GW-7434D
Modbus TCP Server/DeviceNet Master Gateway

User's Manual

Table of Contents

Revision

Version	Date	Author	Description
2.1	2010/08/13	Andy	1. Modify module name “I-7243D” to “GW-7434D”
2.0	2009/02/27	Johney	1. Add Get/Set Attribute functions. 2. Modify the Modbus memory mapping. 3. Firmware update between v1.1 and v2.0 or later must use EEPCopy method (section 5.1.9) to replace the old EEP data. **NOTE:** firmware v1.x is not compact with v2.0 or later
1.1	2008/07/14	Andy	1. I-7243D user manual.

1. General Information

1.1 DeviceNet Introduction

The CAN (Controller Area Network) is a serial communication protocol, which efficiently supports distributed real-time control with a very high level of security. It is an especially suited for networking "intelligent" devices as well as sensors and actuators within a system or sub-system. In CAN networks, there is no addressing of subscribers or stations in the conventional sense, but instead, prioritized messages are transmitted. DeviceNet is one kind of the network protocols based on the CAN bus and mainly used for machine control network, such as textile machinery, printing machines, injection molding machinery, or packaging machines, etc. DeviceNet is a low level network that provides connections between simple industrial devices (sensors, actuators) and higher-level devices (controllers), as shown in Figure 1.1.

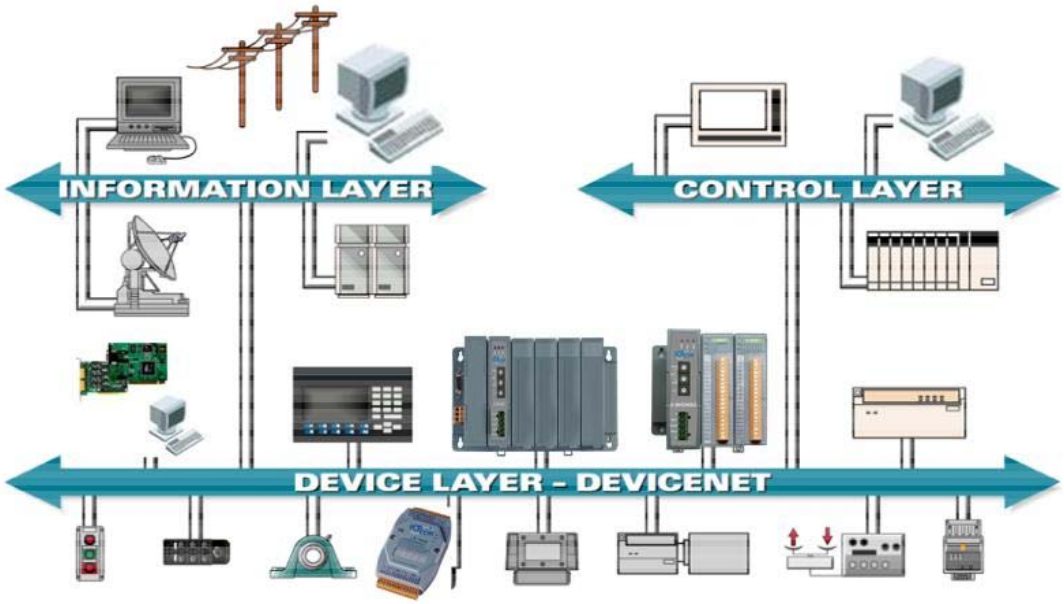


Figure 1.1 Example of the DeviceNet network

DeviceNet is a cost effective solution to one kind application of control c\area network. It reduces the connection wires between devices and provides rapid troubleshooting rejection function. The transfer rate can be up to 500Kbps within 100 meters. The transfer distance can be up to 500 meters in 125Kbps (See Table 1.1). It allows direct peer to peer data exchange between nodes in an organized and, if necessary, deterministic manner. Master/Slave connection model can be supported in the same network. Therefore, DeviceNet is able to facilitate all application communications based on a redefine a connection scheme. However, DeviceNet connection object strands as the communication

path between multiple endpoints, which are application objects that is needed to share data.

Baud rate (bit/s)	Max. Bus length (m)
500 K	100
250 K	250
125 K	500

Table 1.1 The Baud rate and the Bus length

1.2 DeviceNet Applications

DeviceNet is the standardized network application layer optimized for factory automation. It is mainly used in low- and mid-volume automation systems. Some users have also implemented DeviceNet for machine control systems. The main DeviceNet application fields include the following application area (For more information, please refer to www.odva.org):

- Production cell builds and tests CPUs
- Beer brewery
- Equipment for food packing
- Fiberglass twist machine
- Sponge production plant
- Isolation wall manufacturing
- Overhead storage bin production
- Pocket-bread bakery
- Dinnerware production
- HVAC module production
- Textile machines
- Trawler automation system
- LCD manufacturing plant
- Rolling steel door production
- Bottling line
- Tight manufacturing

1.3 Module Characteristics

"Embedded Internet" and "Embedded Ethernet" are hot topics today. Nowadays the Ethernet protocol becomes the de-facto standard for local area network. Via Internet, connectivity is occurring everywhere, from home appliances to vending machines to testing equipment to UPS…etc. Using Ethernet for network in industrial area is appealing because the required cabling is already installed.

The GW-7434D from ICP DAS is a solution that provides a communication protocol transfer the DeviceNet to Modbus/TCP protocol and solves a mission-critical problem: connecting an existing DeviceNet network to Ethernet-base PLCs and PC-based configuration and monitor system. It enables DeviceNet networks to be coupled together over the Internet/Ethernet, whereby remote monitoring and control is possible.

The GW-7434D can be a DeviceNet master device in the CAN bus on the DeviceNet network. It provides "Predefined Master Connection Set", and supports Group 2 only Server functions to communication with slave devices. On the Ethernet network, it acts as a Modbus TCP server. Users can use Modbus TCP class 0, class 1 and partial class 2 function to communicate with it. In addition, we also provide Utility software for users to configure their device parameters for the GW-7434D. The following figure shows the application architecture for the GW-7434D.

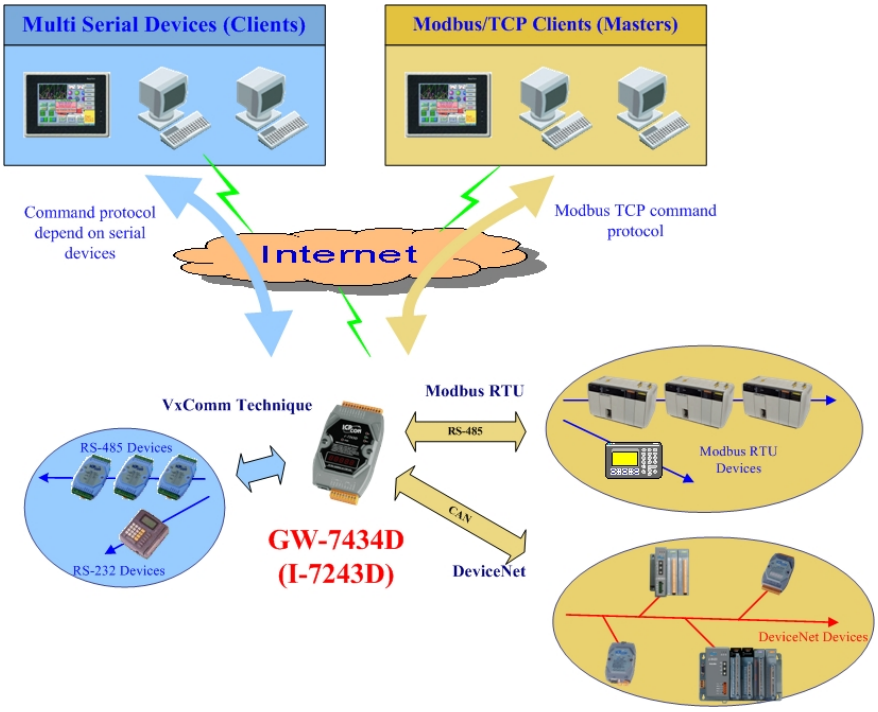


Figure 1-2: Application of GW-7434D

1.4 Hardware and Firmware Features

Hardware Features

- 80186, 80MHz CPU, or compatible
- Philip SJA1000 CAN controller with 16M Hz clock
- Phillip 82C250 CAN Transceiver
- 1K VDC isolation on CAN side.
- Support both CAN specification 2.0A and 2.0B.
- Jumper select 120Ω terminator resistor for CAN channel
- 10/100 BASE-T DM9000 compatible Ethernet Controller
- Support one RS-232 port, one RS-485 port and one CAN port
- Built-in self-tuner ASIC controller on RS-485 port
- 7-segment LED display.
- MS LED , NS LED , RUN LED

Firmware Features

- Programmable DeviceNet Master MAC ID.
- Programmable DeviceNet transfer-rate 125K, 250K, 500K.
- Supports maximum DeviceNet devices up to 63
- Predefined Master/Slave Connection Set
- The maximum Fragment number is (Input/Output) up to 64
- Supports I/O Operation Mode: Poll, Bit-Strobe and Change Of State/Cyclic
- Supports one Poll, one Bit-Strobe, one COS, one Cyclic IO connection for each DeviceNet device when connected with this module.
- Supports on-line adding device into and removing device from DeviceNet network.
- Converters single Mobbus/TCP to multi Modbus/RTU, set by Utility
- Supports VxComm technique for every COM ports of GW-7434D, set by Utility
- Act as a Modbus RTU slave device, set by Utility
- Allowed multI-Modbus TCP clients access simultaneously

1.5 Specifications

RS-232 specification:

- RS-232 interface connector: TXD, RXD, CTS, RTS, GND;
- RS-232 Baud Rate: 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 bps;

RS-485 specification:

- RS-485 interface connector: D2+, D2-;
- Self-turn ASIC inside;

Ethernet specification:

- 10/100 Base-T

CAN specification:

- CAN signal support: CAN_H, CAN_L;
- CAN bus interface: ISO 11898-2 ; 5 pin screw terminal connector.
- Isolation voltage: 1K V_{DC} isolation on the CAN side;

Power requirement:

- Unregulated +10V_{DC} ~ +30V_{DC};
- Power reverse protection, Over-Voltage brown-out protection;
- Power consumption: 3W;

Module specification:

- Dimensions: 123mm x 64.5mm x 19.6mm;
- Operating temperature: -25 to 75°C;
- Storage temperature: -30 to 80°C;
- Humidity: 10 to 95%, non-condensing;
- LEDs: Power led, MS, NS, RUN, 5-digits 7 segment led displays

Software Utility tool:

- Online adding/removing DeviceNet devices via Ethernet
- Online monitoring and configuring devices status via Ethernet.
- Get/Set Modbus/TCP input/output memory address
- Get/Set DeviceNet parameters via DeviceNet explicit message
- Support DeviceNet I/O mapping table.
- Show DeviceNet devices connection status.
- Support communication modes setting.

Application:

- Factory Automation;
- Building Automation;
- Home Automation;
- Control system;
- Monitor system;
- Vehicle Automation;

1.6 Typical Applications

Although the GW-7434D is designed as a Modbus TCP server to DeviceNet master Gateway. But it can be used to link these RS-232/RS-485/DeviceNet devices to central computer as follows:

1.6.1 Modbus TCP to multi-Modbus RTU converter

GW-7434D can be a single Modbus TCP to multi-Modbus RTU converter. You can simple use the GW-7434D Utility software to configure the device and then set connection between SCADA, HMI software and the GW-7434D. The block diagram of this application of GW-7434D is given as follows:

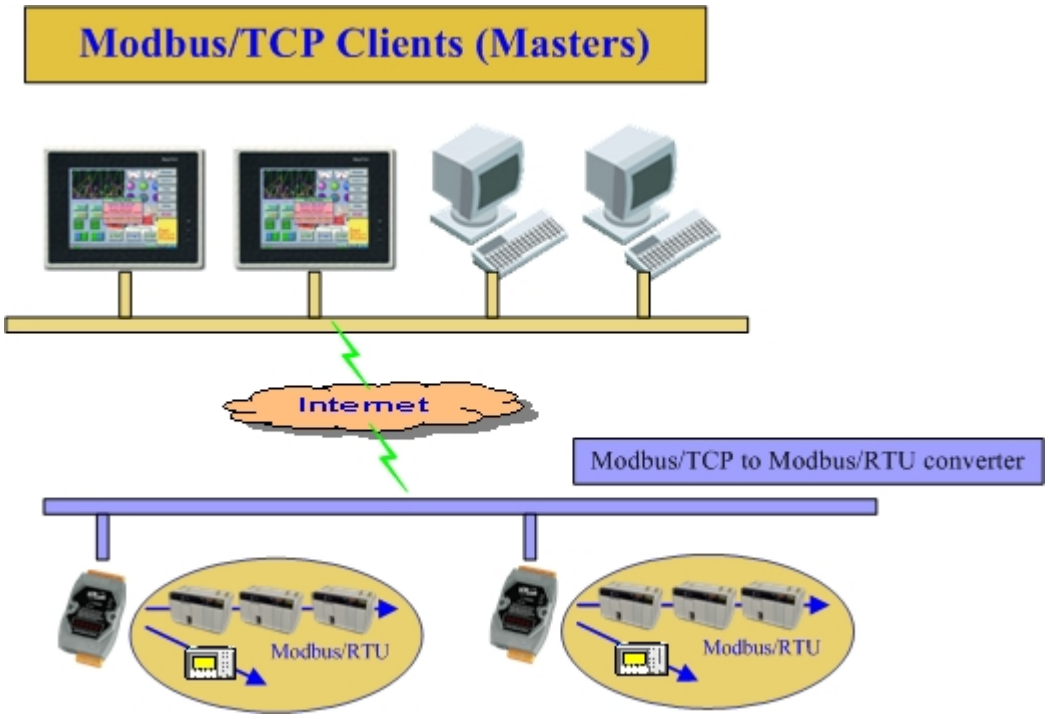


Figure 1-3: Modbus TCP to Modbus RTU application of GW-7434D

1.6.2 Protocol converter with VxComm technology

GW-7434D can also be able to link to serial devices that don't support Modbus/RTU. To use this function, you will need to install VxComm driver on host PCs. After installation, you will be able to access the remote COM ports via the standard serial driver.

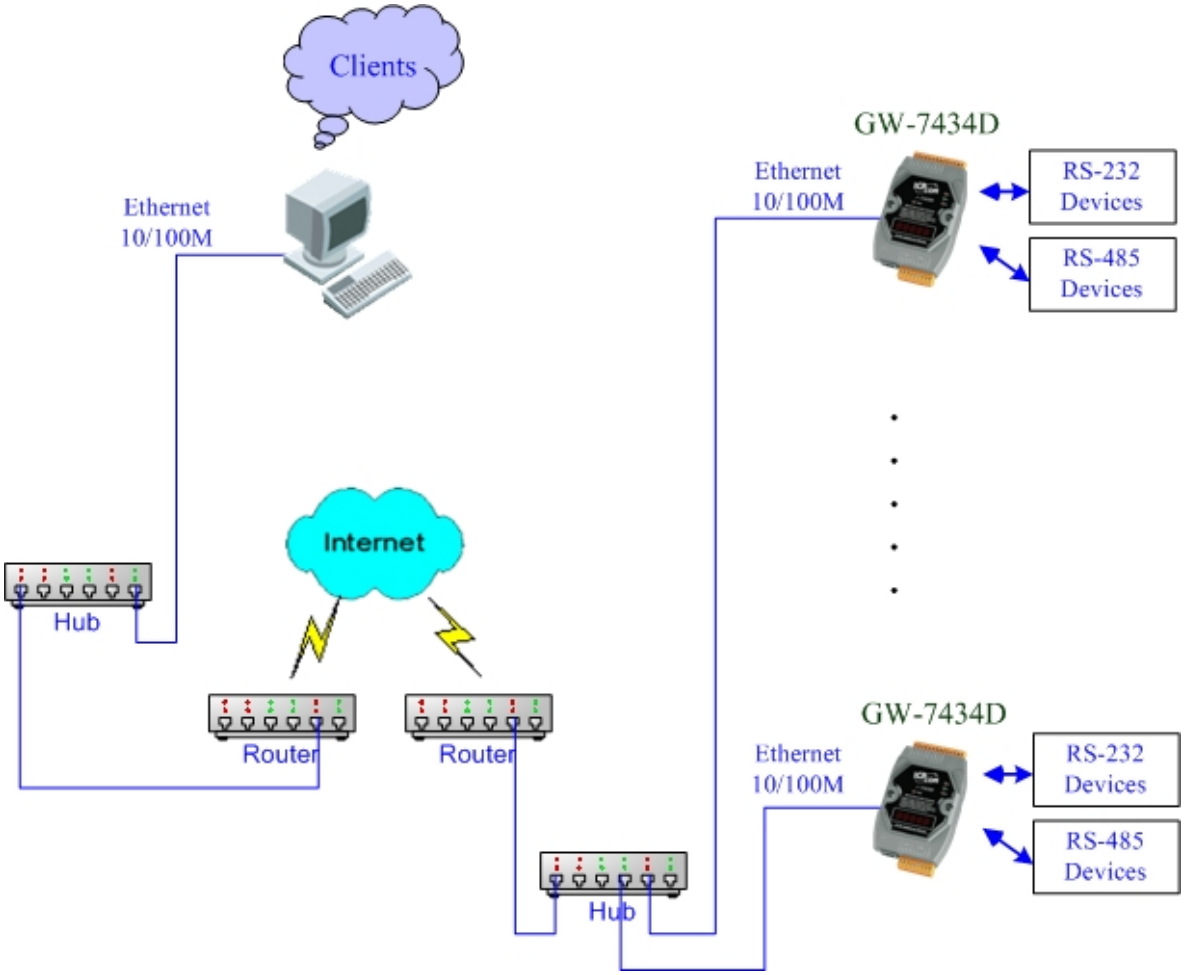


Figure 1-4: VxComm Application_1 of GW-7434D

Compared to the RS-485 network, these Ethernet network hubs are already in existence for system network. Therefore, the RS-232 devices can find the closest hub and link to the central computer with the help of the GW-7434D. The Ethernet network is extremely popular and already existing for most applications, hence, this approach is a very successful. In general, it is more difficult to write a TCP/IP program than a COM 1/2 program. **Therefore**, **the VxComm technology is developed to simulate COM-ports of the GW-7434D to become COM 3/4/5…/256 of the central computer.** Then users can write a COM port program to link these RS-232/RS-485 devices and need not to concern themselves with any TCP/IP problem.

In some factories, there are old systems still running and in case. These old systems only support COM port applications. Therefore, the Vxcomm technology can be used to upgrade these old systems to support Ethernet network.

To recap, the VxComm technology is useful as follows:

- **Provides a much easier interface for software programmers.**
- **Keeps the old systems going without program modification**

The block diagram of VxComm technology for the GW-7434D is given as follows:

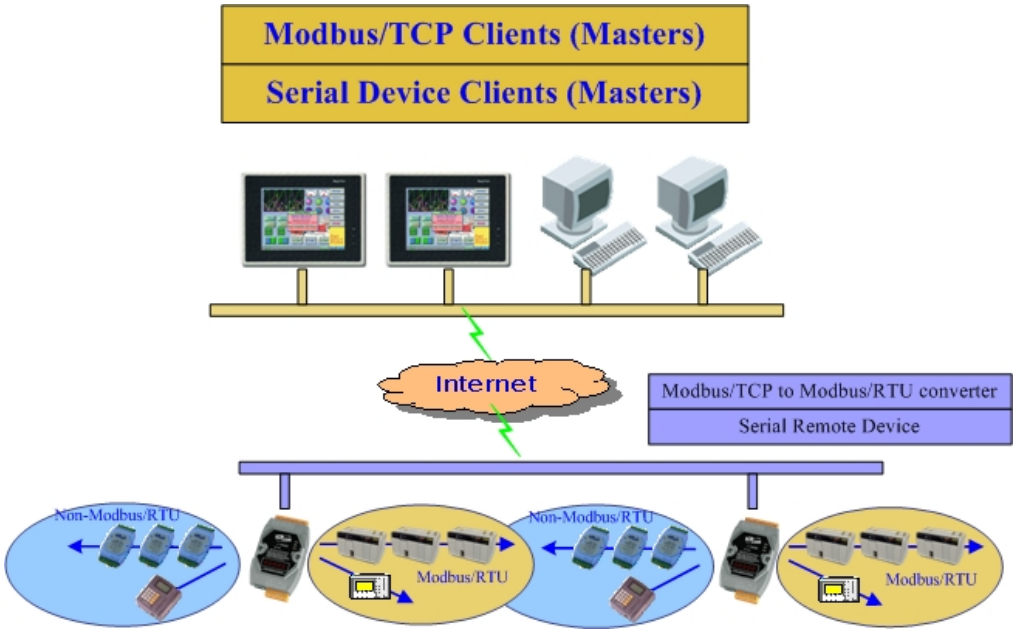


Figure 1-5: VxComm Application_2 of GW-7434D

The VxComm technology can be used to simulate COM ports of the GW-7434D to become a COM port of PC. With the help of VxComm driver, users can access the remote COM ports of the GW-7434D just as them would access the PC's COM port.

1.6.3 Modbus TCP/RTU to DeviceNet gateway

The GW-7434D provides centralized data storage, “Internal Memory”, for data that is shared between the DeviceNet and Modbus/TCP network. Data is placed into the “Internal Memory” by one network interface, allowing the data to be read/wrote through the other network interface.

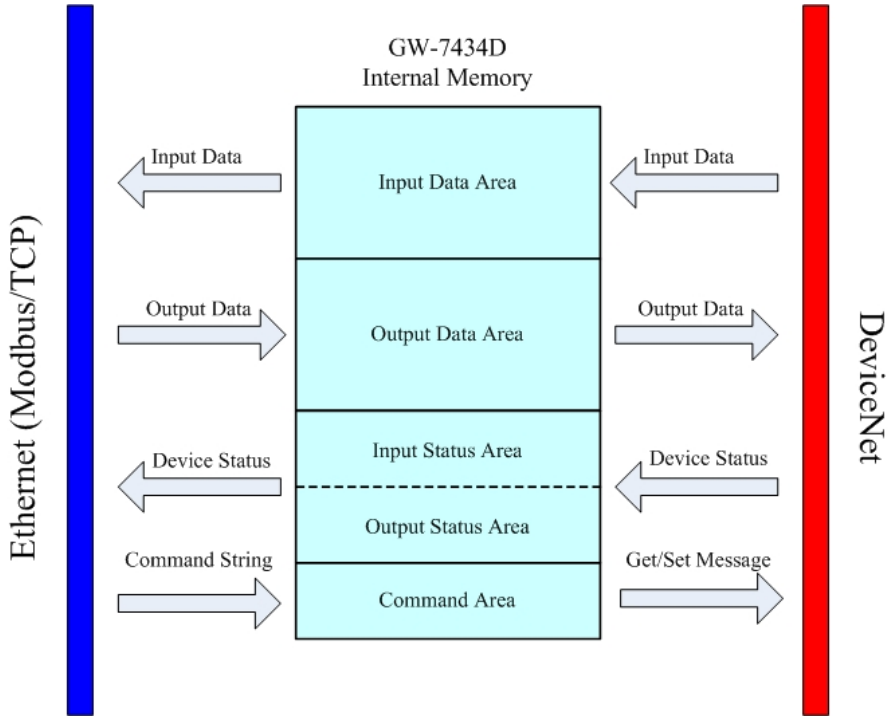


Figure 1-6: Internal Memory Table of GW-7434D

GW-7434D provides users to establish DeviceNet network rapidly by Master/Slave connection model. It can be a Modbus TCP/RTU slave to DeviceNet master gateway. Using the module, users don't need to take care of the detail of the DeviceNet protocol. The module will implement the DeviceNet protocol automatically. It can reduce the complexity of user's DeviceNet Master Software. The module mainly supports the Predefined Master/Slave Connection Set functions to allow users to merge I/O data into DeviceNet network by using Modbus/TCP commands. It can help users to establish the connection with DeviceNet slave devices easily. The general application architecture is demonstrated as Figure 1.7.

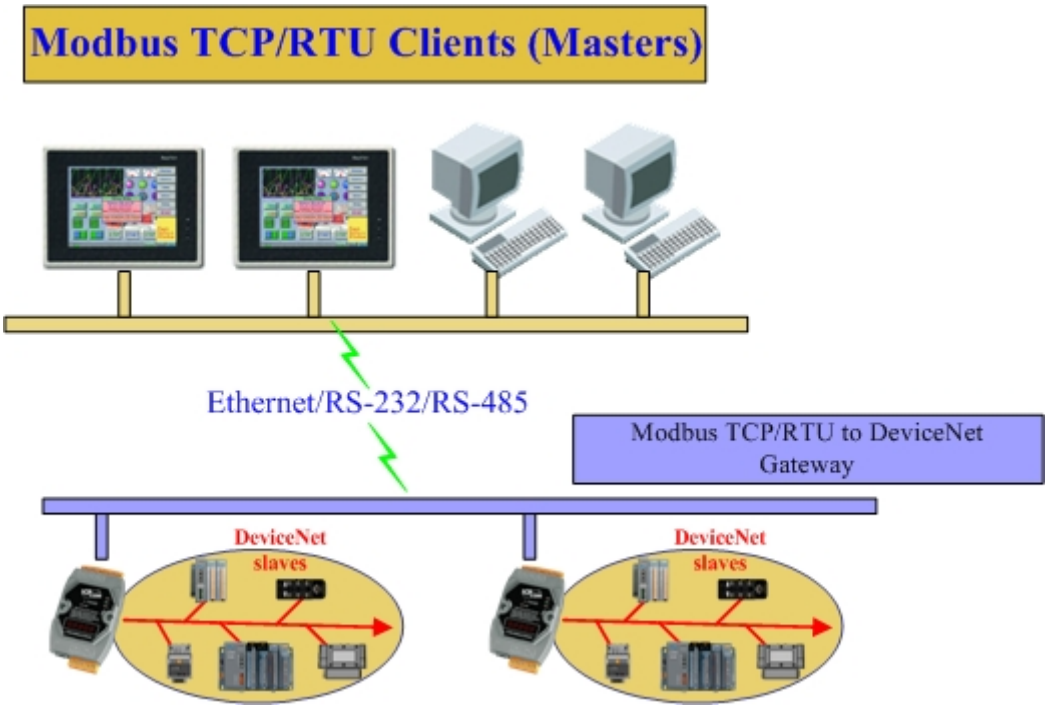


Figure 1-7: Modbus TCP/RTU to DeviceNet application of GW-7434D

The module only provides the DeviceNet Master mechanism to communicate with slave devices by the Predefined Master/Slave Connection Set, which can be clarify as two forms: One is the Explicit Message and others are I/O Messages. Note that before communicating I/O data with DeviceNet slave devices, the Master device must connect to slave devices by explicit message connection to define the connection object. Here, we only provide one explicit message connection and four I/O connections as depicted in Figure 1.8.

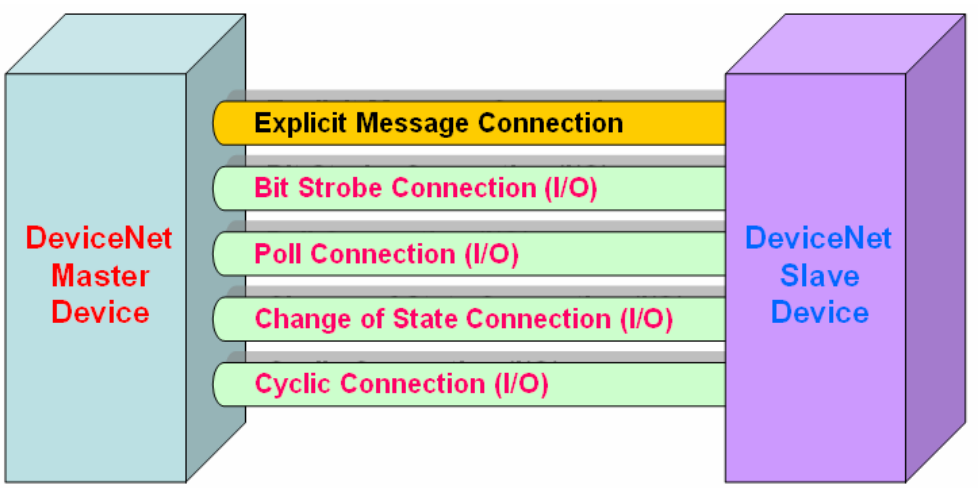


Figure 1-8: DeviceNet Messaging

The DeviceNet Communication Protocol is based on the concept of connections method. Master should create connections with slave devices based on the command of exchanging information and I/O data. To establish master control mechanism, there are only four main steps to be followed. Figure 1.9 demonstrates the basic process for the DeviceNet master communication. The every step function is described in below:

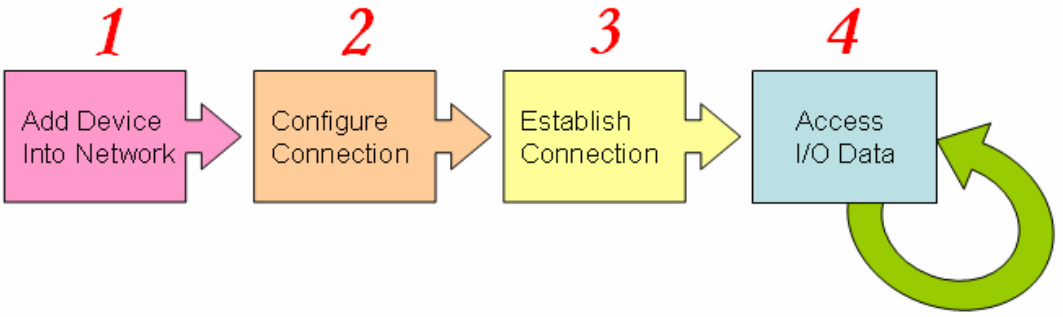


Figure 1-9: Four steps to establish connection

1. **Add device into network**

   You should provide the slave device's MAC ID to add into network.

2. **Configure connection**

   You can check the slave device's I/O connection type and the I/O data length. When configuring the I/O connection, you should provide these parameters.

3. **Establish connection**

   After configuring connections, users can start communicating with slave devices.

4. **Access I/O data**

   After communicating with slave devices, you can access the I/O data with corresponding read/write function.

After establishing the explicit connection, the connection path is then used to exchange the general information from one node to the others. And then users can create the I/O connections in the next step. Once I/O connections have been created, I/O data may be exchanged among devices in the DeviceNet network according to master device demand. Therefore, the master device can access I/O data of the slave devices by one of the four I/O connection methods. The module is not only easy to use but also providing a lot of the Modbus/TCP commands to retrieve and deliver the slave's I/O data. For more information, please refer to command description in section 6.

2. Hardware

2.1 Pin Assignment

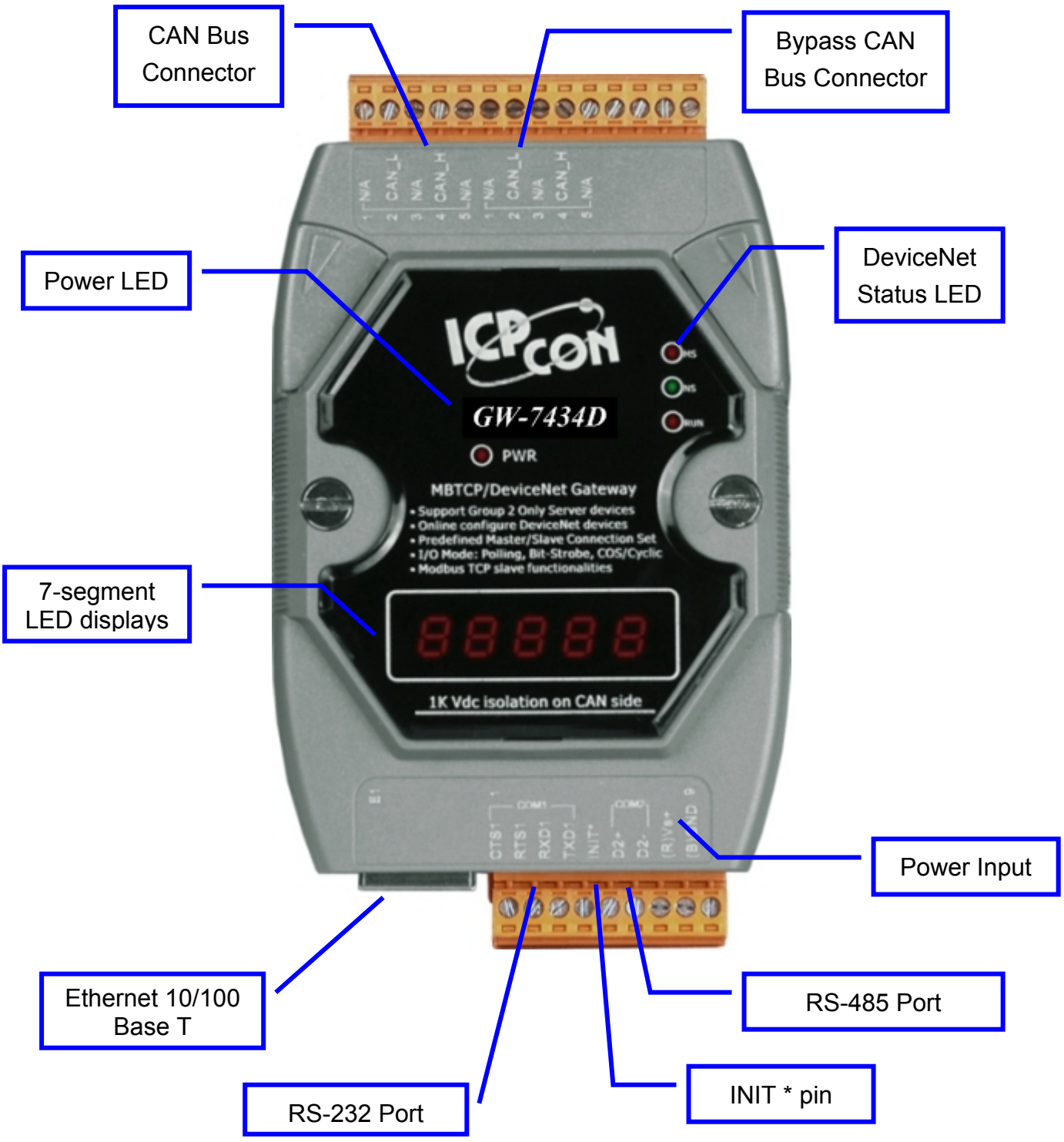
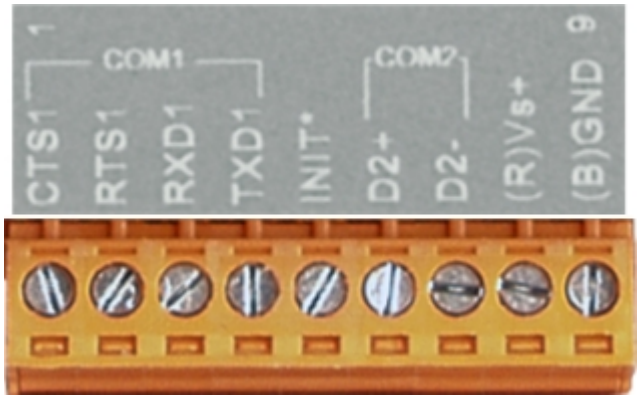


Figure 2-1: Pin assignment on the GW-7434D

2.1.1 RS-232 & RS-485 & Power supply interface

The GW-7434D provides one RS-232 interface and one RS-485 interface with hardware flow control. The GND-signal of COM1 is shared with pin-9, GND. The pin assignment is shown in table 2-1.

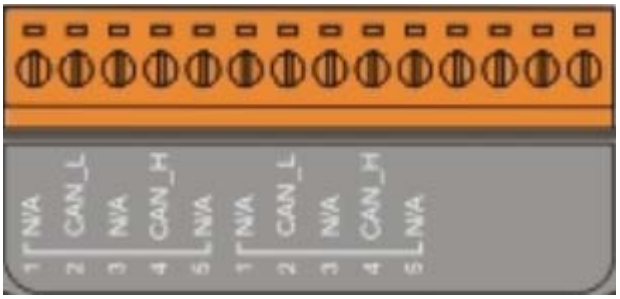


Pin	Name	Description
1	CTS1	CTS pin of COM1 (RS-232)
2	RTS1	RTS pin of COM1 (RS-232)
3	RXD1	RXD pin of COM1 (RS-232)
4	TXD1	TXD pin of COM1 (RS-232)
5	INIT*	Initial pin for enable/disable AUTOEXEC.BAT
6	D2+	Data+ pin of COM2 (RS-485)
7	D2-	Data- pin of COM2 (RS-485)
8	Vs+	V+ of power supply (+10V to +30V DC unregulated)
9	GND	GND of power supply

Table 2-1: COM Connector Pin Assignment

2.1.2 Connect to DeviceNet devices

In order to provide an easy CAN bus wiring, the GW-7434D supplies one CAN port with two CAN bus connector interfaces. Each connecter built on the GW-7434D looks like as figure 2-3 and table 2-2.



Pin No.	Signal	Description
1	N/A	Unavailable
2	CAN_L	CAN_L bus line (dominant low)
3	N/A	Unavailable
4	CAN_H	CAN_H bus line (dominant high)
5	N/A	Unavailable

Table 2-2: CAN bus Connector Pin Assignment

Note that the bypass CAN bus connector is not another CAN channel. It is designed for connecting to another CAN device conveniently. The structure of the inside electronic circuit is displayed as figure 2-2.

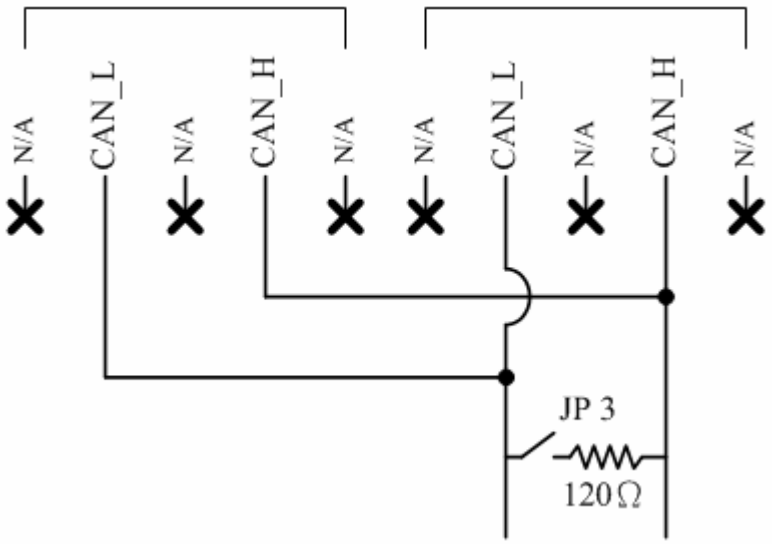
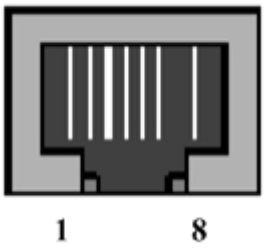


Figure 2-2 Electronic circuit of CAN bus connector

2.1.3 Ethernet connection

The Ethernet (10/100 Base-T) signals are routed to an RJ45 socket for easy connection using a standard CAT 3 or CAT 5 network cable. On power on of the GW-7434D, it will auto-negotiate the network speed and connection.



Pin	Name	Description
1	TX+	Transmit Data +
2	TX-	Transmit Data -
3	RX+	Receive Data +
4	N.C.	Not Connected
5	N.C.	Not Connected
6	RX-	Receive Data -
7	N.C.	Not Connected
8	N.C.	Not Connected

Table 2-3: Ethernet Connector Pin Assignment

2.2 Terminator resistor settings

In order to minimize reflection effects on the CAN bus line, the CAN bus lines have to be terminated at both ends by two terminal resistances. Based on the ISO 11898-2 spec, each terminal resistance is 120Ω (or between 108Ω~132Ω). The length related resistance should have 70 mΩ/m. Users should check the resistances of their CAN bus, before they install a new CAN network as figure 2-3.

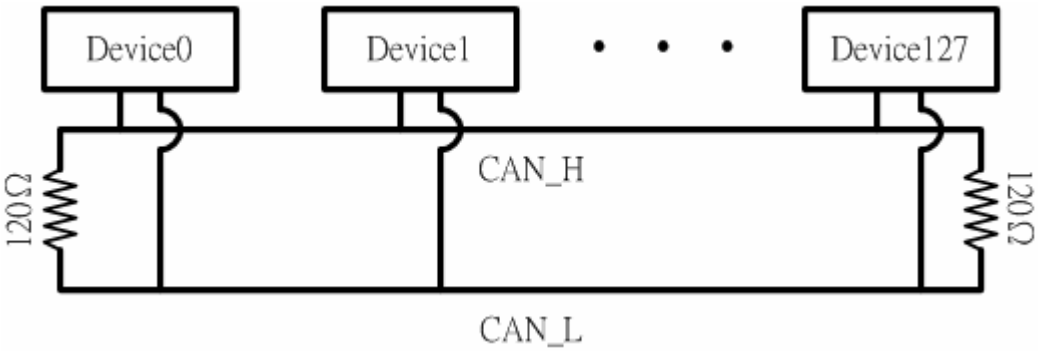


Figure 2-3: Terminator resistor

Moreover, to minimize the voltage drop on long distance, the terminal resistance should be higher than the value defined in the ISO 11898-2. Table 2-4 may be used as a reference.

Bus Length (meter)	**Bus Cable Parameters**		**Terminal Resistance (Ω)**
	Length Related Resistance (mΩ/m)	**Cross Section (Type)**	
0~40	70	0.25(23AWG)~ 0.34mm^2(22AWG)	124 (0.1%)
40~300	< 60	0.34(22AWG)~ 0.6mm^2(20AWG)	127 (0.1%)
300~600	< 40	0.5~0.6mm^2 (20AWG)	150~300
600~1K	< 20	0.75~0.8mm^2 (18AWG)	150~300

Table 2-4: Relation between bus cable and length

Therefore, the GW-7434D module supplies a jumper for users to connect the terminator resistor or not. If users want to use this terminator resistor, please open the GW-7434D cover and use the JP3 jumper to activate the 120Ω terminator resistor built in the system, as in the figure 2-4. Note that the default setting is active. And about the J3 jumper setting, please refer the table 2-5.

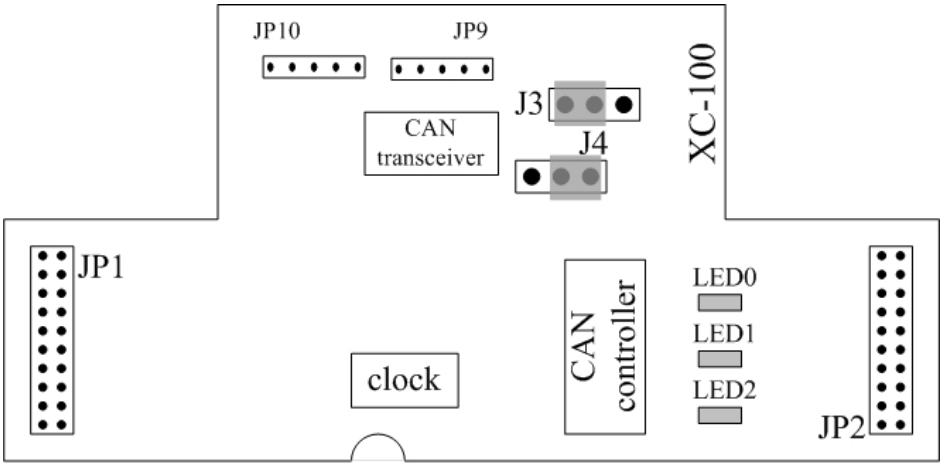


Figure 2-4 XC100 I/O expansion board LAYOUT

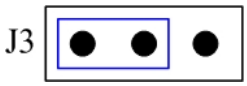
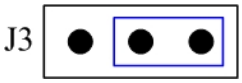
Apply the termination resistor(120Ω)	Don't apply the termination resistor
	

Table 2-5 J3 Jumper Selection

2.3 LED Indication

The GW-7434D acts as a Modbus/TCP to DeviceNet gateway. It provides some LEDs to indicate what situation the GW-7434D is in.

2.3.1 Power LED

There is a red indicator-LED in the GW-7434D as follow:

- ✧ Firmware is running: **flashing red**

The default shipping of GW-7434D will be firmware inside, so the red indicator-LED of GW-7434D will be flashing two times per second periodically.

2.3.2 Module Status indicator LED

The GW-7434D includes three single-color LED displays to indicate the status of module, network and I/O device. They are MS LED (it is red), NS LED (it is green), and RUN LED (it is red). The Indicators assist maintenance personnel in quickly identifying a problem unit. When the GW-7434D events occur, these indicators will be triggered to glitter with different conditions.

- MS LED

This led provides module status and indicates whether or not the module is operating properly. Table 2-6 shows the conditions of MS status. Therefore, when the GW-7434D is operated normally, the MS led must be turned off.

Condition	**Description**
Off	Module is normal; no error occurs
Red	Module has unrecoverable fault
Flashing red	Module has recoverable fault. To recover: Reset device or perform error recovery

Table 2-6 MS led conditions

- NS LED

This led indicates the DeviceNet communication status of the module. Table 2-7 shows the conditions of NS status. When module is online and start to communicate with the devices, it will be solid green. If there are some devices disconnected with the GW-7434D, the NS led will be flashing.

Condition	**Description**
off	Module stops to communicate with all devices
Flashing green	There exists at least one device disconnect with the module
Solid green	Module is online and start to communicate with all devices

Table 2-7 NS led conditions

- RUN LED

This led indicates the configuration status of the GW-7434D. Table 2-8 shows the conditions for RUN status. If there is no configuration, the RUN led will be flashing. After configuring the DeviceNet devices by GW-7434D Utility tool, the GW-7434D will start to communicate with them and the RUN led will becomes solid red.

Condition	**Description**
Off	Some errors occur on the module
Flashing red	Module is waiting for configuring
Solid red	Module has been configured O.K.

Table 2-8 RUN led conditions

2.3.3 5-digits 7-Segment LED Displays

The 5-digits 7-SEG LED will show as figure 2-5.

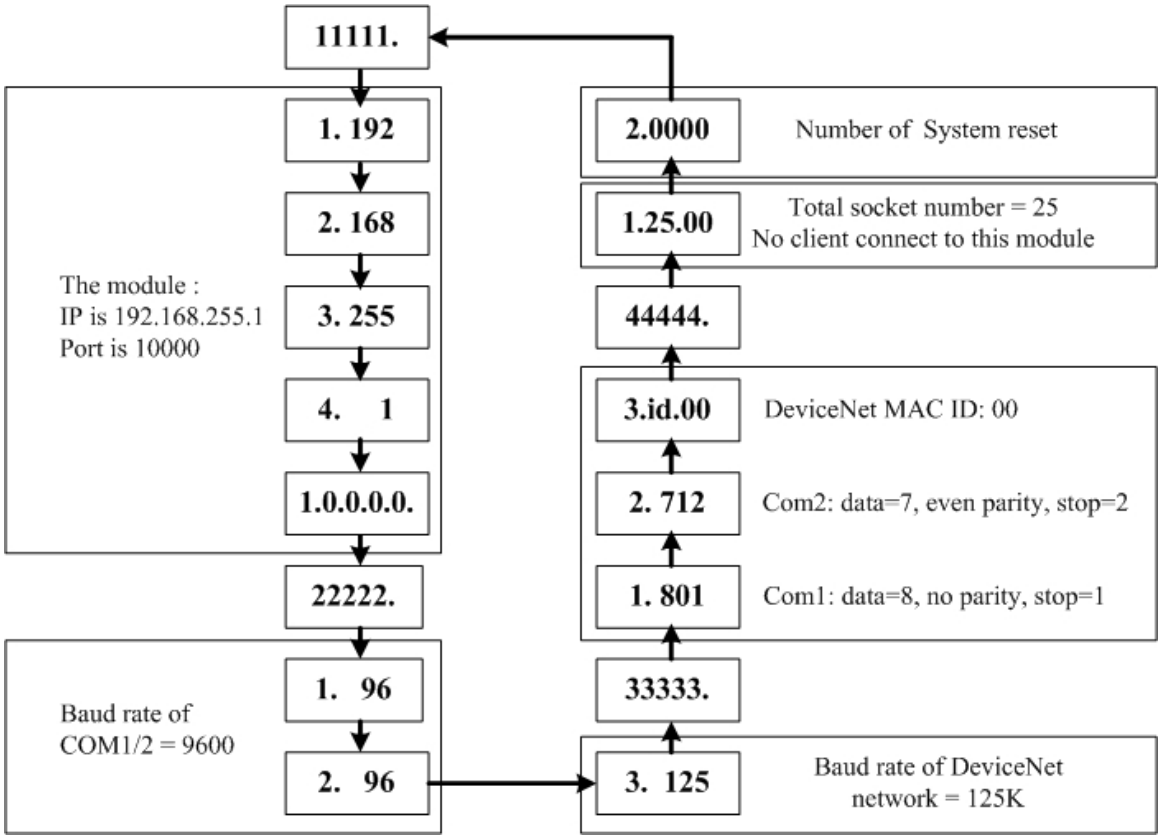


Figure 2-5 7-SEG LED Displays

The important information of GW-7434D can be divided as follows:

- Group-ID 11111: IP information of this GW-7434D
- Group-ID 22222: baud rate of all ports
- Group-ID 33333: configuration of all ports
- Group-ID 44444: client-connected information and system reset state of this GW-7434D

The IP information format of GW-7434D is given as follows:

- Group-ID of 5-digit LED: 11111.
- LED-1: indicator, can be 1 or 2 or 3 or 4
- LED-2~5: IP

The LED will show Group-ID first, and then show its IP as the above diagram indicates. If users change IP, the value shown will change immediately. The default

shipping IP = 192.168.255.1 ➔ the LED-show sequence is given as above diagram.

The baud-rate format of COM ports are given as follows:

- Group-ID of 5-digit LED: 22222.
- LED-1: COM port number
- LED-2~5: value of (baud/100)

The baud-rate format of CAN port is given as follows:

- LED-1: indicator, 3, CAN port.
- LED-2~5: value of (baud/1000)

The COM port are shown in LED-1 and their baud rate is shown in the LED-2~5. The COM port baud rate = (value of LED-2~5)*100. Therefore, shown-value=1. 96 means baud rate of COM1=9600BPS; shown-value= 2.1152 means baud rate of COM2= 115200 BPS. It's the same as the CAN port baud rate. The CAN port baud rate = (value of LED-2~5)*1000. Therefore, shown-value=3. 125 means baud rate of DeviceNet network =125KBPS; shown-value= 3.500 means baud rate of DeviceNet network= 500KBPS. All baud rate of GW-7434D's port will be shown one by one.

The configuration of COM ports are given as follows:

- Group-ID of 5-digit LED: 33333.
- LED-1: COM port number
- LED-3: data bit, 7 or 8
- LED-4: parity bit, 0=no parity, 1=even parity, 2=odd parity
- LED-5: stop bit, 1 or 2

The configuration of CAN port is given as follows:

- LED-2/3: fix string, "id.".
- LED-4/5: DeviceNet MAC ID of this module, default 00.

The connection-client information is given as follows:

- Group-ID of 5-digit LED: 44444.
- LED-2/3: numbers of free sockets are available, default 25.
- LED-4/5: numbers of sockets are used by clients, default 0.

The reset state of system is given as follows:

- LED-2~5: reset number, display in decimal mode.

When the GW-7434D is powered-off or just been reset, the reset number will be increased. If any one client connects to this GW-7434D, the free-sockets will be decreased and used-sockets will be increased. If the free-sockets number is reduced to 0, then no extra client can link to this GW-7434D. The default number of free-sockets is 25. Therefore, the GW-7434D allows 25 clients link to it.

3. DeviceNet Interface

3.1 Network Communication

The GW-7434D, Modbus/TCP to DeviceNet Gateway, acts as a DeviceNet master on DeviceNet network. It can exchange I/O data with up to 63 nodes. Users can use the GW-7434D Utility tool to access the module over Ethernet network via Modbus/TCP commands.

3.2 Slave Device Communication

After the configuration of slave devices, the GW-7434D will start to establish connections with device in the scan list (list of configured slaves). Once connections are established, the module will perform all necessary steps to configure the required I/O messaging.

The GW-7434D provides explicit message proxy services for all “Group 2 Only Server” devices. Once any Group 2 Only devices are configured, the GW-7434D will send message to the devices, explicit message connection value for watchdog_timeout_action to “Deferred Delete”, in addition to the I/O messages. This function prevents the explicit message connections between the GW-7434D and the slave from timing out when communicating with I/O messages.

The GW-7434D supports four I/O messaging types specified by the DeviceNet protocol. These include polling, bit-strobe, COS, Cyclic I/O messages. I/O messaging and I/O parameters are configured by using the GW-7434D Utility tool.

3.3 Scan Cycles

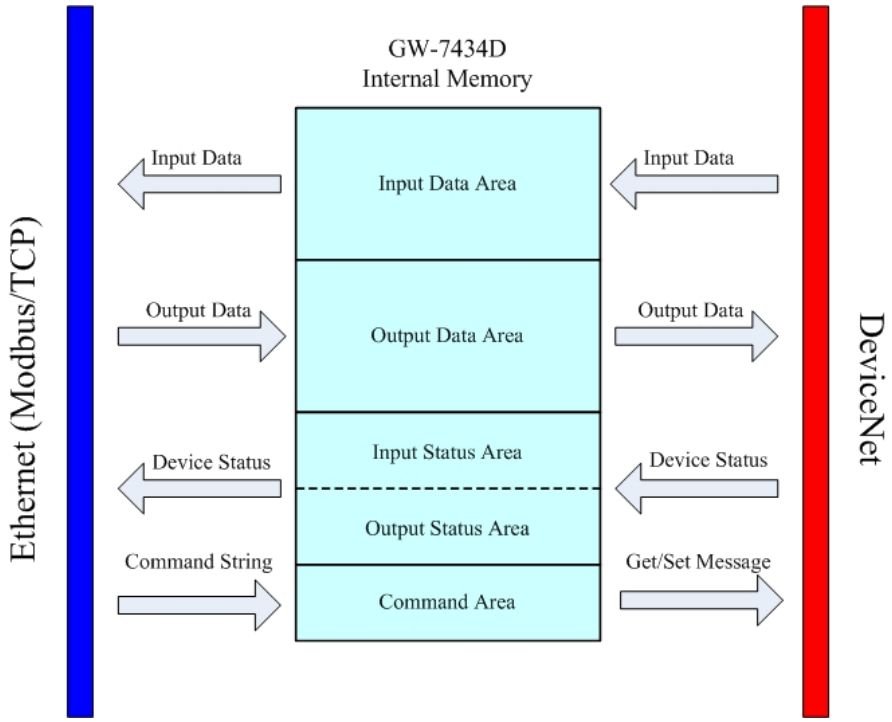
The GW-7434D employs a scan cycle for producing I/O messages. A scan cyclic consists of the following:

- Polling messages to every device configured as polled I/O
- Bit-Strobe output messages to devices configured as bit-strobe.

If the internal timer exceeds the explicit packet rate of certain I/O connections, the I/O message will be sent to the slave devices.

3.4 Interaction with Internal Memory

The DeviceNet interface in the GW-7434D accesses the I/O data areas from slave I/O connections processed by the DeviceNet master; there is no synchronization between the Ethernet and DeviceNet network interfaces. As shown in the following picture.



When an I/O connection with a slave requires that output data be sent to the slave, it will be read from the Output Data Area. The data read is what was placed there by the last write to the Output Data Area by using Modbus/TCP commands.

When input data is received from DeviceNet I/O connection, it is copied to the Input Data Area. This data is available to be read by the Modbus/TCP commands on the next data exchange.

All the connection status and configuration are stored in the Status Area of GW-7434D's Internal Memory. And these data are available to be read by the Modbus/TCP commands at any time.

The Command Area is used for saving Modbus/TCP commands that contain specific data formats packed in the "Force Multiple Registers" command, function code: 16. After receiving this specific data format command, the GW-7434D will unscramble it and perform the relative procedure.

4. Modbus/TCP Interface

The GW-7434D supports Modbus/TCP commands. The implementation of the Modbus/TCP server is done according to the Modbus/TCP specification 1.0. All commands according to class 0, class 1 and partially class 2 slave functionalities are implemented.

The module can handle maximum 25 simultaneous Modbus TCP connections.

4.1 Commands

The following Modbus/TCP commands are supported by the GW-7434D.

Function Code	Function Name	Affects	Address Method
1	Read Coils	IN/OUT	Bit
2	Read Input Discrete	IN/OUT	Bit
3	Read Multiple Registers	IN/OUT	Word (2 bytes)
4	Read Input Registers	IN/OUT	Word (2 bytes)
5	Write Coil	OUT	Bit
6	Write Single Register	OUT	Word (2 bytes)
15	Force Multiple Coils	OUT	Bit
16	Force Multiple Registers	OUT	Word (2 bytes)

Table 4-1: Modbus Commands

4.2 Exception Codes

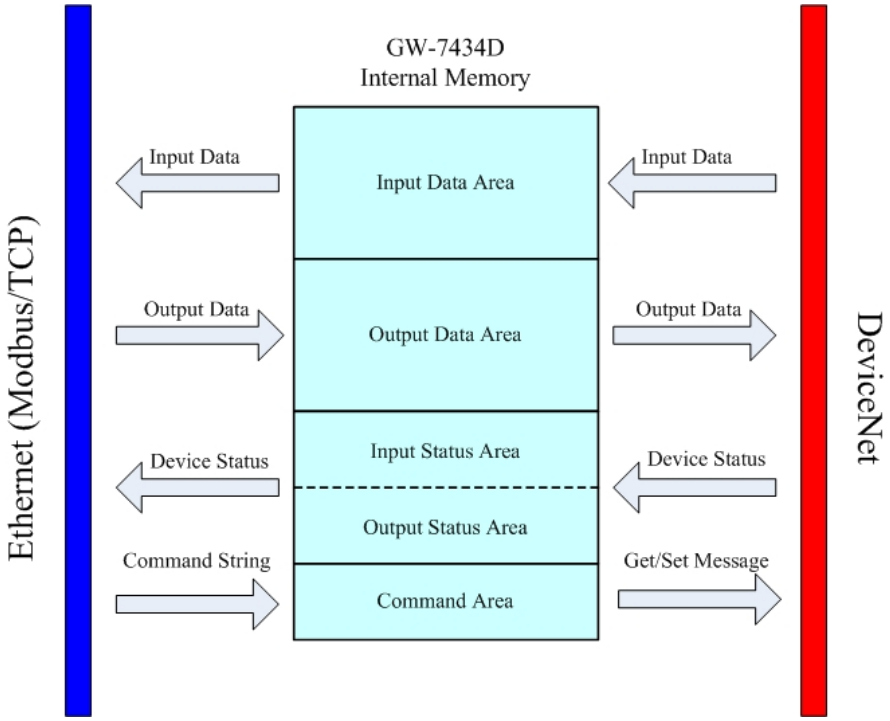
An exception code is returned in the response when the GW-7434D is unable to service the Modbus request that was received. The following exception codes will be used by the GW-7434D.

Exception Code	Name	Description
01	Illegal Function	The module does not support the function code in the query
02	Illegal Data address	The data received in the query is outside the initialized memory area
03	Illegal Data Value	The data in the request is illegal

Table 4-2: Exception Codes

4.3 Modbus/TCP Addressing

The GW-7434D supports an “Internal Memory” for saving I/O data. The input and output data areas are set to a maximum size of 256 words each. The command area is 255 words. The input status area is 4642 words and the output status area is 322 words. When accessing these areas, by Modbus commands, the address is according to the following tables.



4.3.1 Input/Output Data Areas

The I/O Data Area is used for users to access devices I/O data directly. After using Utility tool to set the memory mapping of the I/O devices, users can get/set the I/O data according to the mapping address of each I/O data.

Users can use Modubs FC4 command, read input registers, to get the input data values from GW-7434D's input data area.

Word Address (3x)
0000h
0001h
0002h
:
00FFh

Table 4-3: Input Addressing

Users can use Modbus FC16 command, force multiple registers, to set the output data values into GW-7434D's output data area.

Word Address (4x)
0000h
0001h
0002h
:
00FFh

Table 4-4: Output Addressing

I/O Data Format

The GW-7434D transfers I/O data between Modbus/TCP and DeviceNet without regard to data content or format. Due to this, the user is responsible for making sure that the devices on either network understand the format of the data.

DeviceNet is a little-endian protocol; values are transmitted least significant byte first. Therefore, all data in the I/O Data Areas is assumed, by DeviceNet nodes, to be stored as little-endian.

Users should to make sure the Modbus/TCP master handles input data and transmits output data in a format acceptable to the target DeviceNet devices (least significant byte first). The user must know the I/O Data Areas that DeviceNet data has been mapped.

4.3.2 Command Area

The Command Area is used for saving Modbus/TCP commands that contain specific data formats packed in the “Force Multiple Registers” command, function code: 16. After receiving this specific data format command, the GW-7434D will unscramble it and perform the relative procedure.

Users can use Modubs FC16 command, force multiple registers, to send specific data format into GW-7434D’s command area. These specific data format types, can be accepted by GW-7434D, are listed and described in the section 6.

Word Address (4x)
0C00h
0C01h
0C02h
:
0CFFh

Table 4-5: Command Addressing

4.3.3 Output Status Area

The Output Status Area is used for saving information for DeviceNet output devices. Users can use Modubs FC3 command, read multiple registers, to read the data from GW-7434D's Output Status Area (**0800h ~ 0BF0h**).

Word Address 0800h ~ 0BF0h: Specific data format string.

For saving specific data format string that are sent by Utility tool to configure output device data address. The types of the specific data format are described in following table. Each specific data format contains 32 bytes. And the maximum number of the command string is 100.

Word Address (4x)	Description	Data Length	Data
0800h(High)	Is the slave valid	1 Bytes	0x00 : Not valid 0x01 : Explicit valid 0x03 : I/O valid
0800h(Low)	Is the "start address" valid	1 Bytes	0x00 : Not valid 0x01 : valid
0801h(High)	Slave MAC ID = 0x00	1 Byte	0x00
0801h(Low)	Slave ID 0, Connection Type	1 Byte	0x00: Explicit 0x01: Poll 0x02: Bit-Strobe 0x03: COS 0x04: Cyclic
0802h	Slave ID 0, Start address of Output Data Area	2 Byte	0x0000 ~ 0x01FF
0803h	Slave ID 0, Output data length	2 Byte	0x0000 ~ 0x01FF
0804h	Slave ID 0, Communication Status	2 Byte	Refer to Appendix A.
0805h	Slave ID 0, Expected Packet Rate	2 Byte	0x0001 ~ 0xFFFF
0806h ~ 080Fh	Reserve	22 Byte	Reserve
0810h ~ 081Fh	Slave ID 1 data format	32 Byte	Slave ID 1 data
0820h ~ 082Fh	Slave ID 2 data format	32 Byte	Slave ID 2 data
:	:	:	
0BF0h~0BFFh	Slave ID 63 data format	32 Byte	Slave ID 63 data

4.3.4 Input Status Area

The Input Status Area is used for saving information for each DeviceNet input device. Users can use Modubs FC4 command, read input registers, to read the data from GW-7434D's Input Status Area.

Word Address (3x)	Description
1000h	The information of all searched devices.
1001h	
:	
14FFh	
1500h	Get Attribute Response data.
:	
16FFh	
1700h	Set Attribute Response data.
:	
170Fh	
1710h(High)	Master MAC ID
1710h(Low)	Baud rate
1711h	Firmware version
1712h	GW-7434D DeviceNet Master Status
1713h	The amount of all slave devices
1714h ~ 171Fh	Reserved
1720h(High)	The leading byte of the DNM_xxx command
1720h(Low)	The No. of the DNM_xxx command
1721h	The return code of the DNM_xxx command
1722h ~ 17FFh	Reserved
1800h	Specific data format string for Input command
:	
1BF0h	

Table 4-7: Input Status Addressing

Word Address 1000h ~ 14FFh: All searched slave devices.

The users can use DNM_AutoSearch function to find out all the slaves which are connected with the GW-7434D module. After searching successfully, the slaves' information would be store at this address segment. Each slave contains 6 bytes information. The data format is described in the following table.

Word Address (3x)	Description	Data Length	Data
1000h	The amount of slave devices.	2 Bytes	0x0001 ~ 0x003E
1001h(High)	First slave's MAC ID	1 Bytes	0x00 ~ 0x3F
1001h(Low)	First slave's Connection Type	1 Byte	0x00: Explicit 0x01: Poll 0x02: Bit-Strobe 0x03: COS 0x04: Cyclic
1002h	First slave's input data length	2 Byte	0x0000 ~ 0x01FF
1003h	First slave's output data length	2 Byte	0x0000 ~ 0x01FF
1004h(High)	Second slave's MAC ID	1 Bytes	0x00 ~ 0x3F
1004h(Low)	Second slave's Connection Type	1 Byte	0x00: Explicit 0x01: Poll 0x02: Bit-Strobe 0x03: COS 0x04: Cyclic
1005h	Second slave's input data length	2 Byte	0x0000 ~ 0x01FF
1006h	Second slave's output data length	2 Byte	0x0000 ~ 0x01FF
1007h ~ 1009h	3rd slave's information	6 Byte	
100Ah ~ 100Ch	4rd slave's information	6 Byte	
:	:	:	

Word Address 1500h ~ 16FFh: Get Attribute Response Data.

The users can use DNM_GetAttribute function to get extra slave's information. After calling DNM_GetAttribute, the users would receive response data from the slave. The data format is described in the following table.

Word Address (3x)	Description	Data Length	Data
1500h(High)	Slave ID	1 Byte	0x0000 ~ 0x003F
1500h(Low)	Class ID	1 Byte	0x00 ~ 0xFF
1501h(High)	Instance ID	1 Byte	0x00 ~ 0xFF
1501h(Low)	Attribute ID	1 Byte	0x00 ~ 0xFF
1502h	Attribute Data Length	2 Byte	0x0000 ~ 0x01FF
1503h(High)	Attribute Data 0	1 Bytes	0x00 ~ 0xFF
1503h(Low)	Attribute Data 1	1 Byte	0x00 ~ 0xFF
1504h(High)	Attribute Data 2	1 Bytes	0x00 ~ 0xFF
1504h(Low)	Attribute Data 3	1 Byte	0x00 ~ 0xFF
:	:	:	

Note:

Word Address 1502h: The data length after getting attribute

If the value is 0xFFFF, it means that the "DNM_GetAttribute" command has some errors. The error code would be shown in 1503h. The error code description was shown in Appendix A.

If the value is the range from 0x0001 to 0x01FF, it means that the "DNM_GetAttribute" command executes successfully. The attribute data would be shown between 1503h and 16FFh.

Word Address 1700h ~ 170Fh: Set Attribute Response Data.

The users can use DNM_SetAttribute function to set extra slave's information. After calling DNM_SetAttribute, the users would receive response data from the slave. The data format is described in the following table.

Word Address (3x)	Description	Data Length	Data
1700h(High)	Slave ID	1 Byte	0x0000 ~ 0x003F
1700h(Low)	Class ID	1 Byte	0x00 ~ 0xFF
1701h(High)	Instance ID	1 Byte	0x00 ~ 0xFF
1701h(Low)	Attribute ID	1 Byte	0x00 ~ 0xFF
1702h	Response Data Length	2 Byte	0x0000 ~ 0x01FF
1703h	Response Data	2 Bytes	0x00 ~ 0xFF
1704h ~ 170Fh	Reserve	---	---

Note:

Word Address 1702h: The data length after setting attribute

If the value is 0xFFFF, it means that the "DNM_SetAttribute" command has some errors. The error code would be shown in 1703h. The error code description was shown in Appendix A.

If the value is 0x0002, it means that the "DNM_SetAttribute" command executes successfully. The response data would be shown in 1703h.

Word Address 1710h ~ 17FFh: Module information.

This data field would show the module information. The users can always monitor this field to ensure that the module is working fine or not. The data format is described in the following table.

Word Address (3x)	Description	Data Length	Data
1710h(High)	DeviceNet Master MAC ID	1 Byte	0x0000 ~ 0x003F
1710h(Low)	CAN Bus Baud Rate	1 Byte	0x00: 125K bps 0x01: 250K bps 0x02: 500K bps
1711h	Firmware Version	2 Byte	Note 1.
1712h	DeviceNet Master Status	2 Byte	Refer to Appendix A
1713h	The amount of all slave devices	2 Byte	0x0000 ~ 0x003F
1714h ~ 171Fh	Reserve	---	---
1720h(High)	Leading Byte	1 Byte	0x40(@) or 0x24($) Note 2
1720h(Low)	Command No.	1 Byte	0x00 ~ 0xFF
1721h	Return Code	2 Byte	Refer to Appendix A
1722h ~ 17FFh	Reserve	---	---

Note 1:

Word Address 1711h: The firmware version of GW-7434D

Example1: version = 2.01 , 01711h = 0x0201

Note 2:

Word Address 1720h ~ 1721h: The Command No. and Return Code

When the users send command (DNM_xxxx) to the GW-7434D, this data field would shows the return code of the command. This data field would help users to ensure that the command has been executed successfully or not.

The “Leading Byte” field shows the leading byte of command which has been send by the users.

The “Command No.” field shows the number of the command which has been send by the users.

The “Return Code” field shows the return code after executing the command which has been send by the users.

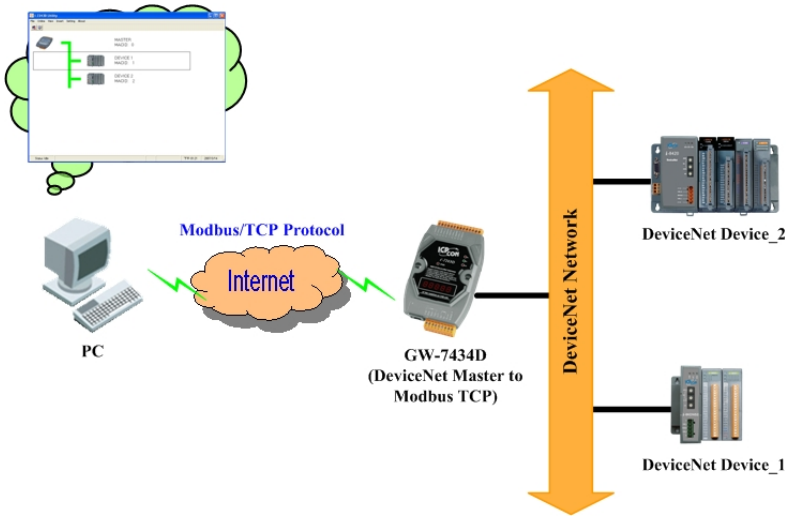
Word Address 1800h ~ 1BF0h: Specific data format string for input command.

For saving specific data format string that are sent by Utility tool to configure input device data address. The type of the specific data format is described in the following table. Each specific data format contains 32 bytes. And the maximum number of the command string is 100.

Word Address (3x)	Description	Data Length	Data
1800h(High)	Is the slave valid	1 Bytes	0x00 : Not valid 0x01 : Explicit valid 0x03 : I/O valid
1800h(Low)	Is the “start address” valid	1 Bytes	0x00 : Not valid 0x01 : valid
1801h(High)	Slave MAC ID = 0x00	1 Byte	0x00
1801h(Low)	Slave ID 0, Connection Type	1 Byte	0x00: Explicit 0x01: Poll 0x02: Bit-Strobe 0x03: COS 0x04: Cyclic
1802h	Slave ID 0, Start address of Input Data Area	2 Byte	0x0000 ~ 0x01FF
1803h	Slave ID 0, input data length	2 Byte	0x0000 ~ 0x01FF
1804h	Slave ID 0, Communication Status	2 Byte	Refer to Appendix A.
1805h	Slave ID 0, Expected Packet Rate	2 Byte	0x0001 ~ 0xFFFF
1806h ~ 180Fh	Reserve	22 Byte	Reserve
1810h ~ 181Fh	Slave ID 1 data format	32 Byte	Slave ID 1 data
1820h ~ 182Fh	Slave ID 2 data format	32 Byte	Slave ID 2 data
:	:	:	
1BF0h~1BFFh	Slave ID 63 data format	32 Byte	Slave ID 63 data

5. Configuration

5.1 GW-7434D Configuration Tool (GW-7434D Utility)



The GW-7434D Utility tool can be used to configure the operation condition of the GW-7434D module. Also it can be used to monitor/configure, add/remove the devices on the DeviceNet network. To start the “GW-7434D Utility”, please install the GW-7434D Utility setup file and run the GW-7434D.exe file. The screenshot of the startup screen for this Utility is given in the below figure. Connect the GW-7434D’s Ethernet port with the PC’s Ethernet port via a standard CAT 3 or CAT 5 network cable. It will auto-negotiate the network speed and connection. Then the user can online monitor and configure the connection status of the GW-7434D. For further information related to this, please refer to section 2 of this manual on how to make a hardware connection.

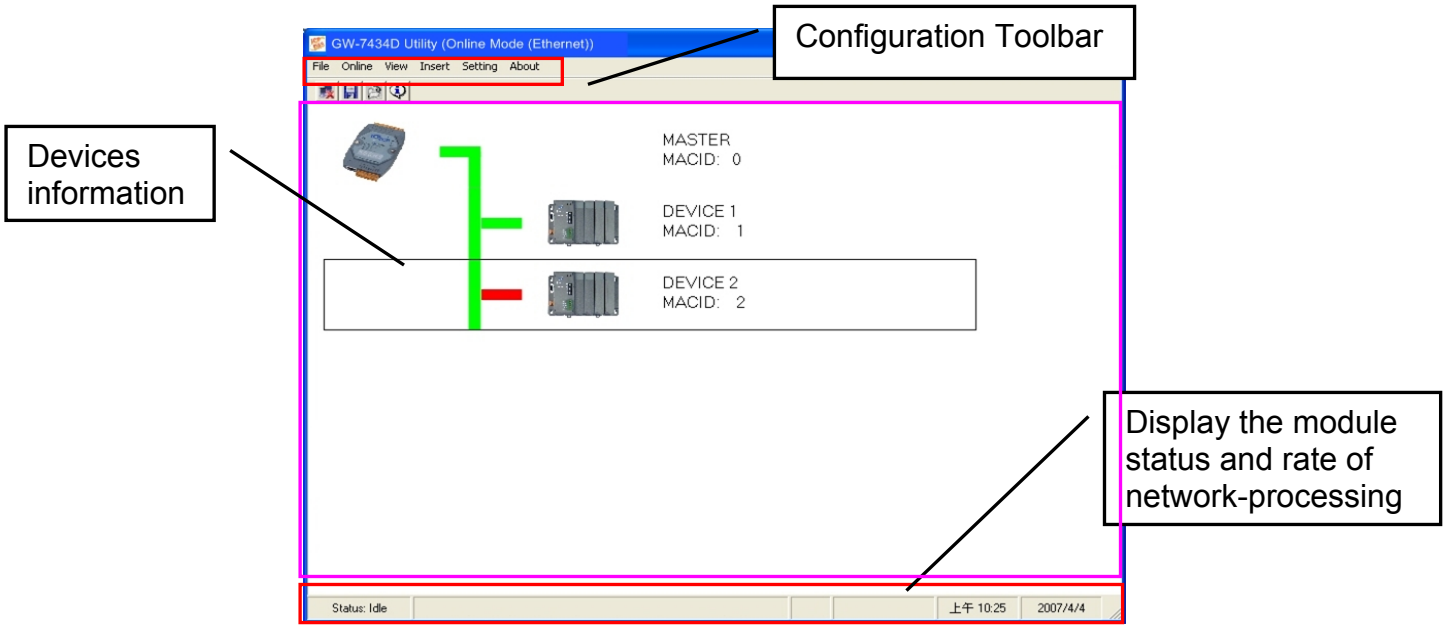


Figure 5-1: GW-7434D Utility

5.1.1 Install & uninstall the GW-7434D Utility

Install GW-7434D Utility

Step1: Download the GW-7434D Utility setup file from the web site http://www.icpdas.com/products/Remote_IO/can_bus/GW-7434D.htm or the CD-ROM disk following the path of “CAN-CD:\\DeviceNet\Gateway \GW-7434D\Utility

Step 2: Execute the setup.exe file to install GW-7434D Utility.

Step 3: A “Welcome” window pops up to prompt user to begin the installation. See figure 5-2.

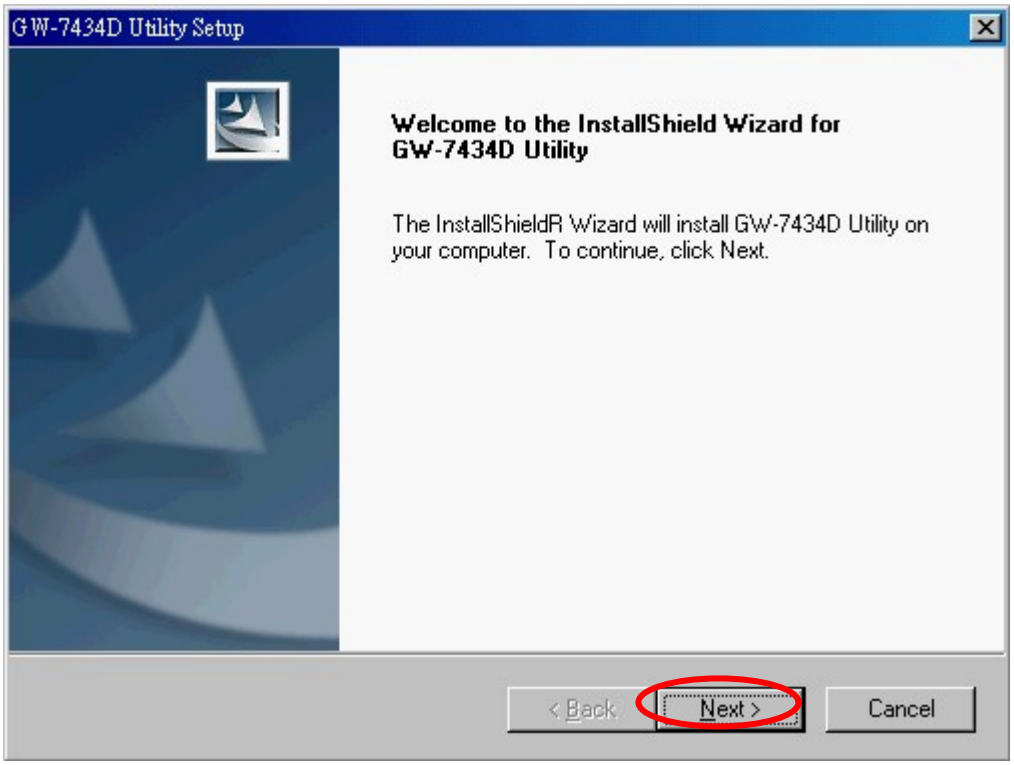


Figure 5-2: Welcome dialog

Step 4: Click the “Next” button and A “Choose Destination Location” window will pop up for deciding the installation path.

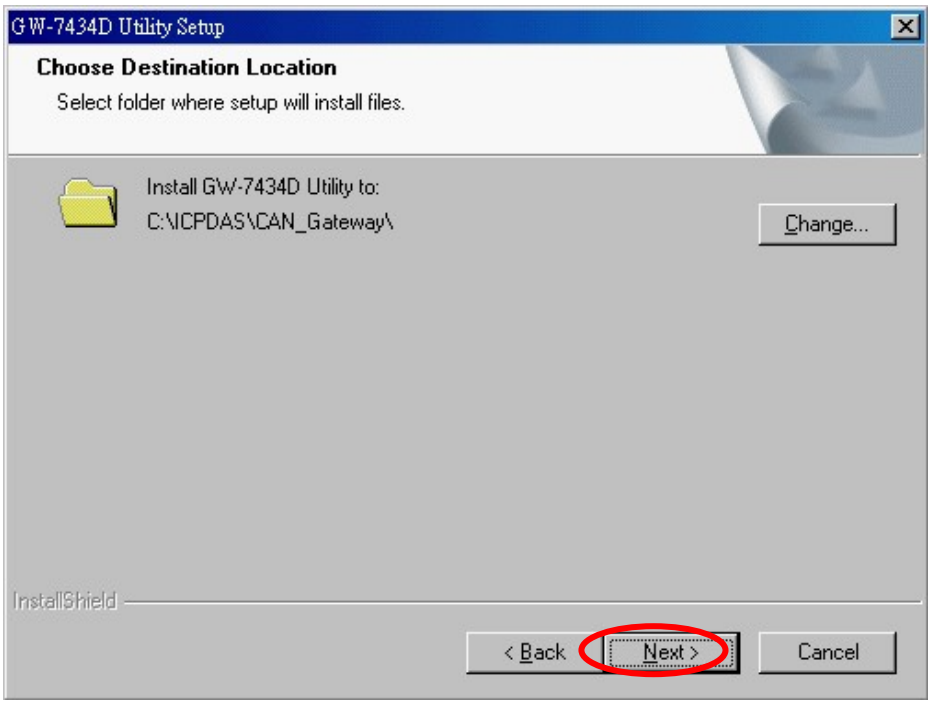


Figure 5-3: “Choose Destination Location” dialog

Step 5: Click “Next” button and a “Ready to Install the Program” window will pop up to prompt user that the wizard is ready to begin the installation See figure 5-4.

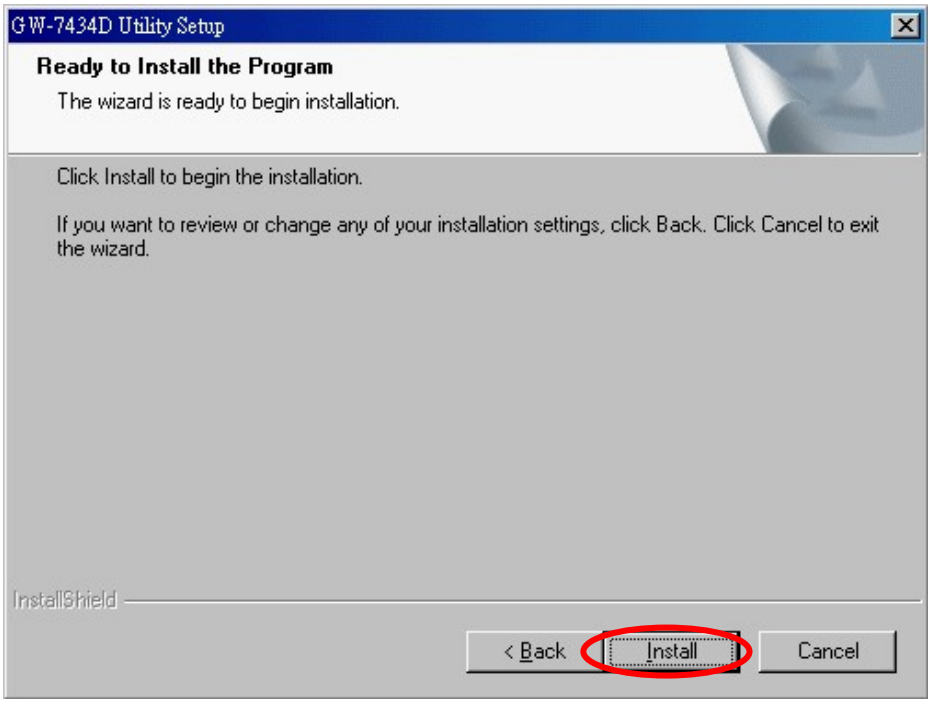


Figure 5-4: “Ready to Install the Program” dialog

Step 6: Click “Install” button and start to install the GW-7434D Utility to the system. After finishing the process, a “Complete” window will pop up to prompt users that the successful completion of the installation. And click “Finish” button to exit. See figure 5-5.

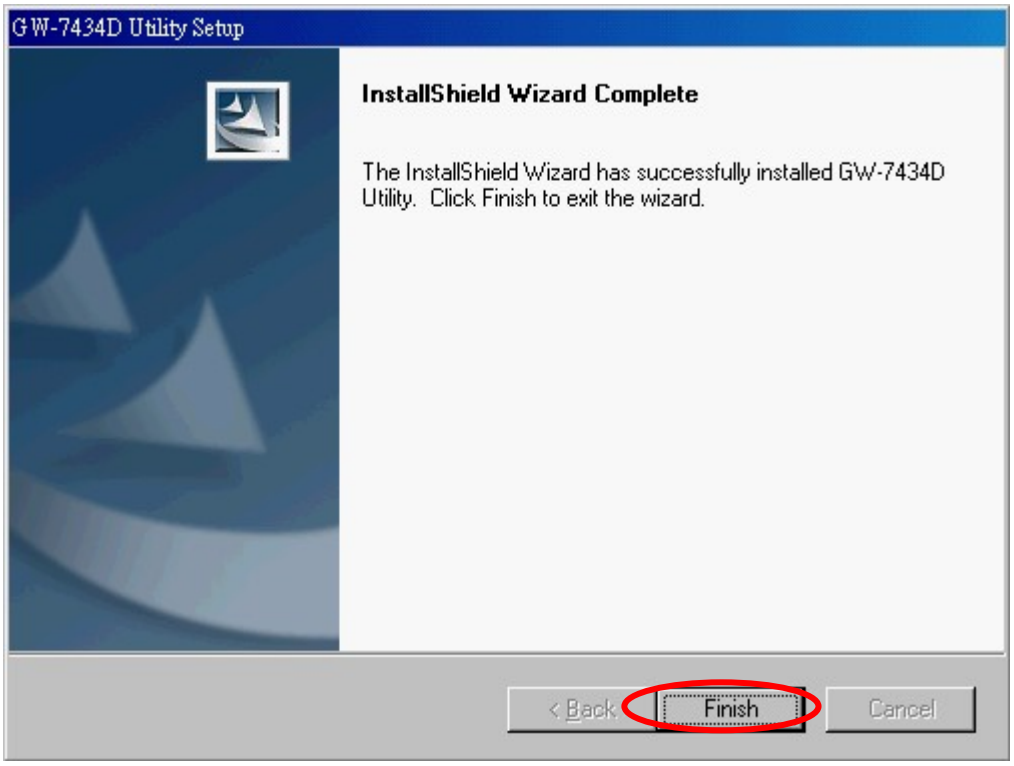


Figure 5-5: “Successful Completion of the Installation” dialog

Step 7: After finishing the installation of the GW-7434D Utility, users can find it as shown in figure 5-6.

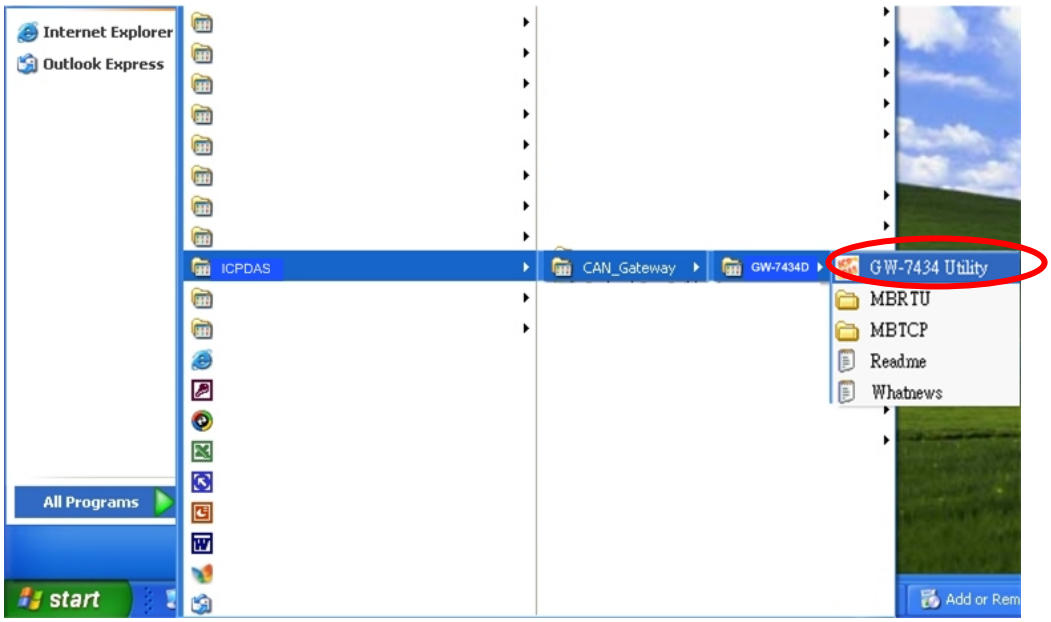


Figure 5-6: You can find “GW-7434D Utility“ at the “Start” in the task bar

Uninstall GW-7434D Utility

You can uninstall GW-7434D Utility software by the following means described below:

Step 1: Click “Start” in the task bar, then click the Control Panel as shown in figure 5-7.

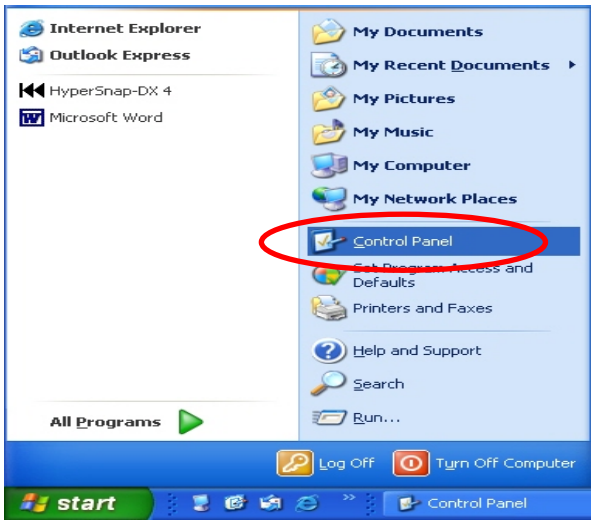


Figure 5-7: Select settings

Step 2: Double click the “Add or Remove Programs” button icon to open the dialog. See figure 5-8.

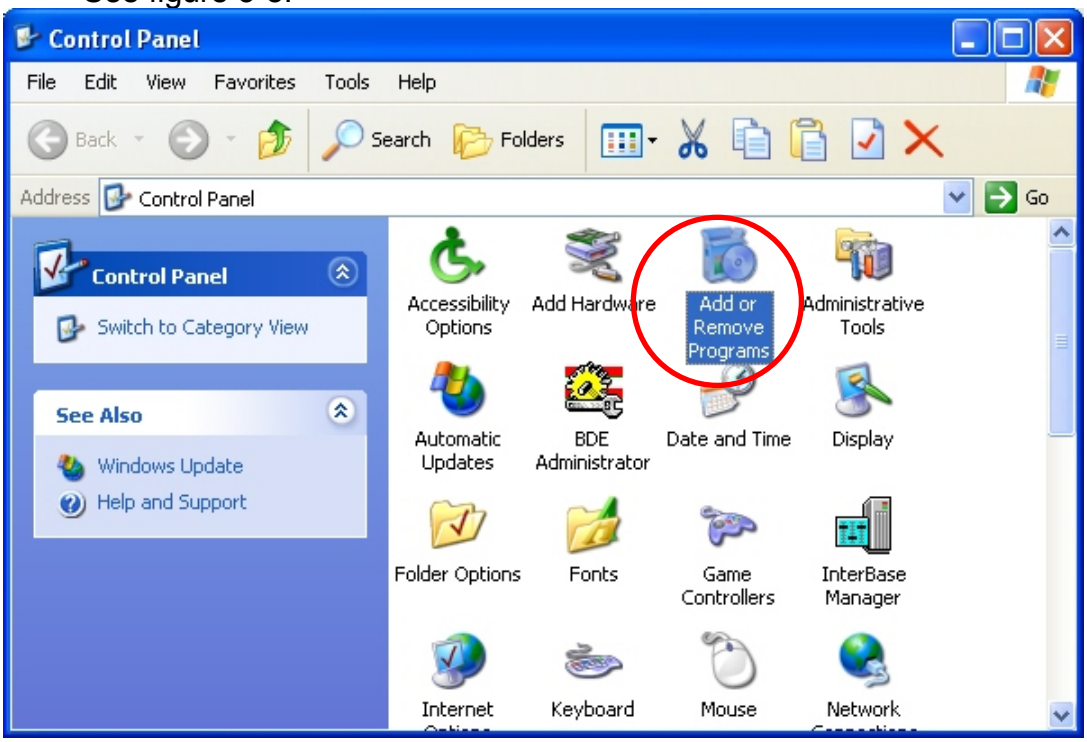


Figure 5-8: “Add or Remove Programs”

Step 3: Find out the GW-7434D Utility, and click the Change/Remove button. See figure 5-9.

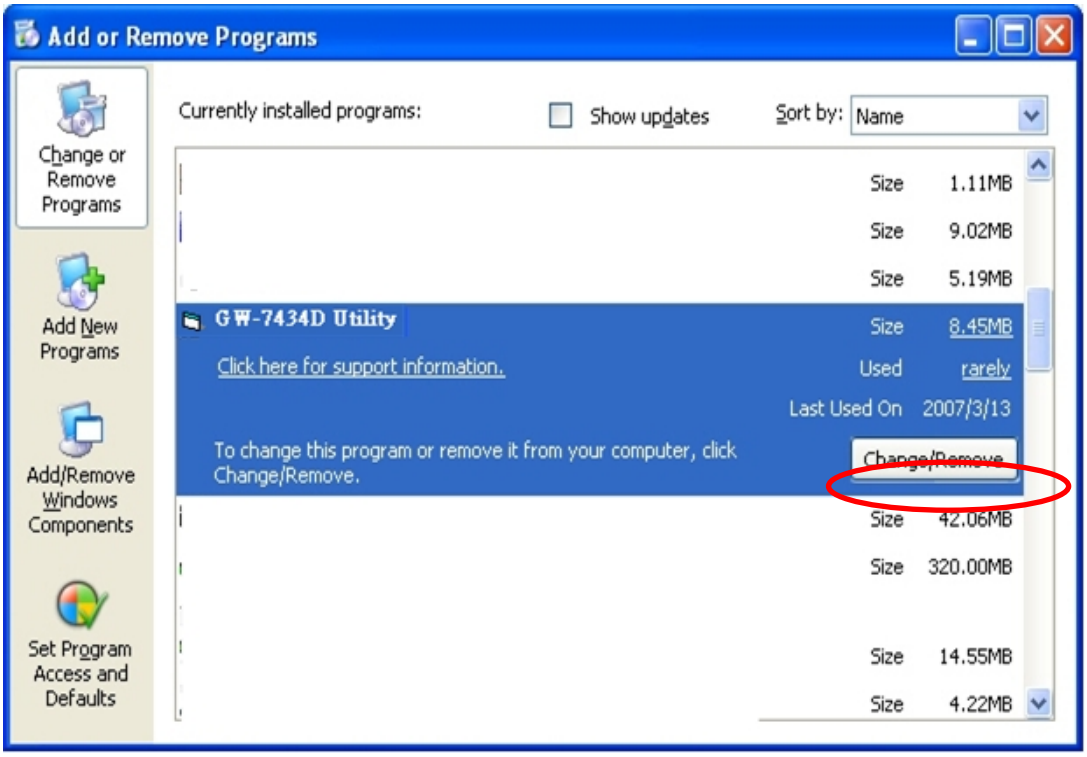


Figure 5-9: Click “Change/Remove” button

Step 4: Select the “Remove” option button, and press the “Next” button to remove GW-7434D Utility. See figure 5-10.

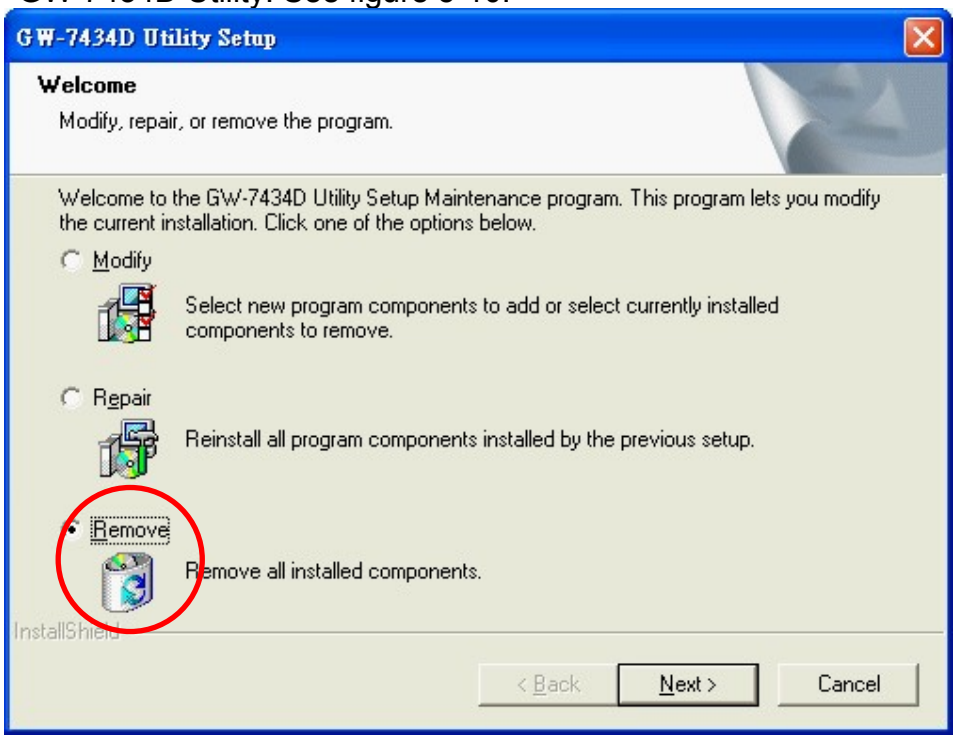


Figure 5-10: “Modify, repair, or remove the program” dialog

Step 5: Click the button “Yes” to remove the software as shown in figure 5-11.

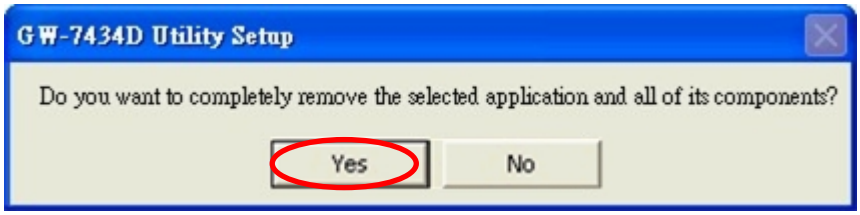


Figure 5-11: Click the button “Yes” to remove the software

Step 6: Finally, click the “Finish” button to finish the uninstall process.

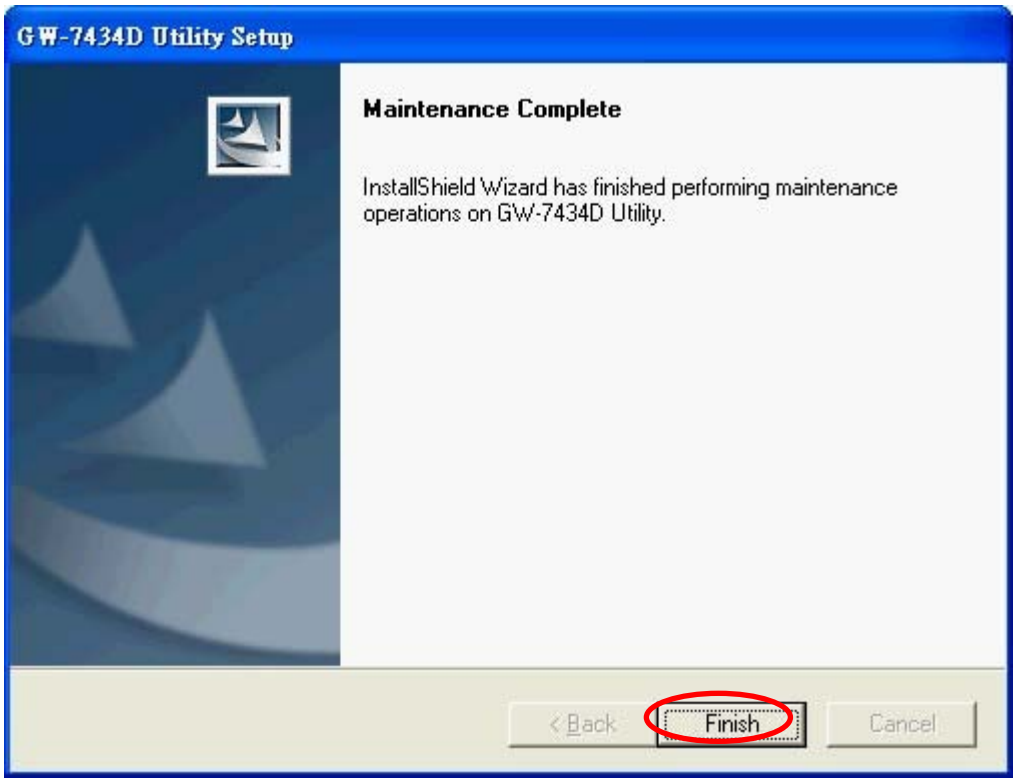
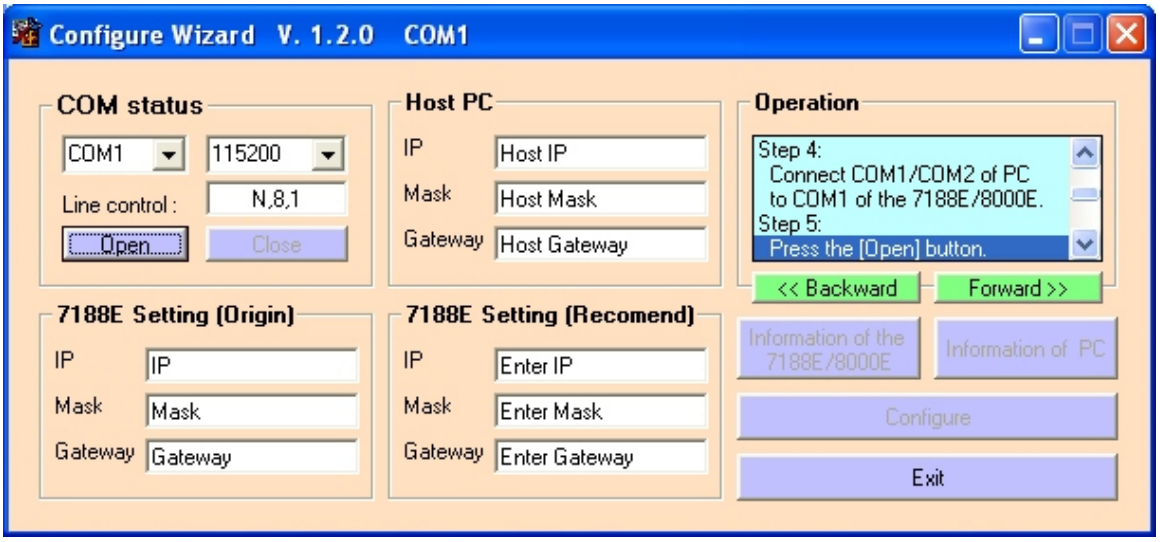


Figure 5-12: “Maintenance Complete” dialog.

5.1.2 How to set/connect with the module

When you first connect/install a GW-7434D, you need to adjust each setting to suit your requirements. You only need to do this once. The following procedure will guide you on how to set and connect the GW-7434D with Utility tools. The configuration steps are depicted as below:

1. Connect the power source (the 10~30 DC volts) into the GW-7434D module.
2. Configure the network settings (IP, Mask, Gateway) for the GW-7434D controllers.



To Use the Configuration Wizard, you must first install PCDiag.
(8000CD:\Napdos\7188e\TCP\PCDiag\Setup\Setup.exe)

3. After using configure wizard to set the network parameters of GW-7434D, please power-off and power-on the power source of it again.
4. Then the GW-7434D module's Power LED will flash approximately once per 0.5 second. And the 5-digits 7-segment LED will scroll to display some messages, please refer to section 2.3 to check what the status it is.
5. The user must run the GW-7434D's Utility software after they have made a wire connection between the PC and the GW-7434D via the network cable
6. Select the "Online" on the GW-7434D Utility menu bar. Then click the "Connect" item. The "Connect…" window will be pop-up. Key-in the IP of the GW-7434D and press the "Connect" button in order to connect with it. As shown in the following figure.

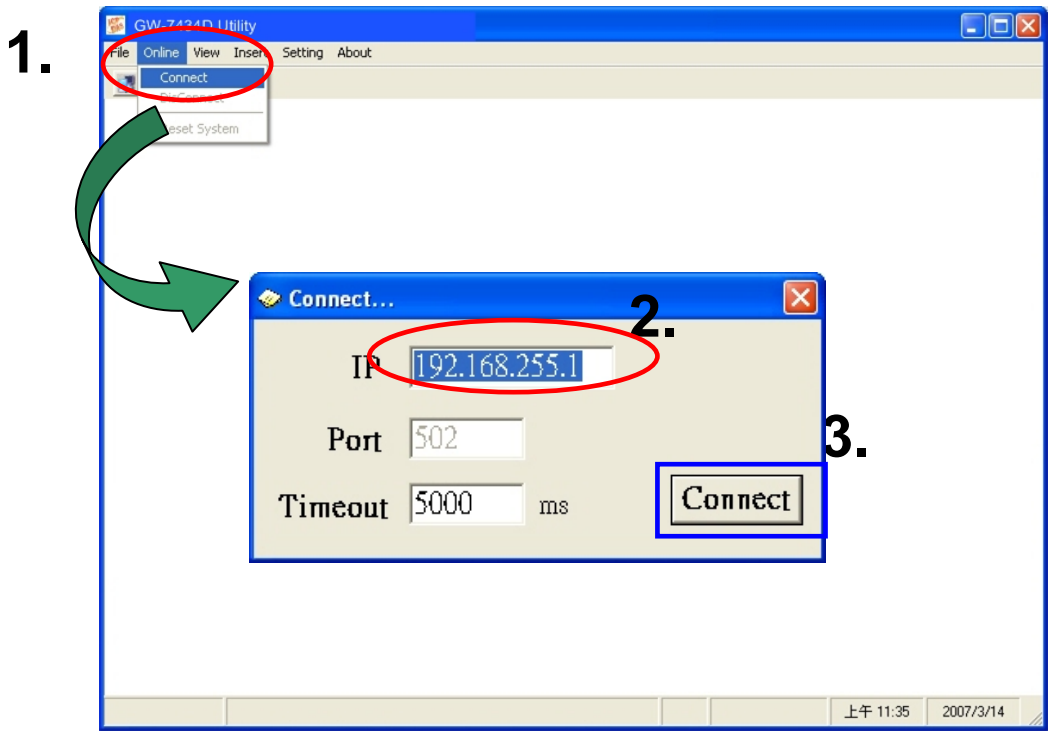


Figure 5-13: Connection setting of GW-7434D

7. If the GW-7434D is online and work normally, the Utility tool will display the connection state and devices at the main frame. Then users configure and set the GW-7434D and its DeviceNet slaves devices at each configuration table. The main frame is shown in the following figure.

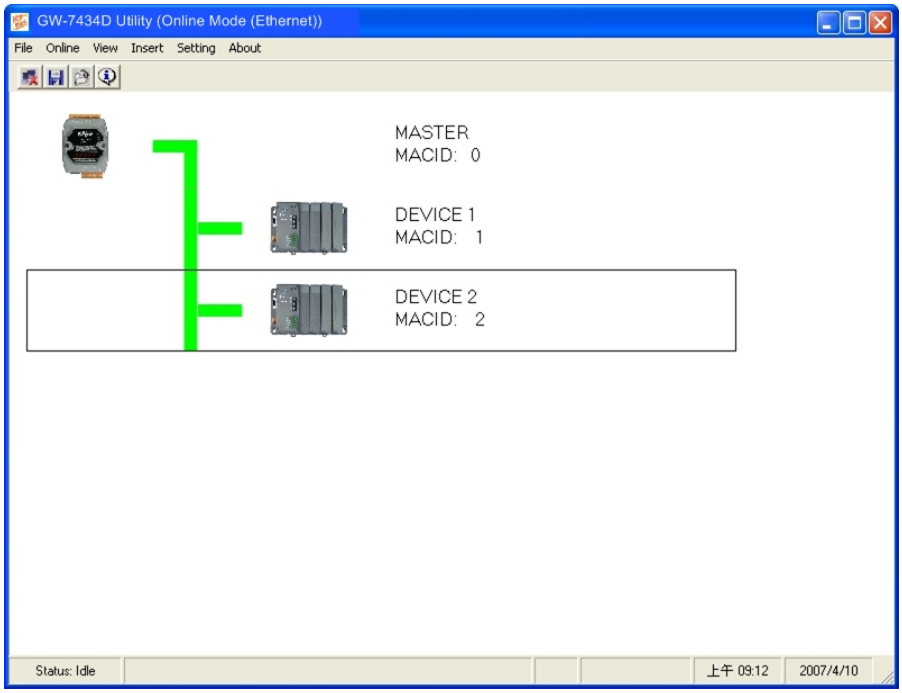


Figure 5-14: Connect to the configuration mode of the GW-7434D

5.1.3 How to configure the module's DeviceNet MACID and Baud rate

Users can configure the GW-7434D's DeviceNet MACID and Baud rate at the "Bus Parameters" Table. The setting of these two parameters will take affect after system is reset. Please do the following steps to change these two parameters.

1 Connect the GW-7434D module with Utility tool, described as section 3.1.2.
2 Then select the master frame, GW-7434D.
3 Select the "Setting" item on the menu bar, or click the mouse right-button on the master frame, then click the "Bus Parameters…" item. Then the "Bus Parameter" window will be pop-up. As shown in figure 5-15.

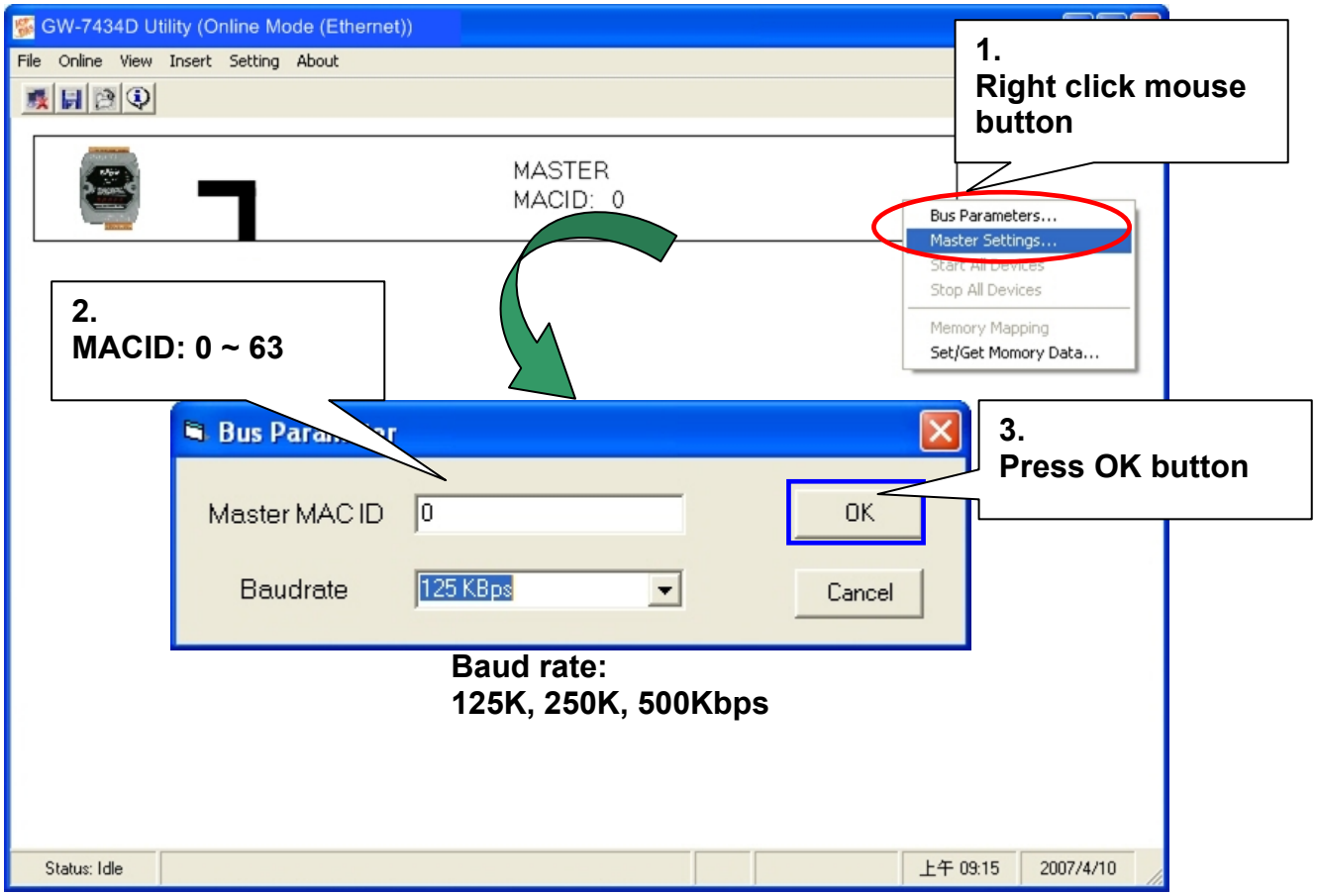


Figure 5-15: Setting of DeviceNet MACID and Baud rate

4 All setting will become affect after reset the system of the GW-7434D.

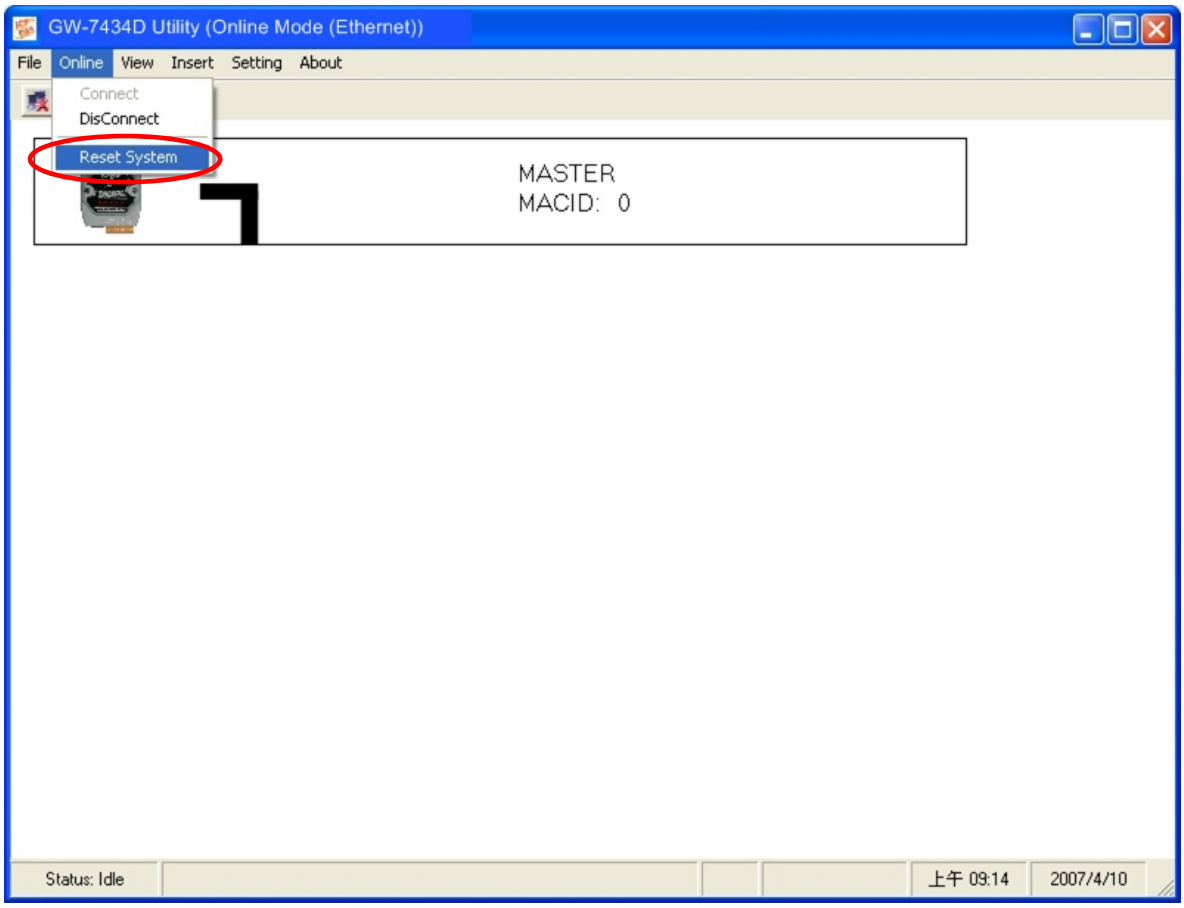


Figure 5-16: Reset the system of GW-7434D

5.1.4 How to configure the module's application mode

The GW-7434D can act as Modbus/TCP to multI-Modbus/RTU converter or protocol converter with VxComm technology. After connect GW-7434D with Utility tool, users can change these application mode at the "MBTCP configuration Table". Please do the following steps to change the application mode of the GW-7434D.

1. Connect the GW-7434D module with Utility tool, described as section 5.1.2.
2. Then select the master frame, GW-7434D.
3. Select the "Setting" item on the menu bar, or click the mouse right-button on the master frame, then click the "Master Settings…" item. Then the "MBTCP Configuration" window will be pop-up. As shown in figure 5-17.

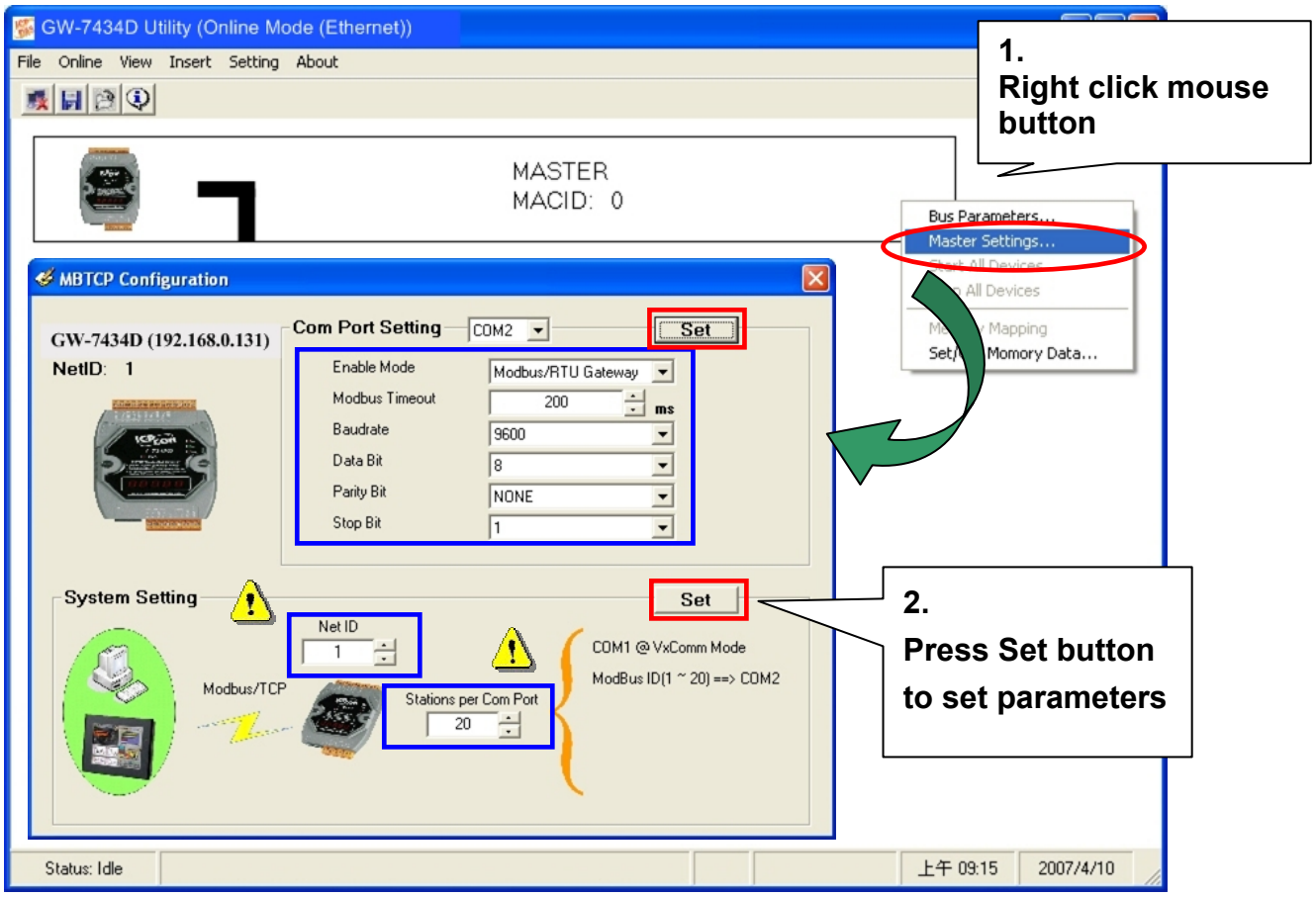


Figure 5-17: GW-7434D Configuration

4. Adjust system settings:

The GW-7434D can be a single-Modbus/TCP to multI-Modbus/RTU converter. The Modbus station number is a very important parameter. It is used to recognize different Modbus stations. But the GW-7434D does not have any hardware design such as dip-switch or jump allowing you to set the Modbus station number (or called NetID). You must use the Utility to set some parameters regarding to Modbus station number. System settings include follows:

Net ID:

If the Modbus station number in a Modbus/TCP request (from PC or HMI) matches the Net ID, the request is passed to the Modbus kernel. Then Modbus kernel will then respond and send the internal registers (**DeviceNet devices parameters**) to the Modbus/TCP client (PC or HMI).

The content of all other registers are listed on the following section.

Station per Com Port:

This value is used to decide how many Modbus/RTU stations can one GW-7434D COM port control. That also means the value can decide which COM the Modbus/TCP request will be passed to.

When click the "Set" button, the Utility shows the gateway mapping.

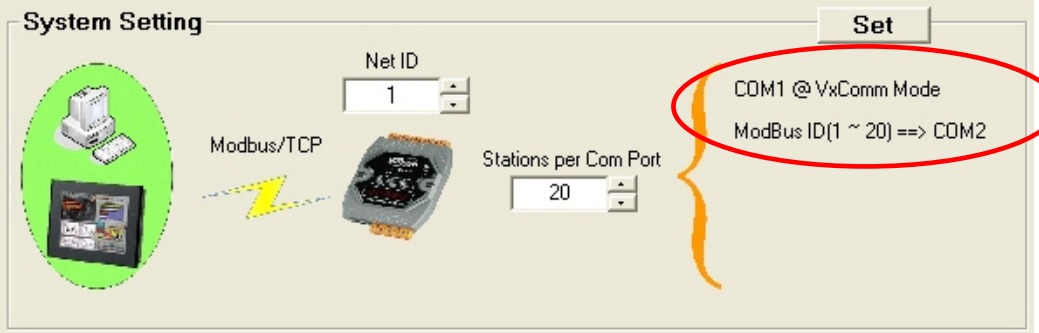


Figure 5-18: MBTCP System Setting

5. Adjust the COM port settings:

Enable Mode (Default = Modbus/RTU gateway):

GW-7434D COM ports can be configured as follows:

1. **VxComm (Virtual COM):**
   Enable VxComm. Remote application program can use Virtual COM (need to install the VxComm Driver) or connect to TCP/IP port 10000 + n to access the GW-7434D COM port via Ethernet. At VxComm mode, the COM port can link to any serial device.
2. **Modbus/RTU Gateway:**
   Enable protocol gateway function to convert Modbus/TCP to Modbus/RTU. At "Modbus/RTU gateway" mode, the COM port can only link to Modbus/RTU slave devices.
3. **No Use:**
   Disable communication of the COM port.
4. **UpLink:**
   Enable the COM port to be a Modbus/RTU slave port.
5. **Debug port**
   The Modbus kernel prints out some messages while communicating with Modbus clients or masters.
   The messages includes
   (0) receives Modbus request
   response to Modbus clients or masters
   (1) by passes Modbus request to COM port
   (2) Send Modbus request to COM port
   (3) Check Modbus response from COM port
   (4) Send Modbus response to Modbus client or masters

Data format of Modbus/RTU

There are several kinds of data format used in the Modbus/RTU protocol, you must change it to suit the Modbus/RTU devices that connect to the COM port.

8 data bits, none parity, 1 stop bits (Default)
8 data bits, none parity, 2 stop bits
8 data bits, odd parity, 1 stop bits
8 data bits, even parity, 1 stop bits

Note: When a GW-7434D receives a Modbus/TCP request that not to its internal registers, it first decides which COM port to send the request. Modbus/RTU must be enabled for this COM port, otherwise the request will be discarded.

Timeout (Default = 200 ms):

After finishing data transmission, the system begins to count time, if timeout expires, the system stop receiving responses.

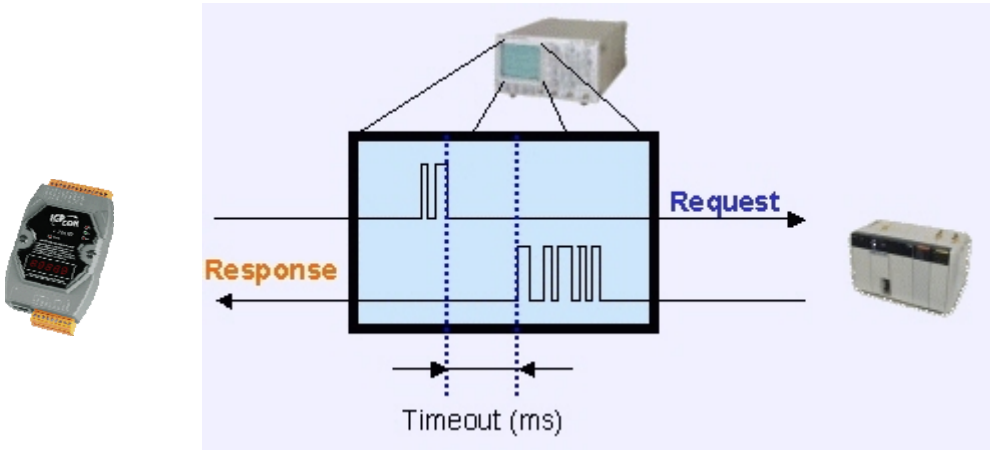


Figure 5-19: Request/response Timeout

5.1.5 How to add/remove/configure DeviceNet devices

On the main frame of GW-7434D Utility tool, users can add/remove/configure the DeviceNet devices by select the necessary items.

Add a DeviceNet device:

Click left of the mouse button on the black space of main frame. Select the "Insert" item on the menu bar, or click right of the mouse button on the black space of main frame, then click the "Device" item. Then the "Insert Device" window will be pop-up. As shown in figure 5-20.

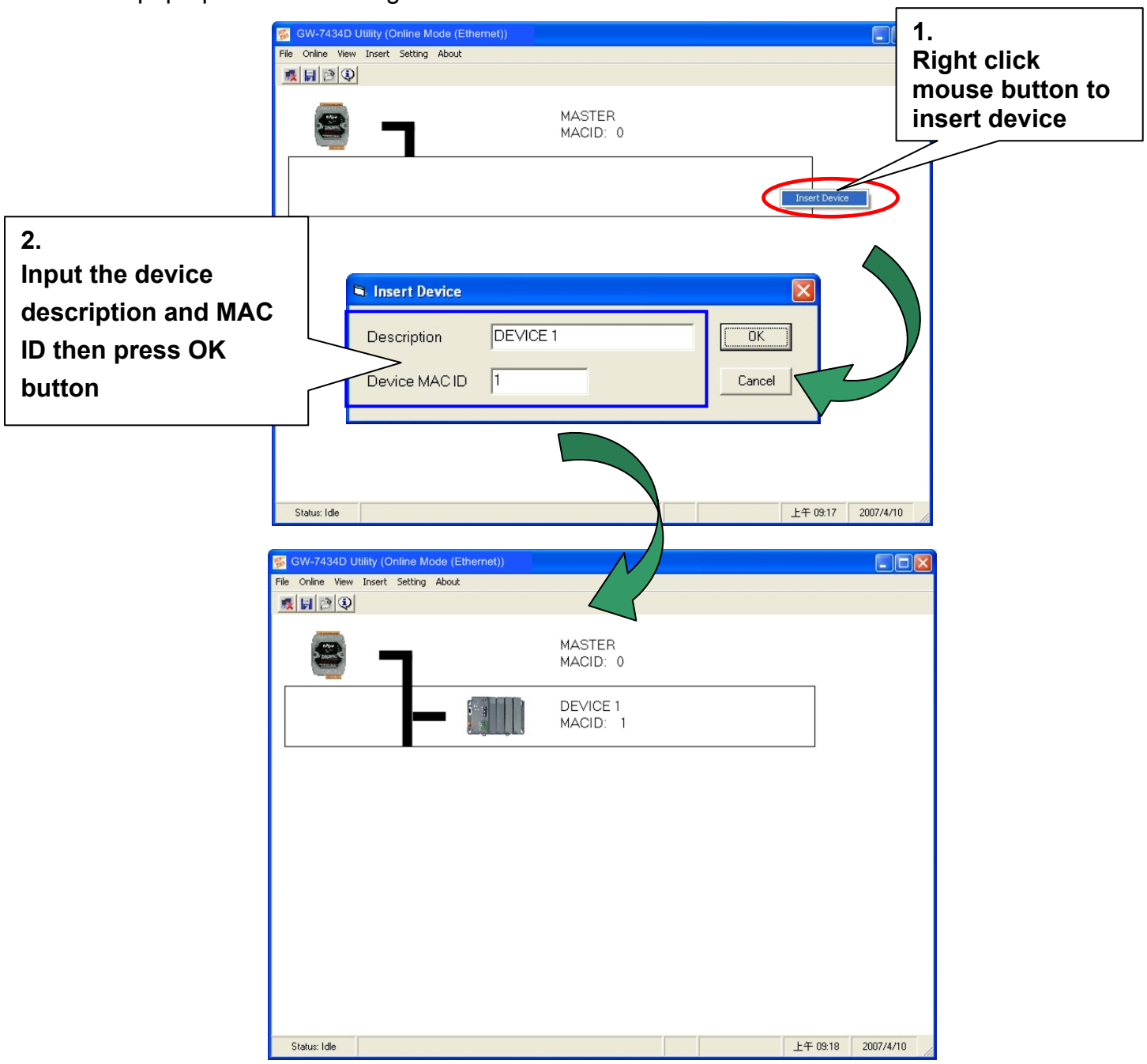


Figure 5-20: Add a DeviceNet device into GW-7434D

Remove a device:

Click left of the mouse button on device that you want to remove. Click right of the mouse button on the device, then click the “Delete” item. Then the selected device will be removed from the GW-7434D. As shown in figure 5-21.

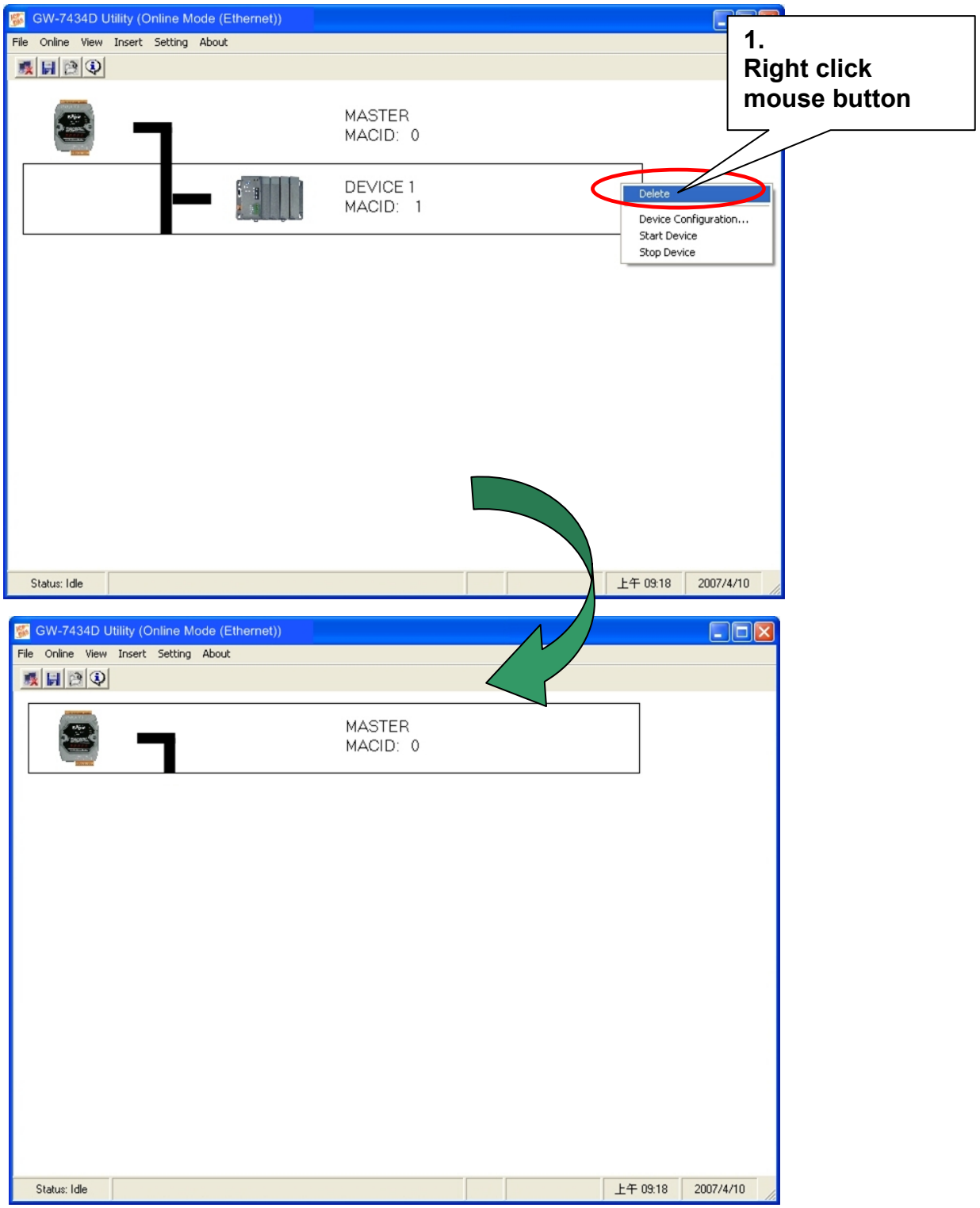


Figure 5-21: Remove a DeviceNet device from GW-7434D

Configure all devices:

1. Click left of the mouse button on device that you want to configure. Select the "Setting" item on the menu bar, or click right of the mouse button on the device, then click the "Device Configuration…" item. Then the "Device Configuration" window will be pop-up. As shown in figure 5-22.

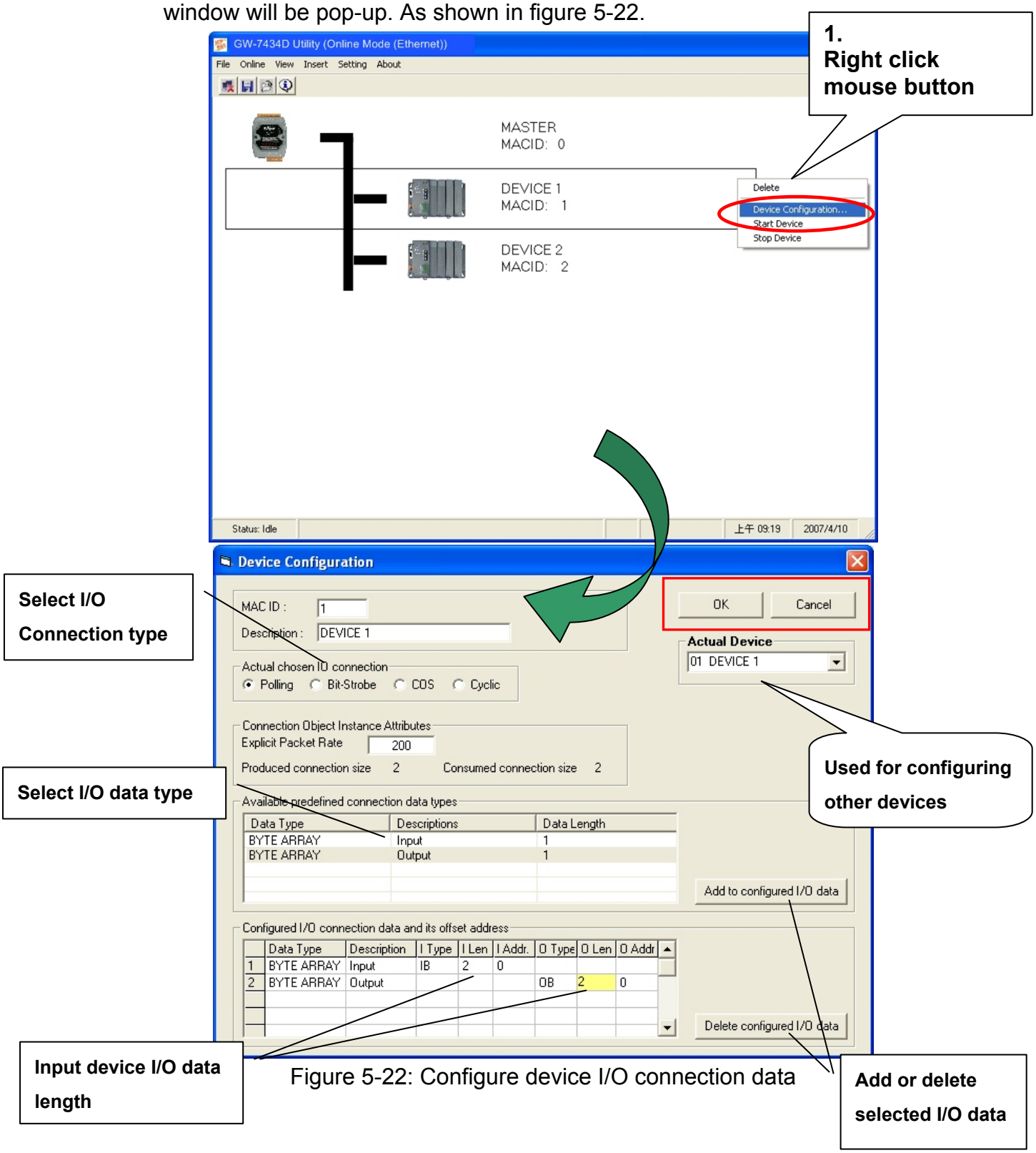


Figure 5-22: Configure device I/O connection data

2. After pressing “OK” button to finish the setting of all actual devices, Utility tool will save all setting into GW-7434D’s EEPROM and GW-7434D start to communicate with these devices automatically. Then the GW-7434D Utility tool will display each configured device’s connection status on the main frame and update the device connection information per 0.8 second. As shown in figure 5-23.

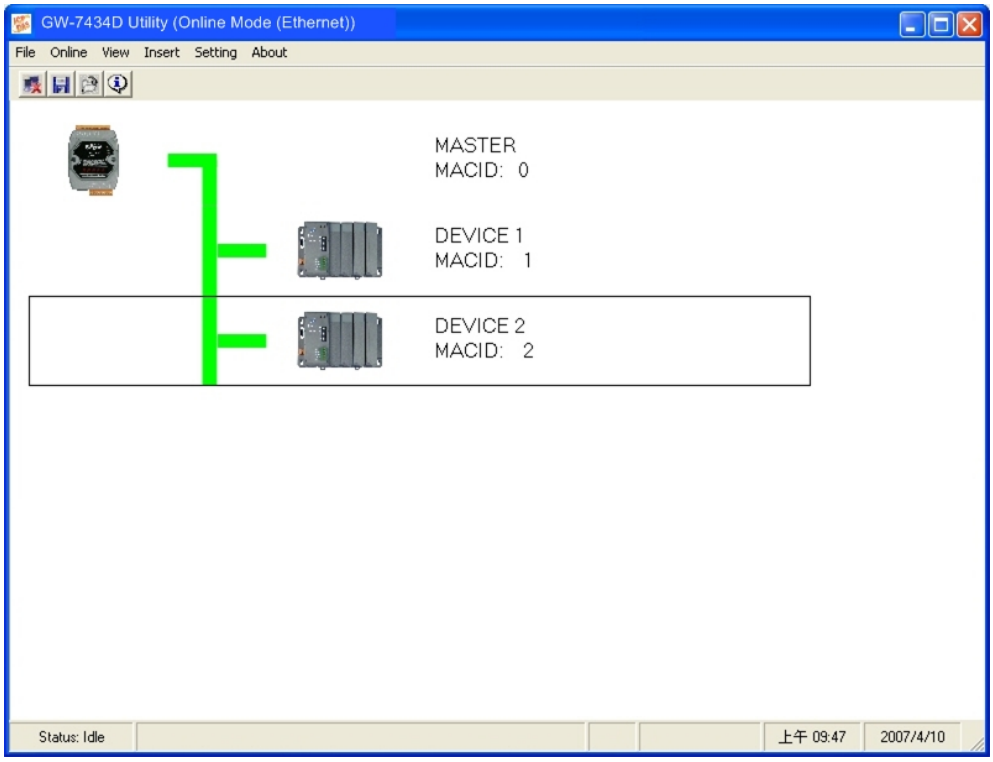


Figure 5-23: Start to monitor the connection status of each device

5.1.6 How to start/stop to communicate with DeviceNet devices

After setting all the DeviceNet devices on the “Device Configuration” window, the GW-7434D will start to communicate with the devices. If there is no error occurs on these devices, the color of the line picture between GW-7434D and DeviceNet devices will be green. Otherwise, it will become red color. If users stop the communication of some other devices, the color of the line picture will become black.

Users can follow the steps to start or stop the communication of each or every DeviceNet devices.

1. Click left of the mouse button on device that you want to start or stop. Select the “Setting” item on the menu bar, or click right of the mouse button on the device, then click the “Start Device” item or “Stop Device”. Then the GW-7434D will start/stop to communicate with the DeviceNet device. As shown in figure 5-24.

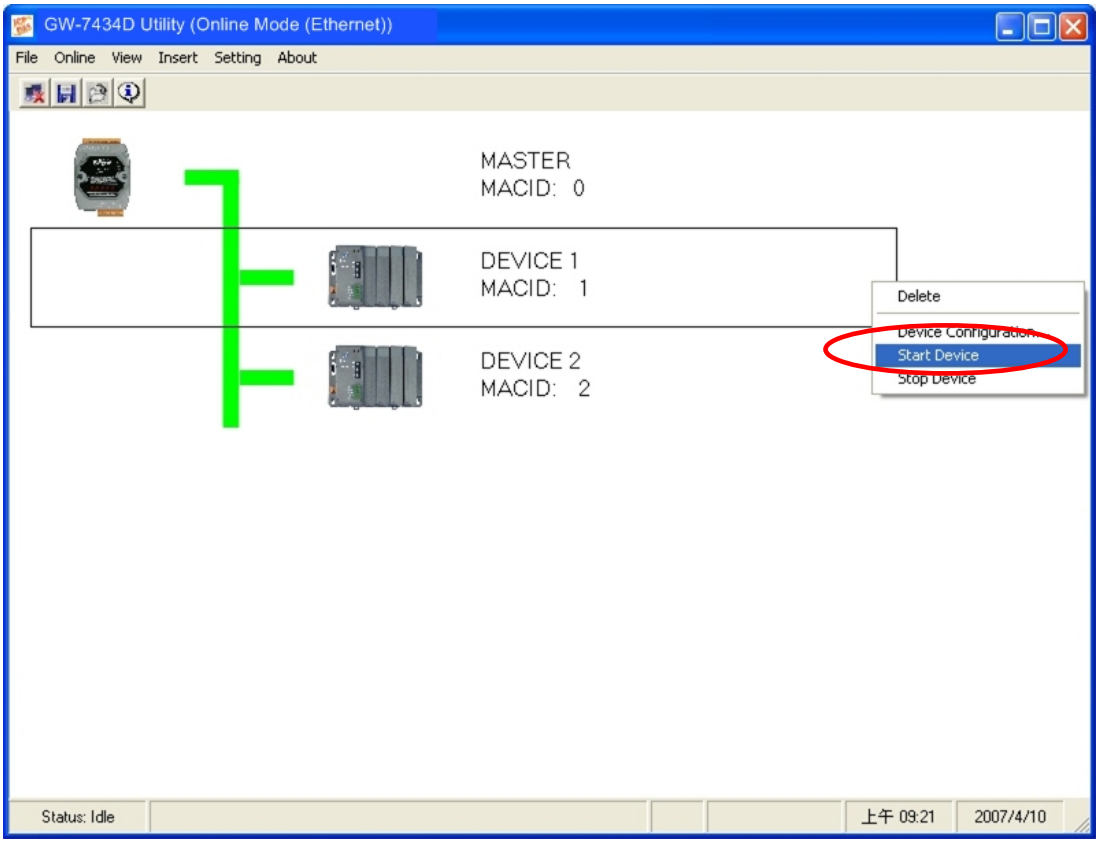


Figure 5-24: Start/stop communicating with DeviceNet devices

2. By Clicking left of the mouse button on master frame, GW-7434D, you can start or stop communicating with all DeviceNet devices. Select the “Setting” item on the menu bar, or click right of the mouse button on the master frame, then click the “Start All Devices” item or “Stop All Device”. Then the GW-7434D will start/stop communicating with all DeviceNet devices. As shown in figure 5-25.

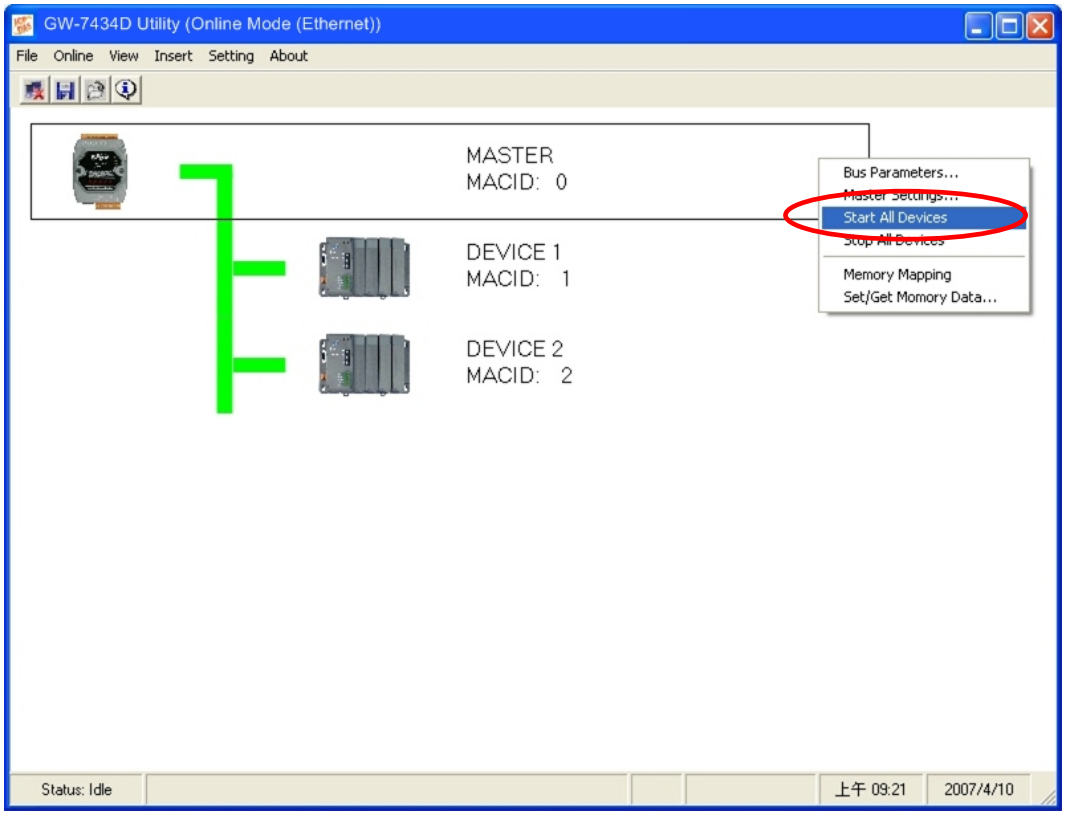


Figure 5-25: Start/stop communicating with all DeviceNet devices

5.1.7 How to get the configuration/status of each DeviceNet device

After configuring the DeviceNet device, the Utility tool will let the GW-7434D start to communicate with these devices. Every 0.6 second, the Utility tool will use Modbus/TCP commands to communicate with GW-7434D and get these DeviceNet devices' information. Users can get the configuration and status of these devices by looking the "Device Table", "Address Table" and "Status Table" over. As shown in figure 5-26, 5-27, 5-28.

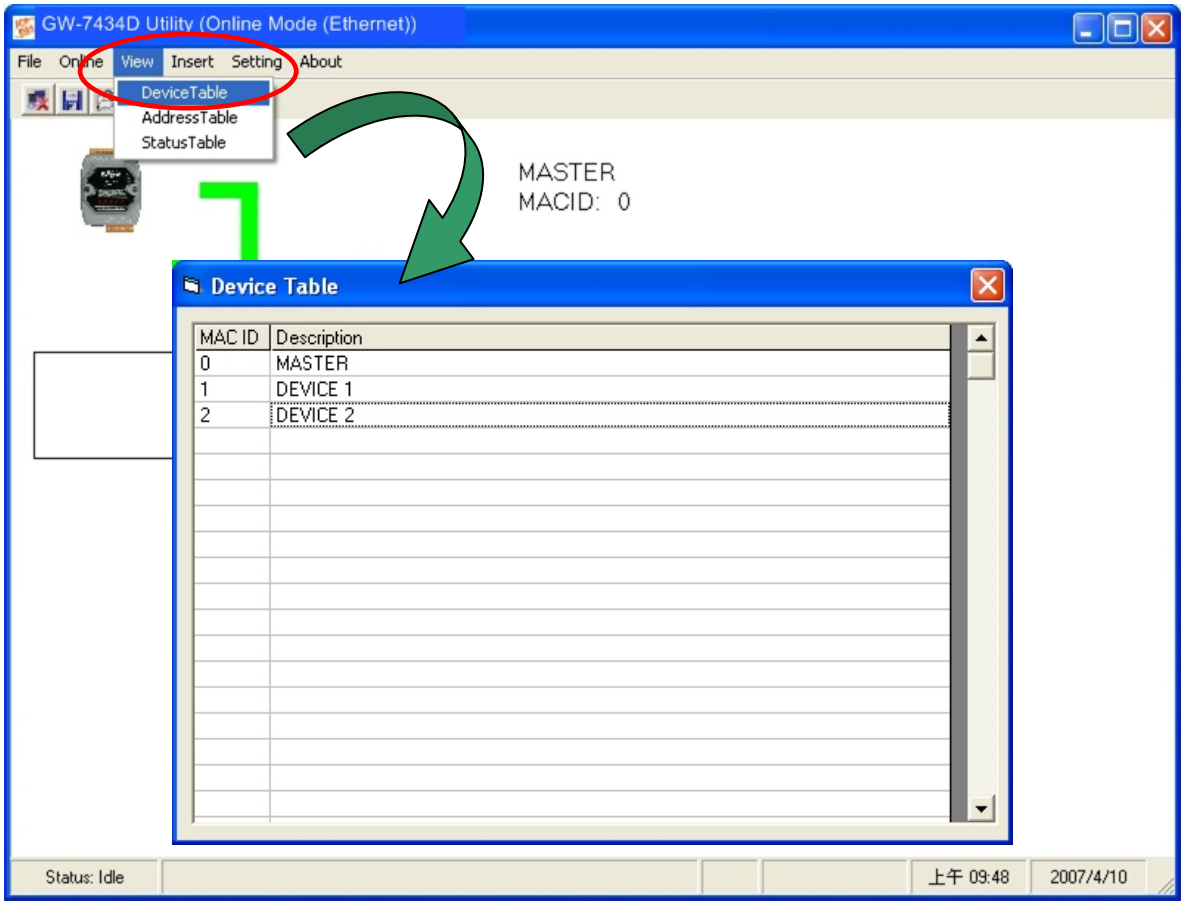


Figure 5-26: Device Table

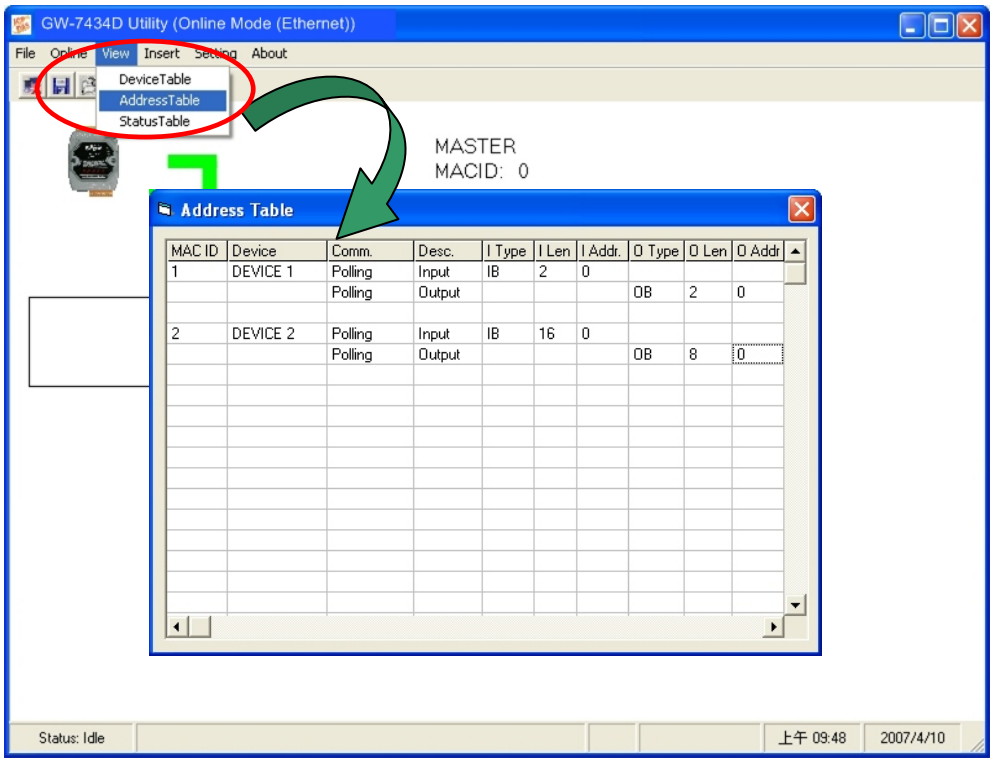


Figure 5-27: Address Table

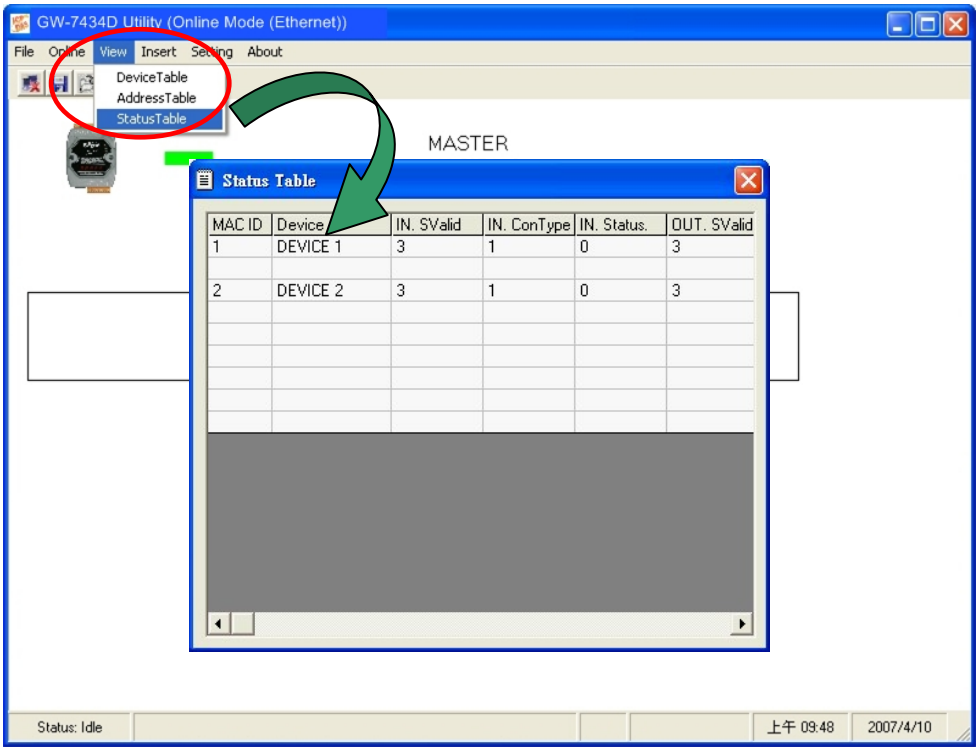


Figure 5-28: Status Table

Please refer to the appendix A to get the meaning of each "Status".

5.1.8 How to get the I/O data of each DeviceNet device

The GW-7434D supports an “Internal Memory” for saving I/O data. The input and output data areas are set to a maximum size of 512 bytes each. After using Utility tool to set the memory mapping of the I/O devices, users can get/set the I/O data according to the mapping address of each I/O data. When accessing these areas, with Modbus commands, the addressing is according to the following tables.

Users can use Modbus Function code 4, read input registers, to get the input data values from GW-7434D’s input data area.

Word Address (3x)	Byte Address	
0000h	0000h	0001h
0001h	0002h	0003h
0002h	0004h	0005h
: :	: :	: :
00FFh	01FEh	01FFh

Table 5-1: Input Addressing

Users can use Modbus Function code 16, force multiple registers, to set the output data values into GW-7434D’s output data area.

Word Address (4x)	Byte Address	
0000h	0000h	0001h
0001h	0002h	0003h
0002h	0004h	0005h
: :	: :	: :
00FFh	01FEh	01FFh

Table 5-2: Output Addressing

Note:

The base address of the input and output address are starting from 0000h

Setting the Input/output data area of the GW-7434D's internal memory

After configuring the DeviceNet devices, user can set the input/output data area of GW-7434D's internal memory. The steps are shown in the following figure.

1. Select the master frame, GW-7434D
2. Select the "Setting" item on the menu bar, or click the mouse right-button on the master frame, then click the "Memory mapping" item. Then the "Internal Memory Configuration" window will be pop-up. As shown in figure 5-29.

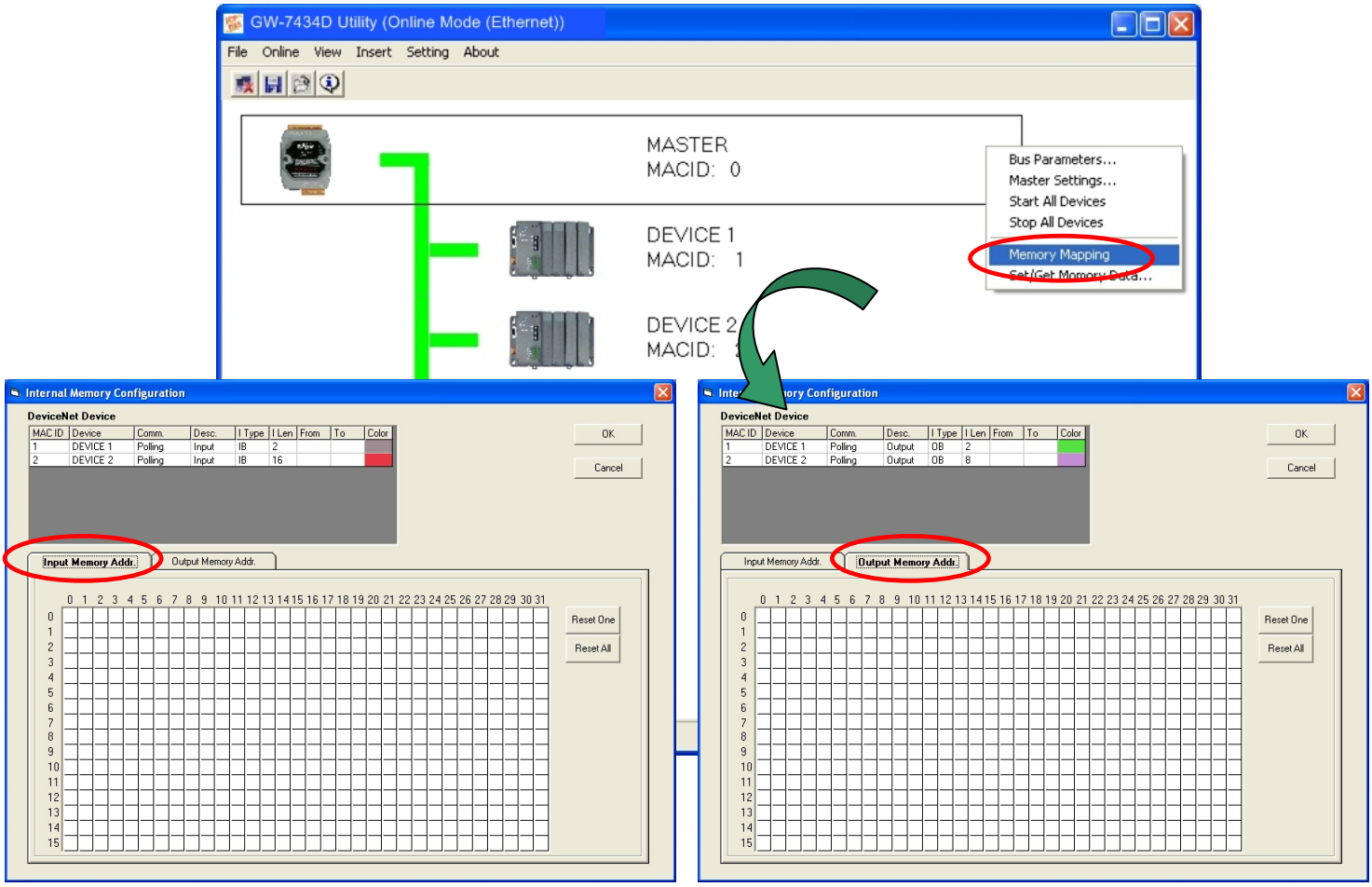


Figure 5-29: Internal Memory Configuration

3. Then users can configure the DeviceNet input devices into Input memory address or the DeviceNet output devices into output memory address. As shown in figure 5-30, 5-31.

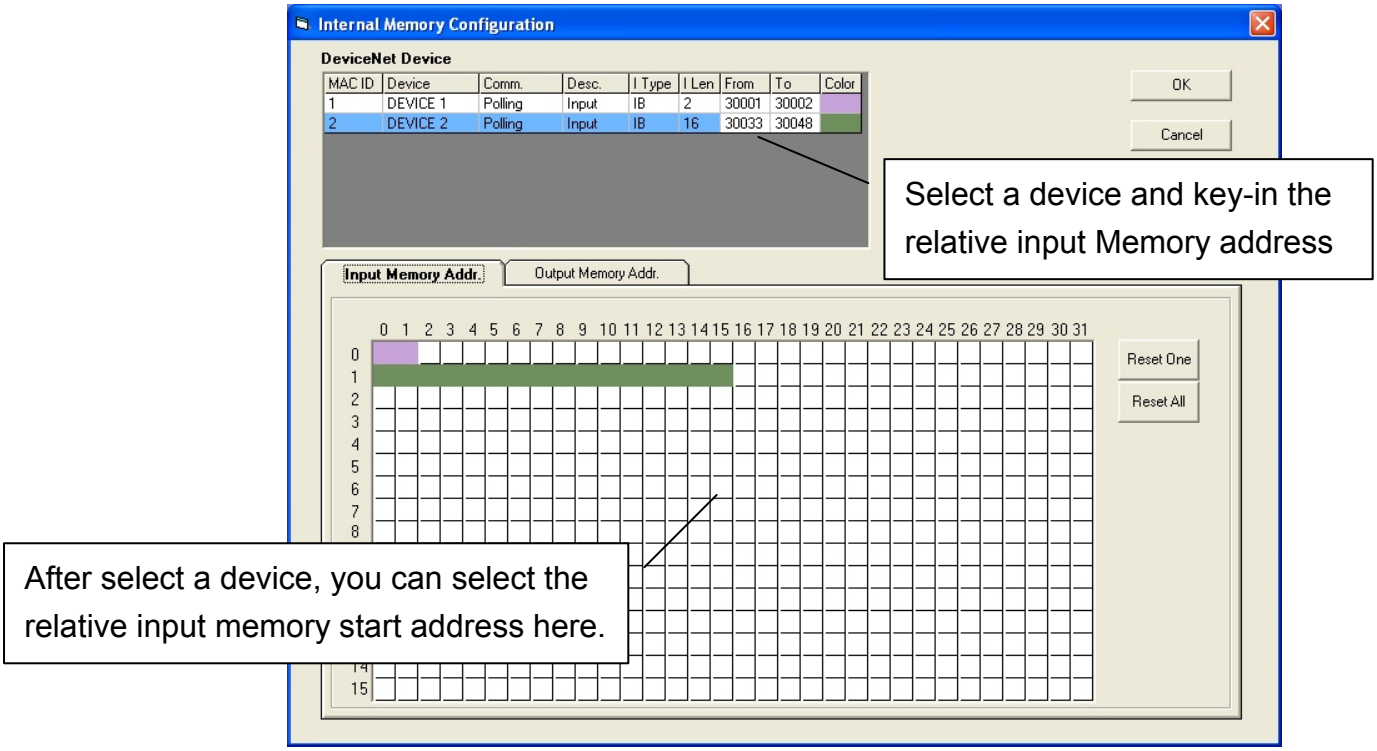


Figure 5-30: Input Memory Address

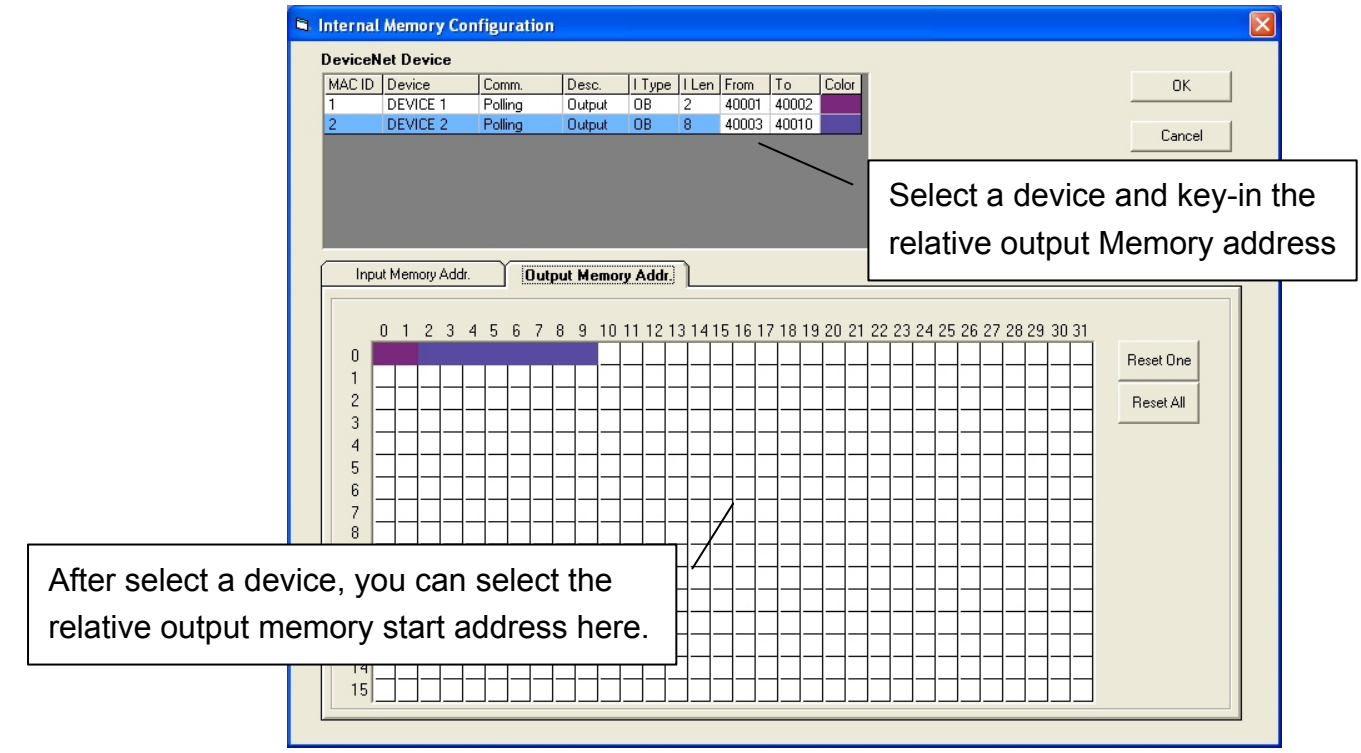


Figure 5-31: Output Memory Address

The “Reset One” and “Reset All” buttons are used for clear the memory address setting of each and every selected device. After the setting of the input/output memory address, please press the “OK” button to save the setting into the GW-7434D’s EEPROM.

Note:
Here the address 3xxxx/4xxxx on the “From” and “To” location meaning that it is input/output address, not real Modbus address.

The base address of the input and output address are starting from 0x0000
It means that :
30001 => 0000h
30002 => 0001h
…..

40001 => 0000h
40002 => 0001h
…..

Get/set devices input/output data into GW-7434D's input/output data area.

By using Utility tool, users can get/set data from the GW-7434D's input/output data area via Mosbus/TCP command. If the GW-7434D starts to communicate with the DeviceNet devices, all setting values on the output data area will be sent to the relative DeviceNet output devices. And the relative input data of DeviceNet input device will be saved into the input data area that you set. Please do the following steps to get/set the I/O data.

1. Select the master frame, GW-7434D
2. Select the "Setting" item on the menu bar, or click the mouse right-button on the master frame, then click the "Set/Get Memory Data…" item. Then the "Set / Get IO Memory Data" window will be pop-up. As shown in figure 5-32.

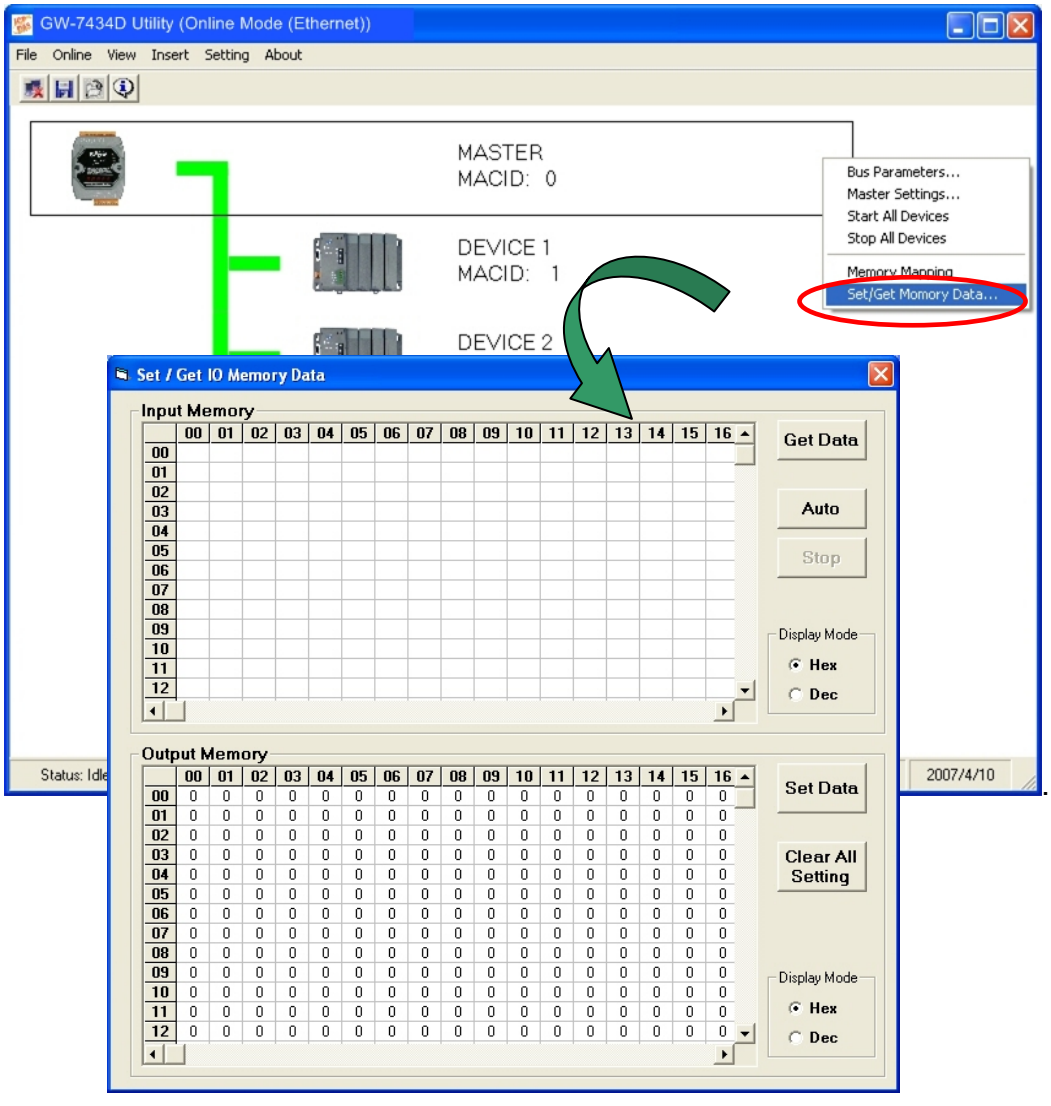


Figure 5-32: Set/Get IO Memory Data

As the setting of figure 5-30 and 5-31, users can get the Device1, 2 bytes polling input data, and Device2, 16 bytes polling data, on the Input Memory Table, 0000~0001 and 0100~0115 after click the “Get Data” button or “Auto button”. As shown in figure 5-33.

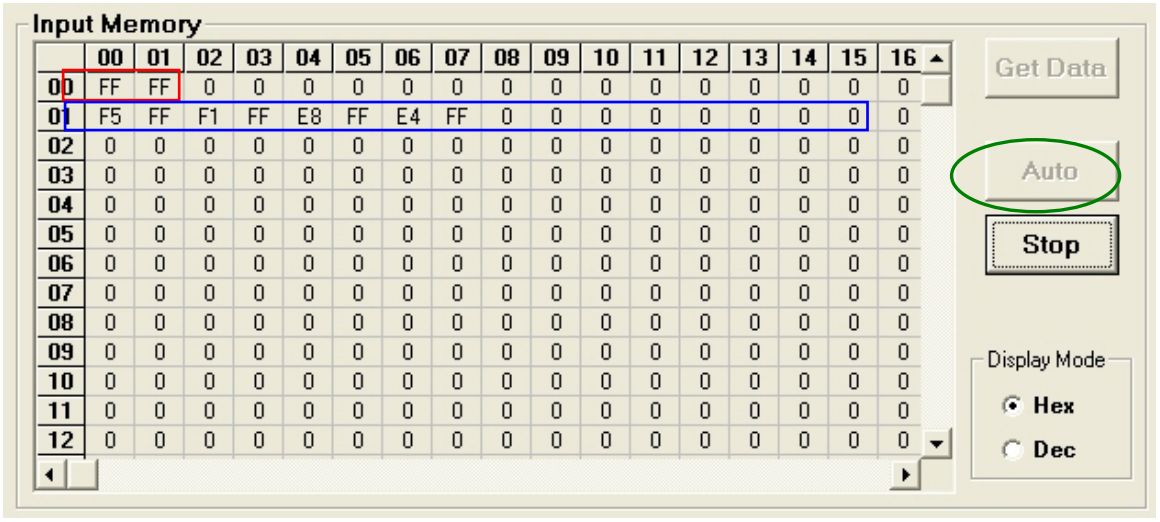


Figure 5-33: Input Memory Table

By clicking the “Set Data” button, users also can set the data on the Output Memory Table into GW-7434D’s output data area. As shown in figure 5-34.

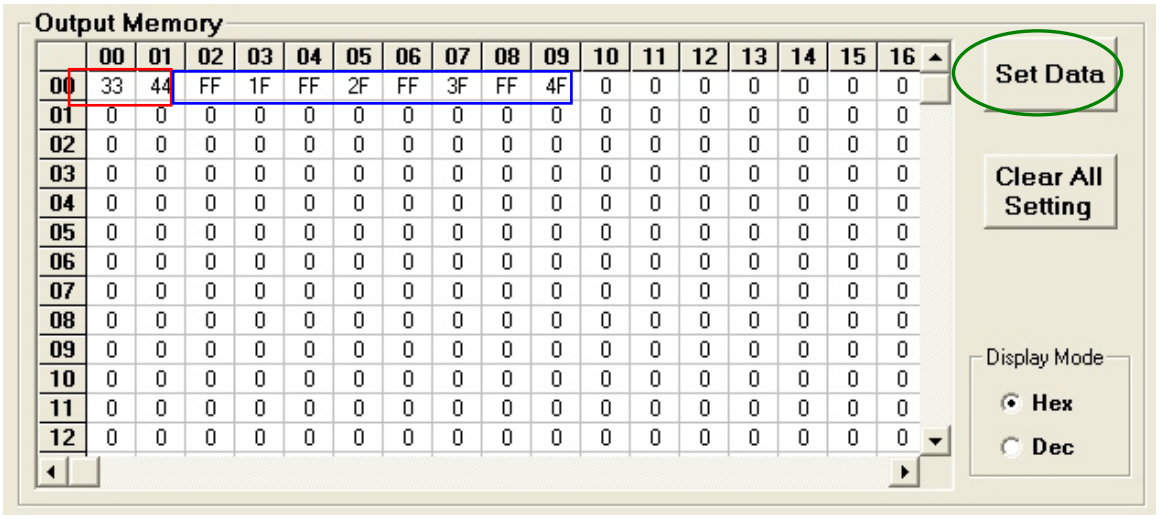


Figure 5-34: Output Memory Table

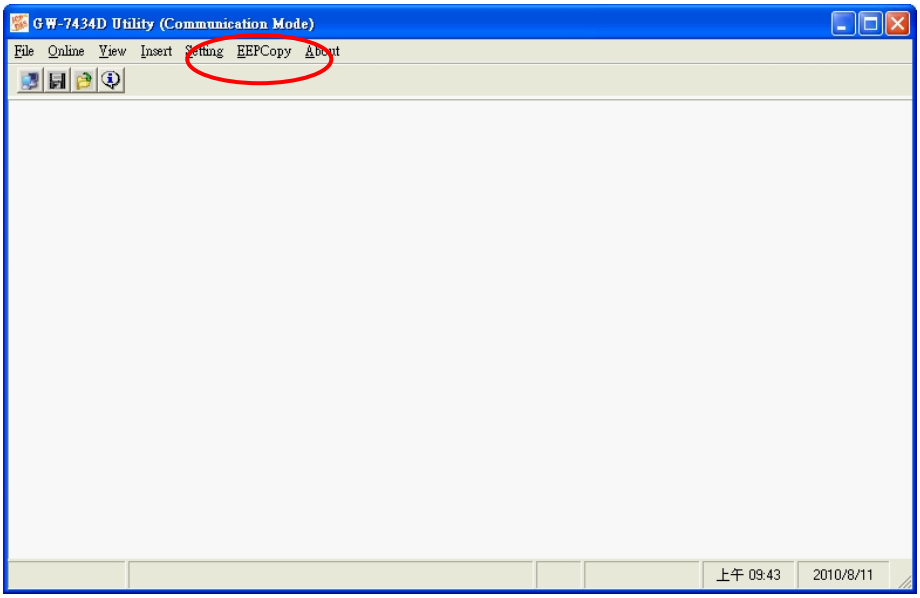
5.1.9 Import/export all configuration to/from GW-7434D's EEPPROM

Users can import all the configuration of the GW-7434D into an ini file and export the data in the ini file to another GW-7434D via the GW-7434D Utility. Please refer the following steps to import GW-7434D EEPROM's data into an ini file or export the data in the ini file to GW-7434D's EEPROM.

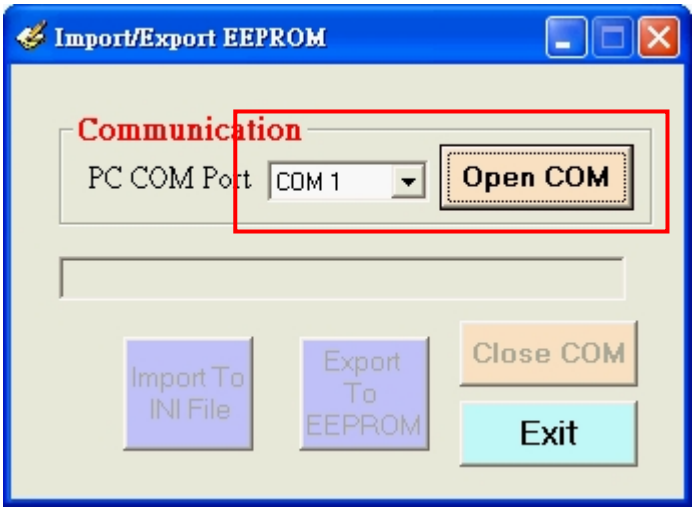
Step1: Turn off the power of the GW-7434D and switch the jump in the back plane of

the it into initial mode. The turn on the power of the GW-7434D.

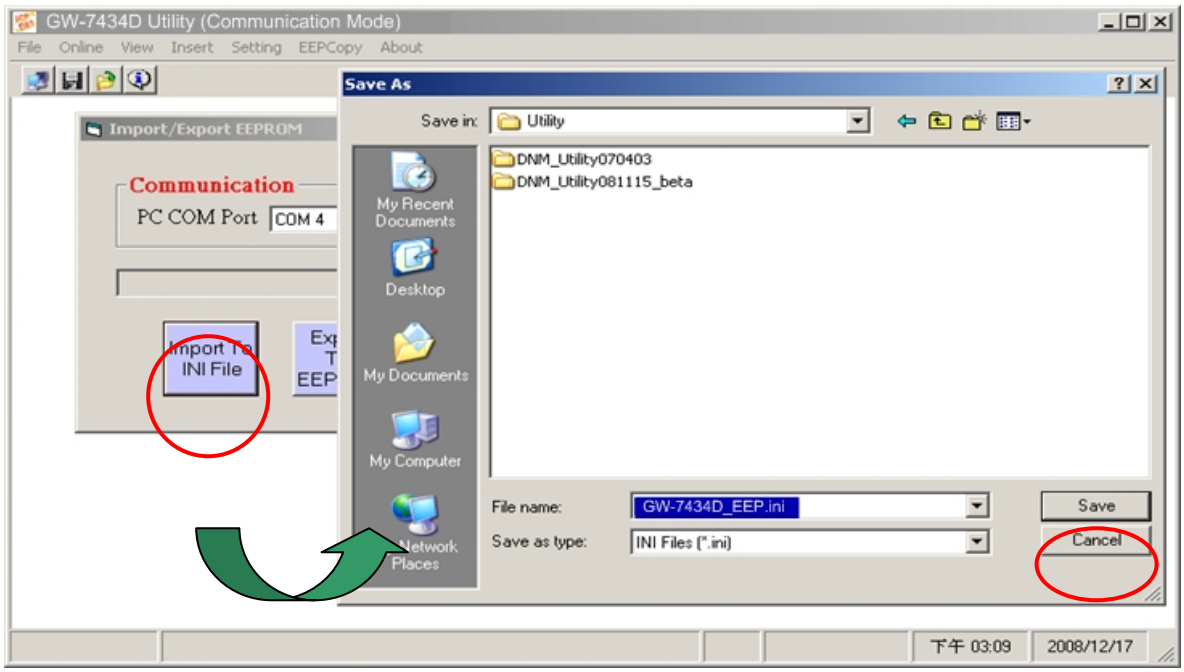
Step2: Press the EEPCopy button.



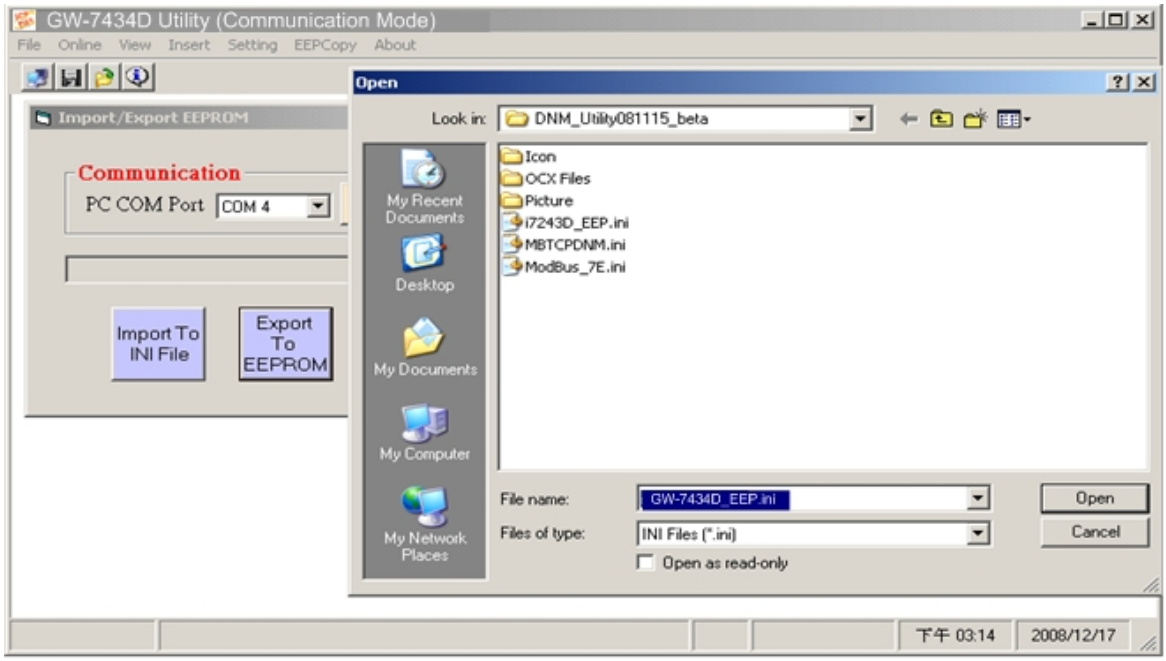
Step3: Select the necessary COM port of the PC and press the "Open COM" button to open the COM port



Step4: Then user can press the “import To Ini file” button to import all the EEPROM data of the GW-7434D into the selected ini file.



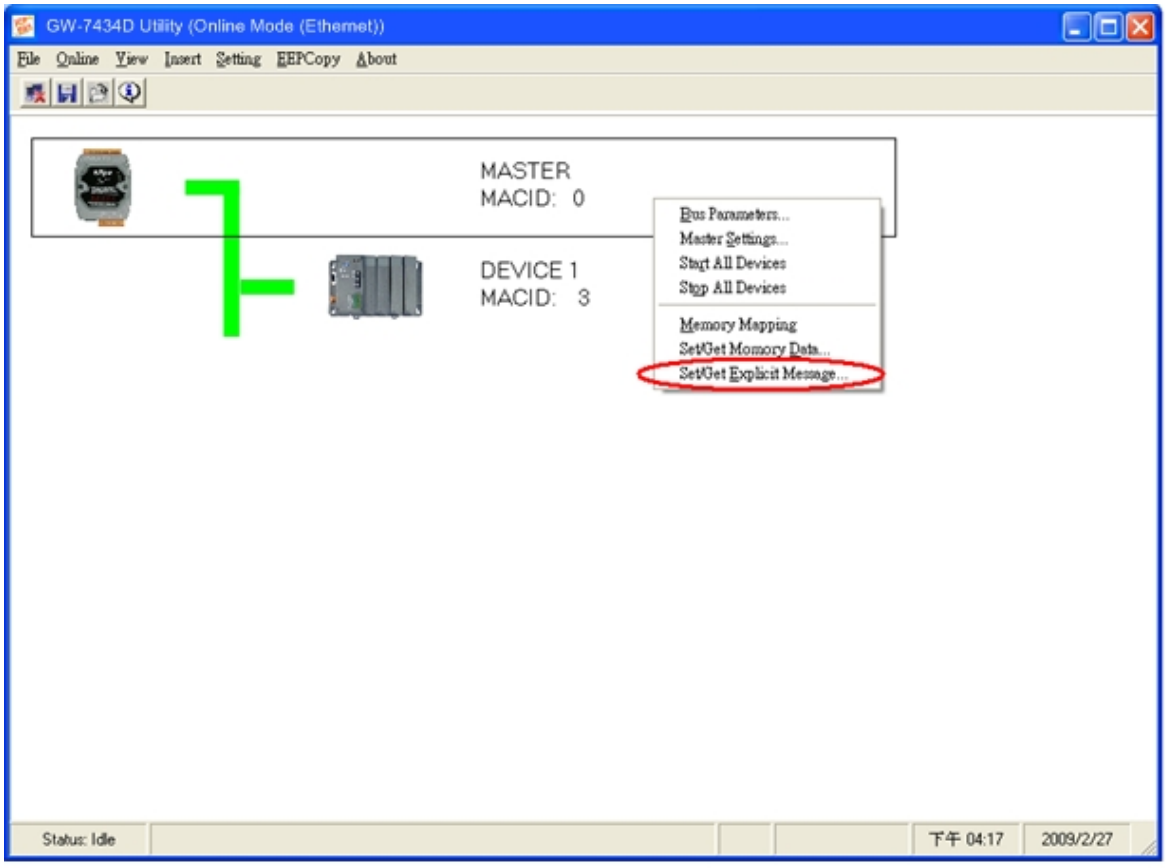
Step5: Or, user can press the “export to eeprom” button to export all the data in the ini file to GW-7434D’s eeprom.



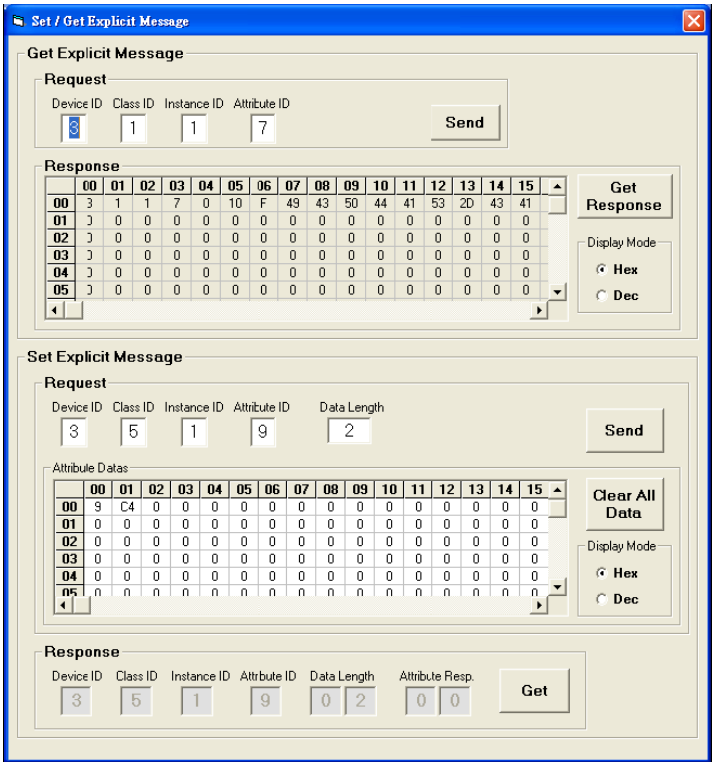
5.1.10 Get/Set Attribute Functions

Users can send “Get Attribute” request to retrieve extra information from the slave devices. Of course, users also can send “Set Attribute” request to change extra data within the slave devices. Here shows the operation steps.

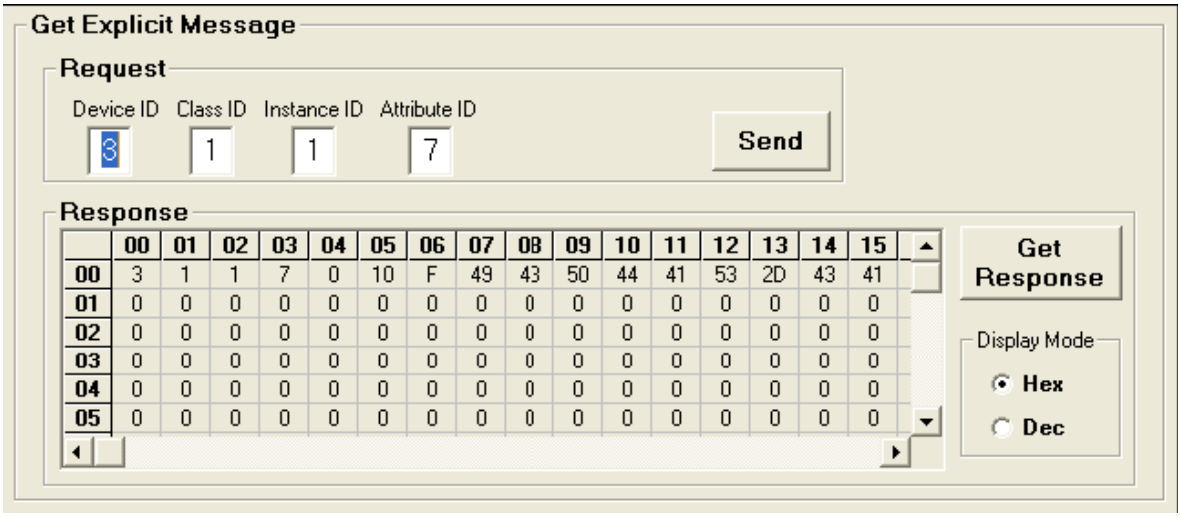
1 Select the master frame, GW-7434D and click the mouse right-button on the master frame. Please select the “Set/Get Explicit Message…” item.



2 The dialog shown below divided into two parts. One is “Get Explicit Message” and the other is “Set Explicit Message”.

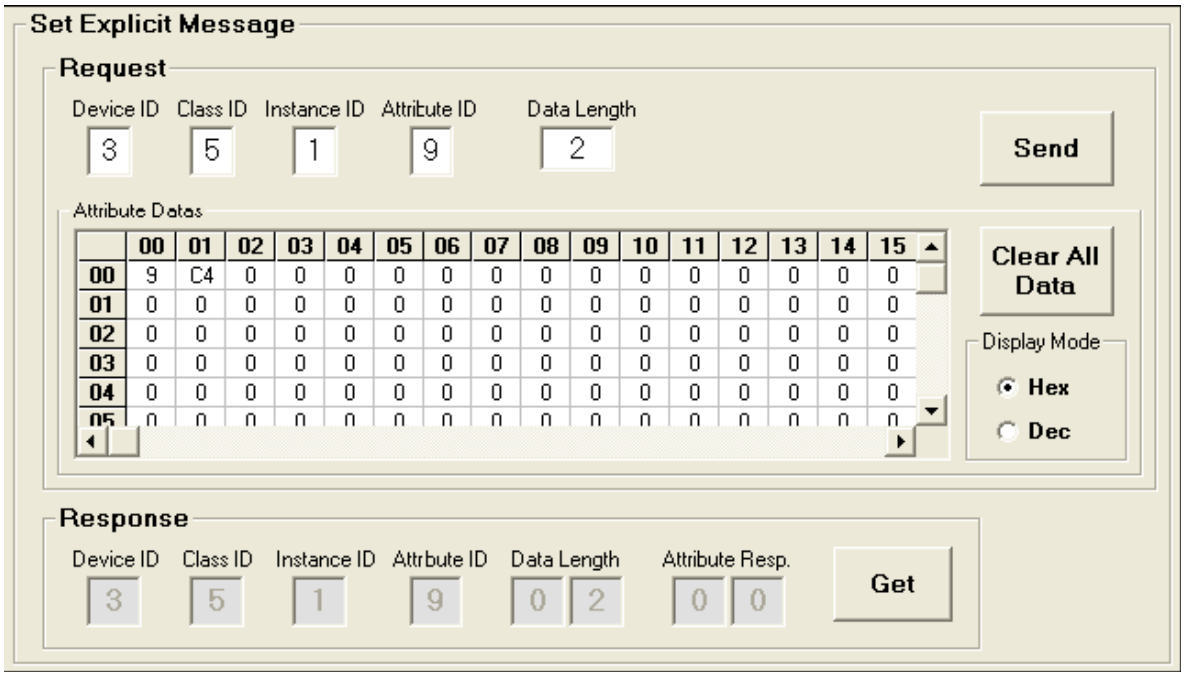


3 In the “Get Explicit Message”, users should input the parameters like “Device ID”,”Class ID”, and etc. Then click “Send” button to send the “Get Attribute” request to the remote slave device. After about 5 ~ 100ms (depend on the slave devices), the user could click “Get Response” to get the attribute data as shown below. Please refer to the description about word address of 1500h ~ 16FFh on page 42 to read the data format.



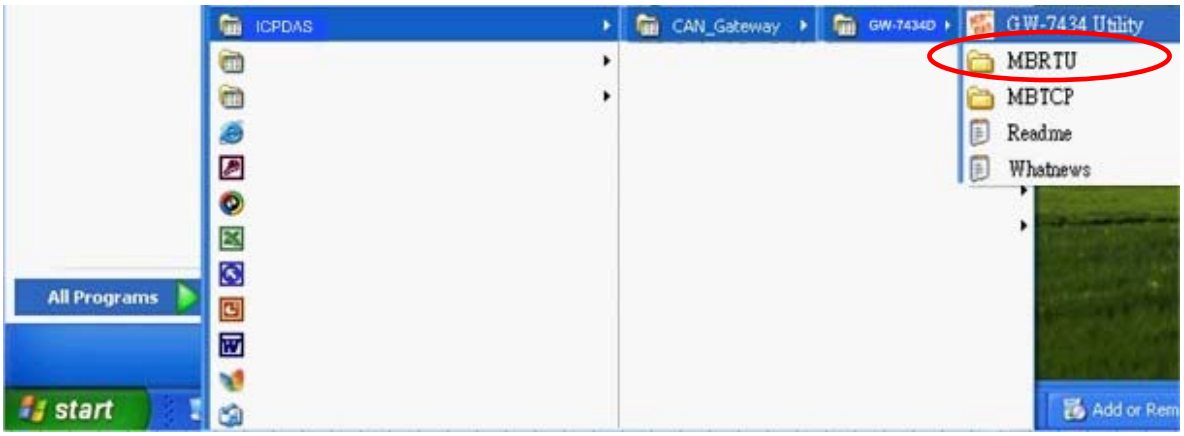
	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
00	3	1	1	7	0	10	F	49	43	50	44	41	53	2D	43	41
01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
02	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
03	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
04	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
05	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

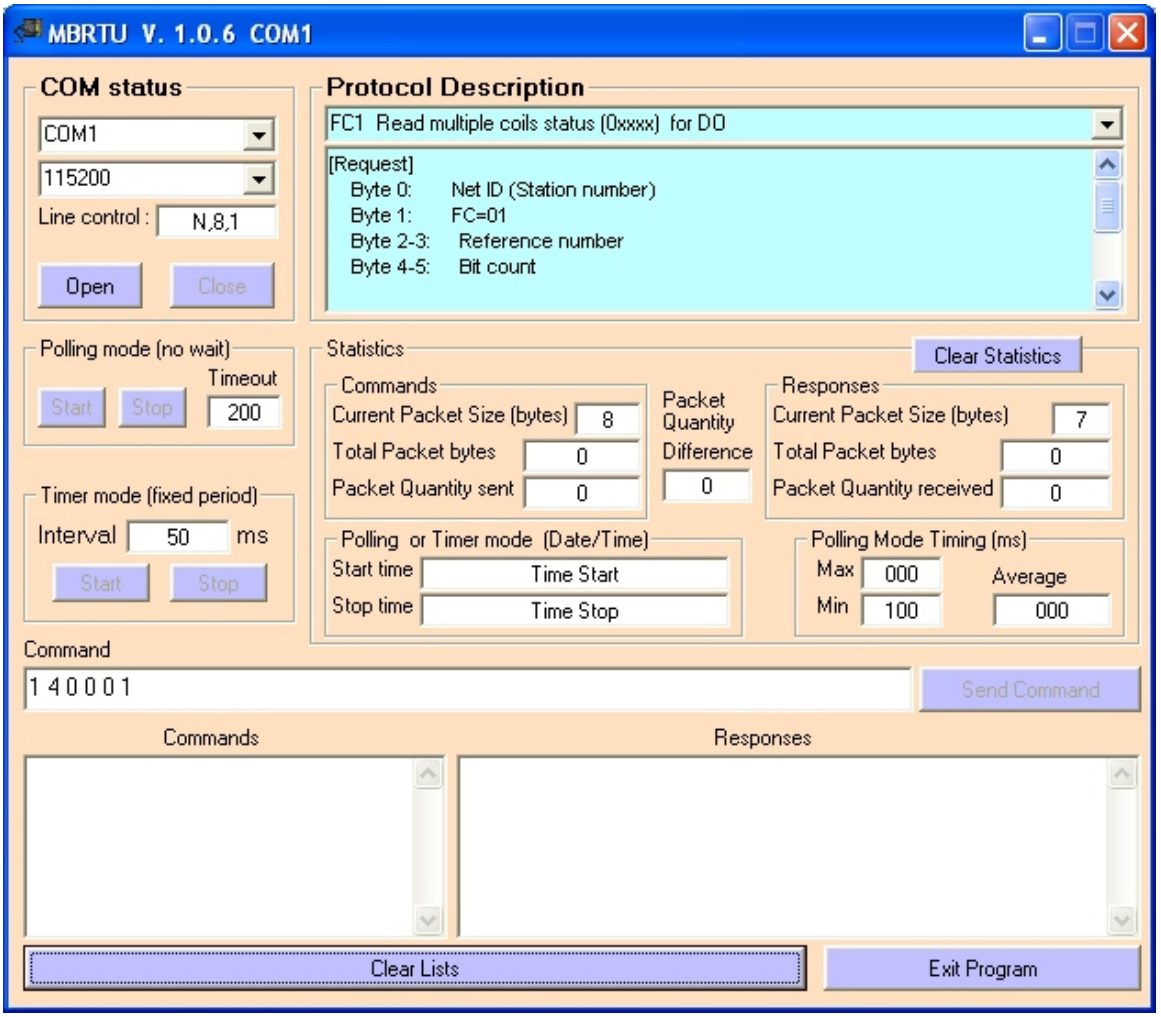
4 In the “Set Explicit Message”, users should input the parameters like “Device ID”,”Class ID”, “Attribute Data”, and etc. The “Data Length” parameter means the amount of the attribute data count in byte. Then click “Send” button to send the “Set Attribute” request to the remote slave device. After about 5 ~ 100ms (depend on the slave devices), the user could click “Get” to get the response data as shown below. Please refer to the description about word address of 1700h ~ 170Fh on page 43 to read the data format.



5.2 MBRTU Tool

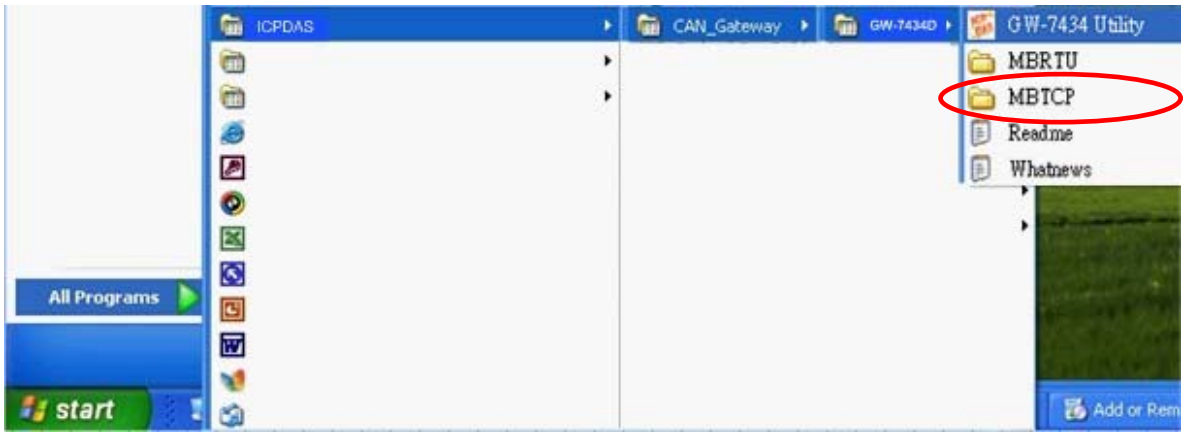
MBRTU: Modbus/RTU client (with source code in VB6) to diagnostic Modbus/RTU slave devices. After installing the GW-7434D Utility tool, users can get it from the following location.

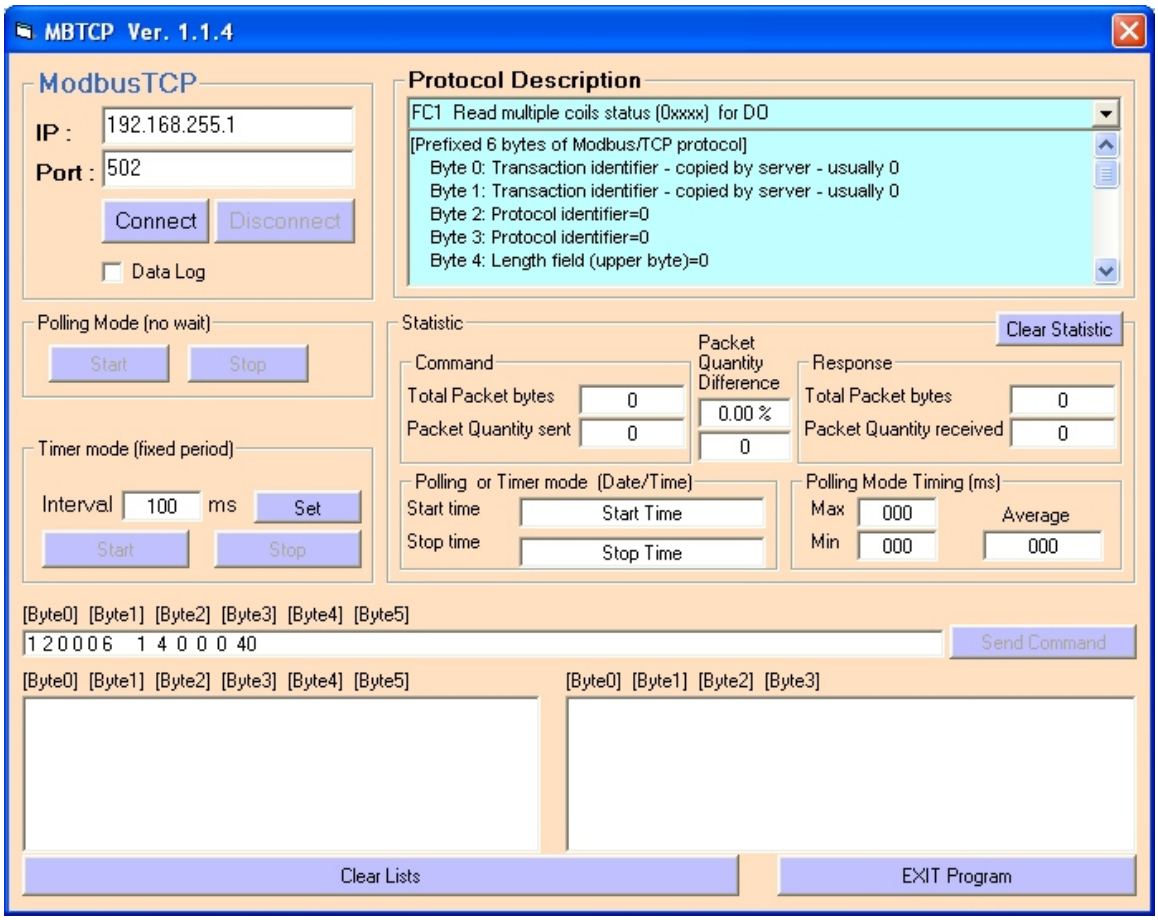




5.3 MBTCP Tool

MBTCP: Modbus/TCP client (with source code in VB6) to diagnostic Modbus/TCP slave devices. After installing the GW-7434D Utility tool, users can get it from the following location.





6. Specific Data Formats

By using Modbus/TCP command that contain specific data formats packed in the “Force Multiple Registers” command, users can operate and monitor the GW-7434D. After receiving this specific data format command, the GW-7434D will unscramble it and perform the relative procedure. The format of this data type is displayed in the following table.

Prefixed 6 bytes of Modbus/TCP protocol:

00	Transaction identifier	1 Byte	Usually 0x00
01	Transaction identifier	1 Byte	Usually 0x00
02	Protocol identifier	1 Byte	0x00
03	Protocol identifier	1 Byte	0x00
04	Length field(upper byte)	1 Byte	Number of bytes following
05	Length field(lower byte)	1 Byte	Number of bytes following

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Byte	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Byte	Dep. On Data
06	Byte Count (B=2*Word Count)	1 Byte	Dep. On Data
07~(B+6)	**Specific Data Format**	Dep. On Data	Dep. On Data

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Byte	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Byte	Dep. On Data

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

The following table displays the specific data formats that are supported by GW-7434D.

Leading byte	Description	Section
@ (0x40)	DNM_Reset	6.1.1
	DNM_SetMasterMACID	6.1.2
	DNM_SetBaudRate	6.1.3
	DNM_AddDevice	6.1.4
	DNM_RemoveDevice	6.1.5
	DNM_RemoveIOConnection	6.1.6
	DNM_ConfigBitStrobe	6.1.7
	DNM_ConfigPoll	6.1.8
	DNM_ConfigCOS	6.1.9
	DNM_ConfigCyclic	6.1.10
	DNM_StartDevice	6.1.11
	DNM_StopDevice	6.1.12
	DNM_StartAllDevice	6.1.13
	DNM_StopAllDevice	6.1.14
	DNM_ClearAllConfig	6.1.15
	DNM_GetAttribute	6.1.16
	DNM_SetAttribute	6.1.17
	DNM_AutoSearch	6.1.18
$ (0x24)	Set_Input_Data_Area	6.2.1
	Set_Output_Data_Area	6.2.2
	Clear_All_Input_Data_Area_Setting	6.2.3
	Clear_All_Output_Data_Area_Setting	6.2.4

6.1 Commands For Configuring DeviceNet Devices

The leading byte of the commands for configuring DeviceNet devices is the char '@', 0x40h. After receiving the command of this specific data format, the GW-7434D will unscramble it and perform the relative procedure. These commands for configuring DeviceNet devices are described in the following sub-section.

6.1.1 DNM_Reset

This command is used to reset the system of the GW-7434D

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x01
06	Byte Count (B=2*Word Count)	1 Byte	0x02
07	Leading byte = '@'	1 Byte	0x40
08	Reset Command	1 Byte	**0x2F**

Response:

No Response. After receiving this command, the GW-7434D will reset the system and disconnect with all other clients.

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.2 DNM_SetMasterMACID

The command is used for users to set the DeviceNet MACID of the GW-7434D. After resetting the system of the GW-7434D, the configuration of MACID will take affect.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x02
06	Byte Count (B=2*Word Count)	1 Byte	0x04
07	Leading byte = '@'	1 Byte	0x40
08	Set MACID Command	1 Byte	**0x02**
09	Separating byte = '!'	1 Byte	0x21
10	MAC ID	1 Byte	0x00~0x3F

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x02

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.3 DNM_SetBaudRate

The command is used for users to set the DeviceNet baud-rate of the GW-7434D. After resetting the system of the GW-7434D, the configuration of baud-rate will take affect.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x02
06	Byte Count (B=2*Word Count)	1 Byte	0x04
07	Leading byte = '@'	1 Byte	0x40
08	Set baud-rate Command	1 Byte	**0x03**
09	Separating byte = '!'	1 Byte	0x21
10	Baud-rate of DeviceNet	1 Byte	0x00: 125K 0x01: 250K 0x02: 500K

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x02

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.4 DNM_AddDevice

The command is used for users to add a DeviceNet device into the GW-7434D's scan-list table. This command is used to add device, but not start to communicate with the device.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x03
06	Byte Count (B=2*Word Count)	1 Byte	0x06
07	Leading byte = '@'	1 Byte	0x40
08	Add Device Command	1 Byte	**0x04**
09	Separating byte = '!'	1 Byte	0x21
10	Device MACID	1 Byte	0x00~0x3F
11	EPR of Explicit connection (high byte)	1 Byte	0x00~0xFF
12	EPR of Explicit connection (low byte)	1 Byte	0x00~0xFF

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x03

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.5 DNM_RemoveDevice

The command is used for users to remove a DeviceNet device from the GW-7434D's scan-list table. After removing the device, the GW-7434D will stop communicating with it. And all the connections between them will be released.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x02
06	Byte Count (B=2*Word Count)	1 Byte	0x04
07	Leading byte = '@'	1 Byte	0x40
08	Remove Device Command	1 Byte	**0x05**
19	Separating byte = '!'	1 Byte	0x21
10	Device MACID	1 Byte	0x00~0x3F

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x02

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.6 DNM_RemoveIOConnection

The command is used for user to remove a connected IO connection between the DeviceNet device and GW-7434D. After using the command to remove the connection, the connection will be released by the GW-7434D. And the configuration of this connection will be erased from the scan-list table of GW-7434D.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x03
06	Byte Count (B=2*Word Count)	1 Byte	0x06
07	Leading byte = '@'	1 Byte	0x40
08	Remove IO Connection Command	1 Byte	**0x06**
09	Separating byte = '!'	1 Byte	0x21
10	Device MACID	1 Byte	0x00~0x3F
11	Connection Type	1 Byte	0x00: Explicit 0x01: Poll 0x02: Bit-Strobe 0x03: COS 0x04: Cyclic
12	End Char = CR (0x0D)	1 Byte	0x0D

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x03

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.7 DNM_ConfigBitStrobe

The command is used for users to configure the Bit-Strobe IO connection between the DeviceNet device and the GW-7434D. This command is used to configure the IO connection, but not start to communicate with the device.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x04
06	Byte Count (B=2*Word Count)	1 Byte	0x08
07	Leading byte = '@'	1 Byte	0x40
08	Configure Bit-Strobe Command	1 Byte	**0x07**
09	Separating byte = '!'	1 Byte	0x21
10	Device MACID	1 Byte	0x00~0x3F
11	Input data length of device (high byte)	1 Byte	Dep. on input data length of device
12	Input data length of device (low byte)	1 Byte	Dep. on input data length of device
13	EPR of Bit-Strobe connection (high byte)	1 Byte	0x00~0xFF
14	EPR of Bit-Strobe connection (low byte)	1 Byte	0x00~0xFF

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x04

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.8 DNM_ConfigPoll

The command is used for users to configure the Poll IO connection between the DeviceNet device and the GW-7434D. This command is used to configure the IO connection, but not start to communicate with the device.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x05
06	Byte Count (B=2*Word Count)	1 Byte	0x0A
07	Leading byte = '@'	1 Byte	0x40
08	Configure Poll Command	1 Byte	**0x08**
09	Separating byte = '!'	1 Byte	0x21
10	Device MACID	1 Byte	0x00~0x3F
11	Input data length of device (high byte)	1 Byte	Dep. on input data length of device
12	Input data length of device (low byte)	1 Byte	Dep. on input data length of device
13	output data length of device (high byte)	1 Byte	Dep. on output data length of device
14	output data length of device (low byte)	1 Byte	Dep. on output data length of device
15	EPR of Poll connection (high byte)	1 Byte	0x00~0xFF
16	EPR of Poll connection (low byte)	1 Byte	0x00~0xFF

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x05

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.9 DNM_ConfigCOS

The command is used for users to configure the COS IO connection between the DeviceNet device and the GW-7434D. This command is used to configure the IO connection, but not start to communicate with the device.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x05
06	Byte Count (B=2*Word Count)	1 Byte	0x0A
07	Leading byte = '@'	1 Byte	0x40
08	Configure COS Command	1 Byte	**0x09**
09	Separating byte = '!'	1 Byte	0x21
10	Device MACID	1 Byte	0x00~0x3F
11	Input data length of device (high byte)	1 Byte	Dep. on input data length of device
12	Input data length of device (low byte)	1 Byte	Dep. on input data length of device
13	output data length of device (high byte)	1 Byte	Dep. on output data length of device
14	output data length of device (low byte)	1 Byte	Dep. on output data length of device
15	EPR of COS connection (high byte)	1 Byte	0x00~0xFF
16	EPR of COS connection (low byte)	1 Byte	0x00~0xFF

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x05

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.10 DNM_ConfigCyclic

The command is used for user to configure the Cyclic IO connection between the DeviceNet device and the GW-7434D. This command is used to configure the IO connection, but not start to communicate with the device.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x05
06	Byte Count (B=2*Word Count)	1 Byte	0x0A
07	Leading byte = '@'	1 Byte	0x40
08	Configure Cyclic Command	1 Byte	**0x0A**
09	Separating byte = '!'	1 Byte	0x21
10	Device MACID	1 Byte	0x00~0x3F
11	Input data length of device (high byte)	1 Byte	Dep. on input data length of device
12	Input data length of device (low byte)	1 Byte	Dep. on input data length of device
13	output data length of device (high byte)	1 Byte	Dep. on output data length of device
14	output data length of device (low byte)	1 Byte	Dep. on output data length of device
15	EPR of Cyclic connection (high byte)	1 Byte	0x00~0xFF
16	EPR of Cyclic connection (low byte)	1 Byte	0x00~0xFF

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x05

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.11 DNM_StartDevice

After configuring the IO connection of each device, users can use this command to let the GW-7434D start to communicate with each DeviceNet device.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x02
06	Byte Count (B=2*Word Count)	1 Byte	0x04
07	Leading byte = '@'	1 Byte	0x40
08	Start Device Command	1 Byte	**0x0B**
09	Separating byte = '!'	1 Byte	0x21
10	Device MACID	1 Byte	0x00~0x3F

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x02

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.12 DNM_StopDevice

Users can use this command to release the connection between GW-7434D and device. The configuration of this connection still is saved in the scan-list table of GW-7434D.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x02
06	Byte Count (B=2*Word Count)	1 Byte	0x04
07	Leading byte = '@'	1 Byte	0x40
08	Stop Device Command	1 Byte	**0x0C**
09	Separating byte = '!'	1 Byte	0x21
10	Device MACID	1 Byte	0x00~0x3F

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x02

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.13 DNM_StartAllDevice

After configuring the IO connection of each device, users can use this command to let the GW-7434D start to communicate with all DeviceNet devices.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x01
06	Byte Count (B=2*Word Count)	1 Byte	0x02
07	Leading byte = '@'	1 Byte	0x40
08	Start All Device Command	1 Byte	**0x0D**

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x01

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.14 DNM_StopAllDevice

Users can use this command to release the connections between GW-7434D and all DeviceNet devices. These configurations of these connections still are saved in the scan-list table of GW-7434D.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x01
06	Byte Count (B=2*Word Count)	1 Byte	0x02
07	Leading byte = '@'	1 Byte	0x40
08	Start All Device Command	1 Byte	**0x0E**

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x01

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.15 DNM_ClearAllConfig

Users can use this command to clear all the configurations of devices, in the GW-7434D's scan-list table. After receiving this command by the GW-7434D, it will release all connection between the devices, and clear all configurations of the DeviceNet devices.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x01
06	Byte Count (B=2*Word Count)	1 Byte	0x02
07	Leading byte = '@'	1 Byte	0x40
08	Start All Device Command	1 Byte	**0x49**

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x01

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.16 DNM_GetAttribute

This function is used to send the request command to retrieve the attribute value of the specific device's instance. Before calling this function, the users must start the device (DNM_StartDevice or DNM_StartAllDevice). After calling this function, the users can read the address 1500h of the "Input Status Area" to get the attribute value returned from the remote slave device.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x04
06	Byte Count (B=2*Word Count)	1 Byte	0x07
07	Leading byte = '@'	1 Byte	0x40
08	Get Attribute Command	1 Byte	**0x0F**
09	Separating byte = '!'	1 Byte	0x21
10	Device MACID	1 Byte	0x00~0x3F
11	Class ID	1 Byte	Class ID
12	Instance ID	1 Byte	Instance ID
13	Attribute ID	1 Byte	Attribute ID

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x04

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.17 DNM_SetAttribute

The method is used to set the attribute of the specific device's instance. Before calling this function, the users must start the device (DNM_StartDevice or DNM_StartAllDevice). After calling this function, the users can read the address 1700h of the "Input Status Area" to get the response value returned from the remote slave device.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	Dep. on Byte Count
06	Byte Count (B=2*Word Count)	1 Byte	0x07 + Attribute Data Length (Byte).
07	Leading byte = '@'	1 Byte	0x40
08	Set Attribute Command	1 Byte	**0x11**
09	Separating byte = '!'	1 Byte	0x21
10	Device MACID	1 Byte	0x00~0x3F
11	Class ID	1 Byte	Class ID
12	Instance ID	1 Byte	Instance ID
13	Attribute ID	1 Byte	Attribute ID
14 ~15	Attribute Data Length	2 Bytes	Data Length
16	Attribute Data 0	1 Byte	0x00~0xFF
17	Attribute Data 1	1 Byte	0x00~0xFF
18	:	:	:

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	Dep. on Byte Count

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.1.18 DNM_AutoSearch

This function is used to retrieve all devices in DeviceNet network. Attention! This function will terminate all communications with remote devices. This function is usually used for developing or debugging applications. After calling this function, please wait for about 30 seconds. Then, the users can visit the address 1000h of the "Input Status Ares" to get the information of all slaves within the DeviceNet network.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	0x00 0x01
06	Byte Count (B=2*Word Count)	1 Byte	0x02
07	Leading byte = '@'	1 Byte	0x40
08	Auto Search Command	1 Byte	**0x25**

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	0x00 0x01

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.2 Commands for Configuring Input/Output Data Area

The leading byte of the commands for configuring I/O Data Area is the char '$', 0x24h. After receiving the command of this specific data format, the GW-7434D will unscramble it and perform the relative procedure. These commands for configuring DeviceNet devices are described in the following sub-section.

6.2.1 Set_Input_Data_Area

Users can use this command to allocate the data of DeviceNet input devices into certain address of Input Data Area. Before using this command, users must know what's IO connection of the input device already been established. The structure of this command is shown below.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	Dep. on device Num.
06	Byte Count (B=2*Word Count)	1 Byte	Dep. on device Num
07	Leading byte = '$'	1 Byte	0x24
08	Configure Input Area Command	1 Byte	**0x00**
09	Device Count	1 Byte	0x01~0x3F
10	Device1, MACID	1 Byte	0x00~0x3F
11~12	Device1, Start address of Input Data Area	2 Bytes	0x0000~0x01FF
13	Device2, MACID	1 Byte	0x00~0x3F
14~15	Device2, Start address of Input Data Area	2 Bytes	0x0000~0x01FF
16	Device3, MACID	1 Byte	0x00~0x3F
17~18	Device3, Start address of Input Data Area	2 Bytes	0x0000~0x01FF
: :	: :	: :	: :

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	Dep. on request command Length

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

Note:

1. The parameter of these parameters must be suitable with the connection attribute values of the input device. Otherwise, the data of input device will not be stored into the Input Data Area.

2. After mapping the device data into the Input Data Area, all the parameters will be stored into the Input Status Area address, 1800h~1BF0h, table 4-7, and EEPROM of GW-7434D.

6.2.2 Set_Output_Data_Area

Users can use this command to allocate certain address of Output Data Area for the data of DeviceNet output devices. Before using this command, users must know what's IO connection of the output device already been established. The structure of this command is shown below.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Bytes	Dep. on device Num.
06	Byte Count (B=2*Word Count)	1 Byte	Dep. on device Num
07	Leading byte = '$'	1 Byte	0x24
08	Configure Output Area Command	1 Byte	**0x01**
09	Device Count	1 Byte	0x01~0x3F
10	Device1, MACID	1 Byte	0x00~0x3F
11~12	Device1, Start address of Input Data Area	2 Bytes	0x0000~0x01FF
13	Device2, MACID	1 Byte	0x00~0x3F
14~15	Device2, Start address of Input Data Area	2 Bytes	0x0000~0x01FF
16	Device3, MACID	1 Byte	0x00~0x3F
17~18	Device3, Start address of Input Data Area	2 Bytes	0x0000~0x01FF
: :	: :	: :	: :

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Bytes	0x0C 0x00
04~05	Word Count	2 Bytes	Dep. on request command Length

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

Note:

1. The parameters of the configuration must be suitable with the connection attribute values of the output device. Otherwise, the data in the Output Data Area will not be sent to the output device.

2. After mapping the device data into the Output Data Area, all the parameters will be stored into the Output Status Area address, 0800h~0BF0h, table 4-6, and EEPROM of GW-7434D.

6.2.3 Clear_All_Input_Data_Area

Users can use this command to clear all the data in the Input Data Area of the GW-7434D. After receiving this command by the GW-7434D, it will set the flag, Total Number of Input commands, table 4-7, and all data in the Input Data Area to zero.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Byte	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Byte	0x00 0x01
06	Byte Count (B=2*Word Count)	1 Byte	0x02
07	Leading byte = '$'	1 Byte	0x24
08	Clear all data in the Input Data Area	1 Byte	**0x02**

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Byte	0x0C 0x00
04~05	Word Count	2 Byte	0x00 0x01

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.2.4 Clear_All_Output_Data_Area

Users can use this command to clear all the data in the Output Data Area of the GW-7434D. After receiving this command by the GW-7434D, it will set the flag, Total Number of output commands, table 4-6, and all data in the Output Data Area to zero.

Request:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Byte	**0x0C 0x00 (Command Area)**
04~05	Word Count	2 Byte	0x00 0x01
06	Byte Count (B=2*Word Count)	1 Byte	0x02
07	Leading byte = '$'	1 Byte	0x24
08	Clear all data in the Output Data Area	1 Byte	**0x03**

Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x10
02~03	Reference Number	2 Byte	0x0C 0x00
04~05	Word Count	2 Byte	0x00 0x01

Error Response:

00	Net ID (Station Number)	1 Byte	Module Net ID
01	Function Code	1 Byte	0x90
02	Exception Code	1 Byte	Refer to table 4-2 for more details.

6.3 The Communicated Flow Diagram

The following flow diagram mainly presents how to configure the GW-7434D by the specific Modbus/TCP commands. To configure the module, users just need to send the specific commands step by step or configure it by the Utility tool. For more information, please refer to the command description, in section 6.1 and 6.2, and operating steps of the Utility tool, in section 5.1.

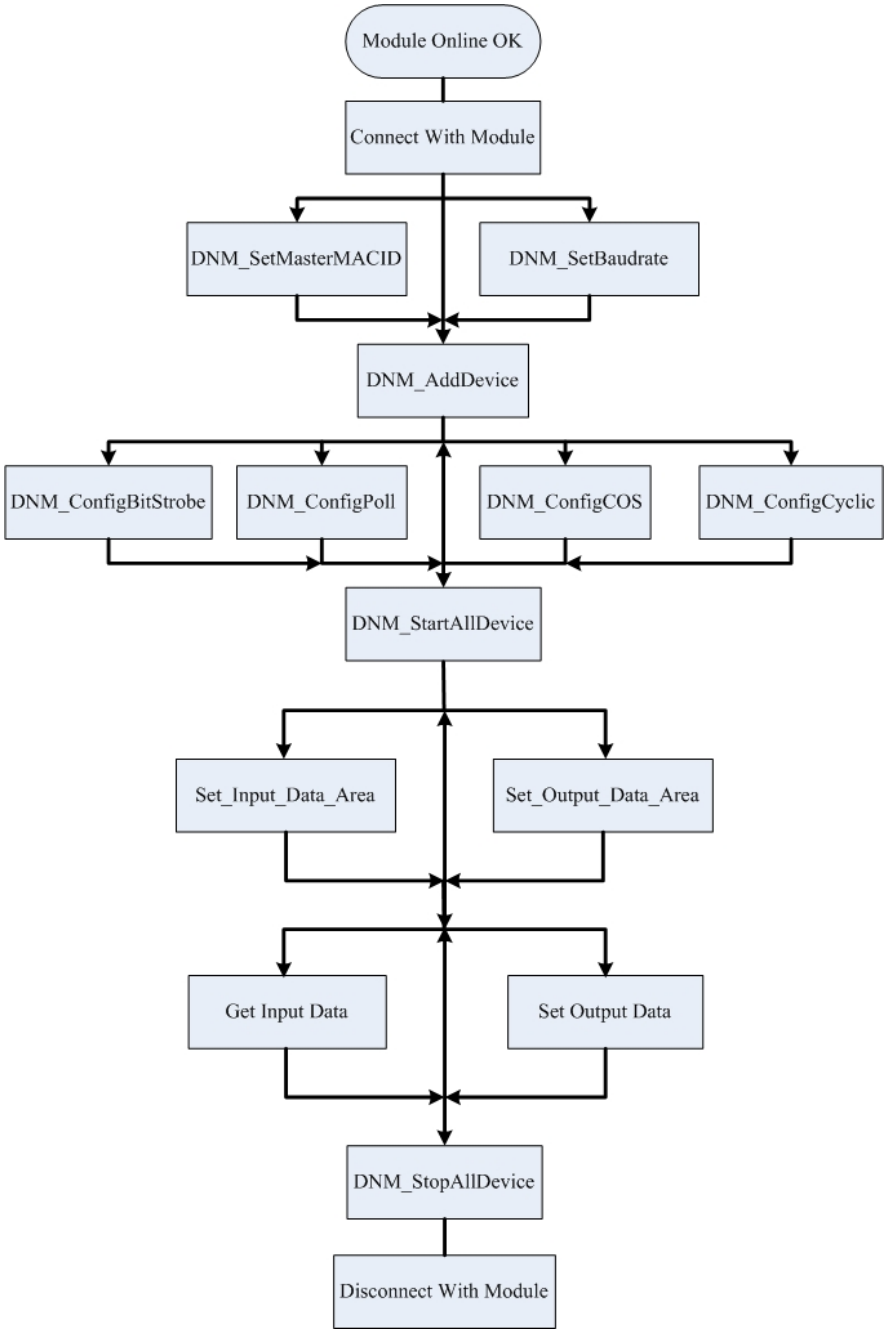


Figure 6-1: Flow diagram of configuring steps

7. VxComm Applications

- Overview
- Installing the VxComm Driver
- Adding a GW-7434D and configuring the VxComm Driver
- Removing a GW-7434D
- Uninstalling the VxComm Driver

7.1 Overview

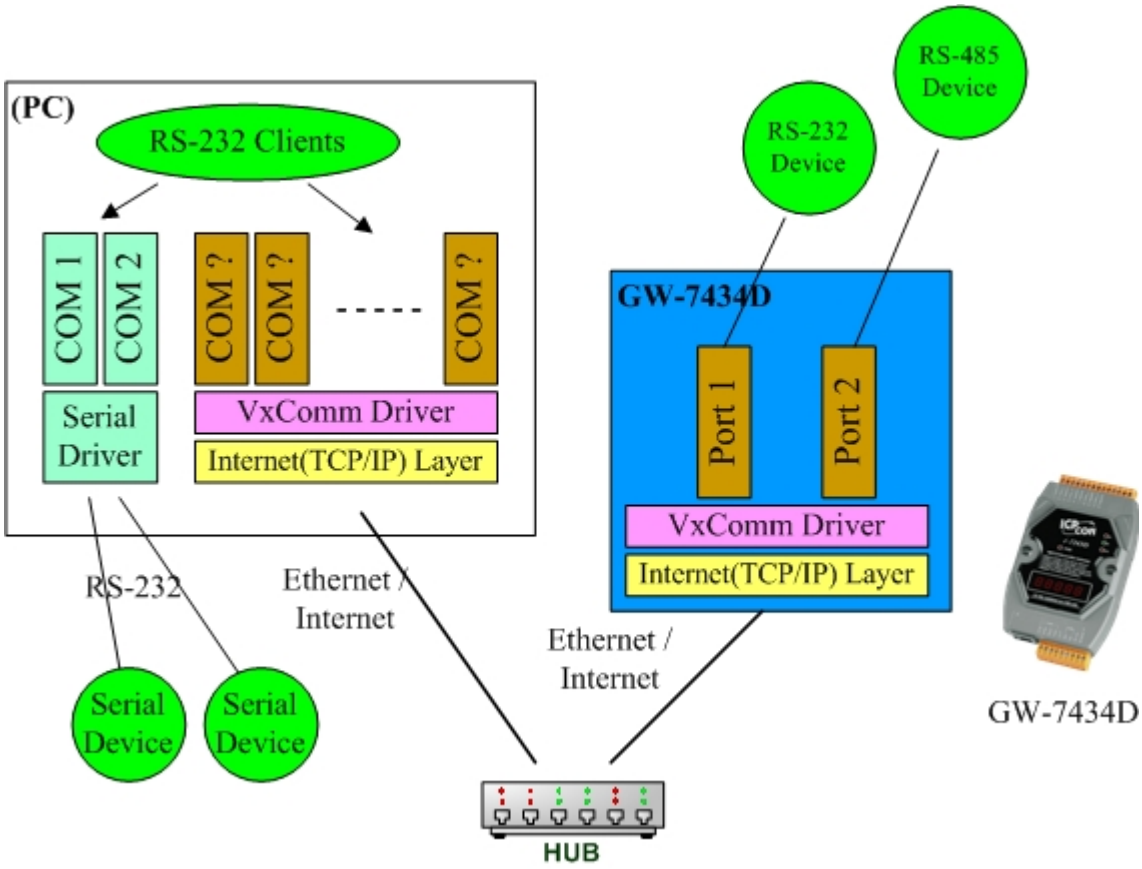
The VxComm (Virtual Comm) Driver and VxComm Utility are very easy to install and use. The first thing to do is to find the installation file in the included CD. The directory is:

- 8000CD:\\Napdos\7188e\Tcp\VxComm\driver(pc)\NT\VxCommNT.exe
  (for Windows NT 4.0) or
- 8000CD:\\Napdos\7188e\Tcp\VxComm\driver(pc)2K\VxComm2K.exe
  (for Windows 2000, Windows XP).

This document shows how to install and configure the driver correctly. The first part instructs users how to install the software. The second part shows how to add an GW-7434D server and configure a COM port. Finally, the third part teaches you how to remove an GW-7434D server.

7.1.1 Architecture

The VxComm Driver creates COM port(s) and maps them to the COM port(s) of the GW-7434D. The user's RS-232 client programs need only to change to the different COM port to access the serial devices that are allocated to the Internet or Ethernet network via the GW-7434D.



7.1.2 Ports mapping

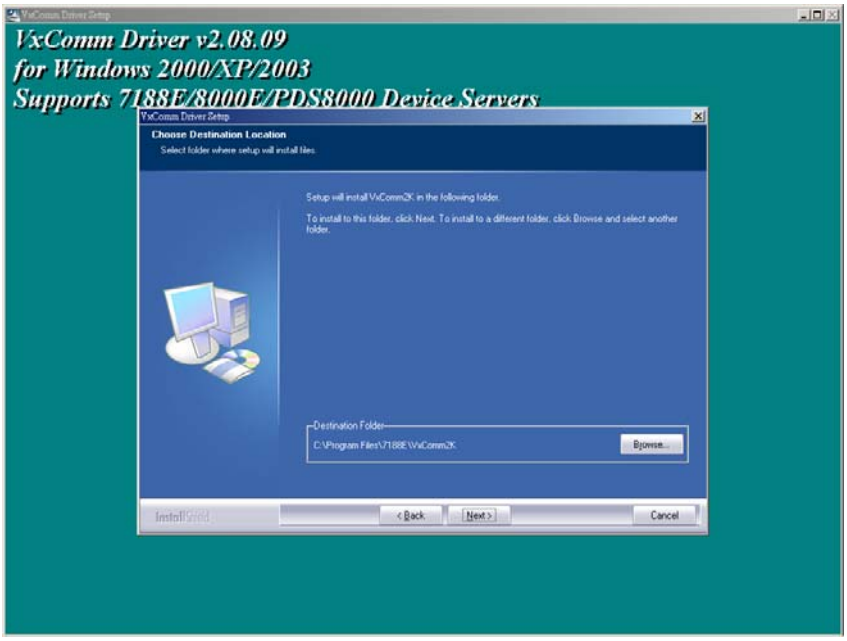
Vxcomm Driver/Utility supports Port 1 to Port 2 in accessing COM1 to COM2 of the GW-7434D. Another Port I/O is designed to access the I/O boards mounted on GW-7434D, but it doesn't use now. With the help of the VxComm Driver/Utility, uses can map remote COM port to become a virtual COM port of PC. One PC can control maximum number of 256 COM ports (including COM1 and COM2).

Local COM Port (PC)	VxComm Driver/Utility (PC)	Remote COM port (GW-7434D)
COM ?	Port 1	COM1
COM ?	Port 2	COM2
COM ?	Port I/O	Reserved

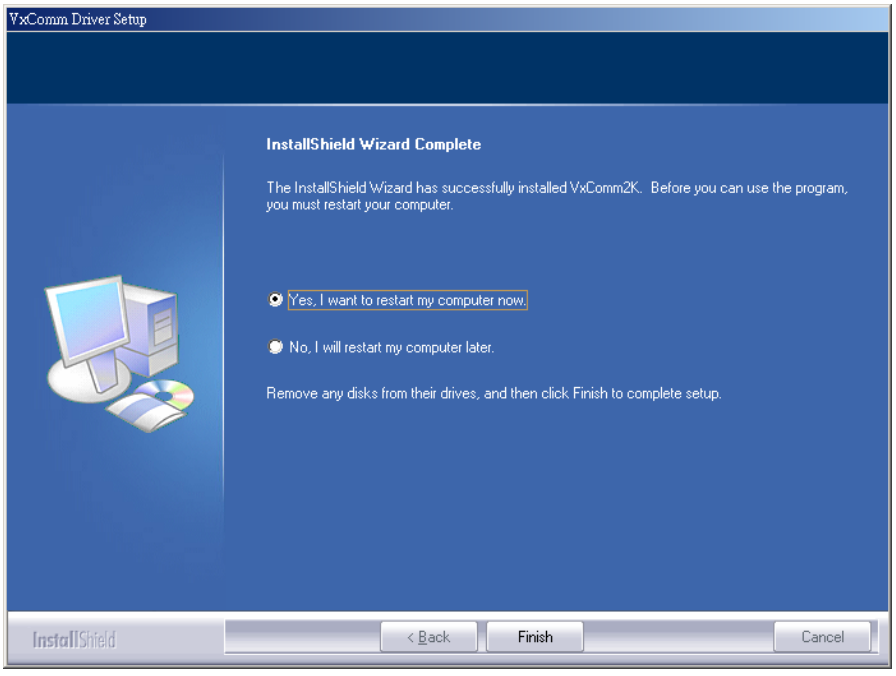
7.2 Installing the VxComm Driver

Step 1: Run VxComm2K.exe (for Windows 2000, Windows XP) or VxCommNT.exe (WindowsNT 4.0) in the packaged CD to start installing.

Step 2: Choose a destination folder.



Step 3: Select the "Yes, ... " option and click the "Finish" button to reboot your computer.

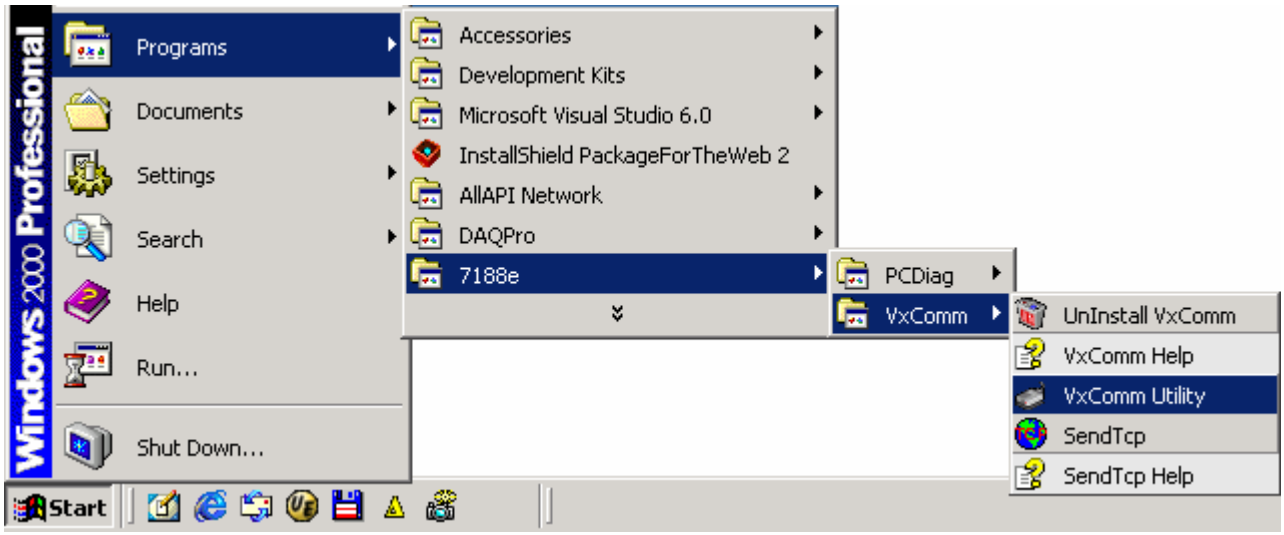


Step 4: After rebooting the computer, the VxComm Utility will ask you to configure the virtual COM port(s). Please refer to the next section for more information.

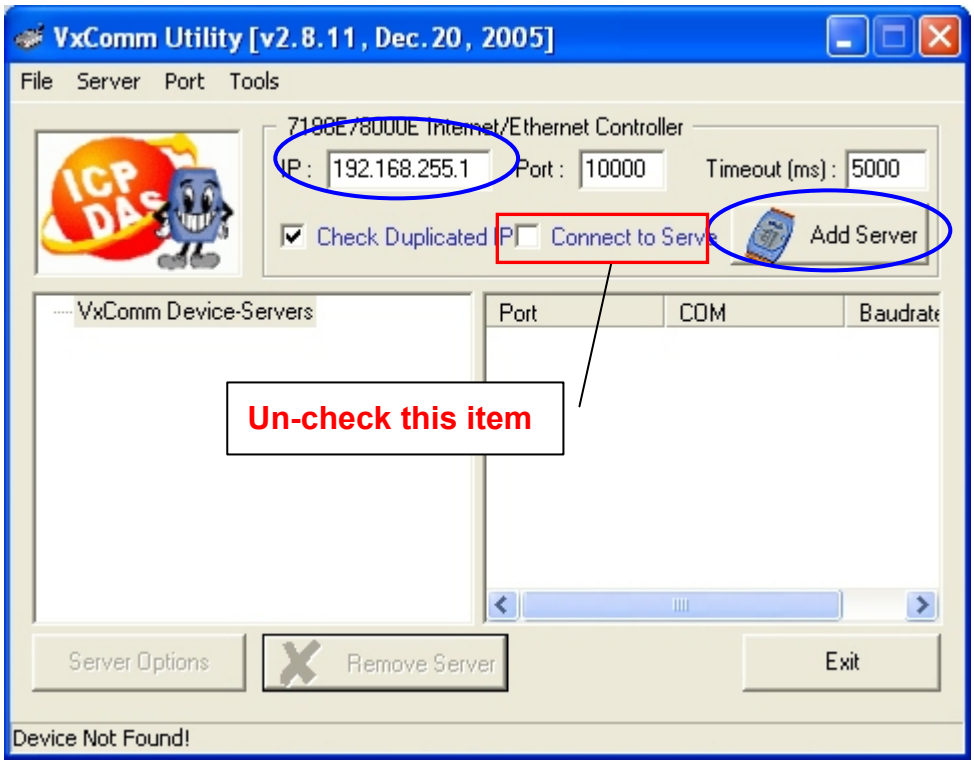
7.3 Adding an GW-7434D and configuring the VxComm Driver

GW-7434D's default IP address is 192.168.255.1.

Step 1: Select the "VxComm Utility".



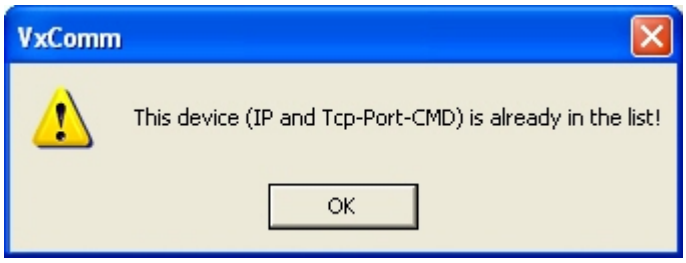
Step 2: Add an GW-7434DD server IP address and Press the "Add Server" Button.



Note:

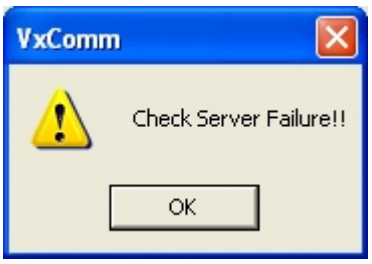
"Check Duplicate" option:

Checks whether the IP address is already listed in the server window (left-hand window). Default is automatically checked. The following window pops up if IP address is duplicated.

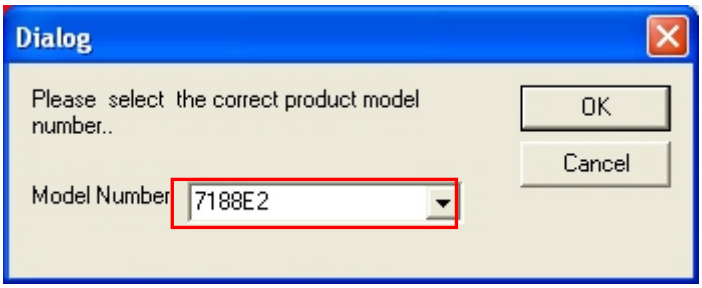


"Check Server" option:

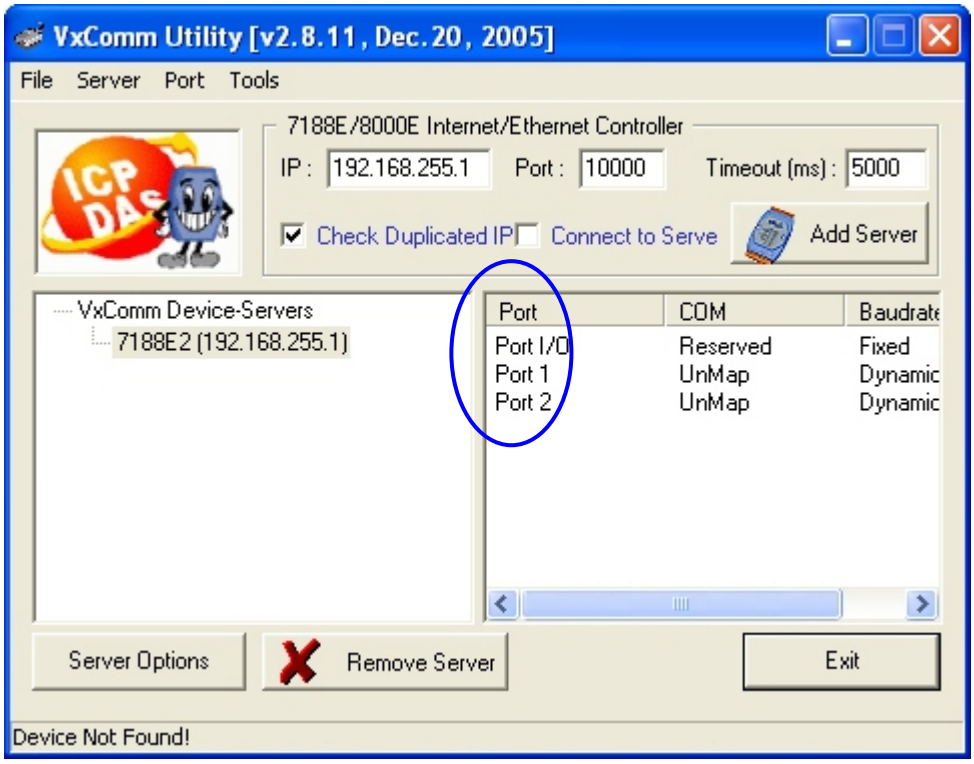
Connects to the GW-7434D and gets the device's name before adding to the server window (left-hand window). Default is automatically checked. The following window pops up if the host fails to connect.



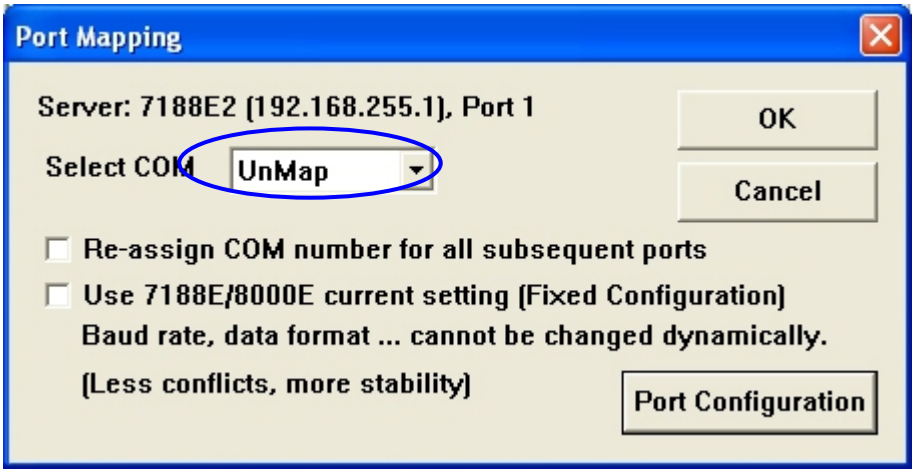
Step 3: Then a “Dialog” window will be popped up. Then select the module Number: “7188E2” to add the GW-7434D module into the server window.



Step 4: And then it will display the name of “7188E2” and it’s IP on the server window. Select one of the GW-7434D devices and configure the virtual COM port(s) by double clicking "Port1", "Port2" or etc.



Step 5: Double click an appropriate COM port number, and then it will pop up a “Port Mapping” window. Then select the virtual COM port of PC on the “Select COM” dialogue.



Note:

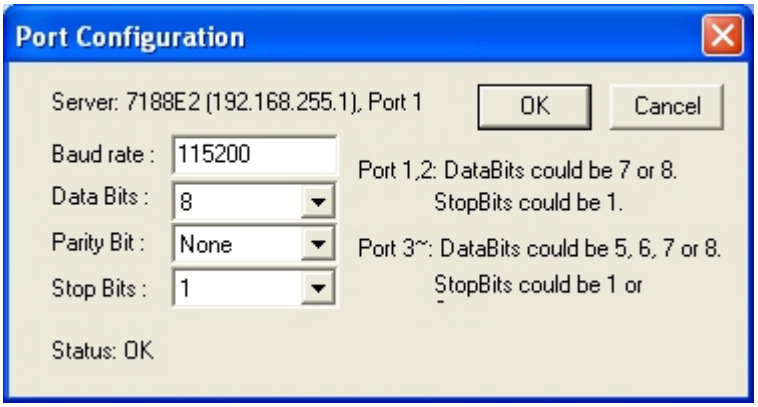
"Re-Assign COM number for all subsequent ports" option:

This option will assign the following ports with the available COM port number sequentially and automatically.

For example: If Port1 = COM4, then Port2 = COM5, Port3 = COM6

"Use 7188E/8000E current setting (Fixed Configuration)" option:

You can click the "Port Configuration" button to configure this port's Baud rate and Data format settings. After the configuration, you can then check this option.



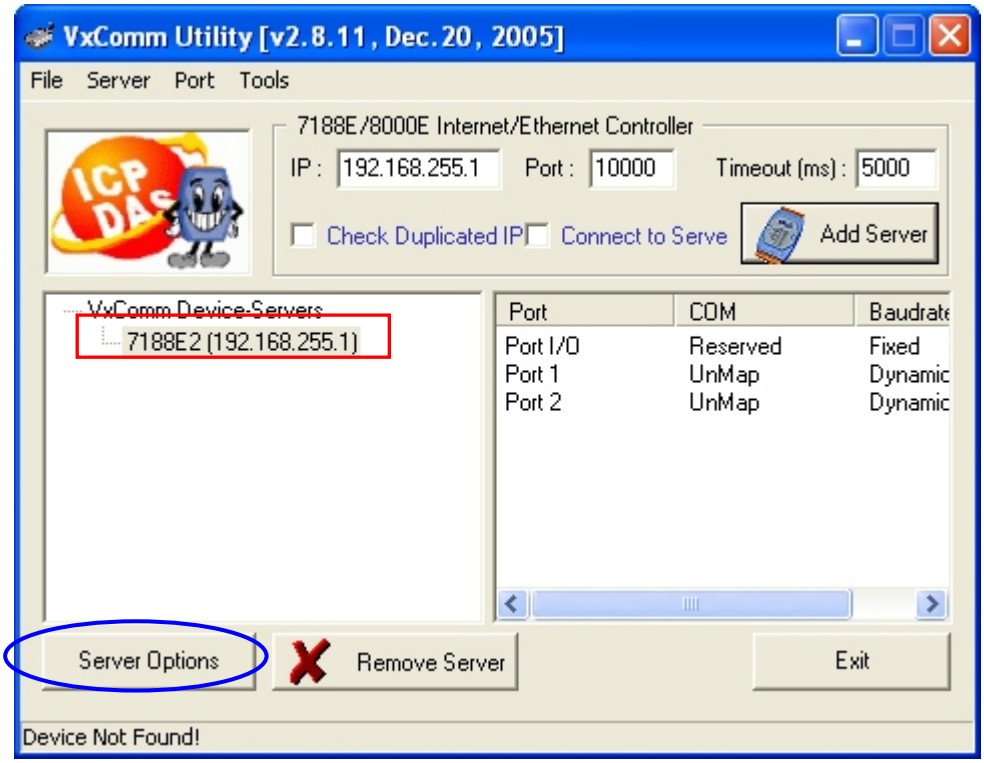
Fixed Configuration (Use GW-7434D's current setting):

By using this feature, the VxComm Driver would not change the GW-7434D's Com port settings dynamically. This is proper when you have multI-clients to access the same GW-7434D server.

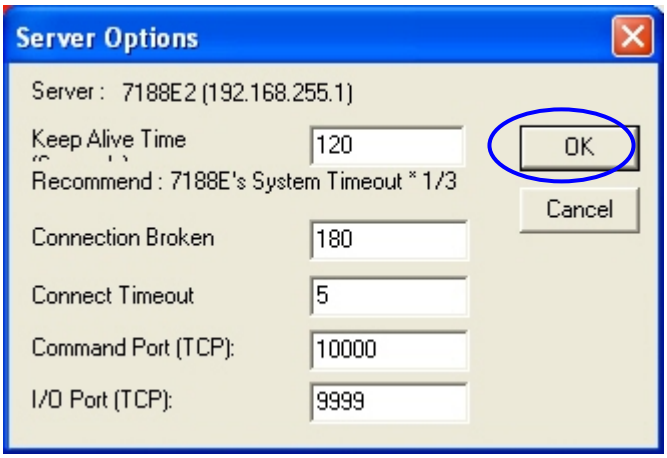
Dynamic Configuration (It's the default method):

The VxComm Driver always change the GW-7434D's settings dynamically. It is proper for working with several different baud rate and data format.

Step 6: Select one of the 7188E/8000E devices and then click the "Server Options" button to configure the options.



Step 7: Key in the value and then click the "OK" button to exit.



Notes:

Keep Alive Time (s) field:

After connecting to the GW-7434D, the VxComm Driver will automatically and periodically sends commands to keep the GW-7434D alive. The timer will be reset after each sending command or receiving data success. The Keep-Alive mechanism won't work until the next timeout.

The default setting of Keep-Alive time is about 7 second. It's recommended setting is (GW-7434D's System Timeout * 1 / 3) or a smaller value.

Connection-Broken (s) field:

The VxComm Driver will try to build a new connection when the connection is broken. When clients send a message to the GW-7434D, the Internet (TCP/IP) layer may respond with a "Disconnect" event to the VxComm Driver if it sends the message failure in 20 seconds or later. Users can set a shorter Connection-Broken timeout [for example: 10 seconds)] to force the VxComm Driver to build the connection again and get a quicker response.

If there is no sending/receiving signal in the connection during the Connection-Broken time, the connection will be marked as broken. The VxComm Driver will build the connection again in Connection-Broken time. Thus, the Keep-Alive Time should be shorter than the Connection-Broken time to make the connection come on-line.

The default System Timeout (/STxxx) value of the GW-7434D is about 300 seconds. After client programs connect to the GW-7434D, the clients must send commands to keep the GW-7434D alive before timeout is up, otherwise the GW-7434D will reset itself and clients must build the connection again.

You can disable the Keep-Alive Time and the Connection-Broken mechanisms by setting their value to 0.

Connect Timeout (ms) field:

The timeout value will be passed into MS TCP/IP driver for reference when connecting and disconnecting.

Command TCP Port field:

By default setting, the GW-7434D use TCP port 10000 as the Command/Configuration port of its COM1 and COM2. If you change it's setting, you must assign the correct one in the field. So the VxComm Driver can connect to the right TCP port.

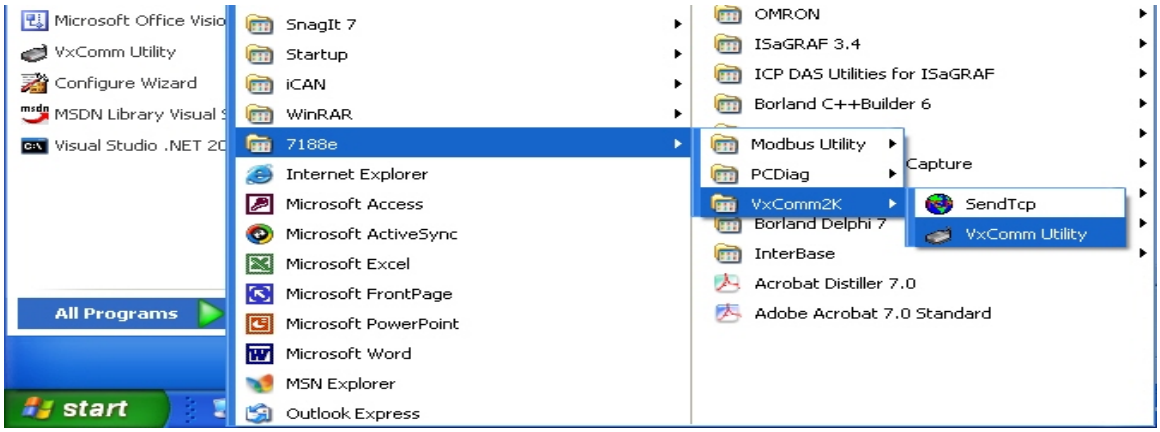
I/O Port field:

By default setting, the GW-7434D use TCP port 9999 as the Port7000 port. Now, this TCP port is reserved.

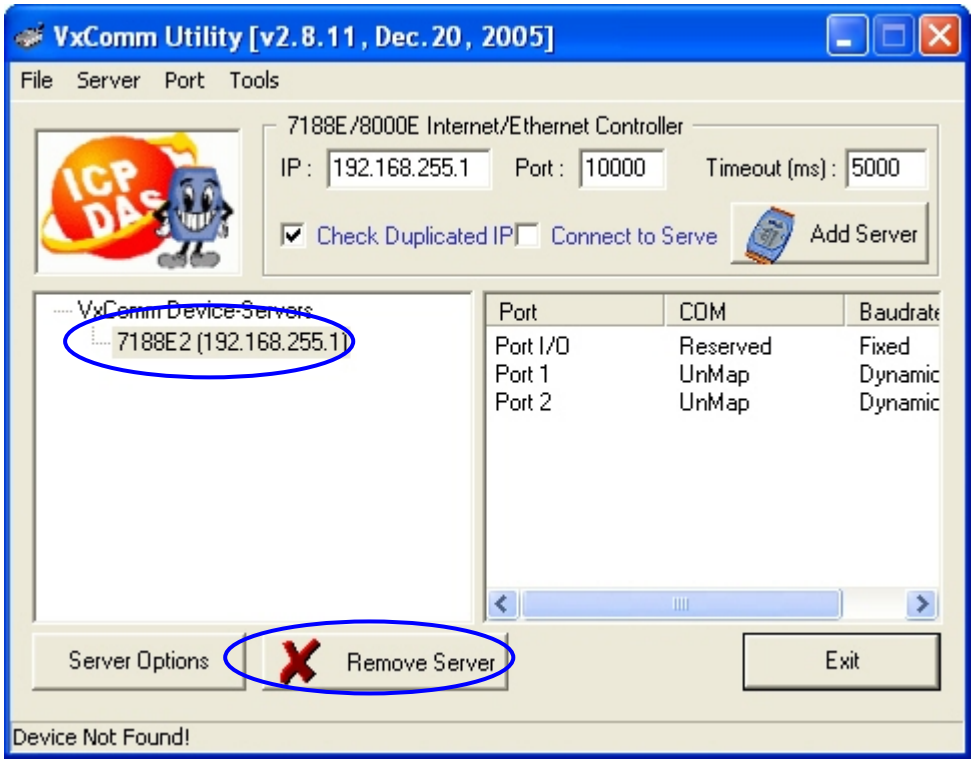
Step 8: Press the "OK" button to save the settings and exit the VxComm Utility.

7.4 Removing an GW-7434D

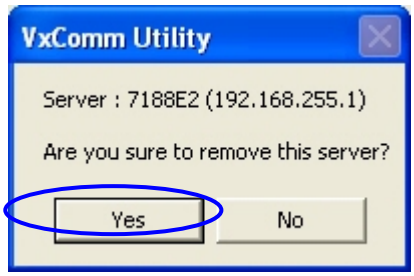
Step 1: Select the "VxComm Utility".



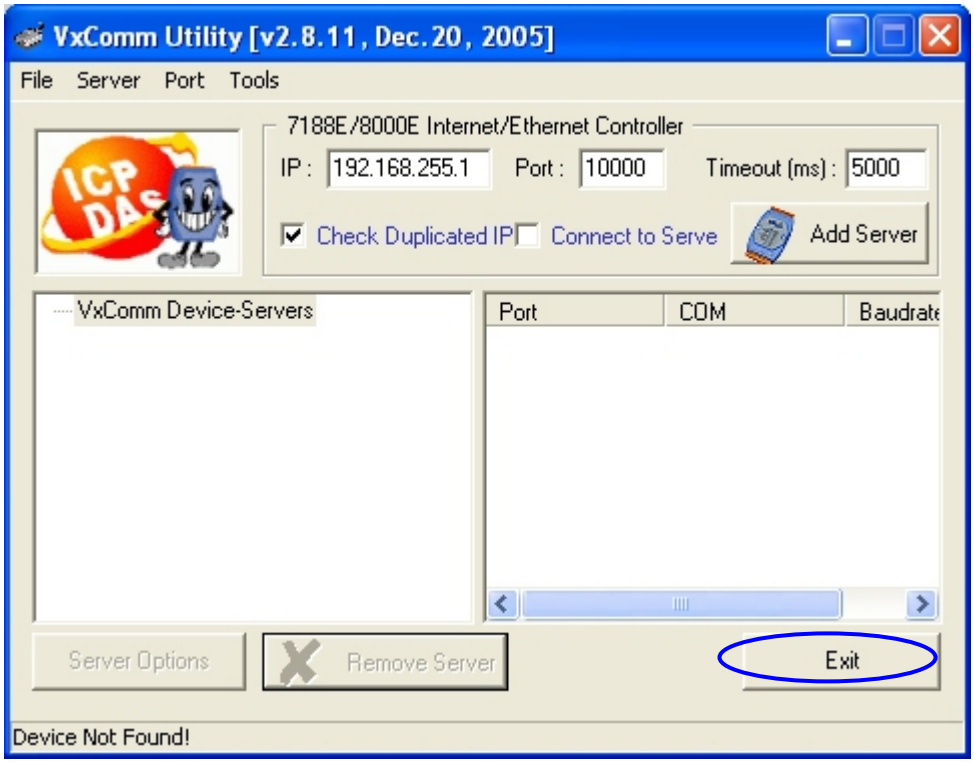
Step 2: Click the server name you want to remove and press the "Remove Server" button.



Step 3: The following window will pop up, please make sure of your choice and press the "Yes" button to remove it.

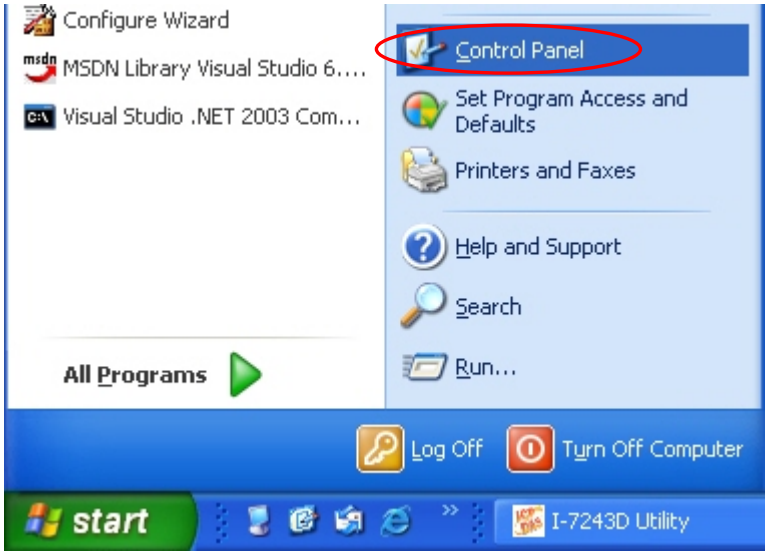


Step 4: Press the "Exit" button to finish this utility.

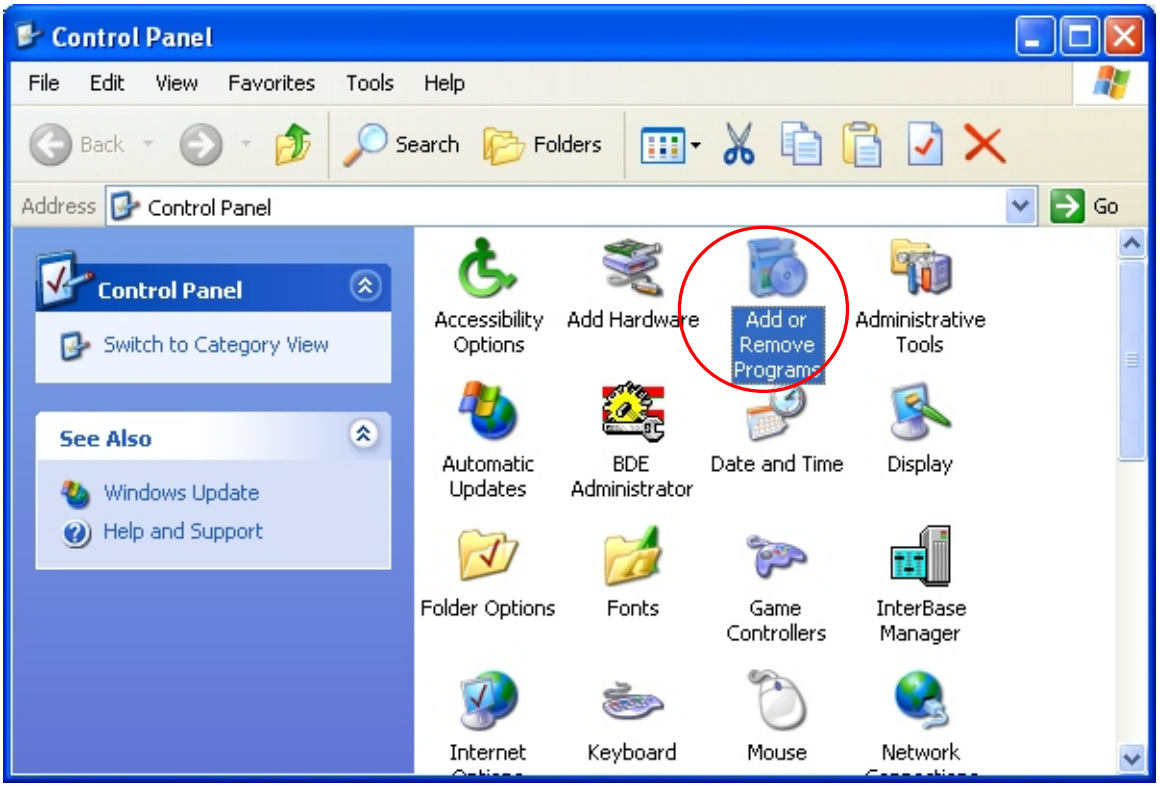


7.5 Uninstalling the VxComm Driver

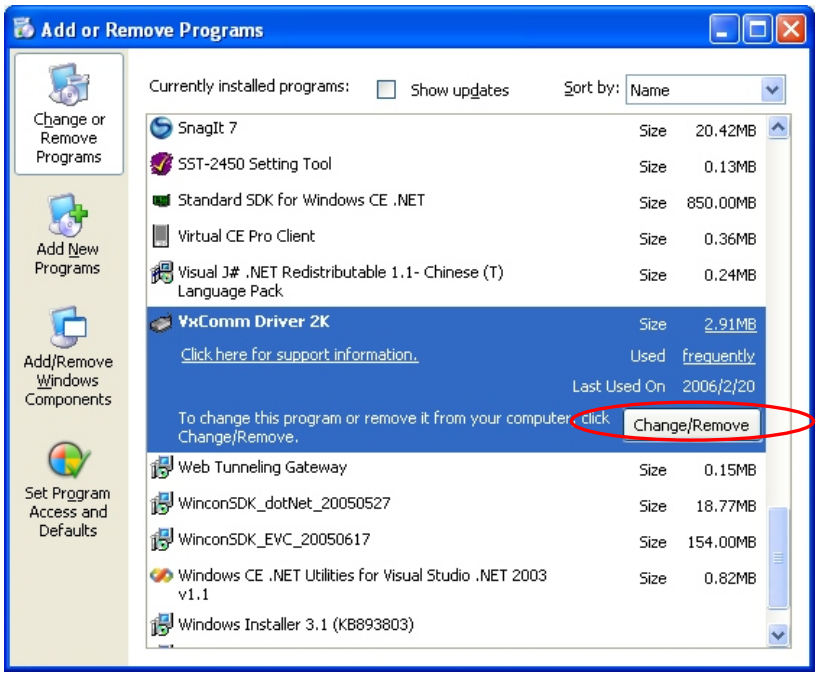
Step 1: Select the "Control Panel".



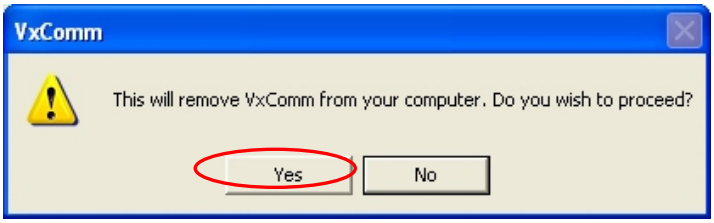
Step 2: Click the "Add or Remove Programs" to open the dialog.



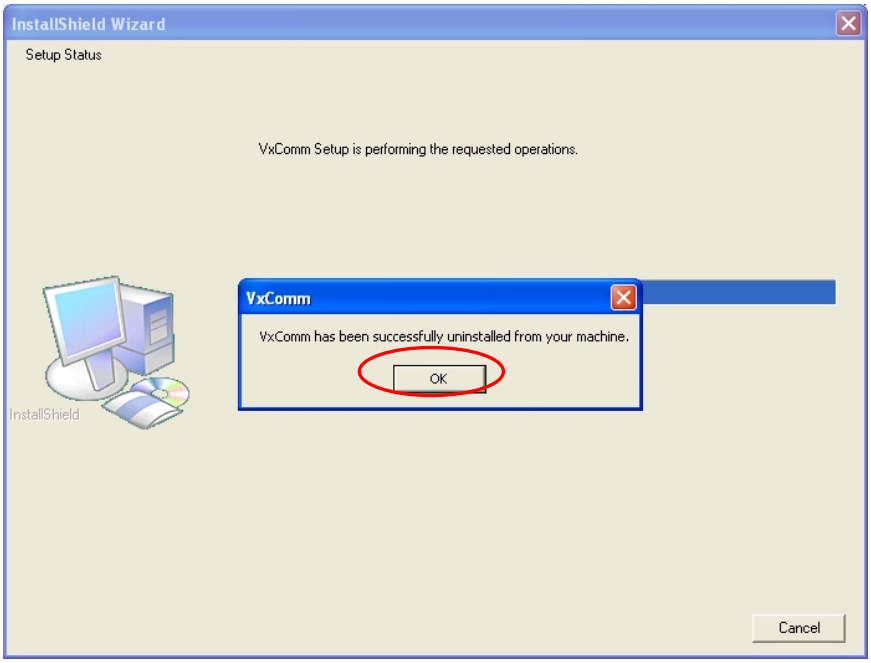
Step 3: Find out the VxComm Driver, and click the “Change/Remove” button.



Step 4: Click the button “Yes” to remove the software.



Step 5: Finally, click the “OK” button to finish the uninstall process.



8. Diagnostics and Troubleshooting

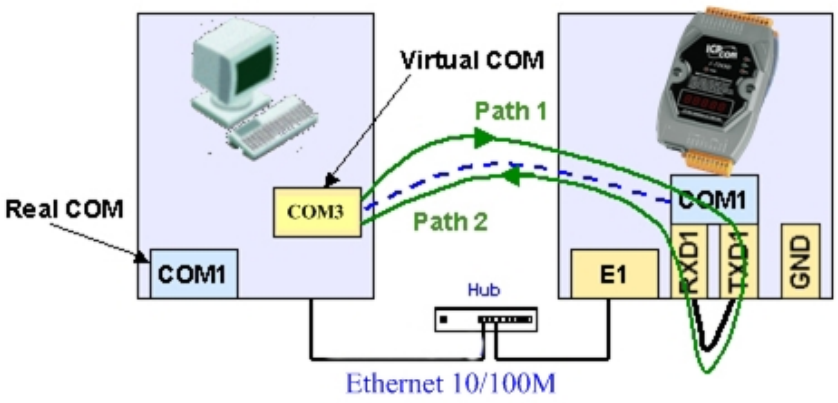
8.1 Diagnostics

After configuring the VxComm Driver by using the VxComm Utility, the VxComm Driver should work without error. However, users can use a simple test to make sure it's working properly.

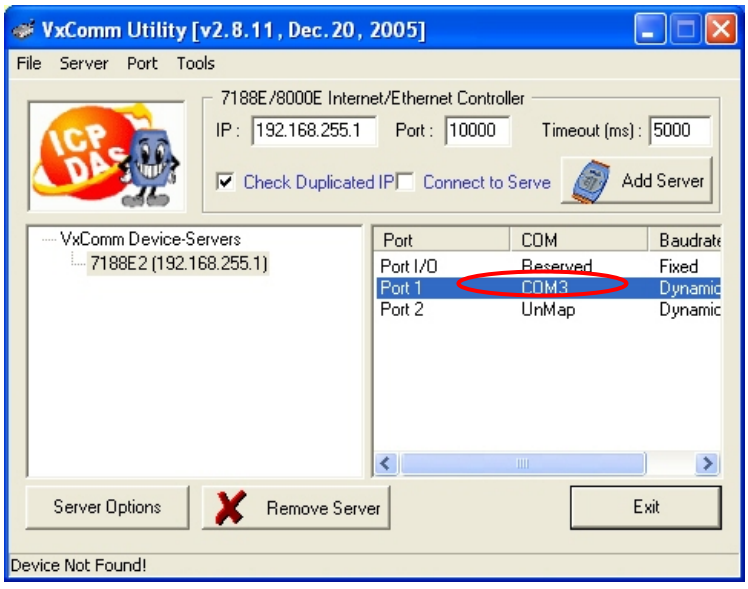
Note: The test method depends on the user's devices and client programs. And users must install the PC-Dialog tool on the CD: 8000CD:\Napdos\7188e\TCP\PCDiag\Setup\Setup.exe

■ Example 1: Loop-Back Testing

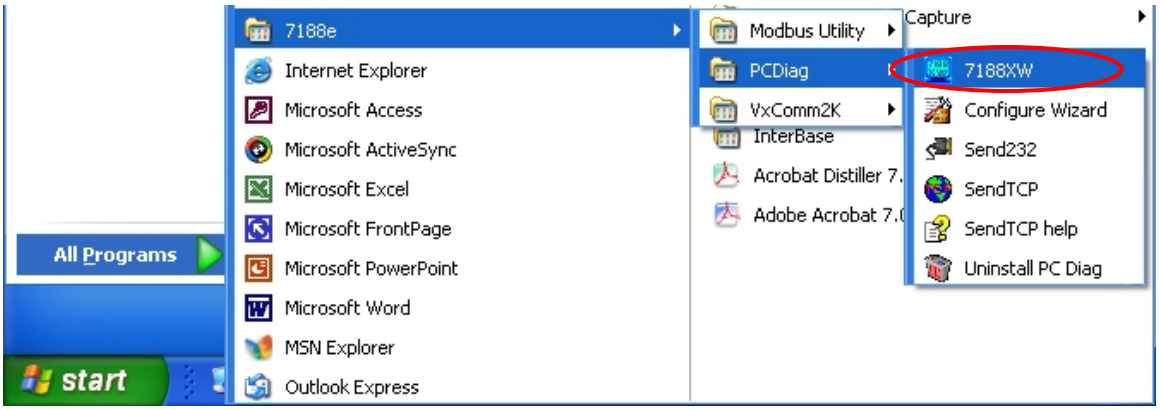
Step 1: Wire the TXD1 with the RXD1 (COM1) of the GW-7434D.



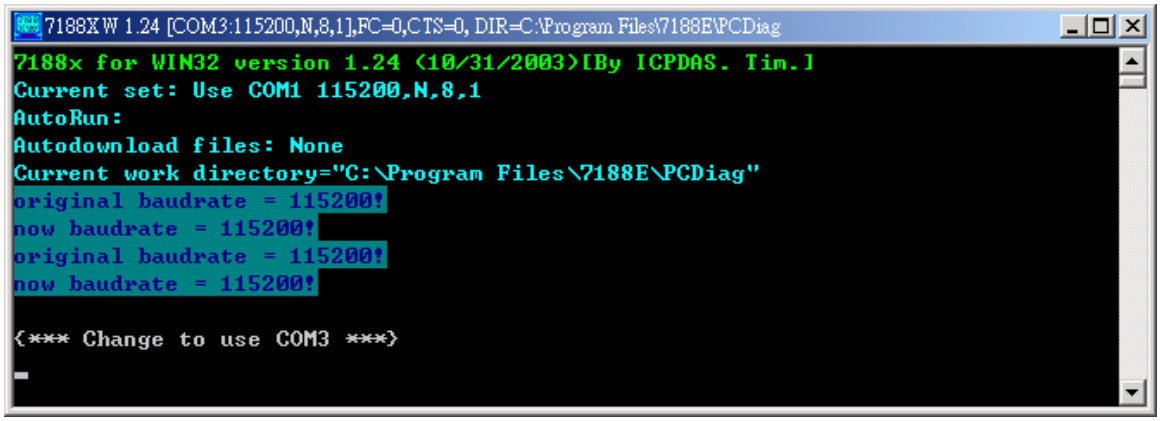
Step 2: GW-7434D's COM1(RS-232 Port) to become PC's COM3 by using the VxComm Utility.



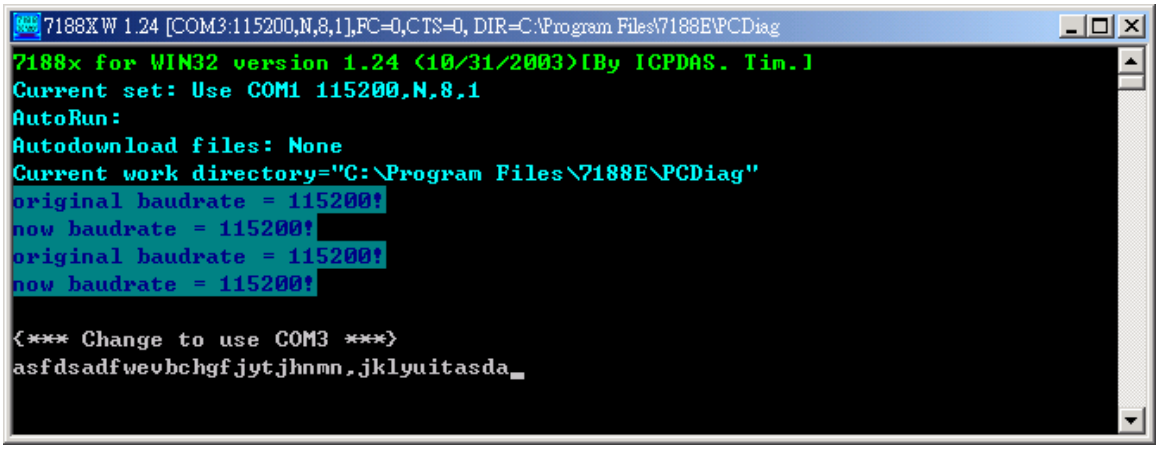
Step 3: Run the 7188xw.exe from the "Start / Run..." menu.



Step 4: Press the <Alt> + <3> keys to use PC's COM3.
It will show "{*** Change to use COM3 ***}" message after changed.

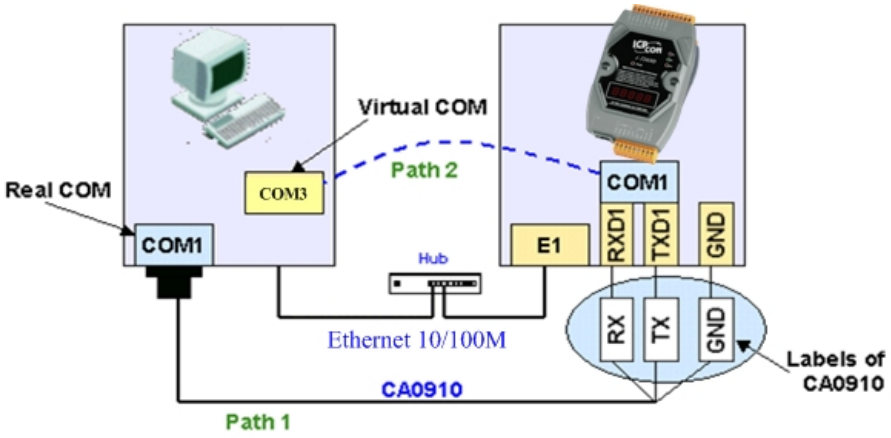


Step 5: Type some characters in the 7188xw.exe window.
The characters will be sent from PC's COM3 to GW-7434D's COM1 (through Path1), and immediately returned from the GW-7434D's COM1 to the PC's COM3 (through Path2) then shown on the PC's monitor.

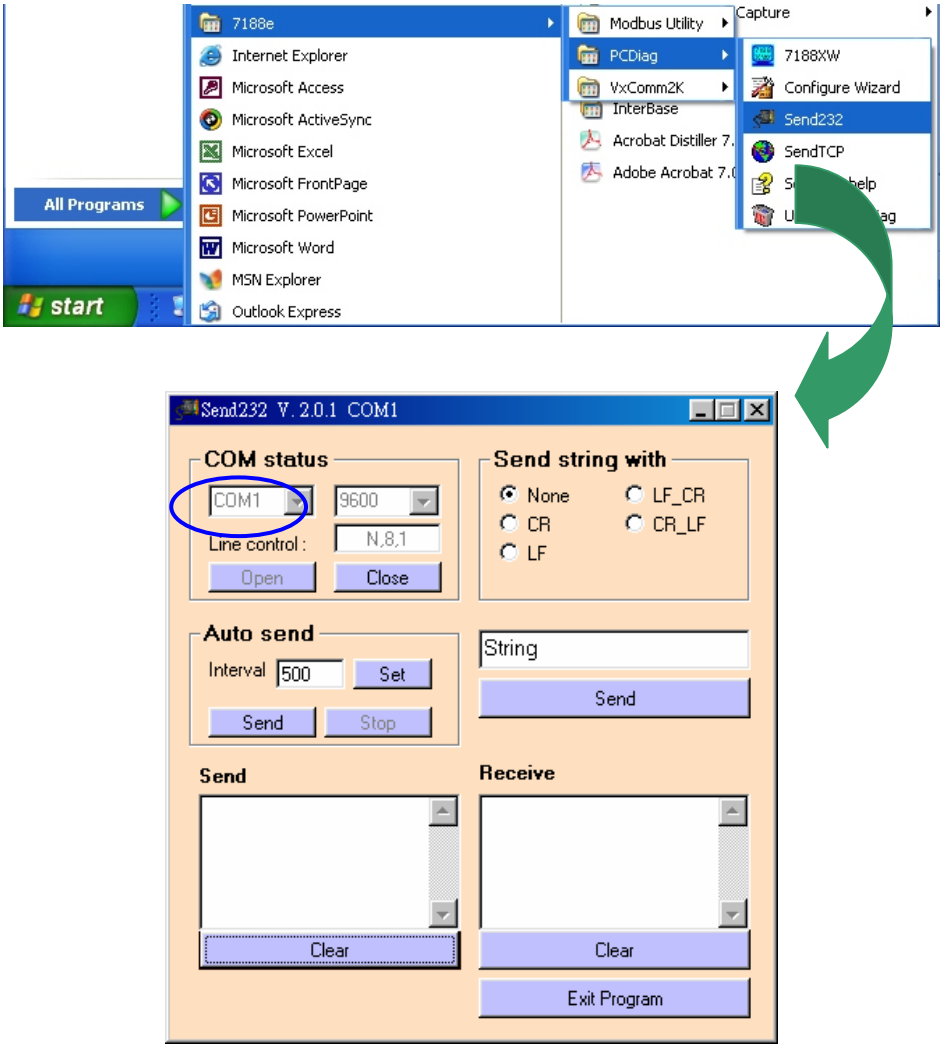


■ Example 2: Close-Loop Testing

Step 1: Build connection as follows:



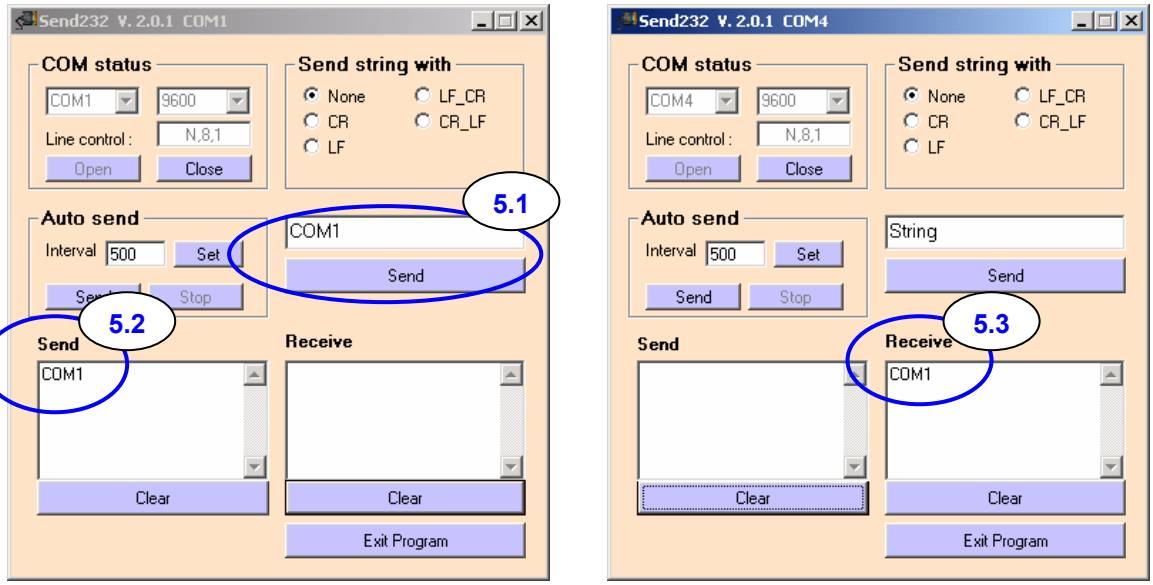
Step 2: Run Send232 and then open PC's COM1.



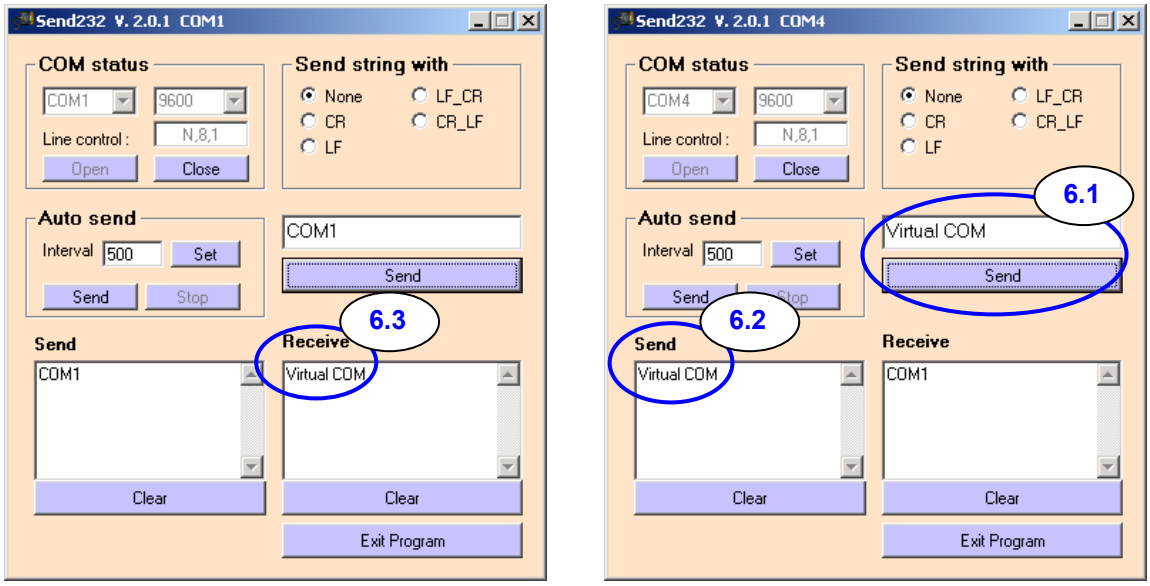
Step 3: Let GW-7434D's COM1 to become PC's COM4 by using the VxComm Utility.

Step 4: Run another Send232 and open PC's virtual COM3.

Step 5: Type "COM1" in left hand window, and press "Send".
Data will be sent from PC's COM1 through Path1 to GW-7434D's COM1 and immediately returned through Path2 to PC's COM4.



Step 6: Type "Virtual COM" in right hand window, and press "Send".
Data will be sent from PC's COM4 through Path2 to GW-7434D's COM1 and immediately returned through Path1 to PC's COM1.



8.2 Troubleshooting

Problem 1: Client program fail to open the COM port that was created by the VxComm Driver.

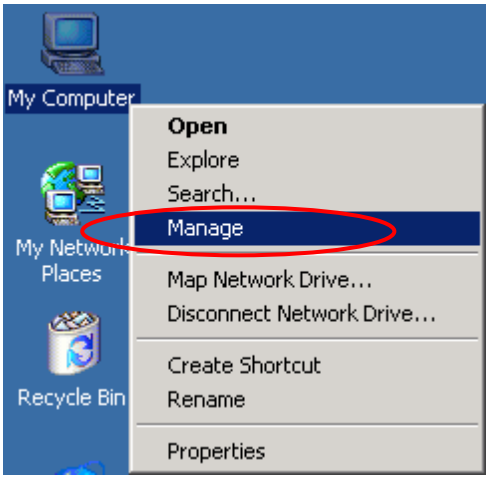
Check:

The GW-7434D's power supply, network cable, IP address, subnet-mask and gateway.

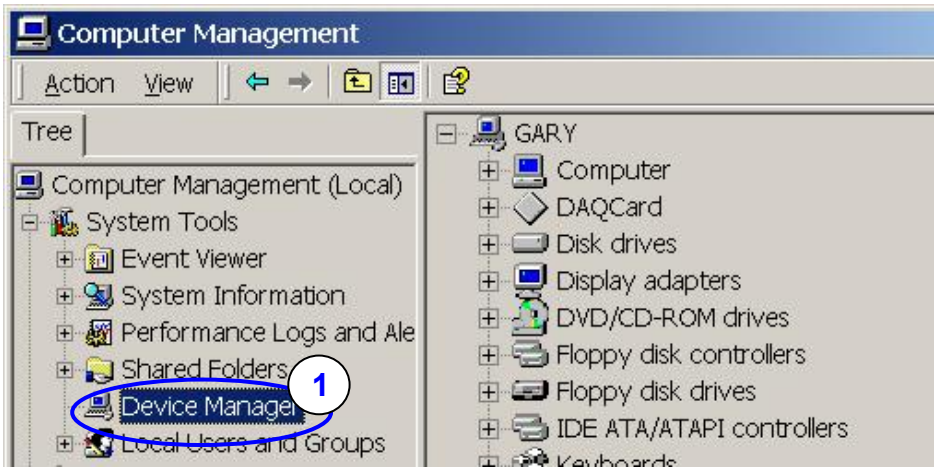
Problem 2: Client program still fails to open the COM port.

Check:

Step 1: Right click the "My computer" icon and select the "Manage" option.

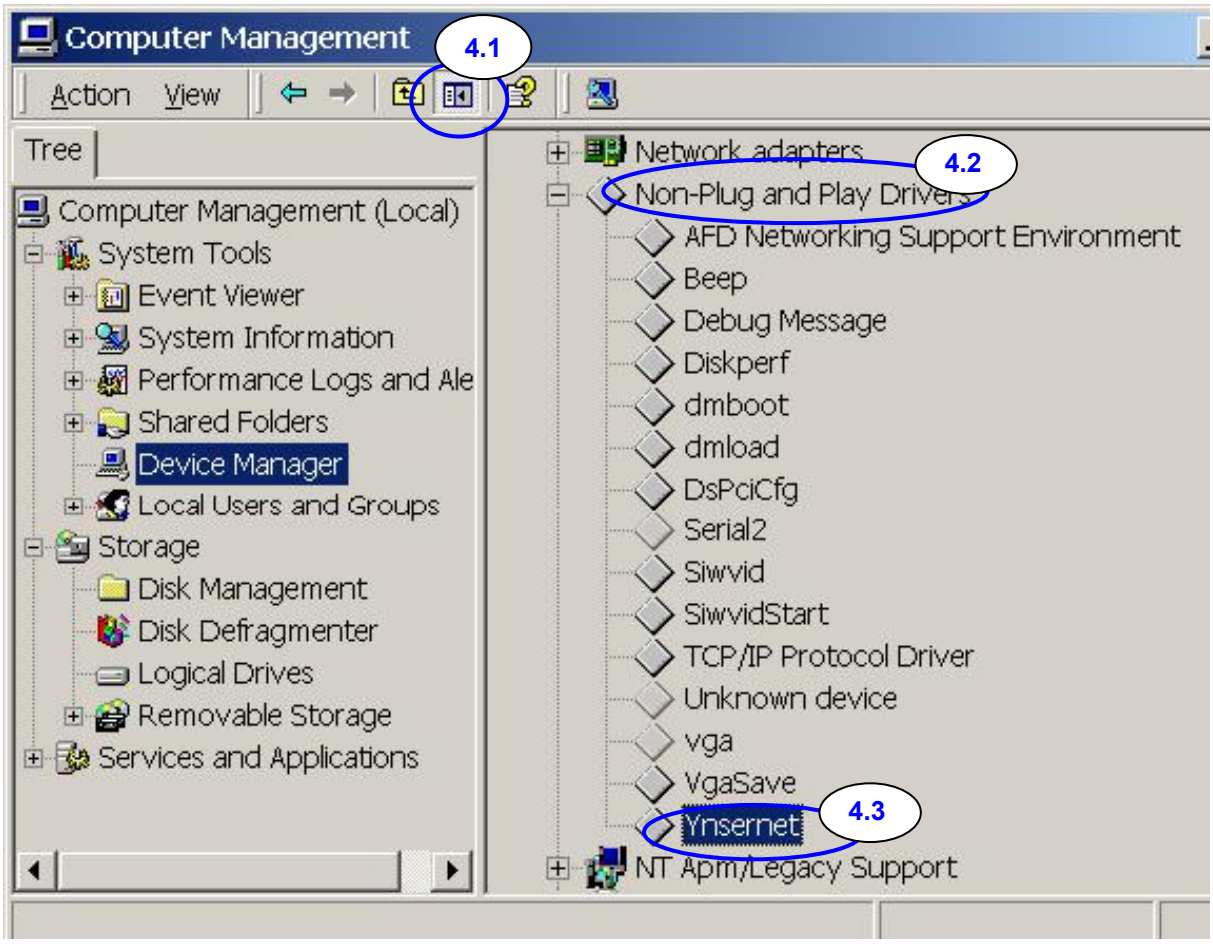


Step 2: Select the "Device Manager" icon from the "Computer Management" program.

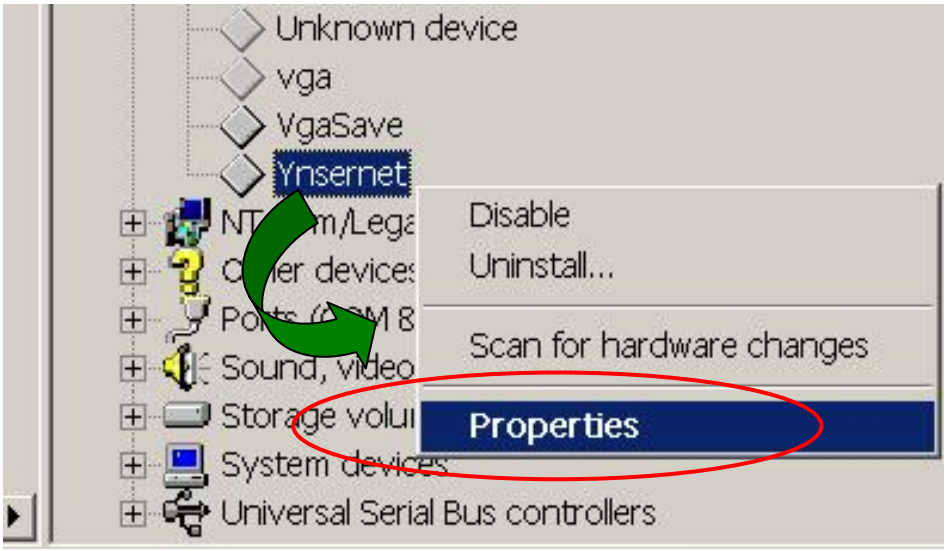


Step 3: Click the menu item "View / Show hidden devices".

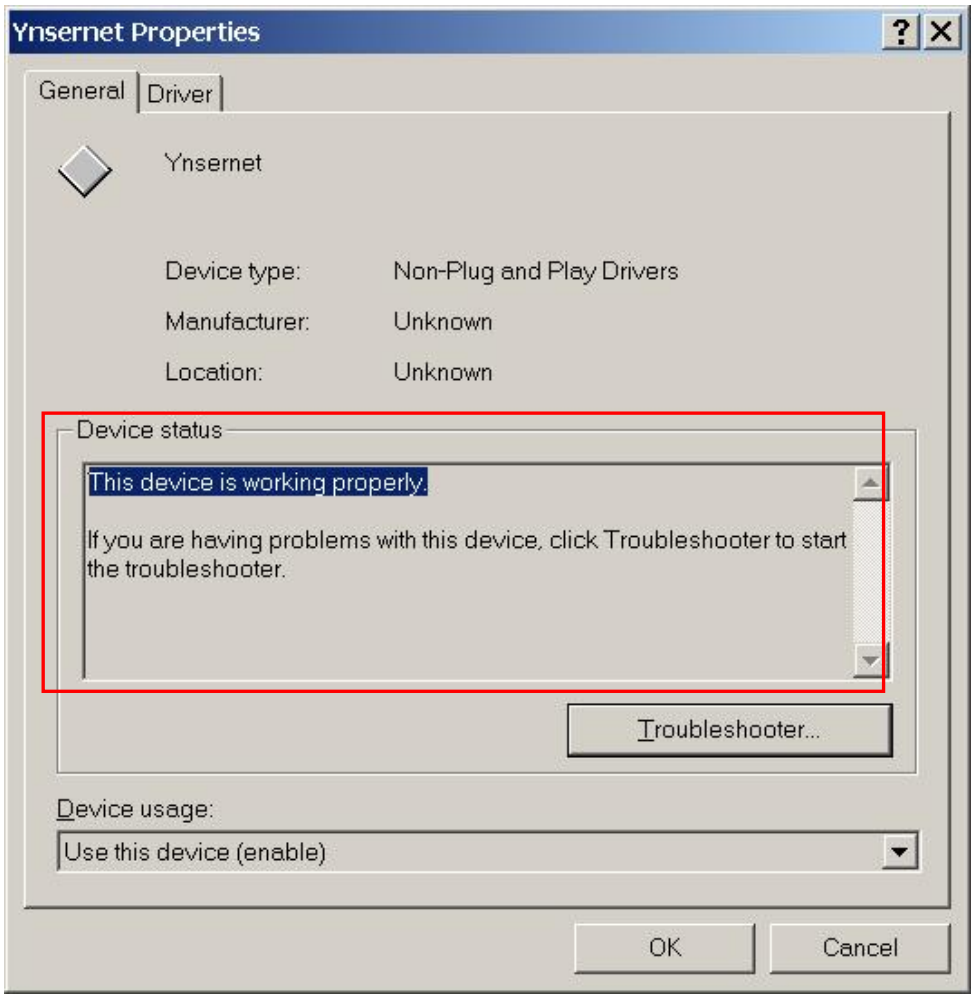
Step 4: Select the item "Non-Plug and Play Drivers / Ynsernet".



Step 5: Right click the mouse button on the "Ynsernet" item and select the "Properties" menu item.



Step 6: Check if it shows the message "This device is working properly."
If the driver does not work properly, please remove it and then re-install and configure it again.



Problem 3: Client programs open the COM port with success, but fail to access the device.

Check:

Check the device's power supply and wiring (RS-232: RXD, TXD; RS-485: D+, D- ; GND ...).

Appendix A: Connection Status Table

Interpretation of the return code (General Error)

Return Code	Error ID	Comment
5000	DNMXS_UnKnowError	The DeviceNet has some unknown errors.
1000	DNMXS_BoardNotActive	The GW-7434D has not been activated.
1001	DNMXS_OnlineError	The master MAC ID collides with other slave device in the DeviceNet network.
1002	DNMXS_CANBusError	The CAN port can't send message. Please check the baud rate or the port of the CAN bus.
1003	DNMXS_Booting	The GW-7434D is still booting.
1050	DNMXS_MACIDError	The MAC ID is exceed the range(0 ~ 63)
1051	DNMXS_BaudRateError	The baud rate is exceed the range(0 ~ 2)
1052	DNMXS_ConnectionTypeError	The connection type is exceed the range (0 ~ 4)
1053	DNMXS_DuplicMasterMACID	The MAC ID is the same with the master's ID.
1054	DNMXS_EEPROMError	The EEPROM is out of order.
1055	DNMXS_NowScanning	The GW-7434D is searching the slave.
1056	DNMXS_ScanListError	The Scan List has some errors.
1057	DNMXS_DeviceExist	The information of the slave device already exists.
1058	DNMXS_DeviceNotExist	The information of the slave device doesn't exist.
1059	DNMXS_MapTableError	The MapTable has some errors.

Interpretation of the return code (I/O Error)

Return Code	MapTable Error	Comment
1100	DNMXS_ExplicitNotAllocate	The Explicit connection is not established.
1101	DNMXS_PollNotAllocate	The Poll connection is not established.
1102	DNMXS_BitStrobeNotAllocate	The Strobe connection is not established.
1103	DNMXS_COSNotAllocate	The COS connection is not established.
1104	DNMXS_CyclicNotAllocate	The Cyclic connection is not established.
1105	DNMXS_PollAlreadyExist	The Poll connection has been established.
1106	DNMXS_BitStrobeAlreadyExist	The Poll connection has been established.
1107	DNMXS_COSAlreadyExist	The COS connection has been established.
1108	DNMXS_CyclicAlreadyExist	The Cyclic connection has been established.
1109	DNMXS_CommunicationPause	The communication between GW-7434D and all slave devices has been suspended.

Interpretation of the return code (Slave Error)

Return Code	DeviceNet Error	Comment
1150	DNMXS_SlaveNoResp	The slave has no any response.
1151	DNMXS_WaitForSlaveResp	The GW-7434D is waiting for the response form the slave device.
1152	DNMXS_SlaveRespError	The slave replied some errors.
1153	DNMXS_OutputDataLenError	The output length of the I/O connection doesn't match the device's output length.
1154	DNMXS_InputDataLenError	The input length of the I/O connection doesn't match the