analog Output Ch6 high byte (Slot#6) | | | | | | | |
| 16 | Analog Output Ch7 low byte (Slot#6) | | | | | | | |
| 17 | Analog Output Ch7 high byte (Slot#6) | | | | | | | |

# 3. Parameter

## 3.1. M9222

- ☐ Parameter length: 3 bytes
- ☐ Parameter Data:

| Offset | Access | Decimal Bit | Description | Default Value |
|---|---|---|---|---|
| 3 | R/W | 00-01 | Word Data Format<br>0: Little-Endian(INTEL)<br>1: Big-Endian (MOTOROLA) | 1 (Motorola) |
| | | 02-07 | Reserved | 0 |
| 4 | R/W | 00-01 | Hot swap<br>0: Hotswap enable<br>0: Hotswap disable | 0 |
| | | 02-07 | Reserved | 0 |
| 5 | R/W | 00-01 | PROFIBUS Disconnection<br>0: Fault values are switched<br>1: Hold Last states | 0 |
| | | 02-07 | Reserved | 0 |

# 4. DPV1 Service

## 4.1. MSAC1 Read(PROFIBUS-DP Extensions to EN50170)

- ■ **MSAC1 Read request**

| Parameter | Description |
|---|---|
| Remote Address | Slave Address(0~99) |
| Slot Number | Slot Number : 0(M9222) |
| Index | 253 : FW revision (Data size : 4 bytes)<br>254 : Vendor code (Data size : 1 byte) |
| Length | 1~128 |

- ■ **MSAC1 Read Confirm(+)**

| Parameter | Description |
|---|---|
| Remote Address | Slave Address(0~99) |
| Length | 1~128 |
| Data | User Data |

- ■ **MSAC1 Read Confirm (-)**

| Parameter | Description |
|---|---|
| Remote Address | Slave Address(0~99) |
| Error Decode | - |
| Error_code 1 | - |
| Error_code 2 | Reserved |

## 4.2. MSAC1 Write(PROFIBUS-DP Extensions to EN50170)

- ■ **MSAC1 Write request**

| Parameter | Description |
|---|---|
| Remote Address | Slave Address(0~99) |
| Slot Number | Slot Number : 0(M9222) |
| Index | 254 : Vendor code (Data size : 5 bytes) |
| Length | 1~128 |
| Data | Vendor(**Don't mention this to the User manual**)<br>- Data[0] : 0xAE<br>- Data[1] : 0xBE |

| | |
|---|---|
| | - Data[2] : 0xCE<br>- Data[3] : 0xDE<br>- Vendor code |

■ **MSAC1 Write Confirm(+)**

| Parameter | Description |
|---|---|
| Remote Address | Slave Address(0~99) |
| Length | 1~128 |

■ **MSAC1 Write Confirm (-)**

| Parameter | Description |
|---|---|
| Remote Address | Slave Address(0~99) |
| Error Decode | - |
| Error_code 1 | - |
| Error_code 2 | Reserved |

| Parameter | Description |
|---|---|
| C_Ref | Communication-Reference |
| Error Decode | - |
| Error_code 1 | - |
| Error_code 2 | Reserved |

## 4.3. Error_Decode (PROFIBUS-DP Extensions to EN50170)

► 0~127 : Reserved
► 128 : DPV1
► 129 ~253 : Reserved
► 254 : FMS
► 255 : HART

## 4.4. Error Code_1 (PROFIBUS-DP Extensions to EN50170)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | ● Error Class<br>✓ 0xA : Application class | | | | ● Error code<br>0 : Read Error<br>1 : Write Error<br>2 : Module Failure<br>3 ~7 : Reserved<br>8 : Version conflict<br>9 : Feature not supported<br>10~15 : User specific | | | |
| 2 | ● Error Class<br>✓ 0xB : Access class | | | | ● Error code<br>0 : Invalid index<br>1 : Write length error<br>2 : Invalid slot<br>3 : Type conflict<br>4 : Invalid area<br>5 : state conflict<br>6 : access denied<br>7 : invalid range<br>8 : invalid parameter<br>9 : invalid type<br>10~15 : User specific | | | |
| 3 | ● Error Class<br>✓ 0xC : Resource class | | | | ● Error code<br>0 : read constrain conflict<br>1 : Write constrain conflict<br>2 : Resource busy | | | |

| | | |
|---|---|---|
| | | 3 : Resource unavailable<br>4 ~7 : Reserved<br>8~15 : User specific |
| 4 | ● Error Class<br>✓ 0xD : M9222 Specific Class | ● Error code<br>1 : Slot Parameter write error<br>2 : Read memory error<br>3 : Write memory error |

# 5. Diagnostics

| Byte | Item | Description |
|---|---|---|
| 0 | Station status 1 | PROFIBUS Standard Diagnostic |
| 1 | Station status 2 | |
| 2 | Station status 3 | |
| 3 | Master Address | |
| 4 | PNO Ident Number High | |
| 5 | PNO Ident Number Low | |

- **Station Status 1~3**

| **Station status** | | | |
|---|---|---|---|
| 1 | Bit 7 | Master_Lock | Slave is parameterized by another master |
| | Bit 6 | Prm_Fault | Last parameter telegram faulty |
| | Bit 5 | Inv._Sl_Res. | Implausible response of the slave |
| | Bit 4 | Not_Supp. | Unknown command detected by the slave |
| | Bit 3 | Ext_Diag | The area Ext_Diag is used for extended diagnostic |
| | Bit 2 | Cfg_Fault | Slave is wrong parameterized |
| | Bit 1 | Sta._Not_Rdy | Slave not ready |
| | Bit 0 | Sta._Non_Exist. | Slave not responding |
| 2 | Bit 7 | Deactivated | Slave not projected |
| | Bit 6 | Reserved | Reserved |
| | Bit 5 | Sync_Mode | Sync-command active |
| | Bit 4 | Freeze_Mode | Freeze-command active |
| | Bit 3 | WD_On | Watchdog activated |
| | Bit 2 | 1 | Always 1 |
| | Bit 1 | Stat_Diag | Get diagnostic from slave, till bit is released |
| | Bit 0 | Prm_ | Slave must be parameterized |
| 3 | Bit 7 | Ext_Diag_Ovfl. | The slave has more diagnostic data available than it can send |
| - | Bit 6~0 | Reserved | Reserved |

# 6. MODBUS INTERFACE

## 6.1. MODBUS Interface Register/Bit Map

- **Register Map**

| Start Address | Read/Write | Description | Func. Code |
|---|---|---|---|
| 0x0000 ~ | Read | Process input image registers (Real Input Register) | 3,4,23 |
| 0x0800 ~ | Read/Write | Process output image registers (Real Output Register) | 3,16,23 |
| | | | |
| 0x1000 * | Read | Adapter Identification special registers. | 3,4,23 |
| 0x1020 * | Read/Write | Adapter Watchdog, other time special register. | 3,4,6,16,23 |
| 0x1100 * | Read/Write | Adapter Information special registers. | 3,4,6,16,23 |
| 0x2000 * | Read/Write | Expansion Slot Information special registers. | 3,4,6,16,23 |

* The special register map must be accessed by read/write of every each address (one address).

- **Register Map**

| Start Address | Read/Write | Description | Func. Code |
|---|---|---|---|
| 0x0000~ | Read | Process input image bits<br>All input registers area are addressable by bit address.<br>Size of input image bit is size of input image register * 16. | 2 |
| 0x1000~ | Read/Write | Process output image bits<br>All output registers area are addressable by bit address.<br>Size of output image bit is size of output image register * 16. | 1,5,15 |

## 6.2. Supported MODBUS Function Codes

| Function Code | Function | Description |
|---|---|---|
| 1(0x01) | Read Coils | Read output bit |
| 2(0x02) | Read Discrete Inputs | Read input bit |
| 3(0x03) | Read Holding Registers | Read output word |
| 4(0x04) | Read Input Registers | Read input word |
| 5(0x05) | Write Single Coil | Write one bit output |
| 6(0x06) | Write Single Register | Write one word output |
| 8(0x08) | Diagnostics | Read diagnostic register |
| 15(0x0F) | Write Multiple Coils | Write a number of output bits |
| 16(0x10) | Write Multiple registers | Write a number of output words |
| 23(0x17) | Read/Write Multiple registers | Read a number of input words /Write a number of output words |

- Refer to MODBUS APPLICATION PROTOCOL SPECIFICATION V1.1a

## 6.3. MODBUS Transmission Mode

Two different serial transmission modes are defined : The RTU mode and the ASCII mode. It defines the bit contents of message fields transmitted serially on the line. It determines how information is packed into the message fields and decoded.

### 6.3.1. RTU Transmission Mode

When devices communicate on a MODBUS serial line using the RTU (Remote Terminal Unit) mode, each 8–bit byte in a message contains two 4–bit hexadecimal characters. The main advantage of this mode is that its greater character density allows better data throughput than ASCII mode for the same baud rate. Each message must be transmitted in a continuous stream of characters.

| Start | Address | Function | Data | CRC Check | End |
|---|---|---|---|---|---|
| ≥ 3.5 char | 1 char | 1 char | Up to 252 chars(s) | 2 chars | ≥ 3.5 char |

### 6.3.2. ASCII Transmission Mode

When devices are setup to communicate on a MODBUS serial line using ASCII (American Standard Code for Information Interchange) mode, each 8–bit byte in a message is sent as two ASCII characters. This mode is used when the physical communication link or the capabilities of the device does not allow the conformance with RTU mode requirements regarding timers management.

| Start | Address | Function | Data | LRC Check | End |
|---|---|---|---|---|---|
| 1 char<br>‘:’ | 2 chars | 2 chars | Up to 2x252 char(s) | 2 chars | 2 chars<br>CR,LF |

### 6.3.3. 1 (0x01) Read Coils

This function code is used to read from 1 to 2000 contiguous status of coils in a remote device. The Request PDU specifies the starting address, i.e. the address of the first coil specified, and the number of coils. In the PDU Coils are addressed starting at zero. Therefore coils numbered 1-16 are addressed as 0-15. The coils in the response message are packed as one coil per bit of the data field. Status is indicated as 1= ON and 0= OFF.

- **Request**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | - | t1-t2-t3 | “:” | 0x3A |
| Slave Address | 0x63 | 0x63 | “63” | 0x36, 0x33 |
| Function Code | 0x01 | 0x01 | “01” | 0x30, 0x31 |
| Starting Address Hi | 0x10 | 0x10 | “10” | 0x31, 0x30 |
| Starting Address Lo | 0x00 | 0x00 | “00” | 0x30, 0x30 |
| Quantity of Outputs Hi | 0x00 | 0x00 | “00” | 0x30, 0x30 |
| Quantity of Outputs Lo | 0x10 | 0x10 | “10” | 0x31, 0x30 |
| Error Check (CRC/LRC) | - | 0x31, 0x44 | “7C” | 0x37, 0x43 |
| End of Frame | - | t1-t2-t3 | CR, LF | 0x0D, 0x0A |

- **Response**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | “:” | 0x3A |
| Slave Address | 0x63 | 0x63 | “63” | 0x36, 0x33 |
| Function Code | 0x01 | 0x01 | “01” | 0x30, 0x31 |
| Byte Count | 0x02 | 0x02 | “02” | 0x30, 0x32 |
| Output Status | 0x00 | 0x00 | “00” | 0x30, 0x30 |
| Output Status | 0x00 | 0x00 | “00” | 0x30, 0x30 |
| Error Check (CRC/LRC) | --- | 0x40, 0x34 | “9A” | 0x39, 0x41 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0x0A |

### 6.3.4. 2 (0x02) Read Discrete Inputs

This function code is used to read from 1 to 2000 contiguous status of discrete inputs in a remote device. The Request PDU specifies the starting address, i.e. the address of the first input specified, and the number of inputs. In the PDU Discrete Inputs are addressed starting at zero. Therefore Discrete inputs numbered 1-16 are addressed as 0-15.
The discrete inputs in the response message are packed as one input per bit of the data field.
Status is indicated as 1= ON; 0= OFF.

- **Request**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "07" | 0x36, 0x33 |
| Function Code | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Starting Address Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Starting Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Inputs Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Inputs Lo | 0x10 | 0x10 | "0A" | 0x31, 0x30 |
| Error Check (CRC/LRC) | --- | 0x71, 0x84 | "ED" | 0x38, 0x42 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- **Response**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Byte Count | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Input Status | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Input Status | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Error Check (CRC/LRC) | --- | 0x40, 0x70 | "99" | 0x39, 0x39 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

### 6.3.5. 3 (0x03) Read Holding Registers

This function code is used to read the contents of a contiguous block of holding registers in a remote device. The Request PDU specifies the starting register address and the number of registers.
The register data in the response message are packed as two bytes per register, with the binary contents right justified within each byte. For each register, the first byte contains the high order bits and the second contains the low order bits.

- **Request**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x03 | 0x03 | "03" | 0x30, 0x33 |
| Starting Address Hi | 0x10 | 0x10 | "10" | 0x31, 0x30 |
| Starting Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Register Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Register Lo | 0x01 | 0x01 | "01" | 0x30, 0x31 |
| Error Check (CRC/LRC) | --- | 0x88, 0x88 | "89" | 0x38, 0x39 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0x0A |

- **Response**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x03 | 0x03 | "03" | 0x30, 0x33 |
| Byte Count | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Output Register#0 Hi | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Output Register#0 Lo | 0xE5 | 0xE5 | "E5" | 0x45, 0x35 |
| Error Check (CRC/LRC) | --- | 0x81, 0x67 | "B1" | 0x42, 0x31 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0x0A |

- In case of address 0x0800, 0x0801 output register value: 0x1122, 0x3344.

### 6.3.6. 4 (0x04) Read Input Registers

This function code is used to read from 1 to approx. 125 contiguous input registers in a remote device. The Request PDU specifies the starting register address and the number of registers. The register data in the response message are packed as two bytes per register, with the binary contents right justified within each byte. For each register, the first byte contains the high order bits and the second contains the low order bits.

- **Request**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x04 | 0x04 | "04" | 0x30, 0x34 |
| Starting Address Hi | 0x10 | 0x10 | "10" | 0x31, 0x30 |
| Starting Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Register Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Register Lo | 0x01 | 0x01 | "01" | 0x30, 0x31 |
| Error Check (CRC/LRC) | --- | 0x3D, 0x48 | "88" | 0x38, 0x38 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0x0A |

- **Response**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x04 | 0x04 | "04" | 0x30, 0x34 |
| Byte Count | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Input Register#0 Hi | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Input Register#0 Lo | 0xE5 | 0xE5 | "E5" | 0x45, 0x35 |
| Error Check (CRC/LRC) | --- | 0x80, 0x13 | "B0" | 0x42, 0x30 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- In case of address 0x0000, 0x0001 input register value: 0x0080, 0x0000.

### 6.3.7. 5 (0x05) Write Single Coil

This function code is used to write a single output to either ON or OFF in a remote device. The requested ON/OFF state is specified by a constant in the request data field. A value of FF 00 hex requests the output to be ON. A value of 00 00 requests it to be OFF. All other values are illegal and will not affect the output.

- **Request**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "36" | 0x36, 0x33 |
| Function Code | 0x05 | 0x05 | "05" | 0x30, 0x35 |
| Output Address Hi | 0x10 | 0x10 | "10" | 0x31, 0x30 |
| Output Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Output Value Hi | 0xFF | 0xFF | "FF" | 0x46, 0x46 |
| Output Value Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Error Check (CRC/LRC) | --- | 0x80, 0xB8 | "8Y" | 0x38, 0x59 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- **Response**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "36" | 0x36, 0x33 |
| Function Code | 0x05 | 0x05 | "05" | 0x30, 0x35 |
| Output Address Hi | 0x10 | 0x10 | "10" | 0x31, 0x30 |
| Output Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Output Value Hi | 0xFF | 0xFF | "FF" | 0x46, 0x46 |
| Output Value Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Error Check (CRC/LRC) | --- | 0x80, 0xB8 | "8Y" | 0x38, 0x59 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- Output bit of address 0x1001 turns ON.

### 6.3.8. 6 (0x06) Write Single Register

This function code is used to write a single holding register in a remote device. Therefore register numbered 1 is addressed as 0. The normal response is an echo of the request, returned after the register contents have been written.

- **Request**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x06 | 0x06 | "06" | 0x30, 0x36 |
| Register Address Hi | 0x08 | 0x08 | "08" | 0x30, 0x38 |
| Register Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Register Value Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Register Value Lo | 0xFF | 0xFF | "FF" | 0x46, 0x46 |
| Error Check (CRC/LRC) | --- | 0xC3, 0xA8 | "90" | 0x39, 0x30 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- **Response**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x06 | 0x06 | "06" | 0x30, 0x36 |
| Register Address Hi | 0x08 | 0x08 | "08" | 0x30, 0x38 |
| Register Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Register Value Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Register Value Lo | 0xFF | 0xFF | "FF" | 0x46, 0x46 |
| Error Check (CRC/LRC) | --- | 0xC3, 0xA8 | "90" | 0x39, 0x30 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- In case of address 0x0800 output register value: 0x0000 changes to 0x1122.

### 6.3.9. 8 (0x08) Diagnostics

MODBUS function code 08 provides a series of tests for checking the communication system between a client ( Master) device and a server ( Slave), or for checking various internal error conditions within a server.
The function uses a two–byte sub-function code field in the query to define the type of test to be performed. The server echoes both the function code and sub-function code in a normal response. Some of the diagnostics cause data to be returned from the remote device in the data field of a normal response.

- **Request**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x07 | 0x07 | "07" | 0x30, 0x37 |
| Function Code | 0x08 | 0x08 | "08" | 0x30, 0x38 |
| Sub-Function Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Sub-Function Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Data Hi | 0x11 | 0x11 | "11" | 0x31, 0x31 |
| Data Lo | 0x22 | 0x22 | "22" | 0x32, 0x32 |
| Error Check (CRC/LRC) | --- | 0x6C, 0x24 | "BE" | 0x42, 0x45 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- **Response**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x07 | 0x07 | "07" | 0x30, 0x37 |
| Function Code | 0x08 | 0x08 | "08" | 0x30, 0x38 |
| Sub-Function Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Sub-Function Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Data Hi | 0x11 | 0x11 | "11" | 0x31, 0x31 |
| Data Lo | 0x22 | 0x22 | "22" | 0x32, 0x32 |
| Error Check (CRC/LRC) | --- | 0x6C, 0x24 | "BE" | 0x42, 0x45 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

**Sub-function 0x0000(0) Return Query Data**
The data passed in the request data field is to be returned (looped back) in the response.
The entire response message should be identical to the request.

| Sub-function | Data Field (Request) | Data Field (Response) | Description |
|---|---|---|---|
| 0x0000(0) | Any | Echo Request Data | |

**Sub-function 0x0001(1) Restart Communications Option**
The remote device could be initialized and restarted, and all of its communications event counters are cleared.
Especially, data field 0x55AA make the remote device to restart with factory default setup of EEPROM.

| Sub-function | Data Field (Request) | Data Field (Response) | Description |
|---|---|---|---|
| 0x0001(1) | 0x0000 or 0xFF00 | Echo Request Data | Reset |
| 0x0001(1) | 0x55AA+0xAB7B+Sumcheck | Echo Request Data | Reset with Factory default 1) |
| 0x0001(1) | 0x55AA+0xAA55+Sumcheck | Echo Request Data | Reset with Factory default 2) |

1),2) Watchdog time value, auto recovery will be the factory defaults value.
2) Mac Address will be the factory default value. This module is not using Mac address.

**Sub-function 0x000A(10) Clear Counters and Diagnostic Register**
The goal is to clear all counters and the diagnostic register. Counters are also cleared upon power–up.

| Sub-function | Data Field (Request) | Data Field (Response) | Description |
|---|---|---|---|
| 0x000A(10) | 0x0000 | Echo Request Data | |

**Sub-function 0x000B(11) Return Bus Message Count**
The response data field returns the quantity of messages that the remote device has detected on the communications system since its last restart, clear counters operation, or power–up.

| Sub-function | Data Field (Request) | Data Field (Response) | Description |
|---|---|---|---|
| 0x000B(11) | 0x0000 | Total Message Count | |

**Sub-function 0x000C(12) Return Bus Communication Error Count**
The response data field returns the quantity of CRC errors encountered by the remote device since its last restart, clear counters operation, or power–up.

| Sub-function | Data Field (Request) | Data Field (Response) | Description |
|---|---|---|---|
| 0x000C(12) | 0x0000 | CRC Error Count | |

**Sub-function 0x000D(13) Return Bus Exception Error Count**
The response data field returns the quantity of MODBUS exception responses returned by the remote device since its last restart, clear counters operation, or power–up.
Exception responses are described and listed in section 3.2.11.

| Sub-function | Data Field (Request) | Data Field (Response) | Description |
|---|---|---|---|
| 0x000D(13) | 0x0000 | Exception Error Count | |

**Sub-function 0x000E(14) Return Slave Message Count**
The response data field returns the quantity of messages addressed to the remote device, or broadcast, that the remote device has processed since its last restart, clear counters operation, or power–up.

| Sub-function | Data Field (Request) | Data Field (Response) | Description |
|---|---|---|---|
| 0x000E(14) | 0x0000 | Slave Message Count | |

**Sub-function 0x000F(15) Return Slave No Response Count**
The response data field returns the quantity of messages addressed to the remote device for which it has returned no response (neither a normal response nor an exception response), since its last restart, clear counters operation, or power–up.

| Sub-function | Data Field (Request) | Data Field (Response) | Description |
|---|---|---|---|
| 0x000F(15) | 0x0000 | Slave No Response Count | |

**Sub-function 0x0064(100) Return Slave ModBus, Internal Bus Status**
The response data field returns the status of ModBus and Internal Bus addressed to the remote device.
This status values are identical with status 1word of input process image.

| Sub-function | Data Field (Request) | Data Field (Response) | Description |
|---|---|---|---|
| 0x0064(100) | 0x0000 | ModBus, Internal Bus Status | Same as status 1word |

**Sub-function 0x0065(101) Return Slave Watchdog Error Count**

The response data field returns the quantity of watchdog error addressed to the remote device since its last restart, clear counters operation, or power–up.

| Sub-function | Data Field (Request) | Data Field (Response) | Description |
|---|---|---|---|
| 0x0065(101) | 0x0000 | Watchdog Error Count | |

### 6.3.10. 15 (0x0F) Write Multiple Coils

This function code is used to force each coil in a sequence of coils to either ON or OFF in a remote device. The Request PDU specifies the coil references to be forced. Coils are addressed starting at zero. A logical '1' in a bit position of the field requests the corresponding output to be ON. A logical '0' requests it to be OFF.
The normal response returns the function code, starting address, and quantity of coils forced.

- **Request**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x0F | 0x0F | "0F" | 0x30, 0x46 |
| Starting Address Hi | 0x10 | 0x10 | "10" | 0x31, 0x30 |
| Starting Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Outputs Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Outputs Lo | 0x10 | 0x10 | "10" | 0x31, 0x30 |
| Byte Count | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Output Value#0 | 0x0F | 0x0F | "0F" | 0x30, 0x46 |
| Output Value#1 | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Error Check (CRC/LRC) | --- | 0x47, 0x73 | "5D" | 0x35, 0x44 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- **Response**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x0F | 0x0F | "0F" | 0x30, 0x46 |
| Starting Address Hi | 0x10 | 0x10 | "10" | 0x31, 0x30 |
| Starting Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Outputs Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Outputs Lo | 0x10 | 0x10 | "10" | 0x31, 0x30 |
| Error Check (CRC/LRC) | --- | 0x58, 0x85 | "6E" | 0x36, 0x45 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- In case of address 0x1015~0x1000 output bit value: 00000000_00000000 changes to 00000001_01010101.

### 6.3.11. 16 (0x10) Write Multiple Registers

This function code is used to write a block of contiguous registers (1 to approx. 120 registers) in a remote device.
The requested written values are specified in the request data field. Data is packed as two bytes per register.
The normal response returns the function code, starting address, and quantity of registers written.

- **Request**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x10 | 0x10 | "10" | 0x31, 0x30 |
| Starting Address Hi | 0x08 | 0x08 | "08" | 0x30, 0x38 |
| Starting Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Registers Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Registers Lo | 0x01 | 0x01 | "01" | 0x30, 0x31 |
| Byte Count | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Register Value#0 Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Register Value#0 Lo | 0xFF | 0xFF | "FF" | 0x46, 0x46 |
| Error Check (CRC/LRC) | --- | 0xDE, 0xB2 | "83" | 0x38, 0x33 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- **Response**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x10 | 0x10 | "10" | 0x31, 0x30 |
| Starting Address Hi | 0x08 | 0x08 | "08" | 0x30, 0x38 |
| Starting Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Registers Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Registers Lo | 0x01 | 0x01 | "01" | 0x30, 0x31 |
| Error Check (CRC/LRC) | --- | 0x0B, 0xEB | "84" | 0x38, 0x34 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- In case of address 0x0800, 0x0801 output register value: 0x0000, 0x0000 changes to 0x1122, 0x3344.

### 6.3.12. 23 (0x17) Read/Write Multiple Registers

This function code performs a combination of one read operation and one write operation in a single MODBUS transaction. The write operation is performed before the read. The request specifies the starting address and number of holding registers to be read as well as the starting address, number of holding registers, and the data to be written. The byte count specifies the number of bytes to follow in the write data field.
The normal response contains the data from the group of registers that were read. The byte count field specifies the quantity of bytes to follow in the read data field.

- **Request**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x17 | 0x17 | "17" | 0x31, 0x37 |
| Read Starting Address Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Read Starting Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Read Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Read Lo | 0x01 | 0x01 | "01" | 0x30, 0x31 |
| Write Starting Address Hi | 0x08 | 0x08 | "08" | 0x30, 0x38 |
| Write Starting Address Lo | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Write Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Quantity of Write Lo | 0x01 | 0x01 | "01" | 0x30, 0x31 |
| Byte Count | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Write Reg. Value#0 Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Write Reg. Value#0 Lo | 0xFF | 0xFF | "FF" | 0x46, 0x46 |
| Error Check (CRC/LRC) | --- | 0x1B, 0xCC | "7B" | 0x37, 0x42 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0x0A |

- **Response**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x63 | 0x63 | "63" | 0x36, 0x33 |
| Function Code | 0x17 | 0x17 | "17" | 0x31, 0x37 |
| Byte Count | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Read Reg. Value#0 Hi | 0x00 | 0x00 | "00" | 0x30, 0x30 |
| Read Reg. Value#0 Lo | 0xFF | 0xFF | "FF" | 0x46, 0x46 |
| Error Check (CRC/LRC) | --- | 0x04, 0x3C | "85" | 0x38, 0x35 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- In case of address 0x0800, 0x0801 output register value: 0x0000, 0x0000 changes to 0x1122, 0x3344.

### 6.3.13. Error Response

In an exception response, the server sets the MSB of the function code to 1. This makes the function code value in an exception response exactly 80 hexadecimal higher than the value would be for a normal response.

- **Exception Response Example**

| Field name | Example | RTU | ASCII | ASCII (bus line) |
|---|---|---|---|---|
| Start of Frame | --- | t1-t2-t3 | ":" | 0x3A |
| Slave Address | 0x07 | 0x07 | "07" | 0x30, 0x37 |
| Function Code | 0x81 | 0x81 | "81" | 0x38, 0x31 |
| Exception Code | 0x02 | 0x02 | "02" | 0x30, 0x32 |
| Error Check (CRC/LRC) | --- | 0x22, 0xC0 | "76" | 0x37, 0x36 |
| End of Frame | --- | t1-t2-t3 | CR, LF | 0x0D, 0xA |

- **Exception Codes**

| Exception Code | Name | Description |
|---|---|---|
| 01 | Illegal Function | The function code received in the query is not an allowable action for the server (or slave). |
| 02 | Illegal Data Address | The data address received in the query is not an allowable address for the server (or slave). |
| 03 | Illegal Data Value | A value contained in the query data field is not an allowable value for server (or slave). |
| 04 | Slave Device Failure | An unrecoverable error occurred while the server (or slave) was attempting to perform the requested action. |
| 05 | Acknowledge | The server (or slave) has accepted the request and is processing it, but a long duration of time will be required to do so. |
| 06 | Slave Device Busy | Specialized use in conjunction with programming commands.<br>The server (or slave) is engaged in processing a long–duration program command. The client (or master) should retransmit the message later when the server (or slave) is free. |
| 08 | Memory Parity Error | The server (or slave) attempted to read record file, but detected a parity error in the memory. The client (or master) can retry the request, but service may be required on the server (or slave) device. |
| 0A | Gateway Path Unavailable | Specialized use in conjunction with gateways, indicates that the gateway was unable to allocate an internal communication path from the input port to the output port for processing the request. |

## 6.4. MODBUS Special Register Map

The special register map can be accessed by function code 3, 4, 6 and 16. Also the special register map must be accessed by read/write of every each address (one address).

### 6.4.1. Adapter Identification Special Register (0x1000, 4096)

| Address | Access | Type, Size | Description |
|---|---|---|---|
| 0x1000(4096) | Read | 1word | Vendor ID = 0x029D, Crevis. Co., Ltd. |
| 0x1001(4097) | Read | 1word | Device type = 0x000C, Network Adapter |
| 0x1002(4098) | Read | 1word | Product Code = 0xA040 |
| 0x1003(4099) | Read | 1word | Firmware revision, if 0x0101, revision 1.01 |
| 0x1004(4100) | Read | 2word | Product unique serial number |
| 0x1005(4101) | Read | String<br>upto 36byte | Product name string (ASCII)<br>"M9222,Profibus Adapter, MBUS" |
| 0x1006(4102) | Read | 1word | Sum check of EEPROM |
| 0x1010(4112) | Read | 2word | Firmware release date |
| 0x1011(4113) | Read | 2word | Product manufacturing inspection date |
| 0x101E(4126) | Read | 7word<br>- 1word<br>- 1word<br>- 1word<br>- 1word<br>- 1word<br>- 2word | Composite Id of following address<br>* RTU mode<br>0x1100(4352), Modbus RS232 Node. (Fixed 0x0001)<br>0x1000(4096), Vendor ID<br>0x1001(4097), Device type<br>0x1002(4098), Product code<br>0x1003(4099), Firmware revision<br>0x1004(4100), Product serial number |

- String Type consists of valid string length (first 1word) and array of characters

### 6.4.2. Adapter Hotswap Register (0x1060, 4192)

| Address | Access | Type, Size | Description |
|---|---|---|---|
| 0x1060(4192) | Read/<br>Write | 1word | Hot swap status 0 : Enable<br>1 : Disable |
| 0x1062(4194)* | Read | 1word | Error slot detection<br>0 : No error slot<br>1 : Error slot detection |
| 0x1063(4195)* | Read | 4word | Error slot location, 8x8 bit |

* 0x1062 and 0x1063 functions are only available if hot swap(0x1060) is enabled.

### 6.4.3. Adapter Information Special Register (0x1100, 4352)

<table>
<tr><th>Address</th><th>Access</th><th>Type, Size</th><th>Description</th></tr>
<tr><td>0x1102(4354)</td><td>Read</td><td>1word</td><td>Start address of input image word register. =0x0000</td></tr>
<tr><td>0x1103(4355)</td><td>Read</td><td>1word</td><td>Start address of output image word register. =0x0800</td></tr>
<tr><td>0x1104(4356)</td><td>Read</td><td>1word</td><td>Size of input image word register.</td></tr>
<tr><td>0x1105(4357)</td><td>Read</td><td>1word</td><td>Size of output image word register.</td></tr>
<tr><td>0x1106(4358)</td><td>Read</td><td>1word</td><td>Start address of input image bit. = 0x0000</td></tr>
<tr><td>0x1107(4359)</td><td>Read</td><td>1word</td><td>Start address of output image bit. =0x1000</td></tr>
<tr><td>0x1108(4360)</td><td>Read</td><td>1word</td><td>Size of input image bit.</td></tr>
<tr><td>0x1109(4361)</td><td>Read</td><td>1word</td><td>Size of output image bit.</td></tr>
<tr><td>0x110A(4362)</td><td>Read</td><td>1word</td><td>Update time for cyclic data change (same as 0x1028)</td></tr>
<tr><td>0x110D(4365)</td><td>Read</td><td>1word</td><td>Dip switch value(MSB)<br>Dip switch value : 0 ~ 7 bit (xxxxxxxx_oooooooo)</td></tr>
<tr><td>0x110E(4366)</td><td>Read</td><td>upto<br>33word</td><td>Expansion slot’s M-number including M<br>First 1word is adapter’s number, if M9222, then 0x9222</td></tr>
<tr><td>0x1110(4368)</td><td>Read</td><td>1word</td><td>Number of expansion slot</td></tr>
<tr><td>0x1113(4371)</td><td>Read</td><td>upto<br>33word</td><td>Expansion slot Module Id.<br>First 1word is adapter’s module id.</td></tr>
<tr><td>0x1119(4377)</td><td>Read</td><td>1word</td><td>Hi byte is ModBus status, low byte is internal status.
<table>
<tr><th>ModBus Status</th><th>Internal bus status(M-Bus)</th></tr>
<tr><td>0x00 : No Error<br>0x01 : Error Dip Switch<br>0x40 : Error CRC LRC<br>0x80 : Error Watchdog</td><td>0x01 : Init State<br>0x02 : Idle State<br>0x03 : Run State<br>0x04 : Stop State<br>0x05 : Fault State<br>0x06 : Reset State<br>0x07 : CRC Error State<br>0x08 : Pause State<br>0x09 : Master Fault State<br>0x80* : At Hot swap mode<br>expansion module error</td></tr>
</table></td></tr>
<tr><td>0x111D(4381)</td><td>Read</td><td>1word</td><td>Adapter M-Series Revision.</td></tr>
</table>

* After the system is reset, the new “Set Value” action is applied.
** If the slot location is changed, set default value automatically (all expansion slot are live).

### 6.4.4. Expansion Slot Information Special Resister (0x2000, 8192)

Each expansion slot has 0x20(32) address offset and same information structure.

Slot#1 0x2000(8192)~0x201F(8223)
Slot#2 0x2020(8224)~0x203F(8255)
Slot#3 0x2040(8256)~0x205F(8287)
Slot#4 0x2060(8288)~0x207F(8319)
Slot#5 0x2080(8320)~0x209F(8351)
Slot#6 0x20A0(8352)~0x20BF(8383)
Slot#7 0x20C0(8384)~0x20DF(8415)
Slot#8 0x20E0(8416)~0x20FF(8447)
Slot#9 0x2100(8448)~0x211F(8479)
Slot#10 0x2120(8480)~0x213F(8511)
Slot#11 0x2140(8512)~0x215F(8543)
Slot#12 0x2160(8544)~0x217F(8575)
Slot#13 0x2180(8576)~0x219F(8607)
Slot#14 0x21A0(8608)~0x21BF(8639)
Slot#15 0x21C0(8640)~0x21DF(8671)
Slot#16 0x21E0(8672)~0x21FF(8703)
Slot#17 0x2200(8704)~0x221F(8735)
Slot#18 0x2220(8736)~0x223F(8767)
Slot#19 0x2240(8768)~0x225F(8799)
Slot#20 0x2260(8800)~0x227F(8831)
Slot#21 0x2280(8832)~0x229F(8863)
Slot#22 0x22A0(8864)~0x22BF(8895)
Slot#23 0x22C0(8896)~0x22DF(8927)
Slot#24 0x22E0(8928)~0x22FF(8959)
Slot#25 0x2300(8960)~0x231F(8991)
Slot#26 0x2320(8992)~0x233F(9023)
Slot#27 0x2340(9024)~0x235F(9055)
Slot#28 0x2360(9056)~0x237F(9087)
Slot#29 0x2380(9088)~0x239F(9119)
Slot#30 0x23A0(9120)~0x23BF(9151)
Slot#31 0x23C0(9152)~0x23DF(9183)
Slot#32 0x23E0(9184)~0x23FF(9215)
Slot#33 0x2400(9216)~0x241F(9247)
Slot#34 0x2420(9248)~0x243F(9279)
…..
Slot#63 0x27C0(10176)~0x27DF(10207)

| Address Offset | Expansion Slot#1 | Expansion Slot#2 | Expansion Slot#3 | Expansion Slot#4 | …….. | Expansion Slot#63 |
|---|---|---|---|---|---|---|
| + 0x00(+0) | 0x2000(8192) | 0x2020(8224) | 0x2040(8256) | 0x2060(8288) | ……. | 0x27C0(10176) |
| + 0x01(+1) | 0x2001(8193) | 0x2021(8225) | 0x2041(8257) | 0x2061(8289) | ……. | 0x27C1(10177) |
| + 0x02(+2) | 0x2002(8194) | 0x2022(8226) | 0x2042(8258) | 0x2062(8290) | ……. | 0x27C2(10178) |
| + 0x03(+3) | 0x2003(8195) | 0x2023(8227) | 0x2043(8259) | 0x2063(8291) | ……. | 0x27C3(10179) |
| + 0x04(+4) | 0x2004(8196) | 0x2024(8228) | 0x2044(8260) | 0x2064(8292) | ……. | 0x27C4(10180) |
| + 0x05(+5) | 0x2005(8197) | 0x2025(8229) | 0x2045(8261) | 0x2065(8293) | ……. | 0x27C5(10181) |
| + 0x06(+6) | 0x2006(8198) | 0x2026(8230) | 0x2046(8262) | 0x2066(8294) | ……. | 0x27C6(10182) |
| + 0x07(+7) | 0x2007(8199) | 0x2027(8231) | 0x2047(8263) | 0x2067(8295) | ……. | 0x27C7(10183) |
| + 0x08(+8) | 0x2008(8200) | 0x2028(8232) | 0x2048(8264) | 0x2068(8296) | ……. | 0x27C8(10184) |
| + 0x09(+9) | 0x2009(8201) | 0x2029(8233) | 0x2049(8265) | 0x2069(8297) | ……. | 0x27C9(10185) |
| + 0x0A(+10) | 0x200A(8202) | 0x202A(8234) | 0x204A(8266) | 0x206A(8298) | ……. | 0x27CA(10186) |
| + 0x0B(+11) | 0x200B(8203) | 0x202B(8235) | 0x204B(8267) | 0x206B(8299) | ……. | 0x27CB(10187) |
| + 0x0C(+12) | 0x200C(8204) | 0x202C(8236) | 0x204C(8268) | 0x206C(8300) | ……. | 0x27CC(10188) |
| + 0x0D(+13) | 0x200D(8205) | 0x202D(8237) | 0x204D(8269) | 0x206D(8301) | ……. | 0x27CD(10189) |
| + 0x0E(+14) | 0x200E(8206) | 0x202E(8238) | 0x204E(8270) | 0x206E(8302) | ……. | 0x27CE(10190) |
| + 0x0F(+15) | 0x200F(8207) | 0x202F(8239) | 0x204F(8271) | 0x206F(8303) | ……. | 0x27CF(10191) |
| + 0x10(+16) | 0x2010(8208) | 0x2030(8240) | 0x2050(8272) | 0x2070(8304) | ……. | 0x27D0(10192) |
| + 0x11(+17) | 0x2011(8209) | 0x2031(8241) | 0x2051(8273) | 0x2071(8305) | ……. | 0x27D1(10193) |
| + 0x12(+18) | 0x2012(8210) | 0x2032(8242) | 0x2052(8274) | 0x2072(8306) | ……. | 0x27D2(10194) |
| + 0x13(+19) | 0x2013(8211) | 0x2033(8243) | 0x2053(8275) | 0x2073(8307) | ……. | 0x27D3(10195) |
| + 0x14(+20) | 0x2014(8212) | 0x2034(8244) | 0x2054(8276) | 0x2074(8308) | ……. | 0x27D4(10196) |
| + 0x15(+21) | 0x2015(8213) | 0x2035(8245) | 0x2055(8277) | 0x2075(8309) | ……. | 0x27D5(10197) |
| + 0x16(+22) | 0x2016(8214) | 0x2036(8246) | 0x2056(8278) | 0x2076(8310) | ……. | 0x27D6(10198) |
| + 0x17(+23) | 0x2017(8215) | 0x2037(8247) | 0x2057(8279) | 0x2077(8311) | ……. | 0x27D7(10199) |
| + 0x18(+24) | 0x2018(8216) | 0x2038(8248) | 0x2058(8280) | 0x2078(8312) | ……. | 0x27D8(10200) |
| + 0x19(+25) | 0x2018(8217) | 0x2038(8249) | 0x2058(8281) | 0x2078(8313) | ……. | 0x27D9(10201) |
| + 0x1A(+26) | 0x201A(8218) | 0x203A(8250) | 0x205A(8282) | 0x207A(8314) | ……. | 0x27DA(10202) |
| + 0x1B(+27) | 0x201B(8219) | 0x203B(8251) | 0x205B(8283) | 0x207B(8315) | ……. | 0x27DB(10203) |
| + 0x1C(+28) | 0x201C(8220) | 0x203C(8252) | 0x205C(8284) | 0x207C(8316) | ……. | 0x27DC(10204) |
| + 0x1D(+29) | 0x201D(8221) | 0x203D(8253) | 0x205D(8285) | 0x207D(8317) | ……. | 0x27DD(10205) |
| + 0x1E(+30) | 0x201E(8222) | 0x203E(8254) | 0x205E(8286) | 0x207E(8318) | ……. | 0x27DE(10206) |
| + 0x1F(+31) | 0x201F(8223) | 0x203F(8255) | 0x205F(8287) | 0x207F(8319) | ……. | 0x27DF(10207) |

| Address Offset | Access | Type, Size | Description |
|---|---|---|---|
| + 0x02(+2) ** | Read | 1word | Input start register address of input image word this slot. |
| + 0x03(+3) ** | Read | 1word | Input word's bit offset of input image word this slot. |
| + 0x04(+4) ** | Read | 1word | Output start register address of output image word this slot. |
| + 0x05(+5) ** | Read | 1word | Output word's bit offset of output image word this slot. |
| + 0x06(+6) ** | Read | 1word | Input bit start address of input image bit this slot. |
| + 0x07(+7) ** | Read | 1word | Output bit start address of output image bit this slot. |
| + 0x08(+8) ** | Read | 1word | Size of input bit this slot |
| + 0x09(+9) ** | Read | 1word | Size of output bit this slot |
| + 0x0A(+10)** | Read | n word | Read input data this slot |
| + 0x0B(+11)** | Read/Write | n word | Read/write output data this slot |
| + 0x0E(+14) | Read | 1word | M-number, if M-1238, returns 0x1238 |
| + 0x0F(+15) | Read | String upto 72byte | First 1word is length of valid character string.<br>If M1238, returns<br>"00 1E 52 54 2D 31 32 33 38 2C 20 38 44 49 2C 20 32 34 56 64 63 2C 20 55 6E 69 76 65 72 73 61 6C 00 00"<br>Valid character size = 0x001E =30 characters,<br>"M-1238, 8DI, 24Vdc, Universal" |
| + 0x10(+16) | Read | 1word | Size of configuration parameter byte |
| + 0x11(+17)** | Read/Write | n word | Read/write Configuration parameter data, up to 8byte. Refer to A.2 *** |
| + 0x17(+23) | Read | 2word | Firmware Revision<br>ex) 0x00010010 (Major revision 1 /Minor revision 16, Rev 1.016) |
| + 0x19(+25) | Read | 2word | Firmware release date. |

* After the system is reset, the new "Set Value" action is applied.
** Nothing of output, input, memory or configuration parameter corresponding slot returns Exception 02.

## 6.5. Supported MODBUS Function Codes

MODBUS Reference Documents

http://www.modbus.org

MODBUS Tools

http://www.modbustools.com, modbus poll
http://www.win-tech.com, modscan32