Application Note
Making TOP Server Modbus Drivers Follow Applicom TSX Premium Addressing

Purpose

This application note shows how to utilize the CSV import capabilities of TOP Server to make any member of the TOP Server Modbus Suite of drivers respond to dynamic tag addresses that follow the Applicom TSX Premium Addressing Scheme. This same methodology can be used to make TOP Server drivers respond to about any addressing scheme of another product.

This methodology is offered as a means for users to use TOP Server to replace other products in advance of product improvements to TOP Server that remove the need for these steps. We recognize that this work requires some effort; but with proper use of Excel or other CSV file editors, can quickly accomplish the desired results today.

Required Software

TOP Server Version 5.0 or higher with the TOP Server Modbus Suite drivers installed on a supported operating system and platform.
(See http://www.toolboxopc.com/html/specifications.html for specifications)

Referenced External Materials

Protocol manual, MODBUS on Ethernet TCP/IP, applicom®/Direct-Link™ PC Network Interfaces, version dated 26/04/2007 – specifically page 12 for the TSX-Premium Addressing

Background Information and the Concept

There are three pieces of key information that come together to make this solution:

1st Key Point: Mapping of Applicom Addresses to Standard Modbus Addresses

The Addressing used by the Applicom product for TSX-Premium maps to standard Modbus memory addressing as shown in the table below, which is based on the table found on page 12 of the Applicom manual referenced by this document.

Modbus Function Code	Applicom Address	Applicom DLL address	Corresponding standard Modbus Address in TOP Server
1- Read output coils	%MX0 to %MX65535	0 to 65535	000001 to 65535
15 – Write output coils	%MX0 to %MX65535	0 to 65535	000001 to 65535
3 – Read Holding Registers	%MW0 to %MW65535	0 to 65535	400001 to 465535

16 – Write Holding Registers	%MW0 to %MW65535	0 to 65535	400001 to 465535
3 – Read Holding Registers	%DW0 to %DW65534 (treating 2 consecutive words as a 32-bit double word data type)	0 to 65534	400001 to 465534
16 – Write Holding Registers	%DW0 to %DW65534	0 to 65534	400001 to 465534
3 – Read Holding Registers	%FW0 to %FW65534 (treating 2 consecutive words as a 32-bit float data type)	0 to 65534	400001 to 465534
16 – Write Holding Registers	%FW0 to %FW65534	0 to 65534	400001 to 465534

Note: by default the "Corresponding standard Modbus Address in TOP Server will have one subtracted when frames are constructed to communicate with a Modbus device, thus 000001 = 0 offset into output coils, 400001 = 0 offset into holding registers, etc.

2nd Key Point: How TOP Server Tagnames Can Be Used

TOP Server allows client applications to tell the product what to read from the device in two ways: Static or Dynamic tags.

Static tags are configured in the TOP Server and are simply an "alias" or "knickname" that maps to an actual device physical address. Typical usage is to provide an alias that relates to the process or machine – i.e "Tank1_Level" as a static tagname pointing to Modbus Address 40001.

Dynamic tags are passed into the TOP Server from a client application in the syntax of channel.device.memoryaddress@datatype. In this usage no tags are configured in TOP Server underneath the device. Many users of Invensys Wonderware Intouch use this syntax in the InTouch tagname's item name field as a way to only configure tags in one place.

Concept behind this solution: When clients use Static tags, they usually use a tagname in TOP server that is related to the process. Fact: the tagname in TOP server can be anything provided it doesn't contain spaces. So a static tagname could in fact be the addressing scheme used by another driver/opc server product.

Example:

1. Define a static tag in TOP Server with tagname %MW0
2. For address enter 400001
3. From your client, pass in channelname.devicename.%MW0

4. So your client "thinks" it is passing us a Dynamic tag – we don't know the difference – TOP Server will receive the request, go to the channelname.device name and look to see "do I have a tag named %MW0 or do I know %MW0 to be an address in the PLC?"
5. In this case the server will have a tag named %MW0 and will know how to read it.

3rd Key Point: When TOP Server Reads Tags

TOP Server ONLY reads a tag when a CLIENT requests it. So you could have thousands, or tens of thousands of Static Tags configured in TOP Server with no performance impact. Small memory impact, yes, larger if you choose to map all 65535 memory locations for each Modbus memory type supported for reading by the applicom (Output Coils and Holding Registers).

TOP Server will ONLY read tags for which it has active subscriptions/connections from clients, whether they are OPC or Suitelink clients.

If you now understand this, let us look at how the CSV Import feature of TOP Server can be used to implement this quickly and easily for hundreds, even thousands of tags.

Implementation

The idea behind making this easy for a customer to implement NOW is to supply them with CSV files pre-configured to map either a portion of or the entire memory space of a Modbus Memory type to the TOP Server addressing.

Customer can edit the CSV files or just use them as is. Customer can then import the desired CSV files for the memory types required into the TOP Server device that needs to support the TSX Premium Addressing and they are done.

Steps to import a CSV file:

1. Configure a channel and device in TOP Server
2. Right click on the device in TOP Server
3. Point to the desired CSV file and pick it.
4. You are done – all tags defined in the CSV file are now defined in the TOP Server.

TOP Server now looks just like it supports the Applicom addressing.

Maintenance is ONLY required IF the customer chooses to only import a portion of the mapping of addresses from the Applicom format to standard Modbus format.

CSV files supplied with this application note

1. Modbus_TCP_TSX_Premium_Addressing_MW_MX_Types.csv – configures 100 each of %MW and %MX addresses

2. Modbus_TCP_TSX_Premium_Addressing_FW_Types.csv – configures 50 %FW addresses

3. Modbus_TCP_TSX_Premium_Addressing_DW_Types.csv – configures 50 %DW addresses