FCM-MTCP

Field Communication Module

Modbus/TCP and Modbus/UDP slave

Software User's Manual

Revision History

Revision	Date	Description of Change
1.04	2018/06/01	Adding description for HART send and receive. Modifying the picture in the firmware upgrade section.
1.03	2013/08/05	Modifying the content in the firmware upgrade section.
1.02	2013/06/03	Modifying the address for the counter, PWM, system information and slot information.
1.01	2013/05/23	Document release

Preface

Warranty

All products manufactured by ICP DAS are under warranty regarding defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICP DAS assumes no liability for damages resulting from the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Trademark

The names used for identification only may be registered trademarks of their respective companies.

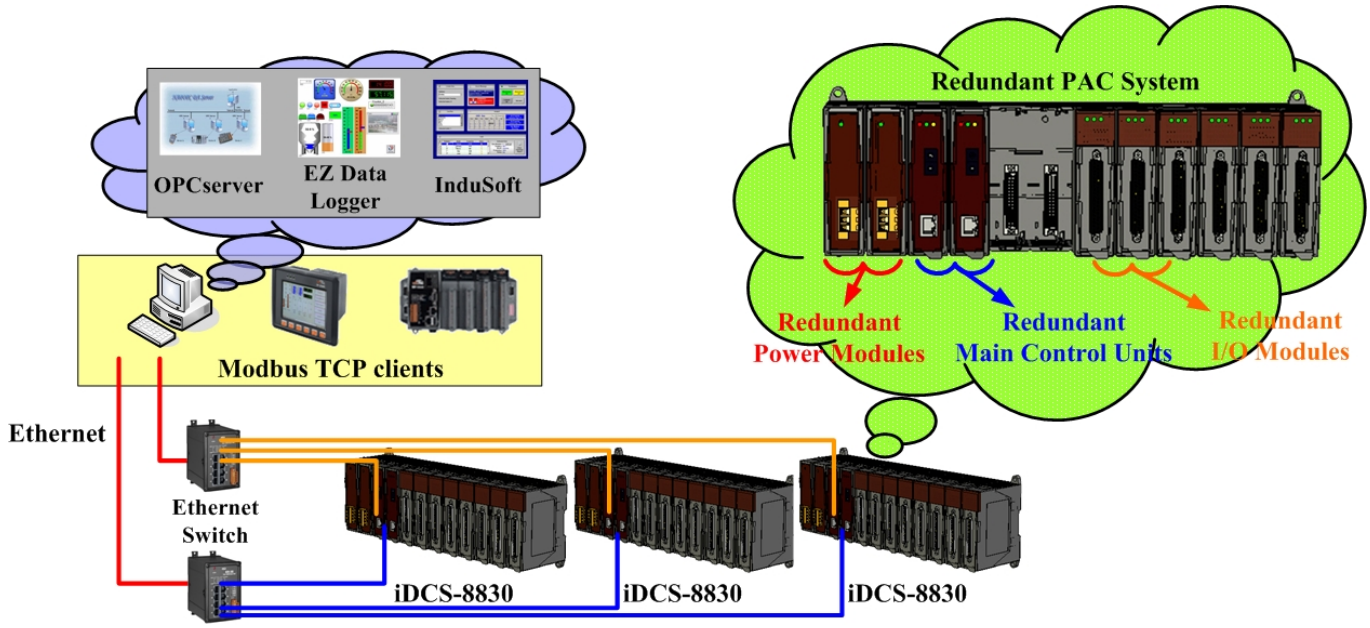
Table of Contents

Section 1 Introduction

1.1 Overview

Modbus is a well-known protocol in the industrial manufacturing and environment monitoring fields. It is suit master-slave or query-response distributed I/O applications. FCM-MTCP is a communication interface with Modbus/TCP and Modbus/UDP slave protocol. It is pre-installed a Modbus firmware and can arrange in pair or groups of I/O modules, like analog input, analog output, digital input, digital output and counter modules. Via Ethernet method, it can be used on industrial application, like remote data acquisition, factory automation, monitor system and power management…etc.

The FCM-MTCP supports redundant functionality. It allows maximum eight IO modules (four redundant IO pairs). The controllers, like PC, HMI or PLC, can communicate with FCM-MTCP to access I/O modules through Modbus/TCP.



1.2 Introduction of Modbus

- **What is Modbus?**

  Modbus is a communication protocol developed by MODICON Inc. in 1979. It's a truly open protocol and the most widely used network communication protocol in industrial automation field. SCADA and HMI software can easily integrate devices together through Modbus protocol.

- **What is Modbus/TCP and Modbus/UDP?**

  Modbus/TCP and Modbus/UDP are variants of Modbus protocol. It was developed in 1999 to allow the Internet community to access Ethernet devices.

- **What software supports Modbus?**

  Most of SCADA (Supervisor Control and Data Acuisition) and HMI softwares support Modbus. For example: Citect, ICONICS, iFIX, InduSoft, Intouch, Entivity Studio, Entivity Live, Entivity VLC, Trace Mode, Wizcon, Wonderware... etc

- **What are the benefits of using Modbus?**

  ✓ Open source, no license fees.

  ✓ Widely supported by SCADA and HMI software

  ✓ Easy to use

  ✓ Easily integrate variant devices

  ✓ Low development cost

  ✓ Wide knowledge case

- **Where to find resources about Modbus?**

  ✓ http://www.modicon.com

  The web page of the original Modbus protocol inventor, MODICON Inc

  ✓ http://www.modbus.org

  A community for Modbus

  ✓ http://www.winsite.com/bin/Info?3088

  An ActiveX freeware to communicate with equipments via Modbus (RTU/ASCII) or Modbus/TCP

  ✓ http://www.pmcrae.freeserve.co.uk/readregu.html

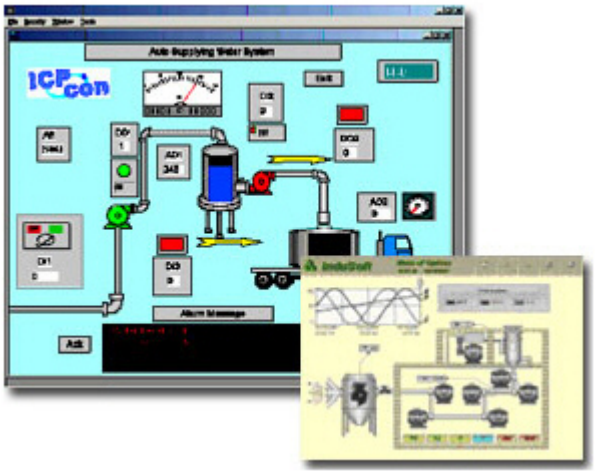
  Some examples in C code for Linux Modbus/RTU communication

1.3 Introduction of SCADA

- **What is SCADA?**

  The SCADA, known as **S**upervisor **C**ontrol **A**nd **D**ata **A**cquisition, is a production of automation and control system based on PCs. It is widely used in fields everywhere, e.g. power generation, water system, oil industry, chemistry, automobile industry. Each field requires different functions, but they all have the following features:

- Graphic interface
- Process simulation
- Real time and historic trend logging
- Alarm system
- Data acquisition and recording
- Data analysis
- Report generation



- **SCADA software accesses to hardware devices**

Methods that SCADA software uses to access to hardware devices can be easily classified into the following:

✓ **Standard communication protocol**

Common communication protocols used in the industrial field are:

ARCNET, CAN Bus, Device Net, Lon Works, Modbus, Profibus

If SCADA software and hardware devices use the same communication protocol, they can talk to each other without any extra software drivers.

✓ **Standard data exchange interface**

Common data exchange interfaces that used in the industrial field are:

DDE (Dynamic Data Exchange)

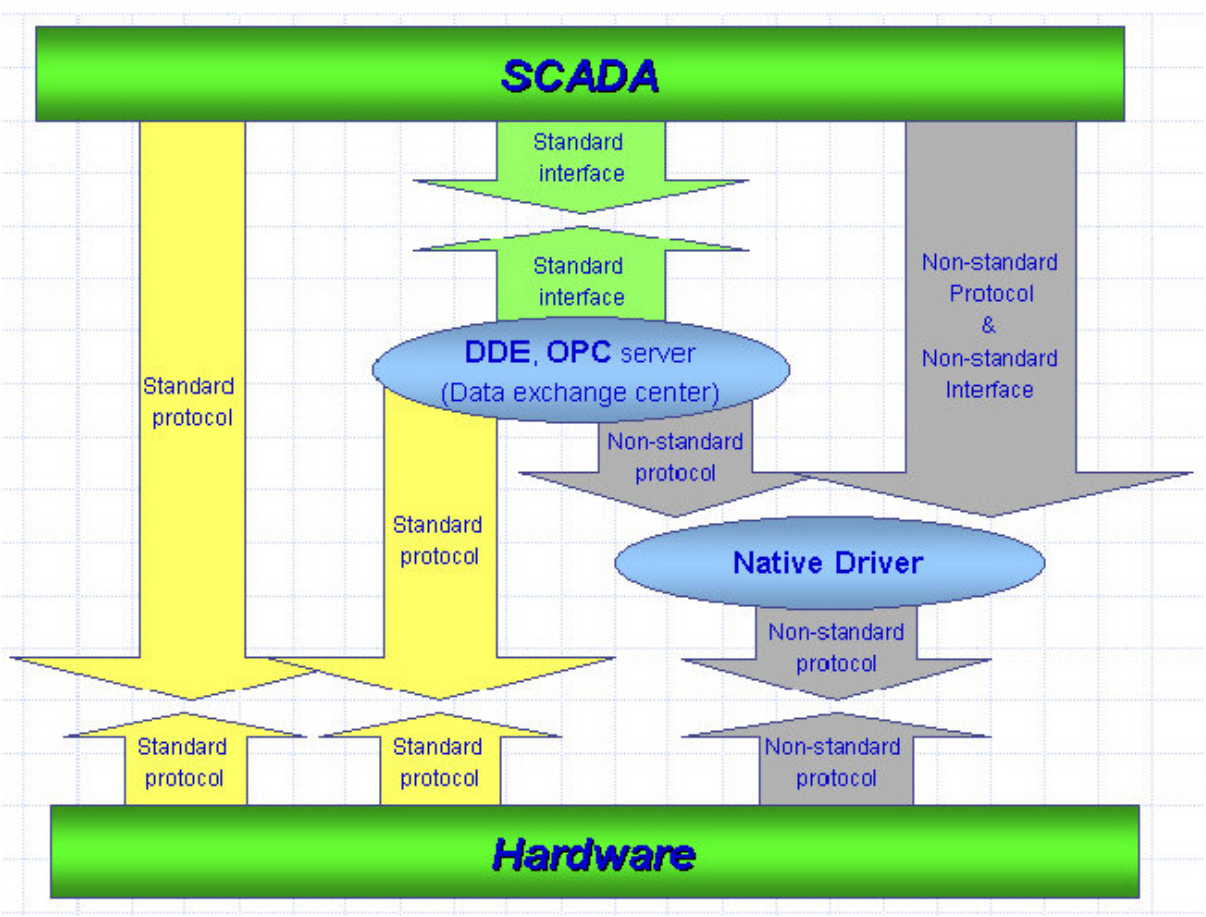
OPC (OLE for Process Control)

Using standard data exchange interfaces, SCADA software can indirectly communicate with hardware devices via DDE and OPC data exchange center. The advantage is that no matter hardware device supports standard communication protocols or not, manufacturers only need to provide one DDE or OPC driver in order to support most SCADA software.

✓ **Native driver**

A driver that has been developed for a specific device or purpose is called "Native Driver". The advantage of using a native driver is that you can improve the execution process compared to using a standard data exchange interface. The main drawback is lack of compatibility as manufacturers need to provide specific drivers for specific SCADA software.

Using one of above methods, you can combine SCADA software and hardware devices. The communication structure is as below:



- **Famous SCADA software**

  Citect, ICONICS, iFix, InduSoft, Intouch, Entivity Studio, Entivity Live, Entivity VLC, Trace Mode, Wizcon, Wonderware, ...etc.

1.4 Feature of FCM-MTCP

✓ **Access I/O through Modbus/TCP slave**

FCM-MTCP is act as a device with Modbus/TCP and Modbus/UDP slave. You can simply use the utility tool to configure the device and I/O, and then create a connection from SCADA or HMI software to the FCM-MTCP.

✓ **Hot-swappable**

It is not necessary to shut down the power to replace or plug I/O modules. It brings a benefit to the whole system which can keep operating without any interruption.

✓ **Auto-configuration**

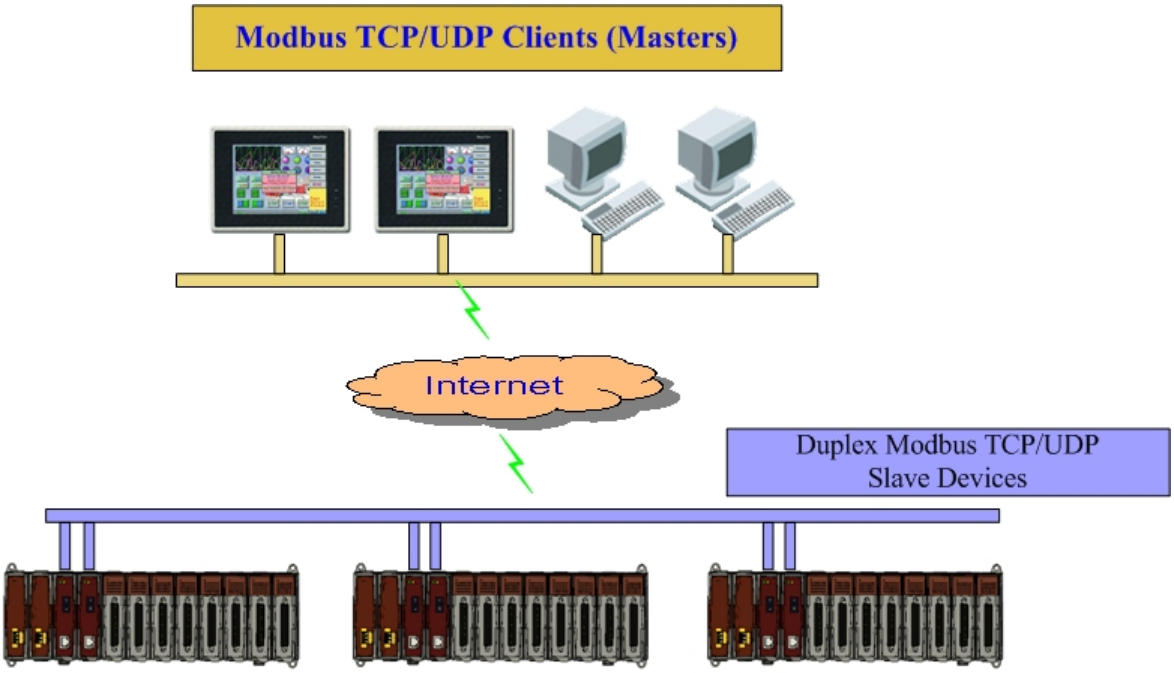
Configuration of I/O module can be pre-configured and store in the memory of the FCM-MTCP. When an I/O module is plugged, the FCM-MTCP will automatically check and restore configurations to each I/O module.

✓ **Allow multi-connection accessing**

The FCM-MTCP supports maximum 4 connections with each communication module as the same time.

1.5 System Architechture

The following figure is the system architechture of iDCS-8000 with FCM-MTCP module. The upper level likes HMI, PLC and controllers that can access I/O by connecting with FCM-MTCP with Modbus/TCP or Modbus/UDP protocol.



1.6 List of Supported Modbus Command

The slave address of FCM-MTCP is fixed to 1 in the definition of Modbus/TCP or Modbus/UDP. The following table shows the supported function codes of FCM-MTCP. For more details please refer to chapter 3.

Function Code	Modbus Command
0x01	Read coil status
0x02	Read input status
0x03	Read holding registers
0x04	Read input registers
0x05	Write single coil
0x06	Write single register
0x0F	Force multiple coils
0x10	Preset multiple registers

1.7 Modbus Address Mapping

The FCM-MTCP is a communication module with Modbus protocol. So, all of information is map into the address in the definition of Modbus. The detail of these address mapping can be refered to chapter 4.

1.8 Configuration Command

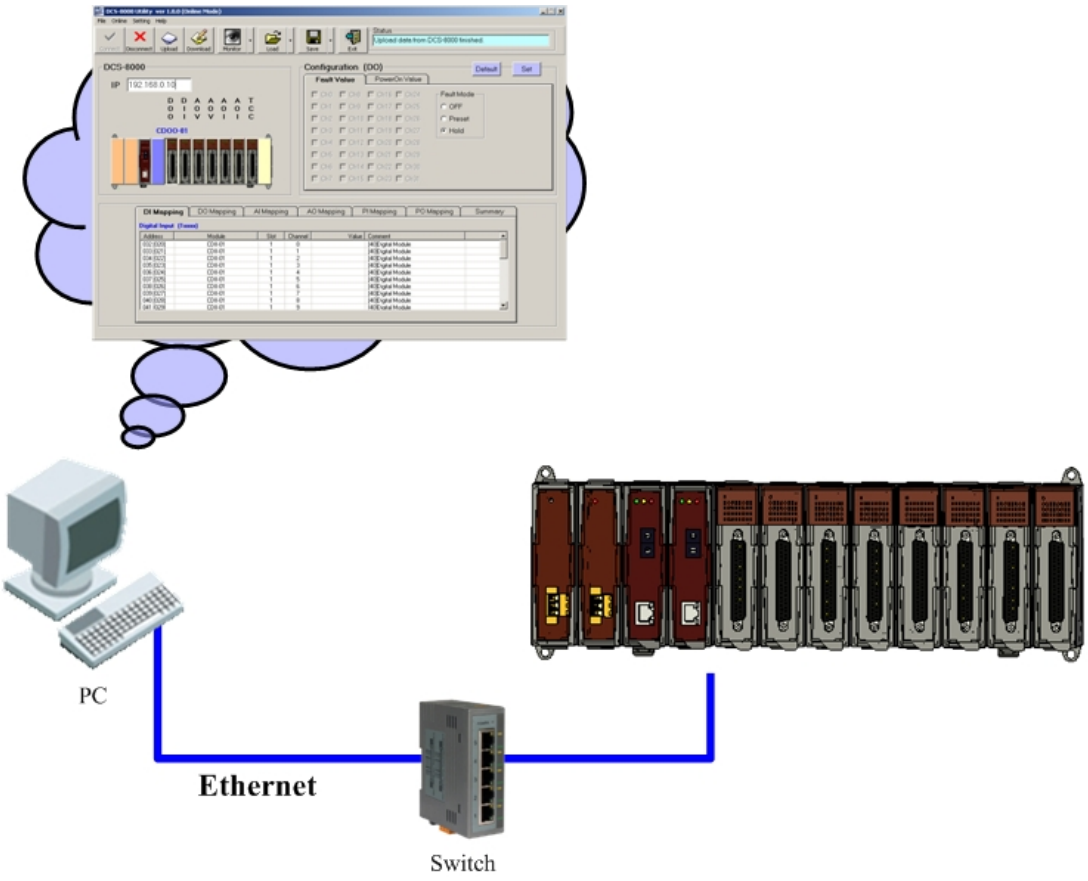
There are many configurations, like I/O range, power-on value, safety value...etc, need to be set through Modbus protocol. The FCM-MTCP provides a pre-defined configuration command to configure all of configuration. The detail of the command is described in chapter 5.

Section 2 Getting Started

The FCM-MTCP is the interface to I/O modules. All of the information is listed in Modbus address for users. Users need to write a program to access all of the information. So, it is too difficult to get all of information or configure parameter instantly. ICPDAS provides a ready-to-use and easy-to-use utililty to access all of the information for iDCS-8000 system. Uses can easily access information without programming. In this chapter, it will introduce utililty which includes, installation, instruction.

2.1 Utility Overview

The following figure is a simple connection example. This example is divided into three parts, one is the Windows based PC installed utility, one is the Ethernet switch, and the other is iDCS-8000 plugged FCM-MTCP. Therefore, the utility can access iDCS-8000.



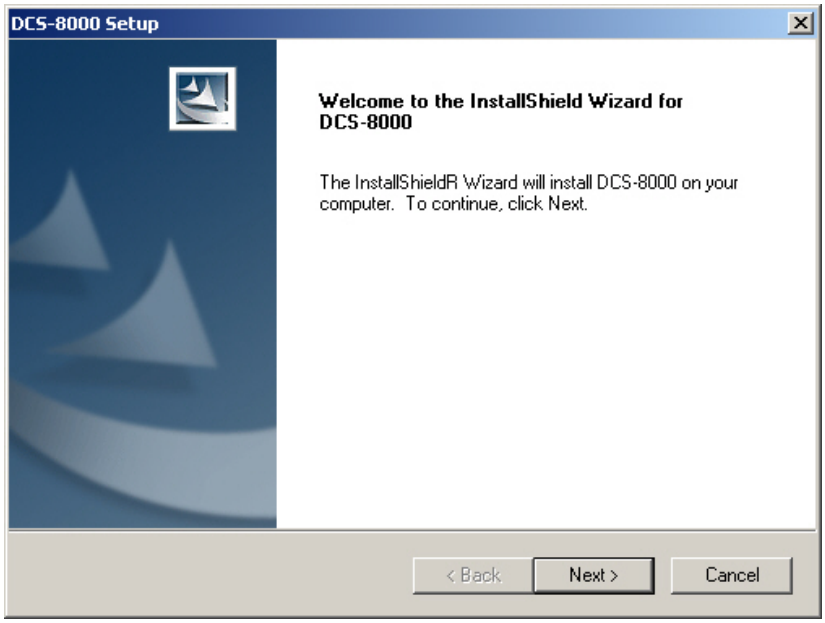
2.2 Utility Installation and Uninstallation

2.2.1 Utility Intallation

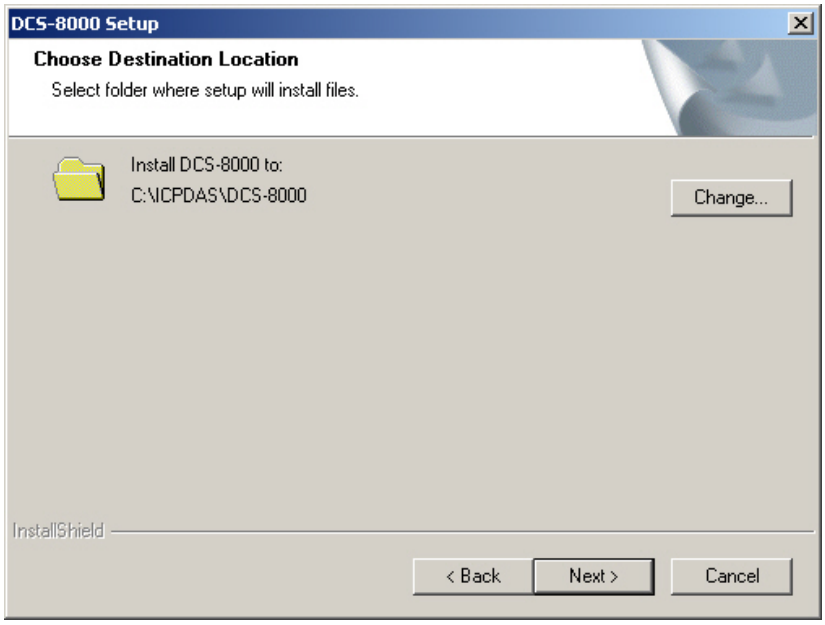
Step 1：Download the iDCS-8000 Utility setup file from the web site or from the CD-ROM disk

Step 2：Execute the setup.exe file to install iDCS-8000 Utility.

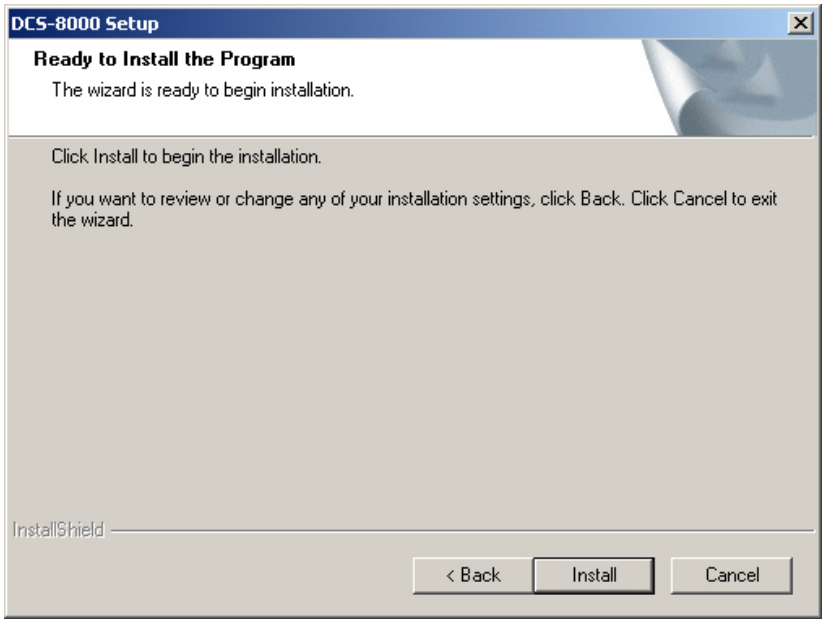
Step 3：A "Welcome" window will pop up to prompt user to begin the installation. Click the "Next" button to go on the next step.



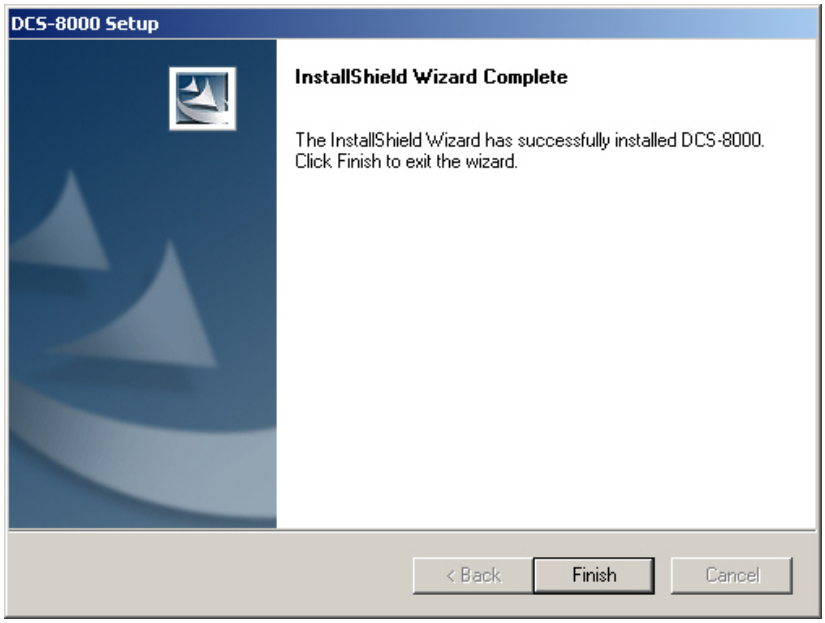
Step 4：A "Choose Destination Location" window will pop up for the installation path. Here, default path is used. Click "Next" button to go on the next step.



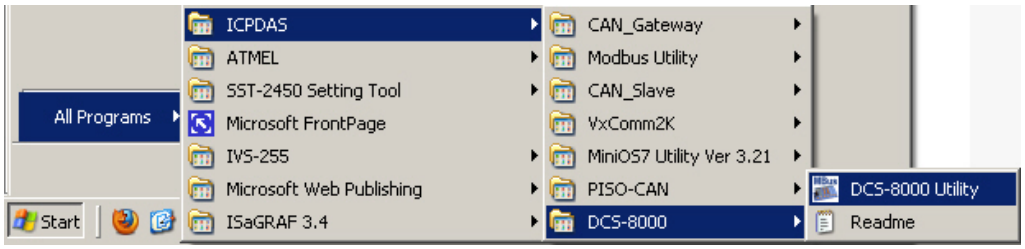
Step 5：A "Ready to Install the Program" is popped up. Click "Install" button.



Step 6：After finishing the process, a "Complete" window will pop up to prompt users that the successful completion of the installation. Then, click "Finish" button to exit.



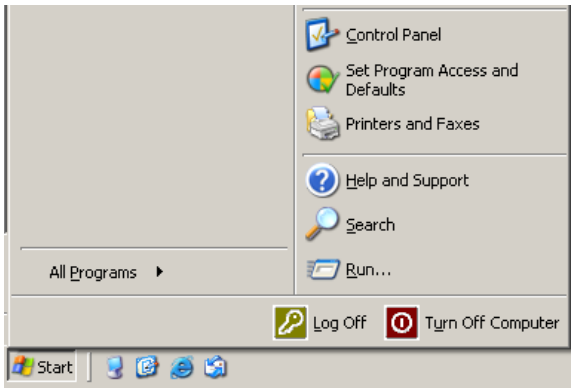
Step 7：After finishing the installation of the iDCS-8000 Utility, users can find iDCS-8000 Utility in the "Start menu" as the following figure.



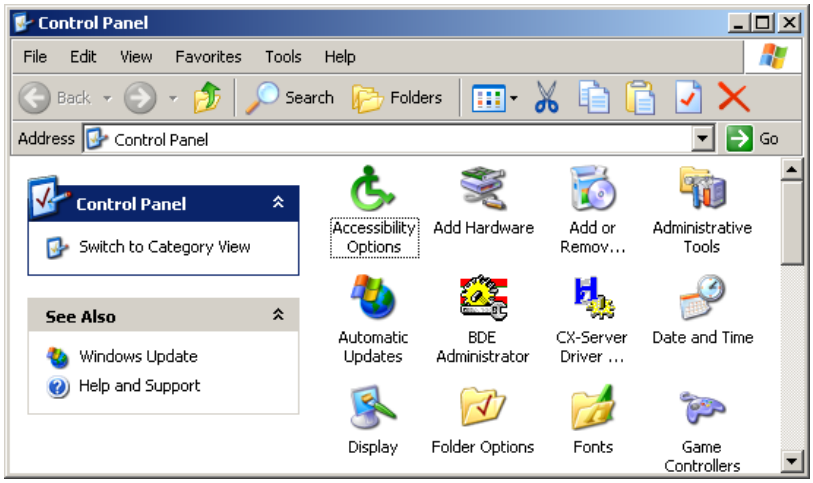
2.2.2 Utility Uninstallation

Following the steps, users can uninstall the iDCS-8000 Utility software by the following steps:

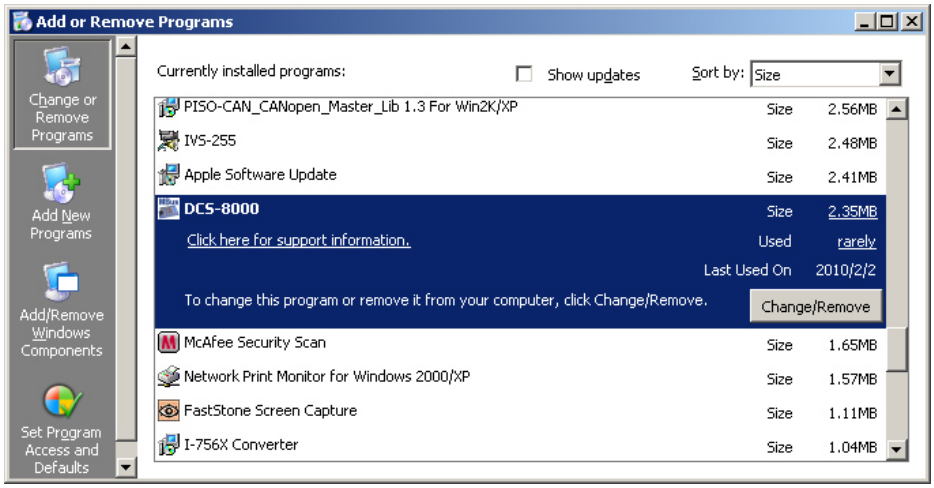
Step 1：Click "Start" in the task bar then selects the "Settings/Control Panel" shown in following figure.



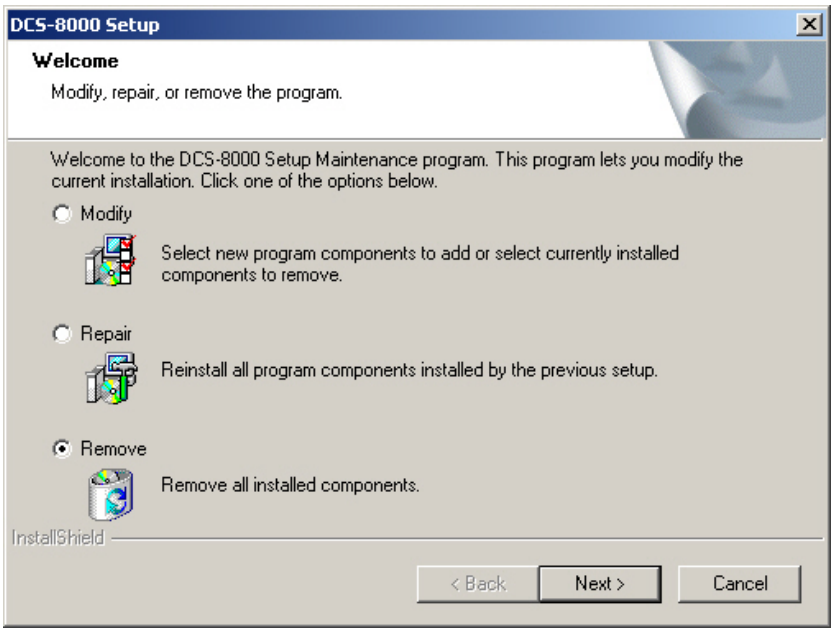
Step 2：Click the "Add/Remove Programs" button icon to open the dialog.



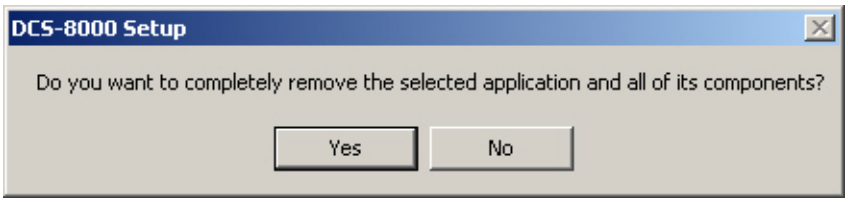
Step 3：Find out the iDCS-8000 Utility, and click the Change/Remove button.



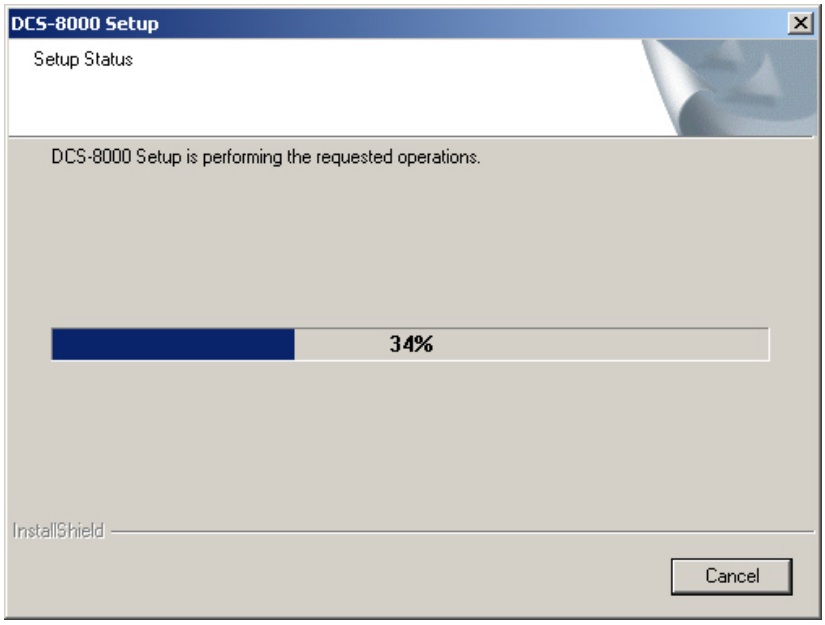
Step 4：Select the "Remove" option button, and press the "Next" button to remove iDCS-8000 Utility.



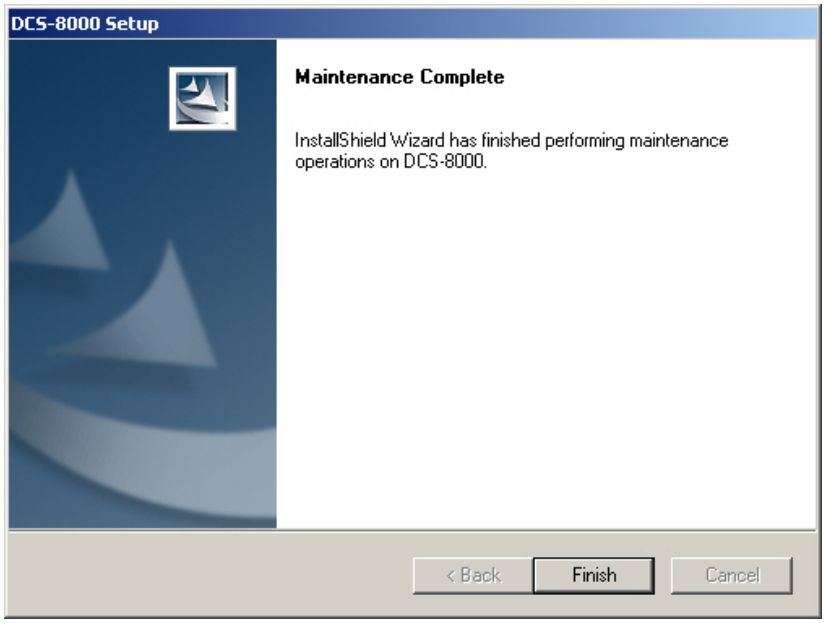
Step 5：Click the button "Yes" to remove the Utility tool.



Step 6：Remove iDCS-8000 Utility.



Step 7：Finally, click the "Finish" button to finish the uninstall process.

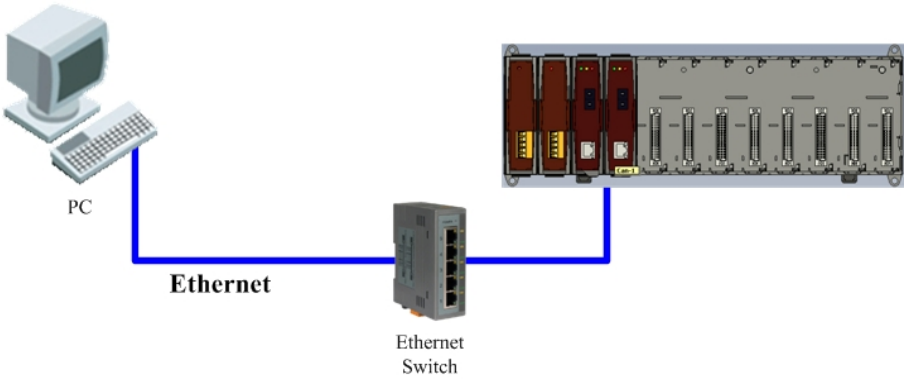


2.3 Hardware Configuration

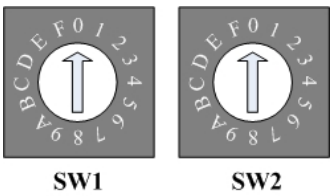
When first connect/install the iDCS-8000 system, you will need to make some adjustment for each setting to suit your requirements. The following instructment is the steps to configure iDCS-8000 I/O and FCM-MTCP.

Step 1：Configure the network settings (IP, Mask and Gateway) for the FCM-MTCP module.

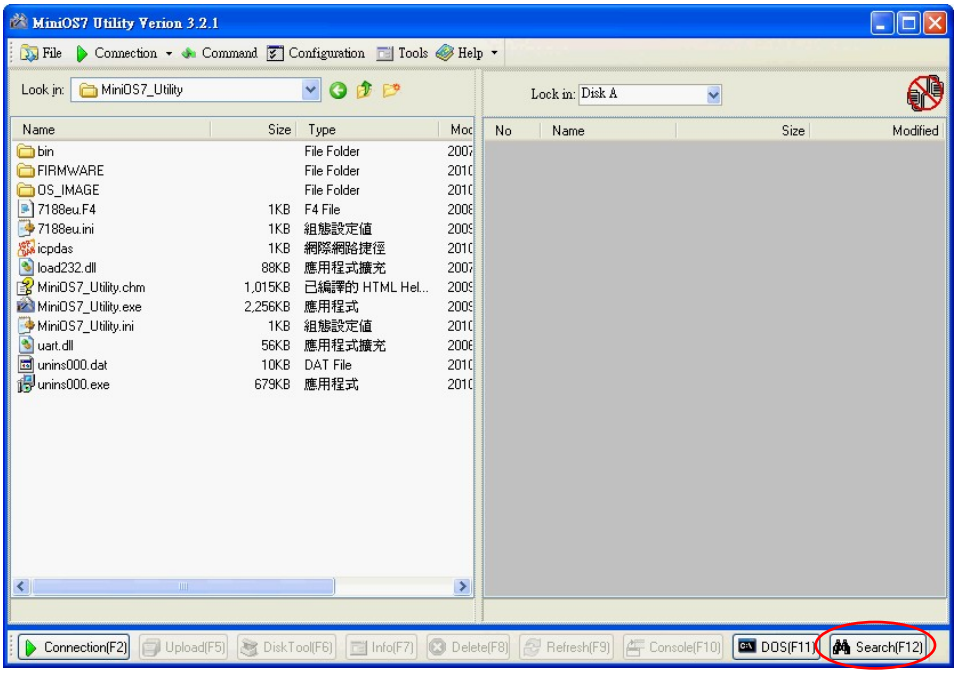
a. Connect PC to FCM-MTCP (The FCM-MTCP also can directly connect to PC without Ethernet switch)



b. Rotate the SW1/SW2 of the MCU for node ID of the FCM-MTCP. The SW1/SW2 means the fourth IP address of the MCU.

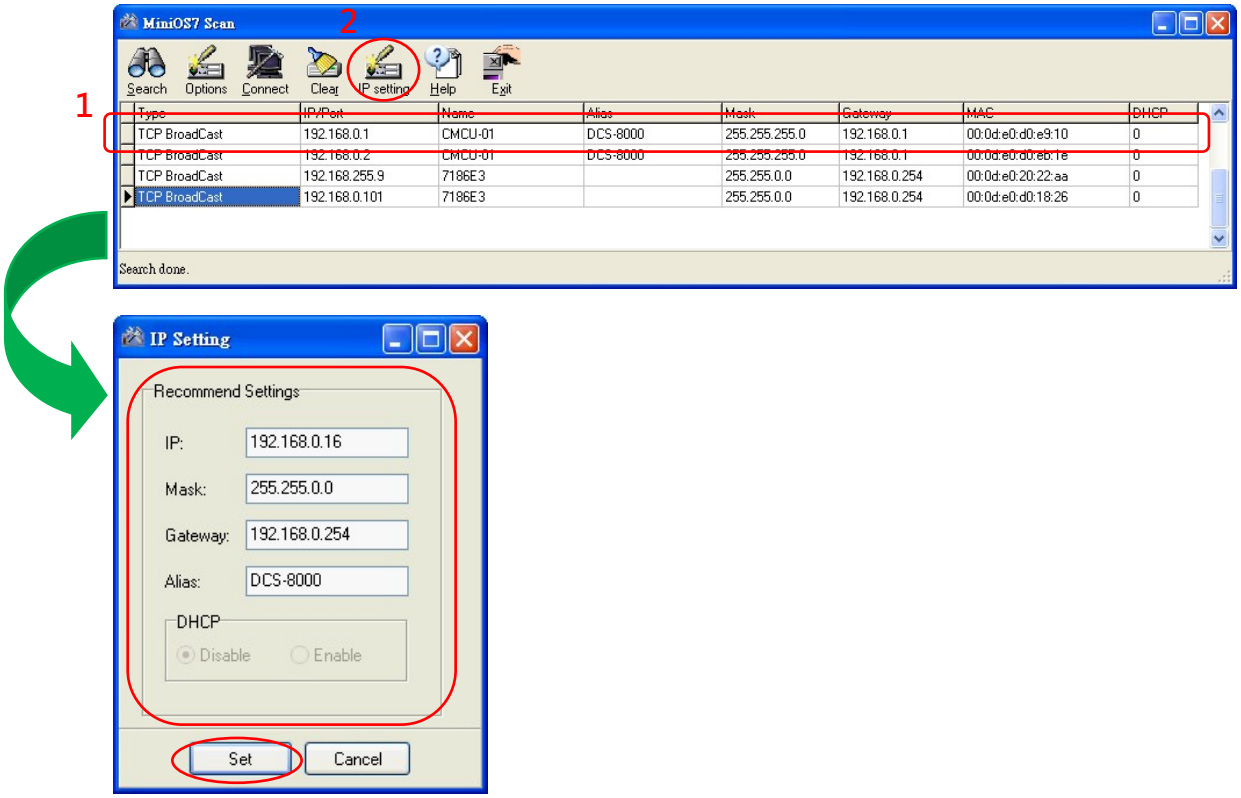


c. Use MiniOS7 Utility tool to search the MCU module.



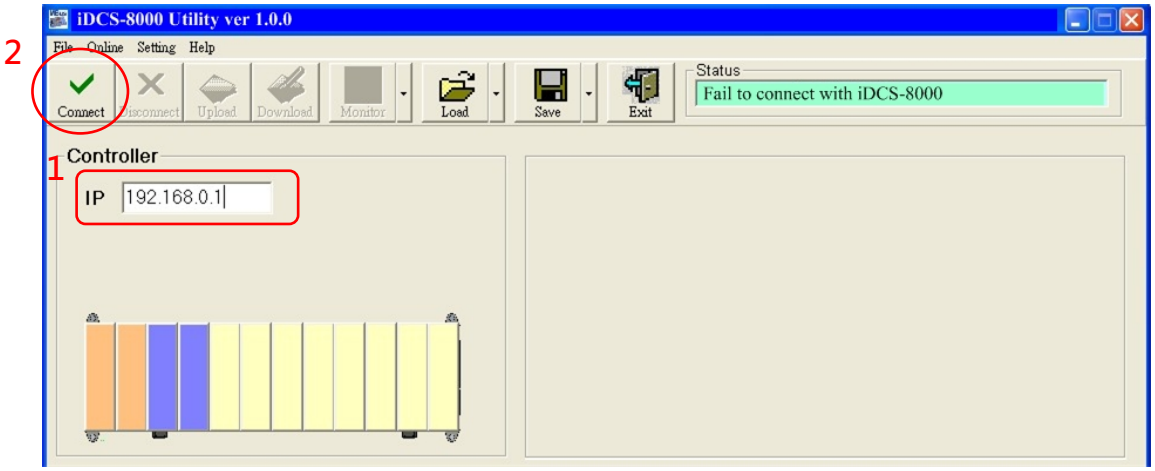
NOTE: To Use the MiniOS7, you must first install MiniOS7 Utility.

d. Searche the MCU and modify its IP/Gateway/Mask.



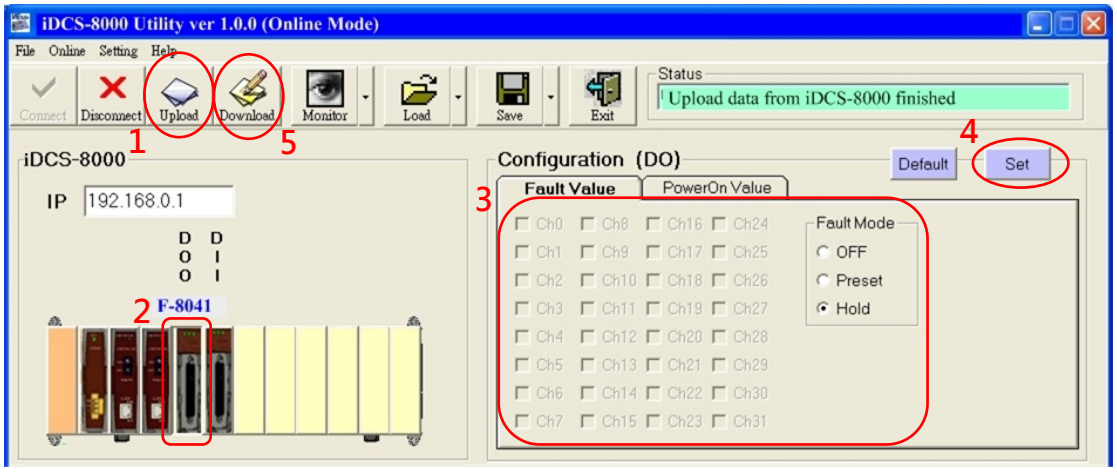
Step 2：Connect to iDCS-8000 controller via Ethernet.

Replace the default IP address with the IP address that you used in step 1, then click "Connect". After connecting and uploading data from to the iDCS-8000, the Utility will give detailed information regarding system settings, I/O configurations, etc.



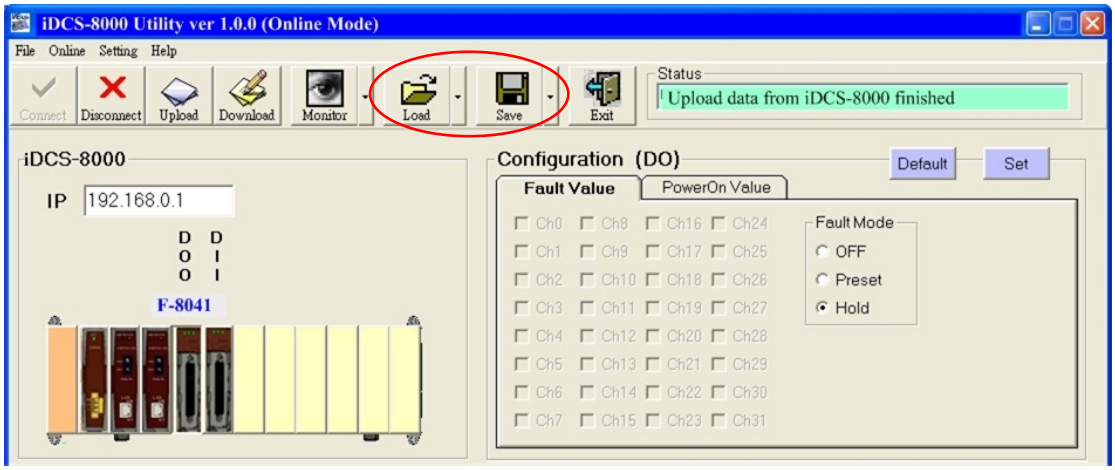
Step 3：Adjustment of module setting

After clicking "Upload", the module plugged in iDCS-8000 system will display in the rack list. Then click module in the rack list, the configuration of the module will display on the right side. Finally, click "Set" and "Download" button will apply the changes.



Step 3：Save/Load all settings.

After clicking "Save", the Utility will generate a file (default is iDCS-8830_MCU.ini). Users can use the "Load" method to load the file and review the settings of specific controller and every modules plug on it.



2.4 Parameter Configuration

This section will introduce how to configure the parameter of the module. Each module will have different parameter, like input type, power-on value…etc. These can be done by a specified command or the utility. The following section will describe the parameter for each module.

2.4.1 Parameter for Digital Input

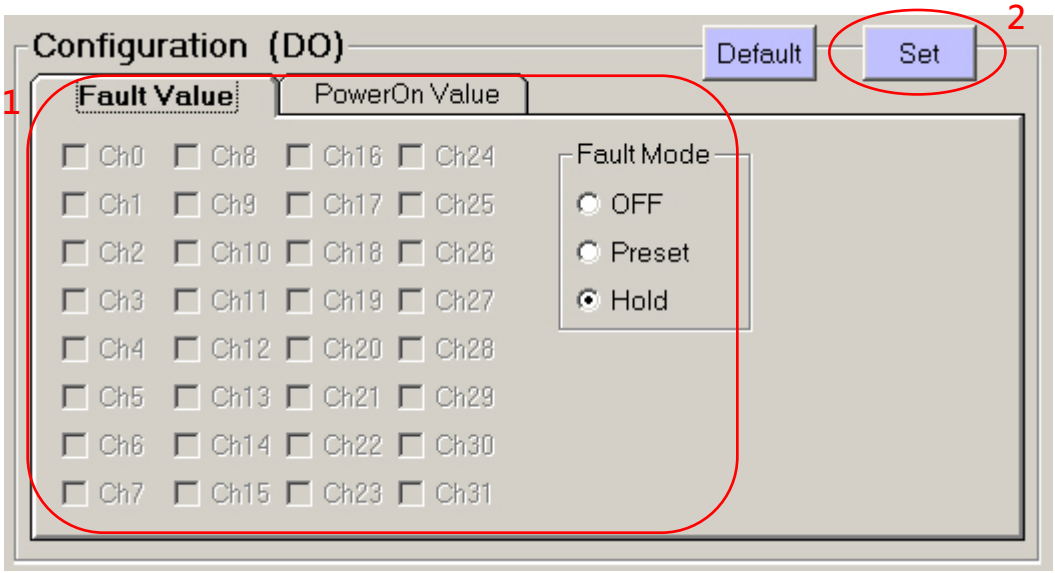
After connecting and uploading data from FCM-MTCP, the configuration of the DI module will be read and displayed on the "Configuration" panel.

NOTE: For DI modules, it do not exist any parameter need to be set, therefore you will see a display with empty panel for DI modules.

2.4.2 Parameter for Digital Output

After connecting and uploading data from FCM-MTCP, the configuration of the DO module will be read and displayed on the "Configuration" panel.

✓ **Configure the parameters for digital output**



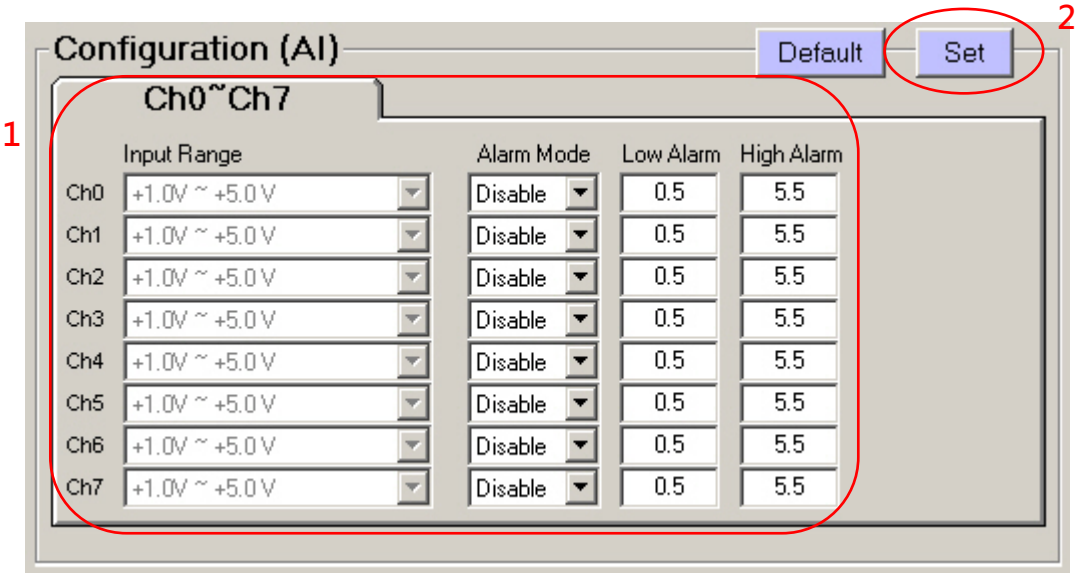
✓ **Download the configuration into module**

2.4.3 Parameter for Analog Input

After connecting and uploading data from FCM-MTCP, the configuration of the AI module will be read and displayed on the "Configuration" panel.

2.4.3.1 General Analog Input Module

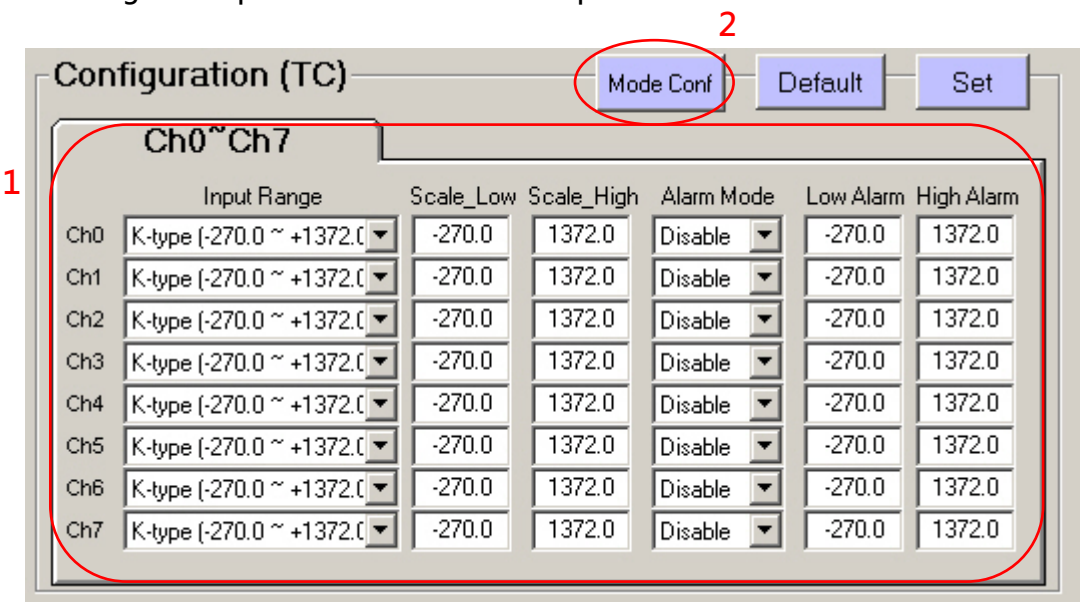
✓ Configure the parameters for analog input



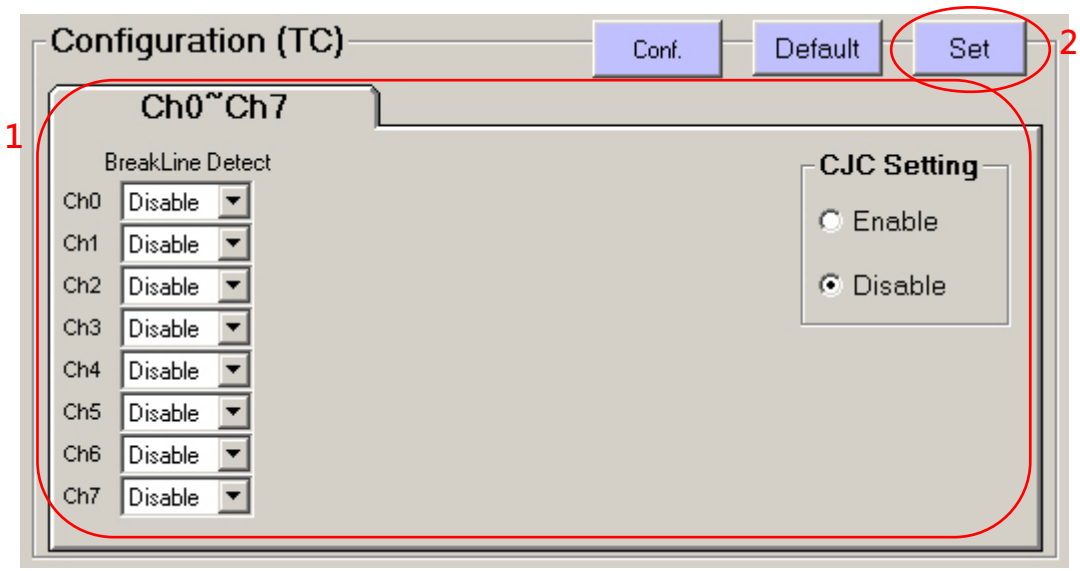
✓ Download the configuration into module

2.4.3.2 Thermocouple

✓ Configure the parameters for thermocouple



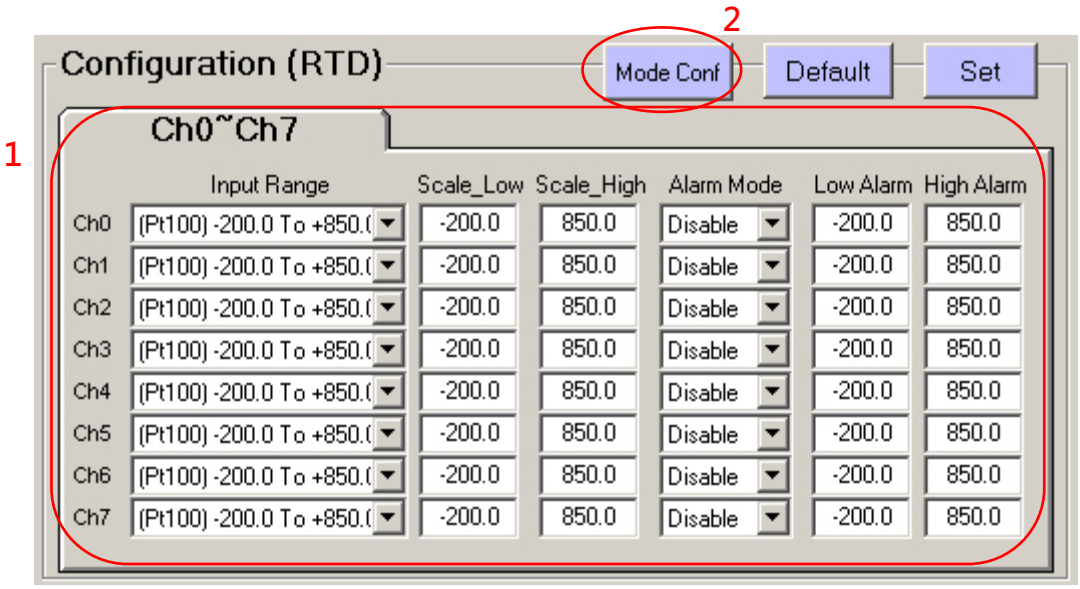
✓ **Configure another parameters for thermocouple**



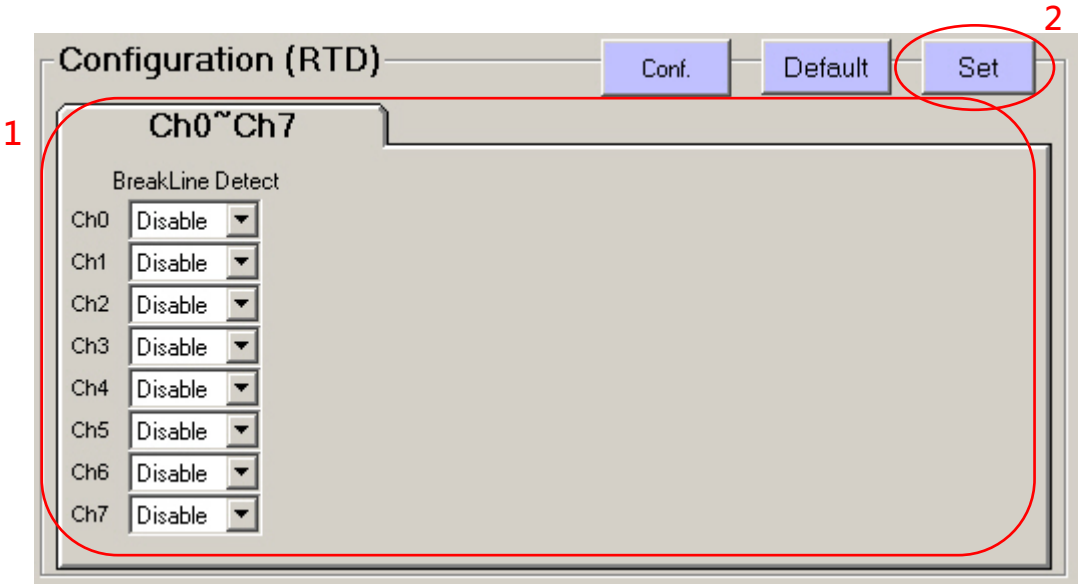
✓ **Download the configuration into module**

2.4.3.3 RTD

✓ **Configure the parameters for thermocouple**



✓ **Configure another parameters for thermocouple**

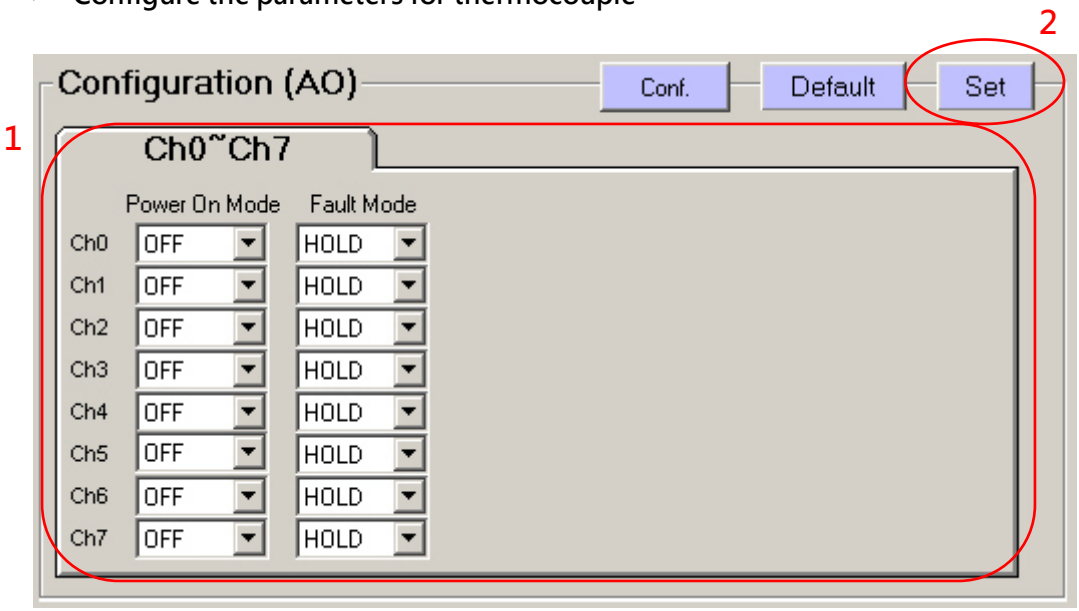


✓ **Download the configuration into module**

2.4.4 Parameter for Analog Output

After connecting and uploading data from FCM-MTCP, the configuration of the AO module will be read and displayed on the "Configuration" panel.

✓ **Configure the parameters for thermocouple**

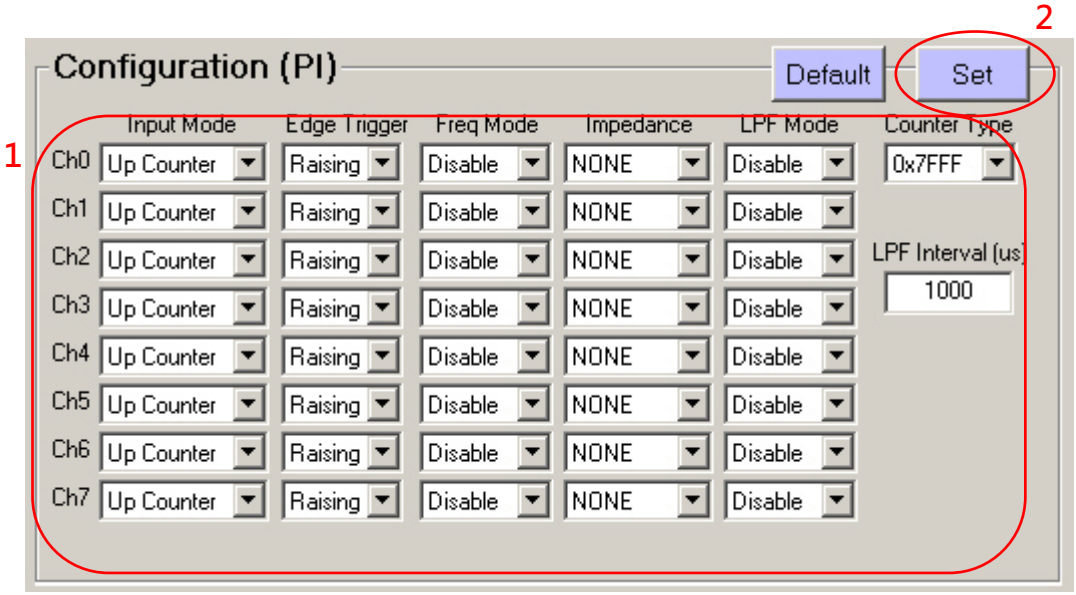


✓ **Download the configuration into module**

2.4.5 Parameter for Counter

After connecting and uploading data from FCM-MTCP, the configuration of the counter module will be read and displayed on the "Configuration" panel.

✓ **Configure the parameters for thermocouple**

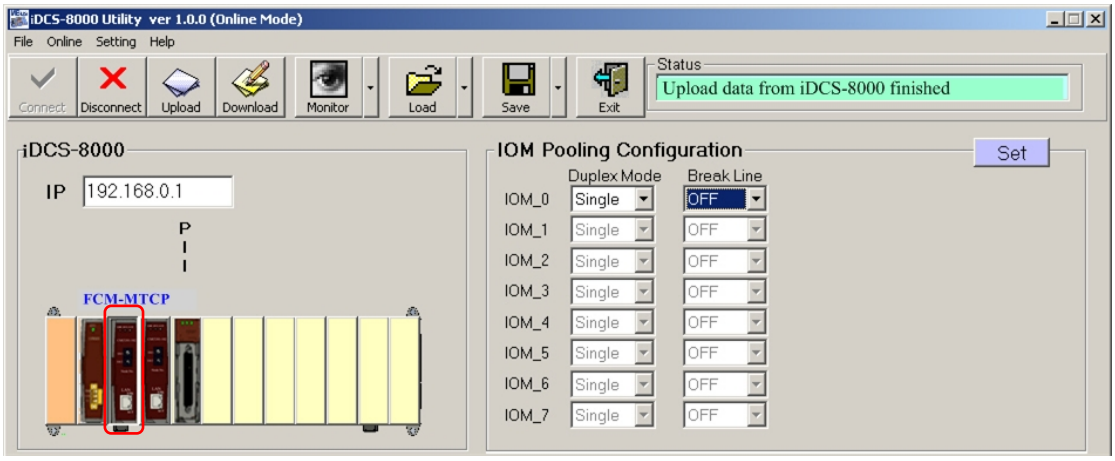


✓ **Download the configuration into module**

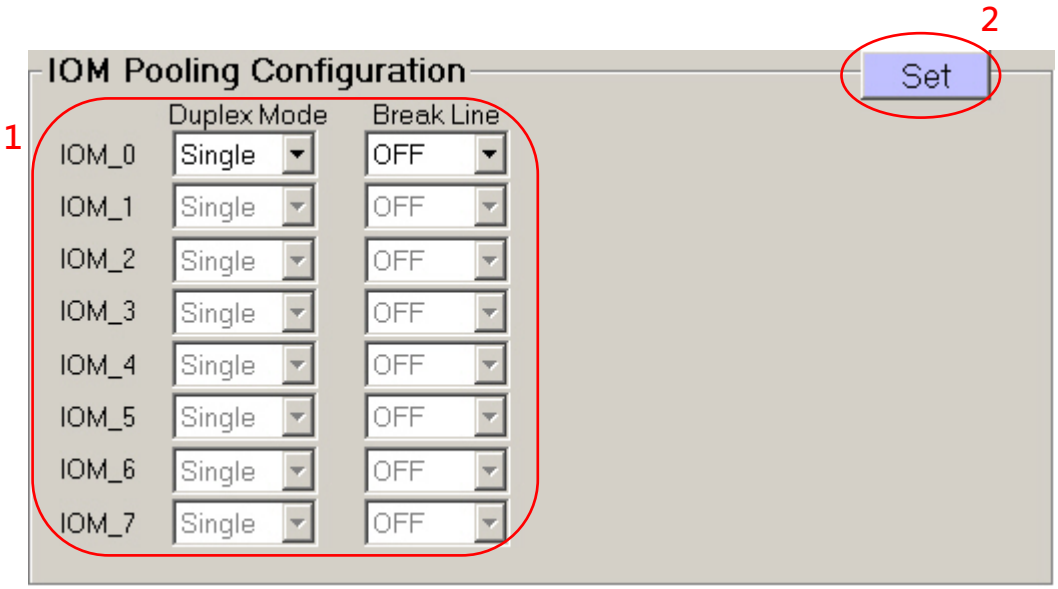
2.4.6 Common Parameters

There are some common parameters like redundancy and cable break-off detection that can be set. These parameters are listed in the MCU panel by clicking the FCM-MTCP module. After clicking FCM-MTCP modules, these parameters will be displaied on the right side of the Utility.

✓ **Click the FCM-MTCP module**



✓ **Set the common configuration**



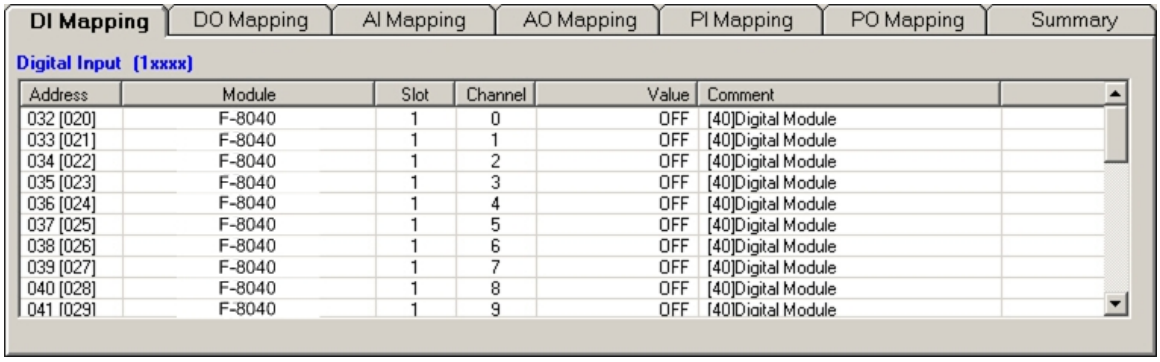
✓ **Download the configuration into module**

2.5 I/O Data Monitoring

The iDCS-8000 utility provides a "Monitor" function for user to read/write I/O data via Modbus/TCP protocol. After connecting and uploading data from iDCS-8000, users can follow the following steps to read/write IOM data.

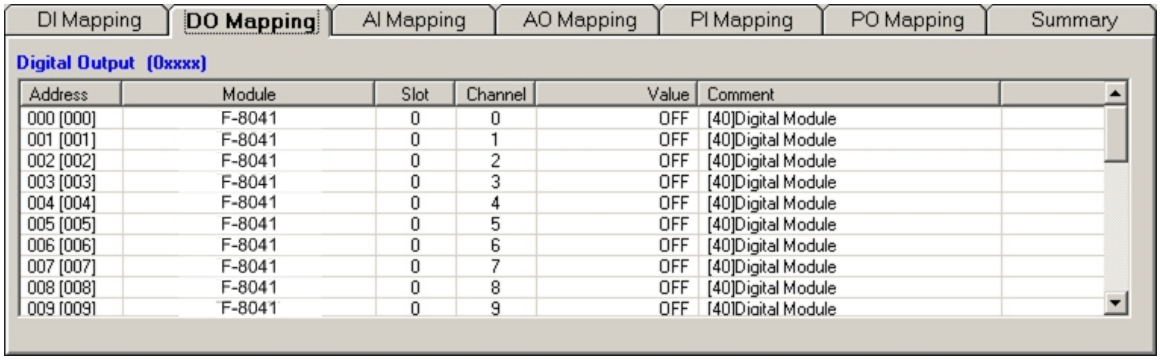
✓ **Enable the "Monitor" function**

✓ **Read I/O data (digital input, analog input and counter)**



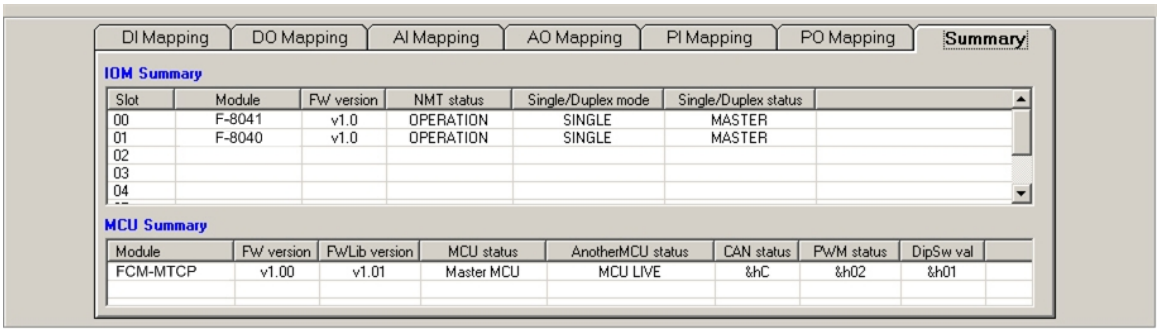
Address	Module	Slot	Channel	Value	Comment
032 [020]	F-8040	1	0	OFF	[40]Digital Module
033 [021]	F-8040	1	1	OFF	[40]Digital Module
034 [022]	F-8040	1	2	OFF	[40]Digital Module
035 [023]	F-8040	1	3	OFF	[40]Digital Module
036 [024]	F-8040	1	4	OFF	[40]Digital Module
037 [025]	F-8040	1	5	OFF	[40]Digital Module
038 [026]	F-8040	1	6	OFF	[40]Digital Module
039 [027]	F-8040	1	7	OFF	[40]Digital Module
040 [028]	F-8040	1	8	OFF	[40]Digital Module
041 [029]	F-8040	1	9	OFF	[40]Digital Module

✓ **Write I/O data (digital output, analog output and PWM)**



Address	Module	Slot	Channel	Value	Comment
000 [000]	F-8041	0	0	OFF	[40]Digital Module
001 [001]	F-8041	0	1	OFF	[40]Digital Module
002 [002]	F-8041	0	2	OFF	[40]Digital Module
003 [003]	F-8041	0	3	OFF	[40]Digital Module
004 [004]	F-8041	0	4	OFF	[40]Digital Module
005 [005]	F-8041	0	5	OFF	[40]Digital Module
006 [006]	F-8041	0	6	OFF	[40]Digital Module
007 [007]	F-8041	0	7	OFF	[40]Digital Module
008 [008]	F-8041	0	8	OFF	[40]Digital Module
009 [009]	F-8041	0	9	OFF	[40]Digital Module

✓ **Status overview**



Slot	Module	FW version	NMT status	Single/Duplex mode	Single/Duplex status
00	F-8041	v1.0	OPERATION	SINGLE	MASTER
01	F-8040	v1.0	OPERATION	SINGLE	MASTER
02					
03					
04					

MCU Summary

Module	FW version	FWLib version	MCU status	AnotherMCU status	CAN status	PWM status	DipSw val
FCM-MTCP	v1.00	v1.01	Master MCU	MCU LIVE	&hC	&h02	&h01

2.6 Saving/Loading Parameter

The iDCS-8000 utility tool offers "Saving" and "Loading" parameters to/from FCM-MTCP. These two functionalities can help users to read parameters at off-line mode and to write same parameters into another FCM-MTCP.

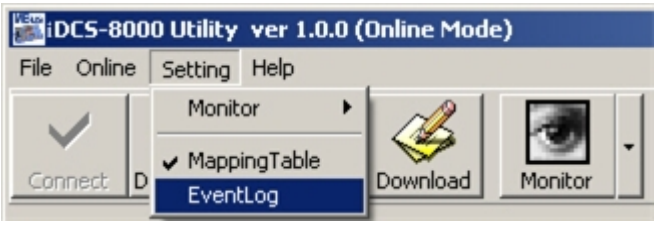
✓ **Load parameters from .ini file**

✓ **Save parameters into .ini file**

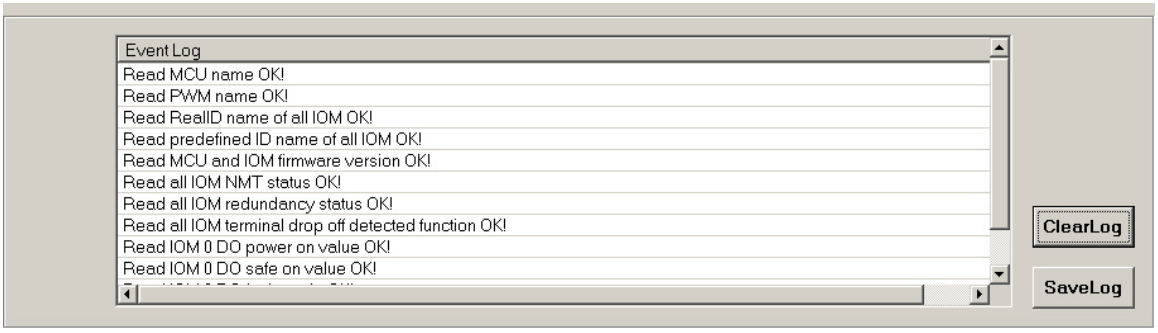
2.7 Event Log

The iDCS-8000 utility also supports event log to record all the process that utility has done.

✓ **Display the event log frame**



✓ **Save data log into file**



Section 3 Modbus Command

FCM-MTCP is a communication module with Modbus/TCP and Modbus/UDP slave protocol. The following table is the supported function code. Other function codes are not implemented. The query and response packet for each of these commands is then described in the following section.

Function Code	Modbus Command
0x01	Read coil status
0x02	Read input status
0x03	Read holding registers
0x04	Read input registers
0x05	Write single coil
0x06	Write single register
0x0F	Force multiple coils
0x10	Preset multiple registers

Each byte of the query and response packet of the Modbus command are described with the excepted of the field shown opposite. These are always present in the queries and responses of all Modbus commands.

Slave Address	Function Code	.....

Slave Address: This is sent from master and can not be changed (Valid Modbus device address: 1 to 247)

Function Code: This is sent from master and can not be changed (Function code of the Modbus command)

It is a better idea to have a standard Modbus document, such as the guide entitled Modicon Modbus Protocol Reference Guide, therefore you can see the correspondence between the elements displayed in Utility and the content of the corresponding Modbus frames.

The following sections are some introduction of Modbus function codes.

3.1 "Read Coil Status" (0x01)

Read the binary status of discrete output in the slave. The query message specifies the starting coil and the quantity of coils to be read. And the coil status in the response message is packed as one coil per bit of the data field.

✓ **Query (Each field represents a byte)**

Slave Address	Function Code	Start Address Hi	Start Address Lo	No. of Point Hi	No. of Point Lo

Here is an example of a request to read coils 20~56 from slave device 17:

Slave Address	Function Code	Start Address Hi	Start Address Lo	No. of Point Hi	No. of Point Lo
0x11	0x01	0x00	0x13	0x00	0x25

✓ **Respone (Each field represents a byte)**

Slave Address	Function Code	Byte Count	Coil Data

Here is an example of a response to the query on the opposite:

Slave Address	Function Code	Byte Count	Coil Data (27~20)		Coil Data (56~52)
0x11	0x01	0x05	0x13		0x1B

3.2 "Read Input Status" (0x02)

Read the binary status of discrete input in the slave. The query message specifies the starting input and quantity of inputs to be read. And the input status in the response message is packed as one input per bit of the data field.

✓ **Query (Each field represents a byte)**

Slave Address	Function Code	Start Address Hi	Start Address Lo	No. of Point Hi	No. of Point Lo

Here is an example of a request to read inputs 197~218 from slave device 16:

Slave Address	Function Code	Start Address Hi	Start Address Lo	No. of Point Hi	No. of Point Lo
0x10	0x02	0x00	0xC4	0x00	0x16

✓ **Respone (Each field represents a byte)**

Slave Address	Function Code	Byte Count	Input Data

Here is an example of a response to the query on the opposite:

Slave Address	Function Code	Byte Count	Input Data (204~197)		Input Data (218~213)
0x10	0x02	0x03	0xAD		0x25

3.3 "Read Holding Registers" (0x03)

Read the binary content of holding registers in the slave. The query message specifies the starting register and quantity of registers to be read. And the register data in the response message are packed as two bytes per register, with the binary contents right justified within each byte.

✓ **Query (Each field represents a byte)**

Slave Address	Function Code	Start Address Hi	Start Address Lo	No. of Point Hi	No. of Point Lo

Here is an example of a request to read register 108~110 from slave device 15:

Slave Address	Function Code	Start Address Hi	Start Address Lo	No. of Point Hi	No. of Point Lo
0x0F	0x03	0x00	0x6B	0x00	0x03

✓ **Respone (Each field represents a byte)**

Slave Address	Function Code	Byte Count	Holding Register Data

Here is an example of a response to the query on the opposite:

Slave Address	Function Code	Byte Count	Reg. Data Hi (Reg. 108)	Reg. Data Lo (Reg. 108)	
0x0F	0x03	0x06	0x30	0xF1	

3.4 "Read Input Registers" (0x04)

Read the binary content of input registers in the slave. The query message specifies the starting register and quantity of registers to be read. And the register data in the response message are packed as two bytes per register, with the binary contents right justified within each byte.

✓ **Query (Each field represents a byte)**

Slave Address	Function Code	Start Address Hi	Start Address Lo	No. of Point Hi	No. of Point Lo

Here is an example of a request to read register 9 & 10 from slave device 11:

Slave Address	Function Code	Start Address Hi	Start Address Lo	No. of Point Hi	No. of Point Lo
0x0B	0x04	0x00	0x08	0x00	0x02

✓ **Respone (Each field represents a byte)**

Slave Address	Function Code	Byte Count	Input Register Data

Here is an example of a response to the query on the opposite:

Slave Address	Function Code	Byte Count	Reg. Data Hi (Reg. 9)	Reg. Data Lo (Reg. 9)	
0x0B	0x04	0x04	0x01	0xA0	

3.5 "Force Single Coil" (0x05)

Force a single coil to either ON or OFF. The query message specifies the coil reference to be forced. The reguested ON/OFF state is specified by a constant in the query data field. A value of FF 00 hex requests the coil to be ON. A value of 00 00 requests it to be OFF. All other values are illegal and will not affect the coil.

✓ **Query (Each field represents a byte)**

Slave Address	Function Code	Coil Address Hi	Coil Address Lo	Force Data Hi	Force Data Lo

Here is an example of a request to force coil 173 ON in slave device 5:

Slave Address	Function Code	Coil Address Hi	Coil Address Lo	Force Data Hi	Force Data Lo
0x05	0x05	0x00	0xAC	0xFF	0x00

✓ **Respone (Each field represents a byte)**

Slave Address	Function Code	Coil Address Hi	Coil Address Lo	Force Data Hi	Force Data Lo

Here is an example of a response to the query on the opposite:

Slave Address	Function Code	Coil Address Hi	Coil Address Lo	Force Data Hi	Force Data Lo
0x05	0x05	0x00	0xAC	0xFF	0x00

3.6 "Force Multiple Coils" (0x0F)

Forces each coil in a sequence of coils to either ON or OFF. The query message specifies the coil references to be forced. The reguested ON/OFF states are specified by contents of the query data field. A logical 1 in a bit position of the field requests the corresponding coil to be ON. A logical 0 requests it to be OFF.

The normal response returns the slave address, function code, starting address, and quantity of registers preset.

✓ **Query (Each field represents a byte)**

Slave Address	Function Code	Coil Addr. Hi	Coil Addr. Lo	Quantity of Coil Hi	Quantity of Coil Lo	Byte Count	Force Data

Here is an example of a request to force coils 20~29 in slave device 9:

Slave Address	Function Code	Coil Addr. Hi	Coil Addr. Lo	Quantity of Coil Hi	Quantity of Coil Lo	Byte Count	Force Data
0x09	0x0F	0x00	0x13	0x00	0x0A	0x02	0xCD

✓ **Respone (Each field represents a byte)**

Slave Address	Function Code	Coil Addr. Hi	Coil Addr. Lo	Quantity of Coil Hi	Quantity of Coil Lo

Here is an example of a response to the query on the opposite:

Slave Address	Function Code	Coil Addr. Hi	Coil Addr. Lo	Quantity of Coil Hi	Quantity of Coil Lo
0x09	0x0F	0x00	0x13	0x00	0x0A

3.7 "Preset Multiple Registers" (0x10)

Preset values into sequence of holding registers. The query message specifies the register reference to be preset. The normal response returns the slave address, function code, starting address, and quantity of register preset.

✓ **Query (Each field represents a byte)**

Slave Address	Function Code	Start Addr. Hi	Start Addr. Lo	No. of Reg. Hi	No. of Reg. Lo	Byte Count	Reg. Data

Here is an example of a request to preset registers 2 & 3 to 0x123A and 0x0245B in slave device 9:

Slave Address	Function Code	Start Addr. Hi	Start Addr. Lo	No. of Reg. Hi	No. of Reg. Lo	Byte Count	Reg. Data
0x09	0x10	0x00	0x01	0x00	0x02	0x04	0x12

✓ **Respone (Each field represents a byte)**

Slave Address	Function Code	Start Addr. Hi	Start Addr. Lo	No. of Reg. Hi	No. of Reg. Lo

Here is an example of a response to the query on the opposite:

Slave Address	Function Code	Start Addr. Hi	Start Addr. Lo	No. of Reg. Hi	No. of Reg. Lo
0x09	0x10	0x00	0x01	0x00	0x02

3.8 Exception Responses

When a slave receives the query without a communication error, but cannot handle it, the slave will return an exception response informing the master of the nature of the error. The structure of an exception response is independent of the Modbus command associated with the "Function" field of the query involved. The whole frame of an exception response is shown below, depended on Modbus devices.

The exception response message has two fields that differentiate it from a normal response:

Function Code Field: In a normal response, the slave echoes the function code of the original query in the function code field of the response. All function codes have a most–significant bit (MSB) of 0 (their values are all below 80 hexadecimal). In an exception response, the slave sets the MSB of the function code to 1. This makes the function code value in an exception response exactly 80 hexadecimal higher than the value would be for a normal response. With the function code’s MSB set, the master’s application program can recognize the exception response and can examine the data field for the exception code.

Data Field: In a normal response, the slave may return data or statistics in the data field (any information that was requested in the query). In an exception response, the slave returns an exception code in the data field. This defines the slave condition that caused the exception.

Here is an example of a master query and slave exception response with exception code 0x02.

✓ **Query (Each field represents a byte)**

Slave Address	Function Code	Start Address Hi	Start Address Lo	No. of Point Hi	No. of Point Lo
0x0A	0x01	0x04	0xA1	0x00	0x01

✓ **Respone (Each field represents a byte)**

Slave Address	Function Code	Exception Code
0x0A	0x81	0x02

The following table is the exception code of the FCM-MTCP. Each code represents different meaning.

Code	Name	Meanning
0x01	Illegal Function	The function code received in the query is not an allowable action for the slave.
0x02	Illegal Data Address	The data address received in the query is not an allowable address for the slave.
0x03	Illegal Data Value	A value contained in the query data field is not an allowable value for the slave.
0x04	Slave Device Failure	An unrecoverable error occurred while the slave was attempting to perform the requested action.

Section 4 Modbus Addresses Mapping

4.1 Channel Value

4.1.1 Digital Input (Supported FC: 0x02)

The channel value of digital input modules in the FCM-MTCP can be read by function code 0x02, and the addree mapping are listed below.

Start Address (Decimal)	No. of Points	Address Range	Description
00001	32	00001~00032	Slot 0, channel 0 ~ 31 value
00033	32	00033~00064	Slot 1, channel 0 ~ 31 value
00065	32	00065~00096	Slot 2, channel 0 ~ 31 value
00097	32	00097~00128	Slot 3, channel 0 ~ 31 value
00129	32	00129~00160	Slot 4, channel 0 ~ 31 value
00161	32	00161~00192	Slot 5, channel 0 ~ 31 value
00193	32	00193~00224	Slot 6, channel 0 ~ 31 value
00225	32	00225~00256	Slot 7, channel 0 ~ 31 value

4.1.2 Digital Output (Supported FC: 0x01, 0x05, 0x0F)

The channel value of digital output modules in the FCM-MTCP can be read or written by function code 0x01, 0x05 and 0x0F, and the addree mapping are listed below.

Start Address (Decimal)	No. of Points	Address Range	Description
00001	32	00001~00032	Slot 0, channel 0 ~ 31 value
00033	32	00033~00064	Slot 1, channel 0 ~ 31 value
00065	32	00065~00096	Slot 2, channel 0 ~ 31 value
00097	32	00097~00128	Slot 3, channel 0 ~ 31 value
00129	32	00129~00160	Slot 4, channel 0 ~ 31 value
00161	32	00161~00192	Slot 5, channel 0 ~ 31 value
00193	32	00193~00224	Slot 6, channel 0 ~ 31 value
00225	32	00225~00256	Slot 7, channel 0 ~ 31 value

4.1.3 Analog Input (Supported FC: 0x04)

The channel value of analog input modules in the FCM-MTCP can be read by function code 0x04, and the addree mapping are listed below.

Start Address (Decimal)	No. of Points	Address Range	Description
00001	16	00001~00016	Slot 0, channel 0 ~ 15 value
00017	16	00017~00032	Slot 1, channel 0 ~ 15 value
00033	16	00033~00048	Slot 2, channel 0 ~ 15 value
00049	16	00049~00064	Slot 3, channel 0 ~ 15 value
00065	16	00065~00080	Slot 4, channel 0 ~ 15 value
00081	16	00081~00096	Slot 5, channel 0 ~ 15 value
00097	16	00097~00112	Slot 6, channel 0 ~ 15 value
00113	16	00113~00128	Slot 7, channel 0 ~ 15 value

4.1.4 Analog Output (Supported FC: 0x03, 0x06, 0x10)

The channel value of analog output modules in the FCM-MTCP can be read or written by function code 0x03, 0x06 and 0x10, and the addree mapping are listed below.

Start Address (Decimal)	No. of Points	Address Range	Description
00001	16	00001~00016	Slot 0, channel 0 ~ 15 value
00017	16	00017~00032	Slot 1, channel 0 ~ 15 value
00033	16	00033~00048	Slot 2, channel 0 ~ 15 value
00049	16	00049~00064	Slot 3, channel 0 ~ 15 value
00065	16	00065~00080	Slot 4, channel 0 ~ 15 value
00081	16	00081~00096	Slot 5, channel 0 ~ 15 value
00097	16	00097~00112	Slot 6, channel 0 ~ 15 value
00113	16	00113~00128	Slot 7, channel 0 ~ 15 value

4.1.5 Counter (Supported FC: 0x04)

The channel value of counter modules in the FCM-MTCP can be read by function code 0x04, and the addree mapping are listed below.

Start Address (Decimal)	No. of Points	Address Range	Description
00129	32	00129~00160	Slot 0, channel 0 ~ 15 value (Each channel takes 2 register)
00161	32	00161~00192	Slot 1, channel 0 ~ 15 value (Each channel takes 2 register)
00193	32	00193~00224	Slot 2, channel 0 ~ 15 value (Each channel takes 2 register)
00225	32	00225~00256	Slot 3, channel 0 ~ 15 value (Each channel takes 2 register)
00257	32	00257~00288	Slot 4, channel 0 ~ 15 value (Each channel takes 2 register)
00289	32	00289~00320	Slot 5, channel 0 ~ 15 value (Each channel takes 2 register)
00321	32	00321~00352	Slot 6, channel 0 ~ 15 value (Each channel takes 2 register)
00353	32	00353~00384	Slot 7, channel 0 ~ 15 value (Each channel takes 2 register)

4.1.6 PWM (Supported FC: 0x03, 0x06, 0x10)

The channel value of PWM modules in the FCM-MTCP can be read or written by function code 0x03, 0x06 and 0x10, and the addree mapping are listed below.

Start Address (Decimal)	No. of Points	Address Range	Description
00257	32	00257~00288	Slot 0, channel 0 ~ 15 value (Each channel takes 1 register)
00289	32	00289~00320	Slot 1, channel 0 ~ 15 value (Each channel takes 1 register)
00321	32	00321~00352	Slot 2, channel 0 ~ 15 value (Each channel takes 1 register)
00353	32	00353~00384	Slot 3, channel 0 ~ 15 value (Each channel takes 1 register)
00385	32	00385~00416	Slot 4, channel 0 ~ 15 value (Each channel takes 1 register)
00417	32	00417~00448	Slot 5, channel 0 ~ 15 value (Each channel takes 1 register)
00449	32	00449~00480	Slot 6, channel 0 ~ 15 value (Each channel takes 1 register)
00481	32	00481~00512	Slot 7, channel 0 ~ 15 value (Each channel takes 1 register)

4.1.7 HART

Users can send the HART command via Modbus TCP protocol by HART modules to field devices. The following sections describe how to send and receive HART by using Modbus TCP protocol.

4.1.7.1 HART Send (Supported FC: 0x10)

The following table shows how to send the HART.

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	0x02 0x00
4~5	Number of Point	2	Round up the value of (HART Data length + 10) / 2
6	Byte Count	1	HART Data length + 10
7~8	Header	2	0x40 0x00
9	Slot	1	Slot: 0x00~0x07
10	Channel	1	Channel: 0x00~0x10
11~12	HART Data Count	2	Total byte of the HART sending command
13~16	Reserved	4	Reserved
17~N	**HART Data**	**N**	**The HART sending data. The byte stream will be re-constructed in word format.** **For example, sending 0xFF 0xFF 0xFF 0xFF 0xFF 0x02 0x80 0x00 0x00 0x82, the byte stream will construct to Modbus register like the following table.** **Register / Value** 0 / 0xFFFF 1 / 0xFFFF 2 / 0x02FF 3 / 0x0080 4 / 0x8200

4.1.7.2 HART Receive (Supported FC: 0x04)

After sending HART command, uses have to poll the following table to see the HART reponse data and the status.

Where M = 256 x Slot ID

Start Address (Decimal)	No. of Points	Description	Remark
M+5121	1	HART process status	Status of HART transaction process
M+5122	1	HART process error code	Error code of HART transaction process
M+5123	1	HART response data length	Response length of HART transaction
M+5124	253	HART response data	Response data of HART transaction

4.2 System Information

The information of iDCS-8000 system is mapped into Modbus addresses. The information, like statuses of FCM-MTCP, bus, power...etc, can be read by function code 0x03 and listed below.

4.2.1 Statuses

Start Address (Decimal)	No. of Points	Description	Remark
00513	1	FCM-MTCP redundant mode	0x0020: Master 0x0021: Slave
00514	1	Neighbor FCM slot status	0x0000: Empty 0x0001: Timeout 0x0002: Undefined 0x0010: Normal
00515	1	System bus status	Refer to Table4.1
00516	1	FPM status	0x0000: No FPM plugged 0x0001: FPM1 GOOD / FPM2 OFF 0x0002: FPM1 OFF / FPM2 GOOD 0x0003: 2 FPMs are GOOD
00517	2	System minor fault status	Refer to Table4.2
00519	2	System major fault status	Refer to Table4.3
00521	1	FCM-MTCP ID	The value of SW1 and SW2

Table 4.1 System bus status

Bit	Description	Remark
7	Bus operating status	1: Bus-off, 0: Normal
6	Bus error indication	1: Bus error, 0: Normal
5	Tx status	1: Transmitting, 0: Idle
4	Rx status	1: Receiving, 0: Idle
3	Tx complete status	1: Complete, 0: Incomplete
2	Tx buffer status	1: Released, 0: Lock
1	Data overrun status	1: Data overrun, 0: Absent
0	Rx buffer status	1: Full, 0: Empty

Table 4.2 System minor fault status

Bit	Description	Remark
31		
30		
29		
28		
27		
26		
25		
24		
23		
22		
21		
20		
19		
18		
17		
16		
15		
14		
13		
12	Ethernet Link error	1: Fault, 0: None
11		
10		
9		
8	Dip switchs setting error when running	1: Fault, 0: None
7		
6		
5		
4		
3		
2		
1		
0		

Table 4.3 System major faut status

Bit	Description	Remark
31		
30		
29		
28		
27		
26		
25	System bus data overrun	1: Fault, 0: None
24	System bus off	1: Fault, 0: None
23		
22		
21		
20	Allocate MCU memory error when initial	1: Fault, 0: None
19		
18		
17		
16	MAC address error when initial	1: Fault, 0: None
15		
14		
13		
12		
11		
10		
9		
8	Dip-switch setting error when initial	1: Fault, 0: None
7		
6	Initial Modbus fail when initial	1: Fault, 0: None
5	Initial MCU fail when initial	1: Fault, 0: None
4	Initial CAN HW error when initial	1: Fault, 0: None
3		
2		
1		
0	MCU reset by WDT error	1: Fault, 0: None

4.2.2 System information

The system information is about the I/O information from slot 0 to 7. It includes Module ID, type, operating status, emergency, under/exceed limit and redundant mode.

Start Address (Decimal)	No. of Points	Description	Remark	
00577	8	Module ID	**Value**	**Module**
			0x0001	F-8040
			0x0010	F-8041
			0x0020	F-8015
			0x0030	F-8019
			0x0048	F-8017C
			0x004A	F-8017CH
			0x0083	F-8028CV
			0x0084	F-8028CH
00585	8	Module type	**Value**	**Type**
			0x0001	DI module
			0x0002	DO module
			0x0004	AI module
			0x0008	AO module
			0x0010	Counter module
			0x0020	PWM module
00593	8	I/ O slot status	**Value**	**Status**
			0x0001	Empty
			0x0002	Halt
			0x0004	Bootup
			0x0008	Bootloader
			0x0010	Pre-operation
			0x0020	Operation
			0x0040	Stop
00601	8	Operation mode	**Value**	**Mode**
			0x0000	Init
			0x0001	Single
			0x0002	Redundant & Slave
			0x0005	Redundant & Master
00609	8	Emergency	**Value**	**Mode**
			0x0100	Cable break-off
			0x0200	CJC error

00617	8	Under limit	Each bit represents 1 channel under limit flag 1: Channel value under limit 0: Normal
00625	8	Under 2nd limit	Each bit represents 1 channel under 2nd limit flag 1: Channel value under 2nd limit 0: Normal
00633	8	Exceed limit	Each bit represents 1 channel exceed limit flag 1: Channel value exceed limit 0: Normal
00641	8	Exceed 2nd limit	Each bit represents 1 channel exceed 2nd limit flag 1: Channel value exceed 2nd limit 0: Normal
00649	16	Channel break status	Each bit represents 1 channel break status, and each slot takes 2 register 1: Channel broken 0: Normal

4.3 Slot Information

The slot information contains the description of I/O slot in Modbus addresses. The information, like firmware version, type code, safe value, module type…etc, is provided as well for users. Each slot has own information table and occupies 512 register. The address of the information for the first slot starts from 1025 and uses Modbus function code 0x03 to read.

N = 512 x Slot ID, M = 256 x Slot ID			
Start Address (Decimal)	**No. of Points**	**Description**	**Remark**
N+1025	1	Module ID	Refer to system information
N+1026	1	Module Type	Refer to system information
N+1027	1	I/O slot status	Refer to system information
N+1028	1	Operation mode	Refer to system information
N+1029	1	Emergency	Refer to system information
N+1030	1	AI/AO under limit	Refer to system information
N+1031	1	AI/AO under 2nd limit	Refer to system information
N+1032	1	AI/AO exceed limit	Refer to system information
N+1033	1	AI/AO exceed 2nd limit	Refer to system information
N+1034	1	Firmware version	Ex: 0x000A → v1.0
N+1035	1	Redundant enable flag	0x0000: Disable, 0x0001: Enable
N+1036	1	Cable break-off detect enable	0x0000: Disable, 0x0001: Enable

N+1037	16	DO/AO power-on enable	0x0000: Disable, 0x0001: Preset Each register represents 1 channel
N+1053	16	DO/AO/PWM safety enable	0x0000: Disable 0x0001: Preset 0x0002: Hold Each register represents 1 channel
N+1069	16	DO/AO power-on value	DO: Each channel takes 1 bit (1st register) AO: Each channel takes 1 register
N+1085	32	DO/AO safety value	DO: Each channel takes 1 bit (1st register) AO: Each channel takes 1 register
N+1117	16	Type code	Refer to Table of Type code
N+1133	16	TC/RTD: Sensor low limit PWM: Cyclic steps	
N+1149	16	TC/RTD: Sensor high limit PWM: Duty step	
N+1165	16	AI/AO: Low limit alarm PWM: Offset timer	Each channel takes 1 register
N+1181	16	AI/AO: High limit alarm	Each channel takes 1 register
N+1197	1	AI/AO alarm enable	0x0000: Disable, 0x0001: Enable
N+1198	1	CJC enable	0x0000: Disable, 0x0001: Enable
N+1199	1	CJC offset value	
N+1200	1	TC/RTD channel broken detection enable	Each channel takes 1 bit 0: Disable, 1: Enable
N+1201	1	TC/RTD channel broken redundant switch	Each channel takes 1 bit 0: Disable, 1: Enable
N+1202	1	Counter filter enable	Each channel takes 1 bit 0: Disable, 1: Enable
N+1203	1	Counter frequency mode	Each channel takes 1 bit 0: 0.33 Sec, 1: 0.1 Sec
N+1204	1	Counter edge mode	Each channel takes 2 bits 0: Raising, 1: Falling, 2: Both
N+1205	2	Counter filter interval	1~32767 ms
N+1207	1	Counter input impedance	Each channel takes 2 bits 0: None, 1: 200 Ω, 2: 500 Ω, 3: 1000 Ω
N+1208	1	Counter type	Each channel takes 1 bit 0: limited to 0x7FFF, 1: limited to 0xFFFF
N+1209	16	PWM base time	Each channel takes 1 register 5~65535 ms

M+5121	1	HART process status	Status of HART transaction process
M+5122	1	HART process error code	Error code of HART transaction process
M+5123	1	HART response data length	Response length of HART transaction
M+5124	253	HART response data	Response data of HART transaction

Section 5 Configuration Command

There are large amounts of configuration, like type code, alarm low limit…etc, to the iDCS-8000 system. These configurations are listed as previous section of slot information. In order to configure these configurations, FCM-MTCP provides a configuring channel in Modbus which is located on start address 0x8000. The list of configuration is as following table.

Start Address	Index	Description	Section
0x8000	0x0000	I/O module port enable	5.1
0x8000	0x0001	I/O module slot status	5.2
0x8000	0x0010	I/O module redundant enable	5.3
0x8000	0x0011	I/O module cable break-off enable	5.4
0x8000	0x0020	DO power-on & safety value	5.5
0x8000	0x0021	DO power-on & safety mode	5.6
0x8000	0x0030	AO power-on value	5.7
0x8000	0x0031	AO safety value	5.8
0x8000	0x0032	AO power-on & safety mode	5.9
0x8000	0x0040	I/O module type code	5.10
0x8000	0x0041	TC/RTD low/high re-scaling value	5.11
0x8000	0x0042	I/O module alarm low/high limit	5.12
0x8000	0x0043	I/O module alarm enable	5.13
0x8000	0x0050	TC CJC enable	5.14
0x8000	0x0051	Channel broken line detection	5.15
0x8000	0x0060	Counter clear	5.16
0x8000	0x0061	Counter filter interval	5.17
0x8000	0x0062	Counter filter / frequency / edge	5.18
0x8000	0x0063	Counter input impedance & limit	5.19
0x8000	0x0070	PWM reference time	5.20
0x8000	0x0071	PWM offset time	5.21
0x8000	0x0072	PWM cyclic & duty	5.22
0x8000	0x0073	PWM safety mode	5.23

5.1 I/O module port enable (0x0000)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03
6	Byte Count	1	0x06
7~8	Index	2	**0x00 0x00**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Port Enable**	**2**	**0x0000: I/O port disable (Single)** **0x0001: I/O port enable (Single)** **0x0005: I/O port enable and master (Redundant)**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.2 I/O module slot status (0x0001)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03
6	Byte Count	1	0x06
7~8	Index	2	**0x00 0x01**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Slot Status**	**2**	**0x0001: Pre-operation mode** **0x0002: Operation mode**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.3 I/O module redundant enable (0x0010)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03
6	Byte Count	1	0x06
7~8	Index	2	**0x00 0x10**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Redundant Enable**	**2**	**0x0001: Single** **0x0002: Redundant**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.4 I/O module cable break-off enable (0x0011)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03
6	Byte Count	1	0x06
7~8	Index	2	**0x00 0x11**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Break-off enable**	**2**	**0x0000: Disable** **0x0001: Enable**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.5 DO power-on & safety value (0x0020)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x06
6	Byte Count	1	0x0C
7~8	Index	2	**0x00 0x20**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~14	**Power-on value**	**4**	**Each channel takes 1 bit**
15~18	**Safety value**	**4**	**Each channel takes 1 bit**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x06

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.6 DO power-on & safety mode (0x0021)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03
6	Byte Count	1	0x06
7~8	Index	2	**0x00 0x21**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11	**Power-on mode**	**1**	**0x00: Disable** **0x01: Preset**
12	**Safety mode**	**1**	**0x00: Disable** **0x01: Preset** **0x02: Hold**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.7 AO power-on value (0x0030)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N
6	Byte Count	1	0x08 + N*2
7~8	Index	2	**0x00 0x30**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Start channel(K)**	**2**	**0x0000~0x000F**
13~14	**Number of channel(N)**	**2**	**0x0001~0x0010**
15~16	**ChK power-on value**	**2**	**Each channel takes 2 bytes**
17~18	**ChK+1 power-on value**	**2**	
19	**…**	⋮	⋮

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.8 AO safety value (0x0031)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N
6	Byte Count	1	0x08 + N*2
7~8	Index	2	**0x00 0x31**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Start channel(K)**	**2**	**0x0000~0x000F**
13~14	**Number of channel(N)**	**2**	**0x0001~0x0010**
15~16	**ChK safety value**	**2**	**Each channel takes 2 bytes**
17~18	**ChK+1 safety value**	**2**	
19	**...**		

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.9 AO power-on & safety mode (0x0032)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N
6	Byte Count	1	0x08 + N*2
7~8	Index	2	**0x00 0x32**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Start channel(K)**	**2**	**0x0000~0x000F**
13~14	**Number of channel(N)**	**2**	**0x0001~0x0010**
15	**ChKpower-on mode**	**1**	**0x00: Disable** **0x01: Preset**
16	**ChK safety mode**	**1**	**0x00: Disable** **0x01: Preset** **0x02: Hold**
17	**ChK+1 power-on mode**	**1**	
18	**ChK+1 safety mode**	**1**	
19~	**…**		

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.10 I/O module type code (0x0040)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + (N+1)/2
6	Byte Count	1	0x08 + N*2
7~8	Index	2	**0x00 0x40**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Start channel(K)**	**2**	**0x0000~0x000F**
13~14	**Number of channel(N)**	**2**	**0x0001~0x0010**
15	**ChK type code**	**1**	**Refer to table of type code**
16	**ChK+1 type code**	**1**	
17~	**...**		

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + (N+1)/2

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.11 TC/RTD low/high re-scaling value (0x0041)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N*2
6	Byte Count	1	0x08 + N*4
7~8	Index	2	**0x00 0x41**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Start channel(K)**	**2**	**0x0000~0x000F**
13~14	**Number of channel(N)**	**2**	**0x0001~0x0010**
15~16	**ChK low re-scaling value**	**2**	
17~18	**ChK high re-scaling value**	**2**	
19~20	**ChK+1 low re-scaling value**	**2**	
21~22	**ChK+1 high re-scaling value**	**2**	
23~	**…**		

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N*2

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.12 I/O module alarm low/high limit (0x0042)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N*2
6	Byte Count	1	0x08 + N*4
7~8	Index	2	**0x00 0x42**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Start channel(K)**	**2**	**0x0000~0x000F**
13~14	**Number of channel(N)**	**2**	**0x0001~0x0010**
15~16	**ChK low alarm value**	**2**	
17~18	**ChK high alarm value**	**2**	
19~20	**ChK+1 low alarm value**	**2**	
21~22	**ChK+1 high ralarm value**	**2**	
23~	**...**		

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N*2

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.13 I/O module alarm enable (0x0043)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03
6	Byte Count	1	0x06
7~8	Index	2	**0x00 0x43**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Alarm Enable**	**2**	**0x0000: Disable** **0x0001: Enable**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.14 TC CJC enable (0x0050)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04
6	Byte Count	1	0x08
7~8	Index	2	**0x00 0x50**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**CJC enable**	**2**	**0x0000: Disable** **0x0001: Enable**
13~14	**CJC value**	**2**	**CJC value**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.15 Channel broken line detection (0x0051)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04
6	Byte Count	1	0x08
7~8	Index	2	**0x00 0x50**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Channel Broken Line Detection Enable**	**2**	**Each bit represents 1 channel** **0: Disable, 1: Enable**
13~14	**Channel Broken Line Redundant Switch Enable**	**2**	**Reserved**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.16 Counter clear (0x0060)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03
6	Byte Count	1	0x06
7~8	Index	2	**0x00 0x60**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Counter Clear**	**2**	**Each bit represents 1 channel** **0: No operation, 1: Clear channel counter**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x03

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.17 Counter filter interval (0x0061)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04
6	Byte Count	1	0x08
7~8	Index	2	**0x00 0x61**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~14	**Counter Filter Interval**	**4**	**1~32767 (us)**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.18 Counter filter / frequency / edge (0x0062)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x05
6	Byte Count	1	0x0A
7~8	Index	2	**0x00 0x62**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Filter enable**	**2**	**Each bit represents 1 channel** **0: Disable, 1: Enable**
13~14	**Update time for frequency mode**	**2**	**Each bit represents 1 channel** **0: 0.33 sec, 1: 0.1 sec**
15~16	**Counter edge trigger mode**	**2**	**2 bits represent 1 channel** **0: raising edge trigger** **1: falling edge trigger** **2: both**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x05

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.19 Counter input impedance & limit (0x0063)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04
6	Byte Count	1	0x08
7~8	Index	2	**0x00 0x63**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Counter input impedance**	**2**	**2 bits represent 1 channel** **0: highest impedance** **1: 200Ω** **2: 500Ω** **3: 1000Ω**
13~14	**Counter limit**	**2**	**Each bit represents 1 channel** **0: 0x7FFF, 1: 0xFFFF**

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.20 PWM reference time (0x0070)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + (N+1)/2
6	Byte Count	1	0x08 + N
7~8	Index	2	**0x00 0x70**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Start channel(K)**	**2**	**0x0000~0x000F**
13~14	**Number of channel(N)**	**2**	**0x0001~0x0010**
15	**ChK reference time**	**1**	**(ms)**
16	**ChK+1 reference time**	**1**	**(ms)**
17	**...**		

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + (N+1)/2

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.21 PWM offset time (0x0071)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + (N+1)/2
6	Byte Count	1	0x08 + N
7~8	Index	2	**0x00 0x71**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Start channel(K)**	**2**	**0x0000~0x000F**
13~14	**Number of channel(N)**	**2**	**0x0001~0x0010**
15	**ChK offset time**	**1**	**(ms)**
16	**ChK+1 offset time**	**1**	**(ms)**
17	**…**	⋮	⋮

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + (N+1)/2

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.22 PWM cyclic & duty (0x0072)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N*2
6	Byte Count	1	0x08 + N*4
7~8	Index	2	**0x00 0x72**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Start channel(K)**	**2**	**0x0000~0x000F**
13~14	**Number of channel(N)**	**2**	**0x0001~0x0010**
15~16	**ChK cyclic step**	**2**	
17~18	**ChK duty step**	**2**	
19~20	**ChK+1 cyclic step**	**2**	
21~22	**ChK+1 duty step**	**2**	
23	**…**		

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N*2

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

5.23 PWM safety mode (0x0073)

Query

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N
6	Byte Count	1	0x08 + N*2
7~8	Index	2	**0x00 0x73**
9~10	Specific Slot	2	Slot: 0x0000~0x0007
11~12	**Start channel(K)**	**2**	**0x0000~0x000F**
13~14	**Number of channel(N)**	**2**	**0x0001~0x0010**
15~16	**ChK safety mode**	**2**	**0x0000: Diable** **0x0002: Hold**
17~18	**ChK+1 safety mode**	**2**	
19	**…**		

Normal Response

Byte Index	Description	Length (Byte)	Remark
0	Slave Address	1	0x01
1	Function Code	1	0x10
2~3	Start Address	2	**0x80 0x00**
4~5	Number of Point	2	0x00 0x04 + N

Exception

Byte Index	Description	Length (Byte)	Remark	
0	Slave Address	1	0x01	
1	Function Code	1	0x90	
2	Exception Code	1	0x03	Incorrect data of received Incorrect number of bytes received

Appendix A

Analog Input Type Code

Code	Input Type	Code	Input Type
0x00	-15 mV ~ +15 mV	0x17	Type L TC, -200 ~ +800°C
0x01	-50 mV ~ + 50 mV	0x18	Type M TC, -200 ~ +100°C
0x02	-100 mV ~ +100 mV	0x19	Type $L_{DIN43710}$ TC, -200 ~ +900°C
0x03	-500 mV ~ +500 mV	0x1A	0 ~ +20 mA
0x04	-1 V ~ +1 V	0x1B	-150 V ~ +150 V
0x05	-2.5 V ~ +2.5 V	0x1C	-50 V ~ +50 V
0x06	-20 mA ~ +20 mA	0x20	Pt 100, α=.00385, -100 ~ +100°C
0x07	+4 mA ~ +20 mA	0x21	Pt 100, α=.00385, 0 ~ +100°C
0x08	-10 V ~ +10 V	0x22	Pt 100, α=.00385, 0 ~ +200°C
0x09	-5 V ~ +5 V	0x23	Pt 100, α=.00385, 0 ~ +600°C
0x0A	-1 V ~ +1 V	0x24	Pt 100, α=.003916, -100 ~ +100°C
0x0B	-500 mV ~ +500 mV	0x25	Pt 100, α=.003916, 0 ~ +100°C
0x0C	-150 mV ~ +150 mV	0x26	Pt 100, α=.003916, 0 ~ +200°C
0x0D	-20 mA ~ +20 mA	0x27	Pt 100, α=.003916, 0 ~ +600°C
0x0E	Type J TC, -210 ~ +760°C	0x28	Nickel 120, -80 ~ +100°C
0x0F	Type K TC, -210 ~ +1372°C	0x29	Nickel 120, 0 ~ +100°C
0x10	Type T TC, -270 ~ +400°C	0x2A	Pt 1000, α=.00392, -200 ~ +600°C
0x11	Type E TC, -270 ~ +1000°C	0x2B	Cu 100, α=.00421, -20 ~ +150°C
0x12	Type R TC, 0 ~ +1768°C	0x2C	Cu 100, α=.00427, 0 ~ +200°C
0x13	Type S TC, 0 ~ +1768°C	0x2D	Cu 1000, α=.00421, -20 ~ +150°C
0x14	Type B TC, 0 ~ +1820°C	0x2E	Pt 100, α=.00385, -200 ~ +200°C
0x15	Type N TC, -270 ~ +1300°C	0x2F	Pt 100, α=.003916, -200 ~ +200°C
0x16	Type C TC, 0 ~ +2320°C		

Analog Output Type Code

Code	Input Type
0x30	0 ~ +20 mA
0x31	4 ~ +20 mA
0x32	0 V ~ +10 V
0x33	-10 V ~ +10 V
0x34	0 V ~ +5 V
0x35	-5 V ~ +5 V

Pulse Input Type Code

Code	Input Type
0x50	Up counter
0x51	Frequency
0x52	Counter with battery backup
0x53	Encoder
0x54	Up/Down counter
0x55	Pulse/Direction counter
0x56	AB phase

Appendix B

How to update the OS image

While you have some special requirements or encounter the troubles during setting up your system, updating the OS of the FCM-MTCP (Main Control Unit) of iDCS-8830 may be needed. ICP DAS will continue releasing more and more features into the newer OS. You can download the latest version OS from the ICP DAS web site. If your system works well, it is not necessary to update the OS of your FCM-MTCP. When you want to update the OS, please refer to the following steps.

Step1: Get the last version of the OS image.

The latest version of the OS image can be obtained from:

ftp://ftp.icpdas.com/pub/beta_version/iDCS-883x/

Step2: Get the MiniOS7 Utility tool

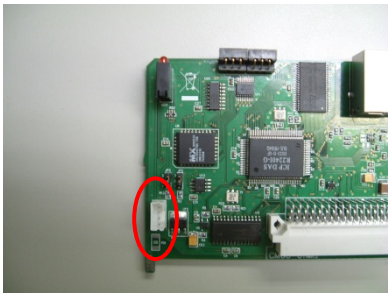
MiniOS7 Utility tool is useful while you want to update the OS of your FCM-MTCP. This tool can be obtained from:

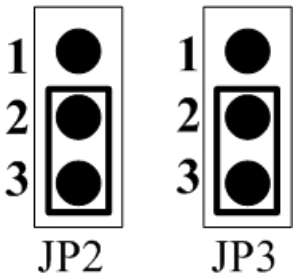
http://ftp.icpdas.com/pub/cd/8000cd/napdos/minios7/utility/minios7_utility/

Step3: Configure the FCM-MTCP to initial mode

Power off the FCM-MTCP, select the jumper JP2 and JP3 as the following figure, and connect the JP1 with CA-0904 cable.

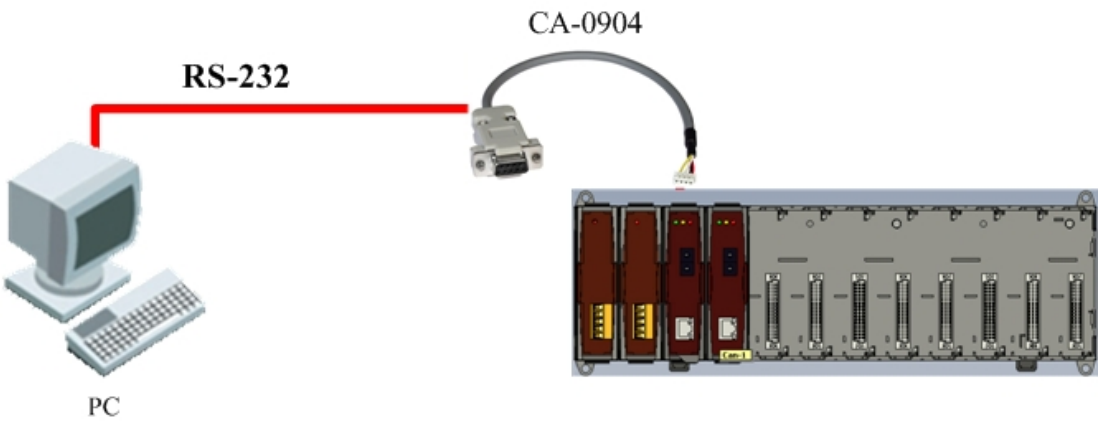
CA-0904





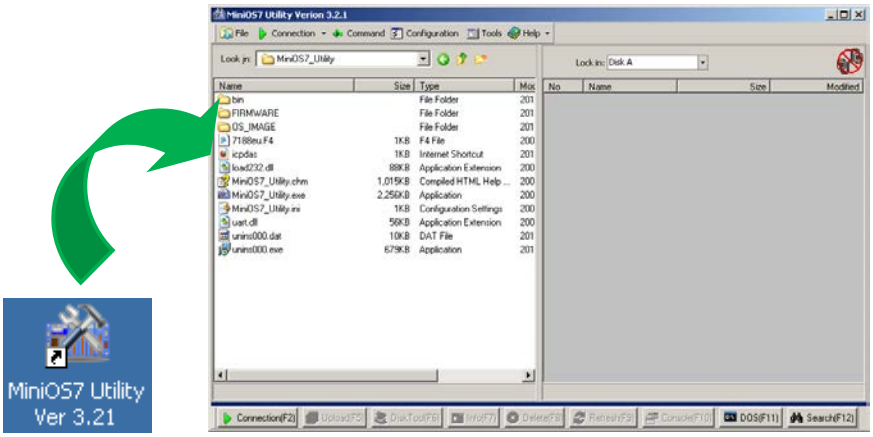
Step4: Connect PC with the iDCS-8830 via the PC COM port

Connect the PC with the iDCS-8830 via the COM port, and then power on the iDCS-8830.

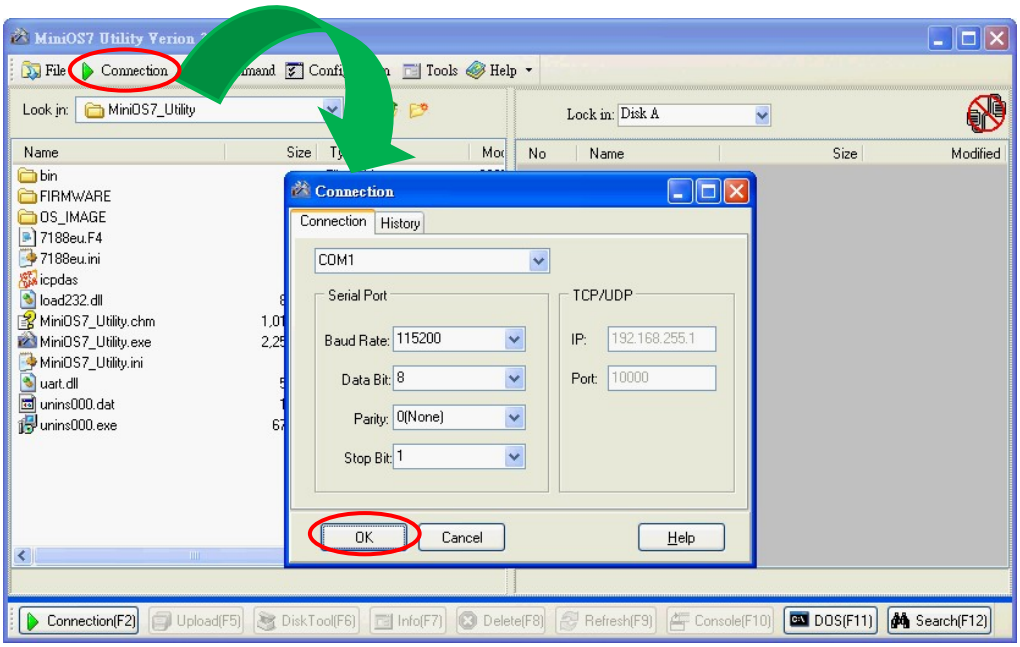


Step5: Start the MiniOS7 Utility tool

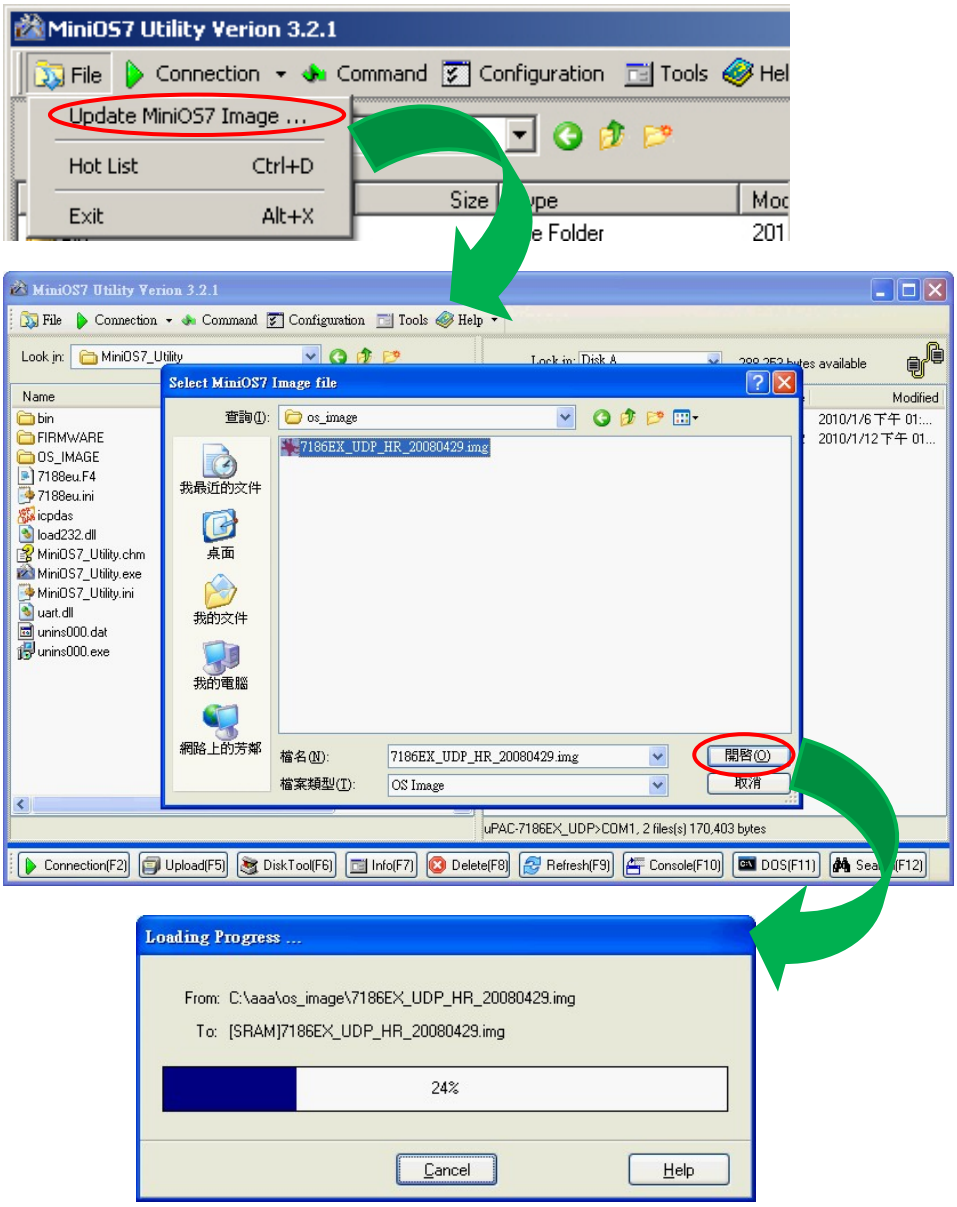
Double click the MiniOS7 Utility icon on the desktop to execute it.



Step6: Connect to the FCM-MTCP

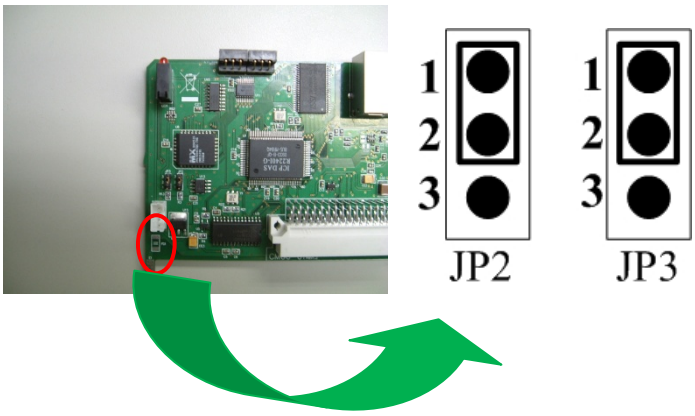


Step7: Update OS image



Step8: Configure the FCM-MTCP to normal mode

After finishing the update procedure, power off the FCM-MTCP, select the jumper JP2 and JP3 as the following figure and remove the CA-0904 cable from JP1.



How to modify IP address

The default IP/Gateway/Mask and Modbus NetID of FCM-MTCP is:

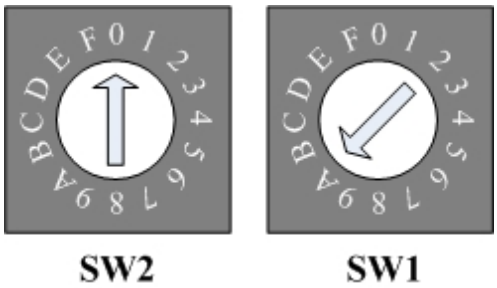
IP	192.168.255.1
Subnet Mask	255.255.0.0
Gateway	192.168.0.1
Modbus Slave Address	1

The rotary switch, SW1 and SW2, are used to configure the 4th-section IP address (from 1 to 254) of the FCM-MTCP. Setting the 4th-section IP address to 0 or 255 will cause a heavy fault in the FCM-MTCP. If you want to modify the 1st to the 3rd section IP address, the MiniOS7 utility tool is required to do this.

Here are examples to show you how these two rotary switches stand for. (**NOTE:** IP address of the FCM-MTCP is "192.168.255.1" in these examples).

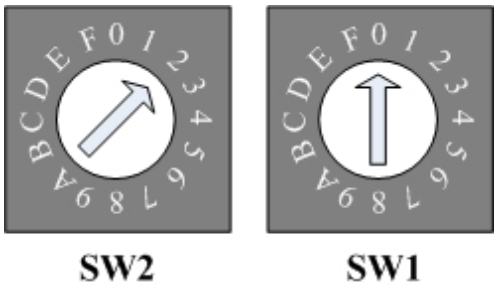
Example 1:

Powering off the iDCS-8000 and swtiching the SW2 and SW1 to '0' and 'A' respectively. After powering on the iDCS-8000, the IP address of FCM-MTCP will be "192.168.255.10"



Example 2:

Switching the SW2 and SW1 to '2' and '0' respectively, the IP address of FCM-MTCP will be "192.168.255.32"



Then the following article will teach you how to modify the 1st to 3rd IP address of the FCM-MTCP.

Step1: Get the MiniOS7 Utility tool

MiniOS7 Utility tool is useful while you want to update the IP address of your FCM-MTCP. This tool can be obtained from:

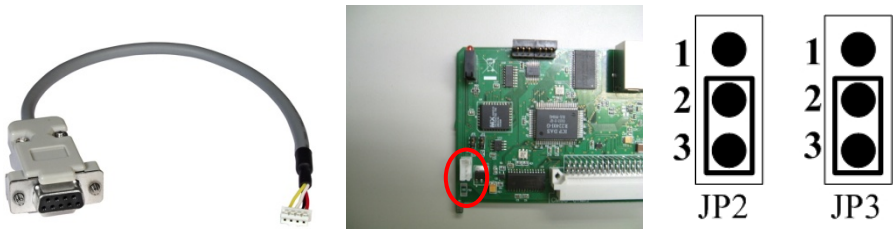
http://ftp.icpdas.com/pub/cd/8000cd/napdos/minios7/utility/minios7_utility/

After installing MiniOS7 Utility, there will two ways to modify IP address of FCM-MTCP.

Modifying IP address via COM Port

Step1: Switching the FCM-MTCP to initial mode

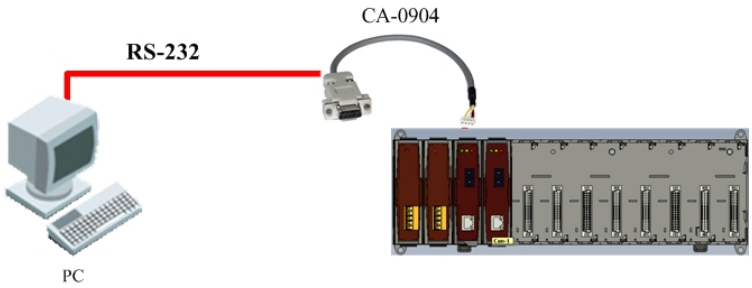
Power off the iDCS-8000, select the jumper JP2 and JP3 of the FCM-MTCP as the following figure, and connect the JP1 with CA-0904 cable.



CA-0904

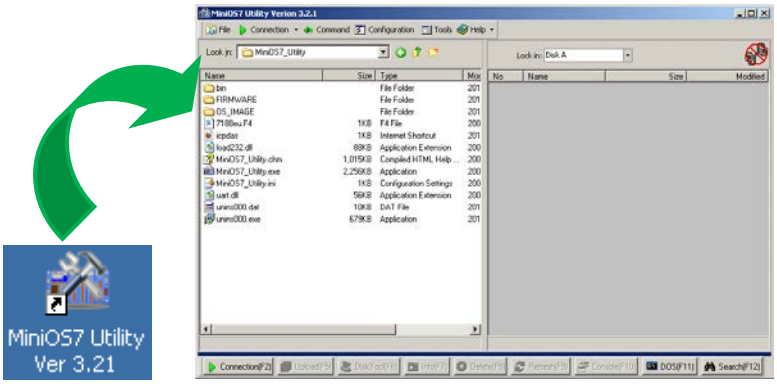
Step2: Connecting PC with the FCM-MTCP via the PC COM port

Connect the PC with the FCM-MTCP via the COM port, and power on the iDCS-8000.

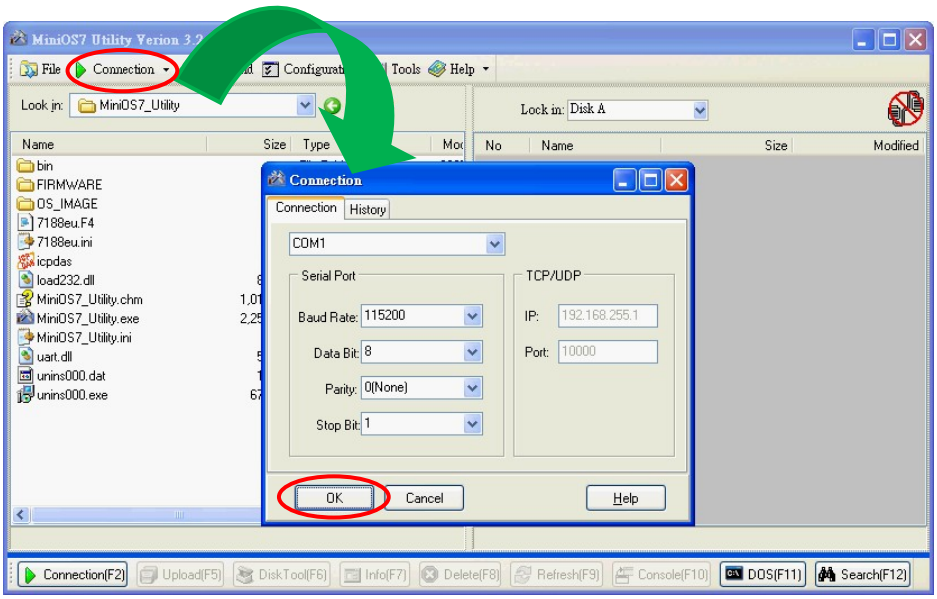


Step3: Execute the MiniOS7 Utility tool

Double click the MiniOS7 Utility icon on the desktop to execute it.

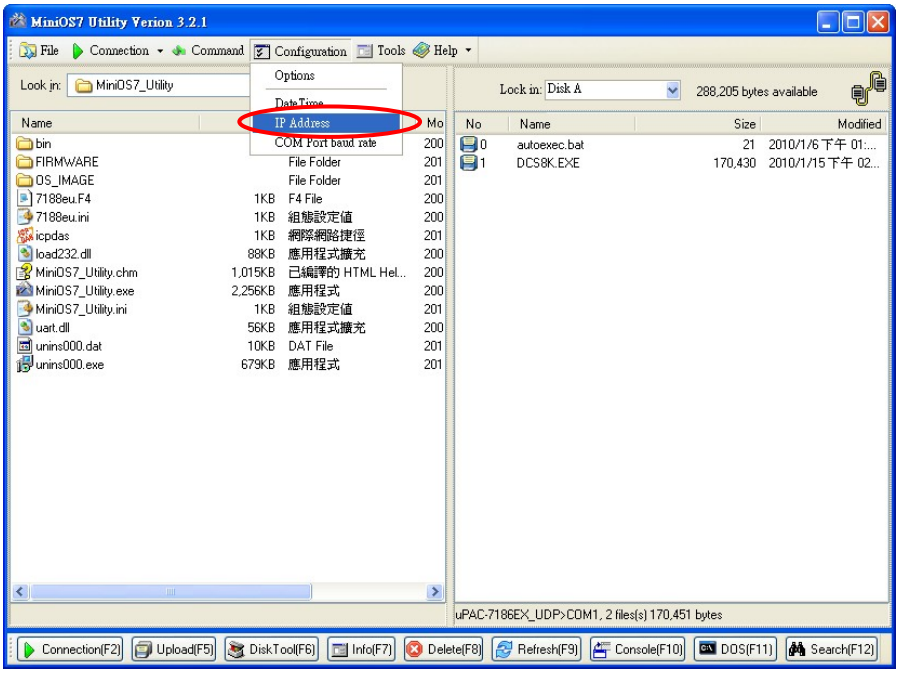


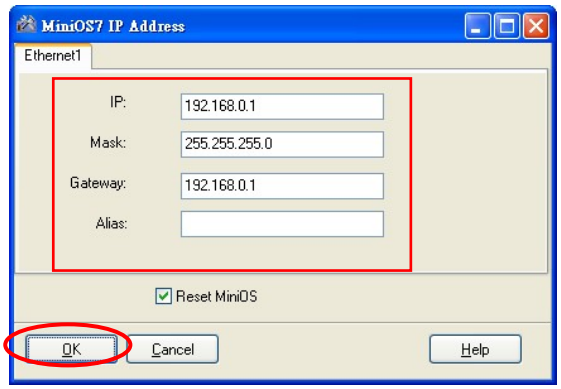
Step4: Connect to the FCM-MTCP



Step5: Modify the IP address of the FCM-MTCP device

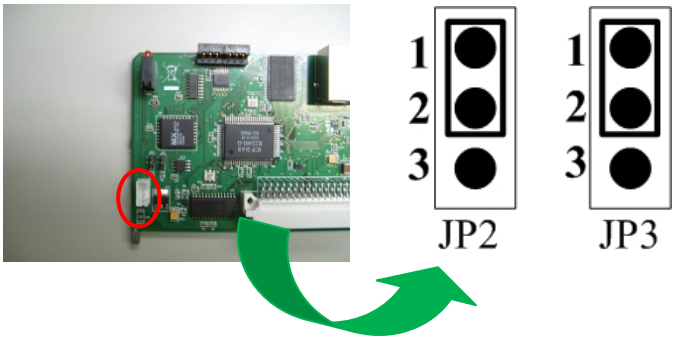
Select the "Configuration/IP Address" to set the IP address of the FCM-MTCP.





Step6: Configure the FCM-MTCP to normal mode

After finishing the update procedure, power off the iDCS-8830, then select the jumper JP2 and JP3 of the FCM-MTCP as the following figure and remove the CA-0904 cable from JP1.

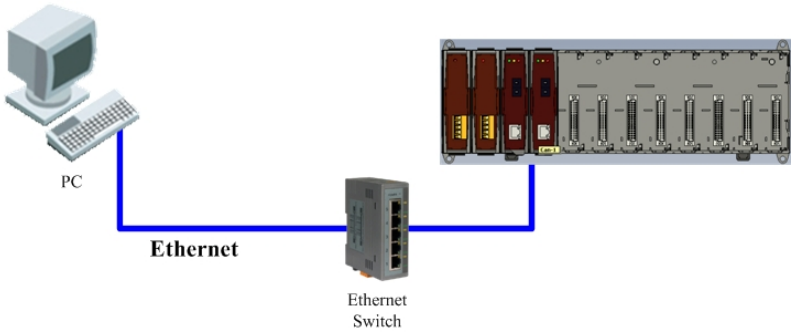


NOTE: After finishing the IP configuration and reboot the iDCS-8000, it will boot up with the IP address modified before and the 4th-section IP address will follow the setting of SW1 and SW2 rotary switches.

Modifying IP address from Ethernet

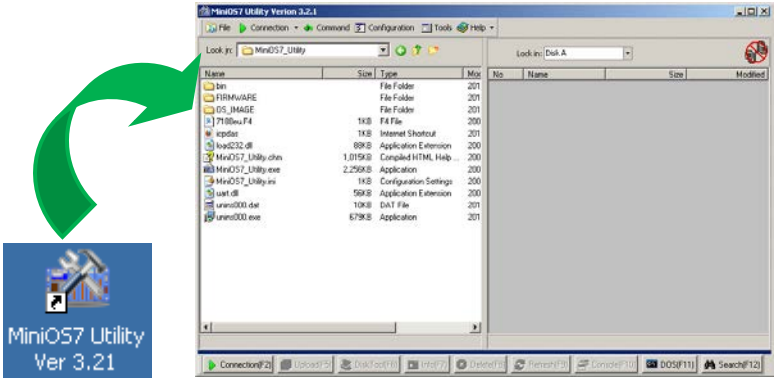
Step1: Connect PC with the iDCS-8830 via the Ethernet port

Power off the iDCS-8000, connect the PC with the iDCS-8000 via the Ethernet port, and power on the iDCS-8000. The Ethernet switch or hub may be needed while you connect the PC with the iDCS-8000.



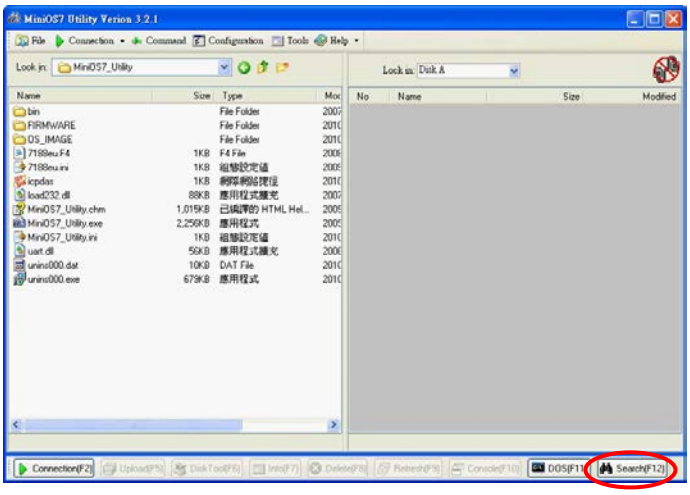
Step2: Start the MiniOS7 Utility tool

Double click the MiniOS7 Utility icon on the desktop to execute it.

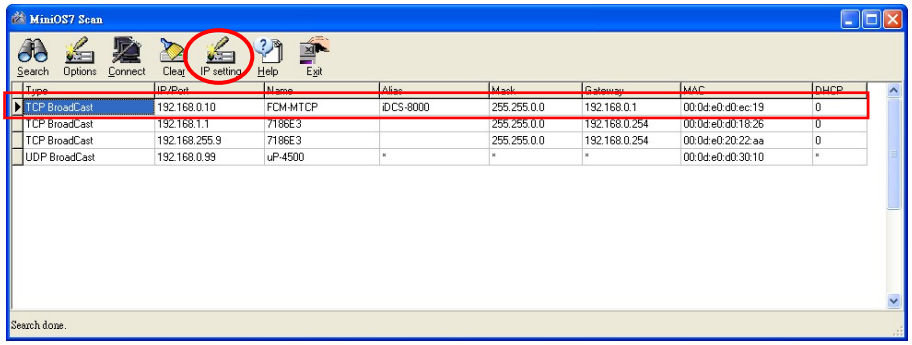


Step3: Search the FCM-MTCP device and modify the IP address

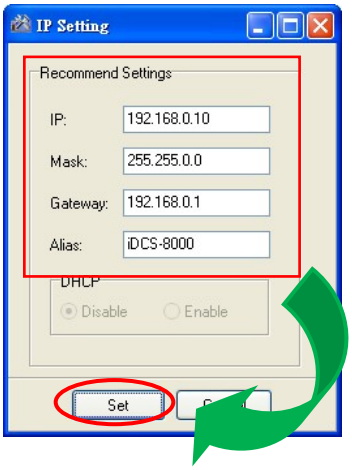
Click the "Search" button on the bottom left corner of the MiniOS7 utility tool.



Select the FCM-MTCP by using current IP address, and click the "IP setting" button to modify the IP address.



Modify the IP/Gateway/Mask



Step4: Reboot the iDCS-8000

After finishing the IP configuration, power off the iDCS-8000 and power it on. It will boot up with the IP address modified before and the 4th-section IP address will follow the setting of SW1 and SW2 rotary switches.

How to update the FCM-MTCP firmware

While you have some special requirements or have some troubles during setting up your system, updating the firmware of the FCM-MTCP may be needed. ICP DAS will continue releasing more and more features into the newer firmware. You can download the latest version firmware from the ICP DAS web site. If your system works well, it is not necessary to update the firmware of your FCM-MTCP. When you want to update the firmware, please refer to the following steps.

Step1: Get the last version of the firmware.

The latest version of the firmware can be obtained from:

ftp://ftp.icpdas.com/pub/beta_version/iDCS-883x/

Step2: Get the MiniOS7 Utility tool

MiniOS7 Utility tool is useful while you want to update the firmware of your FCM-MTCP. This tool can be obtained from:

http://ftp.icpdas.com/pub/cd/8000cd/napdos/minios7/utility/minios7_utility/

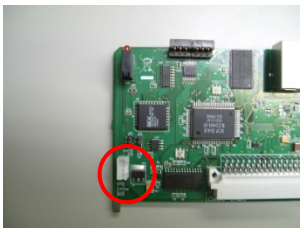
After installing MiniOS7 Utility, there will two ways to update the firmware of FCM-MTCP.

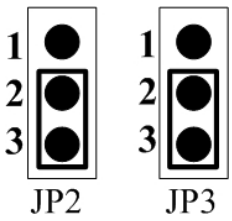
Update firmware via COM Port

Step1: Configure the FCM-MTCP to initial mode

Power off the iDCS-8000, select the jumper JP2 and JP3 as the following figure, and connect the JP1 with CA-0904 cable.

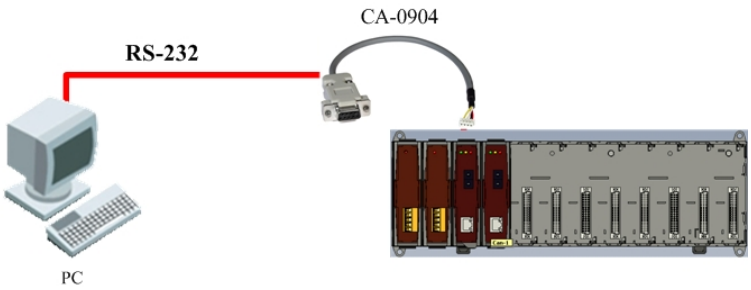
CA-0904





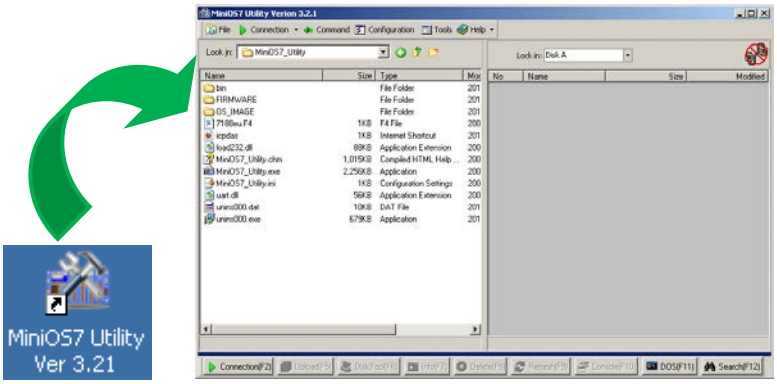
Step2: Connect PC with the FCM-MTCP via the PC COM port

Connect the PC with the FCM-MTCP via the COM port, and power on the iDCS-8000.

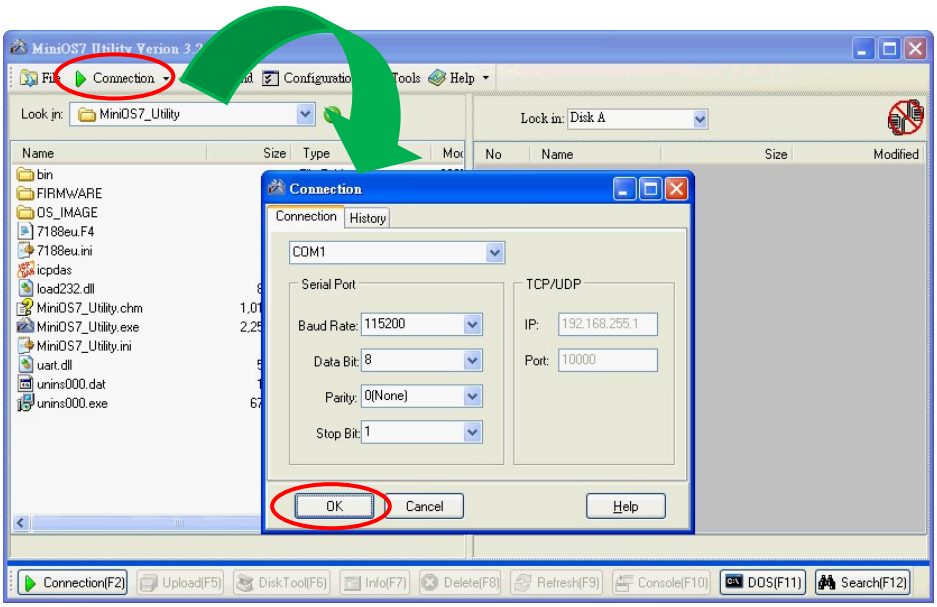


Step3: Start the MiniOS7 Utility tool

Double click the MiniOS7 Utility icon on the desktop to execute it.

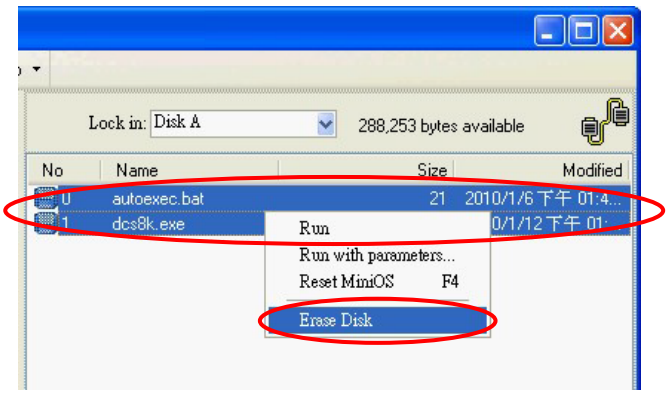


Step4: Connect to the FCM-MTCP

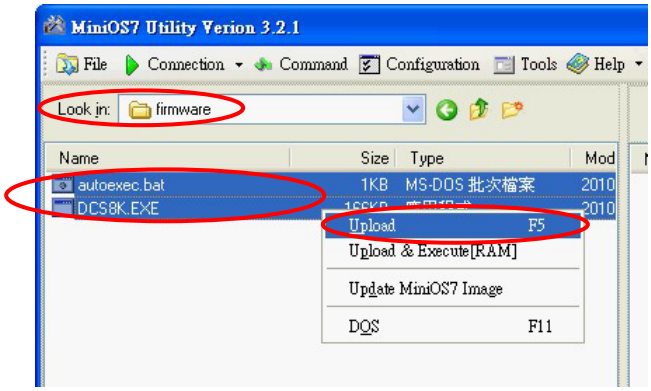


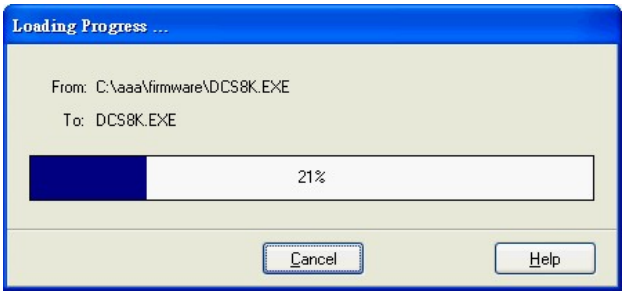
Step5: Update the FCM-MTCP firmware

On the right part of the MiniOS7 utility tool, drug and drop the mouse pointer to select all files, left-click at the selected files, and click "Erase Disk" to erase the firmware of the FCM-MTCP.



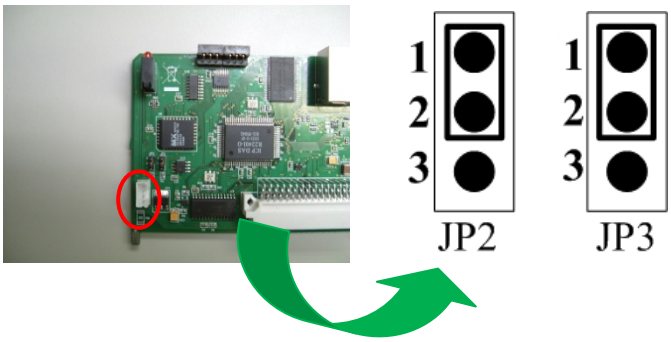
On the left part of the MiniOS7 utility tool, select the firmware folder which you want to update, drug and drop the mouse pointer to select all files, left-click at the selected files, and click "Upload" to update the firmware of the iDCS-8830.





Step6: Configure the FCM-MTCP to normal mode

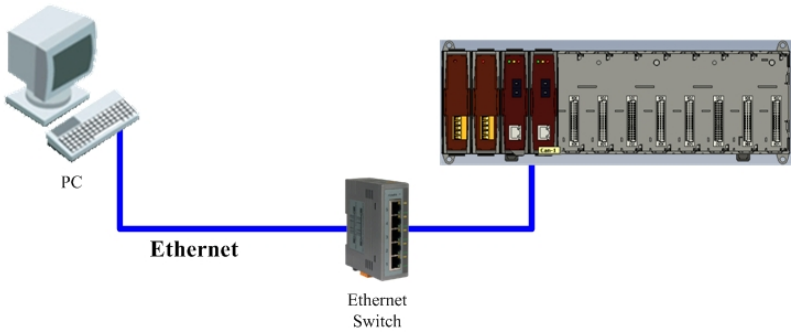
After finishing the update procedure, power off the FCM-MTCP, then select the jumper JP2 and JP3 as the following figure and remove the CA-0904 cable from JP1.



Update firmware via Ethernet

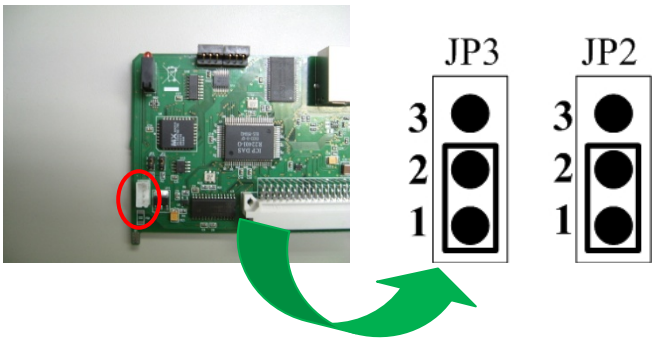
Step1: Connect PC with the iDCS-8830 via the Ethernet port

Power off the iDCS-8000, connect the PC with the FCM-MTCP via the Ethernet port, and power on the iDCS-8000. The Ethernet switch or hub may be needed while you connect the PC with the FCM-MTCP.

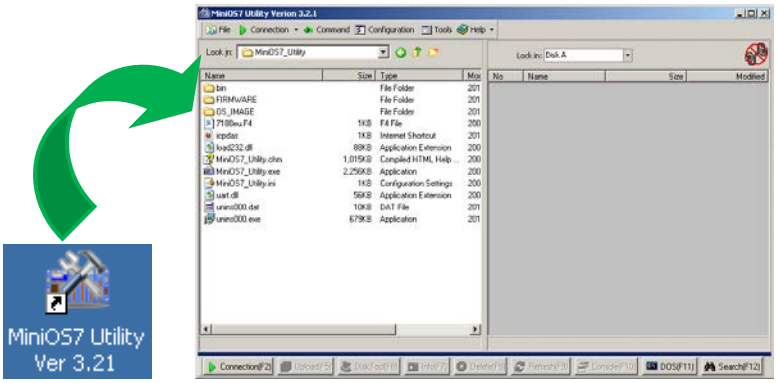


Step2: Configure the FCM-MTCP to initial mode

Select the jumper JP2 and JP3 as the following figure, and power on the iDCS-8000.

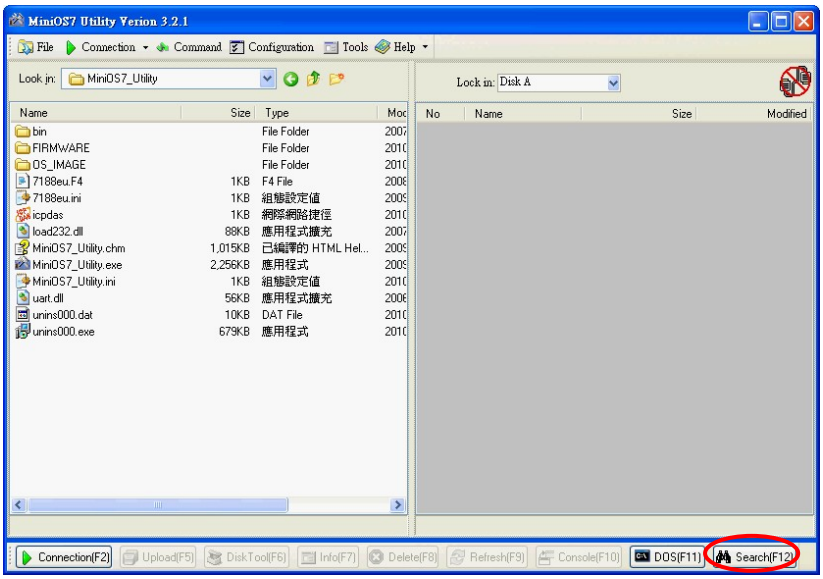


Step3: Start the MiniOS7 Utility tool

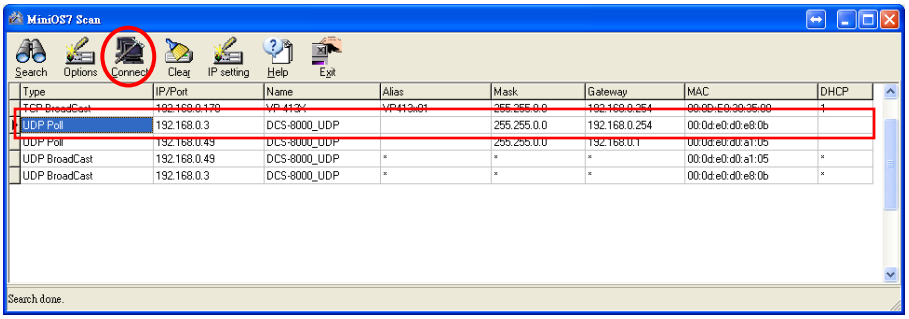


Step4: Search the FCM-MTCP device and modify the IP address

Click the "Search" button on the bottom left corner of the MiniOS7 utility tool.

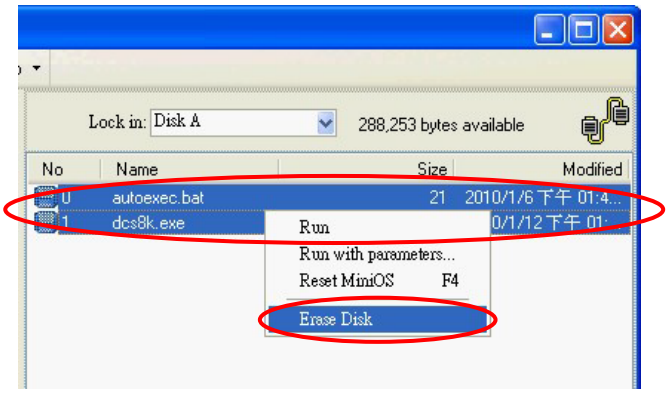


Select the FCM-MTCP by using current IP address, and click the "Connect" button.

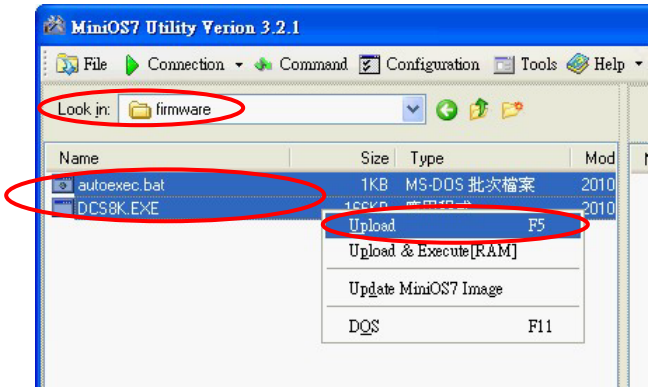


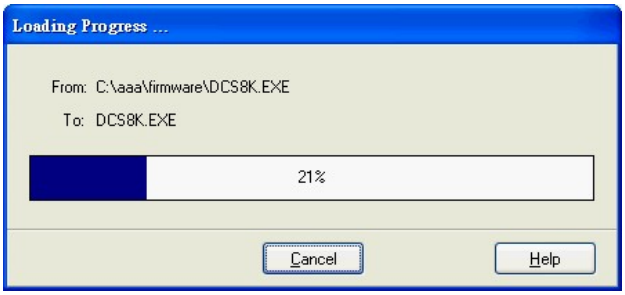
Step5: Update the FCM-MTCP firmware

On the left part of the MiniOS7 utility tool, drug and drop the mouse pointer to select all files, left-click at the selected files, and click "Erase Disk" to erase the firmware of the FCM-MTCP.



On the right part of the MiniOS7 utility tool, select the firmware folder which you want to update, drug and drop the mouse pointer to select all files, left-click at the selected files, and click "Upload" to update the firmware of the FCM-MTCP.





Step6: Setting the of FCM-MTCP to normal mode

After finishing the update procedure, power off the FCM-MTCP, then select the jumper JP2 and JP3 as the following figure.