Open Things Access Protocol

Developer Guide

About This Manual

The Manual includes instructions for using and managing the Product. Pictures, charts, images and all other information hereinafter are for description and explanation only. The information contained in the Manual is subject to change, without notice, due to firmware updates or other reasons. Please find the latest version of this Manual at the HIKMICRO website (http://www.hikmicrotech.com).

Please use this Manual with the guidance and assistance of professionals trained in supporting the Product.

Trademarks

HIKMICRO and other HIKMICRO's trademarks and logos are the properties of HIKMICRO in various jurisdictions. Other trademarks and logos mentioned are the properties of their respective owners.

Disclaimer

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, THIS MANUAL AND THE PRODUCT DESCRIBED, WITH ITS HARDWARE, SOFTWARE AND FIRMWARE, ARE PROVIDED "AS IS" AND "WITH ALL FAULTS AND ERRORS". HIKMICRO MAKES NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, MERCHANTABILITY, SATISFACTORY QUALITY, OR FITNESS FOR A PARTICULAR PURPOSE. THE USE OF THE PRODUCT BY YOU IS AT YOUR OWN RISK. IN NO EVENT WILL HIKMICRO BE LIABLE TO YOU FOR ANY SPECIAL, CONSEQUENTIAL, INCIDENTAL, OR INDIRECT DAMAGES, INCLUDING, AMONG OTHERS, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, OR LOSS OF DATA, CORRUPTION OF SYSTEMS, OR LOSS OF DOCUMENTATION, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), PRODUCT LIABILITY, OR OTHERWISE, IN CONNECTION WITH THE USE OF THE PRODUCT, EVEN IF HIKMICRO HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES OR LOSS.

YOU ACKNOWLEDGE THAT THE NATURE OF THE INTERNET PROVIDES FOR INHERENT SECURITY RISKS, AND HIKMICRO SHALL NOT TAKE ANY RESPONSIBILITIES FOR ABNORMAL OPERATION, PRIVACY LEAKAGE OR OTHER DAMAGES RESULTING FROM CYBER-ATTACK, HACKER ATTACK, VIRUS INFECTION, OR OTHER INTERNET SECURITY RISKS; HOWEVER, HIKMICRO WILL PROVIDE TIMELY TECHNICAL SUPPORT IF REQUIRED.

YOU AGREE TO USE THIS PRODUCT IN COMPLIANCE WITH ALL APPLICABLE LAWS, AND YOU ARE SOLELY RESPONSIBLE FOR ENSURING THAT YOUR USE CONFORMS TO THE APPLICABLE LAW. ESPECIALLY, YOU ARE RESPONSIBLE, FOR USING THIS PRODUCT IN A MANNER THAT DOES NOT INFRINGE ON THE RIGHTS OF THIRD PARTIES, INCLUDING WITHOUT LIMITATION, RIGHTS OF PUBLICITY, INTELLECTUAL PROPERTY RIGHTS, OR DATA PROTECTION AND OTHER PRIVACY RIGHTS. YOU SHALL NOT USE THIS PRODUCT FOR ANY PROHIBITED END-USES, INCLUDING THE DEVELOPMENT OR PRODUCTION OF WEAPONS OF MASS DESTRUCTION, THE DEVELOPMENT OR PRODUCTION OF CHEMICAL OR BIOLOGICAL WEAPONS, ANY

ACTIVITIES IN THE CONTEXT RELATED TO ANY NUCLEAR EXPLOSIVE OR UNSAFE NUCLEAR FUEL-CYCLE, OR IN SUPPORT OF HUMAN RIGHTS ABUSES.

IN THE EVENT OF ANY CONFLICTS BETWEEN THIS MANUAL AND THE APPLICABLE LAW, THE LATTER PREVAILS.

1. Reading Guide

Chapter	Description
Basic Communication Protocol Description	Includes definitions of terms, Modbus protocol principles, frame format, register explanation, function code explanation, and error code explanation.
Device Basic Function Description	Explains the basic functions of the device framework: serial port configuration, baud rate, address code, basic information, sensor capabilities.
Business Function Description	Introduces business functions and defines the function point table.

2. Basic Communication Protocol Description

2.1 Terms and Definitions

- **Modbus Function Code**

  The element of the Modbus request/response PDU. It indicates the operation to be performed by the server.

- **Register Start Address**

  The start address of the register, occupying 16 bits. For example: 00 01.

- **Unit**

  The measurement unit of Modbus device.

- Register: A container for storing the internal state and data of a device, primarily used for storing and transmitting various types of data.
- Number of Registers: The number of registers corresponding to the data, occupying 16 bits. For example: 00 02.
- Register Byte Order: Whether to swap the high and low bytes within a register. Controls the parsing of the byte order within registers. **Default: Big-endian parsing.**Example 1: When writing 12345, the hexadecimal value is 3039. Using big-endian, the send/receive sequence is 30 39; using little-endian, it is 39 30.
  Example 2: If using function code 0x10 to send an RTU message to a slave device with address code 1, writing to the holding register. Using big-endian, the complete message is: 01 10 02 00 01 **30 39** F7 E1; using little-endian, it is: 01 10 02 00 01 **39 30** 33 14.
- **Register Order**

  Whether to swap the positions of registers. Controls the parsing order of registers (only valid for data types other than int16 and uint16). **Default: No swap.**

  - Example 1: If the start address is 0001 and the number of registers is 0002, the register at address 0002 will be swapped with the register at address 0001.
  - Example 2: When sending int64 data (1234567890ABCDEF), it is split into 4 registers. If no swap is performed, the send order is 12 34 56 78 90 AB CD EF; if swapped, the order is CD EF 90 AB 56 78 12 34.

- **Scaling Factor**

  Multiply the data in the register by a scaling factor to obtain the desired value. Configure how the device's raw data is scaled. Typically, when floating-point computation capabilities are limited, floating-point numbers may be represented as integers. If the device employs this method, the scaling factor can be applied to adjust the raw data.

    - Example 1: If the retrieved temperature data is 365 and the scaling factor is 0.1, the actual temperature is calculated as 365 × 0.1 = 36.5.
    - Example 2: If the device uses 1234 to represent 12.34, the scaling factor should be set to 0.01. In this case, any value sent to the device (e.g., 12.34 from the cloud or console) is automatically multiplied by 100 (converted to 1234) before transmission. Conversely, when the device sends 1234, the driver converts it back to 12.34 for the cloud or other devices.

- **Data Type**

  The data type of the register. Data types include int16, uint16, int32, uint32, int64, uint64, float, and double. Different data types use different numbers of registers. For details, refer to the table below: Register Data Type Usage.

Register Number of Different Data Type

Data Type	Number of Registers Used	Description
int16	1	16-bit signed integer.
uint16	1	16-bit unsigned integer.
int32	2	32-bit signed integer.
uint32	2	32-bit unsigned integer.
int64	4	64-bit signed integer.
uint64	4	64-bit unsigned integer.
float	2	Single-precision floating-point (32-bit).
double	4	Double-precision floating-point (64-bit).
string	N	String type.

2.2 Modbus Protocol Principle

2.2.1 Modbus RTU

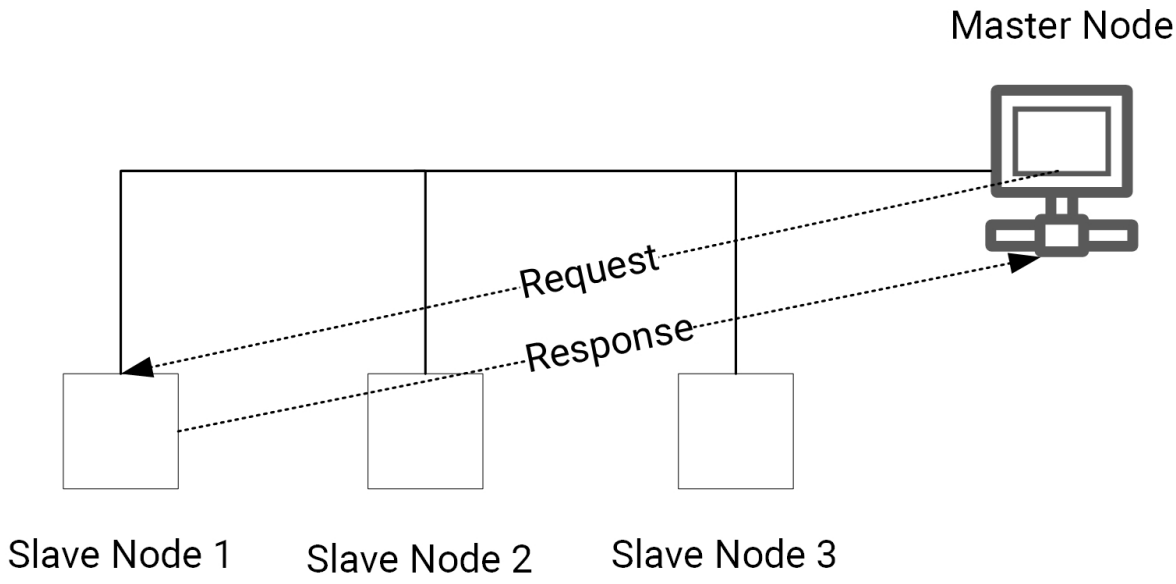
The Modbus RTU protocol is a master-slave protocol:

- The master node will only initiate one Modbus transaction at a time.
- Modbus communication is always initiated by the master node. Slave nodes will not send data unless requested by the master node.
- If an incorrect exchange occurs, the master node will resend the request. If no response is received within a specified time, the master node will declare the slave node as unavailable.

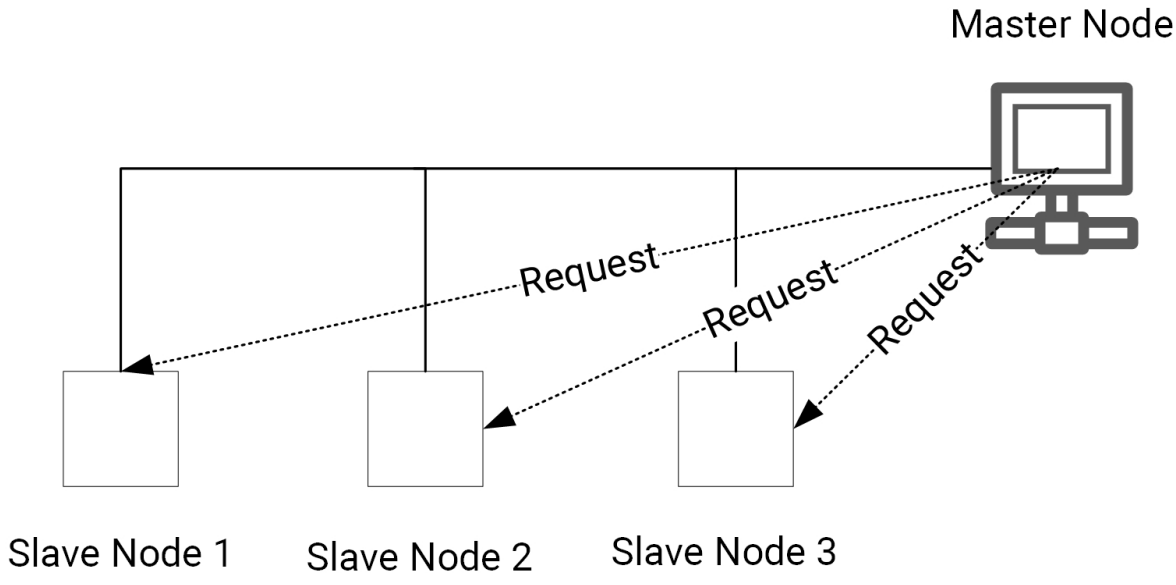
- If a slave node does not understand a message, it will send an exception response to the master node. The master node will determine whether to retry based on the error cause. If the error is temporary (e.g., communication interference), the master node may resend the request; if due to a slave node limitation (e.g., unsupported function), it may not retry.

Two types of communication can occur between the master and slave nodes:

- **Unicast Mode**: The master node accesses a specific slave node using its address. After processing the request, the slave node sends a response to the master node. This mode involves two messages: one request from the master and one response from the slave.



- **Broadcast Mode**: The master node broadcasts requests to all slave nodes without waiting for a response. Slave nodes cannot communicate directly; communication between them must be routed through the master node.



2.3 Frame Format Definition

2.3.1 Modbus RTU Protocol Frame

The Modbus RTU protocol structure is as follows:

Data Frame	Communication Protocol	Description
Initial Structure	≥4 bytes of time	Character: The time taken to send one byte of data via serial port under even parity, i.e. 11-bit time.
Address Code	1 byte	Specific address, unique in the communication network. The default value is 0x01, and it is configurable with a range of 1~247. Address code 0x00 is used for broadcast communication.
Function Code	1 byte	Host instruction function identifier. Exception response function code: Function code + 0x80.
Register Start Address	2 bytes	0x0000 to 0xFFFF.
Data Area	N bytes	The data area contains the specific communication data. Note that 16-bit data has the high byte first.
Error Check	16-bit CRC code	2-byte checksum.
Exception Code	1 byte	01 or 02 or 03 or 04.
End Structure	≥4 bytes of time	

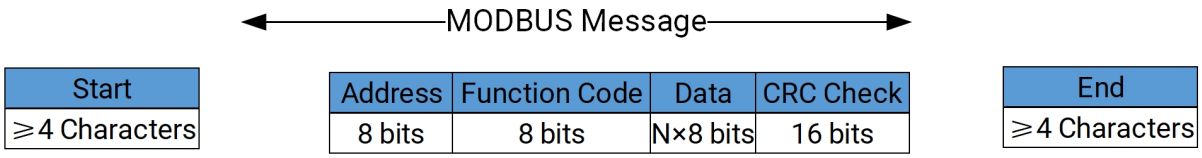
PDU (Protocol Data Unit) refers to: Function code + Register start address + Data area.

- **Modbus RTU Message Frame**:
  The maximum size of a Modbus RTU message frame is 256 bytes.

Slave Node	Function Code	Data	CRC Low	CRC High
1 byte	1 byte	0-252 bytes	1 byte	1 byte

- **Modbus RTU Communication Protocol Data Format**:
  The sending device constructs a Modbus message frame with known start and end markers. This allows devices to detect new frames and determine when a frame ends. Incomplete frames must be detectable, and error flags must be set accordingly. In RTU mode, frames are separated by idle intervals of at least 4 character times. The Modbus RTU bus uses time intervals to determine the end of a

frame; if no new data is received within 4 character times, the current frame is considered complete.



- **Slave Exception Response Frame Structure**:

Address Code	Error Code	Exception Code	CRC Low	CRC High
1 byte	Function Code + 0x80	1 byte	1 byte	1 byte

All register addresses, register counts, and data bytes in commands use high bytes first; CRC codes use low bytes first.

- CRC Check

The process to generate the CRC checksum is as follows:

1. Initialize a 16-bit register with the value 0xFFFF. This register is the CRC register.
2. Perform an XOR operation between the first 8-bit binary data of the communication data frame and the lower 8 bits of the CRC register, and store the result of the XOR operation in the CRC register.
3. Right-shift the contents of the CRC register by one bit. Fill the most significant bit (MSB) with zero, and check the shifted-out bit (0 or 1).
4. If the shifted-out bit is 0, repeat step 3. If it is 1, XOR the CRC register with A001H.
5. Repeat steps 3 and 4 until 8 right shifts are performed. This completes processing the first 8-bit data of the frame.
6. Repeat steps 2, 3, 4, and 5 to process the remaining bytes of the communication data frame sequentially.
7. After processing all bytes, swap the high and low bytes of the 16-bit CRC register.
8. The final content of the CRC register is the CRC checksum.

Common CRC-16 algorithms include the **lookup table method** and the **direct computation method**. For more details, refer to relevant technical documentation.

Calculation Process Example:

```
uint16_t calculate_crc(uint8_t *data, size_t length) {
    uint16_t crc = 0xFFFF; // Initial value
    for (size_t i = 0; i < length; i++) { // Process data byte by byte
        crc ^= data[i]; // XOR operation
        for (int j = 0; j < 8; j++) { // Process 8 bits of each byte
            if (crc & 0x0001) { // Check least significant bit (LSB)
                crc >>= 1; // Right shift by 1 bit
```

```
            crc ^= 0xA001; // XOR with polynomial 0xA001
        } else {
            crc >>= 1; // Right shift by 1 bit
        }
      }
   }
   return crc;
}
```

Take a byte array data and its length length, initialize the CRC value to 0xFFFF, perform an XOR operation on each byte, process all 8 bits of each byte, and return the calculated CRC value.

2.4 Registers

2.4.1 Register Types and Description

In the Modbus protocol, there are several types of registers:

Register Type	Description	Read/Write State	Comparison with PLC	Example
Coil Status	Digital output, relay output	Read/Write	DO digital output or internal read/write variables	Solenoid valve output, relay output.
Input Status	Digital input	Read Only	DI digital input or internal read-only variables	Button input, DIP switch, proximity switch.
Holding Register	Output parameters or maintained parameters	Read/Write	AO analog output or internal read/write registers	Temperature measurement mode, emissivity configuration.
Input Register	Input parameters	Read Only	AI analog input or internal read-only registers	Temperature measurement range.

2.4.2 Register Address Allocation

- Different register types have the following address ranges:

Register Type	PLC Address Range (1~9999)	PLC Address Range (1~65536)	Register Address (Hex)	Supported Function Codes	Read/Write State
Coil Status	00001~09999	000001~065536	0000~FFFF	01, 05, 15	Read/Write
Input Status	10001~19999	100001~165536	0000~FFFF	02	Read Only
Holding Register	40001~49999	400001~465536	0000~FFFF	03, 06, 16, 23	Read/Write
Input Register	30001~39999	300001~365536	0000~FFFF	04	Read Only

- PLC Address and Hex Address Conversion:
  The first digit of the PLC address represents the register type, indicated by 0, 1, 4, or 3 for coil, input status, holding, and input registers, respectively. Since 65536 is a large number and generally unnecessary, the range 1~9999 is sufficient for most

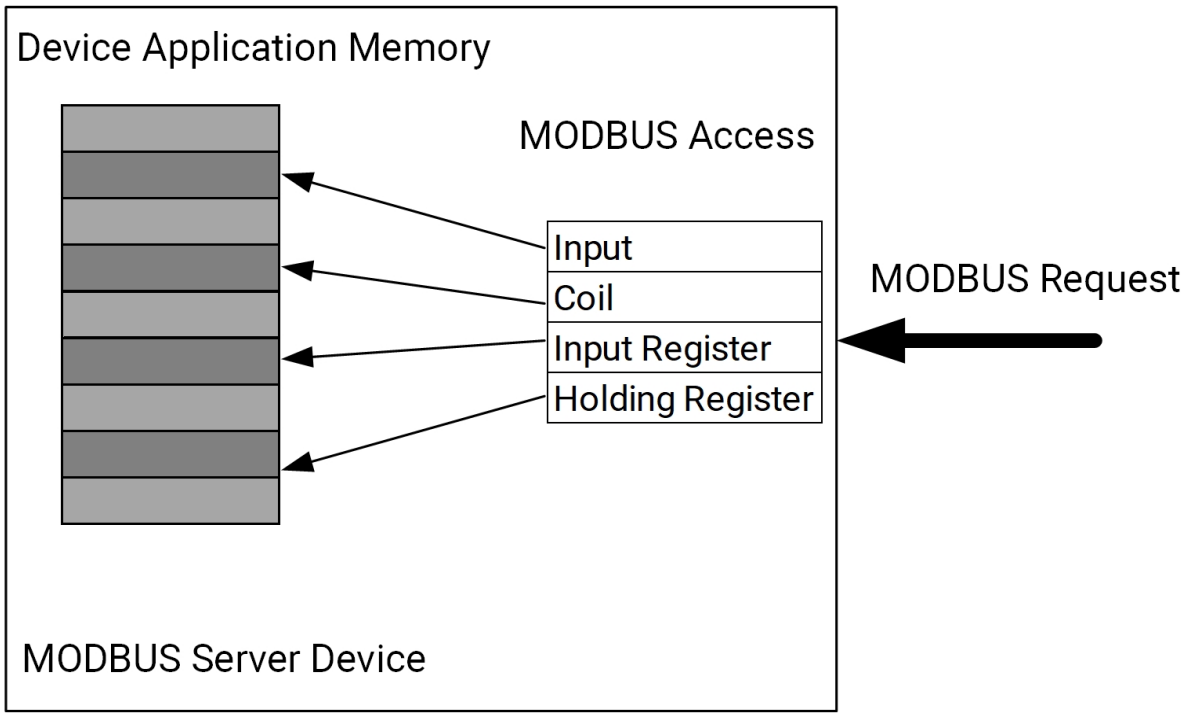
applications.

1. Choose the PLC address range (1~9999 or 1~65536) based on requirements.
2. For holding registers, if using the 1~9999 range, the start address is 40001; if using the 1~65536 range, it is 400001.
3. Convert the hexadecimal address to a decimal value and add the start address to obtain the PLC address.

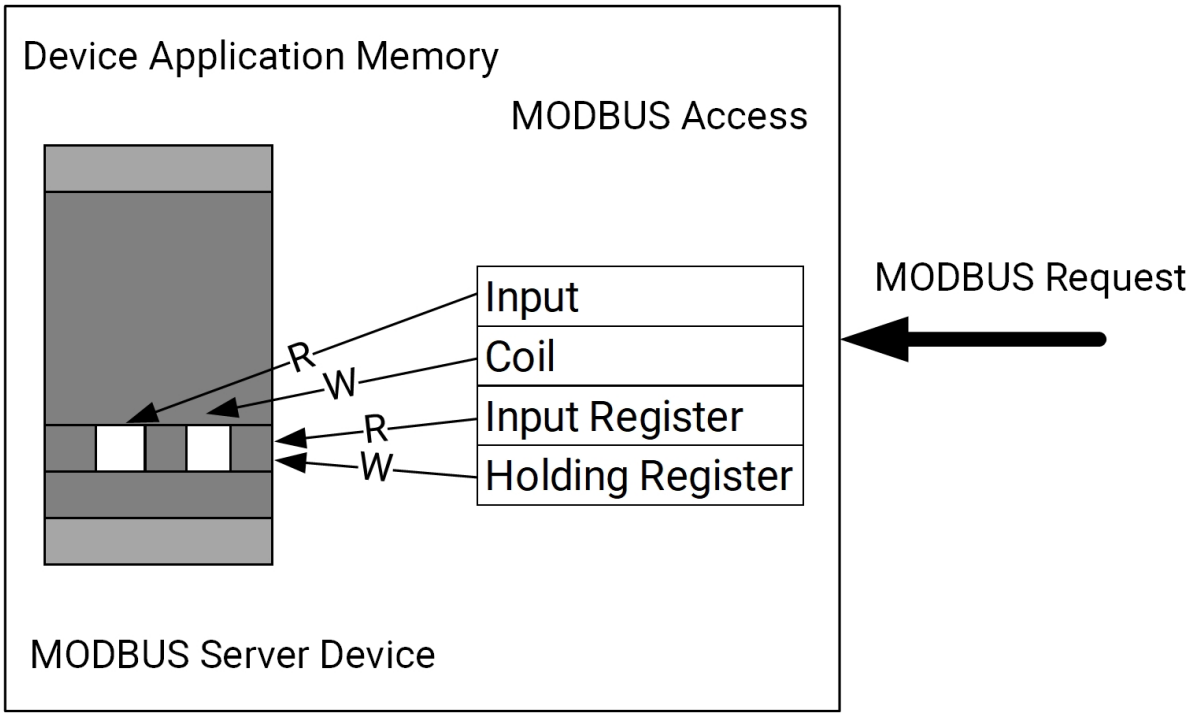
- **Example**:
  - **Holding Register**, Function Code 03, register hexadecimal value `0x0002`, convert to PLC address: `40001 + 2 = 40003`.
  - **Holding Register**, Function Code 03, register hexadecimal value `0x1000`, convert to PLC address: `40001 + 4096 = 44097`.
  - **Input Register**, Function Code 04, register hexadecimal value `0x3217`, convert to PLC address: `300001 + 12823 = 312824`.

2.4.3 Register Data Model

- **Examples of Modbus Register Type Implementations**
  Devices of different types have their own data structures depending on their applications. The following examples illustrate methods for structuring data within devices.
- **Example: Device with Four Independent Blocks**
  The device data structure in this example contains digital and analog data, as well as input and output values. Since the data in different blocks are unrelated, each block operates independently. Each block is accessed via distinct Modbus function codes.



- **Example: Device with a Single Block**
  In this example, the device has only one data block. Multiple Modbus function codes can be used to access the same data block.



2.5 Function Code Description

Function Code	Function Code Name	Purpose	Protocol Support
02 (0x02)	Read Input Status	Reads the status of 1 to 2000 discrete inputs.	No
03 (0x03)	Read Holding Register	Obtains binary values from one or more holding registers.	Yes
04 (0x04)	Read Input Register	Obtains binary values from one or more input registers.	Yes
06 (0x06)	Write Single Register	Writes a specific binary value to a single holding register.	No
16 (0x10)	Write Multiple Registers	Writes binary values to a series of consecutive holding registers.	Yes

The PDU (Protocol Data Unit) varies depending on the function code. The structures of commonly used function code PDUs are listed below.

2.5.1 03 (0x03) Function Code Description

- Request

Function Code	1 Byte	0x03
Register Start Address	2 Bytes	0x0000 to 0xFFFF
Register Number	2 Bytes	1 to 125（0x7D）

- Normal Response

Function Code	1 Byte	0x03
Byte Count	1 Byte	2*N
Register Data	N*2 Bytes	

N = Register Number

- Exception Response

Exception Code	1 Byte	0x83
Exception Code	1 Byte	01 or 02 or 03 or 04

Example: Request to read registers 0x006B-0x006D

Request Field	Request (Hex)	Response Field	Response (Hex)
Function Code	03	Function Code	03
High Starting Address	00	Byte Count	06
Low Starting Address	6B	Register Value Hi (0x006B)	02
High Register Count	00	Register Value Lo (0x006B)	2B
Low Register Count	03	Register Value Hi (0x006C)	00
		Register Value Lo (0x006C)	00
		Register Value Hi (0x006D)	00
		Register Value Lo (0x006D)	64

2.5.2 Function Code 16 (0x10) Instructions

- **Request PDU**

Function Code	**1 byte**	**0x10**
Start Address	2 bytes	0x0000 to 0xFFFF
Number of Registers	2 bytes	0x0001 to 0x0078
Byte Count	1 byte	2×**N***
Register Values	**N***×2 bytes	Values to write

Note: **N** = Number of Registers

- **Normal Response PDU**

Function Code	**1 byte**	**0x10**
Start Address	2 bytes	0x0000 to 0xFFFF
Number of Registers	2 bytes	1 to 123 (0x7B)

- **Exception Response**

Exception Code	**1 byte**	**0x90**
Exception Code	1 byte	01, 02, 03, or 04

Example: Writing hex values `00 0A` and `01 02` to two registers starting at address 2

Request Field	Request (Hex)	Response Field	Response (Hex)
Function Code	10	Function Code	10
Starting Address High	00	Output Address High	00
Starting Address Low	01	Output Address Low	01
Number of Registers High	00	Number of Registers High	00
Number of Registers Low	02	Number of Registers Low	02
Byte Count	04		
Register Value High (1st)	00		
Register Value Low (1st)	0A		
Register Value High (2nd)	01		
Register Value Low (2nd)	02		

2.6 Exception Code Description

Exception Code	Name	Meaning
01	Illegal Function	For the server (or slave), the function code received in the query is an illegal operation. This might be because the function code is only applicable to new devices and is not implemented in the selected unit. Additionally, it indicates that the server (or slave) is in an error state when processing such a request, for example, because it is unconfigured and requires returning register values.
02	Illegal Data Address	For the server (or slave), the data address received in the query is an illegal address. Specifically, the combination of the reference number and transfer length is invalid. For a controller with 100 registers, a request with an offset of 96 and a length of 4 will succeed, while a request with an offset of 96 and a length of 5 will generate exception code 02.
03	Illegal Data Value	For the server (or slave), the value included in the query is an illegal value. This value indicates a fault in the remaining structure of the combined request, for example: the implied length is incorrect. As MODBUS operates without interpreting register value semantics, it does not mean that the data item submitted for storage in the register has a value outside the application's expectations.
04	Slave Device Failure	An unrecoverable error occurred while the server (or slave) was attempting to execute the requested operation.
05	Acknowledge	Used to prevent timeout errors caused by excessive processing time.
06	Slave Device Busy	For example: For the input sources of emissivity, slope, and background temperature compensation, only one parameter source can be configured as external input at a time. If one is in use, configuring another will return BUSY; or when DHCP is enabled, manual IP address modification is not supported. If the IP address is to be modified via protocol, BUSY will also be returned.
08	Memory Parity Error	Memory parity error.
0A	Gateway Path Unavailable	Gateway path unavailable.
0B	Gateway Target Device Failed to Respond	Gateway target device failed to respond.

Exception Code	Name	Meaning
0C	Pyrometer Temperature Exceeds Lower Limit	Pyrometer temperature exceeds lower limit.
0D	Pyrometer Temperature Exceeds Upper Limit	Pyrometer temperature exceeds upper limit.
0E	Restart Required	Restart required, or restart is recommended. For example: After configuring DHCP for the pyrometer, the user is prompted to restart the device.

If the server detects a communication error (parity bit, LRC, CRC), it will not respond, and the client program will handle the request timeout.

3. Device Function Description

3.1 Pyrometer Information Acquisition

3.1.1 Function Description

Obtains the pyrometer's measurement results and the upper and lower limits of the measurement range. The register type used is input registers, with function code 0x04 and register addresses 0x0230-0x023F.

3.1.2 Function Table

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Temperature Information Acquisition	560-561	0x0230-0x0231	0x04	2	uint32		Obtains the pyrometer's measured temperature, retaining three decimal places. Each temperature occupies two registers. Scaling factor: 0.001.
Measurement Lower Limit	562	0x0232	0x04	1	uint16		Obtains the lower limit of the pyrometer's measurement range.
Measurement Upper Limit	563	0x0233	0x04	1	uint16		Obtains the upper limit of the pyrometer's measurement range.
Reserved	564-575	0x0234-0x023F	0x04	12			

3.2 Measurement Parameter

3.2.1 Function Description

Configures or obtains pyrometer measurement parameters, such as measurement mode, emissivity, etc., to ensure accurate data measurement. The register type used is holding registers, with function codes 0x03/0x10 and register addresses 0x0200-0x020F.

3.2.2 Function Table

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Measurement Mode	512	0x0200	0x03/0x10	1	uint16		0: Invalid parameter when setting; not supported when retrieving. 1: 1-color mode. 2: 2-color mode.
Emissivity	513	0x0201	0x03/0x10	1	uint16		Range: 100-1100. Scaling factor: 0.001.
Slope	514	0x0202	0x03/0x10	1	uint16		Range: 850-1150. Scaling factor: 0.001.
2-Color Signal Attenuation Percentage	515	0x0203	0x03/0x10	1	uint16	%	Range: 0-100.
External Optical Transmittance	516	0x0204	0x03/0x10	1	uint16		Range: 50-2000. Scaling factor: 0.001.
Enable Temperature Attenuation Threshold	517	0x0205	0x03/0x10	1	BOOL		
Temperature Attenuation Threshold Parameter	518	0x0206	0x03/0x10	1	uint16		Range: 0-100.
Reserved	519-527	0x0207-0x020F	0x03/0x10	9			

3.3 Time Function

3.3.1 Function Description

Configures or obtains pyrometer time parameters, such as response time and peak/valley hold delay, to optimize data processing in various testing environments. The register type used is holding registers, with function codes 0x03/0x10 and register addresses 0x0210-0x021F.

3.3.2 Function Table

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Response Time	528-529	0x0210-0x0211	0x03/0x10	2	uint32	ms	Range: 1-10000.
Enable Averaging	530	0x0212	0x03/0x10	1	BOOL		0x0000: Disable. 0xFF00: Enable.
Averaging Number	531	0x0213	0x03/0x10	1	uint16		Range: 1-255.
Enable Filter	532	0x0214	0x03/0x10	1	BOOL		0x0000: Disable. 0xFF00: Enable.
Filter Mode	533	0x0215	0x03/0x10	1	uint16		0: Invalid parameter when setting; not supported when reading. 1: Hold mode. 2: Reset mode.
Filter Upper Limit	534	0x0216	0x03/0x10	1	uint16		Range determined by measurement temperature limits (input registers 0x0232-0x0233). Scaling factor: 0.1.
Filter Lower Limit	535	0x0217	0x03/0x10	1	uint16		Range determined by measurement temperature limits (input registers 0x0232-0x0233). Scaling factor: 0.1.

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Enable Peak/Valley Hold	536	0x0218	0x03/0x10	1	BOOL		0x0000: Disable. 0xFF00: Enable.
Peak/Valley Hold Mode	537	0x0219	0x03/0x10	1	uint16		0: Invalid parameter when setting; not supported when reading. 1: Peak mode. 2: Valley mode.
Peak/Valley Max. Duration	538-539	0x021A-0x021B	0x03/0x10	2	uint32		Range: 1~1800000.
Output Time Mode	540	0x021C	0x03/0x10	1	uint16		0: Invalid parameter when setting; not supported when reading. 1: Average mode. 2: Peak mode.
Reserved	541-543	0x021D-0x021F	0x03/0x10	3			

3.4 Alarm

3.4.1 Function Description

Configures or obtains pyrometer alarm parameters, such as alarm switches and alarm thresholds, to meet various alarm requirements. Function codes are 0x03/0x10, and register addresses are 0x0220-0x022F.

3.4.2 Function Table

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Upper Alarm Temperature	544	0x0220	0x03/0x10	1	uint16		Range determined by measurement temperature limits (input registers 0x0232-0x0233). Scaling factor: 0.1.
Lower Alarm Temperature	545	0x0221	0x03/0x10	1	uint16		Range determined by measurement temperature limits (input registers 0x0232-0x0233). Scaling factor: 0.1.
Internal Temperature Upper Limit	547	0x0223	0x03/0x10	1	uint16	℃	Range: 40-65.0. Scaling factor: 0.1.
Reserved	548-559	0x0224-0x022F	0x03/0x10	12			

3.5 Analog Output

3.5.1 Function Description

Configure/retrieve pyrometer analog output parameters (e.g., analog output modes, output sources) to meet diverse output requirements. Uses Function Codes 0x03/0x10 and operates on register addresses 0x0230-0x023F.

3.5.2 Function Table

Field Name	Register Address (Dec)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Analog Output Mode IO1	560	0x0230	0x03/0x10	1	uint16		0: Invalid parameter when setting; unsupported when reading 1: (4–20) mA 2: (0–20) mA 5: (4–20) mA (Saturation And Alarm) (includes saturation zone and alarm zone) 6: (4–20) mA (Alarm) (alarm zone only)
Analog Start Value IO1	561	0x0231	0x03/0x10	1	uint16		Range depends on measured temperature limits (input registers 0x0232-0x0233) Coefficient: 0.1
Analog End Value IO1	562	0x0232	0x03/0x10	1	uint16		Range depends on measured temperature limits (input registers 0x0232-0x0233) Coefficient: 0.1

Field Name	Register Address (Dec)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Analog Output Mode IO2	563	0x0233	0x03/0x10	1	uint16		0: Invalid parameter when setting; unsupported when reading 1: (4–20) mA 2: (0–20) mA 5: (4–20) mA (Saturation And Alarm) (includes saturation zone and alarm zone) 6: (4–20) mA (Alarm) (alarm zone only)
Analog Start Value IO2	564	0x0234	0x03/0x10	1	uint16		Range depends on measured temperature limits (input registers 0x0232-0x0233) Coefficient: 0.1
Analog End Value IO2	565	0x0235	0x03/0x10	1	uint16		Range depends on measured temperature limits (input registers 0x0232-0x0233) Coefficient: 0.1

Field Name	Register Address (Dec)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Analog Output 1 Alarm Mode	567	0x0237	0x03/0x10	1	uint16		0: Invalid parameter when setting; unsupported when reading 1: Low current alarm 2: High current alarm Note: Only effective for modes with alarm zone (holding registers 0x0230, 0x0233)

Field Name	Register Address (Dec)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Alarm Source Configuration for IO1	568	0x0238	0x03/0x10	1	uint16		0: Invalid parameter when setting; unsupported when reading 1: Lens contamination 2: Exceeds upper target temperature limit 3: Exceeds lower target temperature limit 4: Internal ambient temperature upper limit 5: F-type fault 6: Exceeds attenuation rate upper limit Note: Only effective for modes with alarm zone (holding registers 0x0230, 0x0233)

Field Name	Register Address (Dec)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Alarm Source Configuration for IO2	569	0x0239	0x03/0x10	1	uint16		0: Invalid parameter when setting; unsupported when reading 1: Lens contamination 2: Exceeds upper target temperature limit 3: Exceeds lower target temperature limit 4: Internal ambient temperature upper limit 5: F-type fault 6: Exceeds attenuation rate upper limit Note: Only effective for modes with alarm zone (holding registers 0x0230, 0x0233)

Field Name	Register Address (Dec)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Analog Output 2 Alarm Mode	570	0x023A	0x03/0x10	1	uint16		0: Invalid parameter when setting; unsupported when reading 1: Low current alarm 2: High current alarm Note: Only effective for modes with alarm zone (holding registers 0x0230, 0x0233)
Alarm IO1 Switch	571	0x023B	0x03/0x10	1	BOOL		0x0000: Disable 0xFF00: Enable
Relay Mode (Alarm IO2)	572	0x023C	0x03/0x10	1	uint16		0: Invalid parameter 1: Normally closed 2: Normally open 3: Permanently closed 4: Permanently open
Reserved	573-575	0x023D-0x023F	0x03/0x10	3	uint16		

3.6 Simulation

3.6.1 Function Description

Configures or obtains pyrometer simulation parameters, such as simulation mode and simulation enable, to simulate pyrometer measurement results. Function codes are 0x03/0x10, and register addresses are 0x0240-0x024F.

3.6.2 Function Table

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Analog Output Simulation Mode	576	0x0240	0x03/0x10	1	uint16		0: Invalid parameter when setting; not supported when reading. 1: Simulation current. 2: Simulation temperature.
Enable Analog Output Simulation	577	0x0241	0x03/0x10	1	BOOL		0x0000: Disable. 0xFF00: Enable.
Analog Output Simulation Value	578-579	0x0242-0x0243	0x03/0x10	2	uint32		Retains two decimal places, scaling factor: 0.01. 1. When simulation mode is temperature, it represents simulation temperature value, within the measurement temperature range. 2. When simulation mode is current, it represents simulation current value, range: 0-22 mA.
Enable Analog Output 1	580	0x0244	0x03/0x10	1	BOOL		0x0000: Disable. 0xFF00: Enable.

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Enable Analog Output 2	581	0x0245	0x03/0x10	1	BOOL		0x0000: Disable. 0xFF00: Enable.
Analog Output 1 Signal Source	582	0x0246	0x03/0x10	1	uint16		1: 1-color. 2: 2-color.
Analog Output 2 Signal Source	583	0x0247	0x03/0x10	1	uint16		1: 1-color. 2: 2-color.
Analog Output 1 Alarm Temperature Upper Limit	584	0x0248	0x03/0x10	1	uint16		Range determined by measurement range (input register 0x0232-0x0233). Scaling factor: 0.1.
Analog Output 1 Alarm Temperature Lower Limit	585	0x0249	0x03/0x10	1	uint16		Range determined by measurement range (input register 0x0232-0x0233). Scaling factor: 0.1.
Analog Output 2 Alarm Temperature Upper Limit	586	0x024A	0x03/0x10	1	uint16		Range determined by measurement range (input register 0x0232-0x0233). Scaling factor: 0.1.

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Analog Output 2 Alarm Temperature Lower Limit	587	0x024B	0x03/0x10	1	uint16		Range determined by measurement range (input register 0x0232-0x0233). Scaling factor: 0.1.
Reserved	588-591	0x024C-0x024F	0x03/0x10	4			

3.7 Temperature Compensation

3.7.1 Function Description

Configures or obtains pyrometer compensation parameters (Temperature Re-calibration), such as enable switches and calibration coefficients. These parameters allow the device to provide more accurate measurements when the actual environment differs from the factory conditions. Function codes are 0x03/0x10, and register addresses are 0x0250-0x025F.

3.7.2 Function Point Table

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Enable Temperature Compensation	592	0x0250	0x03/0x10	1	BOOL		0x0000: Disable. 0xFF00: Enable.
Temperature Compensation Mode	593	0x0251	0x03/0x10	1	uint16		0: Invalid parameter when setting; not supported when retrieving. 1: Direct parameter configuration compensation. 2: Single-point calibration compensation. 3: Two-point calibration compensation.
Compensation Gain	594	0x0252	0x03/0x10	1	uint16		Range: 0.00~2.00. Scaling factor: 0.01. Valid only when compensation mode is "Parameter Re-Calibration" (holding register 0x0251).
Compensation Bias Temperature	595	0x0253	0x03/0x10	1	uint16		Range: -200.0-200.0. Scaling factor: 0.1. Valid only when compensation mode is "Parameter Re-Calibration" (holding register 0x0251).

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Calibration Compensation - Original Temperature 1	596	0x0254	0x03/0x10	1	uint16		Scaling factor: 0.1. Range determined by measurement temperature limits (input registers 0x0232-0x0233). Valid only when compensation mode is "Single-Point" re-calibration" or "Two-Point" re-calibration (holding register 0x0251).
Calibration Compensation - Calibration 1	597	0x0255	0x03/0x10	1	uint16		Scaling factor: 0.1. Range determined by measurement temperature limits (input registers 0x0232-0x0233). Valid only when compensation mode is "Single-Point Calibration Compensation" or "Two-Point Calibration Compensation" (holding register 0x0251).

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Calibration Compensation-Original Temperature 2	598	0x0256	0x03/0x10	1	uint16		Scaling factor: 0.1. Range determined by measurement temperature limits (input registers 0x0232-0x0233). Valid only when compensation mode is "Two-Point" re-calibration (holding register 0x0251).
Calibration Compensation-Calibration 2	599	0x0257	0x03/0x10	1	uint16		Scaling factor: 0.1. Range determined by measurement temperature limits (input registers 0x0232-0x0233). Valid only when compensation mode is "Two-Point" re-calibration (holding register 0x0251).

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Calculate Compensation Parameters	600	0x0258	0x10	1	uint16		Write-only parameter. 0: Invalid parameter when setting; returns 0 when retrieving. 1: Calculate compensation parameters.
Signal Source	601	0x0259	0x03/0x10	1	uint16		1: 1-color. 2: 2-color.
Reserved	602-607	0x0259-0x025F	0x03/0x10	7	uint16		

3.8 DAC Temperature Re-Calibration

3.8.1 Function Description

Configures or obtains pyrometer DAC temperature re-calibration parameters, such as mode, gain coefficient, and bias current. These parameters allow the device to provide more accurate measurements when the actual environment differs from the factory conditions. Function codes are 0x03/0x10, and register addresses are 0x0280-0x028F.

3.8.2 Function Table

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Switch	640	0x0280	0x03/0x10	1	BOOL		0x0000: Disable. 0xFF00: Enable.
Mode	641	0x0281	0x03/0x10	1	uint16		0: Invalid parameter when setting; not supported when reading. 1: Single-point calibration. 2: Two-point calibration. 3: Parameter calibration.
Gain Coefficient	642-643	0x0282-0x283	0x03/0x10	2	uint32		Range: 0.8-1.25. Retains three decimal places. Valid only when mode is "Parameter Re-Calibration" (holding register 0x0281).
Bias Current	644-645	0x0284-0x0285	0x03/0x10	2	uint32		Range: -1~1. Retains three decimal places. Valid only when mode is "Parameter Re-Calibration" (holding register 0x0281).

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Original Current 1	646-647	0x0286-0x0287	0x03/0x10	2	uint32		Range: 0-20. Retains three decimal places. Valid only when mode is "Single-Point " or "Two-Point " Re-Calibration (holding register 0x0281).
Calibration Current 1	648-649	0x0288-0x0289	0x03/0x10	2	uint32		Range: 0-20. Retains three decimal places. Valid only when mode is "Single-Point " or "Two-Point " Re-Calibration (holding register 0x0281).
Original Current 2	650-651	0x028A-0x028B	0x03/0x10	2	uint32		Range: 0-20. Retains three decimal places. Valid only when mode is "Two-Point " Re-Calibration (holding register 0x0281).

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Calibration Current 2	652-653	0x028C-0x028D	0x03/0x10	2	uint32		Range: 0-20. Retains three decimal places. Valid only when mode is "Two-Point " Re-Calibration (holding register 0x0281).
Channel Number	654	0x028E	0x03/0x10	1	uint16		Range: 1-2.
Calculate Compensation Parameters	655	0x028F	0x10	1	uint16		Write-only parameter. 0: Invalid parameter when setting; returns 0 when reading 1: Calculate compensation parameters.
Reserved	656-671	0x0290-0x029F	0x03/0x10	16			

3.9 Combined Parameter

3.9.1 Function Description

Configures or obtains pyrometer combined temperature measurement parameters. Function codes are 0x03/0x10, and register addresses are 0x02A0-0x02AF.

3.9.2 Function Table

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Combined Parameter Sequence Number	672	0x02A0	0x10	1	uint16		Range: 1-10.
Control Mode	673	0x02A1	0x10	1	uint16		0: Apply. 1: Update. 2: Reset.
Reserved	674-687	0x02A2-0x02AF	0x03/0x10	14			

3.10 Network

3.10.1 Function Description

Configures or obtains pyrometer network parameters. Function codes are 0x03/0x10, and register addresses are 0x02BC-0x02F8.

3.10.2 Function Point Table

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Enable DHCP	700	0x02BC	0x03/0x10	1	uint16		0x0000: Disable. 0xFF00: Enable.
IPv4 Address	701-702	0x02BD-0x02BE	0x03/0x10	2	uint32		Each byte represents an address segment value. The low 8 bits of register 701 represent address segment 1, the high 8 bits of register 701 represent address segment 2, the low 8 bits of register 702 represent address segment 3, and the high 8 bits of register 702 represent address segment 4. For example, sending four 255s represents 255.255.255.255.
IPv4 Subnet Mask	703-704	0x02BF-0x02C0	0x03/0x10	2	uint32		Each byte represents an address segment value. The low 8 bits of register 703 represent address segment 1, the high 8 bits of register 703 represent address segment 2, the low 8 bits of register 704 represent address segment 3, and the high 8 bits of register 704 represent address segment 4. For example, sending four 255s represents 255.255.255.255.

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Gateway	705-706	0x02C1-0x02C2	0x03/0x10	2	uint32		Each byte represents an address segment value. The low 8 bits of register 705 represent address segment 1, the high 8 bits of register 705 represent address segment 2, the low 8 bits of register 706 represent address segment 3, and the high 8 bits of register 706 represent address segment 4. For example, sending four 255s represents 255.255.255.255.
Port Number	707	0x02C3	0x03/0x10	1	uint16		Range: 1024-65535.
Reserved	708-760	0x02C4-0x02F8	0x03/0x10	53			

3.11 Pyrometer Device Restart

3.11.1 Function Description

Pyrometer device restart function. Function code is 0x10, and register addresses are 0x02F9-0x0302.

3.11.2 Function Point Table

Field Name	Register Address (Decimal)	Register Address (Hex)	Function Code	Number of Registers	Data Type	Unit	Description
Mode	761	0x02F9	0x10	1	uint16		0: Reserved. 1: Restart immediately.
Reserved	762-770	0x02FA-0x0302	0x03/0x10	9			