

How to Configure KingSCADA with MGate MB3000

Moxa Technical Support Team
support@moxa.com

Contents

- 1 Application Description 2**
- 2 System Topology..... 2**
- 3 Hardware and Software Requirements 3**
- 4 About KingSCADA 3.1..... 3**
 - 4.1 About KingIO Server 4
 - 4.2 About KingSCADA 4
 - 4.3 KingScada System Architecture..... 5
- 5 KingSCADA Configuration 6**
 - 5.1 Creating an IO Server Project 6
 - 5.2 Creating a SCADA Project.....12
 - 5.3 Building Tags13
 - 5.4 Creating HMI View.....22
 - 5.5 Adding a Script26
 - 5.6 Compiling a Project27
- 6 Runtime Test..... 28**

Copyright © 2014 Moxa Inc.

Released on December 24, 2014

About Moxa

Moxa is a leading manufacturer of industrial networking, computing, and automation solutions. With over 25 years of industry experience, Moxa has connected more than 30 million devices worldwide and has a distribution and service network that reaches customers in more than 70 countries. Moxa delivers lasting business value by empowering industry with reliable networks and sincere service for automation systems. Information about Moxa’s solutions is available at www.moxa.com. You may also contact Moxa by email at info@moxa.com.

How to Contact Moxa

Tel: +886-2-8919-1230
Fax: +886-2-8919-1231



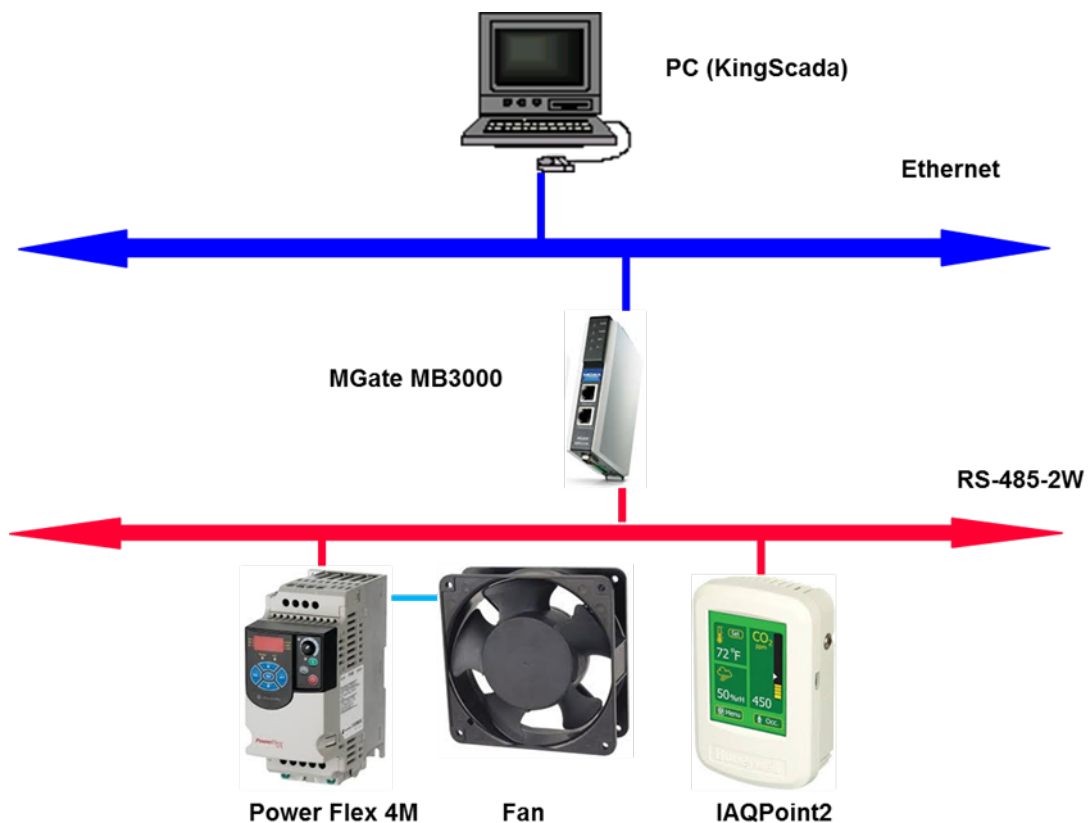
1 Application Description

This application shows how to set up a **KingSCADA** system to communicate with serial IO devices via an **MGate MB3000** Gateway. KingSCADA uses its **Modbus TCP Driver** to send MBTCP commands to MGate MB3000, which then transmits this command to the serial IO devices and returns command responses to KingSCADA.

In this application, the serial IO devices are the temperature meter and converter. The temperature meter detects the environment temperature levels. The converter is an adjustable frequency AC drive that is used to power a fan for cooling.

2 System Topology

The following figure shows a system topology where the Modbus end devices, **PowerFlex 4M** and **IAQPoint2**, are connected to the serial port on **MGate MB3000** through RS-485-2W wiring. **PC** (with **KingSCADA** installed) is connected to an Ethernet port on **MGate MB3000**. A **fan** is connected to **PowerFlex 4M** which outputs electric current to power the fan.



3 Hardware and Software Requirements

- **PowerFlex 4M:**
PowerFlex 4M is an adjustable frequency AC drive (converter).
- **IAQPoint2:**
IAQPoint2 is an indoor air quality monitor. It can detect CO2, temperature and humidity levels.
- **KingSCADA:**
A SCADA system released by WellinTech.
Rev.: V3.1.

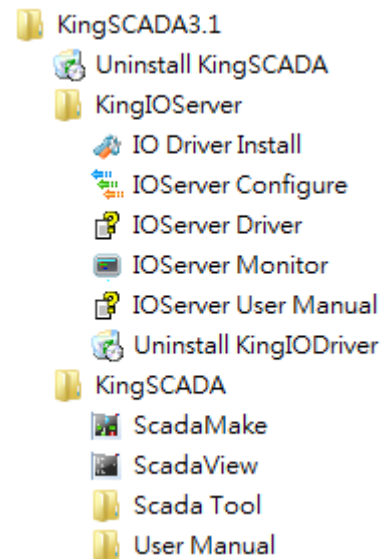
For detailed information, refer to the following tech notes:

- Configuring Allen-Brandly ControlLogix PLC with Moxa MGate 5105-MB-EIP
- How to Configure Pro-face HMI with Allen-Bradley PLC
- How to Configure KingSCADA with MGate 5105

4 About KingSCADA 3.1

KingSCADA 3.1 consists of the following components:

- **KingIOServer:** This component acquires data from I/O devices.
- **KingScada:**
 - **ScadaMake:** This is the development environment.
 - **ScadaView:** This is the runtime application.



The following sections describe these components.

4.1 About KingIO Server

A KingSCADA station communicates with I/O devices through KingIOServer. KingIOServer is used to communicate with the on-site devices and acquire real-time data and control on-site data of the modules.

KingIOServer supports popular PLC, intelligent module, intelligent instrument, transducer, and data acquisition boards, etc.

In addition, KingIOServer can communicate with devices through standard communicate interface to transfer data.

With KingIOServer, site engineers are not required to be familiar with the codes and device communication protocols. Instead, they only need to know how to connect with I/O devices and create tags corresponding with the I/O variables.

4.2 About KingSCADA

You can use KingSCADA is to **create a project with data and display**.

The following lists the major steps to create a project:

Step1: Create a new project

Create a new directory to store the documents associated with the project.

Step2: Configure the hardware

Configure the hardware settings of the equipment used in the project.

Step3: Define variables (tags)

Define global variables including memory variables and I/O devices.

Step4: Create graphics:

Draw monitoring pictures according to the project requirements.

Step5: Define animation links

Based on the on-site monitoring requirements, define the animation effects for static pictures to simulate process control objects.

Step6: Write an event script

Create scripts in order to complete the control process.

Step7: Configure of other necessary functions

Configure settings such as networks, recipes, SQL access, and web browsing.

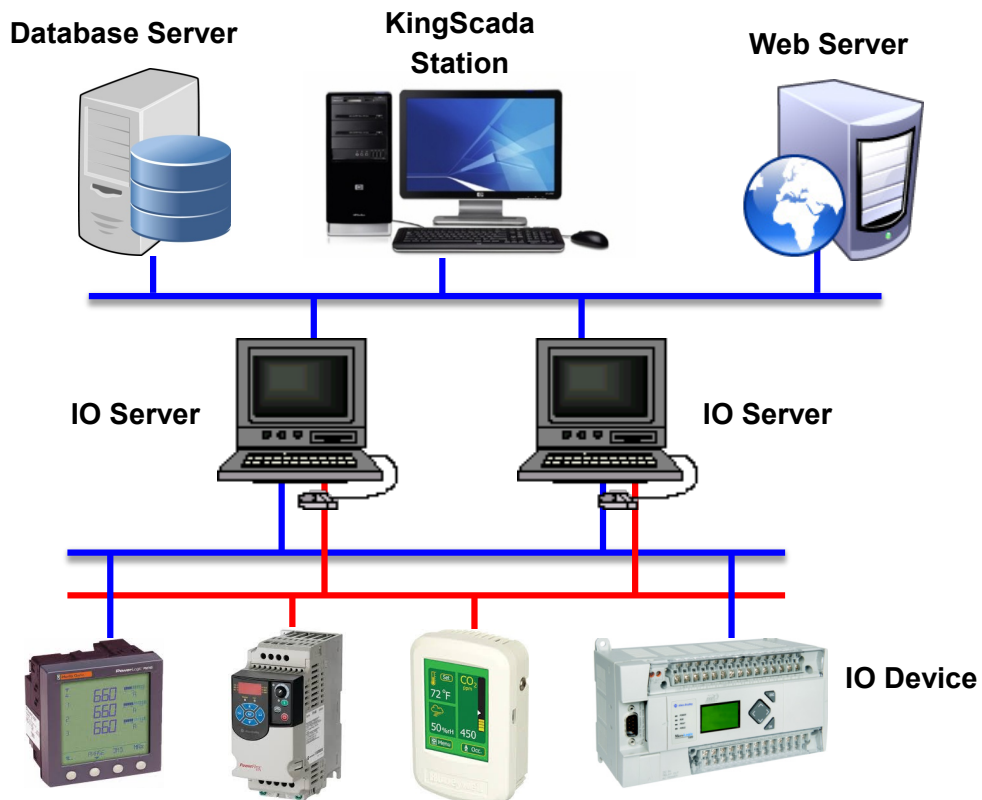
Step8: Operate and debug the project.

After you complete the procedure as described, you can create a simple project. Then, you can run **ScadaView** to start run-time operations.

4.3 KingScada System Architecture

In a large system, the KingScada system may deploy some services on multiple servers for load sharing or for security considerations. For example, a run-time project can be executed on KingScada Station for monitoring, on **HMI View** for controlling, and on another **Database Server** for data acquisition and storage. In addition, the web portal can be hosted on another **Web Server** and data can be obtained through several **IO Servers**.

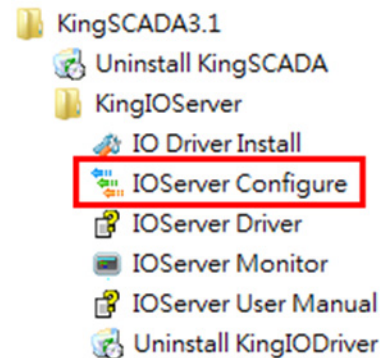
In this document, we show how to create an IO server and Run-Time View on the same PC.



5 KingSCADA Configuration

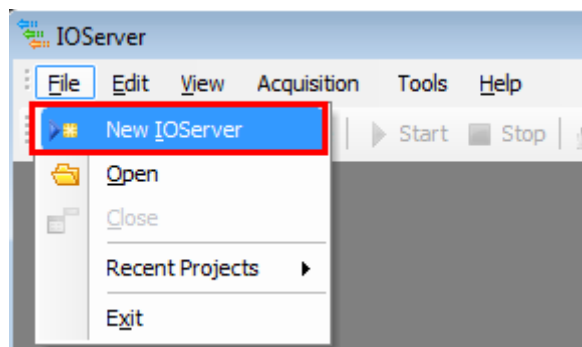
5.1 Creating an IO Server Project

To start the **IOServer Configure** application, click **Start → Program → KingSCADA3.1 → KingIOServer → IO Server Configure** to create an **IO Server** project.

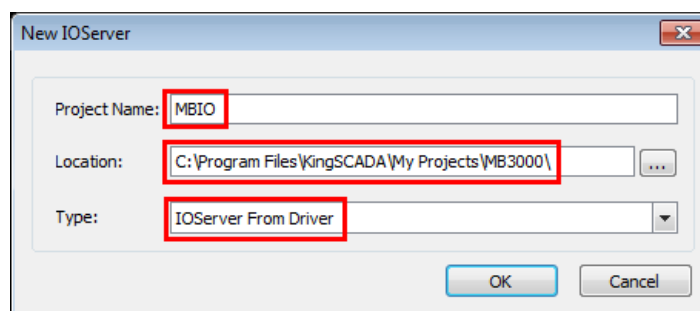


5.1.1 Creating an IO Server

1. In the IOserver screen, click **File → New IOserver** to create a new IO server.

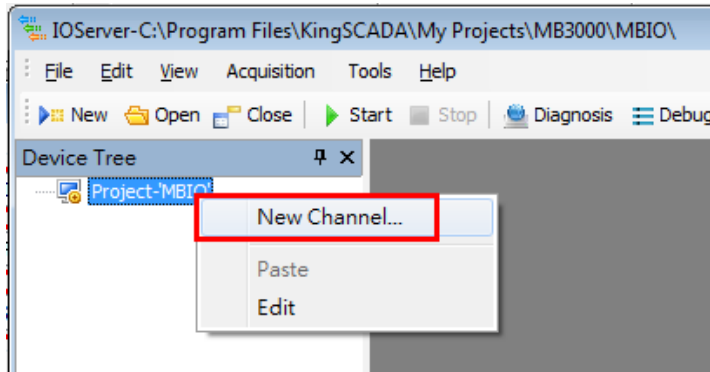


2. In the New IOserver screen, configure the following fields and click **OK**:
 - **Project Name:** Enter a descriptive name.
 - **Location:** Click the ... button to choose a location to store the project.
 - **Type:** Select IOserver From Driver from the drop-down list.

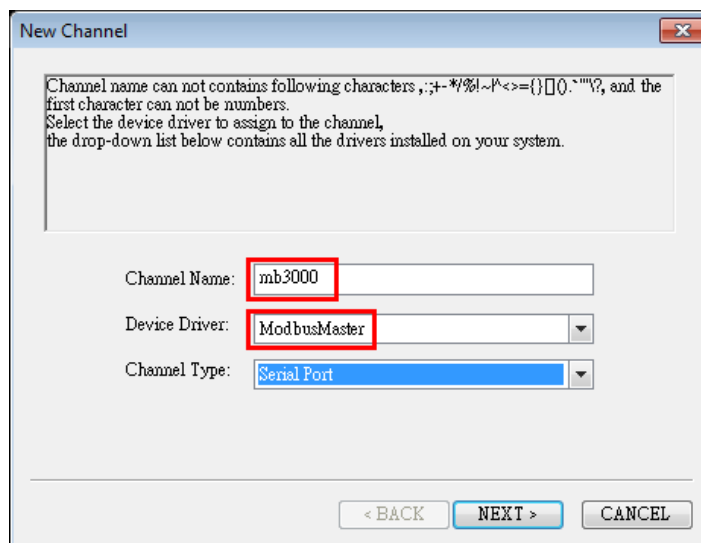


5.1.2 Creating an IO Channel

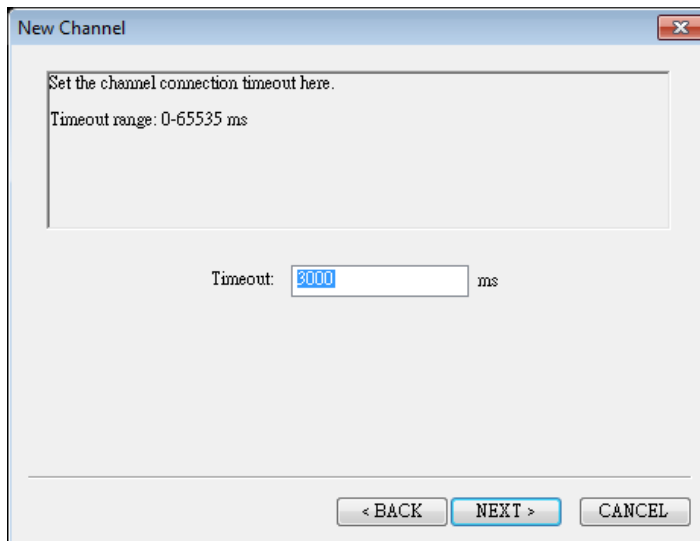
1. Right-click **Device Tree** → **Project-'MBIO'** and select **New Channel**. A **New Channel** dialog box appears.



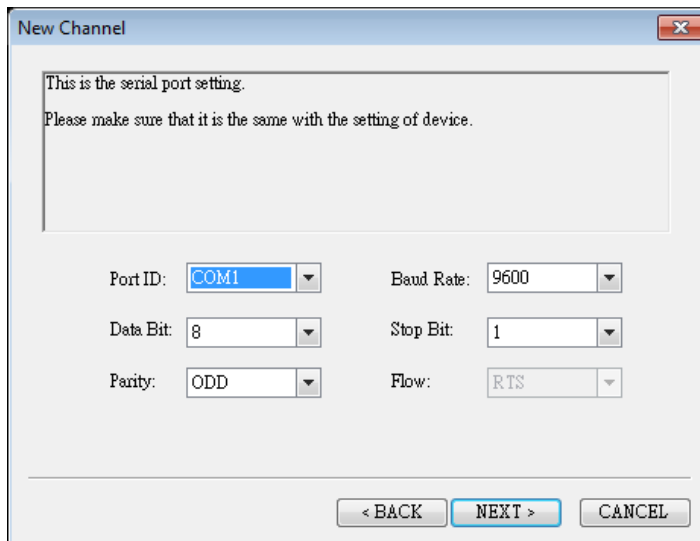
2. In the New Channel dialog box, configure the following fields and click **NEXT**.
 - **Channel Name:** Enter a descriptive name.
 - **Device Driver:** Select **ModbusMaster** from the drop-down list.
 - **Channel Type:** Use the default option.



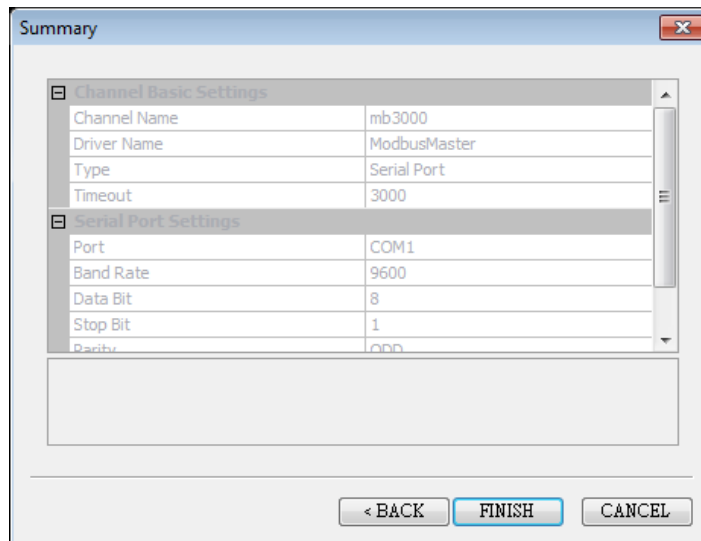
3. Accept the default timeout setting (3000 ms) and click **NEXT**.



4. Since this channel is connected through Ethernet, use the default settings for the serial port. Click **NEXT** to continue.

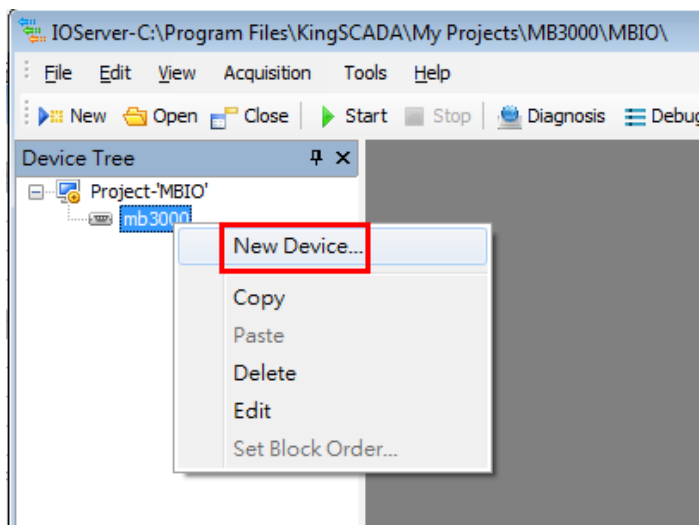


- Click **Finish** to complete the mb3000 channel setting.

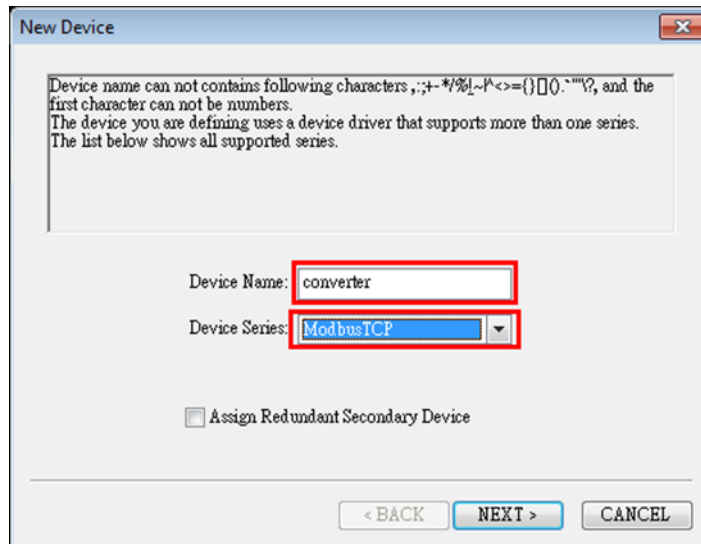


5.1.3 Creating an IO Device

- Right-click the **mb3000** channel and select **New Device**.



- In the **New Device** screen that appears, configure the following fields and click **NEXT**:
 - Device Name:** Enter a descriptive name.
 - Device Series:** Select **ModbusTCP** from the drop-down list.



Device name can not contains following characters , ; - * / % ! ~ ^ < > = () [] 0 . ^ ~ ~ ~ ~ ? , and the first character can not be numbers.
The device you are defining uses a device driver that supports more than one series.
The list below shows all supported series.

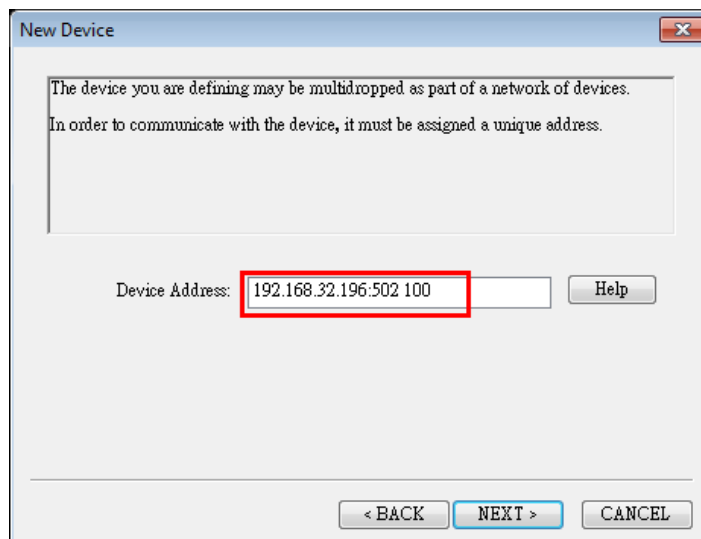
Device Name:

Device Series:

Assign Redundant Secondary Device

< BACK NEXT > CANCEL

- In the **Device Address** field, enter the IP address and communication parameter of **MGate MB3000** in the format [*IP:TCP Port SlaveID*]. Then, click **NEXT**.

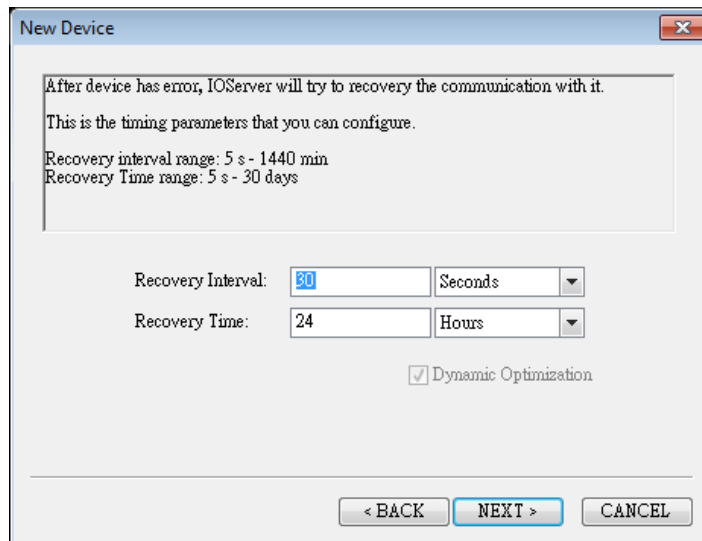


The device you are defining may be multidropped as part of a network of devices.
In order to communicate with the device, it must be assigned a unique address.

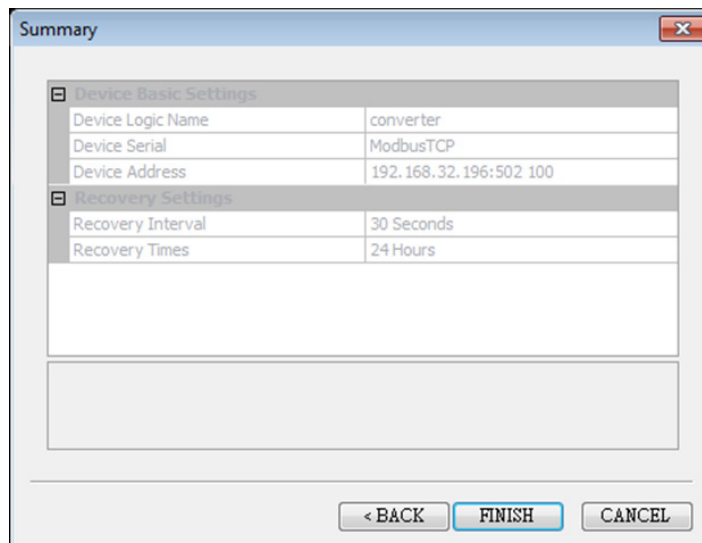
Device Address: Help

< BACK NEXT > CANCEL

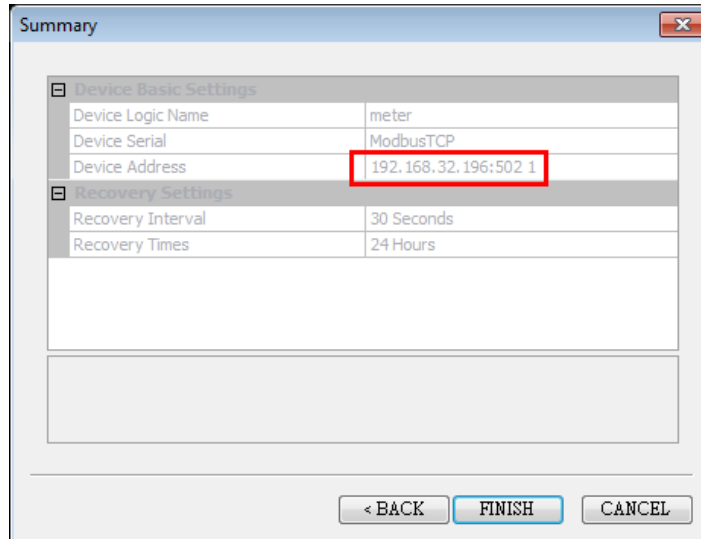
- 4. Accept the default settings and click **NEXT** to continue.



- 5. Click **Finish** to complete the device setting.

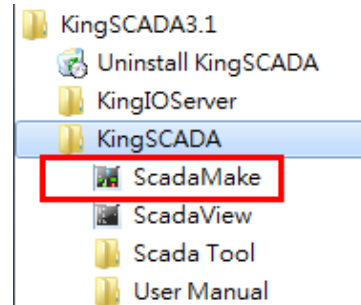


- Repeat steps 1 to 5 to create a **meter** IO device. Configure the fields as shown in the following figure.

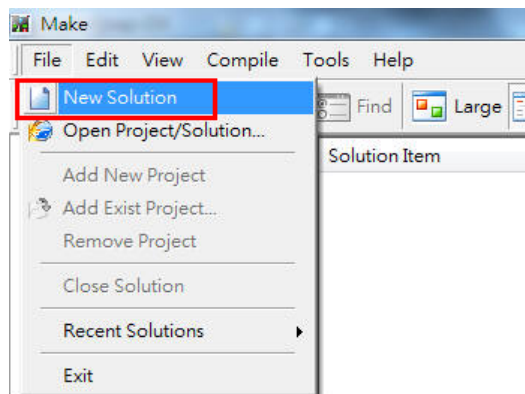


5.2 Creating a SCADA Project

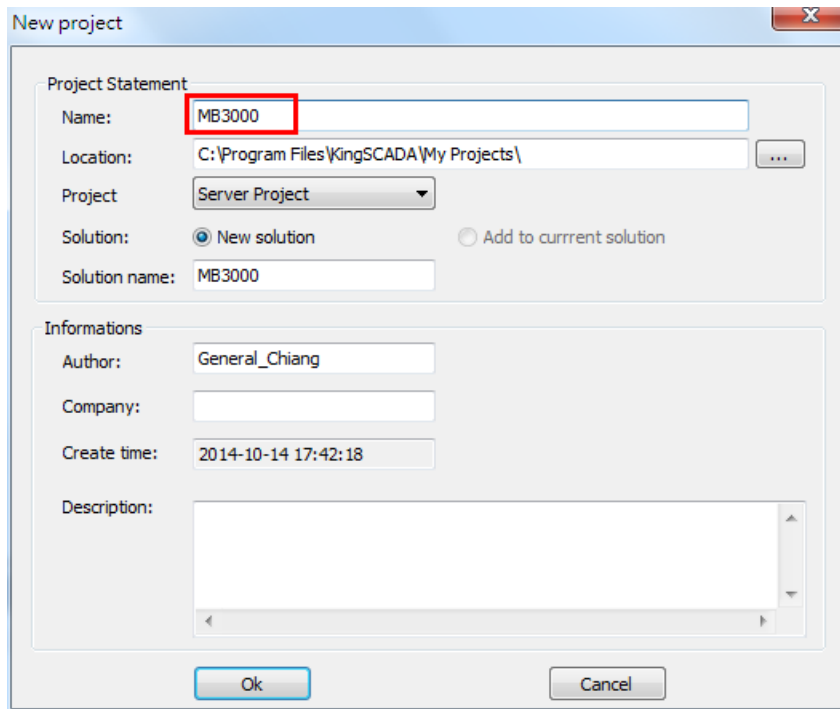
- To start the **ScadaMake** application, click **Start → Program → KingSCADA3.1 → ScadaMake**.



- Click **File → New Solution** to create a new solution.

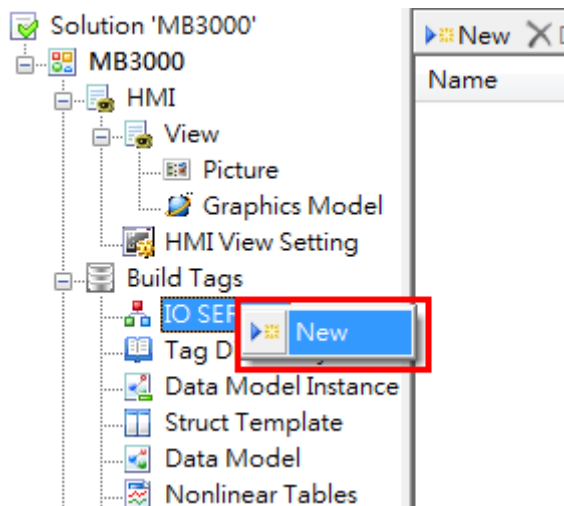


3. Give a project name then click **OK**. System would create a new empty project.

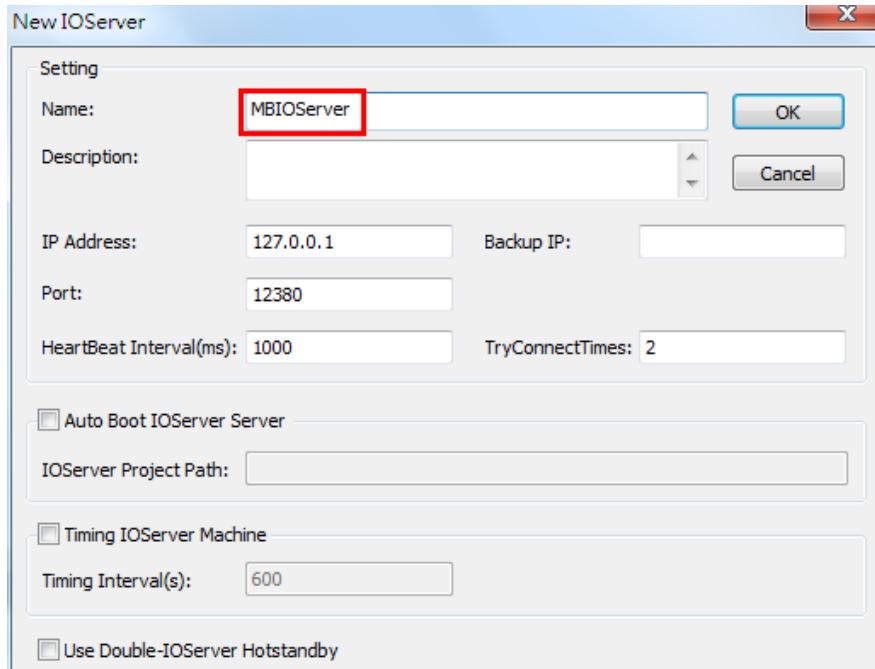


5.3 Building Tags

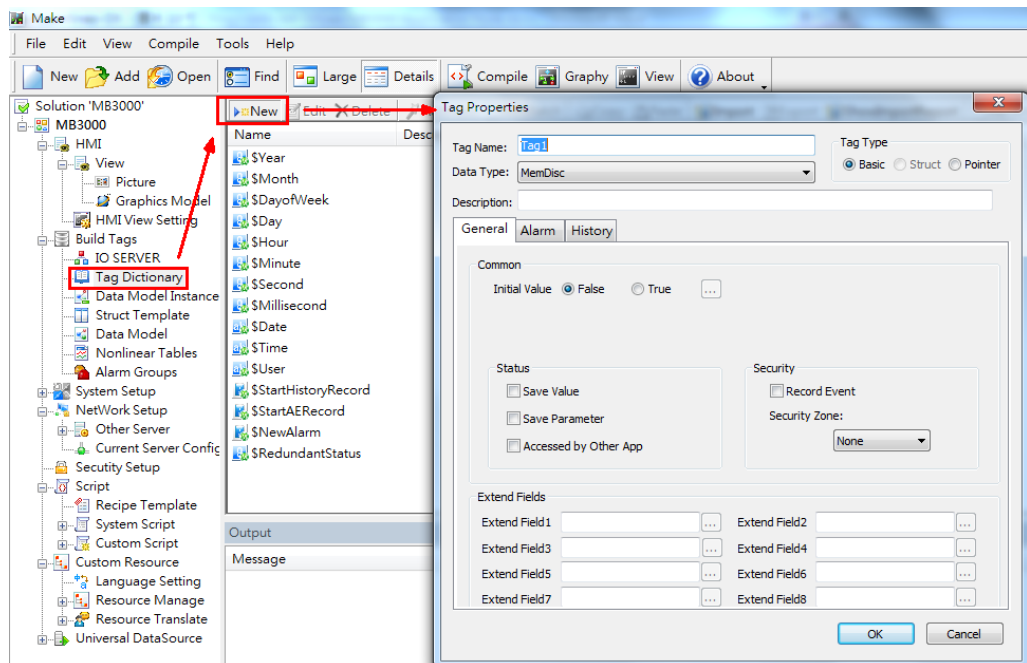
1. Under **Build Tags**, right-click **IO SERVER** and click **New**. The New IOserver screen appears.



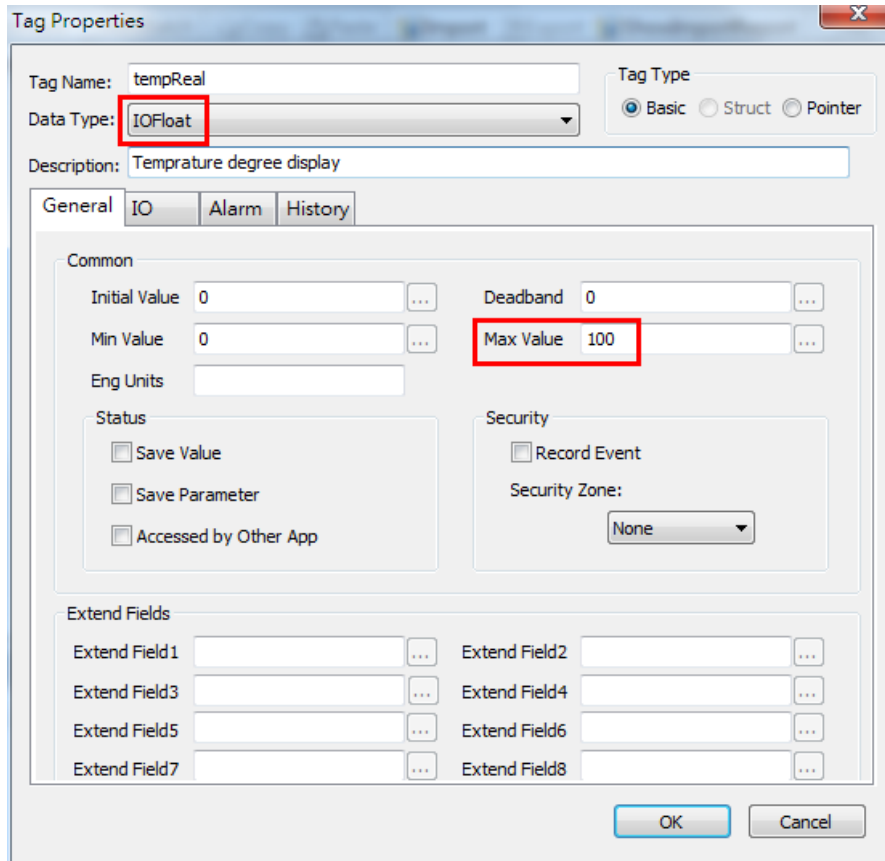
- In the **New IO Server** screen, enter a server name in the **Name** field and click **OK**.
Note: Since this IO Server is hosted on the same computer, the IP address is unchanged at 127.0.0.1.



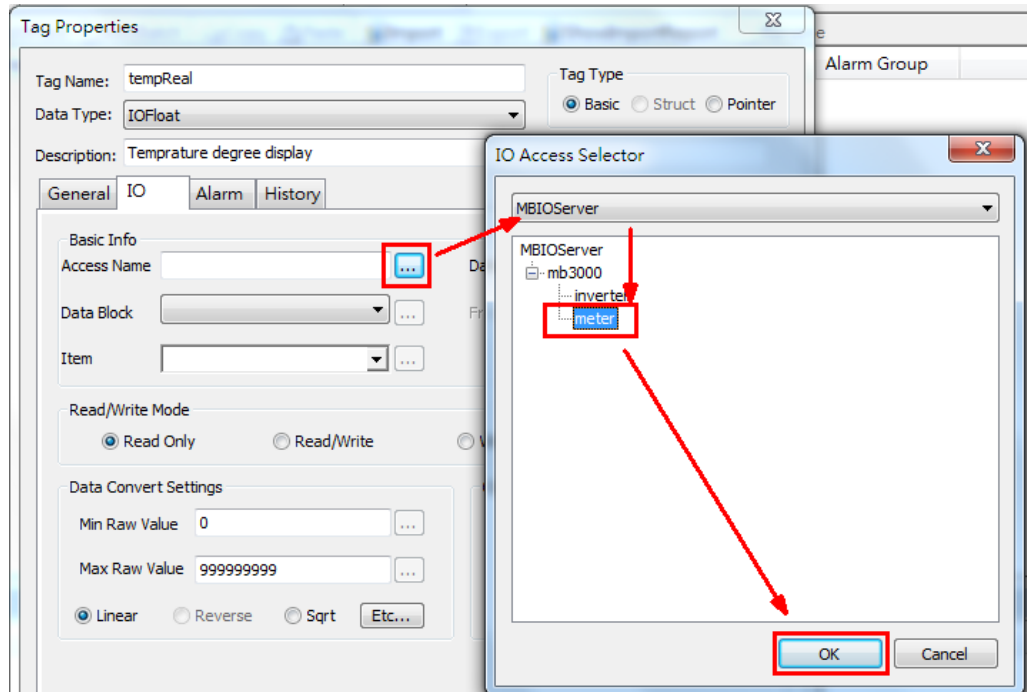
- Click **Build Tags → Tag Dictionary** and click **New**. The Tag Properties screen appears.



4. In the Tag Properties screen, configure the following fields:
 - **Tag Name:** Enter "tempReal".
 - **Data Type:** Select **IOFloat** from the drop-down list.
5. Select the **General** tab and enter "100" in the **Max Value** field.



- 6. Select the **IO** tab. For **Access Name**, click the ... button to select the **meter** IO device (in the IO Access Selector pop-up window, click **MBIOServer** → **mb3000** → **meter**).



7. In the IO tab, configure the following fields and click **OK** to finish:
- **Data Block:** Select **None** from the drop-down list.
 - **Item:** Enter "40103" as the Modbus register address.
 - **Data type:** Select **USHORT** from the drop-down list.
 - **Frequency:** Enter "500".
 - **Max Raw Value:** Enter "1000". In the General tab, the Max Value field is set to 100. Thus the Data Convert setting is defined as register address * Max Value / Max Raw Value (in this example, 40103 *100/1000).

The screenshot shows the 'Tag Properties' dialog box with the 'IO' tab selected. The 'Basic Info' section contains the following fields:

- Tag Name: tempReal
- Data Type: IOFloat
- Description: Temperature degree display
- Access Name: MBIOServer.mb3000.meter
- Data type: USHORT
- Data Block: None
- Frequency: 500 ms
- Item: 40103

The 'Read/Write Mode' section has the 'Read Only' radio button selected.

The 'Data Convert Settings' section has the 'Linear' radio button selected, and the 'Max Raw Value' field is set to 1000.

The 'Collect Settings' section has the 'Enabled' checkbox checked.

Buttons for 'OK' and 'Cancel' are located at the bottom right of the dialog.

- Repeat steps 1 to 7 to create a **speedReal** tag to read Modbus register address **48452** from **MBIOServer.mb3000.converter**.

Configure the fields as shown in the following figure. Enter "1000" in the **Max Raw Value** field. In the General tab, the Max Value field is set to "100". Thus, **Data Convert** is defined as register address * *Max Value* / *Max Raw Value* (in this example, 48452 *100/1000).

Tag Properties

Tag Name: speedReal

Data Type: IOFloat

Tag Type: Basic Struct Pointer

Description:

General IO Alarm History

Basic Info

Access Name: MB3000.mb3000.converter

Data type: USHORT

Data Block: None

Frequency: 10 ms

Item: 48452

Read/Write Mode

Read Only Read/Write Write Only

Data Convert Settings

Min Raw Value: 0

Max Raw Value: 1000

Linear Reverse Sqrt Etc...

Collect Settings

Enabled

Force Read

Force Write

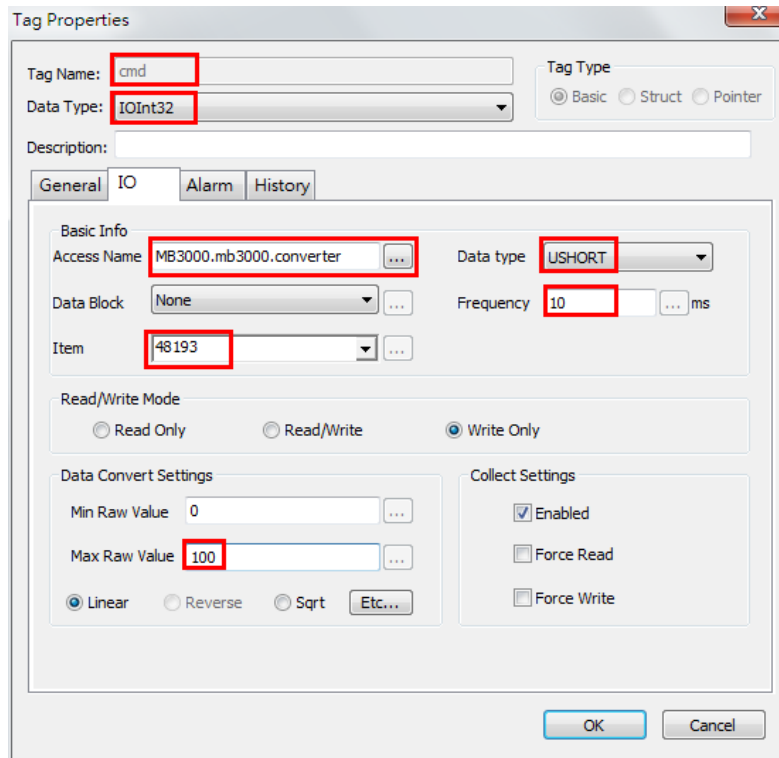
OK Cancel

- Repeat steps 1 to 7 to create a **outSpeedReal** tag to read/write Modbus register address **48194** from/to **MBIOServer.mb3000.converter**.

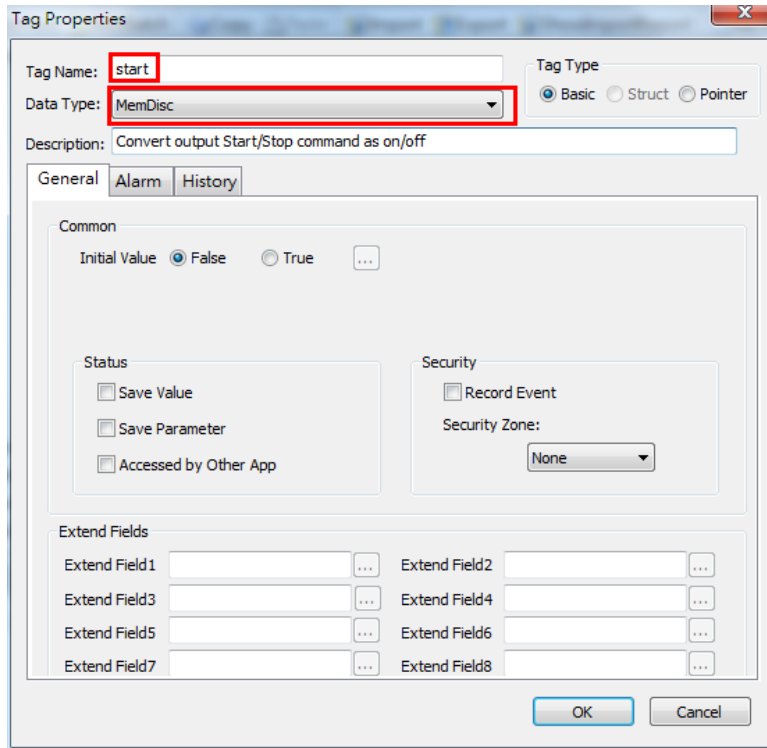
Configure the fields as shown in the following figure. Enter "1000" in the **Max Raw Value** field. In the General tab, the Max Value field is set to "100". Thus, **Data Convert** is defined as register address * Max Value / Max Raw Value (in this example, $48194 * 100 / 1000$).

- Repeat steps 1 to 7 to create a **cmd** tag as the **IOInt32** data type to write Modbus register address **48193** to **MBIOServer.mb3000.converter**.

Configure the fields as shown in the following figure. In the IO tab, select **USHORT** from the **Data type** drop-down list and enter "100" in the **Max Raw Value** field. In the General tab, the Max Value field is set to "100". Thus, the system does not convert the register data.



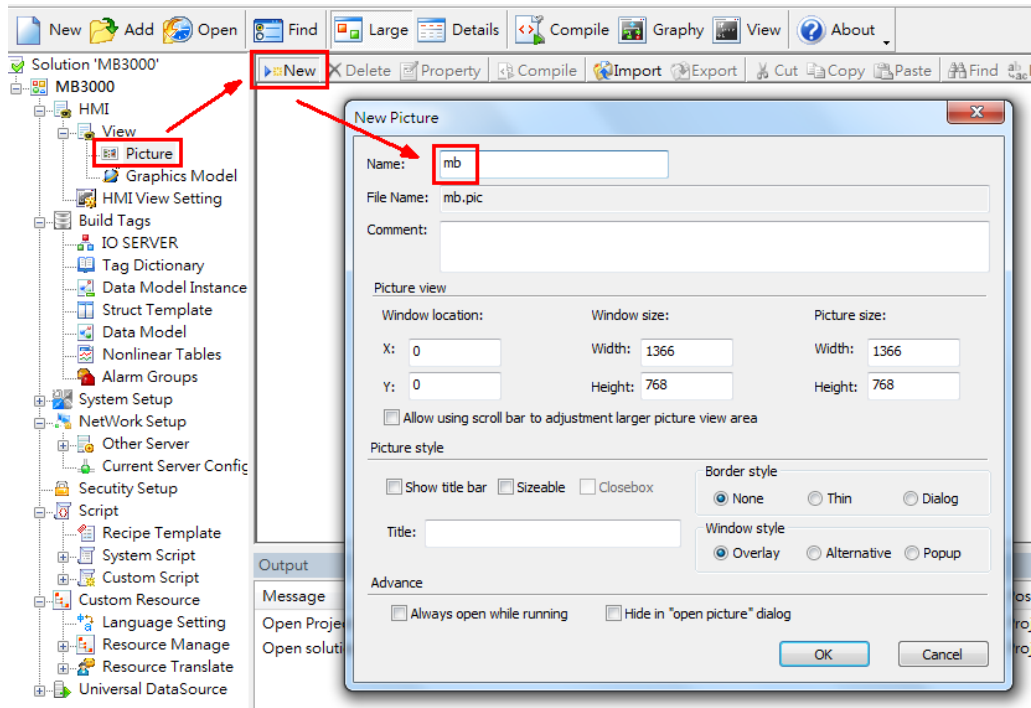
11. Repeat steps 1 to 7 to create a **MemDisc** tag named **start** to store the command status that the system converts from the **cmd** tag value.



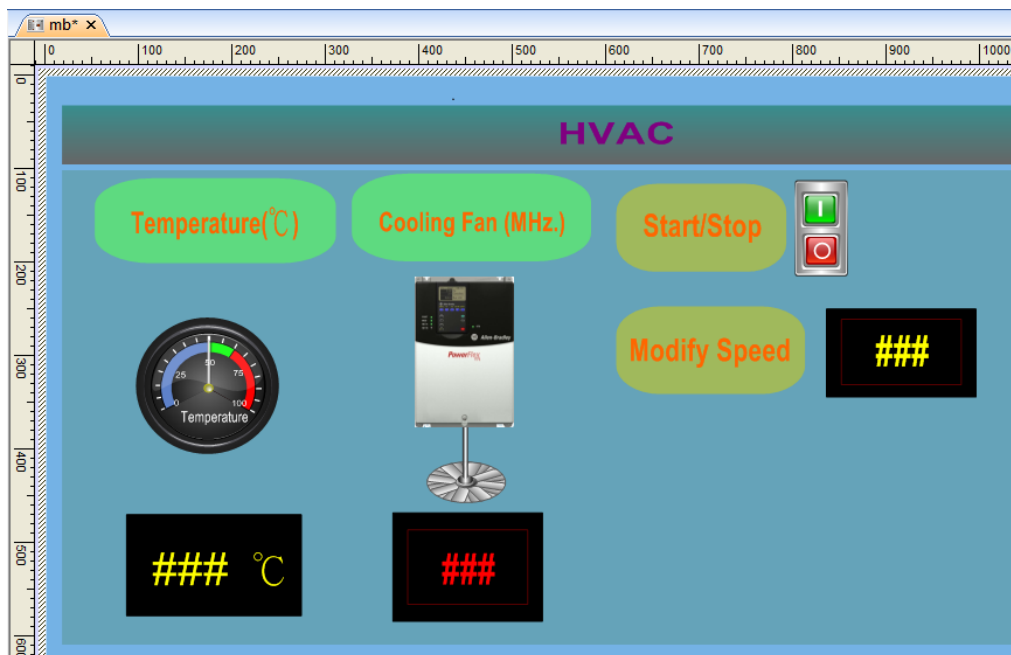
5.4 Creating HMI View

5.4.1 Creating a View Picture

1. Click **HMI** → **View** → **Picture** and click **New** to create a view picture.

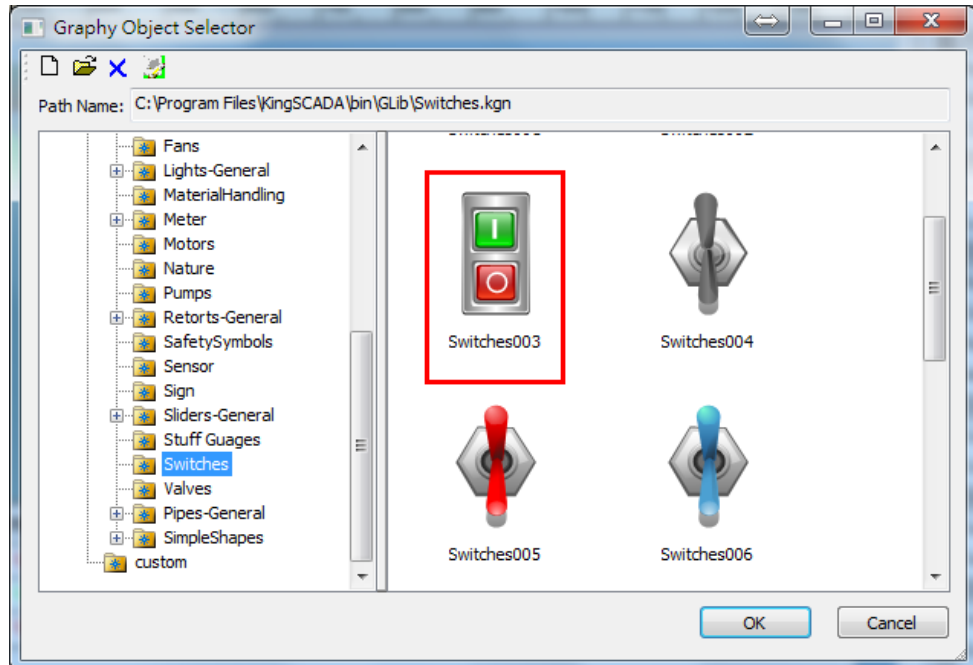


2. After adding object graphics, the following figure shows the complete view picture.

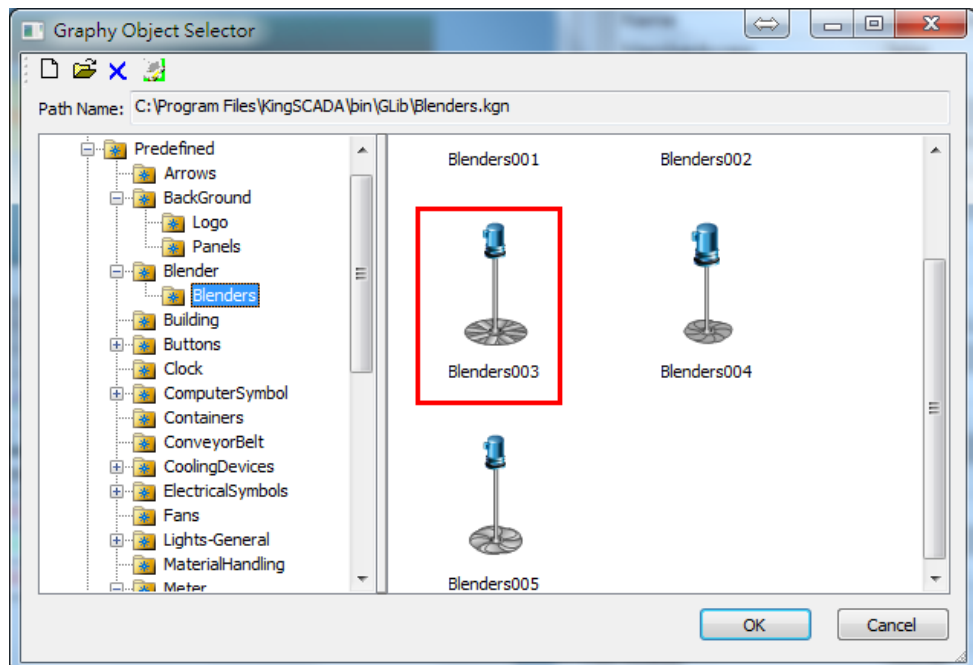


From **File** → **Open** → **Genius**, you can get the Switches, Blenders, and Sign object pictures as shown in the following figures.

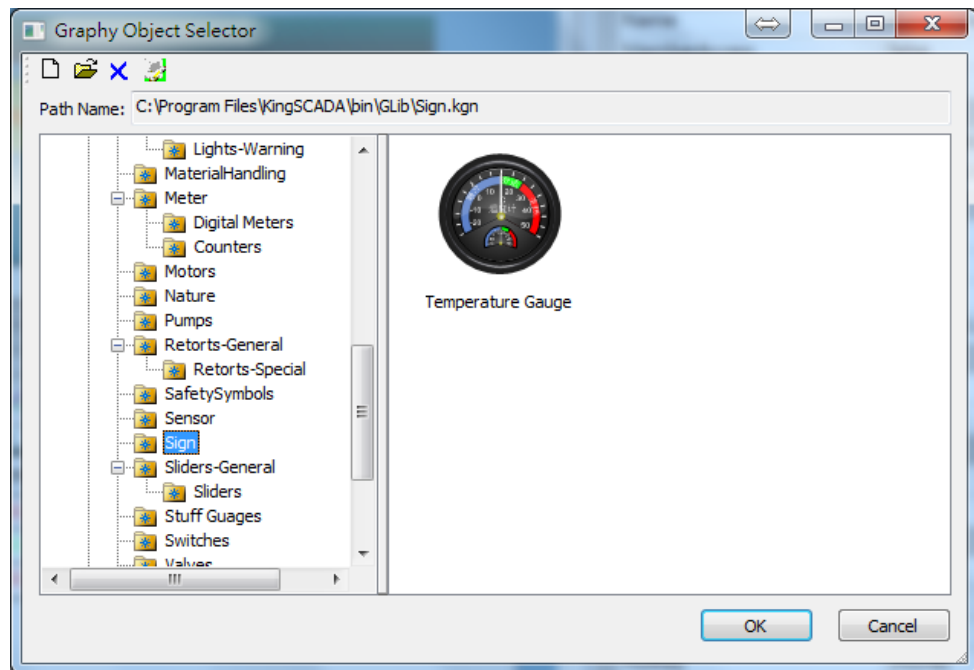
- **Switches**



- **Blenders**



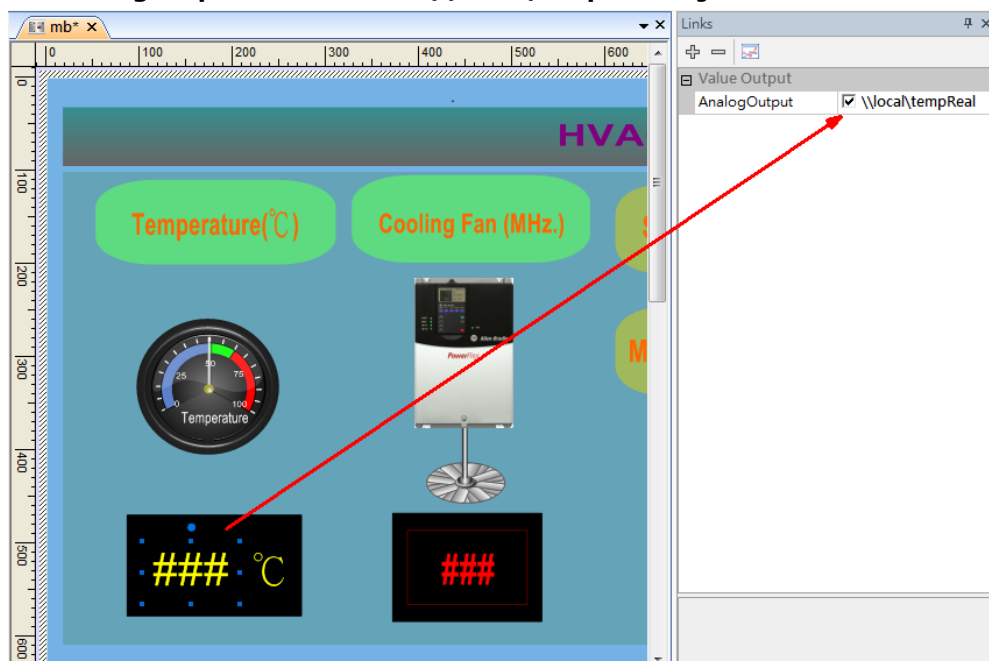
- Sign



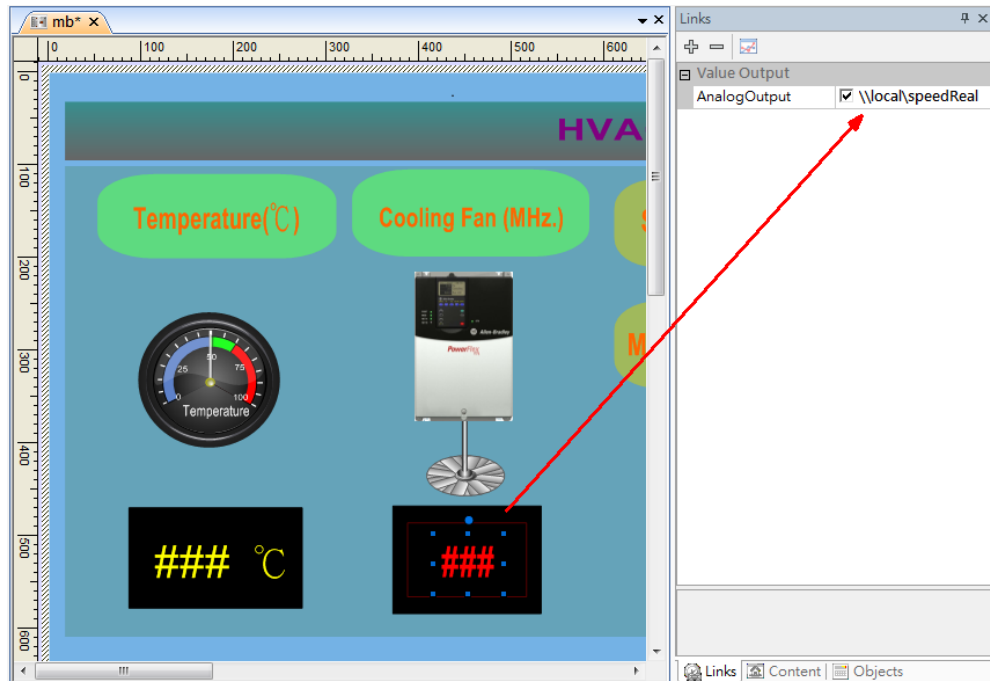
5.4.2 Adding Links

After adding the object pictures, add links to enable the system to dynamically update object graphic animation or input and output element value.

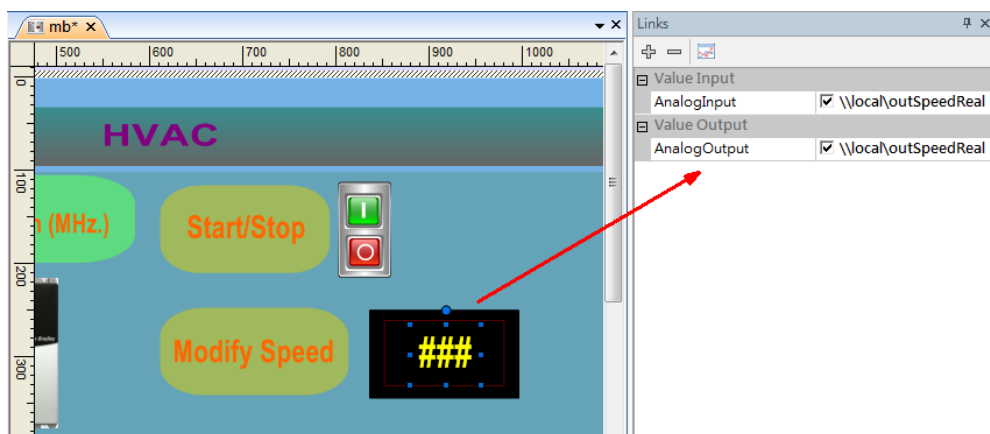
1. Select the **Temperature** input box; then, in the Links window, click the + icon to add the **AnalogOutput** source as the `\\local\tempReal` tag.



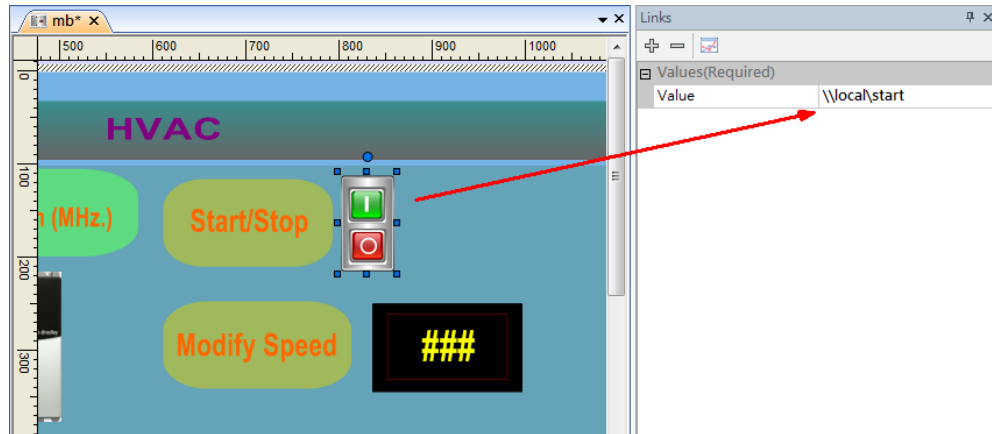
2. Select the **Speed** input box; then, in the Links window, click the + icon to add the **AnalogOutput** source as the `\\local\speedReal` tag.



3. Select the **Modify Speed** input box; then, in the Links window, click the + icon to add the **AnalogOutput** source as the `\\local\outSpeedReal` tag and the **AnalogInput** source as the `\\local\outSpeedReal` tag.

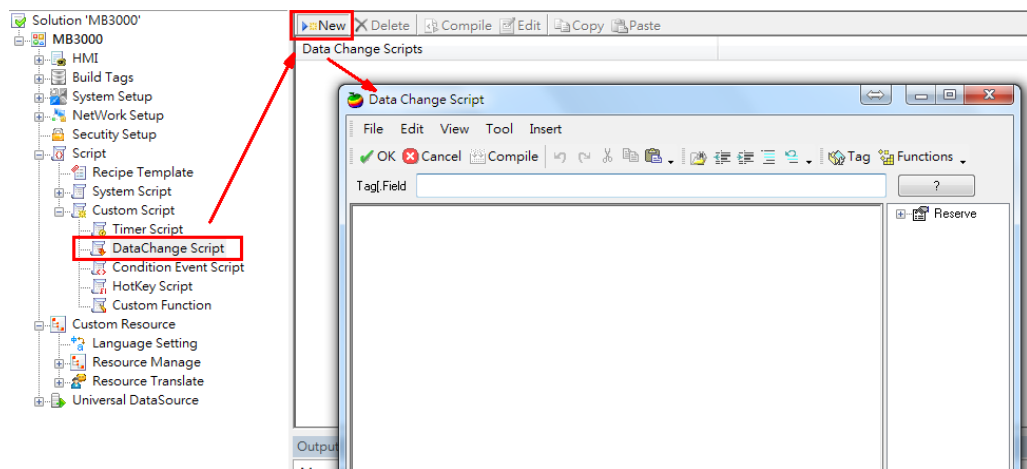


4. Select the **Start/Stop** Switch; then, in the Input window, click the + icon to add the **Value** source as the `\\local\start` tag.



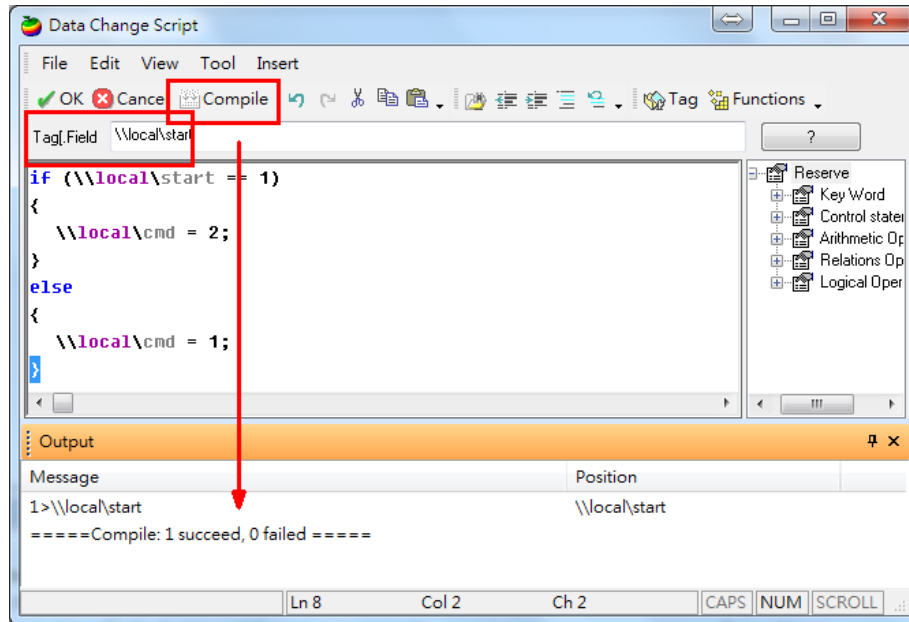
5.5 Adding a Script

1. If you click the **start** switch to change the status to on or off, SCADA will convert the status of the **start** tag to the **cmd** register. You can use the **DataChange Script** to define this data conversion. When this script is triggered, SCADA will send **Modbus Function Code 6** stored in the **cmd** register to an output value.
2. Click **Script** → **Custom Script** → **DataChange Script** and click **New** to add a script.



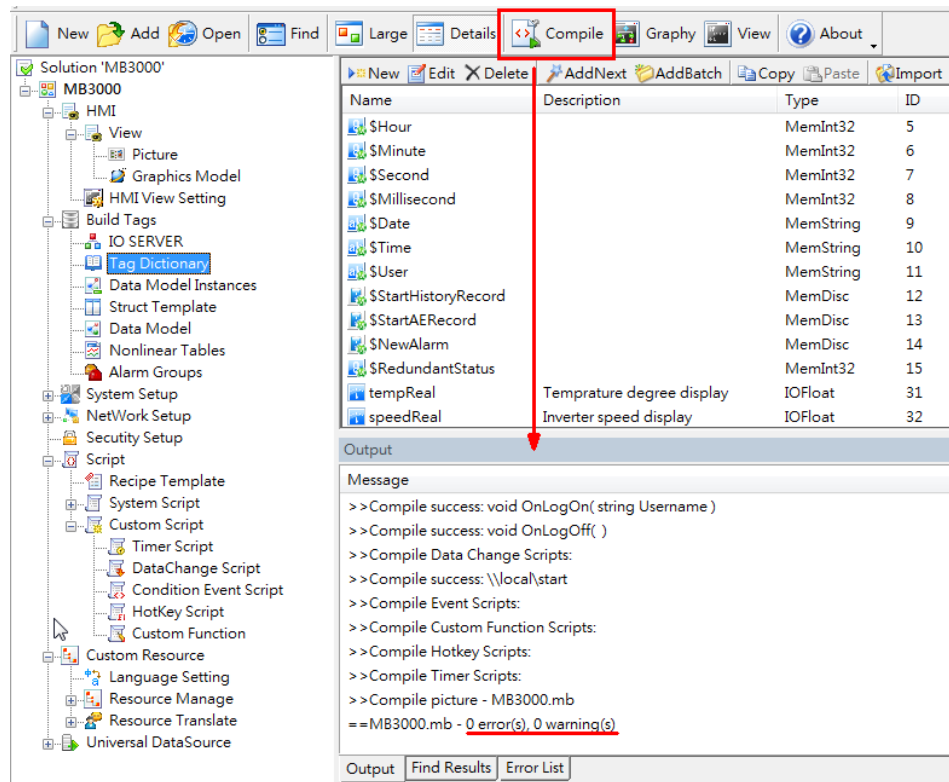
3. In the **Tag.Field** field, enter `“\\local\start”`. Create the script as shown in the following figure.

- Click **Compile** to run the script and check for any error messages. Then, click **OK** to save the script.



5.6 Compiling a Project

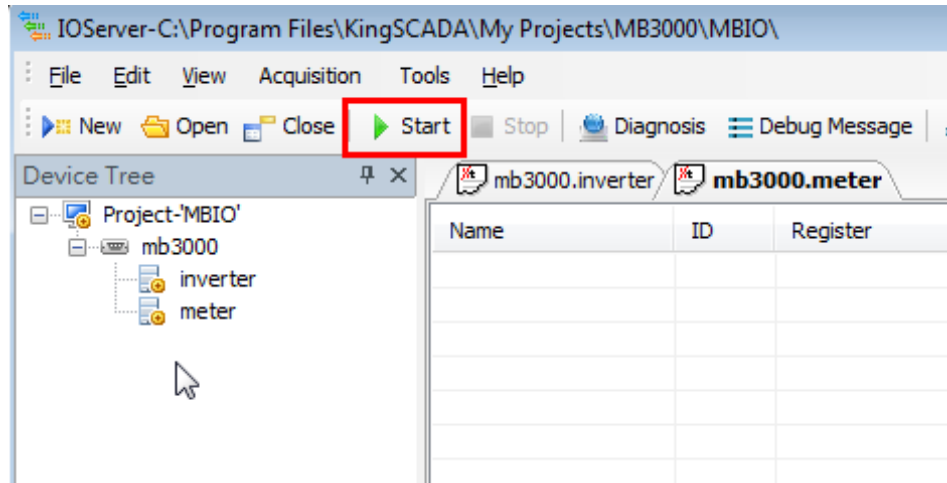
Click **Compile** to compile project. Make sure that there is no error or warning in the **Output** tab window.



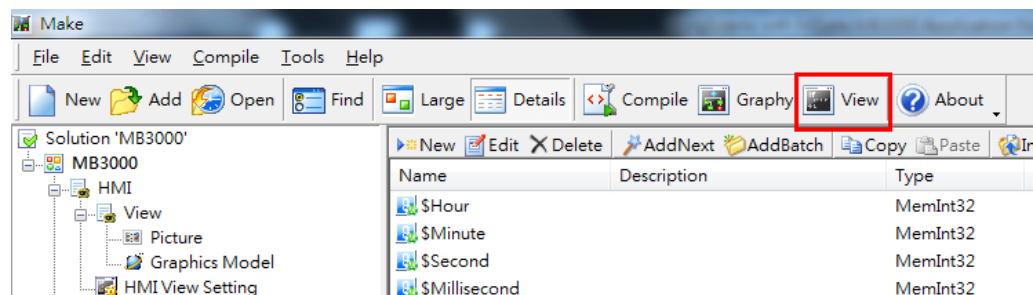
6 Runtime Test

Before you execute ScadaView, you must start the IO server first.

1. In the **IOServer Configure** program, click **Start** to start the IO server.

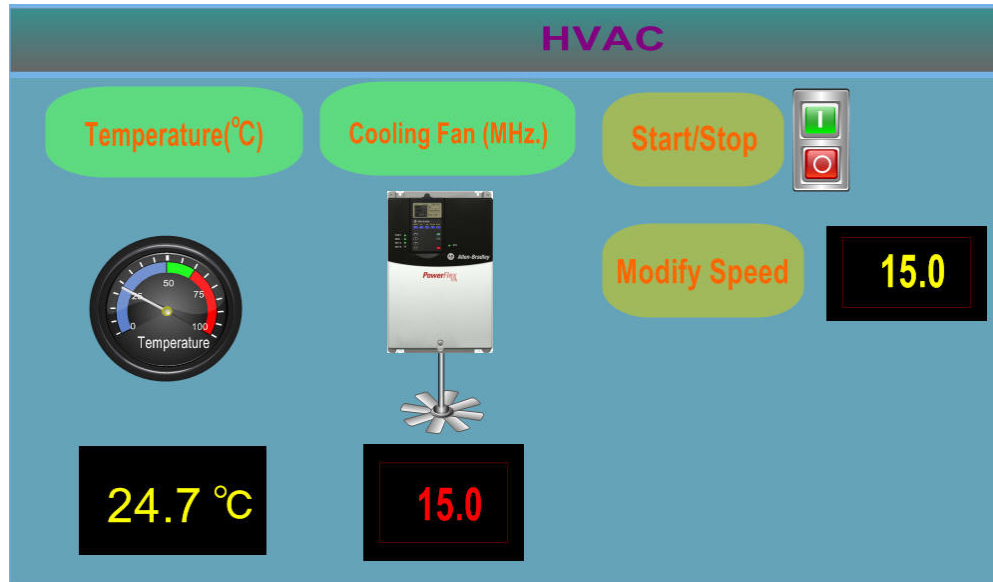


2. In the **ScadaMake** program, click **View** to run the SCADA View system. Alternatively, you can execute the **ScadaView** program to run this project.



The screen displays the HMI View. SCADA polls the Modbus register of the Temperature Meter and the PowerFlex 4M converter through MGate MB3000, and updates values on the HMI View (for example, displaying the current temperature and speed values).

You can click the Start/Stop switch to start or stop the PowerFlex 4M converter. This will set SCADA to send the Modbus Function Code 6 command to the CMD register on the converter.



If you click the Manual Control switch to enable the manual speed control function, an Input dialog box appears that allows you to specify the speed value in the Input analog value field.

