

# Application note

## Creation of Modbus profile

### How to create your own Modbus profile



# Safety Information

## Important Information



Read these instructions carefully before trying to install, configure, or operate this software. The following special messages may appear throughout this bulletin or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

The addition of either symbol to a “Danger” or “Warning” safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

### DANGER

DANGER indicates an imminently hazardous situation which, if not avoided, will result in death or serious injury.

### WARNING

WARNING indicates a potentially hazardous situation which, if not avoided, can result in death or serious injury.

### CAUTION

CAUTION indicates a potentially hazardous situation which, if not avoided, can result in minor or moderate injury.

### NOTICE


NOTICE is used to address practices not related to physical injury. The safety alert symbol shall not be used with this signal word.

## Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction, installation, and operation of electrical equipment and has received safety training to recognize and avoid the hazards involved.

## Safety Precautions

 <b>WARNING</b>
<p><b>HAZARD OF INCORRECT INFORMATION</b></p> <ul style="list-style-type: none"><li>• Do not incorrectly configure the software, as this can lead to incorrect reports and/or data results.</li><li>• Do not base your maintenance or service actions solely on messages and information displayed by the software.</li><li>• Do not rely solely on software messages and reports to determine if the system is functioning correctly or meeting all applicable standards and requirements.</li><li>• Consider the implications of unanticipated transmission delays or failures of communications links.</li></ul> <p><b>Failure to follow these instructions can result in death, serious injury, or equipment damage.</b></p>

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information that is contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

No part of this document may be reproduced in any form or by any means, electronic or mechanical, including photocopying, without express written permission of Schneider Electric.

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

© 2014 Schneider Electric. All rights reserved

# Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>7</b>
<b>2</b>	<b>Modbus device profile .....</b>	<b>8</b>
2.1	Available parameters for Modbus profile.....	9
<b>3</b>	<b>Digital Fan Coil Thermostat TC303.....</b>	<b>11</b>
3.1	Modbus settings and register's map .....	12
3.2	Digital Fan Coil Thermostat TC303 configuration .....	14
3.2.1	Communication settings .....	14
3.2.2	Creation of Modbus profile .....	14
3.3	Shifted register addresses .....	15
3.5	Add Modbus profile and usage .....	16
<b>4</b>	<b>Modicon M221Logic Controller .....</b>	<b>17</b>
4.1	Modicon M221 configuration.....	17
4.1.1	Establish connection between PC and Modicon M221 .....	18
4.1.2	Setting Ethernet communication .....	18
4.1.3	Setting Modbus communication.....	20
4.1.4	Modbus registers configuration.....	21
4.2	Project for Modicon M221 Logic Controller .....	22
<b>5</b>	<b>Touch Panel Magelis HMI STU .....</b>	<b>23</b>
5.1	Magelis HMI STU Introduction.....	23
5.2	Vijeo Designer - Settings of Magelis HMI STU.....	24
5.2.1	Vijeo Designer Introduction .....	24

5.2.2 Set Magelis HMI STU as Modbus Slave ..... 24

5.2.3 Creation of Modbus Variables ..... 26

5.3 Modbus Profile for Magelis HMI STU..... 27

6 Conclusion..... 28

7 Appendix ..... 28

7.1 Glossary ..... 28



# 1 Introduction

This Application note (hereinafter AN) describes creation of Modbus device profile. Wiser for KNX comes with pre-installed Modbus profiles. Modbus devices which are not pre-installed in Wiser for KNX can be integrated via Modbus profile ( \*.json file) and customized according client's request.

AN is primary focused on integration of 3 devices.

- Stand-alone digital fan coil thermostat TC303 (TC303-3A4LM)
- Modicon M221 logic controller (TM221CE24T)
- Touch Panel Magelis HMI STU ( HMISTU655 )

Whole process of integration is applicable on any other Modbus device.

This AN helps you to take advantage of wide range of devices on the market to build robust and suitable solution for integration of Modbus devices in minimum time.

A glossary is available in the appendix chapter of this document. Please refer to it whenever necessary.

## Competencies

This document is intended for readers who have been trained on Wiser for KNX, Wiser for KNX products. The integration should not be attempted by someone who is new to the installation of either products. In addition, we recommend that you are familiar with:

- The technical knowledge of Modbus protocol and PLCs
- JSON

## System prerequisites

Software	Version	Download
Wiser for KNX	2.1 and newer	<a href="http://www.schneider-electric.com">http://www.schneider-electric.com</a>
SoMachine Basic	1.3 SP1	<a href="http://www.schneider-electric.com">http://www.schneider-electric.com</a>
Vijeo Designer	6.2.2 SP2.1	<a href="http://www.schneider-electric.com">http://www.schneider-electric.com</a>

Table 1: software versions of used software

## 2 Modbus device profile

Modbus profile is a \*.json file with structure which defines how to read and write registers in Modbus device. This \*.json file contains parameters as register addresses, description, datatypes, units which is necessary to handle Modbus device. Each Modbus device has its own profile (\*.json file).

Pre-installed Modbus profiles in Wiser for KNX are available on The Exchange Community

<https://exchangecommunity.schneider-electric.com>

This Modbus profiles can be downloaded for free and use for create own profile or in case of delete pre-installed profile from Wiser for KNX.



## 2.1 Available parameters for Modbus profile

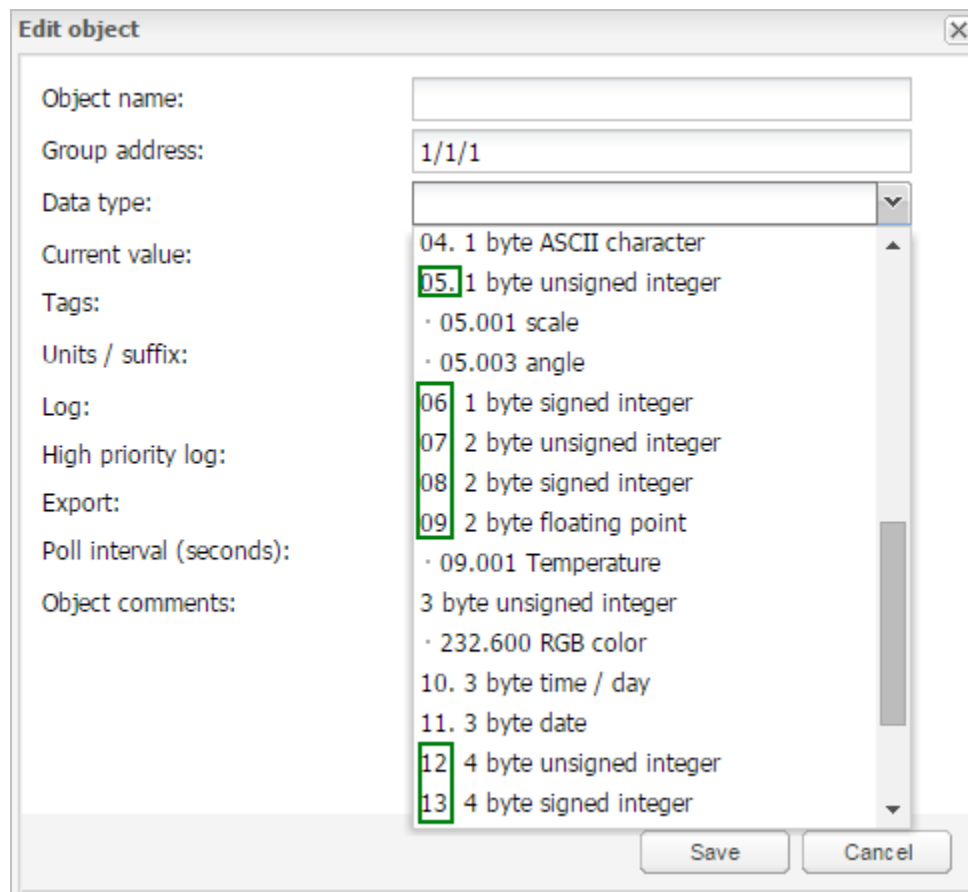
All available parameters which can be used are described in Table 2.

Parameter	Description
name	Object name, e.g. Output 2 (String, <b>Required</b> )
bus_datatype	KNX object data type, key from dt table, e.g. float32 (String/Number, <b>Required</b> )
type	Modbus register type, possible values: coil, discreteinput, register, inputregister (String, <b>Required</b> )
address	Register address (0-based) (Number, <b>Required</b> )
writable	Set to true to enable writing to coil or register
datatype	Modbus value data type. If set, conversion will be done automatically. Possible values: <b>bool</b> , <b>uint16</b> , <b>int16</b> , <b>float16</b> , <b>uint32</b> , <b>int32</b> , <b>float32</b> , <b>uint64</b> , <b>int64</b> , <b>quad10k</b> , <b>s10k</b> (String)
value_delta	New value is sent when the difference between previously sent value and current value is larger than delta. Defaults to 0 (send after each read) (Number)
value_multiplier	Multiply resulting value by the specified number, value = value_base + value * value_multiplier (Number)
value_bitmask	Bit mask to apply, shifting is done automatically based on least significant 1 found in the mask (Number)
value_nan	Array of 16-bit integers. If specified and read operation returns value specified in the array no further processing of value is done (Array)
value_custom	Name of a built-in enumeration or a list of key -> value mapping, resulting value will be 0 if key is not found (String/Object)
internal	Not visible to user when set to true, should be used for scale registers (Boolean)
units	KNX object units/suffix (String)
address_scale	Address of register containing value scale, value = value * 10 ^ scale (Number)
read_count	Number of register to read at once (for devices that only support reading of a specific block of registers) (Number)
read_swap	Swap register order during conversion (endianness) (Boolean)
read_offset	Position of first register of data from the block of registers (0-based) (Number)

Table 2: Available parameters for Modbus device

Bus datatype could be also used as a number instead of string.

Correct numbers of datatypes are in *Configurator -> Objects -> Edit Objects*



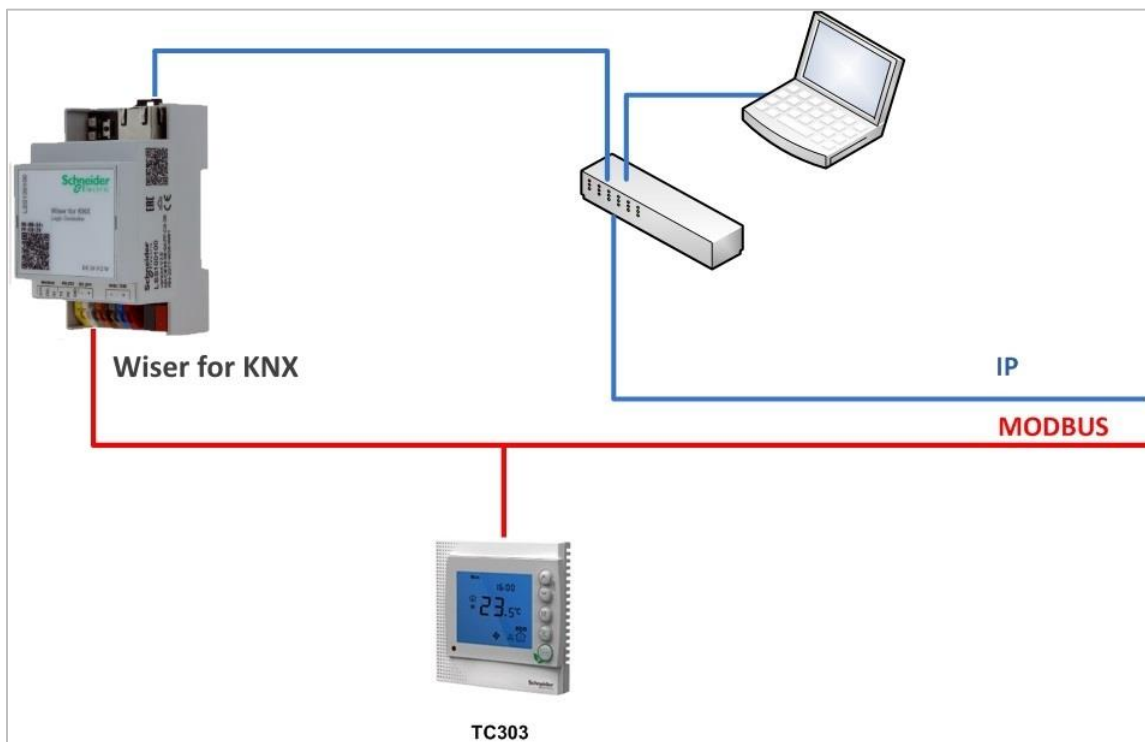
Picture 1: Datatype numbers in Wiser for KNX

### 3 Digital Fan Coil Thermostat TC303

Digital Fan Coil Thermostat TC303 is stand-alone digital fan coil thermostat designed for temperature control in industrial, commercial and residential environments. This thermostat can be ordered with Modbus interface and integrated with Wiser for KNX.



Picture 2: Digital Fan Coil Thermostat TC303



Picture 3: Architecture of Wiser for KNX and Thermostat TC303

## 3.1 Modbus settings and register's map

Modbus settings and description of registers of thermostat TC303 are available in Installation Instructions.

The RS-485 communication parameters are fixed as follows:

- 4800Bd (Baud rate)
- 8 Data bits
- Odd parity
- 1 Stop bit
- 1 – 32 Slave address

Register address	Register Description	Value Definition
1	Cooling Valve in 4-pipe Application	0 = Off 1 = On
5	Fan speed Status - High	0 = Off 1 = On
6	Fan speed Status - Medium	0 = Off 1 = On
7	Fan speed Status - Low	0 = Off 1 = On
8	Heating Valve in 4-pipe Application	0 = Off 1 = On

Table 3: Function Code 01 - Read coils

Register address	Register Description	Value Definition
3	Embedded Temperature Sensor Status	0 = Ok 1 = Fault
4	Remote Temperature Sensor Status	0 = Ok 1 = Fault

Table 4: Read Discrete Inputs

Register address	Register Description	Value Definition
3	Thermostat Mode	0 = Off 1 = On 2 = Frost Protection (Read Only)
4	Operating Mode	1 = Cool 2 = Heat 3 = Ventilation
5	Room Temperature Setpoint	Temperature (5 to 35°C)
6	Fan Mode	0 = High 1 = Medium 2 = Low 3 = Auto
7	Unoccupied Room Temperature Setpoint (Cooling Mode)	Temperature (22 to 32°C)
8	Unoccupied Room Temperature Setpoint (Heating Mode)	Temperature (10 to 21°C)
9	Sleep Mode	0 = Disable 1 = Enable
10	Eco Mode	0 = Disable 1 = Enable
11*	Occupancy Mode	0 = Unoccupied 1 = Occupied
12	Unoccupied Fan Speed Mode	0 = High 1 = Medium 2 = Low
13	Keypad Status	0 = Unlocked

**Table 5: Write Single Register**

\* Read Only

Register address	Register Description	Value Definition
1	Actual Room Temperature	Temperature (0 to 50°C)

**Table 6: Read input registers**

**\*Register Type**

Function Code 01 - Read Coils  
Function Code 02 - Read Discrete Inputs  
Function Code 03 - Read Holding Registers  
Function Code 04 - Read Input Registers  
Function Code 06 – Write Single Register

**\*Data Format**

Binary/Digital  
Binary/Digital  
16-bit unsigned Integer  
16-bit unsigned Integer  
16-bit unsigned Integer

## 3.2 Digital Fan Coil Thermostat TC303 configuration

Configuration of Digital Fan Coil Thermostat TC303 consists of two parts.

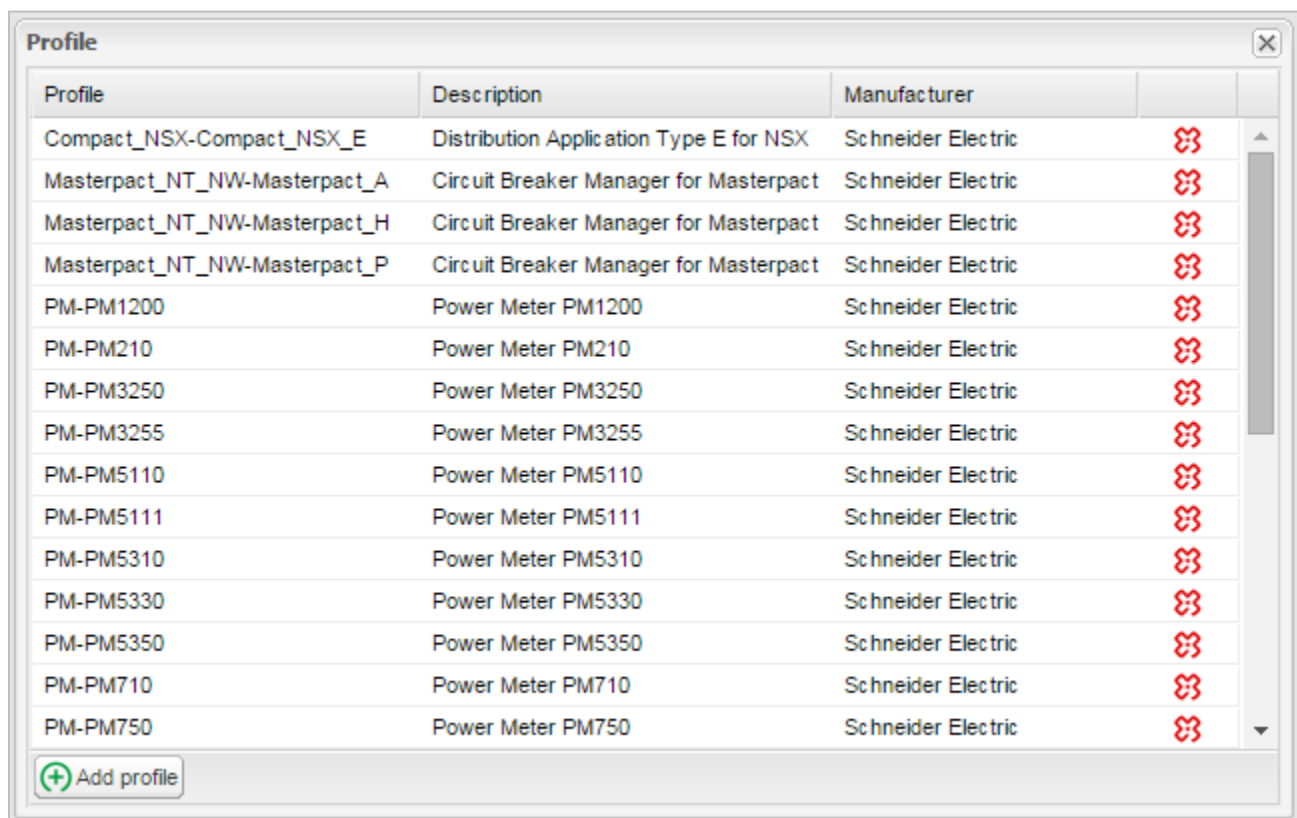
- Communication settings
- Creation of Modbus profile

### 3.2.1 Communication settings

- During power off, press and hold **M button** for 5 seconds
- Press **M button** 4 times to select address setting
- Change Modbus address (01 – 32) – other Modbus settings such as baud rate are preset.

### 3.2.2 Creation of Modbus profile

Use common text editor to create Modbus profile e.g. (Microsoft Notepad, Notepad++). Modbus profile always begins with declaration of manufacturer and description of Modbus device. These texts are used in window with Modbus profiles (see Picture 4).



Profile	Description	Manufacturer	
Compact_NSX-Compact_NSX_E	Distribution Application Type E for NSX	Schneider Electric	✖
Masterpact_NT_NW-Masterpact_A	Circuit Breaker Manager for Masterpact	Schneider Electric	✖
Masterpact_NT_NW-Masterpact_H	Circuit Breaker Manager for Masterpact	Schneider Electric	✖
Masterpact_NT_NW-Masterpact_P	Circuit Breaker Manager for Masterpact	Schneider Electric	✖
PM-PM1200	Power Meter PM1200	Schneider Electric	✖
PM-PM210	Power Meter PM210	Schneider Electric	✖
PM-PM3250	Power Meter PM3250	Schneider Electric	✖
PM-PM3255	Power Meter PM3255	Schneider Electric	✖
PM-PM5110	Power Meter PM5110	Schneider Electric	✖
PM-PM5111	Power Meter PM5111	Schneider Electric	✖
PM-PM5310	Power Meter PM5310	Schneider Electric	✖
PM-PM5330	Power Meter PM5330	Schneider Electric	✖
PM-PM5350	Power Meter PM5350	Schneider Electric	✖
PM-PM710	Power Meter PM710	Schneider Electric	✖
PM-PM750	Power Meter PM750	Schneider Electric	✖

+ Add profile

Picture 4: Pre-installed Modbus profiles

Modbus profiles follows with description of mapping Modbus registers according instructions. Available parameters for Modbus devices are listed in Table 2.

```
{
  "manufacturer": "Schneider Electric",
  "description": "Digital Fan Coil Thermostat",
  "mapping": [
    { "name": "Cooling Valve in 4-pipe Application", "bus_datatype": "bool", "type": "coil",
      "address": 0, "writable": 0, "value_custom": {"0": "Off", "1": "On" } },
    { "name": "Fan speed Status - High", "bus_datatype": "bool", "type": "coil", "address":
      4, "writable": 0, "value_custom": {"0": "Off", "1": "On" } },
    { "name": "Fan speed Status - Medium", "bus_datatype": "bool", "type": "coil", "address":
      5, "writable": 0, "value_custom": {"0": "Off", "1": "On" } },
  ]
}
```

**Note:** We recommend carefully read information about registers in instructions. Especially is important check datatype of registers and register address. Thermostat TC303 have real register addresses shifted -1.

### 3.3 Shifted register addresses

Register addresses normally starts at value 0. Some of Modbus devices have real register addresses shifted of -1. If this fact is not mentioned in Installation Instructions, then best procedure is to choose register with fixed or known value and try to read it out.

For example: Thermostat TC303 have register for Eco Mode associated with button. Then it is easy to check if pressing the button change value in register.

Power meters can be check by choosing registers which contains values shown on display.

Example: Actual Room Temperature value have register address 1 according to Installation Instruction. Entry in \*.json file is following (see table 6 to check Installation Instruction data).

```
{ "name": "Actual Room Temperature", "bus_datatype": "float16", "datatype": "uint16",  
"type": "inputregister", "address": 0, "value_multiplier": 0.1, "units": "°C" }
```

## Function Code 04 - Read Input Registers

Register Address	Register Description	Value Definition
1	Actual Room Temperature	Temperature (0 to 50 °C)

Picture 5: Actual Room temperature – instructions

Keep in mind that temperature value has one decimal place. It is necessary to convert *uint16* datatype to *float16* datatype used in Wiser for KNX and to multiply by 0.1 to get correct value.

## 3.5 Add Modbus profile and usage

When you create \*.json file it is necessary to add profile to Wiser for KNX.

- Open *Configurator -> Utilities -> Modbus*
- Click **Profiles**
- Click **Add profile**
- Choose \*.json file on your hard drive
- Click **Save**

Modbus profile is added in the list of Modbus devices. Next step is to add device and map registers.

- On *Modbus tab* click **Add device**
- Fill in required information
- Click **Save**
- Click **Mapping** and map registers which you need

Registers are mapped and available in *Configurator -> Utilities -> Objects*

Modbus profile for Digital Fan Coil Thermostat TC303 is attached as a file TC303.json as a part of this AN.



## 4 Modicon M221 Logic Controller

Modicon M221 Logic Controller can be ordered in various versions. Modicon TM221CE24T is used for this AN. Software used for configuration, programming and commissioning is called *SoMachine Basic*. Modicon M221 is designed for control in industrial environments.

Overview:

- 14 digital inputs
- 10 digital outputs
- 2 analog inputs
- 1 serial line / RJ45 connector (RS-232 or RS-485)
- 1 Ethernet / RJ45 connector
- 1 USB mini-B programming port



Picture 6: Modicon M221

Modicon M221 is equipped with Modbus interface and can be integrated with Wiser for KNX.

### 4.1 Modicon M221 configuration

Configuration of Modicon M221 is realized via *SoMachine Basic* available on Schneider Electric website.

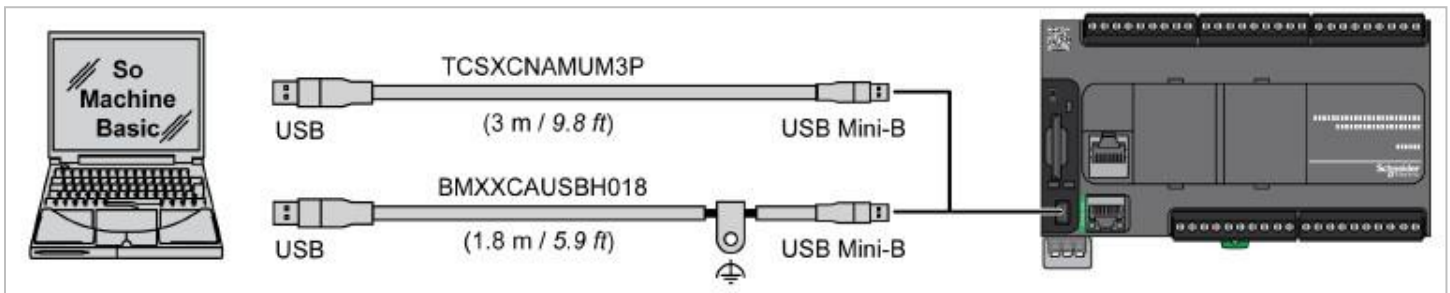
<http://www.schneider-electric.com/products/ww/en/5100-software/5140-pac-plc-programming-software/2226-somachine/>

Configuration of Modicon M221 consists of following steps.

- Establish connection between PC and Modicon M221
- Setting of Ethernet communication
- Setting of Modbus communication
- Creation of Modbus profile

#### 4.1.1 Establish connection between PC and Modicon M221

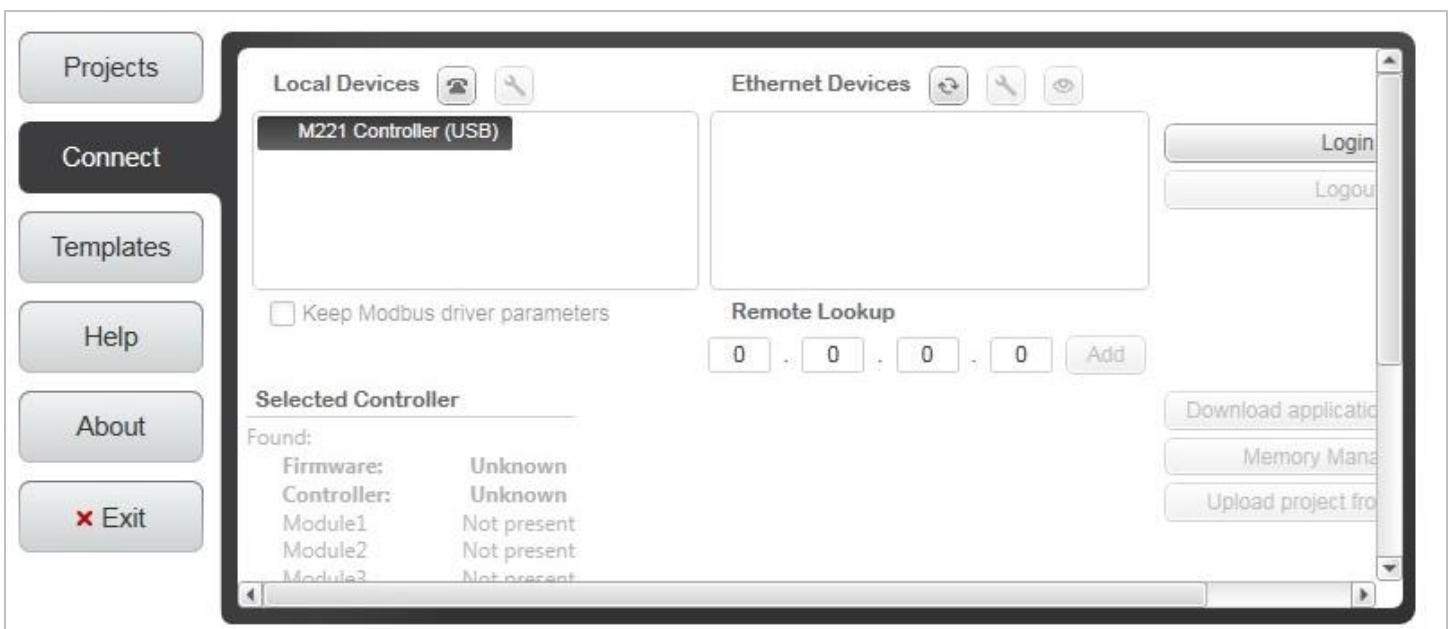
For the first connection between PC and Modicon M221 we recommend to use USB Mini-B cable until Ethernet connection is established.



Picture 7: Connection between PC and Modicon M221

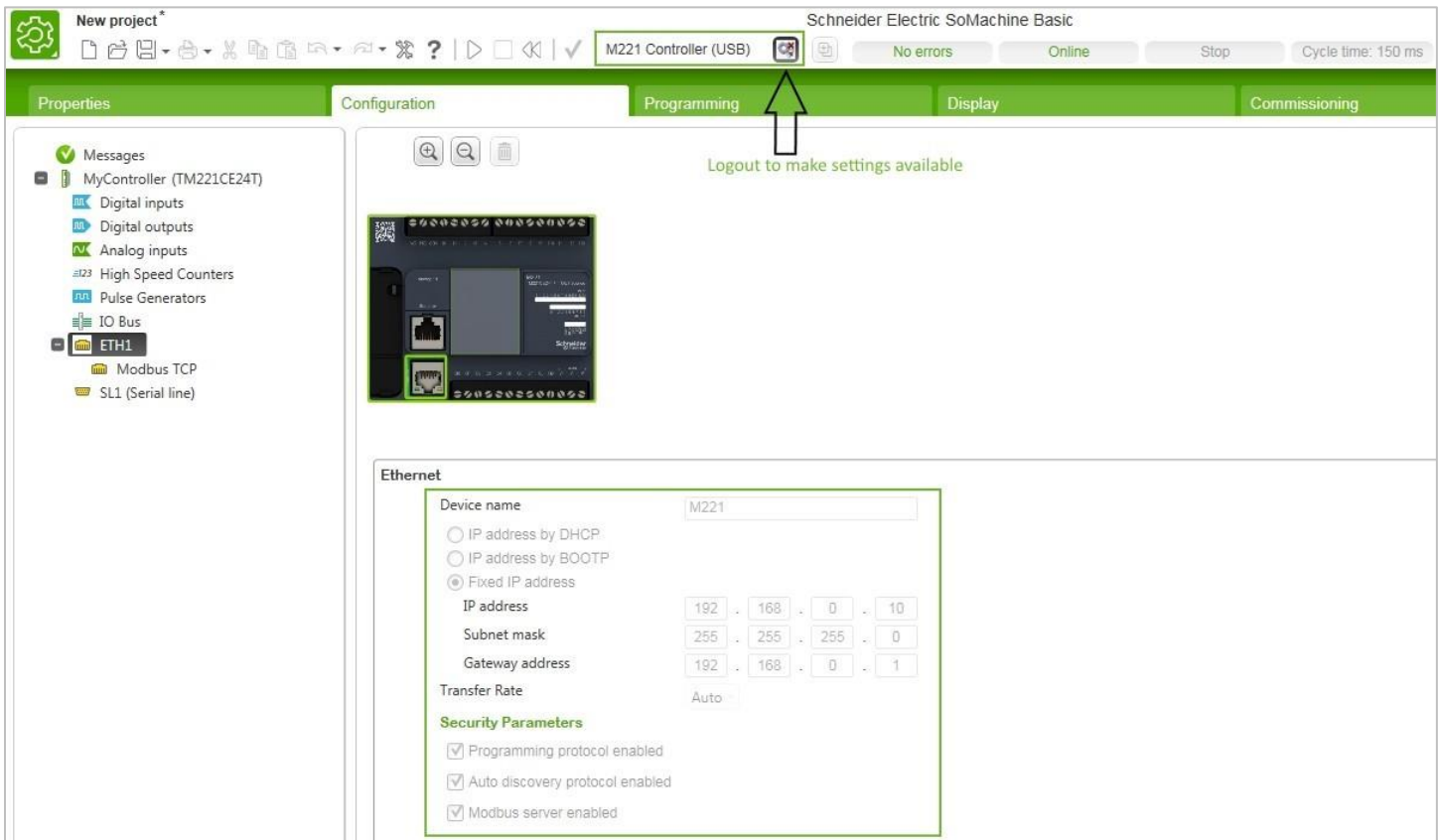
#### 4.1.2 Setting Ethernet communication

- Open Schneider Electric *SoMachine Basic*



Picture 8: USB connection

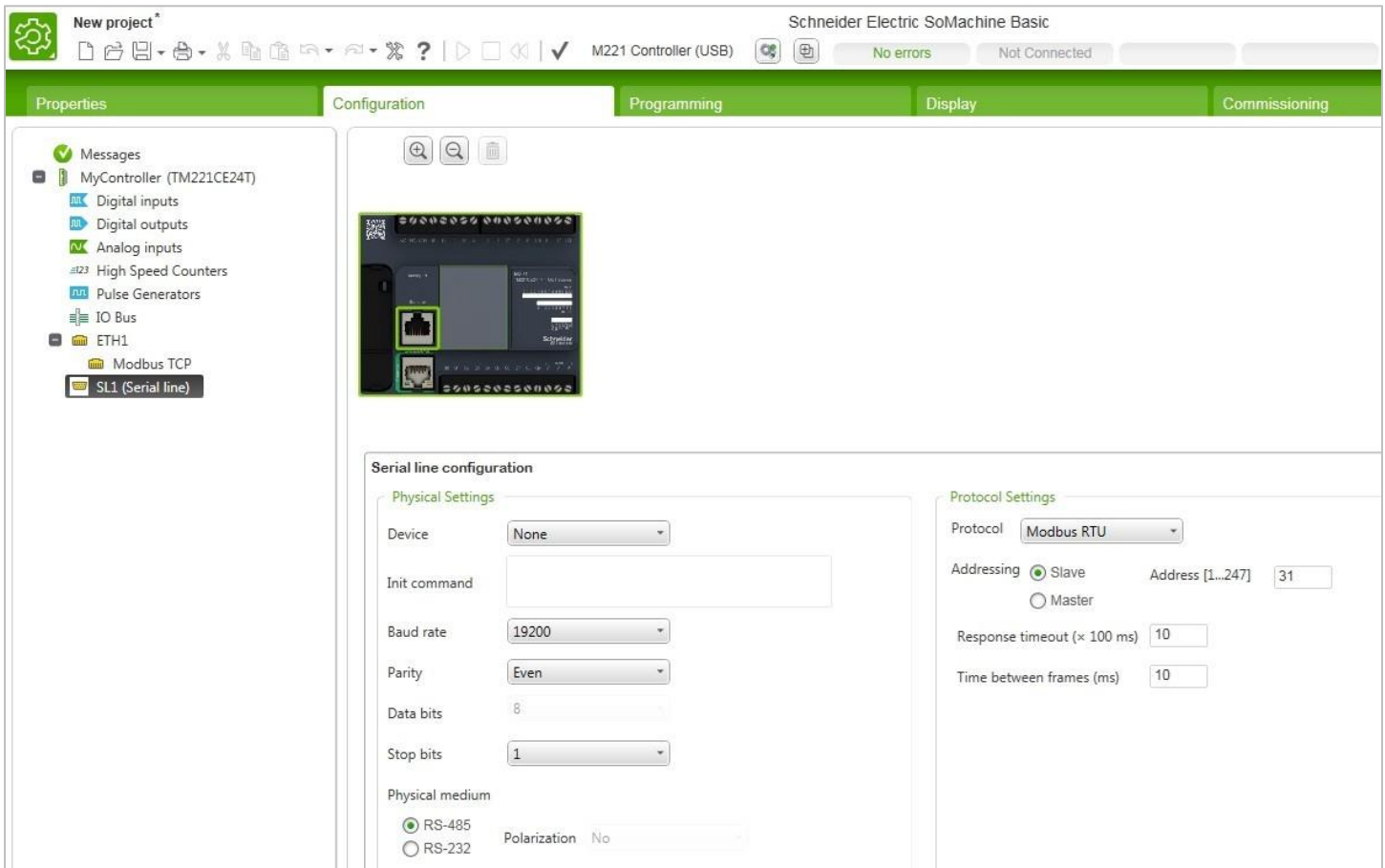
- Login to *SoMachine Basic* (if PC and controller applications are different then **Upload project from controller** firstly)
- Go to *Configuration* -> *ETH1* and set Ethernet according your network
- Logout from Modicon M221 to make Ethernet configuration available
- Configure Ethernet parameters according to your network



Picture 9: Ethernet settings

### 4.1.3 Setting Modbus communication

- Go to *Configuration* -> *SL 1 (Serial line)* and set Modbus according to your network



Picture 10: Modbus setting

- Go to **Commissioning**  
(PC and controller applications are different)
- Click **PC to Controller (download)**

Ethernet and Modbus configuration is done

## 4.1.4 Modbus registers configuration

Registers can be configured in *Programming -> Memory objects*

Based on this association we create \*.json file. The procedure is the same as for Thermostat TC303.

**Note:** Register addresses are not shifted -1!

M\_M properties

Used	Address	Symbol
<input checked="" type="checkbox"/>	%M0	Q0
<input checked="" type="checkbox"/>	%M1	Q1
<input checked="" type="checkbox"/>	%M2	Q2
<input checked="" type="checkbox"/>	%M3	Q3
<input checked="" type="checkbox"/>	%M4	Q4
<input checked="" type="checkbox"/>	%M5	Q5
<input checked="" type="checkbox"/>	%M6	Q6
<input checked="" type="checkbox"/>	%M7	Q7
<input checked="" type="checkbox"/>	%M8	Q8
<input checked="" type="checkbox"/>	%M9	Q9
<input checked="" type="checkbox"/>	%M10	I00
<input checked="" type="checkbox"/>	%M11	I01
<input checked="" type="checkbox"/>	%M12	I02
<input checked="" type="checkbox"/>	%M13	I03
<input checked="" type="checkbox"/>	%M14	I04
<input checked="" type="checkbox"/>	%M15	I05
<input checked="" type="checkbox"/>	%M16	I06
<input checked="" type="checkbox"/>	%M17	I07
<input checked="" type="checkbox"/>	%M18	I08
<input checked="" type="checkbox"/>	%M19	I09
<input checked="" type="checkbox"/>	%M20	I10
<input checked="" type="checkbox"/>	%M21	I11
<input checked="" type="checkbox"/>	%M22	I12
<input checked="" type="checkbox"/>	%M23	I13

\*.JSON file / Modbus profile

```
1 {
2   "manufacturer": "Schneider Electric",
3   "description": "Modicon M221 logic controller.",
4   "mapping": [
5     { "name": "Q0", "bus_datatype": "1", "type": "coil", "address": 0, "writable": "true" },
6     { "name": "Q1", "bus_datatype": "1", "type": "coil", "address": 1, "writable": "true" },
7     { "name": "Q2", "bus_datatype": "1", "type": "coil", "address": 2, "writable": "true" },
8     { "name": "Q3", "bus_datatype": "1", "type": "coil", "address": 3, "writable": "true" },
9     { "name": "Q4", "bus_datatype": "1", "type": "coil", "address": 4, "writable": "true" },
10    { "name": "Q5", "bus_datatype": "1", "type": "coil", "address": 5, "writable": "true" },
11    { "name": "Q6", "bus_datatype": "1", "type": "coil", "address": 6, "writable": "true" },
12    { "name": "Q7", "bus_datatype": "1", "type": "coil", "address": 7, "writable": "true" },
13    { "name": "Q8", "bus_datatype": "1", "type": "coil", "address": 8, "writable": "true" },
14    { "name": "Q9", "bus_datatype": "1", "type": "coil", "address": 9, "writable": "true" },
15    { "name": "I0", "bus_datatype": "1", "type": "coil", "address": 10, "writable": "true" },
16    { "name": "I1", "bus_datatype": "1", "type": "coil", "address": 11, "writable": "true" },
17    { "name": "I2", "bus_datatype": "1", "type": "coil", "address": 12, "writable": "true" },
18    { "name": "I3", "bus_datatype": "1", "type": "coil", "address": 13, "writable": "true" },
19    { "name": "I4", "bus_datatype": "1", "type": "coil", "address": 14, "writable": "true" },
20    { "name": "I5", "bus_datatype": "1", "type": "coil", "address": 15, "writable": "true" },
21    { "name": "I6", "bus_datatype": "1", "type": "coil", "address": 16, "writable": "true" },
22    { "name": "I7", "bus_datatype": "1", "type": "coil", "address": 17, "writable": "true" },
23    { "name": "I8", "bus_datatype": "1", "type": "coil", "address": 18, "writable": "true" },
24    { "name": "I9", "bus_datatype": "1", "type": "coil", "address": 19, "writable": "true" },
25    { "name": "I10", "bus_datatype": "1", "type": "coil", "address": 20, "writable": "true" },
26    { "name": "I11", "bus_datatype": "1", "type": "coil", "address": 21, "writable": "true" },
27    { "name": "I12", "bus_datatype": "1", "type": "coil", "address": 22, "writable": "true" },
28    { "name": "I13", "bus_datatype": "1", "type": "coil", "address": 23, "writable": "true" }
29  ]
30 }
```

Picture 11: Modbus profile according to register mapping in SoMachine

**Note:** How to add and use Modbus profile are described in chapter 3.5

Modbus profile for Modicon M221 Logic Controller is attached as a file M221.json as a part of this AN.

## 4.2 Project for Modicon M221 Logic Controller

Backup of project for Modicon M221 Logic Controller is attached as a file *M221\_project\_backup.zip*. This project can be used for automatic restore of application from SD card which is compatible with Modbus profile.

### Preparing SD Card

1. Format SD card on the PC
2. Unzip and Copy folder structure to SD card  
Folder structure is attached to this AN as *M221\_project\_backup.zip*
3. Remove SD card from your PC

### Restoring or Copying from a Clone SD card

1. Remove power from the Modicon M221
2. Insert the SD card into the Modicon M221
3. Restore power to the controller  
The clone operation is in progress. SD LED is turned ON during the operation
4. Wait until the end of the download (the SD LED is turned off)
5. Remove the SD card to restart the Modicon M221

Modicon M221 Logic Controller now contains project compatible with attached Modbus profile with mapped registers.

Note: For more information about cloning press **F1** in SoMachine Basic and refer keyword *clone management*.

# 5 Touch Panel Magelis HMI STU

## 5.1 Magelis HMI STU Introduction

Magelis HMI STU is a small touch panel from Schneider Electric offer (see Picture 12). It is mainly used by OEM customers in industrial business. It can easily communicate with Schneider Electric PLCs using proprietary SoMachine protocol or with various 3<sup>rd</sup> party devices using Modbus TCP or Modbus RTU protocols.

Overview:

- 5.7" TFT color touch screen, 320x240 pixels (QVGA ) with 65k colors
- 1x RJ45 – Serial port (Modbus RTU)
- 1 x RJ45 – Ethernet port (Modbus TCP, PacDrive, XWAY, ...)
- 2 x USB ports (for downloading the configuration either by a standard USB printer cable or USB memory stick)
- Configured by the Vijeo Designer software
- Software is free when used to configure the Magelis HMISTO and HMISTU



Picture 12: Magelis HMI STU

## 5.2 Vijeo Designer - Settings of Magelis HMI STU

### 5.2.1 Vijeo Designer Introduction

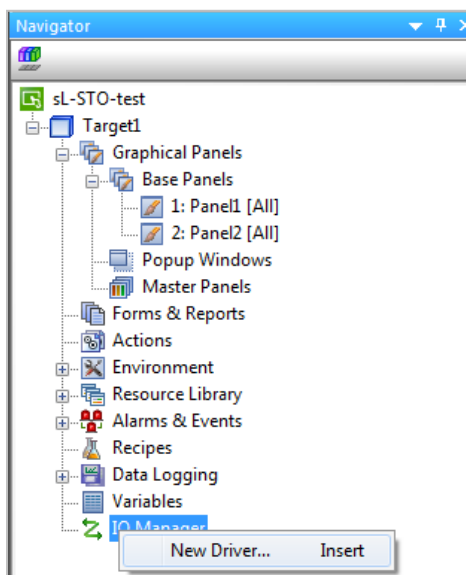
Vijeo Designer is an IDE software for Magelis touch panels. It manages complete functionality of Magelis touch panels involving GUI, stored variables, communication settings, alarms and much more. Vijeo Designer is free of charge to use for configuration of touch panels Magelis HMI STU and Magelis HMI STO.

This chapter does not aim to be a Vijeo Designer user guide. Modbus communication settings is described to establish communication between Magelis touch panel and Wiser for KNX. For further information about Vijeo Designer please refer to Vijeo Designer help.

### 5.2.2 Set Magelis HMI STU as Modbus Slave

Open or create a Vijeo Designer project for Magelis HMI STU and follow the steps below.

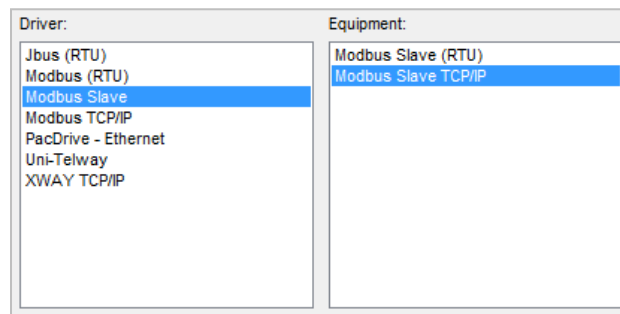
**Step 1**      *Navigator window >> Right-mouse click IO Manager >> New Driver*



Picture 13: Add new communication driver

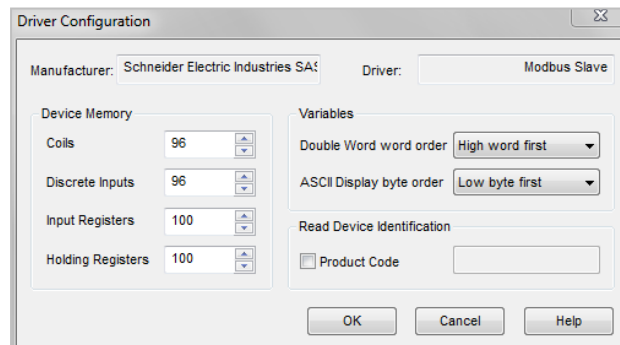


**Step 2**      Select *Modbus Slave* and *Modbus Slave TCP/IP*



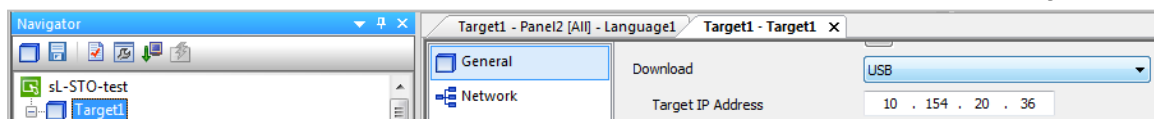
Picture 14: Type of driver selection

**Step 3**      Set the number of coils, discrete inputs, input registers and holding registers.



Picture 15: Set number of Modbus registers

**Step 4**      IP address of Magelis TP can be set in *Navigator >> Target1 >> General >> Target IP address*.



Picture 16: IP settings of Magelis

Note: Standard TCP port 502 is used for Modbus TCP communication.

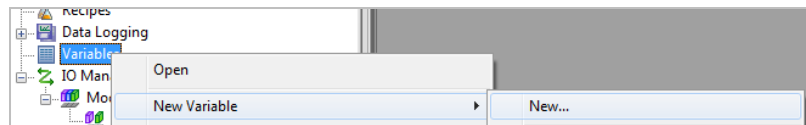
## 5.2.3 Creation of Modbus Variables

All Modbus registers in Magelis HMI STU are available via *external variables*, which defines mapping of Vijeo Designer variables to Modbus registers. Three different data types have been tested and validated as fully compatible with Wiser for KNX:

- BOOL (Boolean)
- INT (int16)
- REAL (float32)

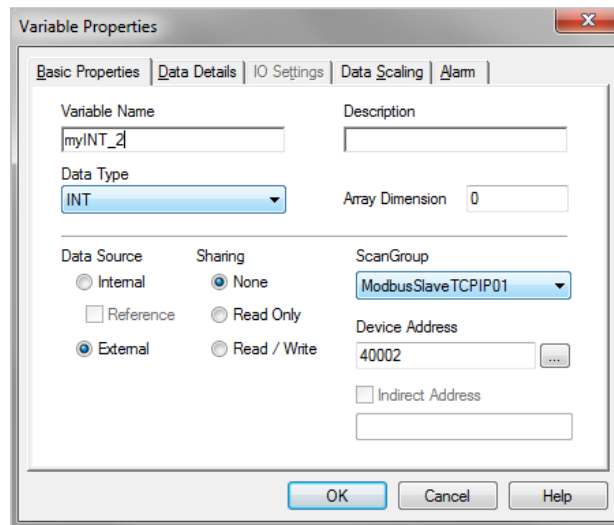
To create Modbus variable, follow the steps below.

**Step 1**     *Navigator >> Right – mouse click on Variables >> New Variable >> New*



Picture 17: Add new variable

**Step 2**     Set the variable properties: Variable Name, Data Type and Device Address.



Picture 18: Variable properties

Pay attention to Device Address setting. You can choose various *Address* range for each *Data Type*. This *Address* range is related to Modbus function used for the register. Address range can start with following numbers:

- 0 - Coil
- 1 – Input

- 3 – Input Register
- 4 – Holding Register




Addressing in Magelis touch panel starts at address 1. It means that it is shifted by -1 to Wiser for KNX addressing. To better understand the Modbus address setting, refer to an example in next subchapter.

## 5.3 Modbus Profile for Magelis HMI STU

Modbus profile (Picture 20), is used in Wiser for KNX for reading/writing Modbus registers from Magelis touch panel. Modbus registers in Magelis TP are listed as external variables in the Picture 19.

Note that Device Address 00001 in Magelis is addressed as *"type": "coil"* and *"address": "0"* in Modbus profile. It means that address is shifted by -1 in Wiser for KNX and first digit of Device Address defines type as a coil.

Modbus profile for Magelis HMI STU is attached as a file Magelis\_STU.json as a part of this AN.

	Name	Data Type	Data Source	Scan Group	Device Address	Alarm Group	Logging Group
1	 myBOOL_1	BOOL	External	ModbusSlaveT...	00001	Disabled	None
2	 myINT_2	INT	External	ModbusSlaveT...	40002	Disabled	None
3	 my_REAL_11	REAL	External	ModbusSlaveT...	40011	Disabled	None

Picture 19: List of variables in Magelis HMI STU

```
{
  "manufacturer": "Schneider Electric",
  "description": "Magelis",
  "mapping": [
    {
      "name": "Coil00001",
      "bus_datatype": "1",
      "type": "coil",
      "datatype": "boolean",
      "address": 0,
      "writable": "true"
    },
    {
      "name": "Holding40002",
      "bus_datatype": "8",
      "type": "register",
      "datatype": "int16",
      "address": 1,
      "writable": "true"
    },
    {
      "name": "Holding40011",
      "bus_datatype": "14",
      "type": "register",
      "datatype": "float32",
      "address": 10,
      "writable": "true"
    }
  ]
}
```

Picture 20: Modbus profile of Magelis HMI STU

## 6 Conclusion

Modbus integration in Wiser for KNX is configured using Modbus tab in Configurator. There are many pre-installed Modbus profiles, which specify the register map of each Modbus device. It is possible to create custom Modbus profile ( *\*.json* file) for any Modbus device and add it to Wiser for KNX. It very simplifies settings of Modbus communication with either Schneider Electric or 3<sup>rd</sup> party devices.

## 7 Appendix

### 7.1 Glossary

The following table describes the acronyms and defines the specific terms used in this document.

Abbreviation	Description
AN	Application Note
hL	Wiser for KNX
HMI	Human Machine Interface
IDE	Integrated Development Environment
JSON	Java script Object Notation
sL	Wiser for KNX
TP	Touch Panel
PLC	Programmable Logic Controller

Table 7: specific terms

Schneider Electric Industries SAS

Head Office

35, rue Joseph Monier

92506 Rueil-Malmaison Cedex

FRANCE

[www.schneider-electric.com](http://www.schneider-electric.com)