

# Weintek HMI to Modbus TCP Slave Devices

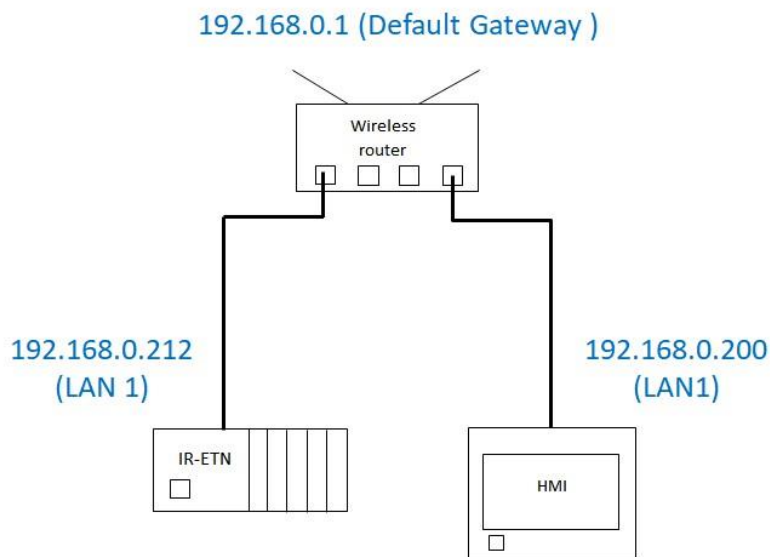


Weintek USA, Inc.  
[www.WeintekUSA.com](http://www.WeintekUSA.com)  
(425) 488-1100

Rev. JAN 28, 2020

## Weintek HMI to Modbus TCP Slave Devices

**Introduction:** This instruction manual discusses how to communicate with Modbus TCP slave devices over ethernet. The Modbus TCP protocol is widely used on many industrial sites and adopted by many manufacturers because this protocol is free, open, and simple. Modbus TCP enables master-slave communication between devices connected through ethernet. Masters queries slave devices, and the slaves only respond to the queries transmitted from the masters.



*Network Diagram*

### Equipment & Software:

1. EasyBuilder Pro v6.03.02.294
2. Weintek HMI cMT3090
3. Modbus TCP slave devices

## Weintek HMI to Modbus TCP Slave Devices

### Knowledge of Modbus TCP Protocol:

A Modbus slave device provides a Modbus master device with the following memory tables to access data.

Object Type	Access (Read-write)	Address Range	Read	Write Single	Write Multiple
Coil (Bit)	R/W	00001-09999 (0x)	FC01	FC05	FC15
Discrete input (Bit)	R	10001-19999 (1x)	FC02	N/A (Read only)	N/A (Read only)
Input register (16-bits)	R	30001-39999 (3x)	FC04	N/A (Read only)	N/A (Read only)
Holding register (16-bits)	R/W	40001-49999 (4x)	FC03	FC06	FC16

Note: FC means Modbus Function Code

The supported Modbus Function Codes vary from the manufacturers. The common function codes are shown below.

Function Code (Decimal)	Access	Access Object Type
01	Read	Coil
02	Read	Discrete input
03	Read	Holding register
04	Read	Input register
05	Write single	Coil
06	Write single	Holding register
15	Write multiple	Coil
16	Write multiple	Holding register

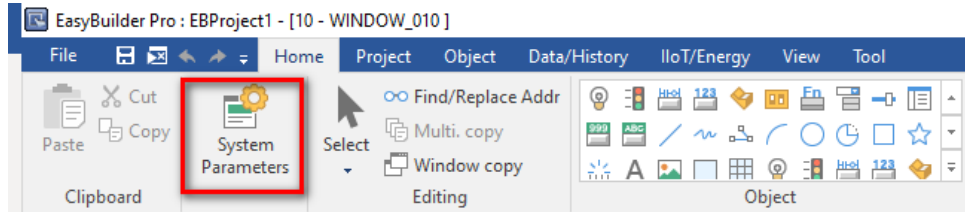
A Modbus map is a list of parameters stored in Modbus addresses. It provides the essential information for users to access data. Most slave devices are built with fixed map defined by the manufacturer. While some Modbus slave devices, such as PLCs or HMIs, allow programmers to configure custom maps. You will need to know the following information defined by your devices.

- Where is data stored? (which tables and addresses)
- How is data stored? (data types and byte, word ordering)

## Weintek HMI to Modbus TCP Slave Devices

- How do you find the range?

**Detail of the HMI Programming:** Open a new project and choose the HMI model cMT3090. To get the HMI talking to the Modbus slaves, go to the [HOME] tab on the top of the menu and then click on the [System Parameters] button.



You will need to select one of the drivers based on the specification of your devices.

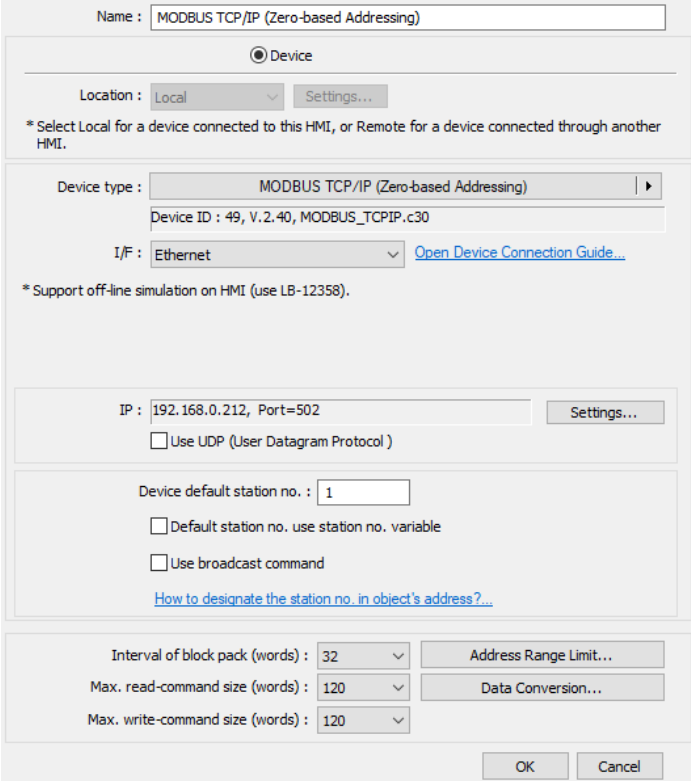
Driver Name	Description
Modbus TCP/IP	The addresses for the parameters start from 1 (1 based)
Modbus TCP/IP (Zero-based Addressing)	The addresses for the parameters start from 0 (0 based)

In this case, the **Modbus TCP(Zero-based Addressing)** driver is used.

I/F: **Ethernet**

Device default station no.: Use the default station number.

## Weintek HMI to Modbus TCP Slave Devices



Device Settings

Name : MODBUS TCP/IP (Zero-based Addressing)

Device

Location : Local

\* Select Local for a device connected to this HMI, or Remote for a device connected through another HMI.

Device type : MODBUS TCP/IP (Zero-based Addressing)

Device ID : 49, V.2.40, MODBUS\_TCPIP.c30

I/F : Ethernet

\* Support off-line simulation on HMI (use LB-12358).

IP : 192.168.0.212, Port=502

Use UDP (User Datagram Protocol)

Device default station no. : 1

Default station no. use station no. variable

Use broadcast command

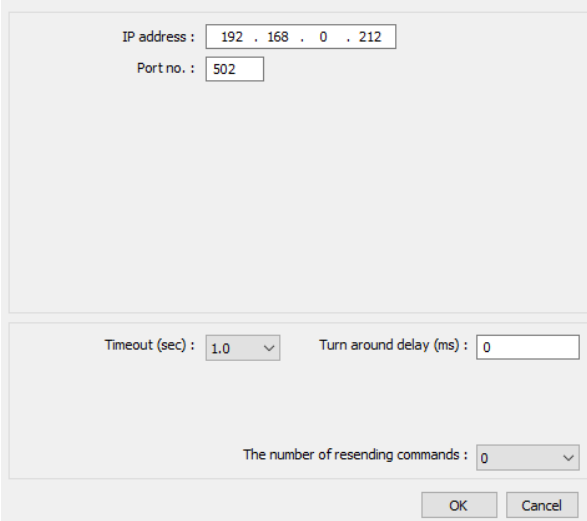
[How to designate the station no. in object's address?...](#)

Interval of block pack (words) : 32

Max. read-command size (words) : 120

Max. write-command size (words) : 120

Click on the [Settings..] button to enter the IP address of your Modbus slave device and the port number.



IP Address Settings

IP address : 192 . 168 . 0 . 212

Port no. : 502

Timeout (sec) : 1.0  Turn around delay (ms) : 0

The number of resending commands : 0

## Weintek HMI to Modbus TCP Slave Devices

Click on the [Data Conversion] button to implement **Byte swap**, **Word swap**, or **Double Word swap**.

- Each character, such as “A”, represents one byte.
- **AB->BA** does a byte swap.
- **ABCD->CDAB** does a word swap.
- You can do a byte swap and word swap with **3x\_Double** and **4x\_Double**.
- **ABCDEFGH->EFGHABCD** does a double word swap (for CMT HMIs only).
- You can do a byte swap, word swap, and double word swap with **3x\_QWord** **4x\_QWord** (for CMT HMIs only).

Since the Modbus protocol does not define exactly how data is stored in the registers. You will need to check with the manufacturer to find out which ordering format your slave device stores data.

By default the Modbus TCP master driver in Weintek HMI uses **Low byte (or word, double word)** first as ordering.

16-bit data (0x4E20)	
Low byte (0x20)	High byte (0x4E)

Low byte first

## Weintek HMI to Modbus TCP Slave Devices

32-bit data (0xAE41,5652)	
Low word (0x5652)	High word (0xAE41)

Low word first

64-bit data (0x7048,860F,9180)	
Low double word (0x860F,9180)	High double word (0x7048)

Low double word first

### Accessible device memory in EasyBuilder Pro

The Weintek HMI uses the following Modbus Function Codes.

Address	Read/Write	Use Function Code (Decimal)	Description
0x <span style="color: red;">*1</span>	R	01	Reads <b>coils</b>
	W	05	Writes a single <b>coil</b>
1x <span style="color: red;">*1</span>	R	02	Reads <b>discrete inputs</b>
0x_multi_coil	R	01	Reads <b>coils</b>
	W	15	Writes multiple <b>coils</b>
3x_Bit <span style="color: red;">*1</span>	R	04	Reads <b>input register (3x)</b> 's bit
4x_Bit <span style="color: red;">*1</span>	R	03	Reads <b>holding register (4x)</b> 's bit
	W	16	Writes multiple <b>holding register (4x)</b> 's bit
6x_Bit <span style="color: red;">*1</span>	R	03	Reads <b>holding register</b> 's bit
	W	06	Writes multiple <b>holding register (4x)</b> 's bit
3x	R	04	Reads <b>input registers</b>
4x	R	03	Reads <b>holding registers</b>
	W	16	Writes multiple <b>holding registers</b>
5x <span style="color: red;">*2</span>	R	03	Reads <b>holding registers</b>
	W	16	Writes multiple <b>holding registers</b>
6x <span style="color: red;">*3</span>	R	03	Reads <b>holding register</b>
	W	06	Writes a single <b>holding register</b>
3x_Double	R	04	Reads <b>input register (32-bit data)</b> Defaults to 32-bit numeric format
4x_Double	R	03	Reads <b>holding register (32-bit data)</b> Defaults to 32-bit numeric format
	W	16	Writes <b>holding register (32-bit data)</b> Defaults to 32-bit numeric format

## Weintek HMI to Modbus TCP Slave Devices

**\*1.** The **Modbus TCP (Zero-based Addressing)** driver reads a group of 16 bits at a time. Bit groups are 0-15, 16-31, 32-47,48-63, etc. All bits in the group must be available in the controller for the HMI to read. Otherwise, errors will result.

**\*2.** The 5x is exactly the same as the 4x. Use the 5x when reading/writing to a 32-bit registers using the low word first format. For example,

4x contains the following data,

Address	1	2
Data (word)	0x4E20	0x7530
Data (Double word)	0x075304E20	

Then use 5x instead of 4x, it will be

Address	1	2
Data (word)	0x4E20	0x7530
Data(Double word)	0x4E207530	

**\*3.** By default the Weintek HMI uses a Function Code **16** to write multiple registers, even if it is only writing to one register. The 6x forces the HMI to transmit a Function Code **06** to write a single register.

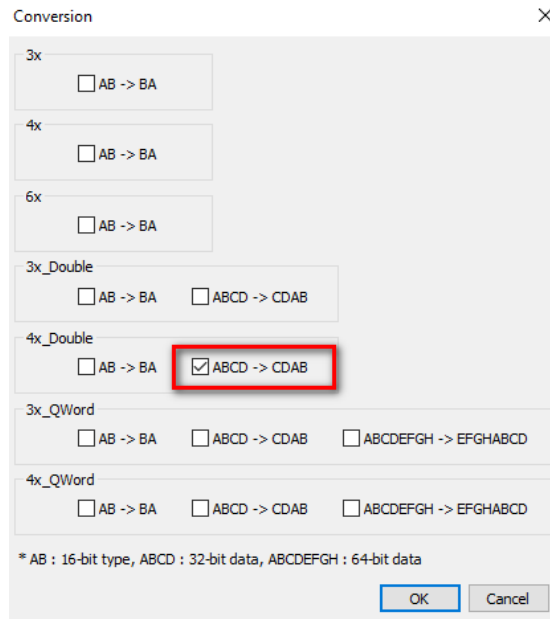
### Special device memory in EasyBuilder Pro

Address	Read/Write	Description
0x_single_bit	R/W	Reads a single <b>0x</b> bit at a time instead of a group of 16 consecutive bits
1x_single_bit	R	Reads a single <b>1x</b> bit at a time instead of a group of 16 consecutive bits

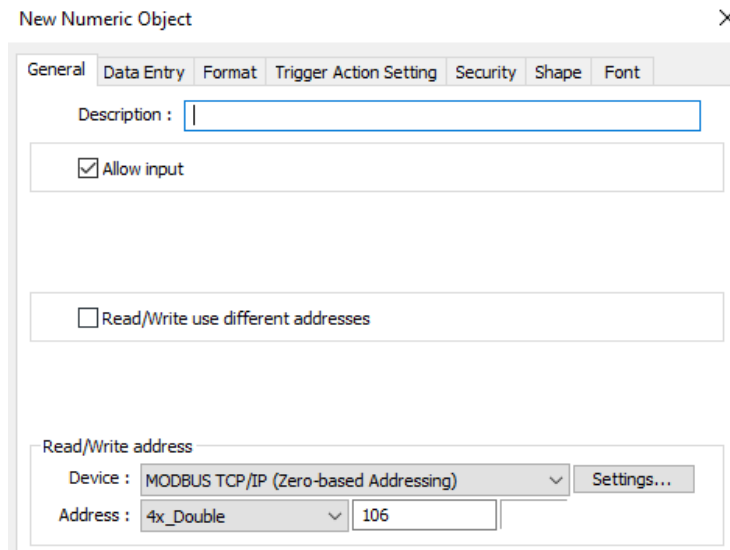
# Weintek HMI to Modbus TCP Slave Devices

## How to read/write 32-bit unsigned data

To read 32-bit unsigned data from register 40106 (combined with 40107 to generate 32-bit data) with **high word first** format, please check the **Word swap [ABCD ->CDAB]** option on [Data Conversion].



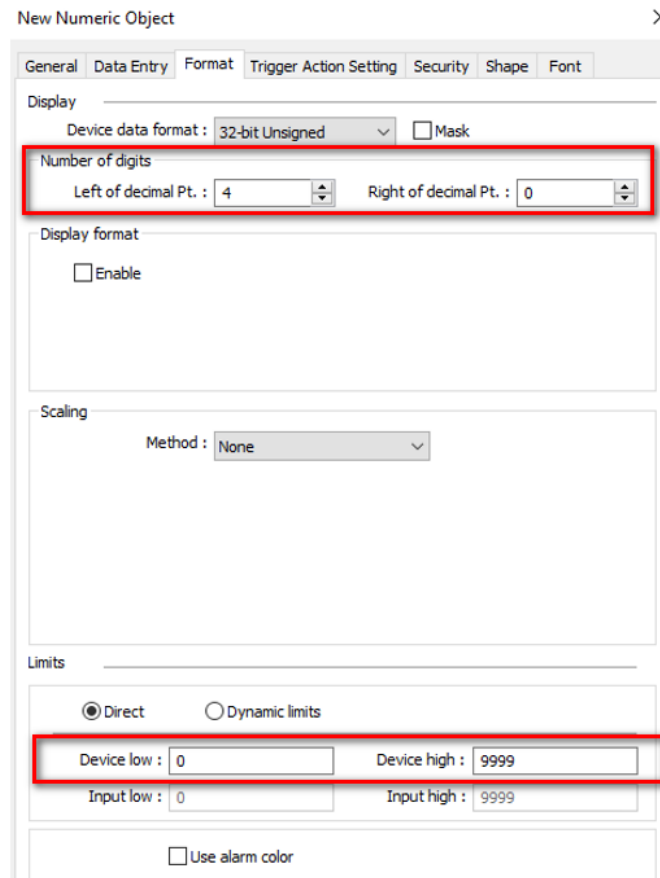
Create a Numeric object and specify the address **4x\_Double - 106** on the [General] tab as below.





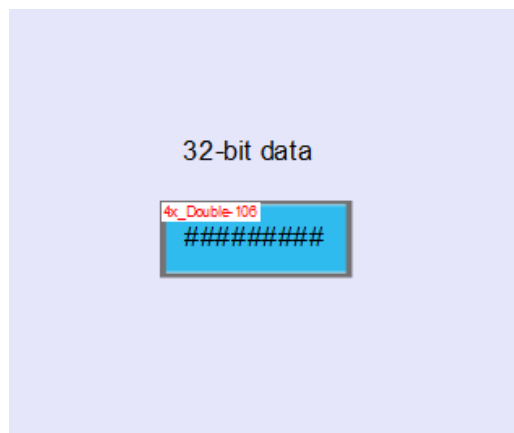
## Weintek HMI to Modbus TCP Slave Devices

Under the [Format] tab, enter the number of digits used in this parameter as well as the device's low limit and high limit. Click the [OK] button to finish setting up this object.



The screenshot shows the 'New Numeric Object' dialog box with the 'Format' tab selected. The 'Display' section is highlighted with a red box, showing 'Device data format' set to '32-bit Unsigned', 'Number of digits' set to 4, 'Left of decimal Pt.' set to 4, and 'Right of decimal Pt.' set to 0. The 'Limits' section is also highlighted with a red box, showing 'Direct' selected, 'Device low' set to 0, 'Device high' set to 9999, 'Input low' set to 0, and 'Input high' set to 9999. Other options like 'Mask', 'Enable', 'Scaling Method', and 'Use alarm color' are visible but not highlighted.

Place the Numeric object onto the editing area.



## Weintek HMI to Modbus TCP Slave Devices

### How to read/write bits in the 4x/3x memory tables

The 4x\_Bit is used to read/write to individual bits in the 4x memory table. To access a bit in 4x memory table, select the **4x\_Bit** as the Address for bit-type objects such as Bit Lamp. Under the **Address**, use the format DDDDDdd to enter the **word** memory area, followed by the two-digit bit reference.

For example, to monitor the second bit of 40030, enter "3001" into the Address. (DD=30, dd=01)

New Bit Lamp/Toggle Switch Object

General Security Shape Label

Comment : |

Bit Lamp  Toggle Switch

Read address

Device : MODBUS TCP/IP (Zero-based Addressing) Settings...

Address : 4x\_Bit 3001

Invert signal

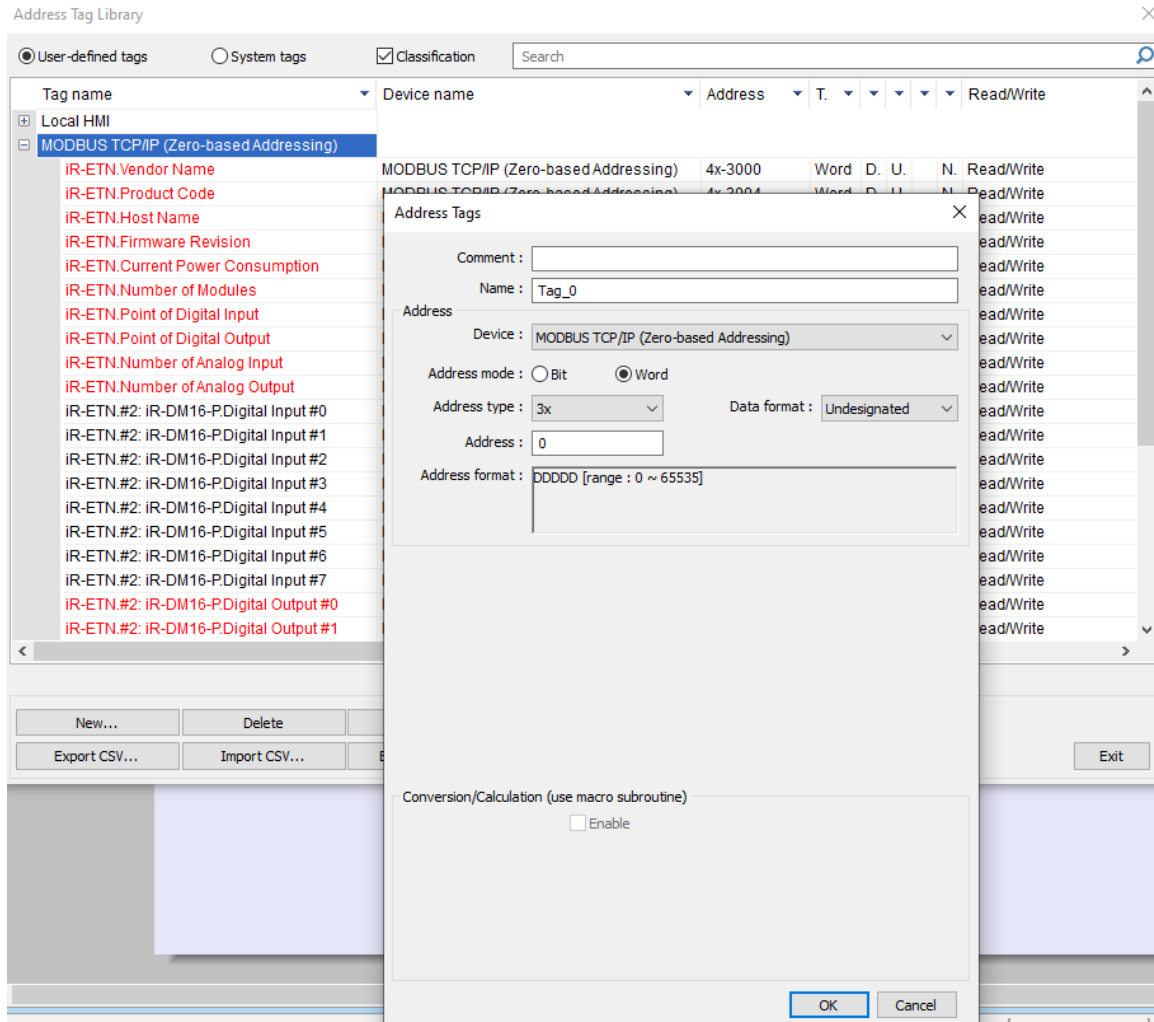
The **3x\_Bit** works the same as for the **4x\_Bit**, except that it is used for accessing bits in a 3x memory table (input register, read only).

# Weintek HMI to Modbus TCP Slave Devices

## How to create tags for the Modbus memories

You can define the parameters in the [Address Tag Library] before creating objects. It not only avoids accidental reuse of addresses but also improves project readability.

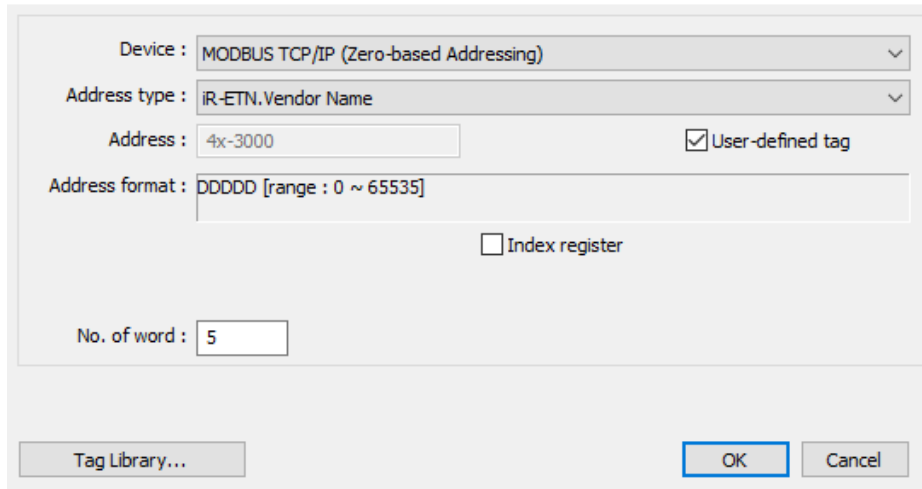
To use this feature, click the [New] button to add a parameter. Enter the address of the parameter and give a tag name. The tags used in this project will be highlighted in red.



## Weintek HMI to Modbus TCP Slave Devices

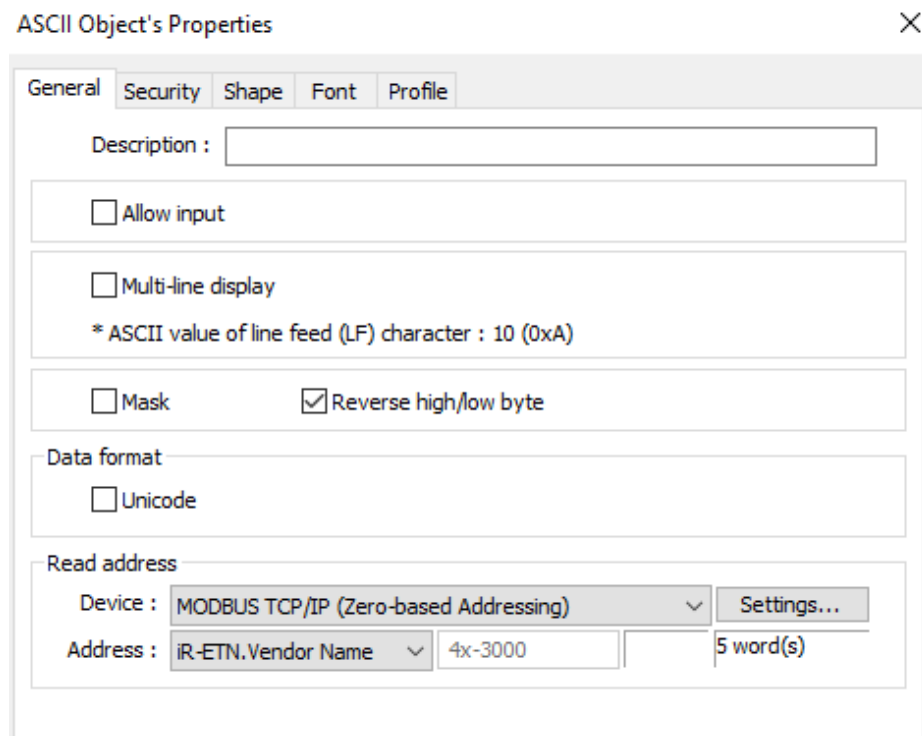
### How to read/write String data

On an **ASCII** object. Select the tag created from [Address Tag Library] and enter the number of words used in this string.



The screenshot shows a configuration dialog box for a Modbus device. The 'Device' dropdown is set to 'MODBUS TCP/IP (Zero-based Addressing)'. The 'Address type' dropdown is set to 'IR-ETN.Vendor Name'. The 'Address' text box contains '4x-3000' and the 'User-defined tag' checkbox is checked. The 'Address format' text box contains 'DDDDD [range : 0 ~ 65535]'. The 'Index register' checkbox is unchecked. The 'No. of word' text box contains '5'. At the bottom, there are buttons for 'Tag Library...', 'OK', and 'Cancel'.

Because the byte ordering format of the string data is different from the format in the HMI, select [Reverse high/low byte] is required.

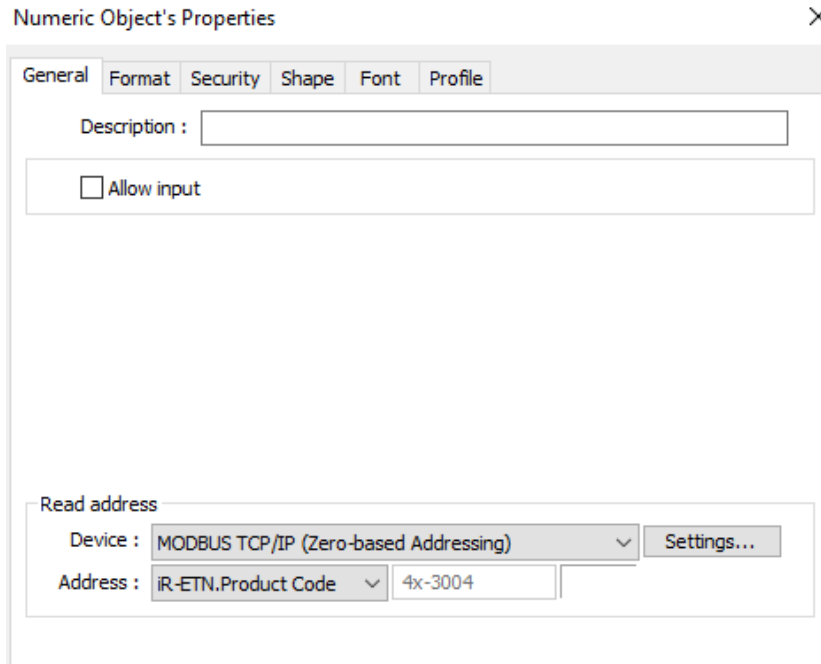


The screenshot shows the 'ASCII Object's Properties' dialog box. The 'General' tab is selected. The 'Description' text box is empty. The 'Allow input' checkbox is unchecked. The 'Multi-line display' checkbox is unchecked. The '\* ASCII value of line feed (LF) character : 10 (0xA)' is displayed. The 'Mask' checkbox is unchecked and the 'Reverse high/low byte' checkbox is checked. The 'Data format' section has the 'Unicode' checkbox unchecked. The 'Read address' section has the 'Device' dropdown set to 'MODBUS TCP/IP (Zero-based Addressing)', the 'Address' dropdown set to 'IR-ETN.Vendor Name', the 'Address' text box containing '4x-3000', and the 'Read address' text box containing '5 word(s)'. There is a 'Settings...' button next to the 'Device' dropdown.

# Weintek HMI to Modbus TCP Slave Devices

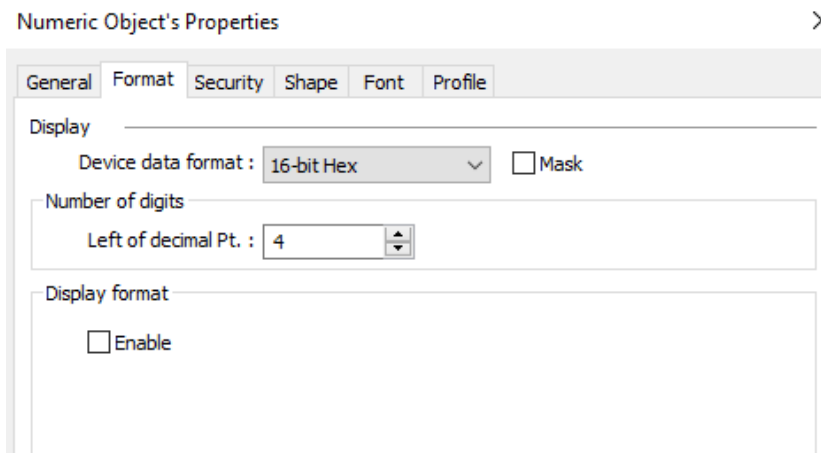
## How to read/write 16-bit data with Hex format

Create a **Numeric** object and select the tag from the [Address Tag Library].



The screenshot shows the 'Numeric Object's Properties' dialog box with the 'General' tab selected. The 'Description' field is empty. There is an unchecked checkbox for 'Allow input'. The 'Read address' section is expanded, showing 'Device' set to 'MODBUS TCP/IP (Zero-based Addressing)' and 'Address' set to 'iR-ETN.Product Code' with a value of '4x-3004'. A 'Settings...' button is visible next to the device dropdown.

On the [format] tab, select [16-bit Hex] within **Device data format**.

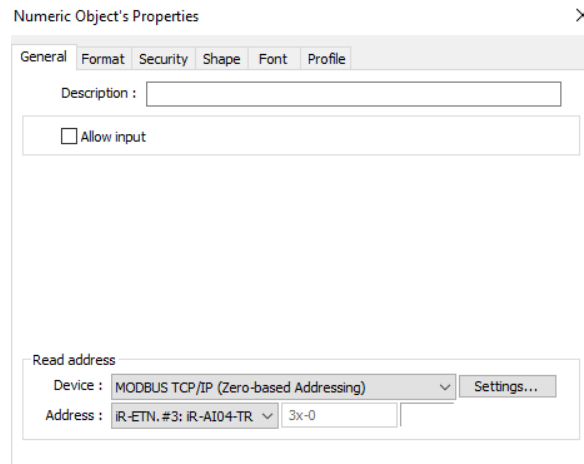


The screenshot shows the 'Numeric Object's Properties' dialog box with the 'Format' tab selected. The 'Device data format' is set to '16-bit Hex'. There is an unchecked checkbox for 'Mask'. The 'Number of digits' section is expanded, showing 'Left of decimal Pt.' set to '4'. The 'Display format' section is expanded, showing an unchecked checkbox for 'Enable'.

# Weintek HMI to Modbus TCP Slave Devices

## How to read/write 16-bit data with decimal point

Create a **Numeric** object and select the tag from the [Address Tag Library].



Numeric Object's Properties

General Format Security Shape Font Profile

Description :

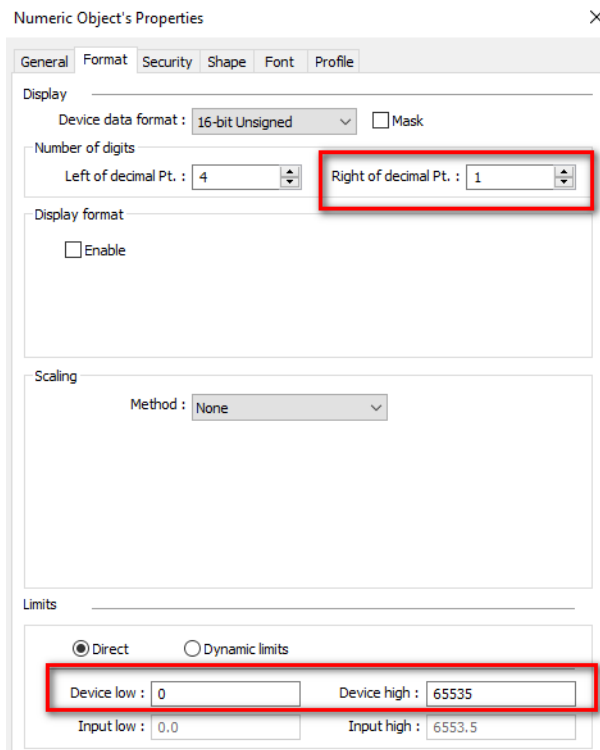
Allow input

Read address

Device : MODBUS TCP/IP (Zero-based Addressing) Settings...

Address : IR-ETN.#3: IR-A104-TR 3x-0

Because this parameter is a value that has one decimal place, enter 1 into [Right of decimal Pt.] and specify the range.



Numeric Object's Properties

General Format Security Shape Font Profile

Display

Device data format : 16-bit Unsigned  Mask

Number of digits

Left of decimal Pt. : 4 Right of decimal Pt. : 1

Display format

Enable

Scaling

Method : None

Limits

Direct  Dynamic limits

Device low : 0 Device high : 65535

Input low : 0.0 Input high : 6553.5

# Weintek HMI to Modbus TCP Slave Devices

## How to use Multi-State Switch object to set up a parameter

The unit of temperature is interchangeable based on the assigned value. (0= Celsius, 1=Fahrenheit). You will need to set up a **Multi-State Switch** object on the screen for selecting the unit. On the [General] tab, enter the number of states used in this parameter within **Attribute**. Select [JOG+] and enable [Cyclical].

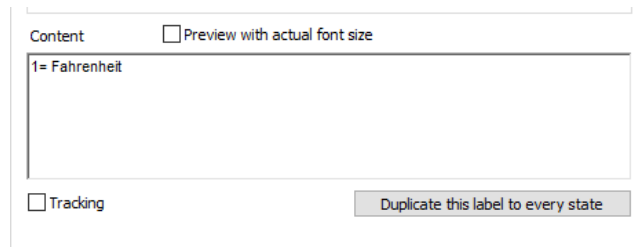
The screenshot shows the 'Multi-State Switch/Word Lamp Object's Properties' dialog box with the 'General' tab selected. The 'Comment' field is empty. The 'Word Lamp' radio button is unselected, and the 'Multi-State Switch' radio button is selected. The 'Mode' dropdown is set to 'Value'. The 'Offset' field contains '0'. The 'Read/Write use different addresses' checkbox is unselected. The 'Read/Write address' section shows 'Device' as 'MODBUS TCP/IP (Zero-based Addressing)' and 'Address' as 'iR-ETN.#3: iR-AI04-TR' with a value of '4x-21019' and '16-bit Unsigned'. The 'Write when button is released' checkbox is unselected. The 'Error handling...' button is visible. The 'Attribute' section shows 'Switch style' as 'JOG+', 'No. of states' as '2', and 'Cyclical' as 'Enable'. The 'User-defined mapping' checkbox is unselected. The 'Send notification after writing successfully' checkbox is unselected.

On the [Label] tab, the state 0 is labelled "0=Celsius."

The screenshot shows the 'Label' tab of the dialog box. The 'Content' field contains the text '0 = Celsius'. The 'Preview with actual font size' checkbox is unselected. The 'Tracking' checkbox is unselected. The 'Duplicate this label to every state' button is visible.

## Weintek HMI to Modbus TCP Slave Devices

The state 1 is labelled "1=Fahrenheit."



The screenshot shows a configuration window for a label. At the top left, the word "Content" is displayed. To its right is a checkbox labeled "Preview with actual font size". Below this, a rectangular text box contains the text "1= Fahrenheit". At the bottom left of the window is a checkbox labeled "Tracking". At the bottom right is a button labeled "Duplicate this label to every state".



# Weintek HMI to Modbus TCP Slave Devices

Screen shot of the Final Project shows the parameters pulled out from the Modbus TCP slave.

The screenshot displays the following parameters and status indicators:

- Device vender Name:** Weintek
- Device product code:** 0702
- Device name:** iR-ETN
- Device firmware version:** 1010
- Device current consumption:** 2.725
- Number of IO modules:** 3
- Number of DI:** 16
- Number of AI:** 4
- Number of DO:** 16
- Number of AO:** 0
- Slot 2:** Eight digital output (DO) indicators labeled DO-0 through DO-7, all currently dark.
- Slot 3:** Four analog input (AI) indicators labeled AI-0 through AI-3. AI-0 shows 70.9, AI-1 shows 0.0, AI-2 shows 0.0, and AI-3 shows 0.0.
- Unit of temperature:** 1 = Fahrenheit

You can toggle the digital outputs and monitor the analog inputs (temperature signal). You can select Celsius as the unit of temperature on the HMI screen.

The screenshot displays the following parameters and status indicators:

- Device vender Name:** Weintek
- Device product code:** 0702
- Device name:** iR-ETN
- Device firmware version:** 1010
- Device current consumption:** 2.725
- Number of IO modules:** 3
- Number of DI:** 16
- Number of AI:** 4
- Number of DO:** 16
- Number of AO:** 0
- Slot 2:** Eight digital output (DO) indicators labeled DO-0 through DO-7. DO-0 and DO-2 are currently lit blue, while DO-1, DO-3, DO-4, DO-5, DO-6, and DO-7 are dark.
- Slot 3:** Four analog input (AI) indicators labeled AI-0 through AI-3. AI-0 shows 21.5, AI-1 shows 0.0, AI-2 shows 0.0, and AI-3 shows 0.0.
- Unit of temperature:** 0 = Celsius

## Weintek HMI to Modbus TCP Slave Devices



Founded in 1996, WEINTEK LABS is a global-leading HMI manufacturer and is dedicated to the development, design, and manufacturing of practical HMI solutions. WEINTEK LAB's mission is to provide quality, customizable HMI-solutions that meet the needs of all industrial automation requirements while maintaining customer satisfaction by providing "on-demand" customer service. WEINTEK LABS brought their innovative technology to the United States in 2016, WEINTEK USA, INC., to provide quality and expedient solutions to the North American industrial market.

6219 NE 181s Street STE 120  
Kenmore, WA 98028  
425-488-1100