# ICP DAS

# I-8000SeriesFamily (for C Language)

# User Manual

**Service and usage information for**

I-8411/I-8811

I-8431/I-8831

I-8431-80/I-8831-80

## Warranty

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

## Warning

ICP DAS assumes no liability for any damage 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, not 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.

## Contact US

If you have any problem, please feel free to contact us.

You can count on us for quick response.

Email: service@icpdas.com

# Table of Contents

# 1. I-8000 Family (C Language)

I-8000 Compact Embedded Controller Contents
The I-8000 is a new family of compact, modular, intelligent and rugged, distributed I/O (input/output) systems,

The 8000 controller comprises a central processor (CPU), power supply, I/O bus, communication interfaces, front panel control facilities and connectors to plug in various I/O modules. The 8000 can operate either as an intelligent front end, connected to a host machine under control of a SCADA package, or as an autonomous controller running an embedded application. In this mode it is possible to customize the unit to specific applications.

The main central processing module (CPM) is based on the powerful AMD 80188/80186 series of microprocessors. The CPM can be considered to be an integrated PC comprising up to 32Mbytes of Flash storage and up to 512K bytes of battery supported SRAM. The processor runs it's own version of DOS optimised for real time applications. The extension of the processor bus into the I/O bus permits high-speed data access from the high performance parallel I/O modules.

In addition to the standard network interface each CPM is supplied with two serial ports which can be configured to be either RS232 or 485. These ports can be used to extend the I/O capability of the unit using or alternatively programmed to drive external serial devices.

## 1.1. Introduction

The I-8000 are serial/Ethernet controllers designed for data acquisition, remote measurement, and control applications. Equipped with MiniOS7,an embedded OS, similar to DOS, can boot

The I-8000 series family are divided into the following .

| | I/O Slots | CPU | Ethernet | RS-232/RS-485 |
|---|---|---|---|---|
| I-8411 | 4 | 40 MHz | - | 4 |
| I-8811 | 8 | | | |
| I-8431 | 4 | | 1 (10-Base-T) | 3 |
| I-8831 | 8 | | | |
| I-8431-80 | 4 | 80 MHz | | |
| I-8831-80 | 8 | | | |

### 1.1.1. I-8411/I-8811

The I-8411/I-8811 are serial embedded controller with 4/8 I/O slots. Both are equipped with MiniOS7,an embedded OS similar to DOS that is developed by ICPDAS Co., LTD.

The MiniOS7 can boot up in a very short time (0.4 ~ 0.8 second).It has built-in hardware diagnostic function, and supports the full functions to access all 8K and 87K series I/O modules, such as DI, DO, DIO, AI, AO, Counter/Frequency, motion modules etc.

The I-8411/I-8811 back plane is equipped four serial COM Port, they include RS-232 and RS-485 ports and can be used in remote data acquisition and control applications, including environment monitoring, power management and factory automation. By using S-256 (256 KBytes) or S-512 (512 KBytes)battery backup SRAM, they provide data logger storage function.

Note: S256 and S512 are optional accessories.

### 1.1.2. I-8431/I-8831

The I-8431/ I-8831 are ethernet embedded controller with 4/8 I/O slots. Both are equipped with MiniOS7,an embedded OS similar to DOS that is developed by ICPDAS Co., LTD. The MiniOS7 can boot up in a very short time (0.4~0.8 second).It has built-in hardware diagnostic function, and supports the full functions to access all 8K and 87K series I/O modules, such as DI, DO, DIO, AI, AO, Counter/Frequency, motion modules etc.

To simplify the TCP/IP program developing process, we designed aXServer architecture. It is a reliable, opened, expandable, all purposed,and easily to be used library. The Xserver implements 90% functionalities of Ethernet communication. Refer the rich demo programs we provided, software engineer can easily finish the 10% remaining funtionalities and greatly shorten the developing time.

The I-8431/I-8831are equipped an ethernet port and their back planes are equipped three serial COM Port, they include RS-232 and RS-485 ports and can be used in remote data acquisition and control applications, including environment monitoring, power management and factory automation. By using S-256 (256 KBytes) or S-512 (512 KBytes)battery backup SRAM, they provide data logger storage function.

Note: S256 and S512 are optional accessories.

### 1.1.3. I-8431-80/I-8431-80

The I-8431-80/I-8831-80 are ethernet embedded controller with 4/8 I/O slots. Both are equipped with MiniOS7,an embedded OS similar to DOS that is developed by ICPDAS Co., LTD. The MiniOS7 can boot up in a very short time (0.4~0.8 second).It has built-in hardware diagnostic function, and supports the full functions to access all 8K and 87K series I/O modules, such as DI, DO, DIO, AI, AO, Counter/Frequency, motion modules etc.

To simplify the TCP/IP program developing process, we designed aXServer architecture. It is a reliable, opened, expandable, all purposed,and easily to be used library. The Xserver implements 90% functionalities of Ethernet communication. Refer the rich demo programs we provided, software engineer can easily finish the 10% remaining funtionalities and greatly shorten the developing time.

The I-8431-80/I-8831-80 are equipped an ethernet port and their back planes are equipped three serial COM Port, they include RS-232 and RS-485 ports and can be used in remote data acquisition and control applications, including environment monitoring, power management and factory automation. By using S-256 (256 KBytes) or S-512 (512 KBytes)battery backup SRAM, they provide data logger storage function.

Note: S256 and S512 are optional accessories.

## 1.2. Features

I-8000 offers the most comprehensive configuration to meet specific application requirements. The following list shows the hardware and software features designed to simplify installation, configuration and application.

### ➤ Serial port based embedded controller

The i-8411/i-8811 are serial embedded controller that can allow COM port application to access and control remote I/O in RS-232 or industrial RS-485 network.

### ➤ Equipped with MiniOS7 (DOS like OS)

The I-8000 is equipped with the MiniOS7, a friendly DOS like OS. Users can use C compilers that can create 16 bit executable file (*.exe) to develop their own program then download it to I-8000.

### ➤ Provides API Functions for：i-8K, i-87K I/O, 7-SEG LED display, RTC (Real Time Clock), EEPROM, and more

The I-8000 supports the MiniOS7 API functions that has hundreds of pre-defined functions, such as I-8000, i-87K I/O, 7-SEG LED, RTC, EEPROM,… etc, and provides mostly required for users to program own applications.

### ➤ Upgrading the firmware, and downloading programs via the RS-232 port

➤ When should the firmware be updated?

Firmware should be updated when ICPDAS announces.

- Support for new I/O modules
- The addition of new functions
- Bug fixes and revision

The i-8411/i-8811's COM1 port can be used to download programs, and update the MiniOS image file.

## ➤ Hardware designs to protect software

The i-8411/i-8811 is equipped with a unique 64-bit hardware serial number. User’s software can check this number to avoid illegal software copy. Another method to archive the goal, users can choose the ASICKey method. The ASICKey can be numbered from 00~ 99. Each number only sold to one customer. User’s software can check specific ASICKey number to decide whether quit or run forward.

**64-bit hardware serial number (built in)**
**ASICKey (optional)**

For more information about the 64-bit hardware serial number and AsicKey, please refer to “Appendix G. How to avoid illegal software copy”.

## ➤ Innovation design on reliability、flexibility、expansibility and stretch

The i-8411/i-8811 is equipped with 4/8 I/O slots and multi-serial ports. It not only supports i-8K and i-87K series I/O modules such as DI, DO, DIO, AI, AO and Counter/Frequency for I/O slots application, but also the I-7K series I/O modules for a wide range of RS-485 network application.

Each I/O modules allows various numbers of channels. For example, with i-8040 or i-8041, the i-8810 provides max 256 digital input or digital output channels.

For more information on i-8K and i-87K series modules, please refer to “Appendix B. i-8K and i-87K series I/O modules”.

## ➤ Built-in watchdog circuit

The built-in watchdog circuit will reset the CPU module if a failure occurs in either the hardware or software. If the application program does not refresh the watchdog timer within 0.8 sec, the watchdog circuit will initiate a reset of the CPU.

## ➤ Input Protection circuitry

The protection circuitry on both the network and power supply protects the system from external signals such as main spikes and ambient electrical noise. In addition, the central processing module is isolated from external signals in three ways. This is achieved through I/O isolation of 3KV, power isolation to 3KV and network isolation to 2KV.

### ➤ High performance integrated power supply

The built-in isolated 20W power supply is rated to perform linearly up to full loading.

### ➤ Ventilated housing design to work between -25 °C ~ +75°C

The i-8411/i-8811 is housed in a plastic base box with a column-like ventilator that can help to cool the work environment inside the box and allow the i-8411/i-88111 to operator between -25 °Cand +75°C.

## 1.3. Specification

The table below summarizes the specifications of I-8000, and lists the accessories that I-8000 supports.

| Product | I-8411 | I-8811 | I-8431 | I-8831 | I-8431-80 | I-8431-80 |
|---|---|---|---|---|---|---|
| **CPU Module** | | | | | | |
| CPU | 80188 or compatible<br>(8-bit and 40 MHz) | | | | 80186 or compatible<br>(16-bit and 80 MHz) | |
| SRAM | 512 K Bytes | | | | | |
| Flash | 512 K Bytes | | | | | |
| EEPROM | 2 K Bytes | | | | | |
| NVRAM | Yes | | | | | |
| RTC (real time clock) | Yes | | | | | |
| Hardware Serial Number | Yes | | | | | |
| Built-in Watchdog Timer | Yes | | | | | |
| **Communicate Interface** | | | | | | |
| COM0 | Internal Communication with the 87K modules | | | | | |
| COM1 | RS-232 (to update firmware) | | | | | |
| COM2 | RS-485 | | - | | | |
| COM3 | RS-232/RS-485 | | | | | |
| COM4 | RS-232 | | | | | |
| Ethernet Port | - | | 10 Base NE2000 compatible | | | |
| **SMMI** | | | | | | |
| LED Display | Yes | | | | | |
| Programmable LED Indicators | Yes | | | | | |
| 4-Push Buttons | Yes | | | | | |
| **I/O Expansion Slots** | | | | | | |
| Slot Number | 4 slots | 8 slots | 4 slots | 8 slots | 4 slots | 8 slots |
| **Mechanical** | | | | | | |
| Dimensions (W x L x H) | 230 x 110 x 75.5 | 354 x 110 x 75.5 | 230 x 110 x 75.5 | 354 x 110 x 75.5 | 230 x 110 x 75.5 | 354 x 110 x 75.5 |
| **Operating Environment** | | | | | | |
| Operating Temperature | -25 ~ +75 °C | | | | | |
| Storage Temperature | -30 ~ +80 °C | | | | | |
| Humidity | 10 ~ 90 % RH, non-condensing | | | | | |

| Product | I-8411 | I-8811 | I-8431 | I-8831 | I-8431-80 | I-8431-80 |
|---|---|---|---|---|---|---|
| **Power** | | | | | | |
| Protection | Power reverse polarity protection | | | | | |
| Power requirement | +10 ~ +30 $V_{DC}$ | | | | | |
| Power Supply | 20 W | | | | | |
| Power Consumption | 3.9 W | 5.1 W | 3.9 W | 5.1 W | 3.9 W | 5.1 W |

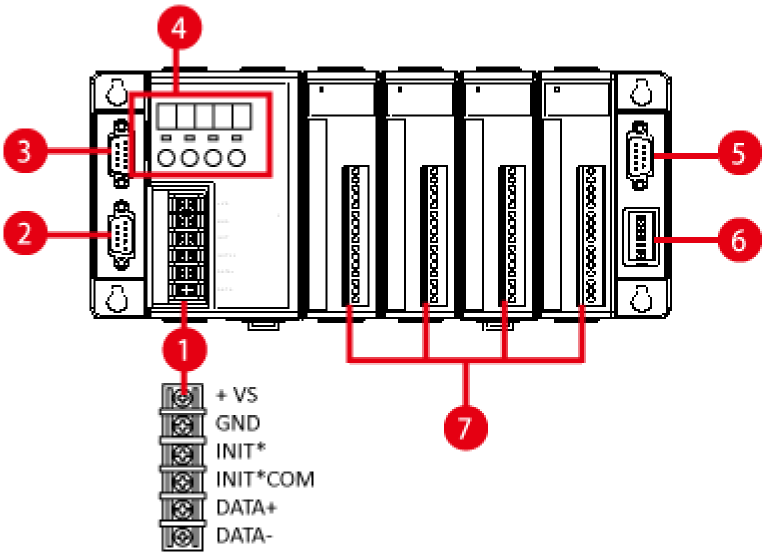
## Accessories

| Model | |
|---|---|
| S256 | 256 K battery backup SRAM module |
| S512 | 512 K battery backup SRAM module |
| KA-52F | AC 100 ~ 250 V input, DC 24V/1A output, flat type power supply |
| DIN-KA52F | KA-52F with DIN-Rail mount |
| DP-660 | 24 VDC/2.5 A, 60 W and 5 VDC/0.5 A, 2.5 W Power Supply with DIN-Rail Mounting |
| DP-665 | AC 85 ~ 270 V input, DC 24V/1.7A and 5V/0.5A output power supply |
| DP-1200 | 24 VDC/2.5 A, 60 W and 5 VDC/0.5 A, 2.5 W Power Supply with DIN-Rail Mounting |

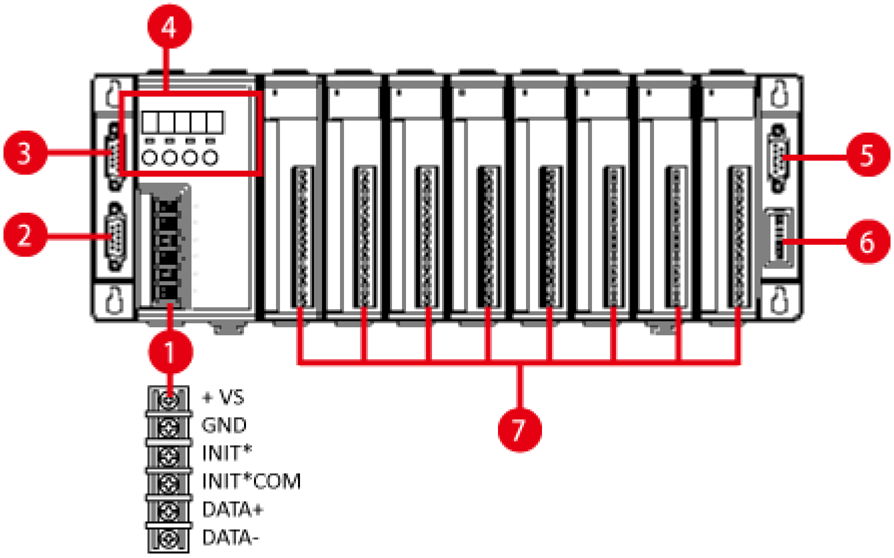
# 1.4. Overview

I-8000 consists of several different components that integrate with ICP DAS system. Here is an overview of the components and its descriptions. The following list shows the details of the components:

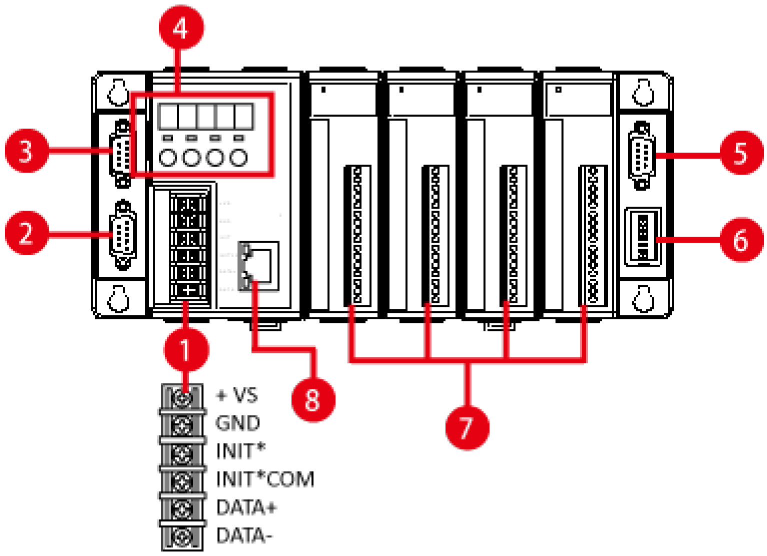
## ➤ I-8411



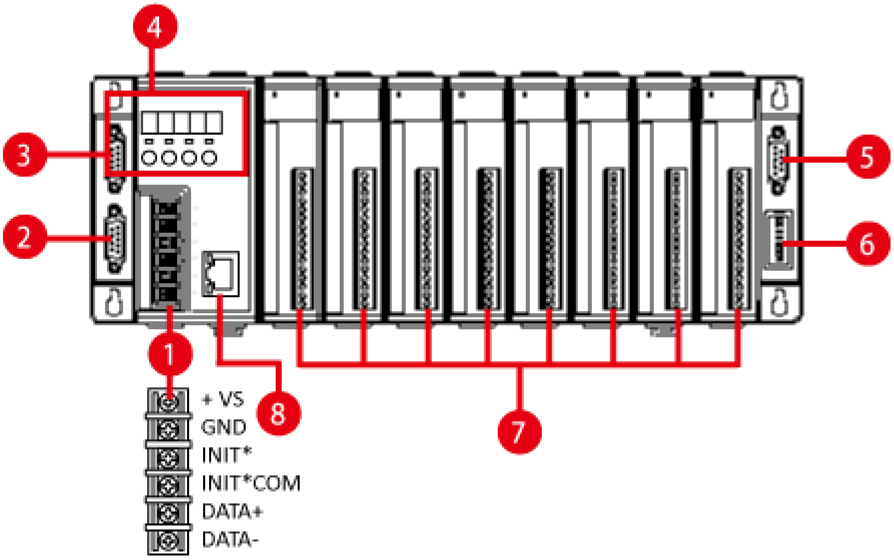
## ➤ I-8811



## ➤ I-8431/I-8431-80

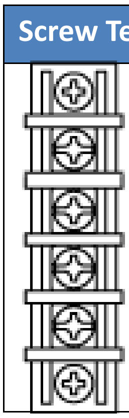


## ➤ I-8431/I-8431-80

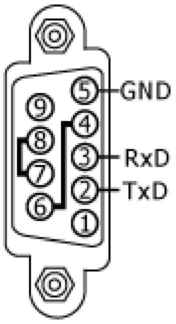


### Terminial Block

The I-8000 has a screw terminal block with 6 screw-downs. The table below describes the terminal block designations and its functions.

| Screw Terminal | | Signal | Description |
|---|---|---|---|
|  | 1 | +VS | Power Input |
| | 2 | GND | |
| | 3 | INIT* | Initial Mode |
| | 4 | INIT*COM | |
| | 5 | DATA+ | COM2 (RS-485) |
| | 6 | DATA- | |

### COM1 (RS-232)



**Port Type**: Female

**Baud Rate**: 115200, 57600, 38400, 19200, 9600, 4800, 2400, 1200 bps

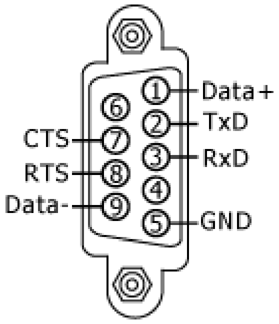
**Data Bits**: 7, 8

**Parity**: None, Even, Odd

**Stop Bits**: 1

**FIFO**: 1 byte

### COM3 (RS-232/RS-485)



**Port Type**: Male

**Baud Rate**: 115200, 57600, 38400, 19200, 9600, 4800, 2400, 1200 bps

**Data Bits**: 5, 6, 7, 8

**Parity**: None, Even, Odd, Mark (Always 1), Space (Always 0)

COM3 can be configured as either RS-232 or RS-485, that only can select one at a time and its configuration depends on the pin connections as follows:

**RS-232** (RXD, TXD, CTS, RTS and GND)

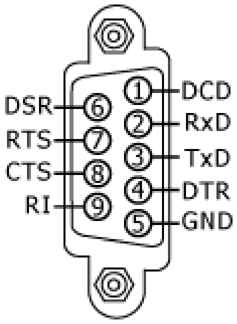
**RS-485** (Data+ and Data-)

There is no software configuration or hardware jumper needed.

### SMMI (Small Main-Machine Interface)

SMMI consists of 5-digital LED displays, 4 programmable LEDs, and 4 push buttons,

### COM4 (RS-232)



**Port Type**: Male

**Baud Rate**: 115200, 57600, 38400, 19200, 9600, 4800, 2400, 1200 bps

**Data Bits**: 5, 6, 7, 8

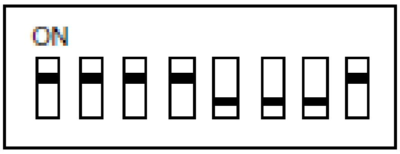
**Parity**: None, Even, Odd, Mark (Always 1), Space (Always 0)

**Stop Bits**: 1, 2

**FIFO**: 16 bytes

### DIP Switch

The DIP switch can be used to set the Module ID to a number from 0 to 255. Do not use Module ID 0 for communication.



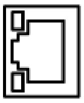
### Expansion I/O Slots

I-8000 are equipped with some extra I/O slots, that can be used to expand capabilities of I-8000. The number of the expansion I/O slot in each type of I-8000:

I-8411/I-8431/I-8431-80: 4 I/O slots

I-8811/I-8831/I-8831-80: 8 I/O slots

### Ethernet Port (for I-8431/I-8831/I-8431-80/I-8831-80 only)



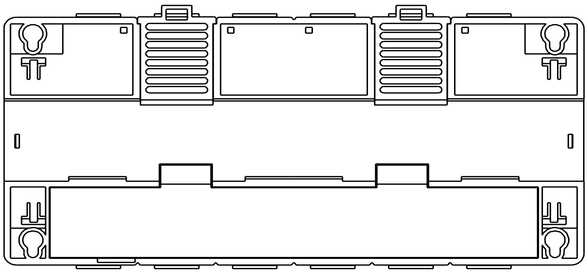
The Ethernet port can be used to connect a PC or other networked controller. Each Ethernet port has two LED indicators, which are used to indicate the network speed and Link/Acting, as described below.

# 1.5. Dimension

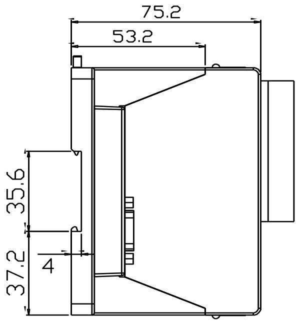
The following information provides the dimensions of I-8000 to use in defining your enclosure specifications. Remember to leave room for potential expansion if you are using other components in your system.

The height dimension is the same for all I-8000 family. The width depending on your choose of I/O expansion slots. All dimensions are in millimeters.

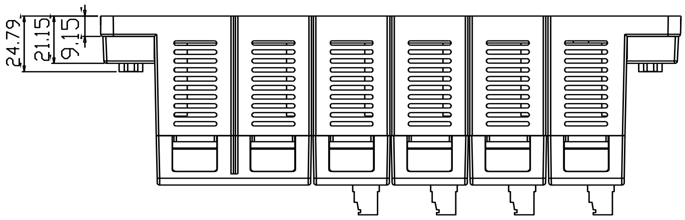
### 1.5.1. I-8411/I-8431/I-8431-80



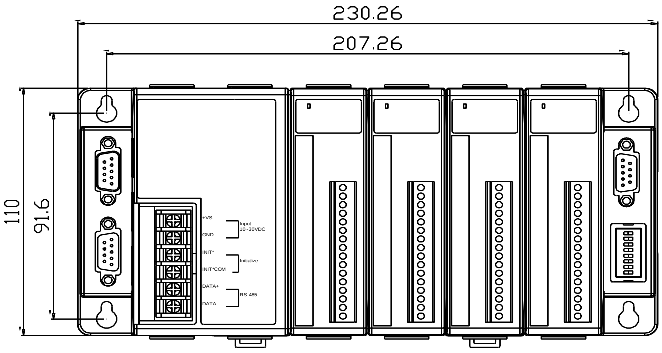
Back View



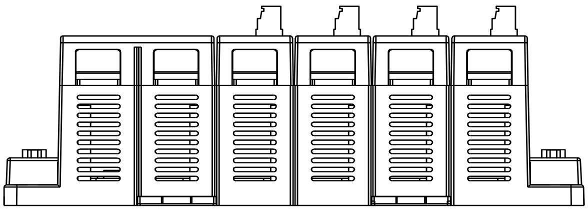
Side View



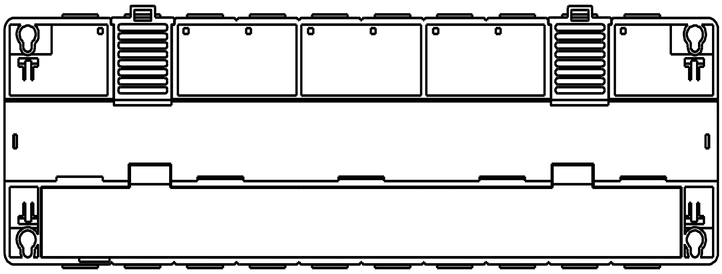
Top View



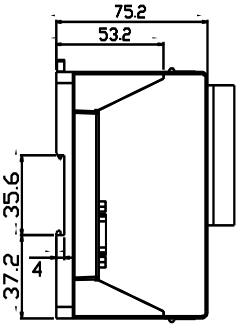
Front View



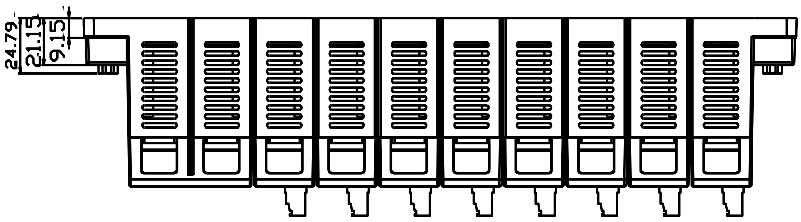
## 1.5.2. I-8411/I-8831/I-8831-80



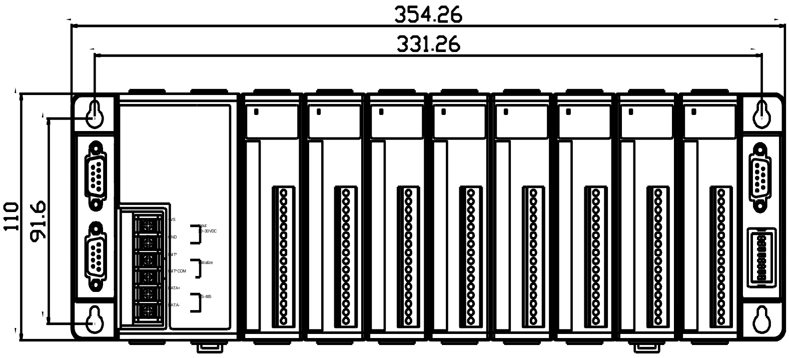
Back View



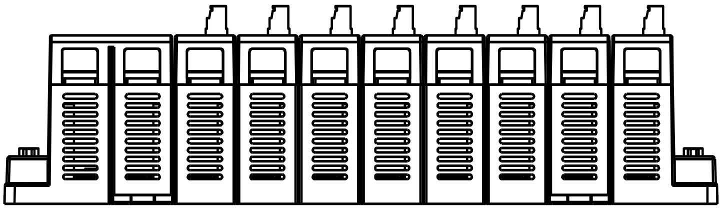
Side View



Top View



Front View



# 2. Getting Started

If you are a new user, begin with this chapter, it includes a guided tour that provides a basic overview of installing, configuring and using the I-8000.

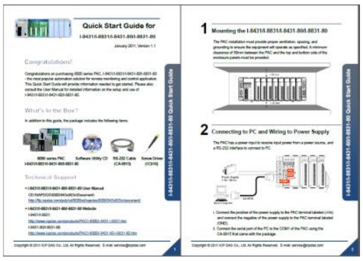
Before starting any task, please check the package contents. If any of the following package contents are missing or damaged, contact your dealer, distributor.

**I-8000 Series Module**

I-8411/I-8811

I-8431/I-8831

I-8431-80/I-8831-80



Quick Start Guide for

I-8431/I-8831/I-8431-80/I-8831-80

January 2011, Version 1.1

Congratulations!

What's in the Box?

In addition to this guide, the package includes the following items:

8000 series PAC — Software Utility CD — RS-232 Cable — Screw Driver

Technical Support

1 Mounting the I-8431/I-8831/I-8431-80/I-8831-80

2 Connecting to PC and Wiring to Power Supply

The PAC has a power input to receive input power from a power source, and a RS-232 interface to connect to PC.

**Release Note**

**RS-232 Cable**

(CA0910)

**Screw Driver**

(1C016)

# 2.1. Mounting the Hardware

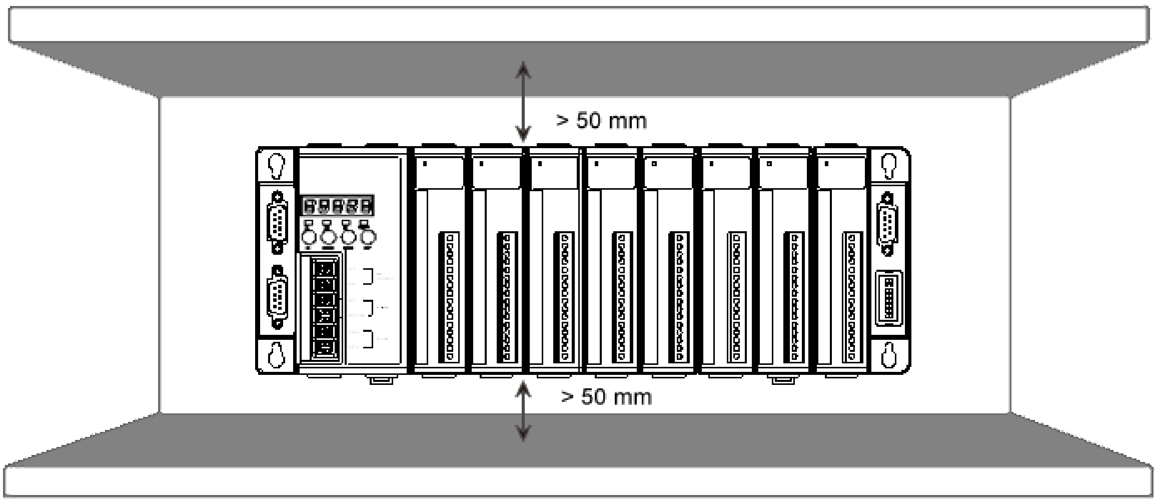
Before you work with I-8000, you should have a basic understanding of the hardware specifications, such as the dimensions, the usable input-voltage range of the power supply, and the type of communication interfaces.

For more information about the hardware details, see section 1.2., “Specifications.”

For more information about the hardware dimensions, see section 1.4., “Dimension.”

## 2.1.1. Mounting the I-8000

I-8000 can be mounted with the bottom of the chassis in the standard 35 mm DIN rail, or any other screw-mountable surface.It is necessary that a minimum clearanceof 50mm between the I-8000 and the top and bottom side of the enclosure panels.



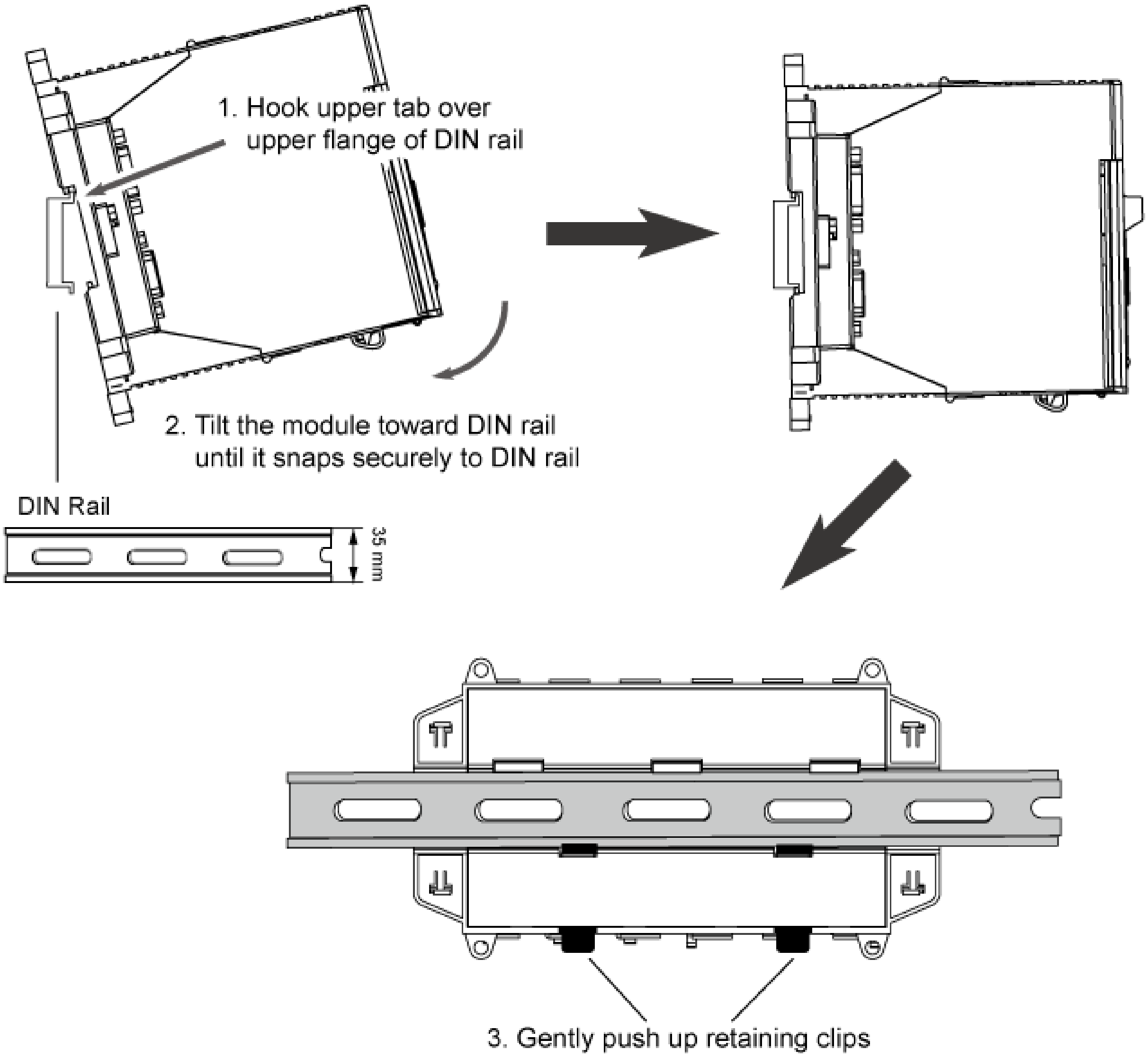
There are two ways for mounting the I-8000.

### 1. To mount the I-8000 on a DIN rail

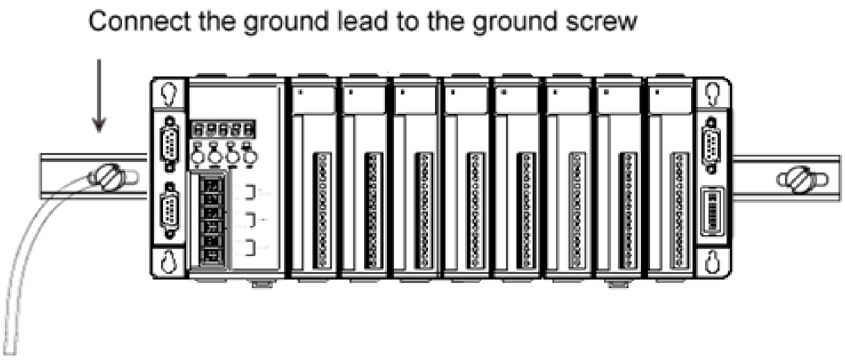
i. Hook upper tab over upper flange of DIN rail

ii. Tilt the module toward DIN rail until it snaps securely to DIN rail

iii. Push up retaining clips


1. Hook upper tab over upper flange of DIN rail
2. Tilt the module toward DIN rail until it snaps securely to DIN rail
DIN Rail
35 mm
3. Gently push up retaining clips

Tips & Warnings



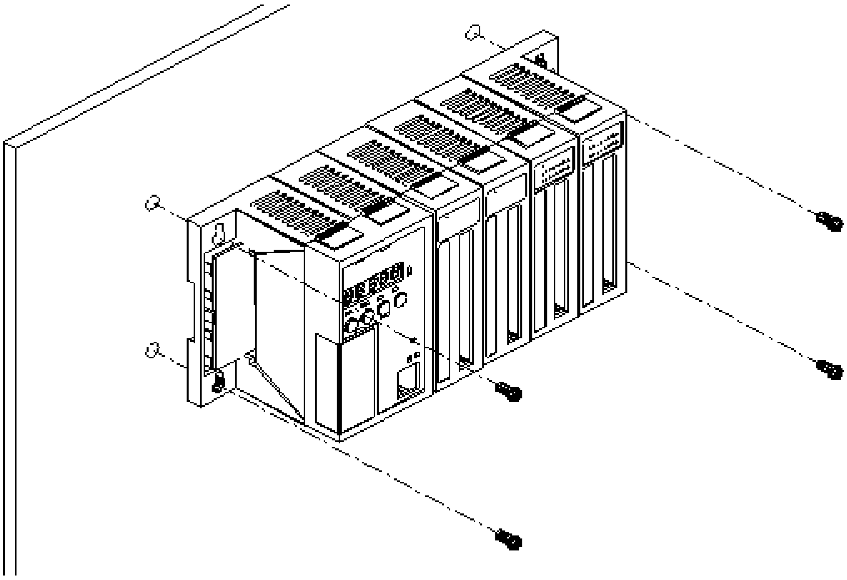
**Grounding**

A good common ground reference (earth ground) is essential for proper operation of the I-8000. One side of all control circuits, power circuits and the ground lead must be properly connected to earth ground by either installing a ground rod in close proximity to the enclosure or by connecting to the incoming power system ground. There must be a single-point ground (i.e. copper bus bar) for all devices in the enclosure that require an earth ground.

## 2. To mount the I-8000 on a surface

i. Install the four mounting screws into the 4 keyhole mounting holes

ii. Fasten the screws securely



### 2.1.2. Wiring the I-8000

The package includes a RS-232 cable for connecting the I-8000 to a PC/Laptop. The I-8000 has the power supply interface for supplying power from the power supply.

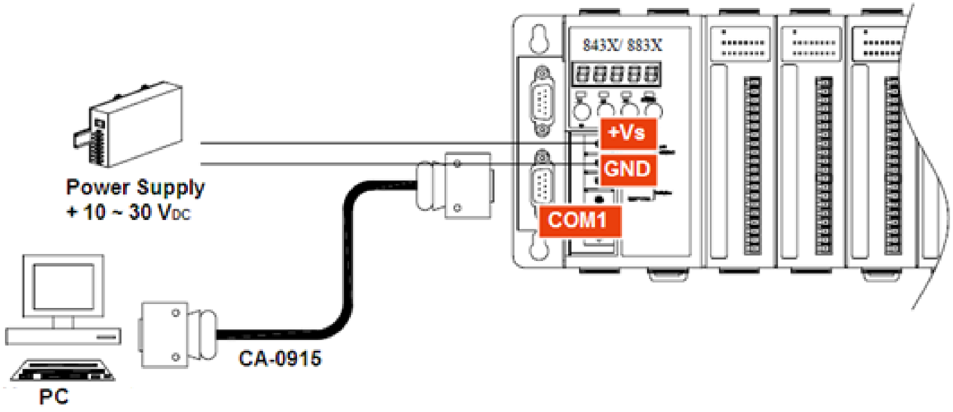
**Step 1: Connect to the power supply**

The I-8000 requires a 10 to 30 $V_{DC}$ power supply to operate.

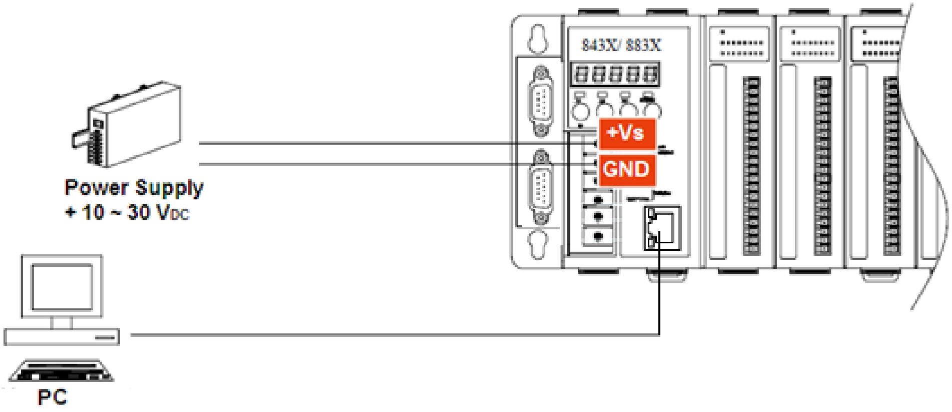
i. Connect the +Vs of the I-8000 to the positive of the power supply.
ii. Connect the GND of the I-8000 to the negative of the power supply.

**Step 2:Connect to a PC**

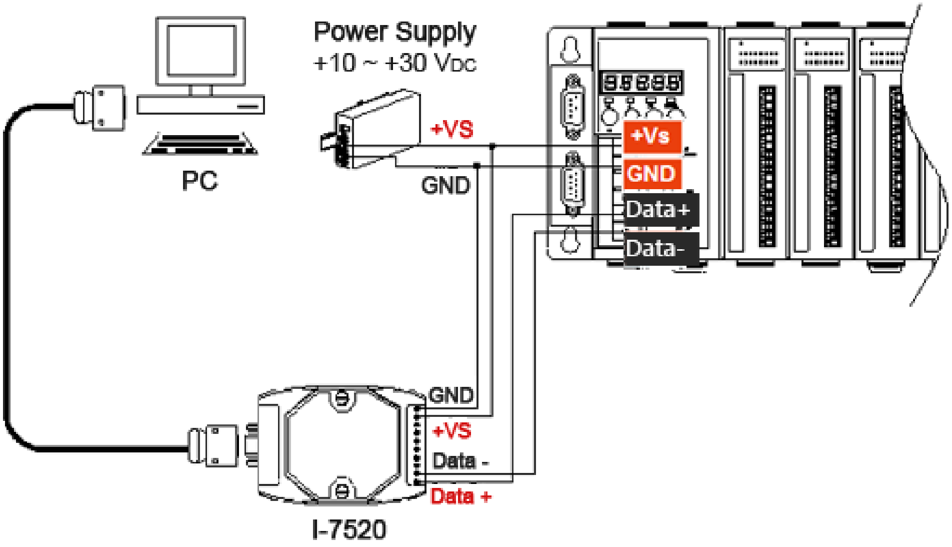
Connect the COM1 of the I-8000 to a PC using a RS-232 cable that was provided with the package.



If your I-8000 are Ethernet controller, you also can connect to PC by using an ethernet cable.



If PC/Laptop has no COM port, you can use the I-7560 (USB to RS-232 converter) for connection between I-8000 and PC/Laptop.

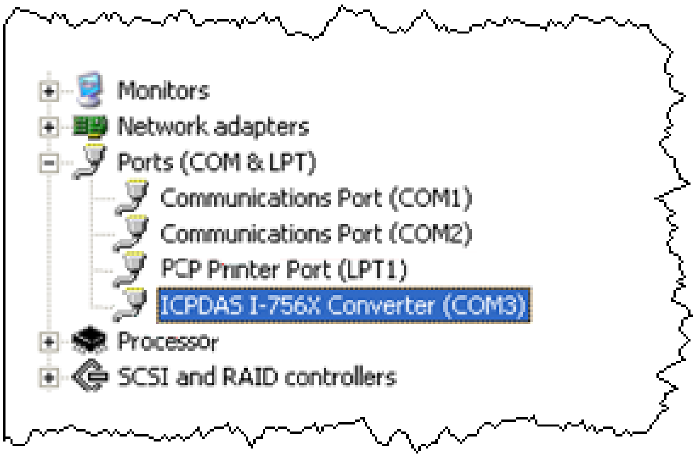


Before using the I-7560 converter, you must install the USB driver.

The USB driver can be obtained from:

https://www.icpdas.com/en/download/index.php?model=I-7561

After installing the USB driver, please check the "Device Manager to make sure the driver has been installed and the COM port number which is assigned to USB interface.


Monitors
Network adapters
Ports (COM & LPT)
Communications Port (COM1)
Communications Port (COM2)
PCP Printer Port (LPT1)
ICPDAS I-756X Converter (COM3)
Processor
SCSI and RAID controllers

## 2.1.3. Inserting I/O Modules

I-8000 has 4/8 I/O expansion slots to expand the functions of I-8000, allowing it to communicate with external I/O modules, and before choosing the right I/O modules; you first need to know the I/O expansion capacities in order to choose the best expansion module for achieving maximal efficiency.

There are more than 30 high profile I/O modules available for interfacing many different measurements, including thermocouple, voltage, RTD, current, resistance, strain, digital,…, etc., and these modules have their own manuals, so if you are using them you should supplement this manual with the manual specifically designed for the special module.

For more information about the I/O expansion modules that are compatible with I-8000, see

https://www.icpdas.com/en/product/guide+Remote  I O  Module  and  Unit+PAC  %EF%BC %86amp;  Local  I O  Modules+I-8K I-87K  Series  (High  Profile)

**Step 1: Read the I/O user manual**

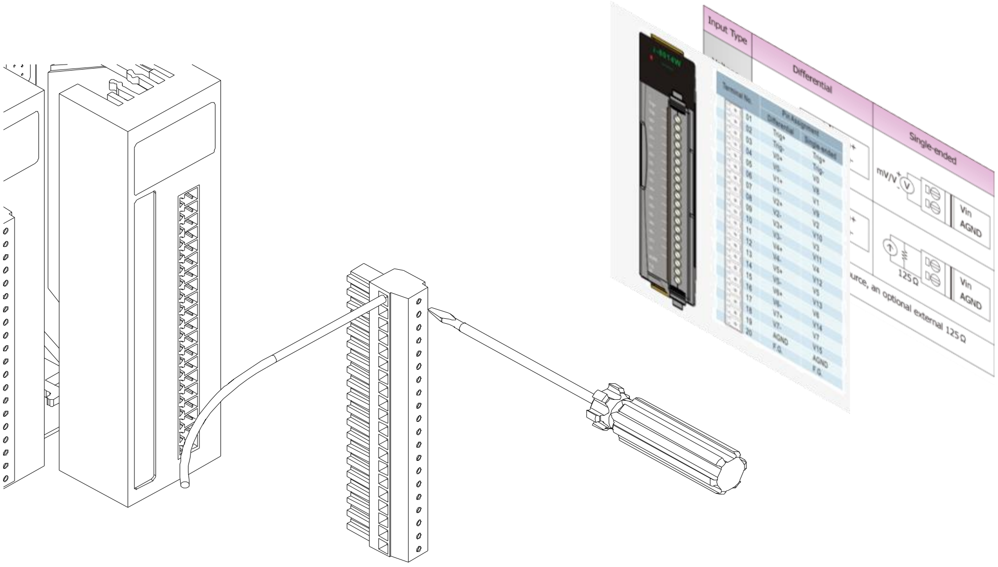
These modules have their own manuals, so if you are using them you should supplement this manual with the manual specifically designed for the special module.

The I/O user manuals can be found separately on the CD that was provided with the package or by downloading the latest version from ICP DAS web site.

https://www.icpdas.com/en/product/guide+Remote  I O  Module  and  Unit+PAC  %EF%BC %86amp;  Local  I O  Modules+I-8K I-87K  Series  (High  Profile)

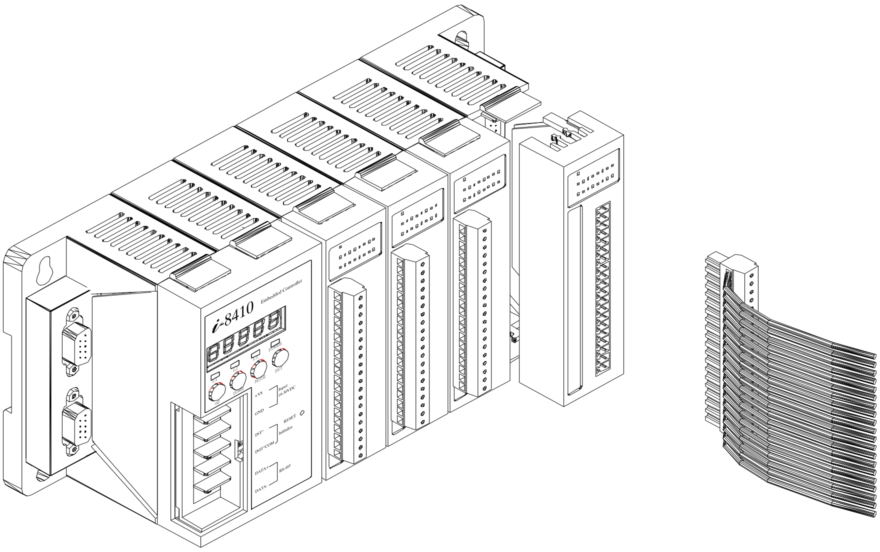
### Step 2: Wiring the I/O module

All I/O user manuals include the I/O module specifications, pin assignments, wire connections.



### Step 3: Insert the I/O module into I-8000

Align circuit card with slot and press firmly to seat module into connector and then Pull top and bottom locking tabs toward module face. Click indicates lock is engaged



# 2.2. Installing Software

The Companion CD includes complete sets of APIs, demo programs and other tools for developing your own applications.

## 2.2.1. Installing the I-8000 header and libraries files

The I-8000 header and libraries files can be downloaded from ICP DAS web site.

https://www.icpdas.com/en/download/show.php?num=2778&model=I-8831

## 2.2.2. Installing the MiniOS7 Utility

MiniOS7 Utility is a suite of tool for managing MiniOS7 devices (I-8000, µPAC-5000, iPAC-8000, µPAC-7186,. etc.). It’s comprised of four components – System monitor, communication manager, file manager and OS loader.

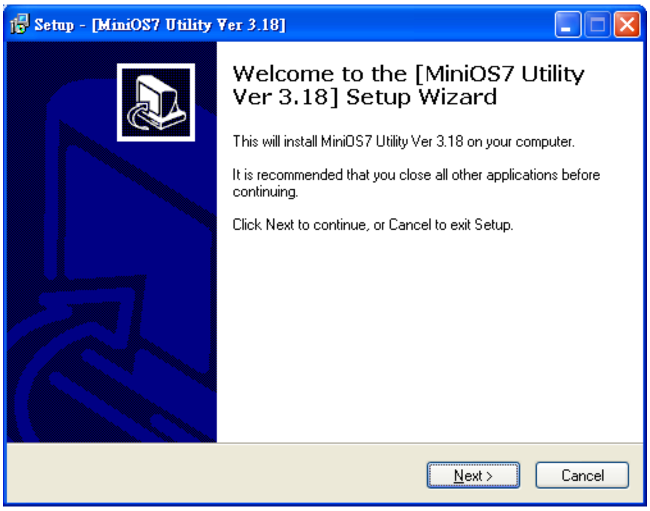
### Step 1: Get the MiniOS7 Utility

The MiniOS7 Utility can be found from ICP DAS web site.

https://www.icpdas.com/en/product/guide+Software+Development__Tools+MiniOS7

### Step 2: Follow the prompts to complete the installation

After the installation has been completed, there will be a new short-cut for MiniOS7 Utility on the desktop.



Setup - [MiniOS7 Utility Ver 3.18]

Welcome to the [MiniOS7 Utility Ver 3.18] Setup Wizard

This will install MiniOS7 Utility Ver 3.18 on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

## 2.2.3. Installing the VxComm Utility

The VxComm Driver/Utility can help users to create virtual COM ports in Windows and map them to the serial ports of the tDS/7188E/8000E/DS/PDS/iDS series device servers through the Internet/Ethernet network. User's RS-232/485 client programs need only to change to the virtual COM port to access the serial devices connected to the device servers through the network.

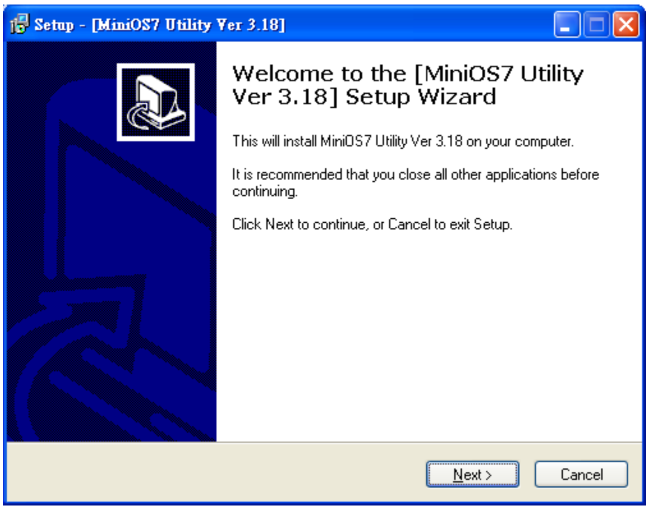
### Step 1: Get the VxComm Utility

The VxComm Utility can be found from ICP DAS web site.

https://www.icpdas.com/en/product/guide+Software+Utility_Driver+VxComm__(Virtual__COM)

### Step 2: Follow the prompts to complete the installation

MiniOS7 Utility
Ver 3.18


Setup - [MiniOS7 Utility Ver 3.18]

Welcome to the [MiniOS7 Utility Ver 3.18] Setup Wizard

This will install MiniOS7 Utility Ver 3.18 on your computer.

It is recommended that you close all other applications before continuing.

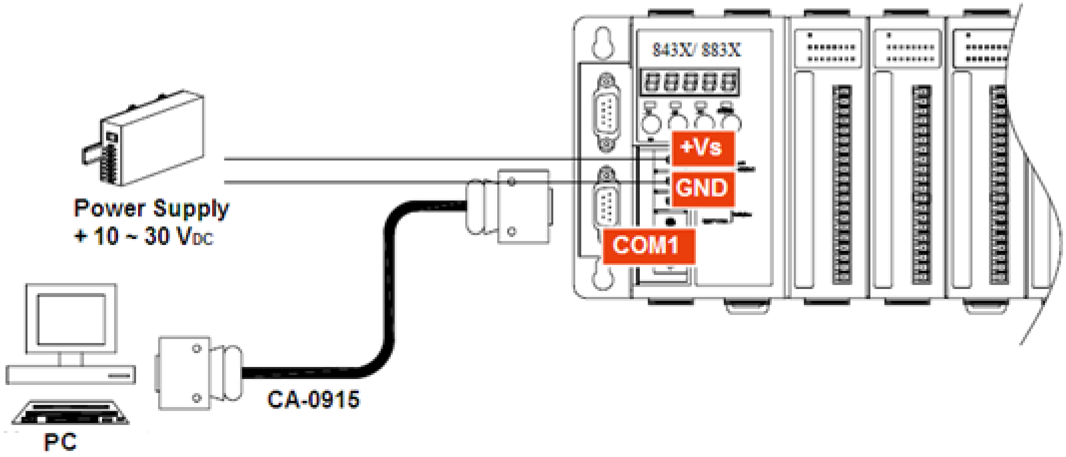
Click Next to continue, or Cancel to exit Setup.

# 2.3. Configuring the Boot Mode

The I-8000 has two modes of operation, normal mode and initializemdoe.

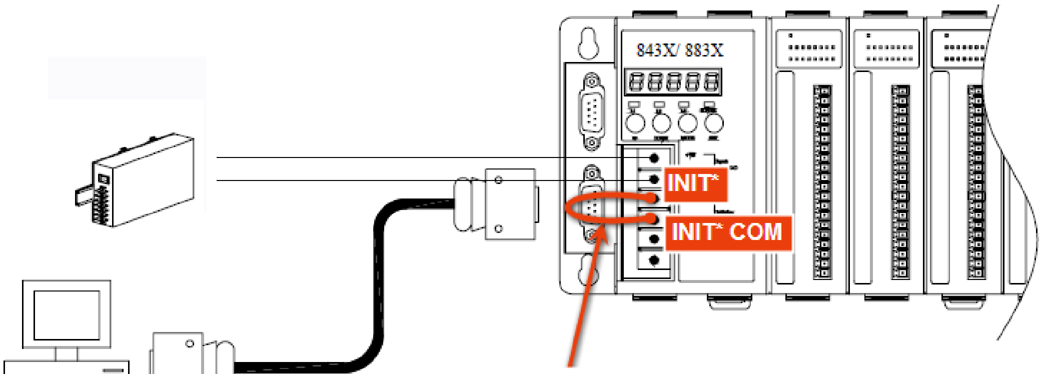
### Normal Mode

Normal mode is the default mode of operation and the one you will use most of the time. Use this mode for more tasks and configurations. Programs also are executed in this mode.



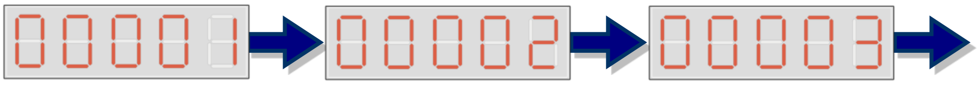
### Initial Mode

Initial mode is used to update the OS image and upgrade the firmware. Short INIT* and INIT* CON then reset the power, the I-8000 will enter OS operation mode.



Short INIT* & INIT* COM together to disable AUTOEXEC.BAT in the beginning stage of power up.

in this mode the 5-digital 7-SEG LEDs will count the number as shown below:



## 2.4. Assigning a New IP Address (for Ethernet Conntrolleronly)

The I-8431/I-8831/I-8431-80/I-8831-80 are Ethernet embedded controller, has which comes with a default IP address. If you want to add a I-8000 to your network, you must assign a new IP address, subnet mask, and gateway to your I-8000.

The factory default IP settings are as follows:

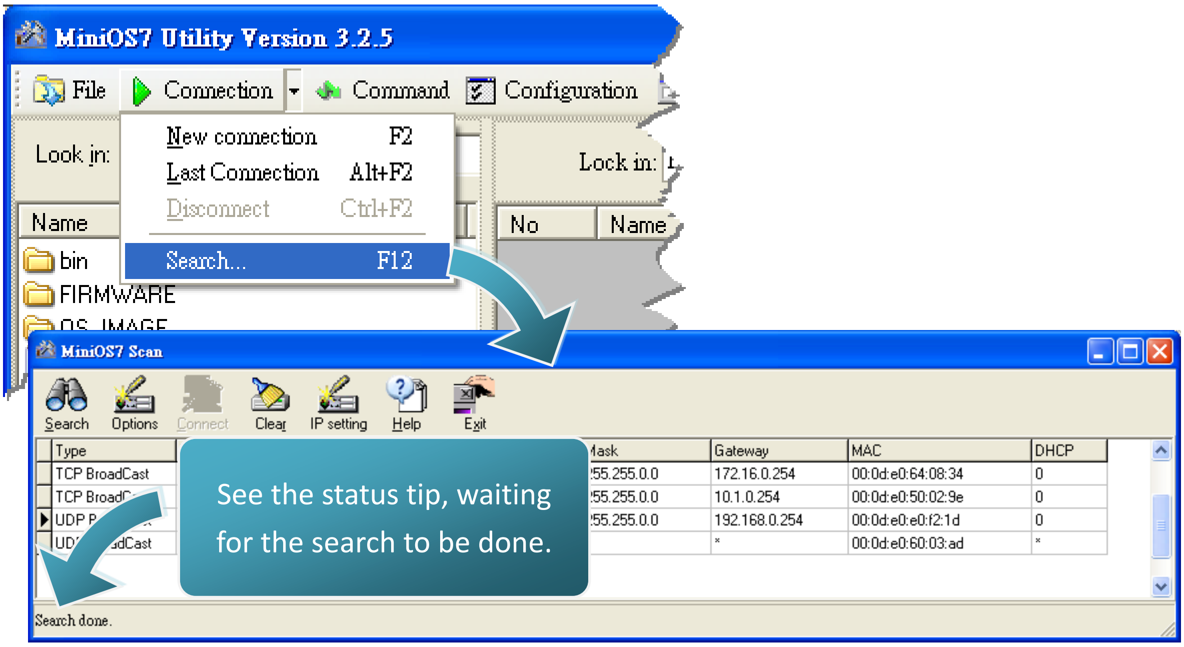
| Item | Default |
|---|---|
| IP Address | 192.168.255.1 |
| Subnet Mask | 255.255.0.0 |
| Gateway | 192.168.0.1 |

MiniOS7 Utility is an easy-to-use network configuration Utility. You can use it to configure the IP address of MiniOS7 devices easily.

**Step 1: Run the MiniOS7 Utility**

### Step 2: Click the “Search” from the “Connection” menu

you need to wait for the process to be done.

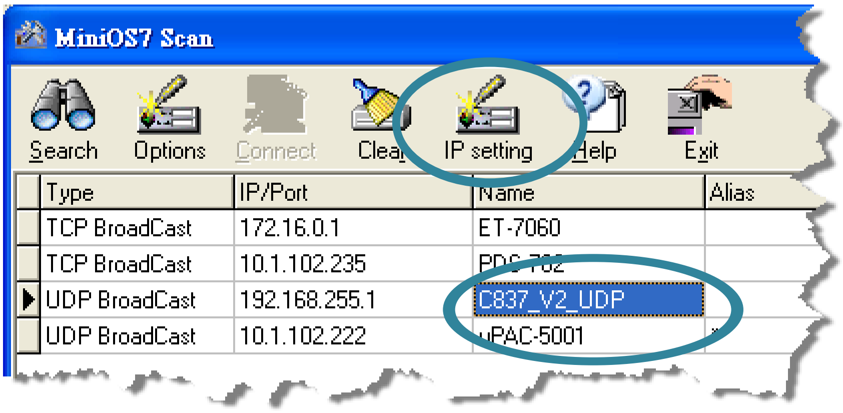


| Type | Mask | Gateway | MAC | DHCP |
|---|---|---|---|---|
| TCP BroadCast | 255.255.0.0 | 172.16.0.254 | 00:0d:e0:64:08:34 | 0 |
| TCP BroadC | 255.255.0.0 | 10.1.0.254 | 00:0d:e0:50:02:9e | 0 |
| UDP P | 255.255.0.0 | 192.168.0.254 | 00:0d:e0:e0:f2:1d | 0 |
| UD dCast |  | * | 00:0d:e0:60:03:ad | * |

See the status tip, waiting for the search to be done.

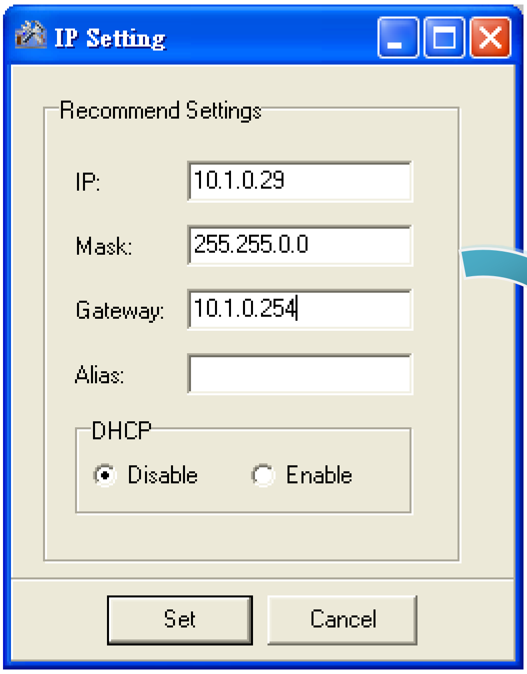
Search done.

### Step 3: Choose the OS image name of I-8000 module (which comes with a default IP address “192.168.255.1”) from the list and then click the “IP setting” from toolbar



| Type | IP/Port | Name | Alias |
|---|---|---|---|
| TCP BroadCast | 172.16.0.1 | ET-7060 |  |
| TCP BroadCast | 10.1.102.235 | PDS-702 |  |
| UDP BroadCast | 192.168.255.1 | C837_V2_UDP |  |
| UDP BroadCast | 10.1.102.222 | uPAC-5001 | * |

**Step 4: Configure the "IP" settings and then click the "Set" button**



IP Setting

Recommend Settings

IP: 10.1.0.29

Mask: 255.255.0.0

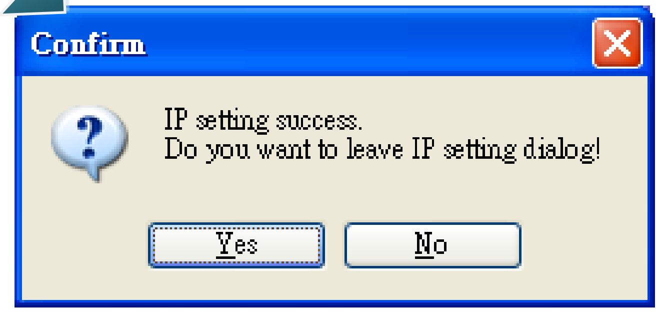
Gateway: 10.1.0.254

Alias:

DHCP

● Disable ○ Enable

**Step 5: Click "Yes" button**



Confirm

IP setting success.
Do you want to leave IP setting dialog!

## 2.5. UploadingI-8000 Programs

MiniOS7 Utility is a suite of tool for managing MiniOS7 devices (I-8000, µPAC-5000, iPAC-8000, µPAC-7186,. etc.). Before uploading programs, make sure that MiniOS7 Utility is installed on your PC.

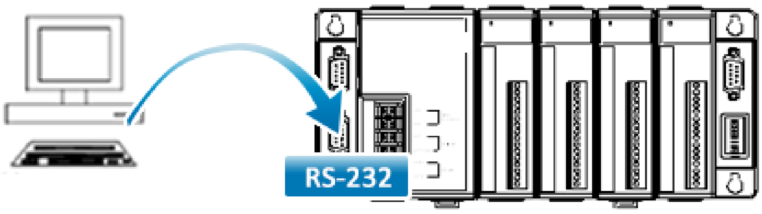
For more information about how to install the MiniOS7 utility, see section 2.2.2., “Installing the MiniOS7 Utility”

### 2.5.1. Establishing a Connection

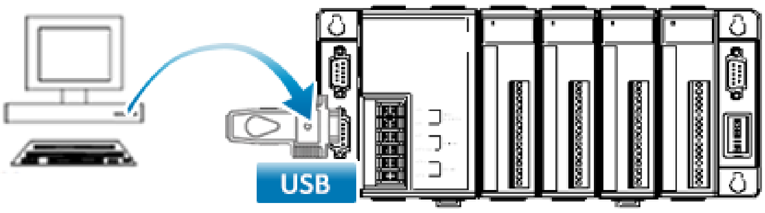
Before you use MiniOS7 Utility to upload programs, ensure that iPAC-8000 is connected to PC. For more information on how to connect iPAC-8000 to PC, see section 2.1.2., “Wiring the I-8000.”

The connection can be divided into the following three types according to the type of wire:

1. RS-232 (see section 2.5.1.1)

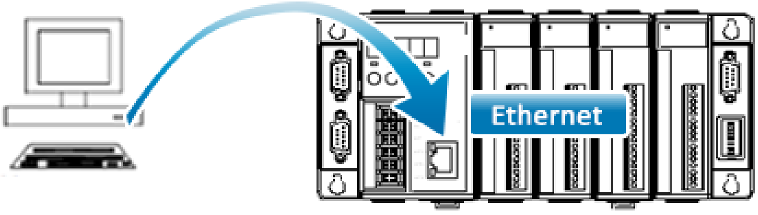


2. USB (see section 2.5.1.2)



3. Ethernet (see section 2.5.1.3)

(forEthernet series controller onl

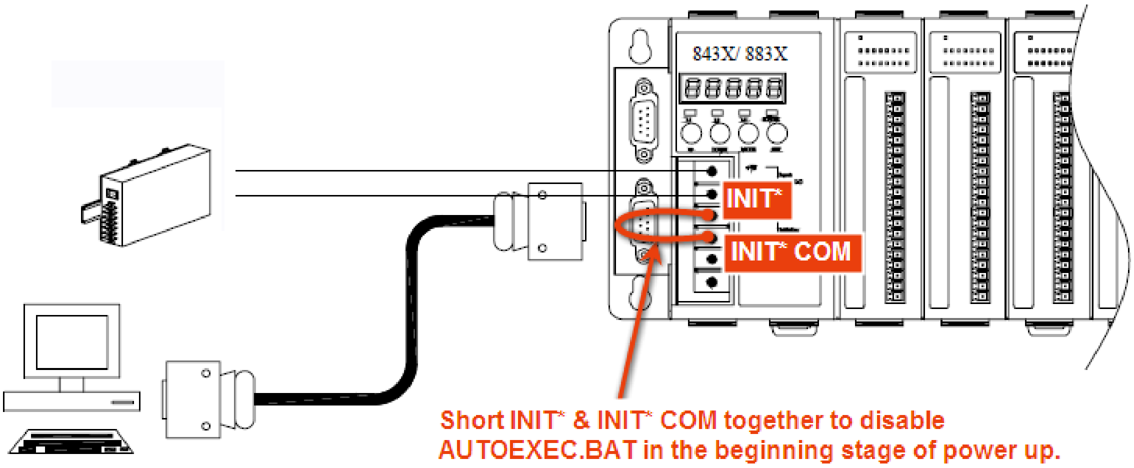


Each of the types of connection will be described in detail later.

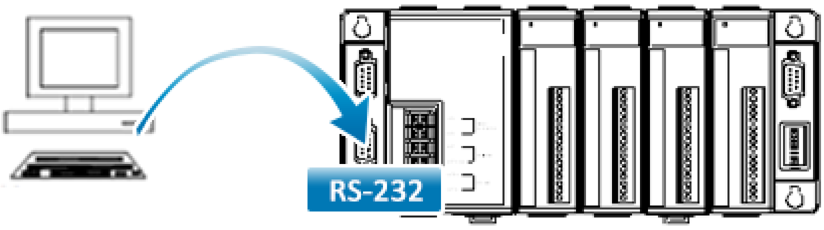
### 2.5.1.1. Using RS-232 to Establish a Connection

Below are step-by-step instructions on how to connect to PC using a RS-232 connection.

**Step 1: Boot the I-8000 into Initial mode**

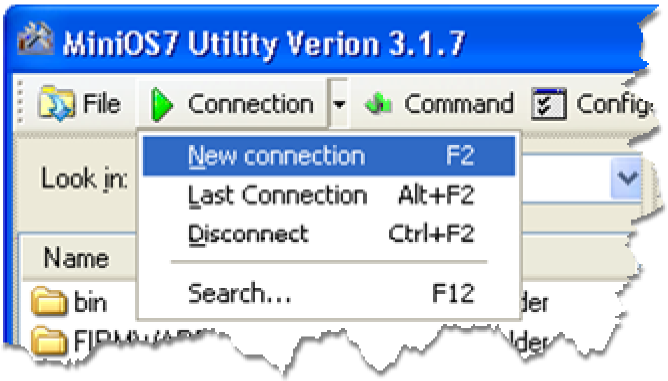

843X/ 883X
INIT*
INIT* COM
Short INIT* & INIT* COM together to disable
AUTOEXEC.BAT in the beginning stage of power up.

**Step 2: Use the RS-232 Cable (CA-0915) to connect to PC**

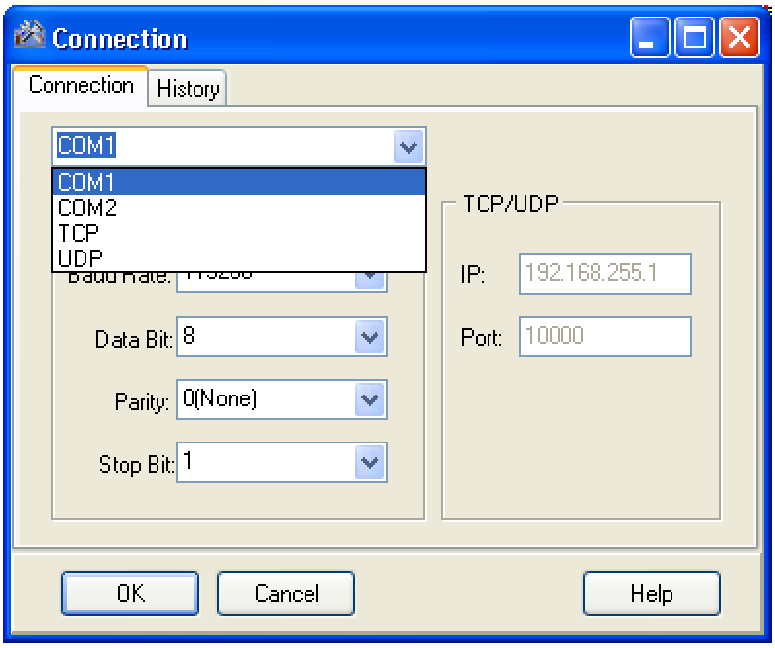


**Step 3: Run the MiniOS7 Utility**

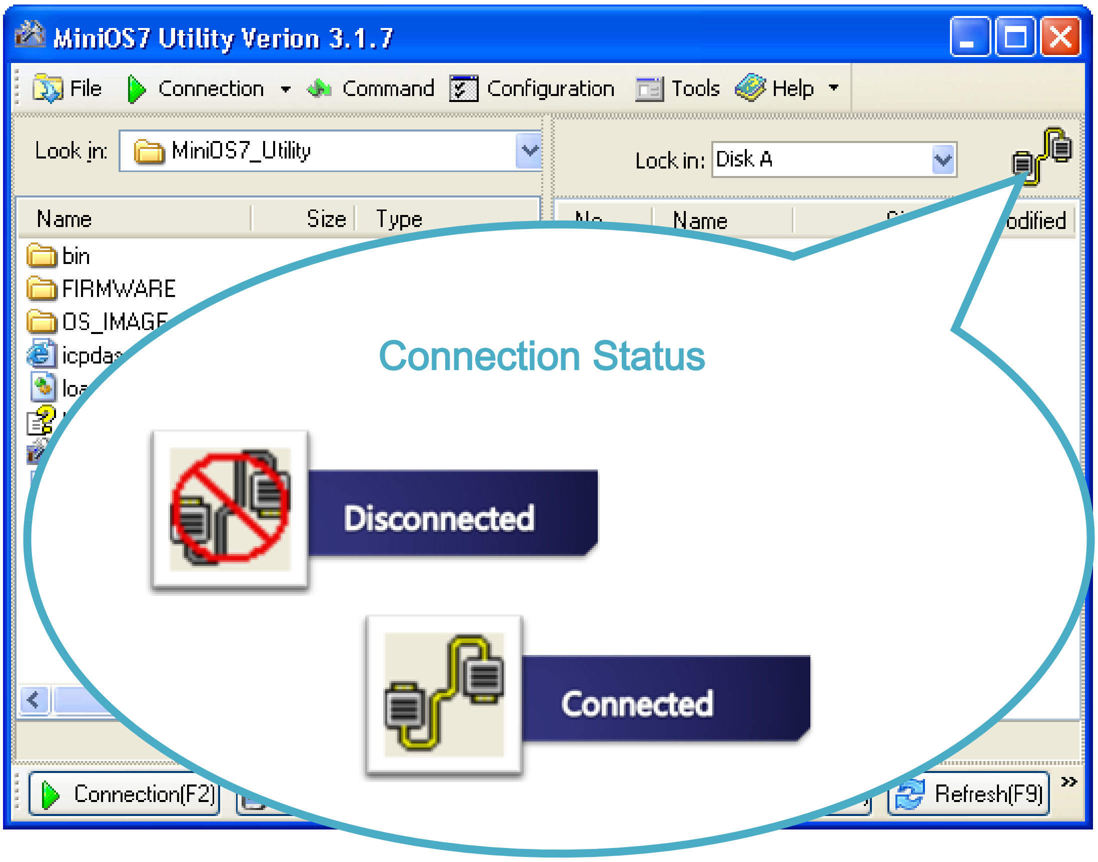
**Step 4: Click the "New connection" function from the "Connection" menu**



**Step 5: On the "Connection" tab of the "Connection" dialog box, select "COM1" from the drop down list, and then click "OK"**



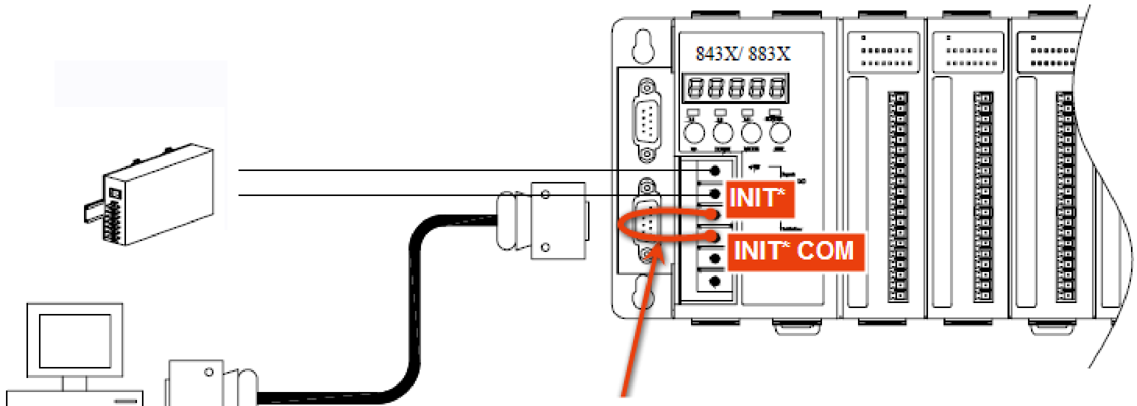
**Step 6: The connection has already established**



## 2.5.1.2. Using USB to Establish a Connection

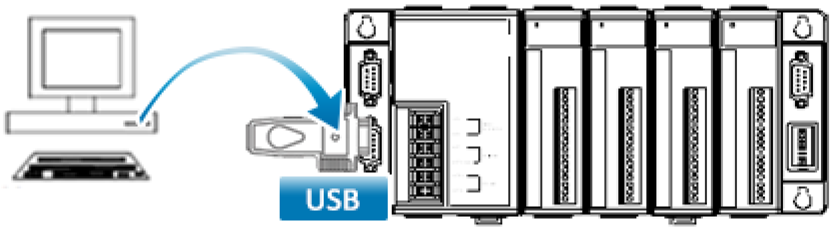
Below are step-by-step instructions on how to connect to PC using an USBconnection.

### Step 1: Boot the I-8000 into Initial mode



Short INIT* & INIT* COM together to disable AUTOEXEC.BAT in the beginning stage of power up.

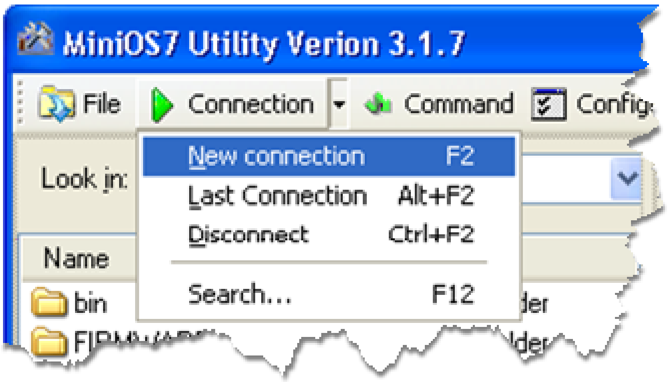
### Step 2: Use the USB cable to connect to PC



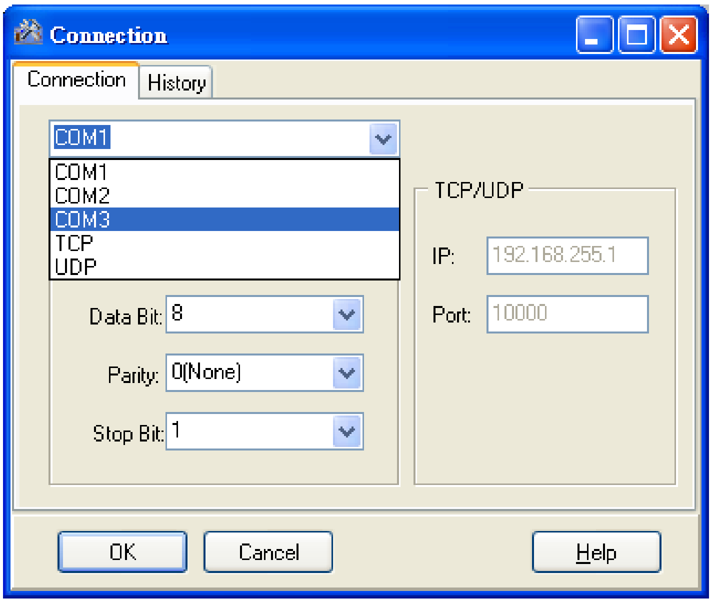
Before using the USB connection, ensure the I-7560 driver that you have installed. If they are not installed, please refer to “section 2.1. Hardware Installation”.

**Step 3: Run the MiniOS7 Utility**

**Step 4: Click the "New connection" function from the "Connection" menu**

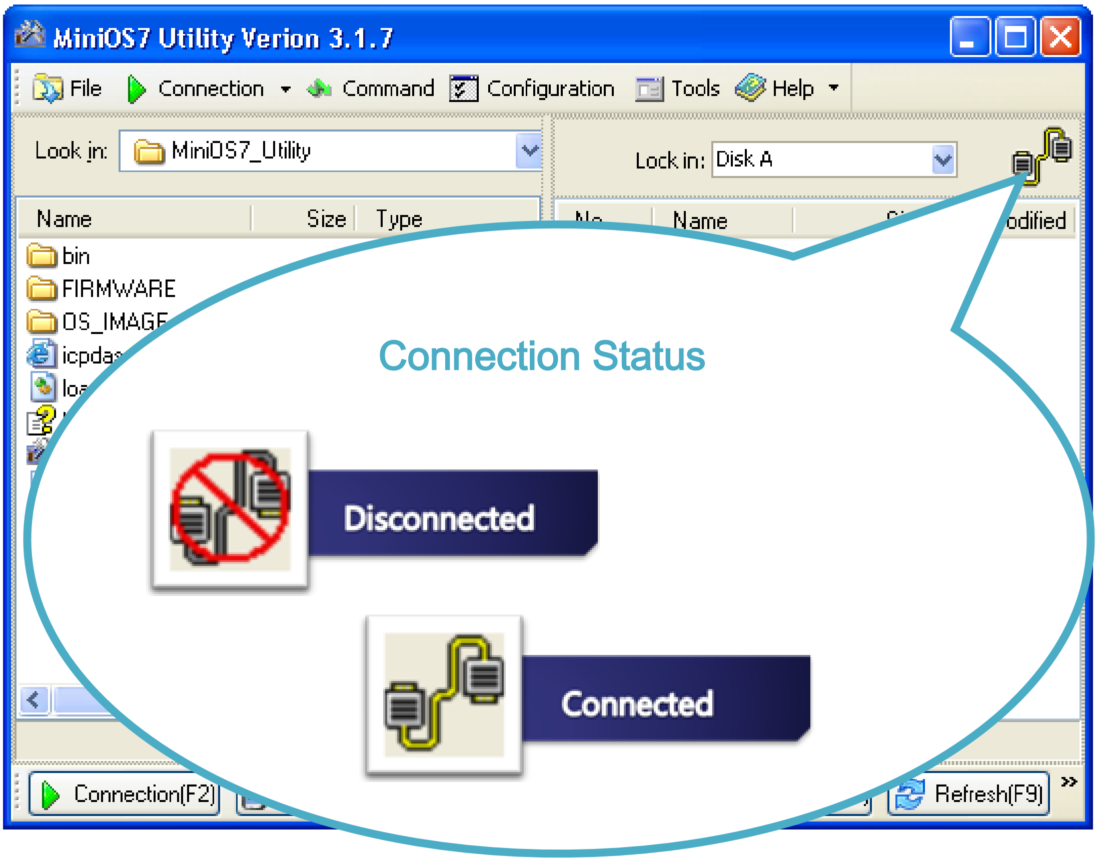


**Step 5: On the "Connection" tab of the "Connection" dialog box, select "COM3" from the drop down list, and then click "OK"**



Data Bit: 8
Parity: 0(None)
Stop Bit: 1
IP: 192.168.255.1
Port: 10000

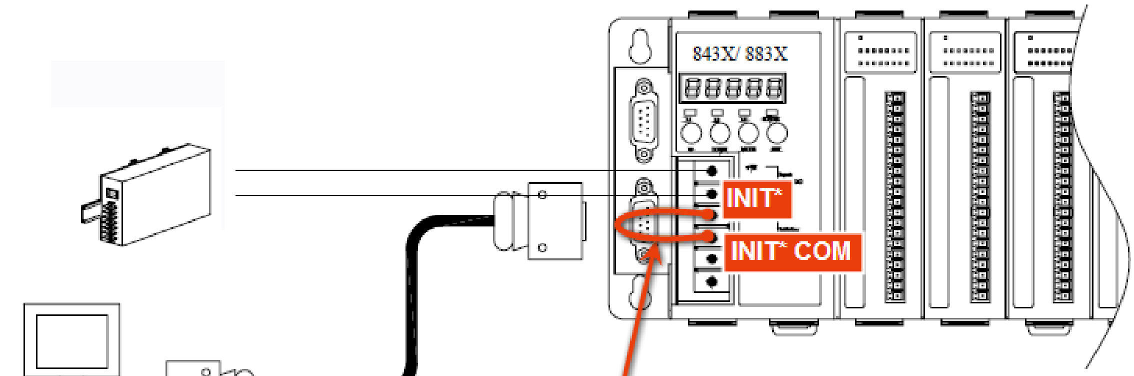
**Step 6: The connection has already established**



### 2.5.1.3. Using Ethernet to Establish a Connection (for Ethernet Series Controller only)

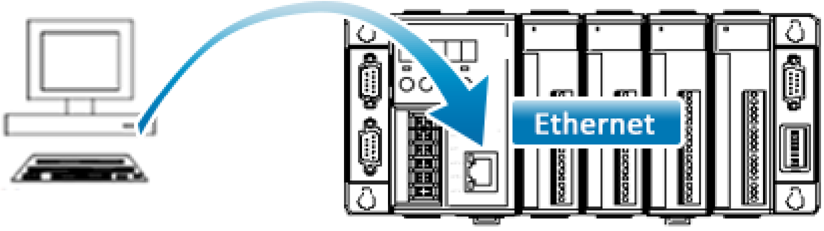
Below are step-by-step instructions on how to connect to PC using an Ethernet connection.

**Step 1: Boot the I-8000 into Initial mode**



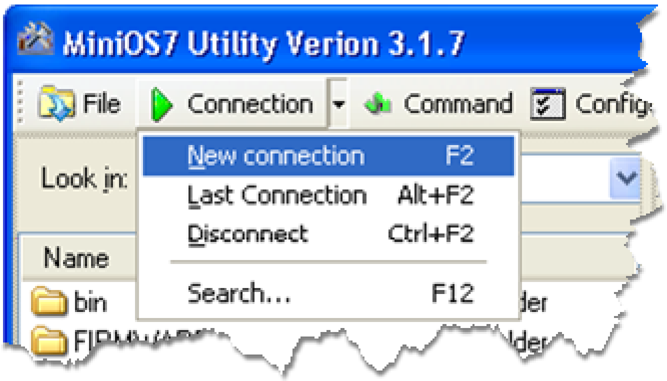
Short INIT* & INIT* COM together to disable AUTOEXEC.BAT in the beginning stage of power up.

**Step 2: Use the Ethernet cable to connect to PC**

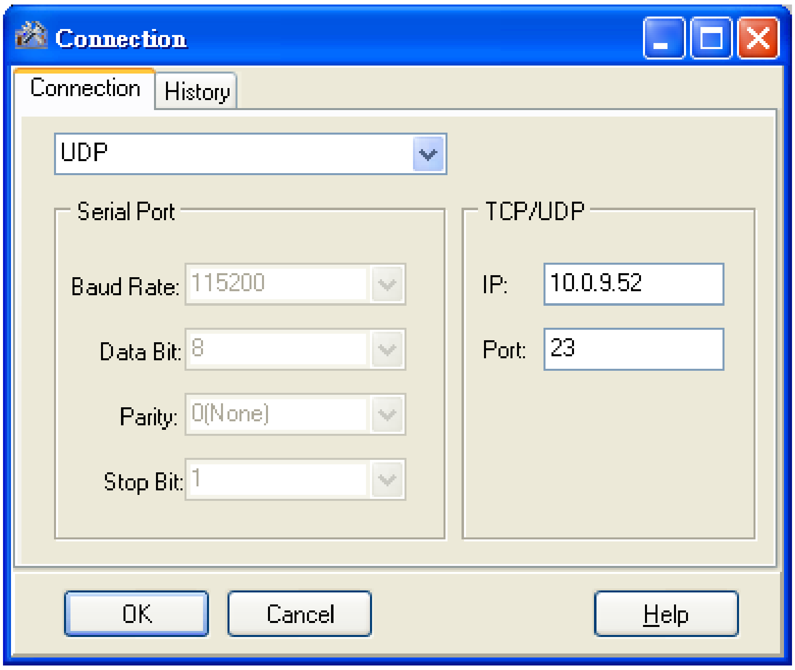


**Step 3: Run the MiniOS7 Utility**

**Step 4: Click the “New connection” function from the “Connection” menu**



**Step 5: On the “Connection” tab of the “Connection” dialog box, select “UDP” from the drop down list, type the IP address which you are assigned, and then click “OK”**



UDP

Baud Rate: 115200

Data Bit: 8

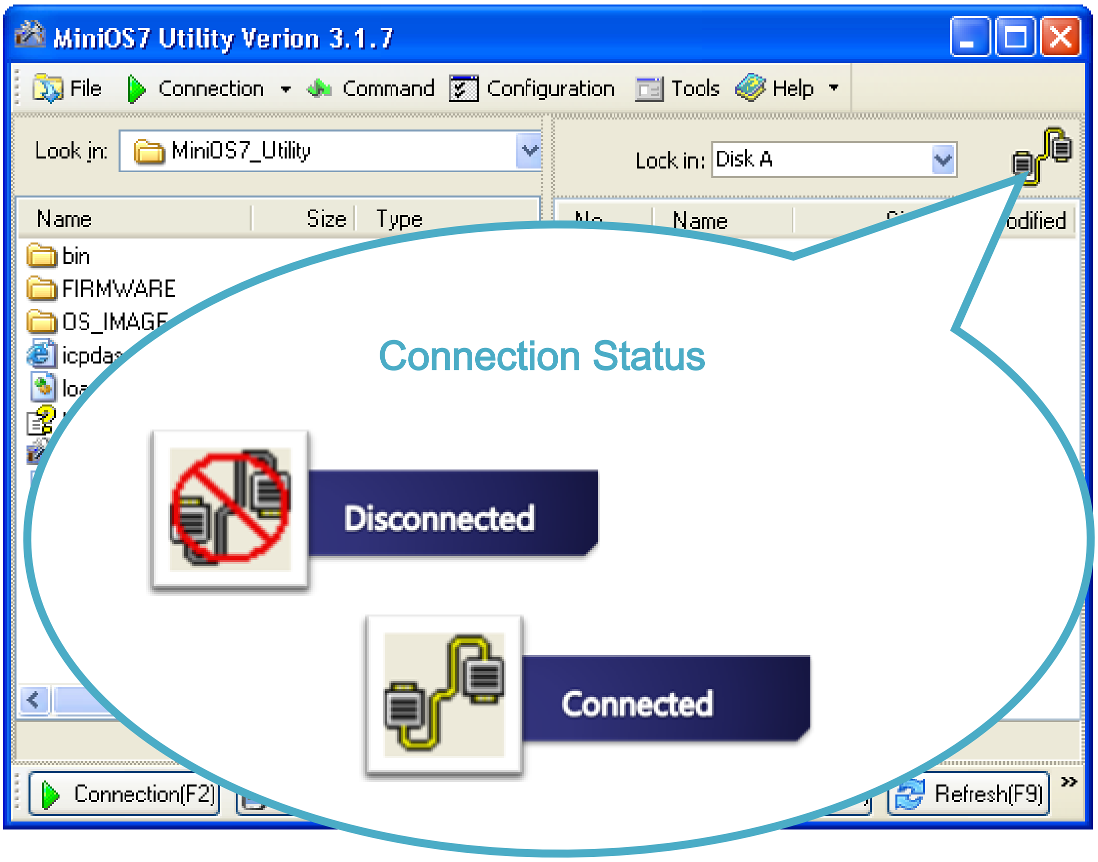
Parity: 0(None)

Stop Bit: 1

IP: 10.0.9.52

Port: 23

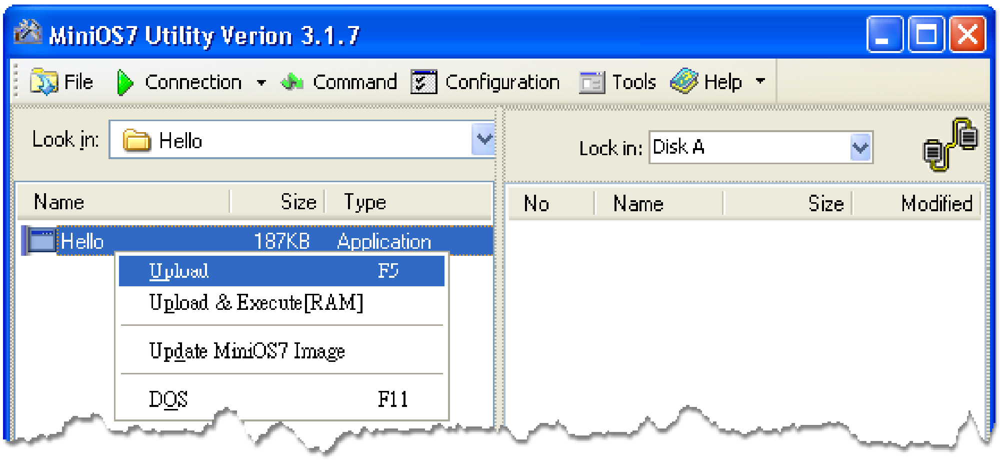
**Step 6: The connection has already established**



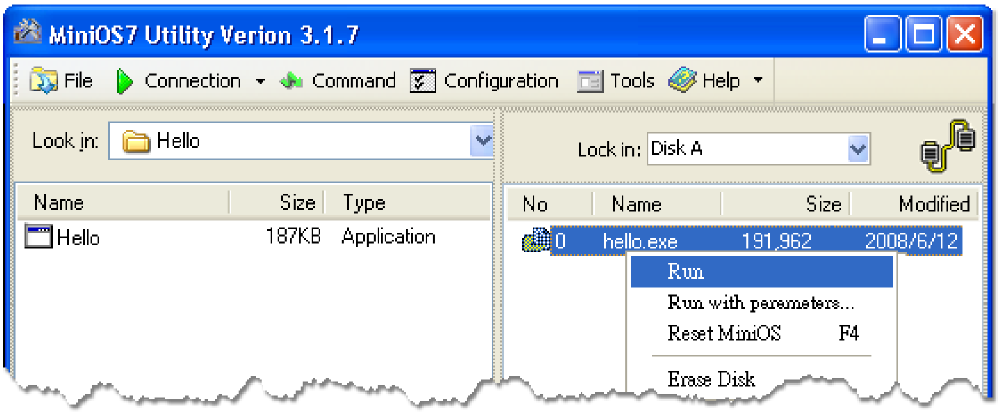
## 2.5.2. Uploading and Executing Programs

Before uploading and executing I-8000programs, you must firstly establisha connection between PC and I-8000, for more detailed information about this process, please refer to section “2.5.1. Establishing a connection”

**Step 1: On PC side, right click the file name that you wish to upload and then select the “Upload”**



**Step 2: On the module side, right click the file name that you wish to execute and then select the “Run”**



## 2.5.3. Making programs start automatically

After upload programs to the I-8000, if you need programs to start automatically after the I-8000 start-up, it is easy to achieve it, to create a batch file called autoexec.bat and then upload it to the I-8000, the program will start automatically in the next start-up.

For example, to make the program “hello” run on start-up.

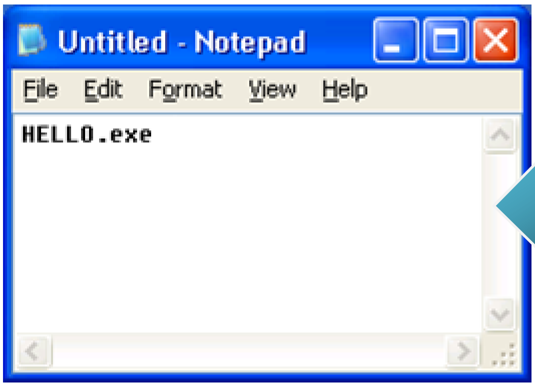
**Step 1: Createan autoexec.bat file**

i. Open the “Notepad”

ii. Type the command

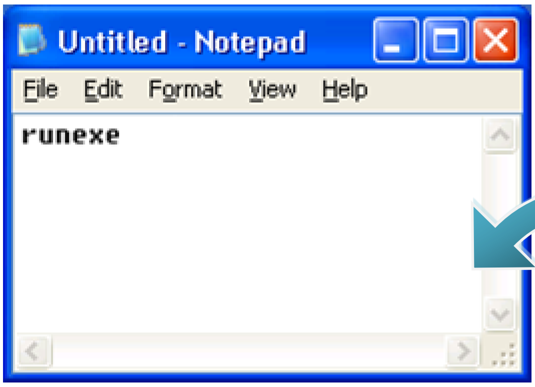
The command can be either the file name “HELLO.exe” (run the specified file) or “runexe” (run the last exe file)

iii. Save the file as autoexec.bat



```
HELLO.exe
```

The file name that need to be run on start-up

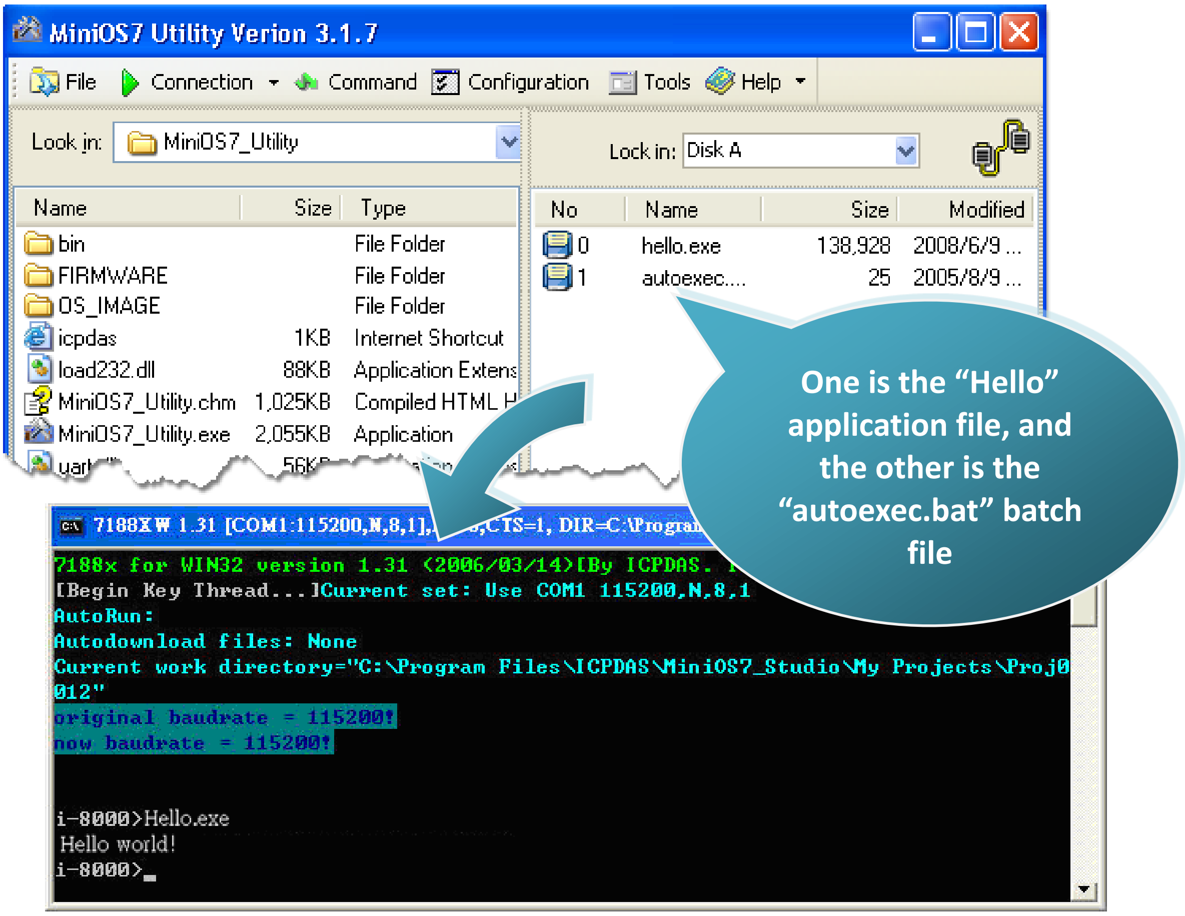


```
runexe
```

The function that can run last exe file

### Step 2: Upload programs to I-8000 using MiniOS7 Utility

For more detailed information about this process, please refer to section “2.5.2. Uploading and Executing programs”



| Name | Size | Type |
|---|---|---|
| bin | | File Folder |
| FIRMWARE | | File Folder |
| OS_IMAGE | | File Folder |
| icpdas | 1KB | Internet Shortcut |
| load232.dll | 88KB | Application Extens |
| MiniOS7_Utility.chm | 1,025KB | Compiled HTML H |
| MiniOS7_Utility.exe | 2,055KB | Application |
| uart | 56KB | |

| No | Name | Size | Modified |
|---|---|---|---|
| 0 | hello.exe | 138,928 | 2008/6/9 ... |
| 1 | autoexec.... | 25 | 2005/8/9 ... |

```
7188XW 1.31 [COM1:115200,N,8,1],  ,CTS=1, DIR=C:\Progra
7188x for WIN32 version 1.31 (2006/03/14)[By ICPDAS.
[Begin Key Thread...]Current set: Use COM1 115200,N,8,1
AutoRun:
Autodownload files: None
Current work directory="C:\Program Files\ICPDAS\MiniOS7_Studio\My Projects\Proj0
012"
original baudrate = 115200!
now baudrate = 115200!


i-8000>Hello.exe
 Hello world!
i-8000>_
```

One is the “Hello” application file, and the other is the “autoexec.bat” batch file

## Tips & Warnings

Before restaring the module for settings to take effect, you must firstlydisconnect the wire of Init and init*com.

# 3. “Hello World” - Your First Program

When you learn every computer programming language you may realize that the first program to demonstrate is "Hello World", it provides a cursory introduction to the language's syntax and output.

## 3.1. C Compiler Installation

C is prized for its efficiency, and is the most popular programming language for writing applications.

Before writing your first I-8000 program, ensure that you have the necessary C/C++ compiler and the corresponding functions library on your system.

The following is a list of the C compilers that are commonly used in the application development services.

- Turbo C++ Version 1.01
- Turbo C Version 2.01
- Borland C++ Versions 3.1 - 5.2.x
- MSC
- MSVC ++

We recommend that you use Borland C++ compiler.

**Tips & Warnings**

Before compiling an application, you need to take care of the following matters.

Generate a standard DOS executable program

Set the CPU option to 80188/80186

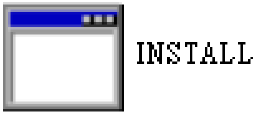
Set the floating point option to EMULATION if floating point computation is required. (Be sure not to choose 8087)

Cancel the Debug Information function as this helps to reduce program size. (MiniOS7 supports this feature.).

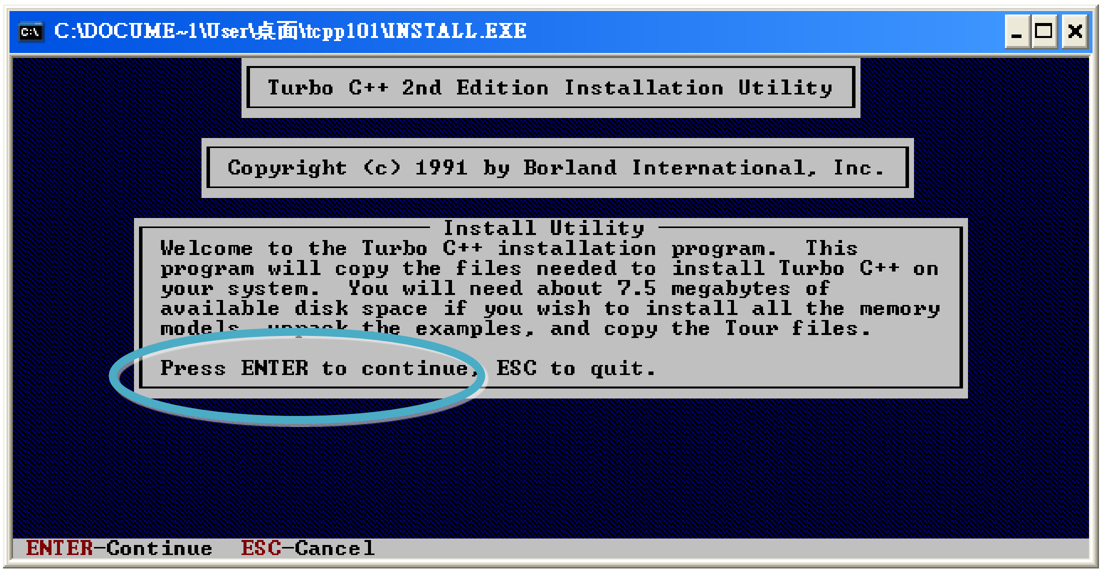
## 3.1.1. Installing the C compiler

If there is no compiler currently installed on your system, installation of the compiler should be the first step.

### Step 1: Double click the Turbo C++ executable file to start setup wizard

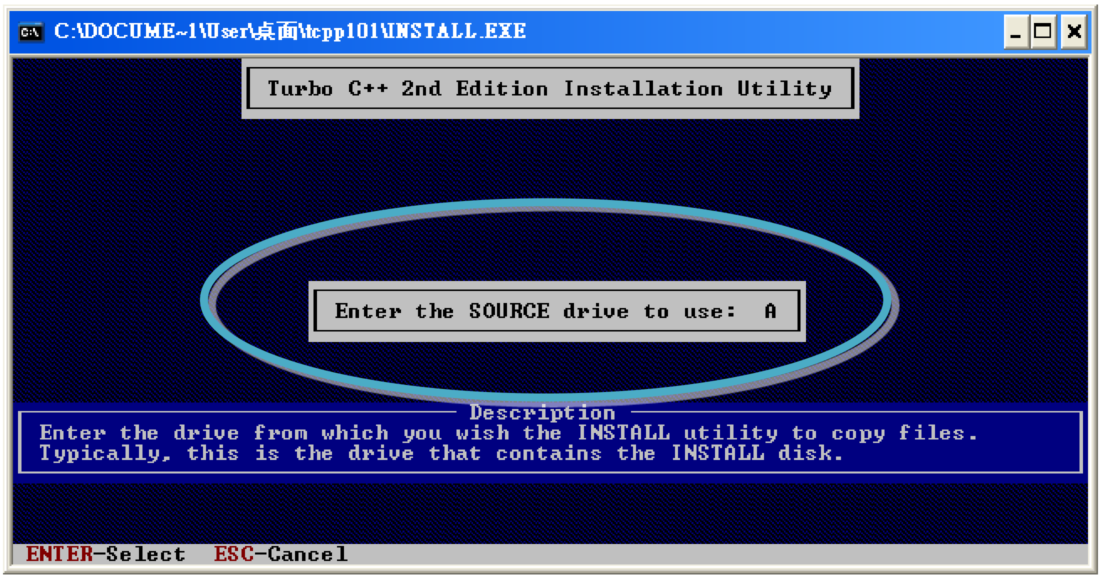


### Step 2: Press “Enter” to continue



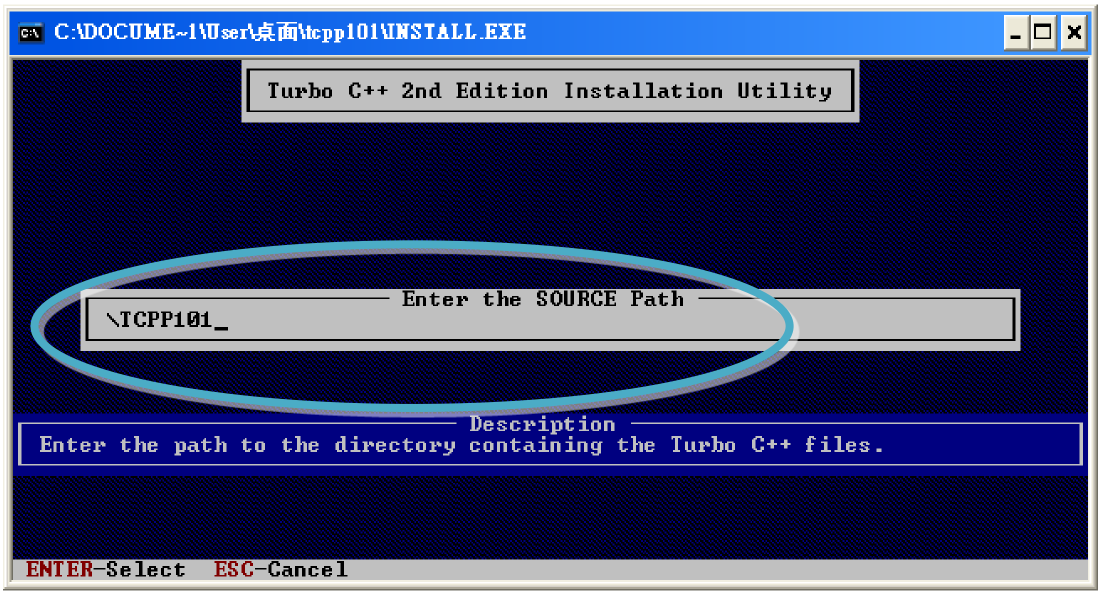
```
Turbo C++ 2nd Edition Installation Utility

Copyright (c) 1991 by Borland International, Inc.

                 Install Utility
Welcome to the Turbo C++ installation program.  This
program will copy the files needed to install Turbo C++ on
your system.  You will need about 7.5 megabytes of
available disk space if you wish to install all the memory
models, unpack the examples, and copy the Tour files.

Press ENTER to continue, ESC to quit.

ENTER-Continue  ESC-Cancel
```

**Step 3: Enter the letter of the hard drive you wish to install the software**



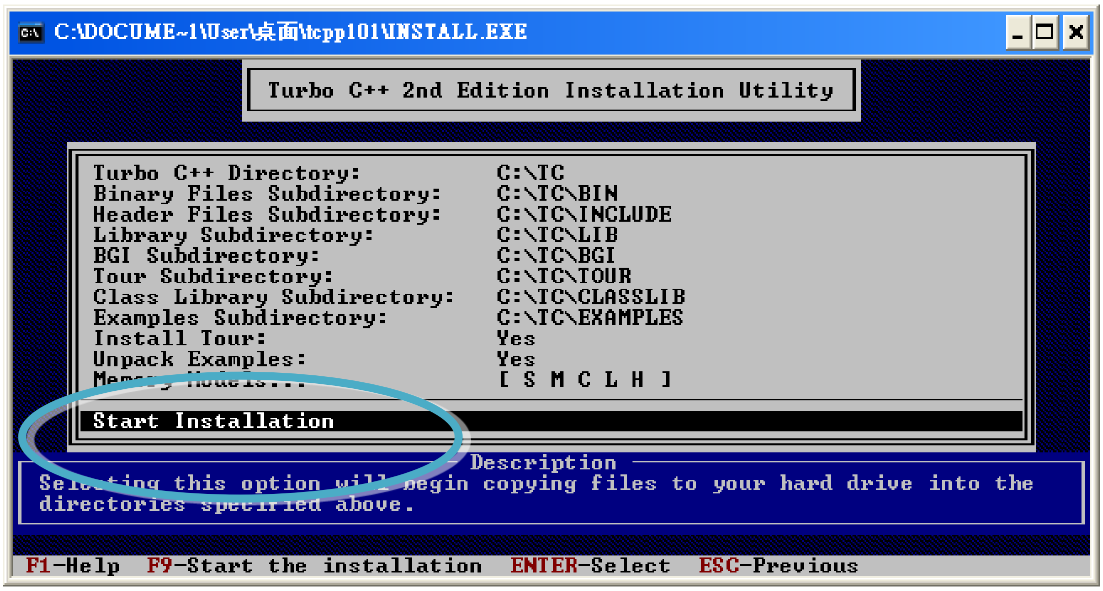
```
Turbo C++ 2nd Edition Installation Utility

Enter the SOURCE drive to use:  A

Description
Enter the drive from which you wish the INSTALL utility to copy files.
Typically, this is the drive that contains the INSTALL disk.

ENTER-Select  ESC-Cancel
```

**Step 4: Enter the path to the directory you wish to install files to**



```
Turbo C++ 2nd Edition Installation Utility

Enter the SOURCE Path
\TCPP101_

Description
Enter the path to the directory containing the Turbo C++ files.

ENTER-Select  ESC-Cancel
```

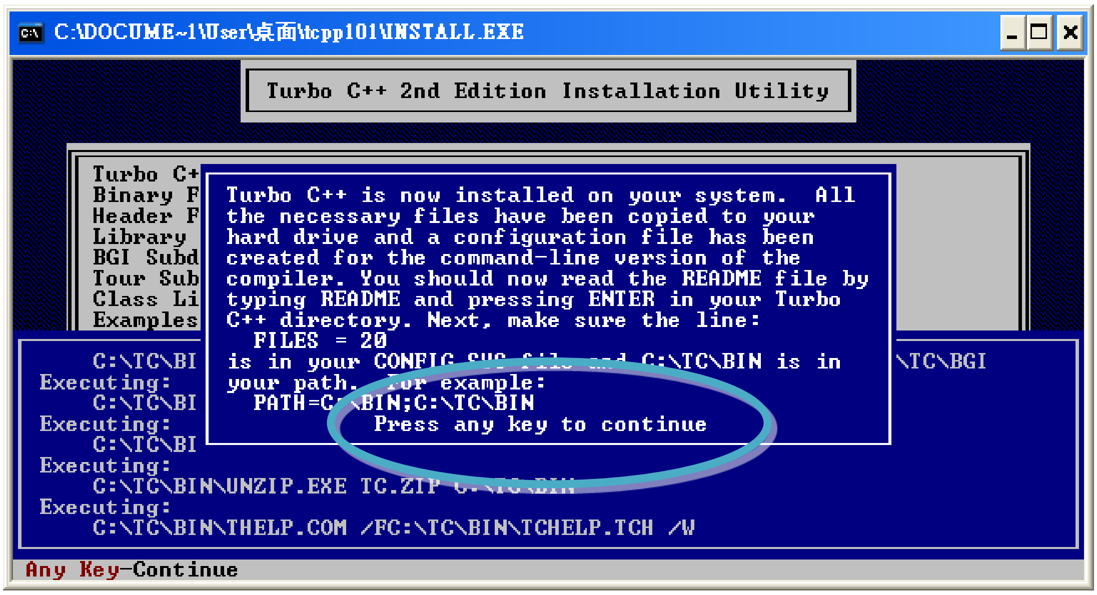
### Step 5: Select “Start Installation” to begin the install process



C:\DOCUME~1\User\桌面\tcpp101\INSTALL.EXE

```
Turbo C++ 2nd Edition Installation Utility

Turbo C++ Directory:             C:\TC
Binary Files Subdirectory:       C:\TC\BIN
Header Files Subdirectory:       C:\TC\INCLUDE
Library Subdirectory:            C:\TC\LIB
BGI Subdirectory:                C:\TC\BGI
Tour Subdirectory:               C:\TC\TOUR
Class Library Subdirectory:      C:\TC\CLASSLIB
Examples Subdirectory:           C:\TC\EXAMPLES
Install Tour:                    Yes
Unpack Examples:                 Yes
Memory Models...                 [ S M C L H ]
Start Installation

Description
Selecting this option will begin copying files to your hard drive into the
directories specified above.

F1-Help  F9-Start the installation  ENTER-Select  ESC-Previous
```

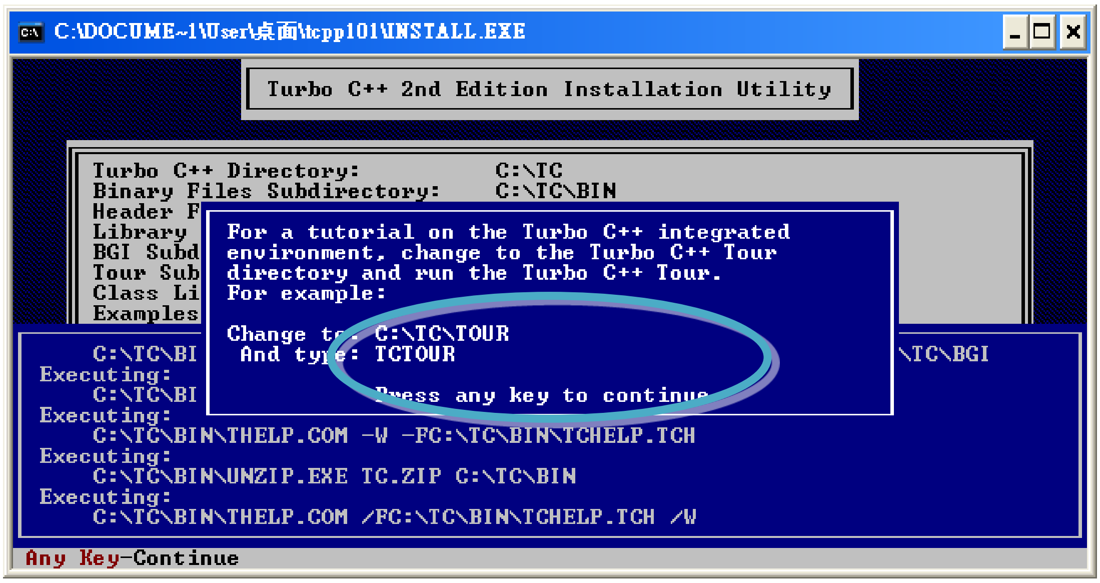
### Step 6: Press any key to continue



C:\DOCUME~1\User\桌面\tcpp101\INSTALL.EXE

```
Turbo C++ 2nd Edition Installation Utility

Turbo C++ is now installed on your system.  All
the necessary files have been copied to your
hard drive and a configuration file has been
created for the command-line version of the
compiler. You should now read the README file by
typing README and pressing ENTER in your Turbo
C++ directory. Next, make sure the line:
  FILES = 20
is in your CONFIG.SYS file and C:\TC\BIN is in
your path.  For example:
  PATH=C:\TC\BIN;C:\TC\BIN
          Press any key to continue

Executing:
    C:\TC\BIN\UNZIP.EXE TC.ZIP C:\TC\BIN
Executing:
    C:\TC\BIN\THELP.COM /FC:\TC\BIN\TCHELP.TCH /W

Any Key-Continue
```

## Step 7: Press any key to continue



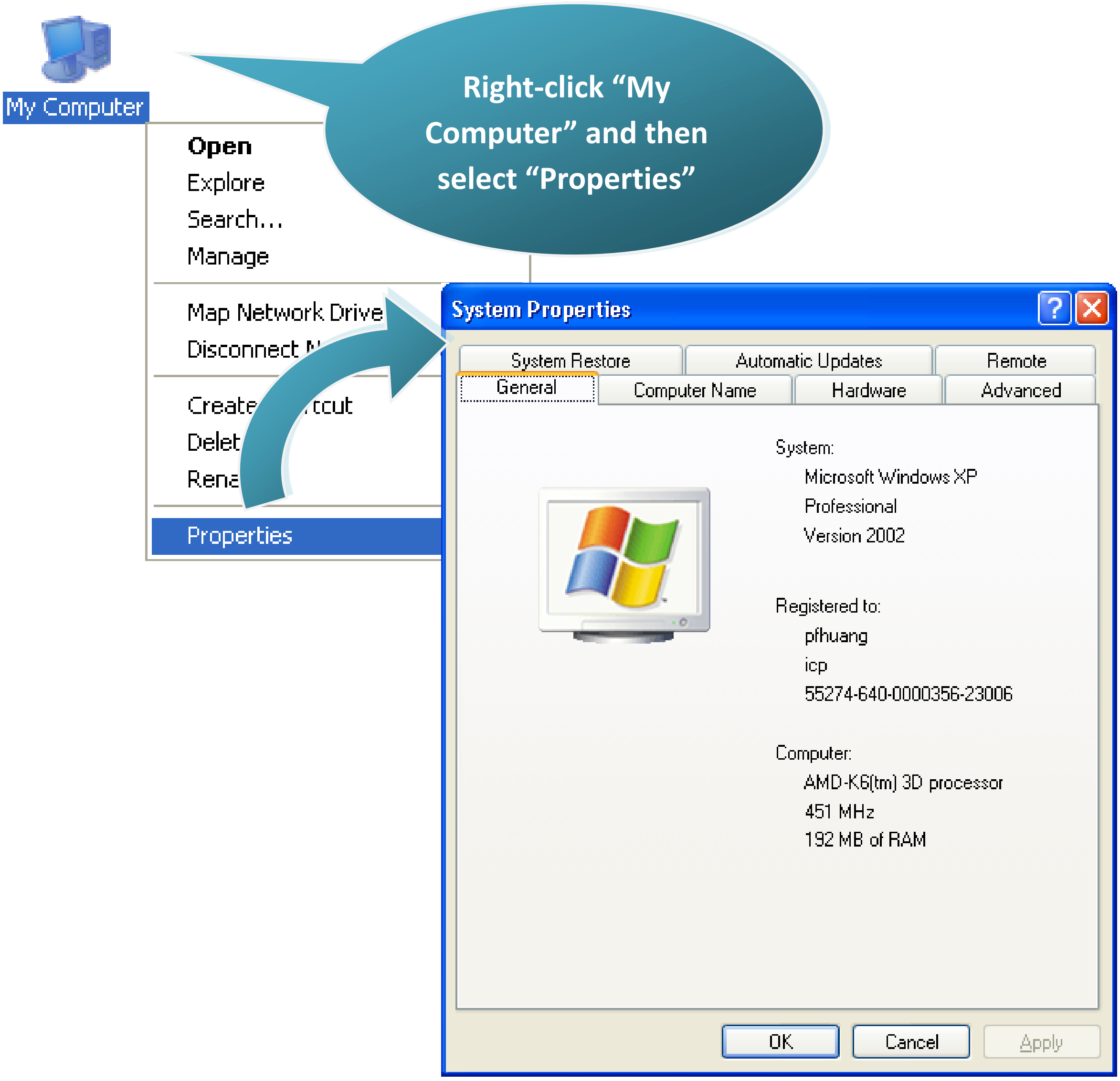
```
              Turbo C++ 2nd Edition Installation Utility

    Turbo C++ Directory:          C:\TC
    Binary Files Subdirectory:    C:\TC\BIN
    Header F
    Library
    BGI Subd
    Tour Sub
    Class Li
    Examples

       For a tutorial on the Turbo C++ integrated
       environment, change to the Turbo C++ Tour
       directory and run the Turbo C++ Tour.
       For example:

       Change to: C:\TC\TOUR
        And type: TCTOUR

                   Press any key to continue

    C:\TC\BI                                                 \TC\BGI
Executing:
    C:\TC\BI
Executing:
    C:\TC\BIN\THELP.COM -W -FC:\TC\BIN\TCHELP.TCH
Executing:
    C:\TC\BIN\UNZIP.EXE TC.ZIP C:\TC\BIN
Executing:
    C:\TC\BIN\THELP.COM /FC:\TC\BIN\TCHELP.TCH /W
Any Key-Continue
```

## Step 8: Installation is complete

### 3.1.2. Setting up the environment variables

After installing the compiler, several compilers will be available from the Windows Command line. You can set the path environment variable so that you can execute this compiler on the command line by entering simple names, rather than by using their full path names.

**Step 1: Right click on the "My Computer" icon on your desktop and select the"Properties" menu option**



Right-click "My Computer" and then select "Properties"

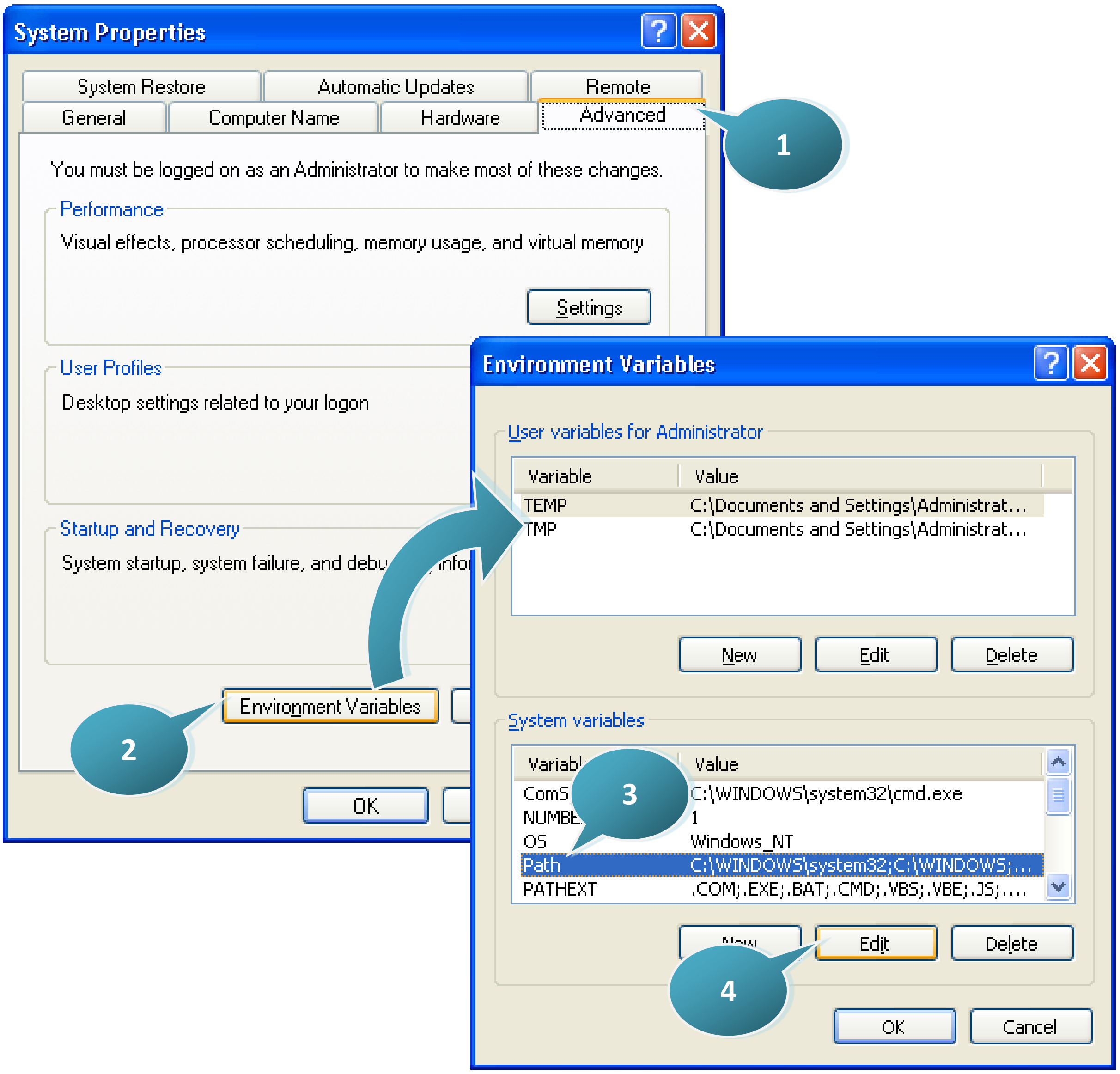
System:
Microsoft Windows XP
Professional
Version 2002

Registered to:
pfhuang
icp
55274-640-0000356-23006

Computer:
AMD-K6(tm) 3D processor
451 MHz
192 MB of RAM

**Step 2: On the "System Properties" dialog box, click the "Environment Variables"button located under the "Advanced" sheet**

**Step 3: On the "Environment Variables" dialog box, click the "Edit" button located in the "System variables" option**



You must be logged on as an Administrator to make most of these changes.

Performance: Visual effects, processor scheduling, memory usage, and virtual memory

User Profiles: Desktop settings related to your logon

Startup and Recovery: System startup, system failure, and debu... infor...

User variables for Administrator

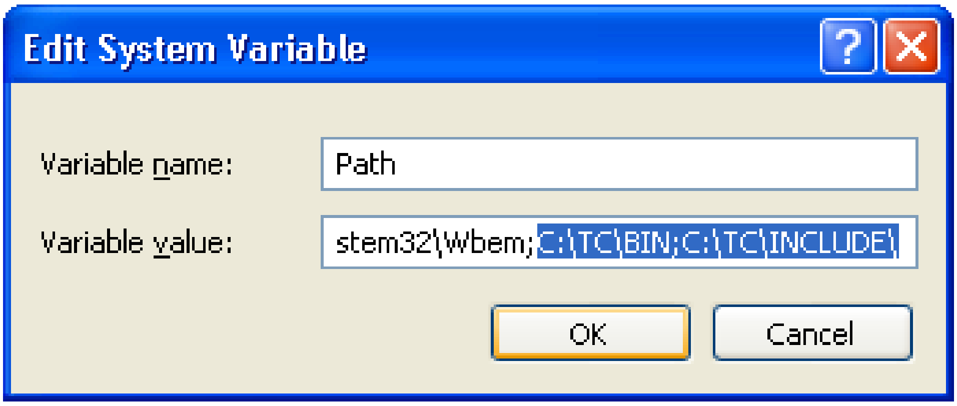
| Variable | Value |
|---|---|
| TEMP | C:\Documents and Settings\Administrat... |
| TMP | C:\Documents and Settings\Administrat... |

System variables

| Variabl | Value |
|---|---|
| ComS | C:\WINDOWS\system32\cmd.exe |
| NUMBE | 1 |
| OS | Windows_NT |
| Path | C:\WINDOWS\system32;C:\WINDOWS;... |
| PATHEXT | .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.... |

### Step 4: Add the target directory to the end of the variable value field

A semi-colon is used as the separator between variable values.
For example, ”;c:\TC\BIN\;c:\TC\INCLUDE\”



Variable name: Path

Variable value: stem32\Wbem;C:\TC\BIN;C:\TC\INCLUDE\

### Step 5: Restart the computer to allow your changes to take effect

## 3.2. I-8000APIs

There are several APIs for customizing the standard features and integrating with other applications, devices and services.

For more detailed information regarding I-8000 APIs, please refer to

https://www.icpdas.com/en/download/show.php?num=2778&model=I-8831

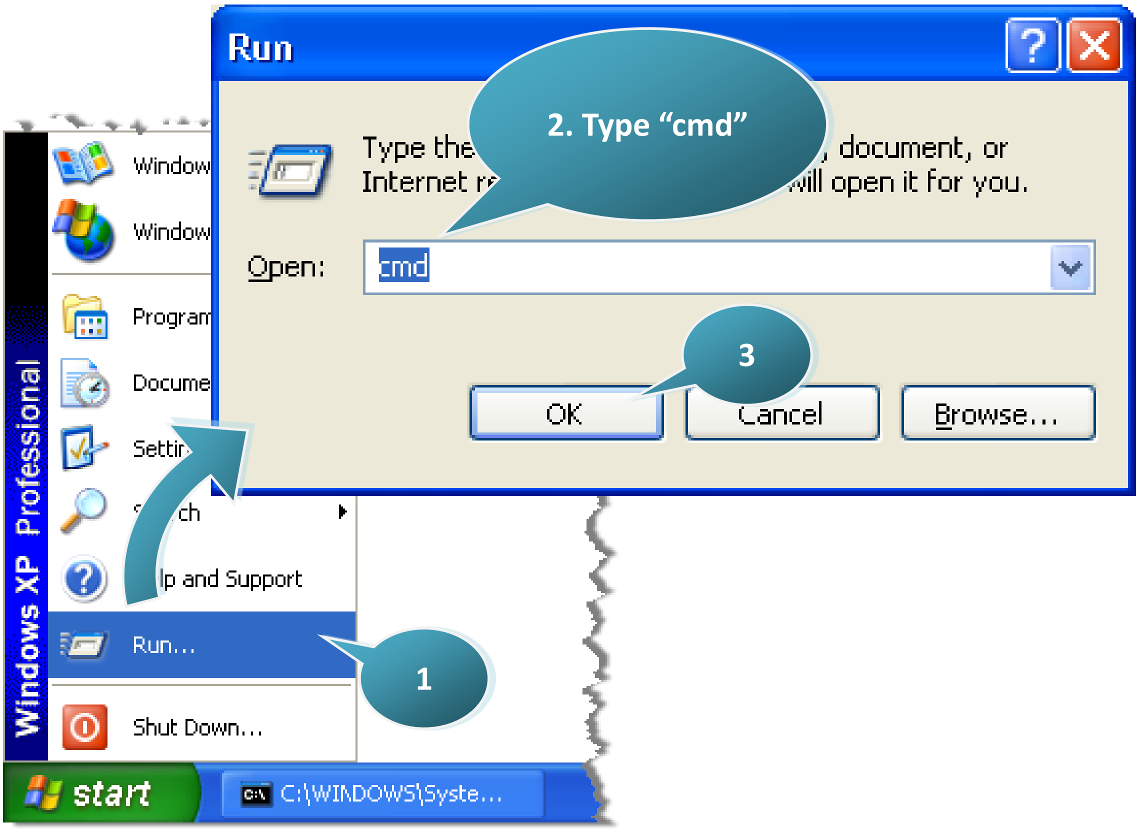
Before creating the application, ensure them that you have installed. If they are not installed, please refer to “section 2.2. Installing Software”.

# 3.3. First Program in I-8000

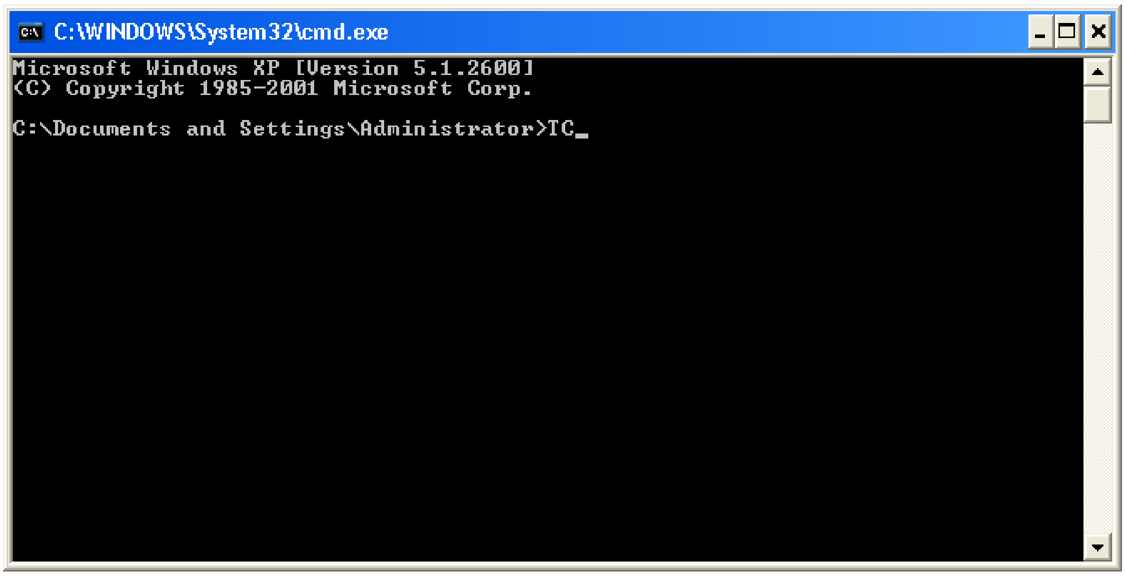
Here we assume you have installed the Turbo C++ 1.01 (as the section “3.1. C Compiler Installation”) and the I-8000APIs (as the section “2.2.Installing Software”) under the C driver root folder.

### Step 1: Open a MS-DOS command prompt

i. Select “Run” from the “Start” menu
ii. On the “Run” dialog box, type “cmd”
iii. Click the “OK” button

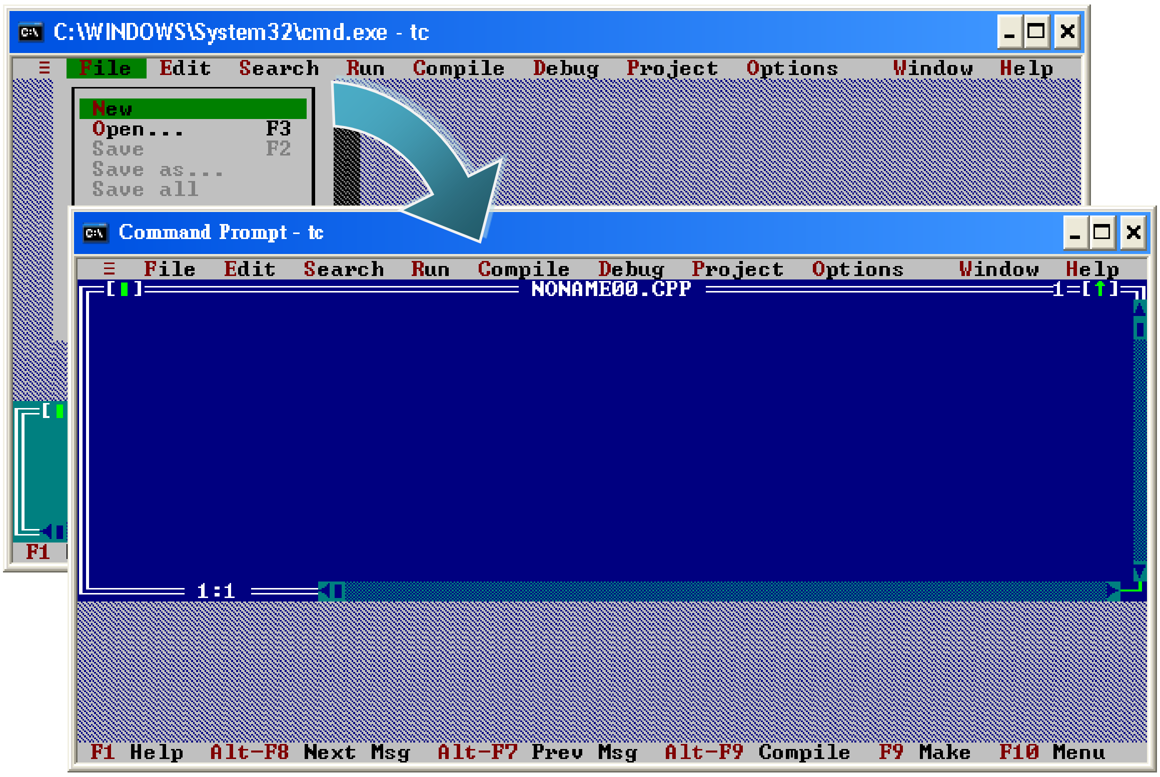


**Step 2: At the command prompt, type "TC" and then press "Enter"**



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>TC_
```

**Step 3: Select "New" from the "File" menu to create a new source file**



C:\WINDOWS\System32\cmd.exe - tc
≡ File Edit Search Run Compile Debug Project Options Window Help
New
Open... F3
Save F2
Save as...
Save all
F1
Command Prompt - tc
≡ File Edit Search Run Compile Debug Project Options Window Help
NONAME00.CPP
1
1:1
F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt-F9 Compile F9 Make F10 Menu

**Step 4: Type the following code. Note that the code is case-sensitive**

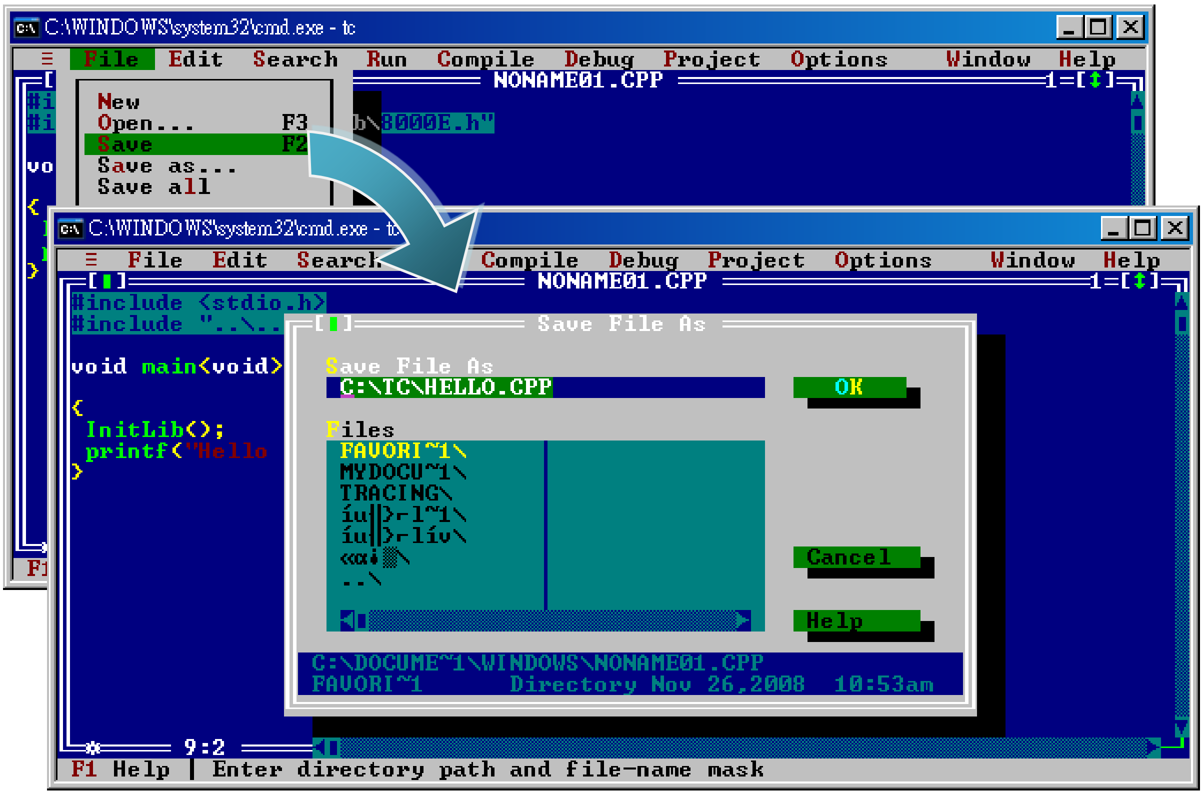
```
#include <stdio.h>
#include “..\..\demo\Lib\8000E.h”
/* Include the header file that allows 8000E.lib functions to be used */

void main(void)
{
InitLib();      /* Initiate the 8000E library */
  Print(“Hello world!\r\n”);      /* Print the message on the screen */
}
```

### Step 5: Save the source file

i. Select “Save” from the “File” menu

ii. Type the file name “Hello”

iii. Select “OK”


C:\WINDOWS\system32\cmd.exe - tc
≡ File Edit Search Run Compile Debug Project Options Window Help
NONAME01.CPP
New
Open... F3
Save F2
Save as...
Save all
Save File As
OK
Cancel
Help

```
#include <stdio.h>
#include "..\..

void main(void)

{
  InitLib();
  printf("Hello
}
```

Save File As
C:\TC\HELLO.CPP

Files
FAVORI~1\
MYDOCU~1\
TRACING\
..\

C:\DOCUME~1\WINDOWS\NONAME01.CPP
FAVORI~1 Directory Nov 26,2008 10:53am

F1 Help | Enter directory path and file-name mask

## Tips & Warnings

You can write the code as shown below with your familiar text editor or other tools; please note that

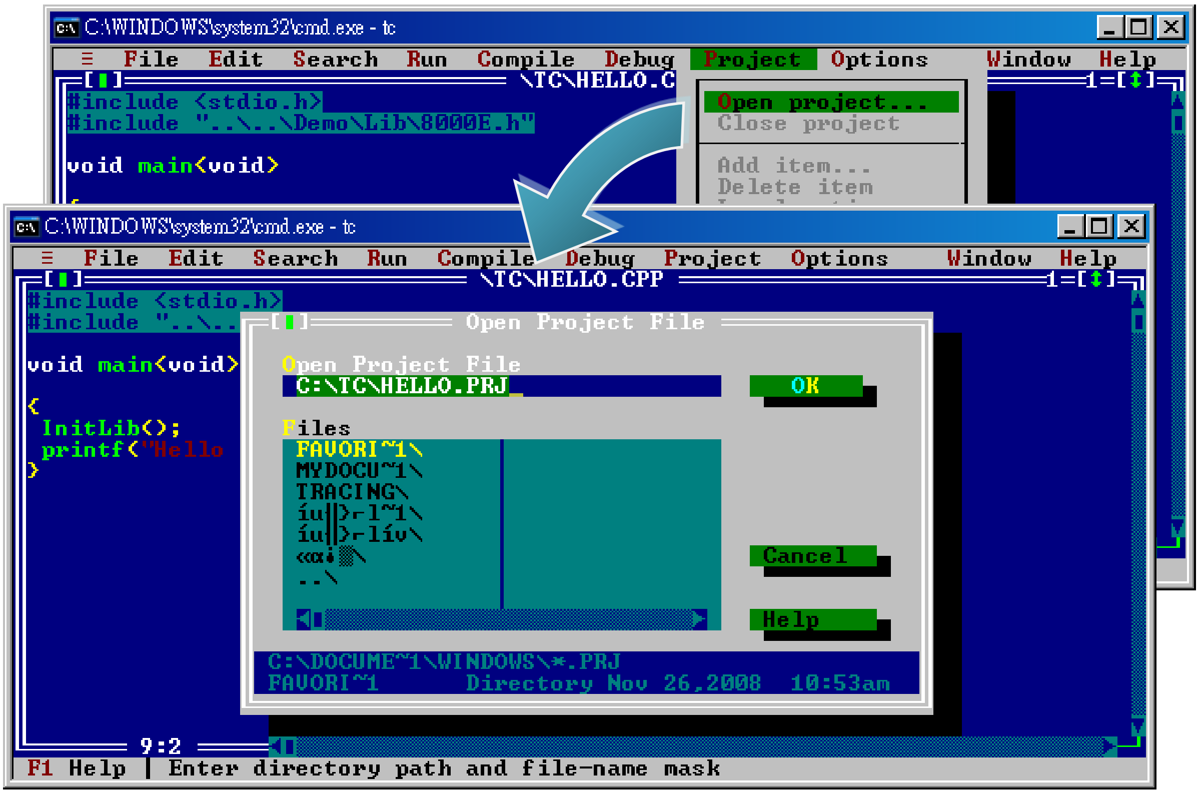
you must save the source code under a filename that terminates with the extension “C”.

### Step 6: Create a project (*.PRJ)

i. Select “Open project…” from the “Project” menu

ii. Type the project name “HELLO.PRJ”

iii. Select “OK”



```
#include <stdio.h>
#include "..\..\Demo\Lib\8000E.h"

void main(void)
{
 InitLib();
 printf("Hello
}
```

Open Project File: C:\TC\HELLO.PRJ

C:\DOCUME~1\WINDOWS\*.PRJ
FAVORI~1          Directory Nov 26,2008   10:53am

F1 Help | Enter directory path and file-name mask

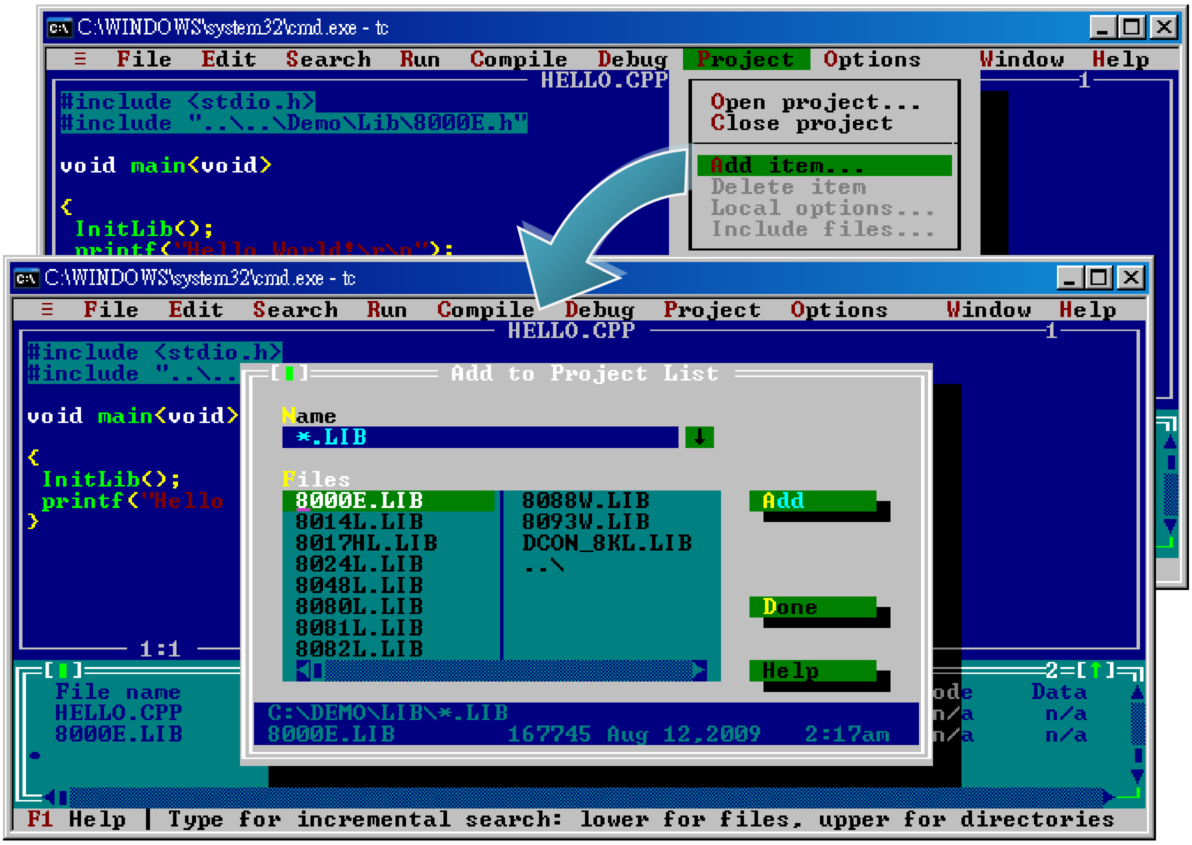
### Step 7: Add the necessary function libraries to the project (*.LIB)

i. Select "Add item..." from the "Project" menu

ii. Select the source file (HELLO.CPP) and then click the "Add" button

iii. Select the function library (8000E.LIB) and then click the “Add” button

iv. Select "Done" to exit



```
#include <stdio.h>
#include "..\..\Demo\Lib\8000E.h"

void main(void)

{
 InitLib();
 printf("Hello World!\r\n");
}
```

Name: *.LIB

Files:

```
8000E.LIB      8088W.LIB
8014L.LIB      8093W.LIB
8017HL.LIB     DCON_8KL.LIB
8024L.LIB      ..\
8048L.LIB
8080L.LIB
8081L.LIB
8082L.LIB
```

```
C:\DEMO\LIB\*.LIB
8000E.LIB          167745  Aug 12,2009     2:17am
```

```
File name                    Code   Data
HELLO.CPP                    n/a    n/a
8000E.LIB                    n/a    n/a
```

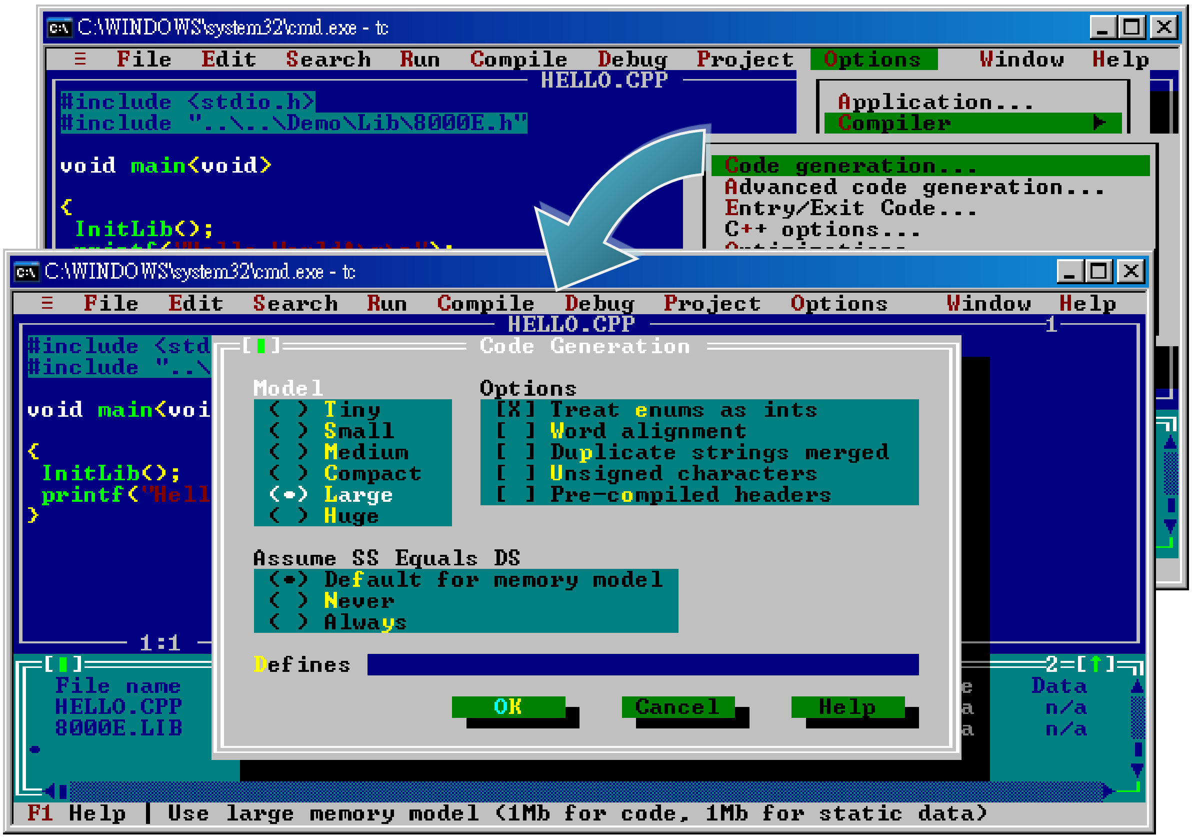
F1 Help | Type for incremental search: lower for files, upper for directories

## Step 8: Set the memory model to large

i. Select “Compiler” from the “Options” menu and then select “Code generation…”

ii. On “Model” option, select “Large”

iii. Select “OK”



```
#include <stdio.h>
#include "..\..\Demo\Lib\8000E.h"

void main(void)
{
 InitLib();
 printf("Hell
}
```

```
Model                     Options
 ( ) Tiny                  [X] Treat enums as ints
 ( ) Small                 [ ] Word alignment
 ( ) Medium                [ ] Duplicate strings merged
 ( ) Compact               [ ] Unsigned characters
 (•) Large                 [ ] Pre-compiled headers
 ( ) Huge

Assume SS Equals DS
 (•) Default for memory model
 ( ) Never
 ( ) Always

Defines
```

```
F1 Help | Use large memory model (1Mb for code, 1Mb for static data)
```

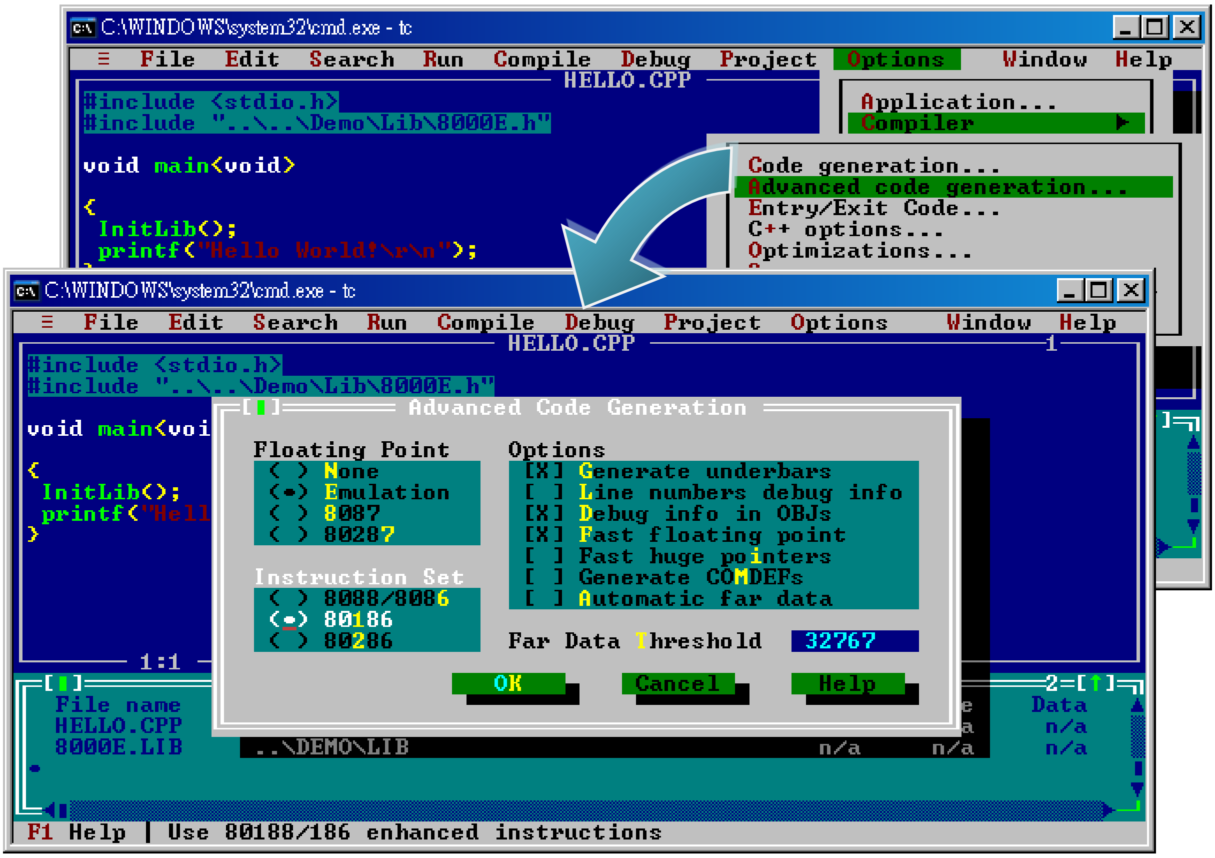
### Step 9: Set the Floating Point to Emulation and the Instruction Set to 80186

i. Select “Compiler” from the “Options” menu and then select “Advanced code generation…”

ii. On “Floating Point” option, select “Emulation”

iii. On “Instruction Set” option, select “80186”

iv. Select “OK”


C:\WINDOWS\system32\cmd.exe - tc
≡ File Edit Search Run Compile Debug Project Options Window Help
HELLO.CPP
Application...
Compiler ►
Code generation...
Advanced code generation...
Entry/Exit Code...
C++ options...
Optimizations...
C:\WINDOWS\system32\cmd.exe - tc
≡ File Edit Search Run Compile Debug Project Options Window Help
HELLO.CPP
1:1
File name
HELLO.CPP
8000E.LIB
..\DEMO\LIB
n/a
Data
n/a
n/a
OK
Cancel
Help

```
#include <stdio.h>
#include "..\..\Demo\Lib\8000E.h"

void main(void)

{
 InitLib();
 printf("Hello World!\r\n");
```

Advanced Code Generation

```
Floating Point         Options
 ( ) None               [X] Generate underbars
 (•) Emulation          [ ] Line numbers debug info
 ( ) 8087               [X] Debug info in OBJs
 ( ) 80287              [X] Fast floating point
                        [ ] Fast huge pointers
Instruction Set         [ ] Generate COMDEFs
 ( ) 8088/8086          [ ] Automatic far data
 (•) 80186
 ( ) 80286              Far Data Threshold  32767
```

F1 Help │ Use 80188/186 enhanced instructions

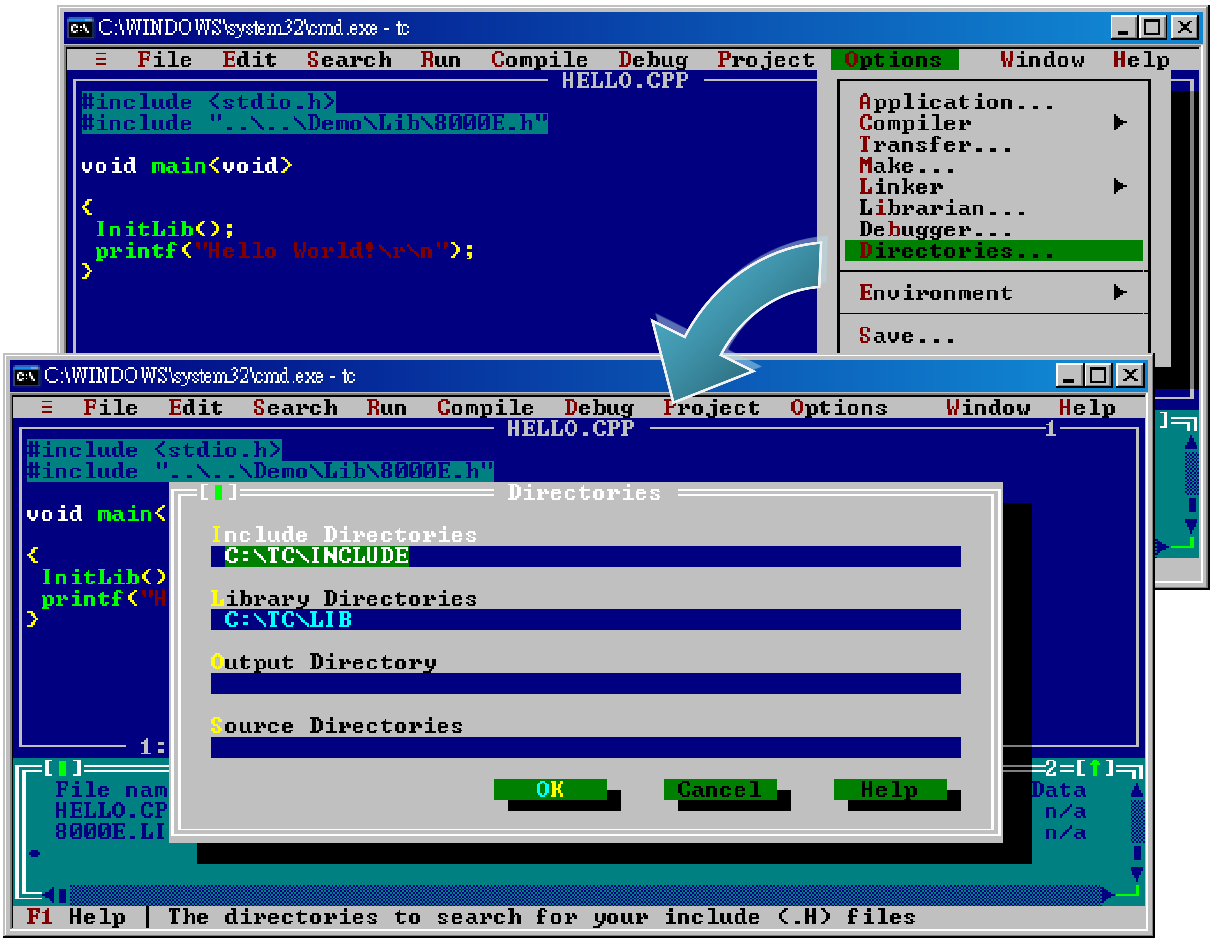
### Step 10: Set the TC compiler include and library directories

i. Select “Directories…” from the “Options” menu

ii. On “Include Directories” option, specify the header file

iii. On “Library Directories” option, specify the function library file

iv. Select “OK”



```
#include <stdio.h>
#include "..\..\Demo\Lib\8000E.h"

void main(void)

{
 InitLib();
 printf("Hello World!\r\n");
}
```

Include Directories: C:\TC\INCLUDE

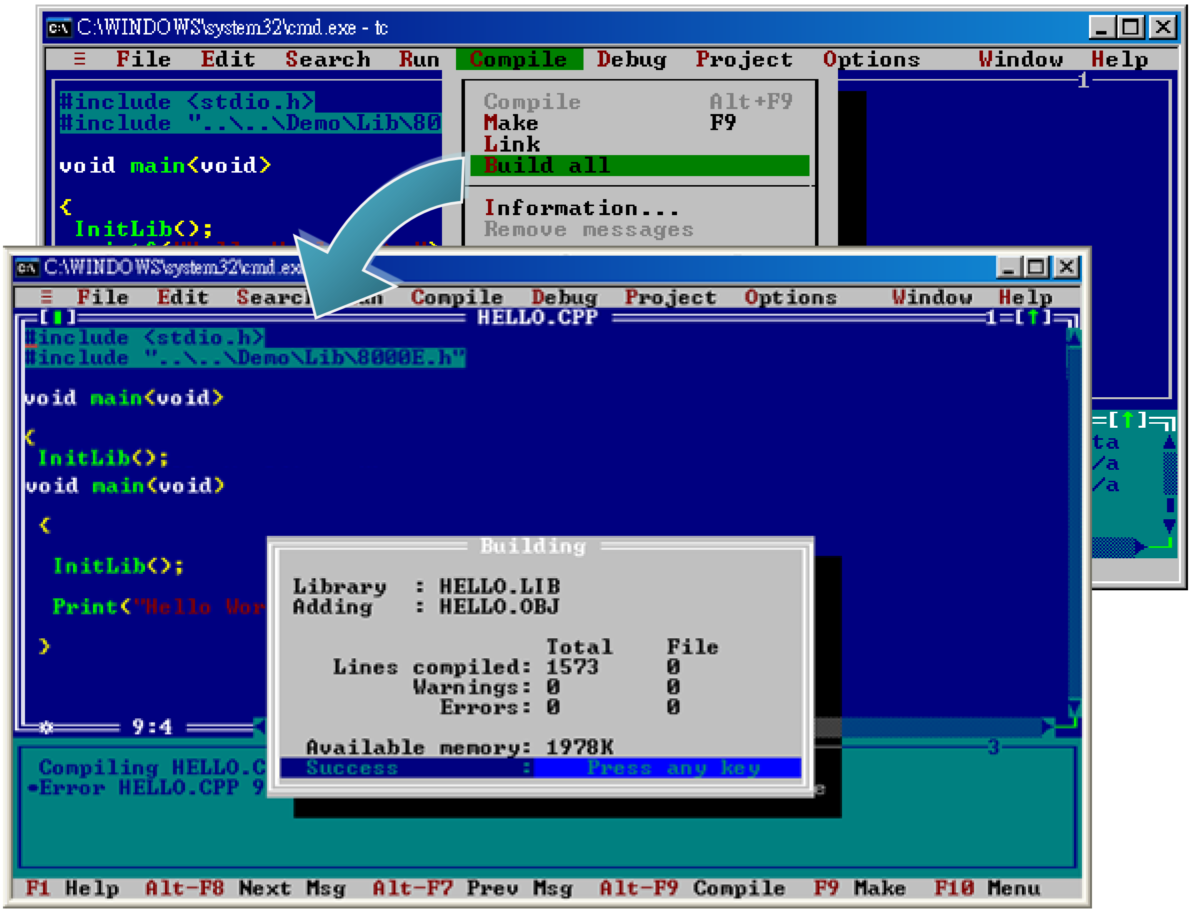
Library Directories: C:\TC\LIB

Output Directory:

Source Directories:

F1 Help | The directories to search for your include (.H) files

## Step 11: Select “Build all” from the “Compile” menu to build the project


C:\WINDOWS\system32\cmd.exe - tc
≡ File Edit Search Run Compile Debug Project Options Window Help
Compile Alt+F9
Make F9
Link
Build all
Information...
Remove messages
C:\WINDOWS\system32\cmd.exe
HELLO.CPP
9:4
F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt-F9 Compile F9 Make F10 Menu

```
#include <stdio.h>
#include "..\..\Demo\Lib\8000E.h"

void main(void)

{
 InitLib();
void main(void)

 {

  InitLib();

  Print("Hello Wor

 }
```

```
                    Building

Library   : HELLO.LIB
Adding    : HELLO.OBJ

                    Total     File
   Lines compiled:  1573      0
         Warnings:  0         0
           Errors:  0         0

Available memory:   1978K
Success           :  Press any key
```

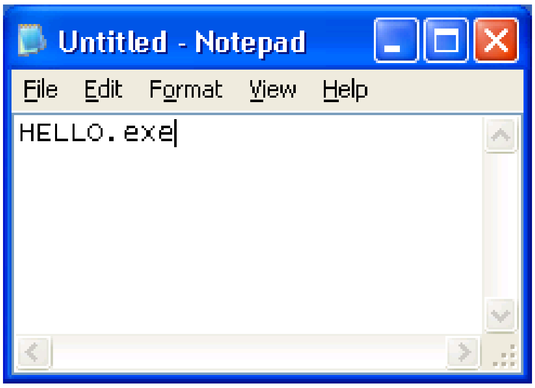
```
 Compiling HELLO.C
•Error HELLO.CPP 9
```

### Step 12: Configure the operating mode

For more detailed information about this process, please refer to section “2.3. Configuring the Boot Mode”

### Step 13: Create an autoexec.bat file

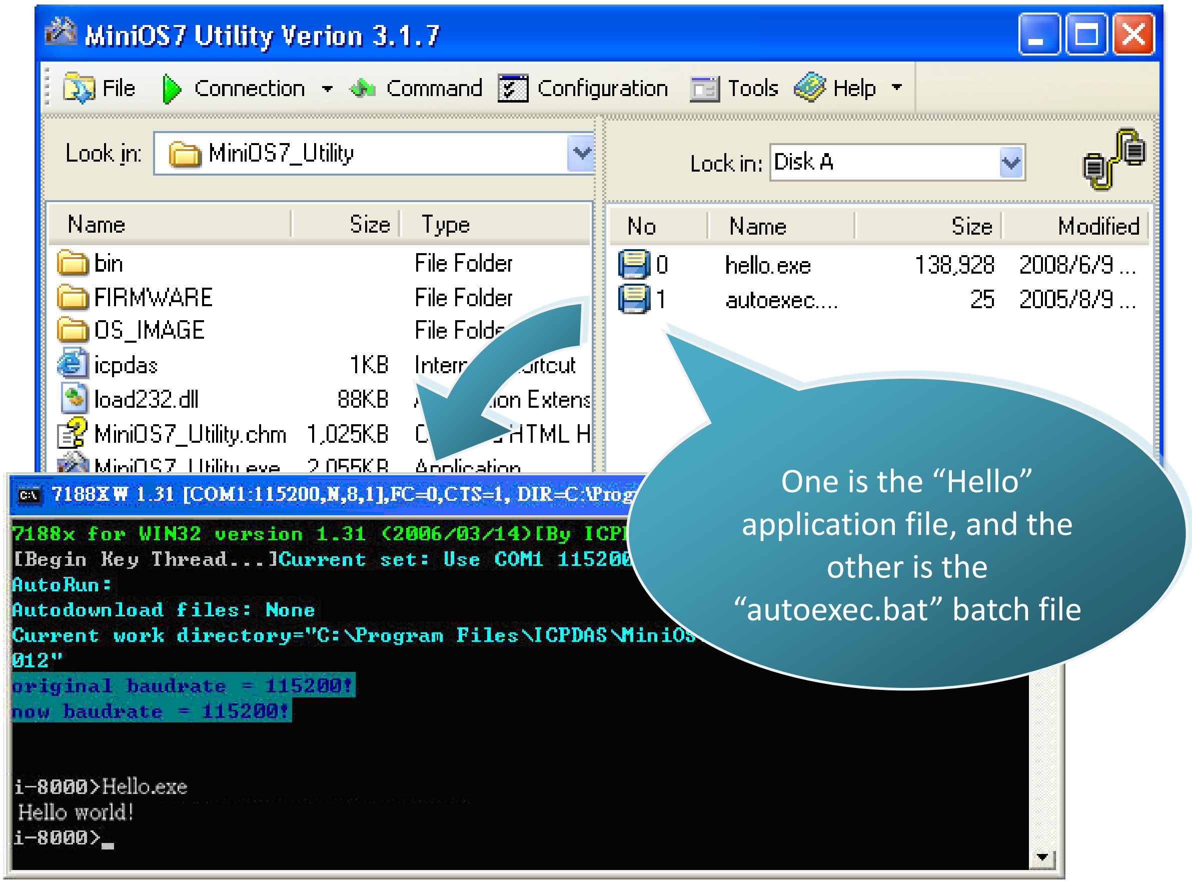
i. Open the “Notepad”
ii. Type the “HELLO.exe”
iii. Save the file as autoexec.bat



```
HELLO.exe
```

### Step 14: Upload programs to I-8000 using MiniOS7 Utility

For more detailed information about this process, please refer to section “2.5.2. Uploading and Executing Programs”



| Name | Size | Type |
|---|---|---|
| bin | | File Folder |
| FIRMWARE | | File Folder |
| OS_IMAGE | | File Folde |
| icpdas | 1KB | Intern ortcut |
| load232.dll | 88KB | A ion Extens |
| MiniOS7_Utility.chm | 1,025KB | C HTML H |
| MiniOS7_Utility.exe | 2,055KB | Application |

| No | Name | Size | Modified |
|---|---|---|---|
| 0 | hello.exe | 138,928 | 2008/6/9 ... |
| 1 | autoexec.... | 25 | 2005/8/9 ... |

```
7188XW 1.31 [COM1:115200,N,8,1],FC=0,CTS=1, DIR=C:\Prog
7188x for WIN32 version 1.31 (2006/03/14)[By ICP
[Begin Key Thread...]Current set: Use COM1 115200
AutoRun:
Autodownload files: None
Current work directory="C:\Program Files\ICPDAS\MiniOS
012"
original baudrate = 115200!
now baudrate = 115200!

i-8000>Hello.exe
 Hello world!
i-8000>_
```

One is the “Hello” application file, and the other is the “autoexec.bat” batch file

# 4. APIs and Demo References

There are several APIs and demo programs that have been designed for I-8000.

You can examine the APIs and demo source code, which includes numerous functions and comments, to familiarize yourself with the MiniOS7 APIs and quickly develop your own applications quickly by modifying these demo programs.

The following table lists the APIs grouped by functional category.

| API Description | Header File | Library |
|---|---|---|
| CPU driver | 8000E.H | 8000e.lib |
| Ethernet driver | Tcpip32.h | Tcpip32.lib |
| Modbus TCP driver | MBTCP.h | MBT8_172.LIB |
| Modbus RTU driver | MBRTU.h | MBR8_170.LIB |
| XServer | VXCOMM.H | XS8_3203.LIB |

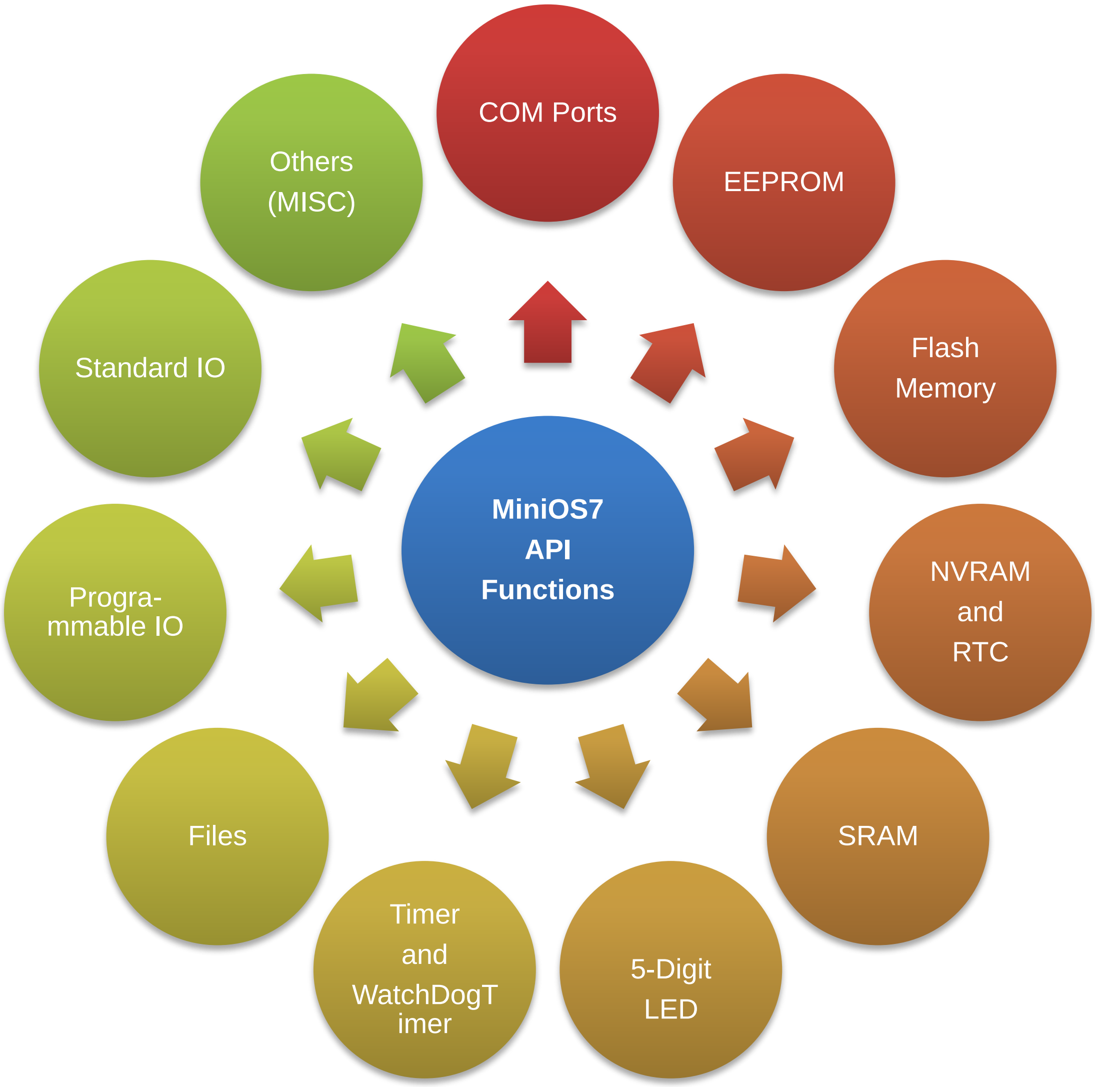
The following introduces the core API, MiniOS7 API, which is integrated into the I-8000 API set.

### Functions Library – 8000e.lib

This file contains the MiniOS7 API (Application Programming Interface) and has hundreds of pre-defined functions related to μPAC-5000

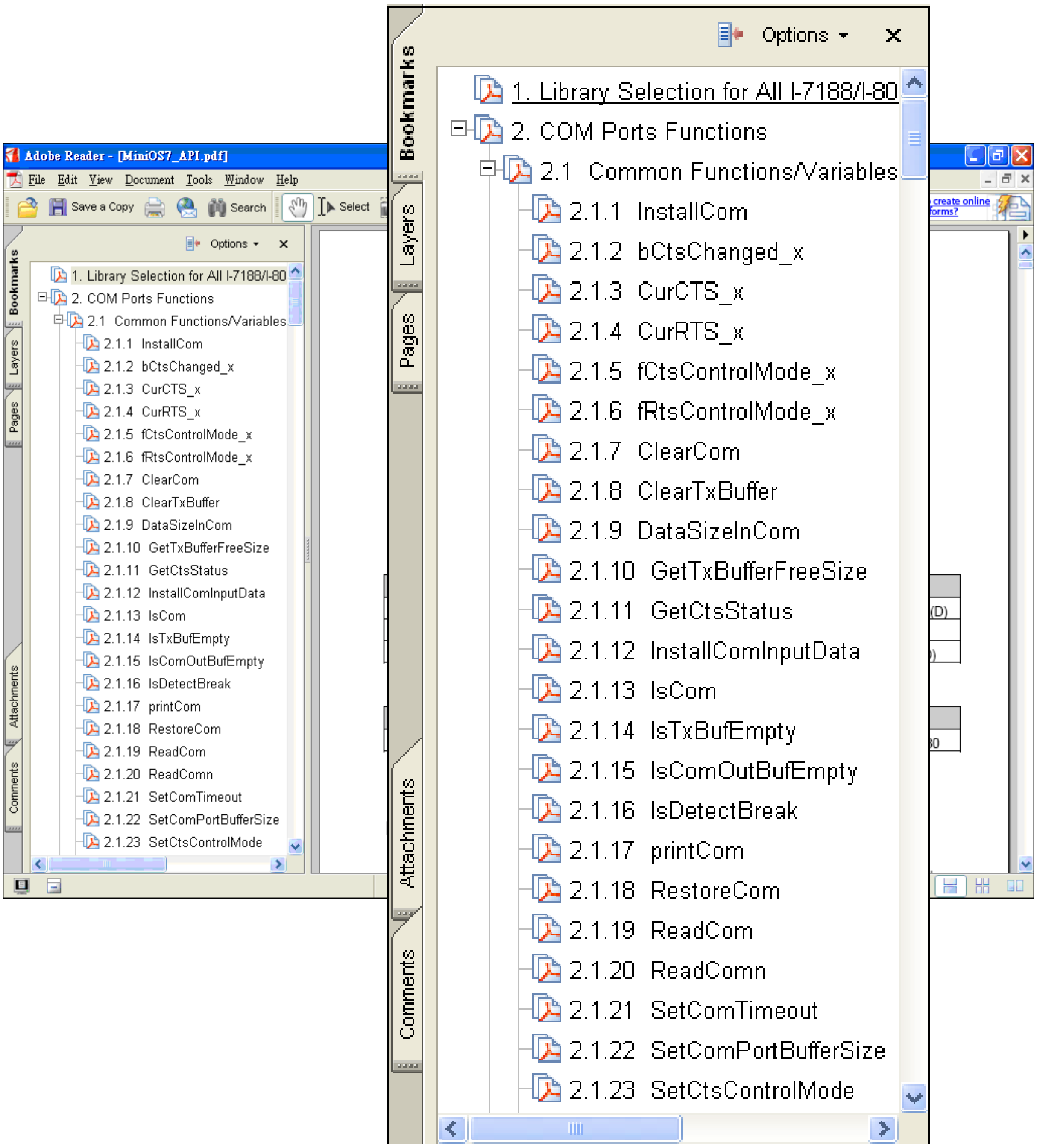
### Header File – 8000E.h

This file contains the forward declarations of subroutines, variables, and other identifiers used for the MiniOS7 API.



For full usage information regarding the description, prototype and the arguments of the functions, please refer to the “MiniOS7 API Functions User Manual” located at:

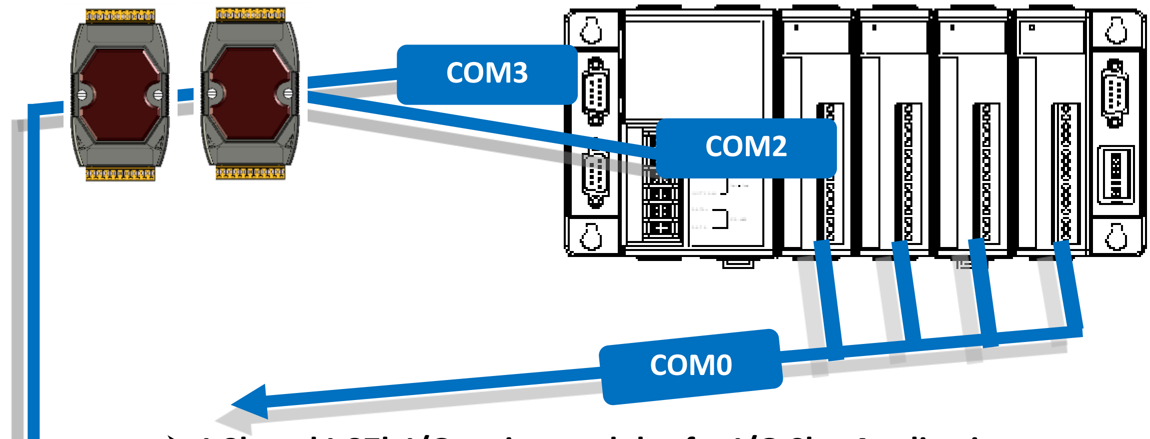
https://www.icpdas.com/en/download/show.php?num=1057&model=I-8831


Adobe Reader - [MiniOS7_API.pdf]
File Edit View Document Tools Window Help
Save a Copy
Search
Select
Options
Bookmarks
Layers
Pages
Attachments
Comments

1. Library Selection for All I-7188/I-80
2. COM Ports Functions
2.1 Common Functions/Variables
2.1.1 InstallCom
2.1.2 bCtsChanged_x
2.1.3 CurCTS_x
2.1.4 CurRTS_x
2.1.5 fCtsControlMode_x
2.1.6 fRtsControlMode_x
2.1.7 ClearCom
2.1.8 ClearTxBuffer
2.1.9 DataSizeInCom
2.1.10 GetTxBufferFreeSize
2.1.11 GetCtsStatus
2.1.12 InstallComInputData
2.1.13 IsCom
2.1.14 IsTxBufEmpty
2.1.15 IsComOutBufEmpty
2.1.16 IsDetectBreak
2.1.17 printCom
2.1.18 RestoreCom
2.1.19 ReadCom
2.1.20 ReadComn
2.1.21 SetComTimeout
2.1.22 SetComPortBufferSize
2.1.23 SetCtsControlMode

The following table lists the demo programs grouped by functional category.

| Folder | Demo | Explanation |
|---|---|---|
| File | Config_1_Basic | Reads information from text files(basic). |
| | Config_2_Advanced | Reads config file (text file)(advanced). |
| Hello | Hello_C | Reads library version and flash memory size. |
| | Hello_C++ | |
| Misc | Reset | Software's reset. |
| | Runprog | To select item and run it. |
| | Serial | To get 64-bit hardware unique serial number. |
| | Watchdog | Enabled WDT or bypass enable watchdog. |
| Smmi | SystemKey | User can operate systemkey simple and easily. |
| | Led | To control red led and 7-segment display. |
| Memory | S256 | To read or write 256K bytes battery backup. |
| DateTime | DateTime | User can read and write the date & time of RTC. |
| Timer | Please refer to the following location：<br>ftp://ftp.icpdas.com.tw/pub/cd/8000cd/napdos/8000/common/minios7/demo/ | |
| Com port | C_Style_IO | (1) Show how to write a function for input data.<br>(2) To get a string.<br>(3) To use C function: sscanf or just use Scanf(). |
| | Receive | Receive COM port.<br>Slv_COM.c is non-blocked mode<br>Receive.c is blocked mode. |
| | Slv_COM | Slave COM port demo for (request/reply) or (command/response) application. |
| | ToCom_In_Out | How to Read/Write the byte data via COM port. |
| **... more demo programs** | | |



### ➤ I-8k and I-87k I/O series modules for I/O Slot Applications

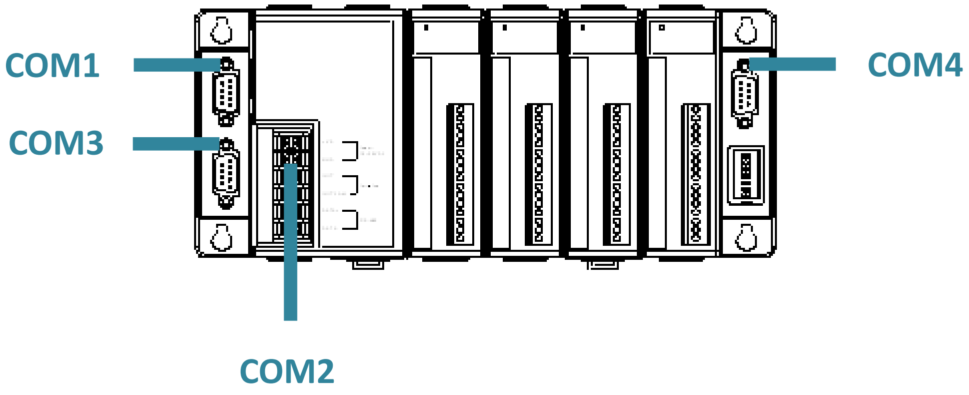
| Folder | Demo | Explanation |
|---|---|---|
| IO_in_Slot | 8K_DI | This demo program is used by 8K series DI modules, such as 8040, 8051., etc. |
| | 8073 | This demo program is used for 8073 General Functions. |
| | 87K_DI | This demo program is used by 87K series DI modules in Com0, such as 87040, 87051, etc. |
| | 87024 | This demo program is used by the 87024 AO module. |
| **… more demo programs** | | |

### ➤ I-7K and I-87k series modules for RS-485 Network Applications

| Folder | Demo | Explanation |
|---|---|---|
| 7K 87K_for_Com | 7K87K_DI_for_Com | "COM Port" can be used to connect and controll-7k or I-87k series modules.<br>■ For iPAC-8000 module and can use, COM2, COM3.<br>■ For iPAC-8000 module and (CPU 40 and 80M) can use, COM3, COM4. |
| | 7K87K_DO_for_Com | |
| | 7K87K_AI_for_Com | |
| | AO_22_26_for_Com | |
| | AO_024_for_Com | |

# 4.1. API for COM Port

The I-8000 provides four built-in COM ports, COM1 , COM2, COM3 and COM4.



### 4.1.1. Types of COM port functions

There are two types of functions below for using COM port.

1. MiniOS7 COM port functions
2. (C style) Standard COM port functions

#### Tips & Warnings

(C style) Standard COM port functions only can be used with the COM1, if you use the COM1 port, you'll have the alternative of MiniOS7 COM ports functions or (C style) Standard COM port functions. If you choose the ones, then another cannot be used.

Summarize the results of the comparison between MiniOS7 COM port functions and (C style) Standard COM port functions:

| Types of Functions | COM Port | Buffer | Functions | | | | |
|---|---|---|---|---|---|---|---|
| MiniOS7 COM port | 1, 2, etc. | 1 KB | 1 KB | IsCom() | ToCom() | ReadCom() | printCom() |
| (C style) Standard COM port | 1 | 512 Bytes | 256 Bytes | Kbhit() | Puts()<br>Putch() | Getch() | Print() |

## 4.1.2. API for MiniOS7 COM port

### ➤ API for using COM ports

**1. InstallCom()**

Before any COM Port can be used, the driver must be installed by calling InstallCom().

**2. RestoreCom()**

If the program calls InstallCom(), the RestoreCom()must be called to release the COM Port driver.

### ➤ API for checking if there is any data in the COM port input buffer

3. IsCom()

Before reading data from COM port, the IsCom() must be called to check whether there is any data currently in the COM port input buffer.

### ➤ API for reading data from COM ports

4. ReadCom()

After IsCom() confirms that the input buffer contains data, the ReadCom() must be called to read the data from the COM port input buffer.

### ➤ API for sending data to COM ports

5. ToCom()

Before sending data to COM ports, the ToCom() must be called to send data to COM ports.

For example, reading and receiving data through the COM1.

```
#include <stdio.h>
#include “8000E.h”

void main(void)
{
 int quit=0, data;

 InitLib();      /* Initiate the upac5000 library */
 InstallCom(1, 115200, 8, 0, 1);      /* Install the COM1 driver */

 while(!quit)
 {
  if(IsCom(1))      /* Check if there is any data in the COM port input buffer */
  {
   data=ReadCom(1);      /* Read data from COM1 port */
   ToCom(1, data);      /* Send data via COM1 port */
   if(data==’q’) quit=1;      /* If ‘q’ is received, exit the program */
  }
 }
 RestoreCom(1);      /* Release the COM1 */
}
```

## ➤ API for showing data from COM ports

6. printCom()

Functions such as printfCom() in the C library allow data to be output from COM ports.

For example, showing data from the COM1 port.

```
#include <stdio.h>
#include “8000E.h”

void main(void)
{
 int i;

 /* Initiate the upac5000 library */
 InitLib();
 InstallCom(1, 115200, 8, 0, 1);      /* Install the COM1 driver */
 for (i=0;i<10;i++)
 {
  printCom(1,”Test %d\n\r”, i);
 }
 Delay(10);      /* Wait for all data are transmitted to COM port */
 RestoreCom(1);
}
```

### 4.1.3. API for standard COM port

The standard COM port is used to upload program from PC to the µPAC-5000.

**Tips & Warnings**

(C style) Standard COM port functions only can be used with the COM1 port, the following configurations of the COM1 port are fixed:

Baudrate = 115200 bps, Data format = 8 bits

Parity check = none, Start bit = 1, Stop bit = 1

- **API for checking if there is any data in the input buffer**

1. Kbhit()

Before reading data from standard I/O port, the kbhit() must be called to check whether there is any data currently in the input buffer.

- **API for reading data from standard I/O port**

2. Getch()

After kbhit() confirms that the input buffer contains data, the Getch() must be called to read data from the input buffer.

## ➤ API for sending data to standard I/O port

3. Puts() – For sending a string

Before sending data to standard I/O port, the Puts() must be called to send data to COM Port..

4. Putch( ) – For sending one character

Before sending data to standard I/O port, the Putch() must be called to send data to COM Port.

## ➤ API for showing data from standard I/O port

5. Print()

Functions such as Print() in the C library allow data to be output from the COM port.

For example, reading and receiving data through COM1.

```
#include<stdio.h>
#include “8000E.h”

void main(void)
{
 int quit=0, data;

 InitLib();      /* Initiate the upac5000 library */
 while(!quit)
 {
  if(Kbhit())      /* Check if any data is in the input buffer */
  {
   data=Getch();      /* Read data from COM1 */
   Putch(data);      /* Send data to COM1 */
   if(data==’q’) quit=1;      /* If ‘q’ is received, exit the program */
  }
 }
}
```

For example, showing data through COM1.

```
#include <stdio.h>
#include “8000E.h”

void main(void)
{
 int i;

 /* Initiate the upac5000 library */
 InitLib();
 for(i=0;i<10;i++)
 {
  Print(“Test %d\n\r”,i);
 }
}
```

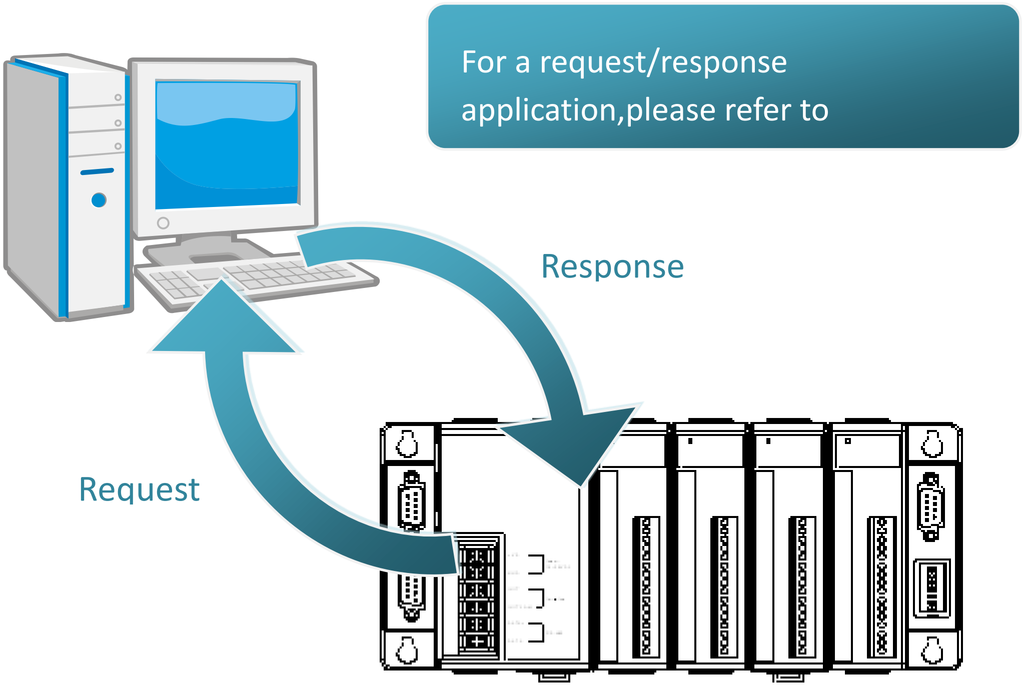
### 4.1.4. Port functions Comparison

For example, learning to show the ASCII code.

| MiniOS7 COM port functions | Standard COM port functions |
|---|---|
| #include<stdio.h> | #include<stdio.h> |
| #include “8000E.h” | #include “8000E.h” |
| | |
| void main(void) | void main(void) |
| { | { |
| unsigned char item; | unsigned char item; |
| InitLib(); | InitLib(); |
| InstallCom(1, 115200, 8, 0, 1); | Print("Hits any key.\n"); |
| printCom(1,”Hits any key.\n”); | Print("Hits the ESC to exit !\n"); |
| printCom(1,”Hit the ESC to exit!\n”); | for(;;) { |
| for(;;) { | if(kbhit()) { |
| if(IsCom(1)) { | item=Getch(); |
| item=ReadCom(1); | if(item==’q’){ |
| if(item==’q’){ | return; |
| return; | } |
| } | else{ |
| else{ | Print(”----------\n\r”); |
| printCom(1,”----------\n\r”); | Print(“char:“); |
| printCom(1,”char:”); | Putch(item); |
| ToCom(1,item); | Print("\n\rASCII(%c)\n\r”,item); |
| printCom(1,"\n\rASCII(%c)\n\r”,item); | Print(“Hex(%02X)\n\r”,item); |
| printCom(1,“Hex(%02X)\n\r”,item); | } |
| } | } |
| } | } |
| } | |
| Delay(10); | |
| RestoreCom(1); | } |
| } | |

### 4.1.5. Request/Response protocol define on COM port

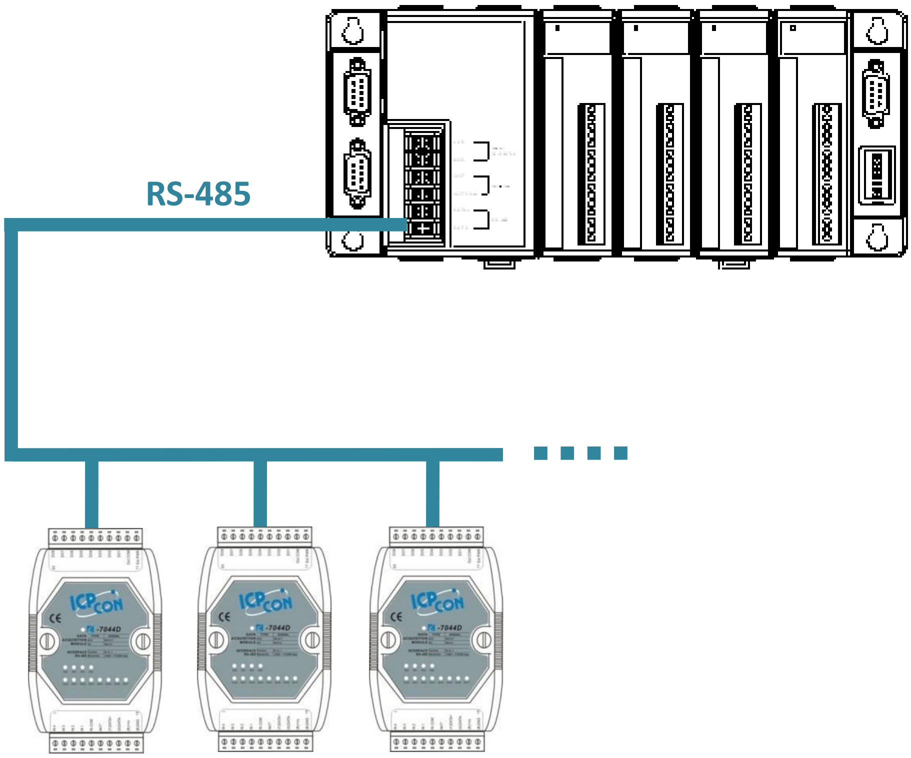
Request/Response communication is very typical protocol architecture. If you want to design a command set of communication protocol as table below, you can refer to “slave_com” demo.



| Request | Response |
|---|---|
| c1 | Debug information: Command1<br>Command1 |
| c2 | Debug information: Command2<br>Command2 |
| Q | Debug information: Quick program |
| Other command | Debug information: Unknown command |

## 4.2. API for I/O Modules

The I-8000are equipped with a RS-485 communication interface, COM2, to access I-7K series I/O modules for a wide range of RS-485 network application, as shown below.



### ➤ Steps for communicating with i-7K series I/O modules:

Step 1: Use Installcom() to install the COM port driver.

Step 2: Use SendCmdTo7000(2,…) to send commands

Step 3: Use ReceiveResponseFrom7000_ms() to get the response.

Step 4: Use RestoreCom() to restore the COM port driver

For example, to send a command ‘$01M’ to I-7K I/O module for getting the module name.

```
#include <stdio.h>
#include “8000E.h”

void main(void)
{
 unsigned char InBuf0[60];
 InitLib();      /* Initiate the 8000E library */

 InstallCom(1,115200,8,0,1);      /* Initiate the COM1 driver */
 InstallCom(2,115200,8,0,1);      /* Initiate the COM2 driver */

 SendCmdTo7000(2,”$01M”,0);      /* Send a command to COM2 */

 /* Timeout = 50ms, check sum disabled */
 ReceiveResponseFrom7000_ms(2,InBuf0,50,0);

 printCom(1,”Module Name = %s”, InBuf0);
 Delay(10);      /* Wait for all data are transmitted to COM port */
 RestoreCom(1);      /* Release the COM1 */

 RestoreCom(2);      /* Release the COM2 */
}
```

# 4.3. API for EEPROM

- The EEPROM contains 64 blocks (block 0 ~ 63), and each block has 256 bytes (address 0 ~ 255), with a total size of 16,384 bytes (16K) capacity.
- The default mode for EEPROM is write-protected mode.
- The system program and OS are stored in EEPROM that are allocated as shown below.

## ➤ API for writing data to theEEPROM

**1. EE_WriteEnable()**

Before writing data to the EEPROM, the EE_WriteEnable() must be called to write-enable the EEPROM.

**2. EE_WriteProtect()**

After the data has finished being written to the EEPROM, the EE_WriteProtect() must be called to in order to write-protect the EEPROM.

**3. EE_MultiWrite()**

After using the EE_WriteEnable() to write-enable EEPROM, the EE_MultiWrite()must be called to write the data.

## ➤ API for reading data from the EEPROM

**4. EE_MultiRead()**

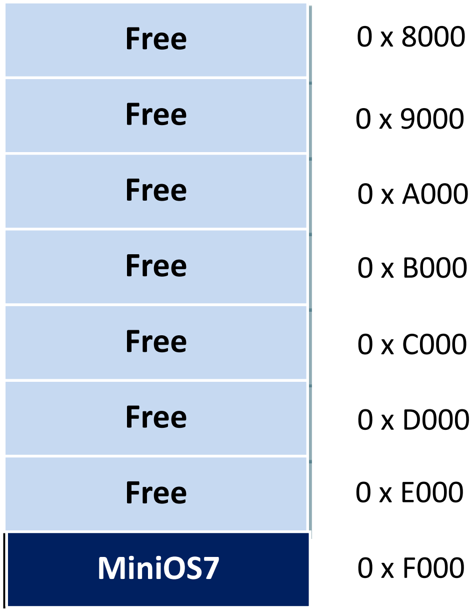
The EE_WriteEnable() must be called to read data from the EEPROM no matter what the current mode is.

For example, to write data to block1, address 10 of the EEPROM.

```
#include <stdio.h>
#include “8000E.h”

void main(void)
{
 int data=0x55, data2;

 InitLib();      /* Initiate the 8000E library */
 EE_WriteEnable();
 EE_MultiWrite(1,10,1,&data);
 EE_WriteProtect();

 EE_MultiRead(1,10,1,&data2);     /* Now data2=data=0x55 */
}
```

## 4.4. API for Flash Memory

- The I-8000 contains 512 Kbytes of Flash memory.
- MiniOS7 uses the last 64K bytes; the other parts of the memory are used to store user programs or data.
- Each bit of the Flash memory only e can be written from 1 to 0 and cannot be written from 0 to 1.
- Before any data can be written to the Flash memory, the flash must be erased, first which returns all data to 0xFF, meaning that all data bits are set to “1”. Once there is completed, new data can be written.



Free: 448 K bytes

MiniOS7: 64 K bytes

Total Size: 512 K bytes

## ➤ API for erasing data from the Flash Memory

### 1. EraseFlash()

The only way to change the data from 0 to 1 is to call the EraseFlash() function to erase a block from the Flash memory.

## ➤ API for writing data to the Flash Memory

### 2. FlashWrite()

The FlashWrite() must be called to write data to the Flash Memory.

## ➤ API for reading data from the Flash Memory

### 3. FlashRead()

The FlashRead() must be called to read data from the Flash Memory.

For example, to write an integer to segnment 0xD000, offset 0x1234 of the Flash memory.

```
#include <stdio.h>
#include “8000E.h”

void main(void)
{
int data=0xAA55, data2;
char *dataptr;
int *dataptr2;

InitLib();      /* Initiate the 8000E library */
EraseFlash(0xd000);       /* Erase a block from the Flash memory */
dataptr=(char *)&data;
FlashWrite(0xd000,0x1234, *dataptr++);
FlashWrite(0xd000,0x1235, *dataptr);

/* Read data from the Flash Memory (method 1) */
dataprt=(char *)&data2;
*dataptr=FlashRead(0xd000,0x1234);
*(dataptr+1)=FlashRead(0xd000,0x1235);

/* Read data from the Flash Memory (method 2) */
dataptr2=(int far *)_MK_FP(0xd000,0x1234);
data=*data;
}
```

# 4.5. API for NVRAM

- The I-8000 is equipped with an RTC (Real Time Clock), 31 bytes of NVRAM can be used to store data.
- NVRAM is SRAM, but it uses battery to keep the data, so the data in NVRAM does not lost its information when the module is power off.
- NVRAM has no limit on the number of the re-write times. (Flash and EEPROM both have the limit on re-write times)  If the leakage current is not happened, the battery can be used 10 years.

## ➤ API for writing data to the NVRAM

### 1. WriteNVRAM()

The WriteNVRAM() must be called in order to write data to the NVRAM.

## ➤ API for reading data from the NVRAM

### 2. ReadNVRAM()

The ReadNVRAM() must be called in order to write data to the NVRAM.

For example, use the following code to write data to the NVRAM address 0.

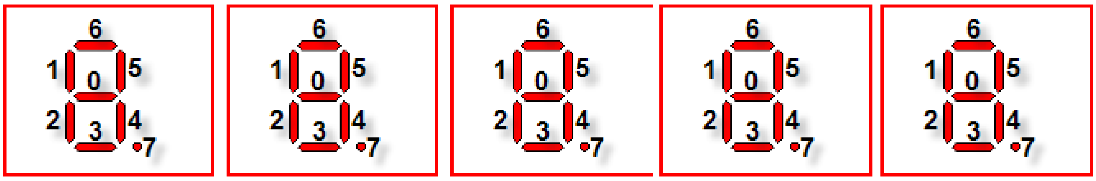
```
#include <stdio.h>
#include “8000E.h”

void main(void)
{
 int data=0x55, data2;
 InitLib();      /* Initiate the 8000E library */
 WriteNVRAM(0,data);
 data2=ReadNVRAM(0);      /* Now data2=data=0x55 */
}
```

For example, the following can be used to write an integer (two bytes) to NVRAM.

```
#include <stdio.h>
#include “8000E.h”

void main(void)
{
int data=0xAA55, data2;
char *dataptr=(char *)&data;

InitLib();      /* Initiate the 8000E library */
WriteNVRAM(0, *dataptr);      /* Write the low byte */
WriteNVRAM(1, *dataptr+1);      /* Write the high byte */
dataptr=(char *) &data2;
*dataptr=ReadNVRAM(0);      /* Read the low byte */
(*dataptr+1)=ReadNVRAM(1);      /* Read the high byte */
}
```

# 4.6. API for 5-Digital LED

The I-8000 contains a 5-Digit 7-SEG LED with a decimal point on the right-hand side of each digit, which be used to display numbers, IP addresses, time, and so on.



## ➤ API for starting the 5-Digit 7-SEG LED

### 1. Init5DigitLed()

Before using any LED functions, the Init5DigitLed() must be called to initialize the 5-Digit 7-SEG LED.

## ➤ API for displaying a message on the 5-Digit 7-SEG LED

### 2. Show5DigitLed()

After the Init5DigitLed() is used to initialize the 5-Digit 7-SEG LED, the Show5DigitLed() must be called to display information on the 5-Digits 7-SEG LED.

For example, use the following code to display “8000E” on the 5-Digit 7-SEG LED.

```
#include <stdio.h>
#include “8000E.h”

void main(void)
{
 InitLib();      /* Initiate the 8000E library */

 Init5DigitLed();
 Show5DigitLed(1,8);
 Show5DigitLed(2,0);
 Show5DigitLed(3,0);
 Show5DigitLed(4,0);
 Show5DigitLed(5,14);      /* The ASCII code for the letter ‘E’ is 14 */


}
```

# 4.7. API for Timer

- The I-8000 can support a single main time tick, 8 stop watch timers and 8 counts down timers.
- The I-8000 uses a single 16-bit timer to perform these timer functions, with a timer accuracy of 1 ms..

### ➤ API for starting the Timer

**1. TimerOpen()**

Before using the Timer functions, the TimerOpen() must be called at the beginning of the program.

### ➤ API for reading the Timer

**2. TimerResetValue()**

Before reading the Timer, the TimerResetValue() must be called to reset the main time ticks to 0.

**3. TimerReadValue()**

After the TimerResetValue() has reset the main time ticks to 0, the TimerReadValue() must be called to read the main time tick.

### ➤ API for stopping the Timer

**4. TimerClose()**

Before ending the program, the TimerClose() must be called to stop the Timer.

For example, the following code can be used to read the main time ticks from 0.

```
#include <stdio.h>
#include “8000E.h”

void main(void)
{
 Unsigned long time iTime;

 InitLib();      /* Initiate the 8000E library */
 TimerOpen();
 While(!quit)
 {
  If(Kbhit())
  TimerResetValue();      /* Reset the main time ticks to 0 */

  iTime=TimerReadValue();      /* Read the main time ticks from 0 */
 }
 TimerClose();      /* Stop using the I-8000 timer function */
}
```

# 4.8. API for WatchDog Timer (WDT)

- The I-8000 equips the MiniOS7, the small-cored operating system. MiniOS7 uses the Timer 2 (A CPU internal timer) as system Timer. It is 16-bits Timer, and generate interrupt every 1 ms. So the accuracy of system is 1 ms.
- The Watch Dog Timer is always enabled, and the system Timer ISR (Interrupt Service Routine) refreshes it.
- The system is reset by WatchDog. The timeout period of WatchDog is 0.8 seconds.

## ➤ API for refreshing WDT

### 1. EnableWDT()

The WDT is always enabled, before user's programming to refresh it, the EnableWDT() must be called to stop refreshing WDT.

### 2. RefreshWDT()

After EnableWDT() stop refreshing WDT, the RefreshWDT() must be called to refresh the WDT.

### 3. DisableWDT()

After user's programming to refresh WDT, the DisableWDT() should be called to automatically refresh the WDT.

For example, to refresh the Watchdog Timer.

```
#include <stdio.h>
#include “8000E.h”

void main(void)
{
 Unsigned long time iTime;

 InitLib();      /* Initiate the 8000E library */
 Enable WDT();
 While(!quit)
 {
  RefreshWDT();
  User_function();
 }
 DisableWDT();
}
```

# 5. I-8000 Updates

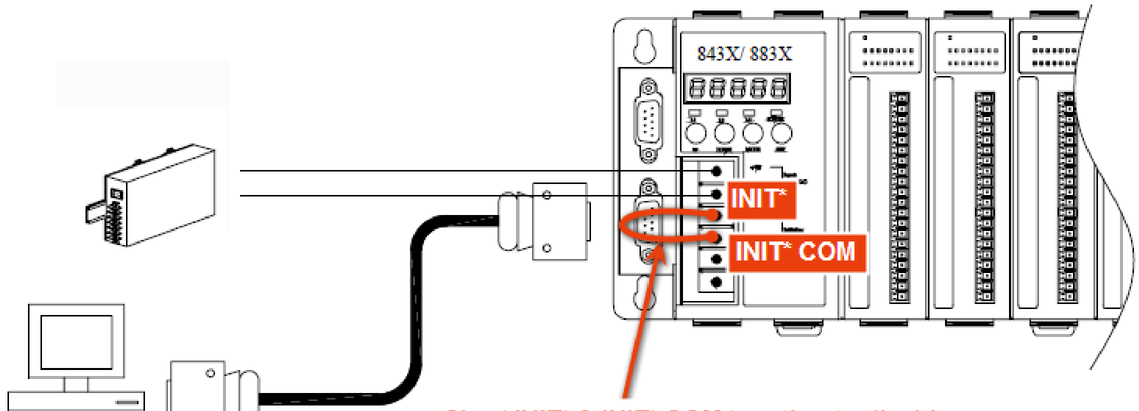
ICP DAS will continue to add additional features to OS and firmware of iPAC-8000 in the future, so we advise you to periodically check the ICP DAS web site for the latest updates.

iPAC-8000 updates services provides a software update service for iPAC-8000. It can be divided into two categories, OS updates and firmware updates.

Both the OS updates services and SDK updates services can be found separately on the CD that was provided with the package or by downloading the latest version from ICP DAS web site.

The OS image is stored in flash memory and can be updated to fix functionality issues or add additional features, so we advise you to periodically check the ICP DAS web site for the latest updates.

### Step 1: Boot the I-8000 into Initial mode



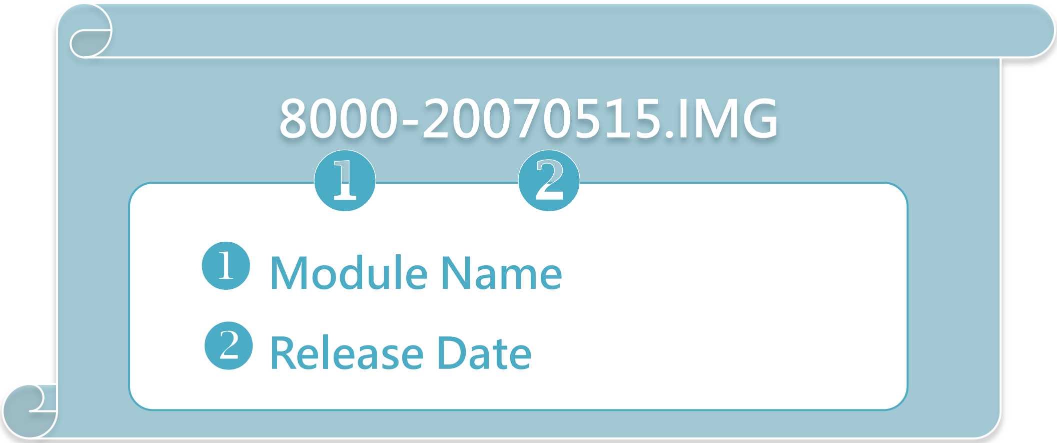
Short INIT* & INIT* COM together to disable AUTOEXEC.BAT in the beginning stage of power up.

### Step 2: Get the I-8000 OS image

The I-8000 OS imagecan be found separately on the CD that was provided with the package or by downloading the latest version from ICP DAS web site.

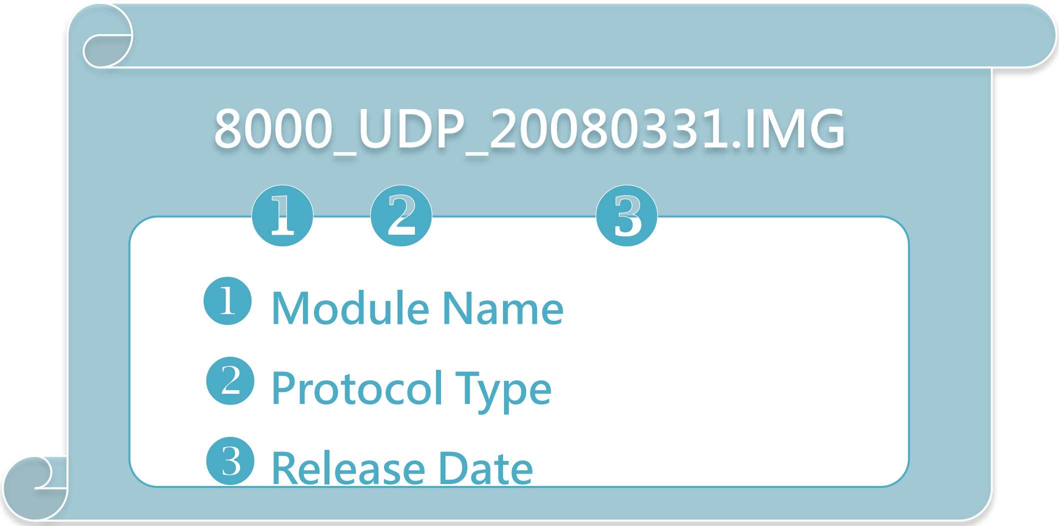
- **For I-8411/I-8811 serial embedded controller:**

https://www.icpdas.com/en/download/show.php?num=2767&model=I-8411



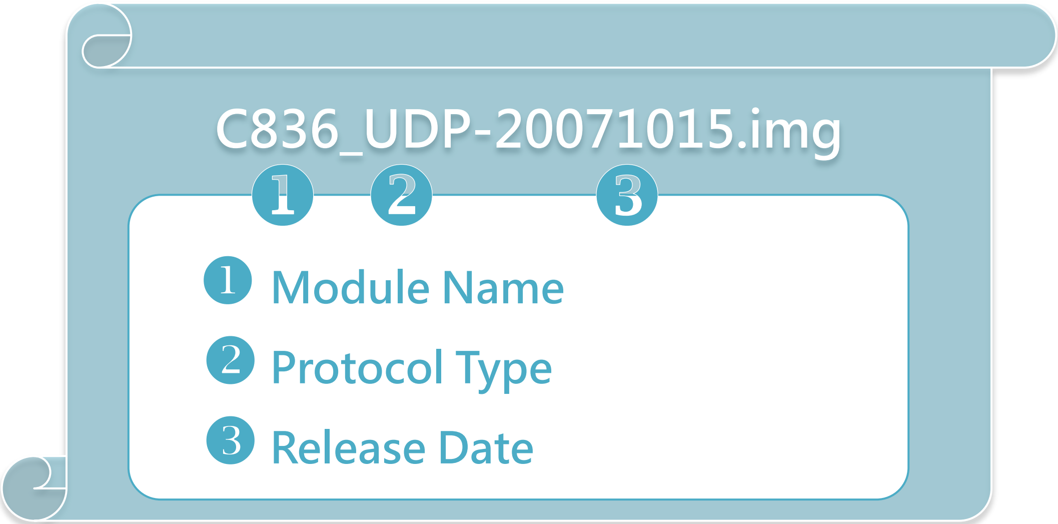
- **For I-8431/I-8831 ethernet embedded controller:**

https://www.icpdas.com/en/download/show.php?num=2769&model=I-8431



- **For I-8431-80/I-8831-80 ethernet embedded controller:**

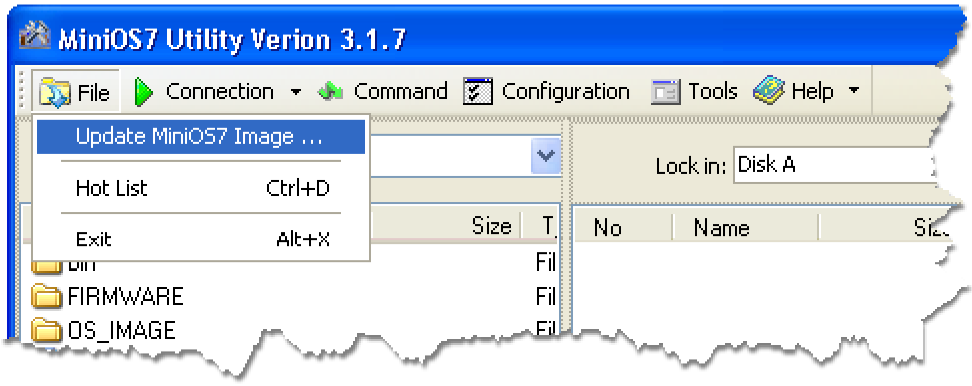
https://www.icpdas.com/en/download/show.php?num=2770&model=I-8431-80



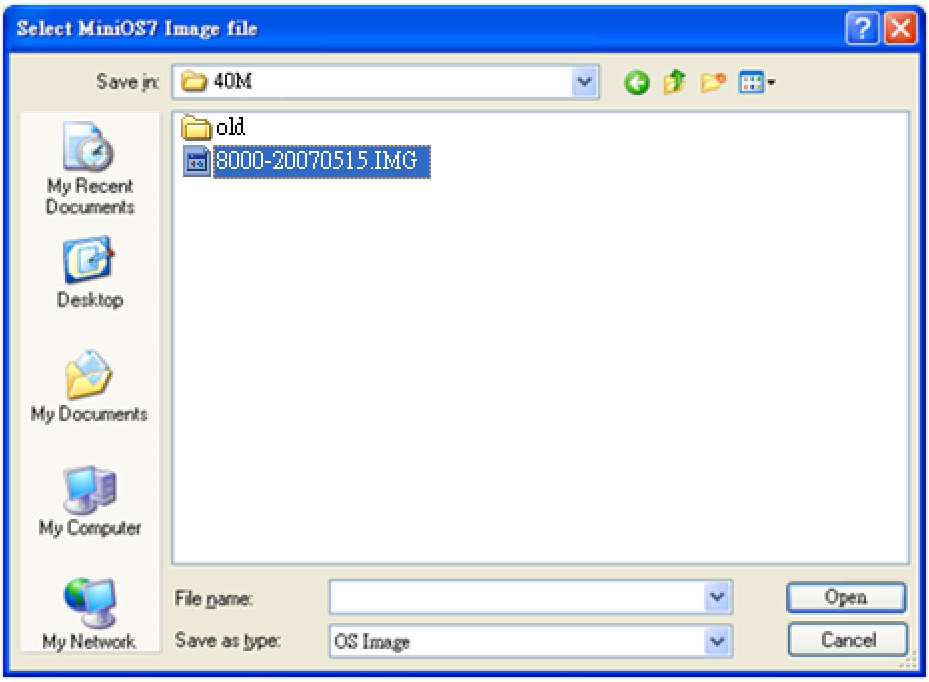
**Step 3: Establish a connection**

For more detailed information about this process, please refer to section “2.4.1. Establishing a connection”

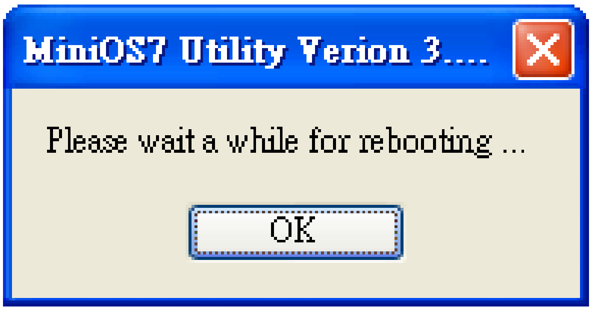
**Step 4: Click the “Update MiniOS7 Image …” from the “File” menu**



### Step 5: Select the latest version of the MiniOS7 OS image

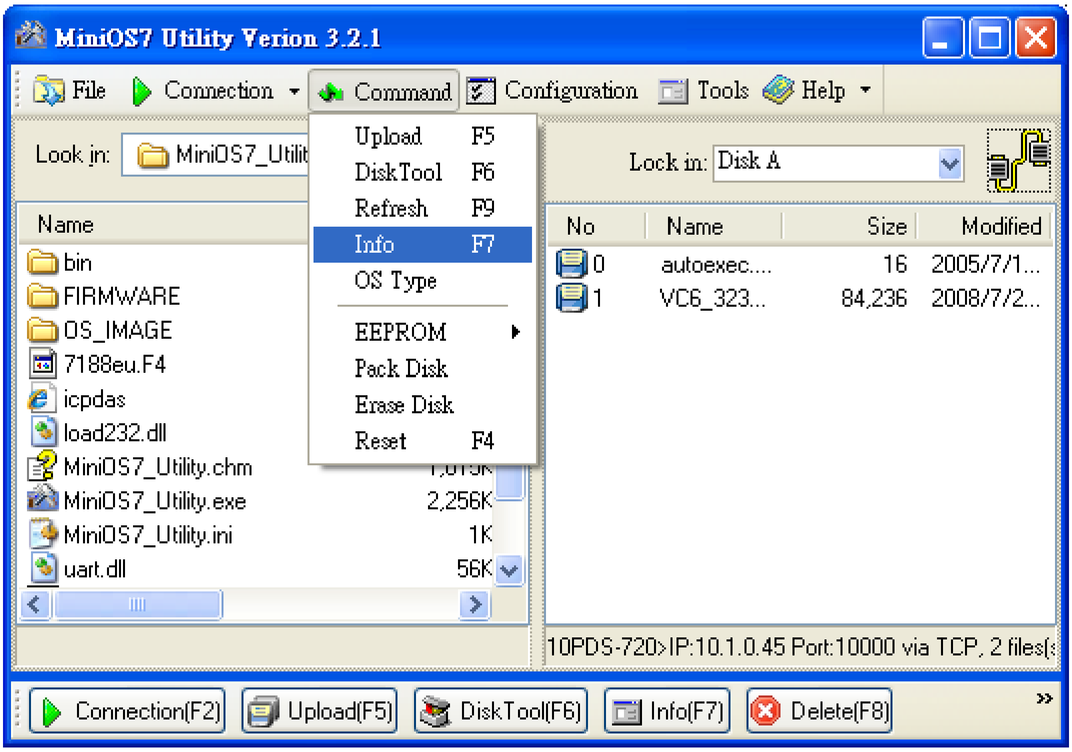


### Step 6: Click the “OK”



Please wait a while for rebooting ...

**Step 7: Click the “Info” from the “Command” menu to check the version of the OS image**



| No | Name | Size | Modified |
|---|---|---|---|
| 0 | autoexec.... | 16 | 2005/7/1... |
| 1 | VC6_323... | 84,236 | 2008/7/2... |

10PDS-720>IP:10.1.0.45 Port:10000 via TCP, 2 files(

# Appendix A. What is MiniOS7?

MiniOS7 is an embedded operating system design by ICP DAS. It is functionally equivalent to other brands of DOS, and can run programs that are executable under a standard DOS.

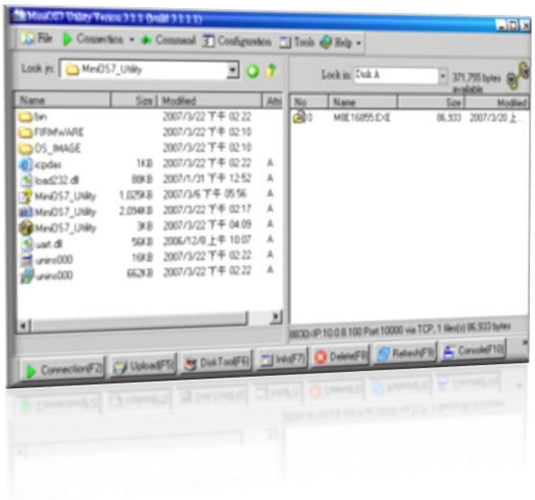
### Tip & Warnings

⚠ DOS is a set of commands or code that tells the computer how to process information. DOS runs programs, manages files, controls information processing, directs input and output, and performs many other related functions.

The following table summarizes the features of MiniOS7:

| Feature | MiniOS7 |
|---|---|
| Power-up time | 0.1 sec |
| More compact size | < 64 K bytes |
| Support for I/O expansion bus | Yes |
| Support for ASIC key | Yes |
| Flash ROM management | Yes |
| OS update (Upload) | Yes |
| Built-in hardware diagnostic functions | Yes |
| Direct control of 7000 series modules | Yes |
| Customer ODM functions | Yes |
| Free of charge | Yes |

# Appendix B. What is MiniOS7 Utility?



| Name | Size | Modified | Attr |
|---|---|---|---|
| bin | | 2007/3/22 下午 02:22 | |
| FIRMWARE | | 2007/3/22 下午 02:10 | |
| OS_IMAGE | | 2007/3/22 下午 02:10 | |
| icpdas | 1KB | 2007/3/22 下午 02:22 | A |
| load232.dll | 88KB | 2007/1/31 下午 12:52 | A |
| MiniOS7_Utility | 1,025KB | 2007/3/6 下午 05:56 | A |
| MiniOS7_Utility | 2,094KB | 2007/3/22 下午 02:17 | A |
| MiniOS7_Utility | 3KB | 2007/3/22 下午 04:09 | A |
| uart.dll | 56KB | 2006/12/8 上午 10:07 | A |
| unins000 | 16KB | 2007/3/22 下午 02:22 | A |
| unins000 | 662KB | 2007/3/22 下午 02:22 | A |

| No | Name | Size | Modified |
|---|---|---|---|
| 0 | M8E16855.EXE | 86,933 | 2007/3/20 上... |

8830-IP:10.0.8.100 Port 10000 via TCP, 1 file(s) 86,933 bytes

MiniOS7 Utility is a tool for configuring, uploading files to all products embedded with ICP DAS MiniOS7.

Since version 3.1.1, the Utility can allow users remotely access the controllers (7188E, 8000E..., etc) through the Ethernet.

| **Functions** | **Including frequently used tools** |
|---|---|
| ● Supported connection ways | a. 7188XW |
| 1. COM port connection (RS-232) | b. 7188EU |
| 2. Ethernet connection (TCP &UDP) | c. 7188E |
| (Supported since version 3.1.1) | ------------------------------ |
| ● Maintenance | d. SendTCP |
| 1. Upload file(s) | e. Send232 |
| 2. Delete file(s) | f. VxComm Utility |
| 3. Update MiniOS7 image | |
| ● Configuration | |
| 1. Date and Time | |
| 2. IP address | |
| 3. COM port | |
| 4. Disk size (Disk A, Disk B) | |
| ● Check product information | |
| 1. CPU type | |
| 2. Flash Size | |
| 3. SRAM Size | |
| 4. COM port number | |
| ..., etc. | |

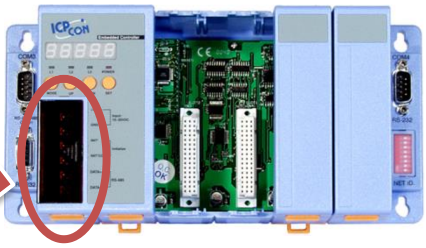
Upload location:
https://www.icpdas.com/en/product/guide+Software+Development__Tools+MiniOS7

# Appendix C. RS-485 Master/Slave Application

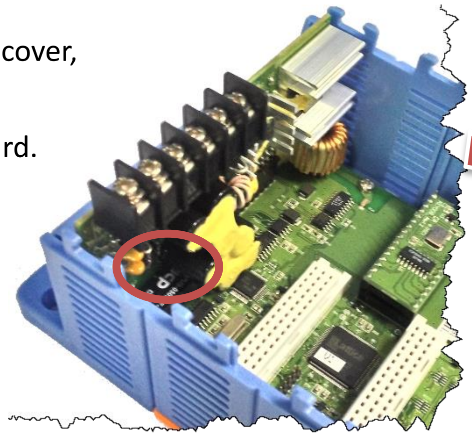
The RS-485 network is based on a Master-Slave architecture that consists of a single Master device and one or more Slave devices.

The I-8000 provides two RS-485 serial ports, COM2 and COM3. Each RS-485 port includes a pull-hgh/pull-down resistor that can be set as a Master or Slave, and only one of the two ports may be selected for Master operation. Here we recommend that you choose the COM2 to have protection from from external signals.

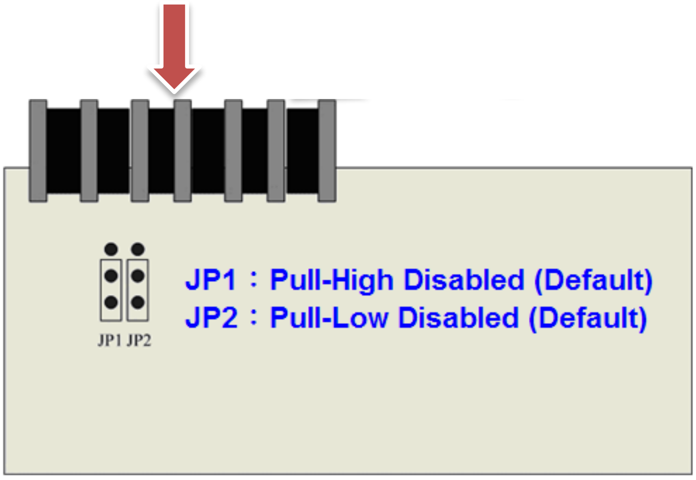
**Location of the COM2 pull-high/pull-low resistor**



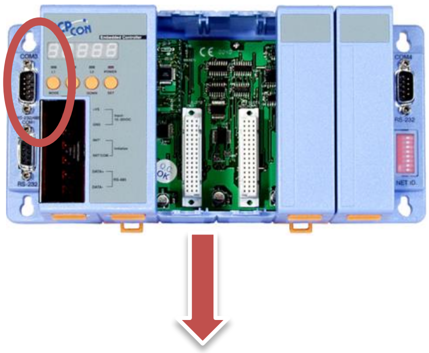
Remove the maincover, you can see the I-8000 power board.



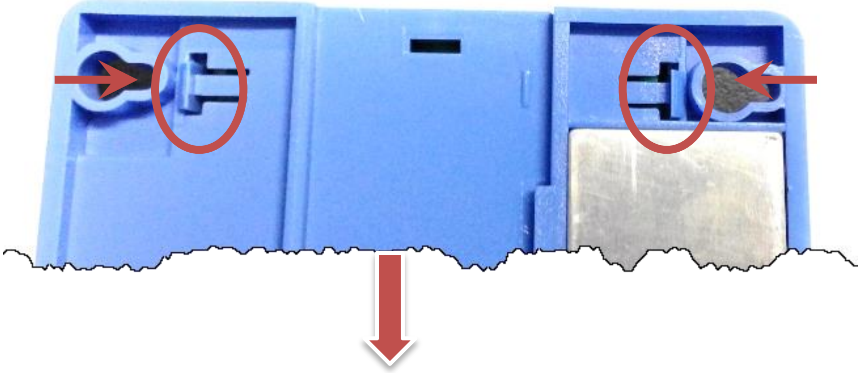
On the I-8000 power board, you can see the COM2 pull-high/pull-low resistor.


JP1 JP2
JP1 : Pull-High Disabled (Default)
JP2 : Pull-Low Disabled (Default)

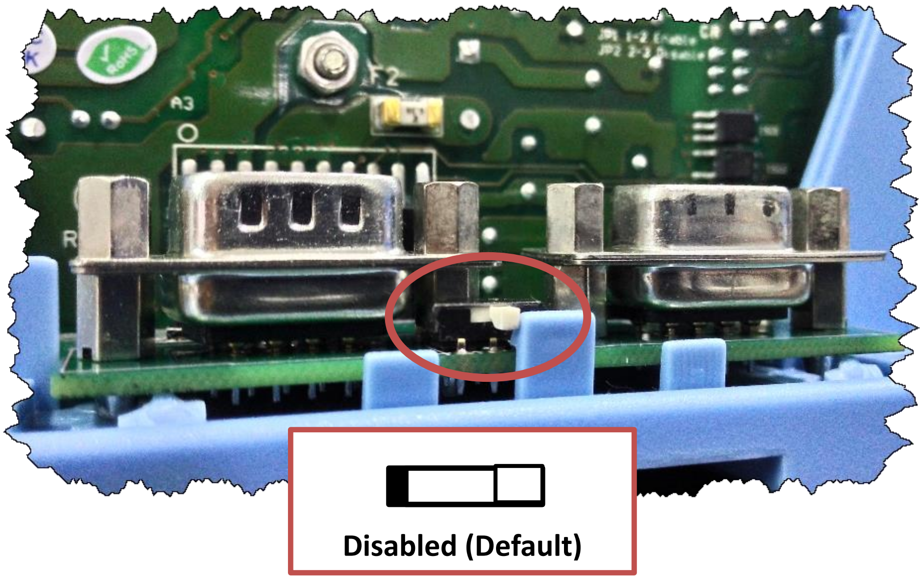
### Location of the COM3 pull-high/pull-low resistor



On the back side, pull the locking tabs toward inside



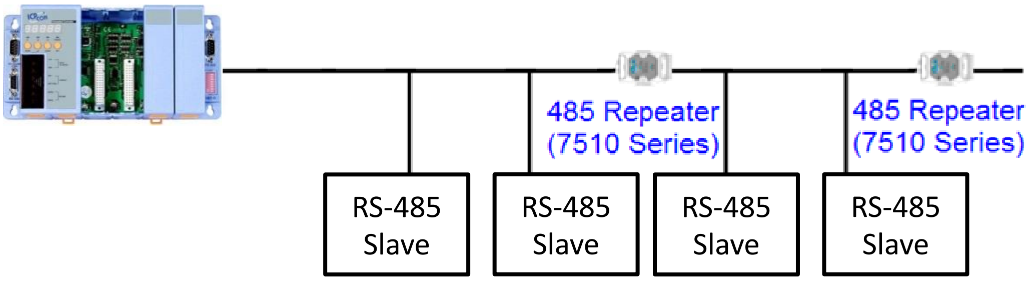
Remove the cover, you can see the COM3 pull-high/pull-low resistor.



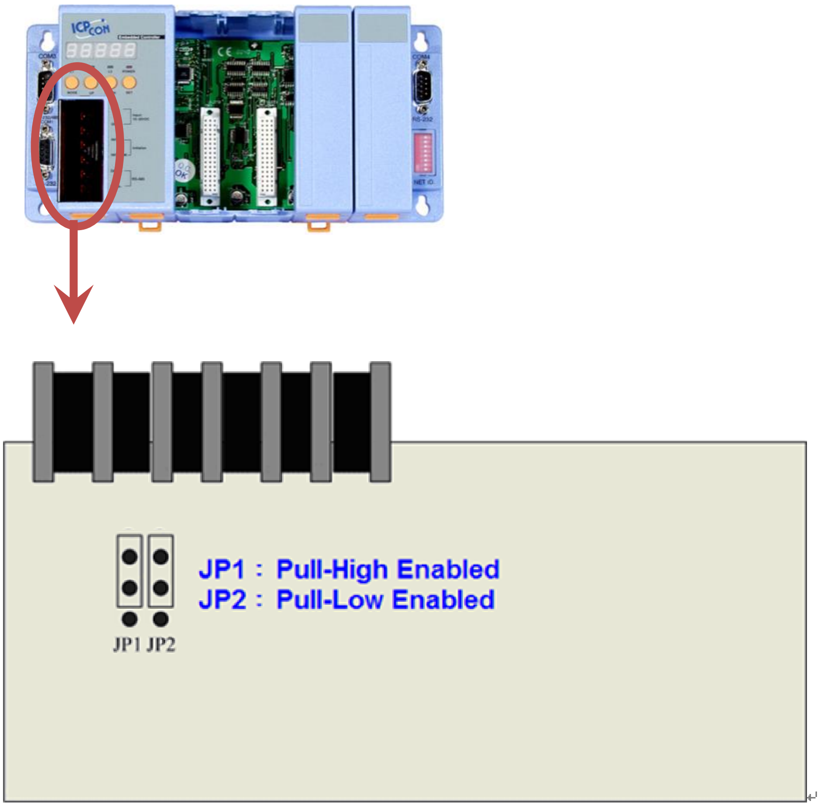
Disabled (Default)

# C.1. I-8000 as a Master

When one of I-8000 is set to Master, then all of the other devices must be set to Slave. If the network is up to 1.2 KM, it will need a repeater (7510 series) to extend the network length.

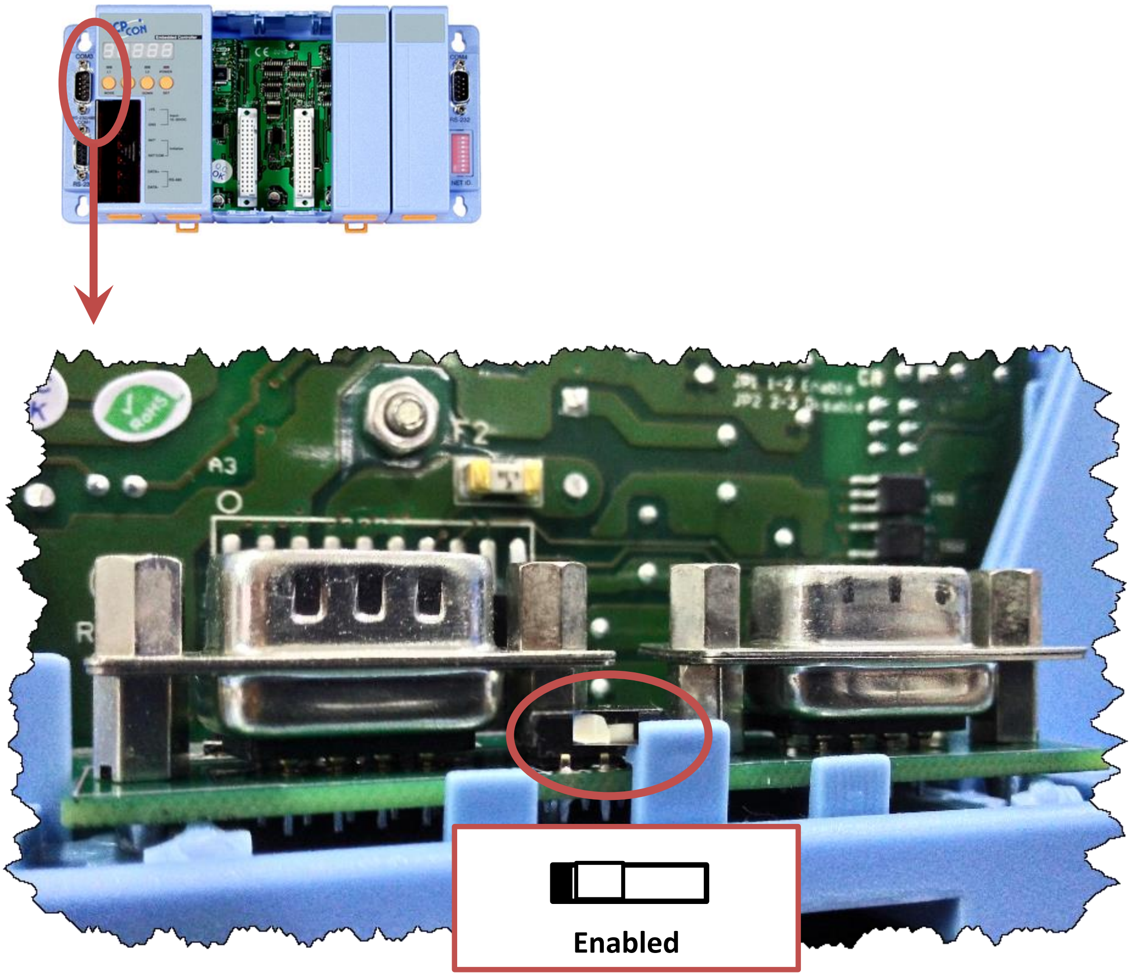


When I-8000 as a master by using COM2, the status of the COM2 pull-high/pull-low are as below shown.



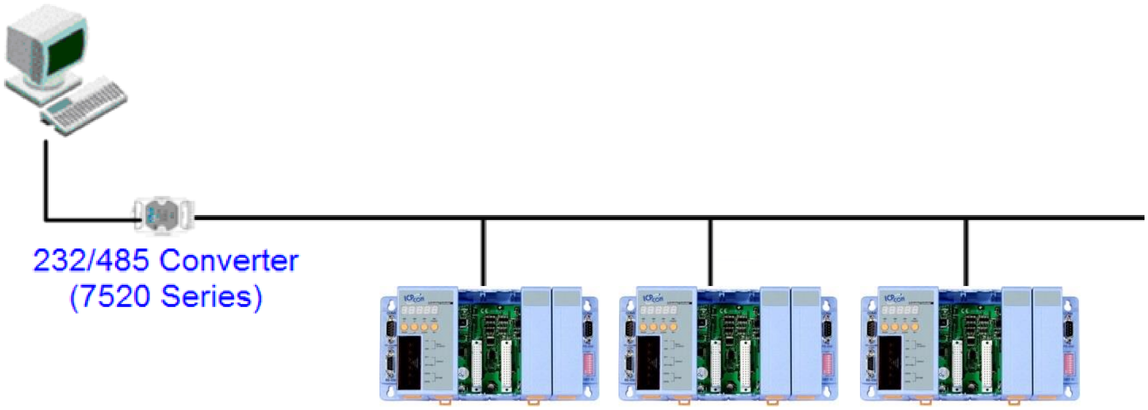
When I-8000 as a master by using COM3, the status of the COM3 pull-high/pull-low are as below shown.

Here we recommend that you choose the COM2 to have protection from from external signals.

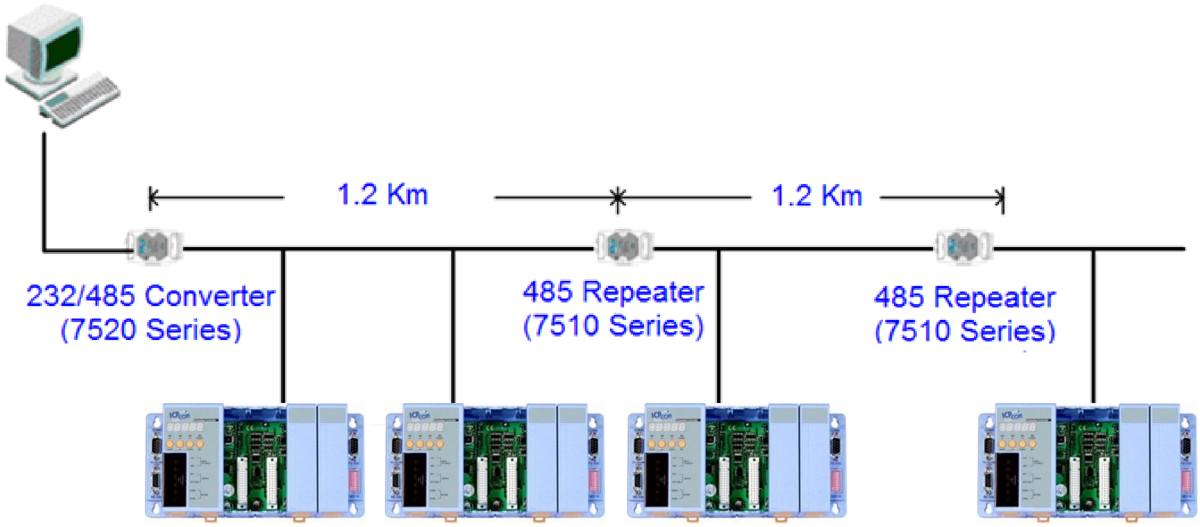


# C.1. I-8000 as a Slave

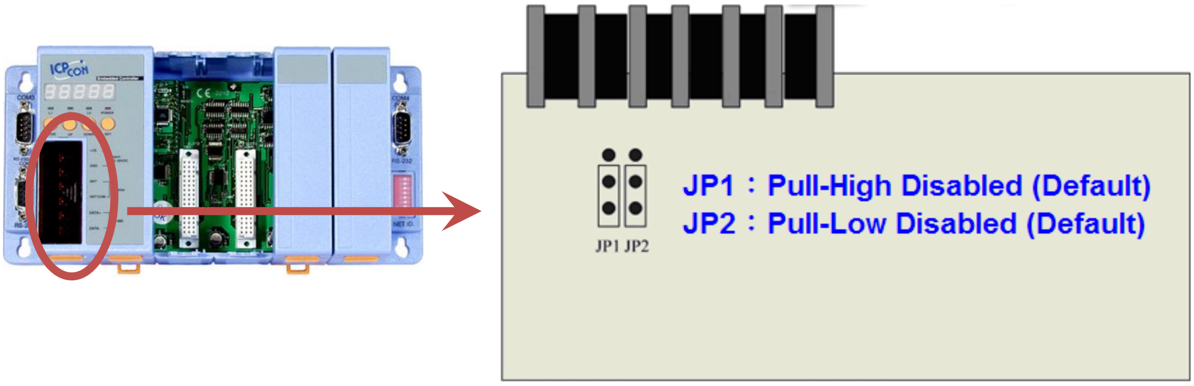
For most of application, when using one 7520 series as RS-232/485 converter, its pull-high/pull-low resistors are set to enabled. Then the I-8000and all the other devices on this network must be slave mode (the pull-high/pull-low resistors must be disabled).



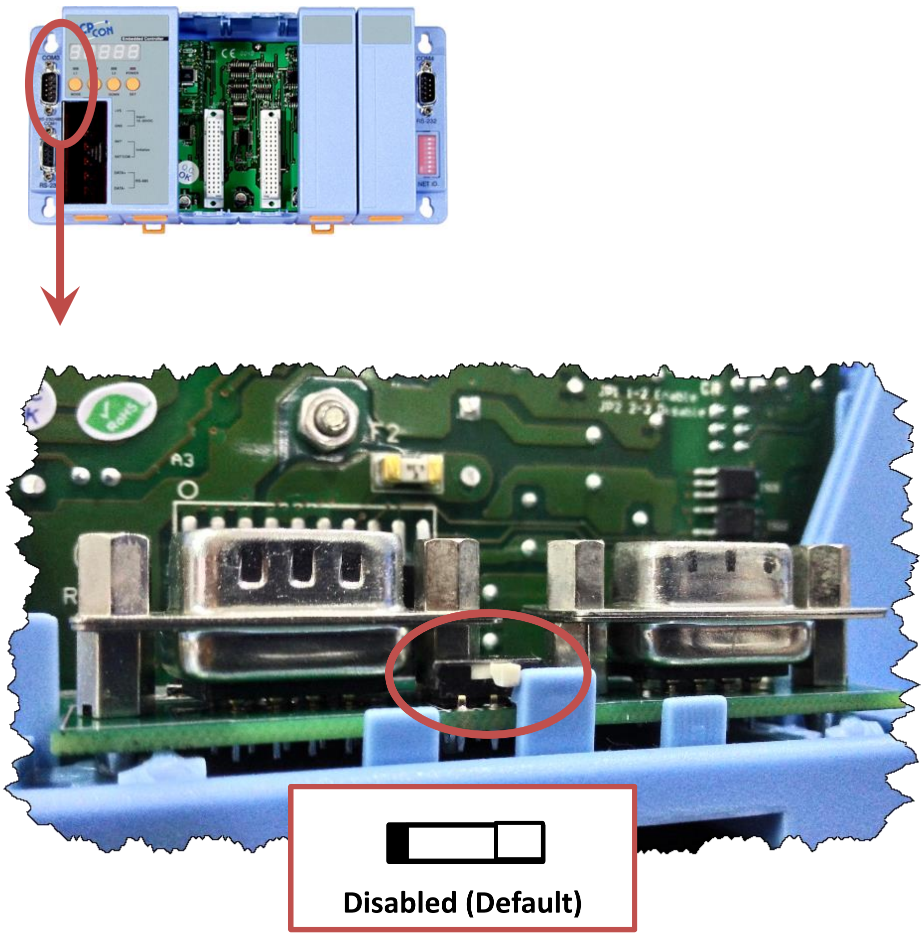
If there are repeaters on the RS-485 network, there will be pull-high/pull-low resistors on both sides of the repeaters (7510 series)



When I-8000 as a slave by using COM2, the status of the COM2 pull-high/pull-low are as below shown.


JP1：Pull-High Disabled (Default)
JP2：Pull-Low Disabled (Default)
JP1 JP2

When I-8000 as a slave by using COM3, the status of the COM3 pull-high/pull-low are as below shown.



# Appendix D. More C Compiler Settings

This section describes the setting of the following compilers:

- Turbo C 2.01 Compiler
- BC++ 3.1 IDE
- MSC 6.00 Compiler
- MSVC 1.50 Compiler

# D.1. Turbo C 2.01

You have a couple of choices here, you can:

1: Using a command line

For more information, please refer to

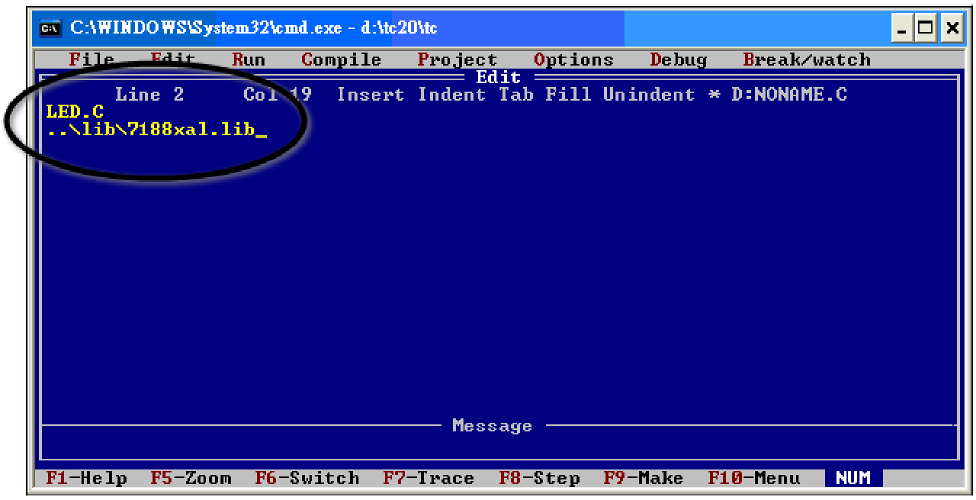
CD:\8000\NAPDOS\8000\841x881x\Demo\hello\Hello_C\gotc.bat
tcc -Ic:\tc\include -Lc:\tc\lib hello1.c ..\..\lib\8000e.lib

2: Using the TC Integrated Environment

### Step 1: Executing the TC 2.01

### Step 2: Editing the Project file

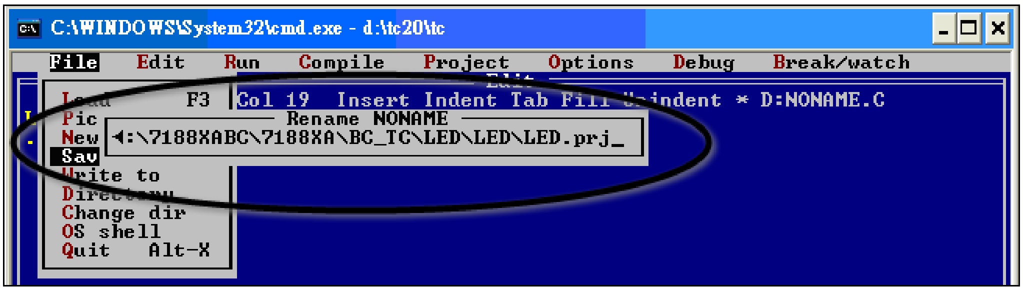
Adding the necessary library and file to the project



```
LED.C
..\lib\7188xal.lib_
```

C:\WINDOWS\System32\cmd.exe - d:\tc20\tc
File Edit Run Compile Project Options Debug Break/watch
Edit
Line 2 Col 19 Insert Indent Tab Fill Unindent * D:NONAME.C
Message
F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F10-Menu NUM

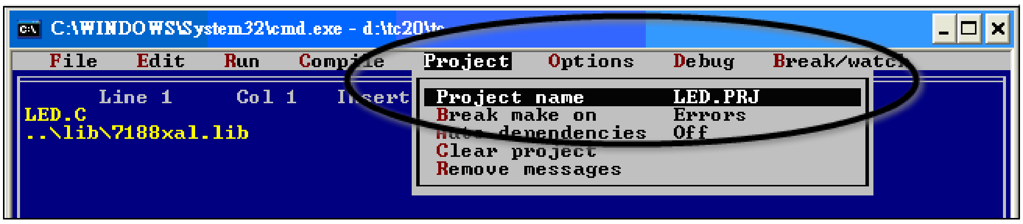
### Step 3: Save the project and entering a name, such as LED.prj



C:\WINDOWS\System32\cmd.exe - d:\tc20\tc
File Edit Run Compile Project Options Debug Break/watch
Load F3
Pick
New
Save
Write to
Directory
Change dir
OS shell
Quit Alt-X

```
Col 19  Insert Indent Tab Fill Unindent * D:NONAME.C
Rename NONAME
d:\7188XABC\7188XA\BC_TC\LED\LED\LED.prj_
```

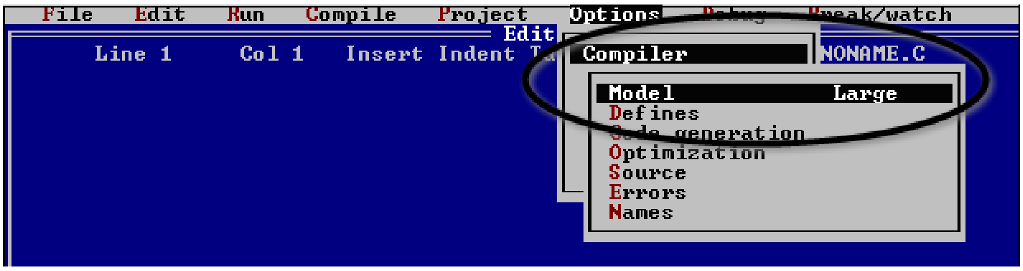
### Step 4: Load the Project



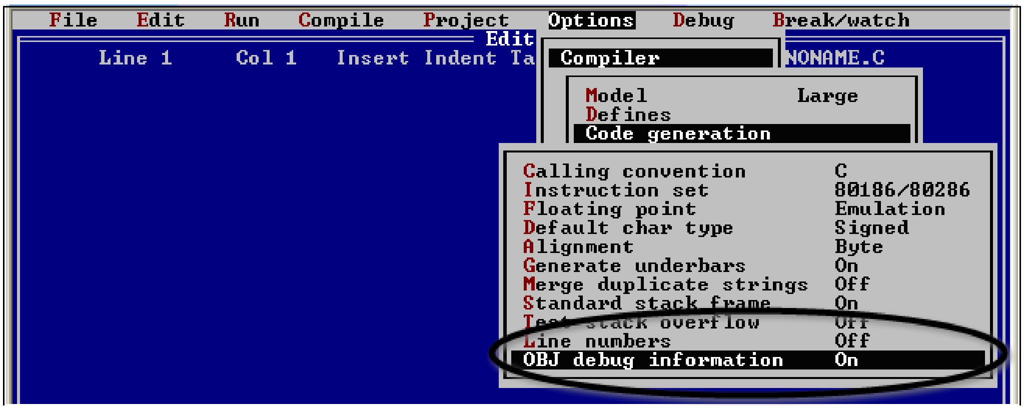
C:\WINDOWS\System32\cmd.exe - d:\tc20\tc
File Edit Run Compile Project Options Debug Break/watch

```
Line 1     Col 1    Insert
LED.C
..\lib\7188xal.lib

Project name        LED.PRJ
Break make on       Errors
Auto dependencies   Off
Clear project
Remove messages
```

### Step 5: Change the Memory model (Large for I-8000.lib) and set the Code Generation to 80186/80286

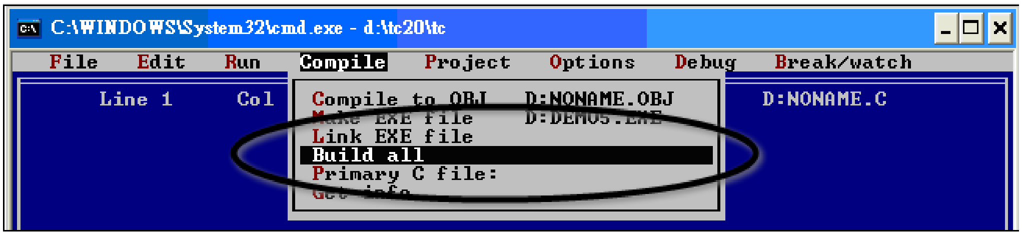


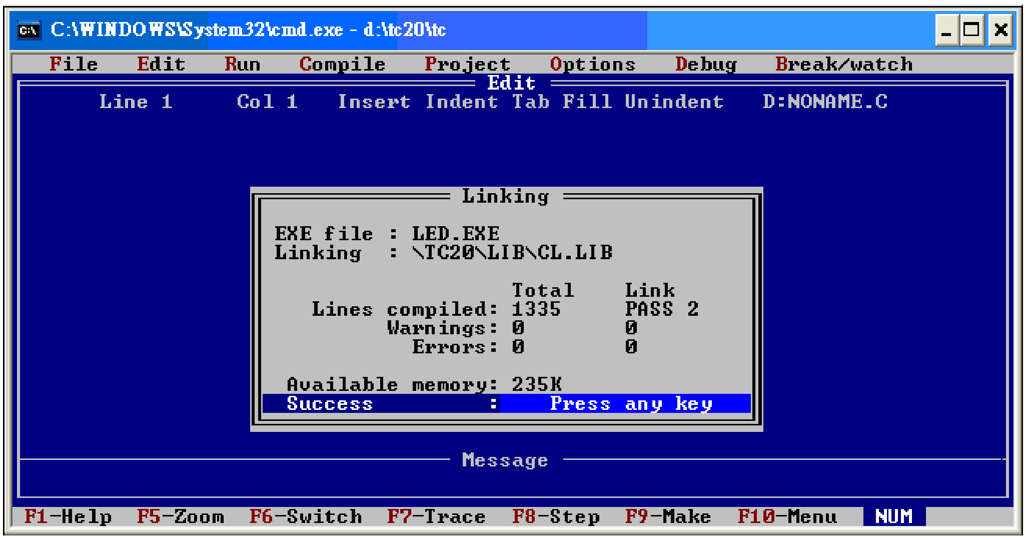
```
Line 1     Col 1    Insert Indent Ta

Model                Large
Defines
Code generation
Optimization
Source
Errors
Names
```



```
Line 1     Col 1    Insert Indent Ta

Model                Large
Defines
Code generation

Calling convention          C
Instruction set             80186/80286
Floating point              Emulation
Default char type           Signed
Alignment                   Byte
Generate underbars          On
Merge duplicate strings     Off
Standard stack frame        On
Test stack overflow         Off
Line numbers                Off
OBJ debug information       On
```

### Step 6: Building the project



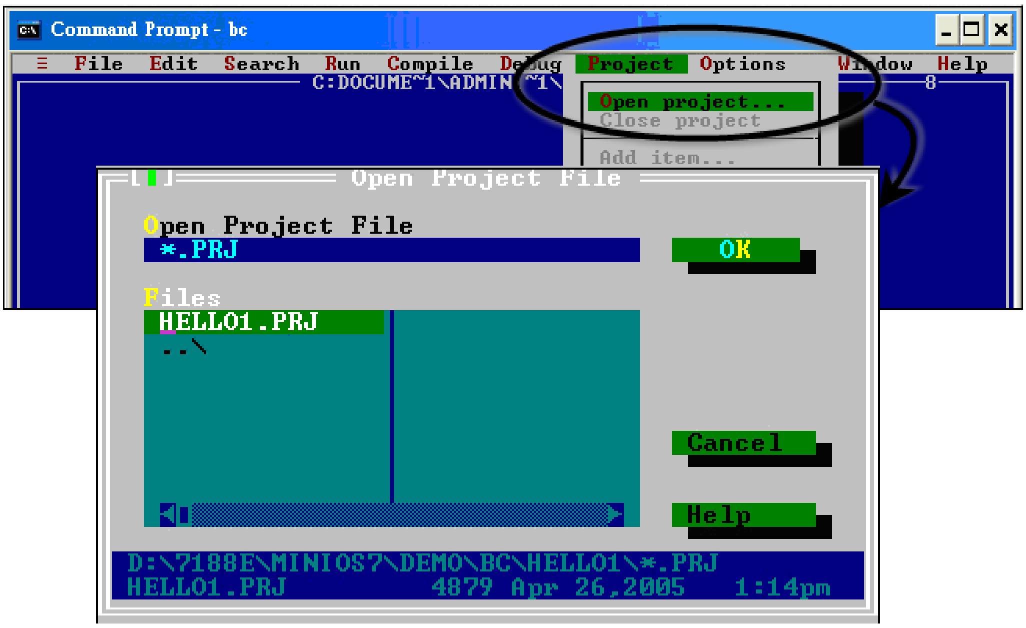


```
                 Linking
 EXE file  : LED.EXE
 Linking   : \TC20\LIB\CL.LIB

                   Total     Link
    Lines compiled: 1335     PASS 2
          Warnings: 0        0
            Errors: 0        0

  Available memory: 235K
  Success          :     Press any key
```

# D.2. BC++ 3.1. IDE

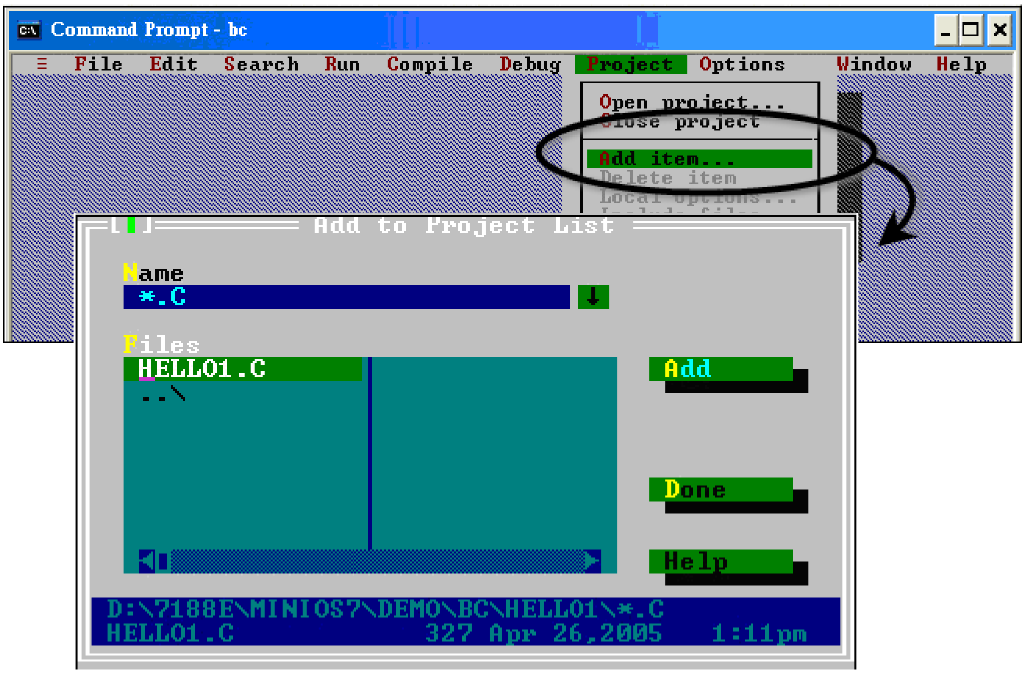
**Step 1: Executing the Borland C++ 3.1**

**Step 2: Creating a new project file (*.prj)**



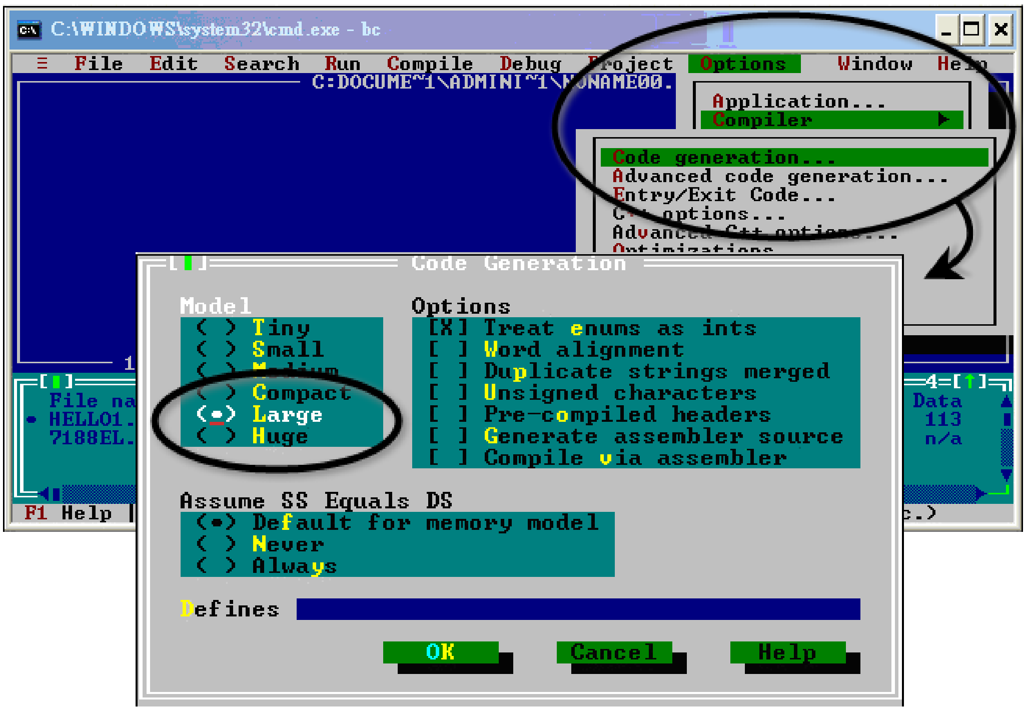
```
Open Project File
*.PRJ
Files
HELLO1.PRJ
..\
D:\7188E\MINIOS7\DEMO\BC\HELLO1\*.PRJ
HELLO1.PRJ            4879 Apr 26,2005    1:14pm
```

**Step 3: Add all the necessary files to the project**



```
Name
*.C
Files
HELLO1.C
..\
D:\7188E\MINIOS7\DEMO\BC\HELLO1\*.C
HELLO1.C                327 Apr 26,2005    1:11pm
```

### Step 4: Change the Memory model (Large for 8000e.lib)


C:\WINDOWS\system32\cmd.exe - bc
≡ File Edit Search Run Compile Debug Project Options Window Help
C:\DOCUME~1\ADMINI~1\NONAME00.
Application...
Compiler
Code generation...
Advanced code generation...
Entry/Exit Code...
C++ options...
Advanced C++ options...
Optimizations...
F1 Help

Code Generation

Model
( ) Tiny
( ) Small
( ) Medium
( ) Compact
(•) Large
( ) Huge

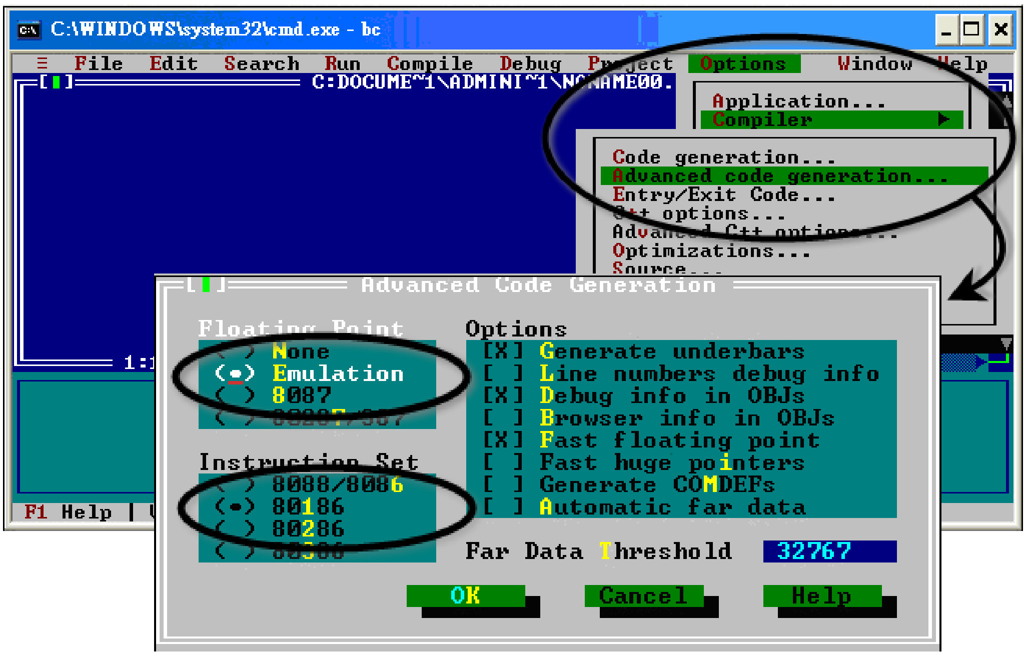
Options
[X] Treat enums as ints
[ ] Word alignment
[ ] Duplicate strings merged
[ ] Unsigned characters
[ ] Pre-compiled headers
[ ] Generate assembler source
[ ] Compile via assembler

Assume SS Equals DS
(•) Default for memory model
( ) Never
( ) Always

Defines

OK Cancel Help

### Step 5: Set the Advanced code generation options and Set the Floating Point to Emulation and the Instruction Set to 80186


C:\WINDOWS\system32\cmd.exe - bc
≡ File Edit Search Run Compile Debug Project Options Window Help
C:\DOCUME~1\ADMINI~1\NONAME00.
Application...
Compiler
Code generation...
Advanced code generation...
Entry/Exit Code...
C++ options...
Advanced C++ options...
Optimizations...
Source...
F1 Help

Advanced Code Generation

Floating Point
( ) None
(•) Emulation
( ) 8087
( ) 80287/387

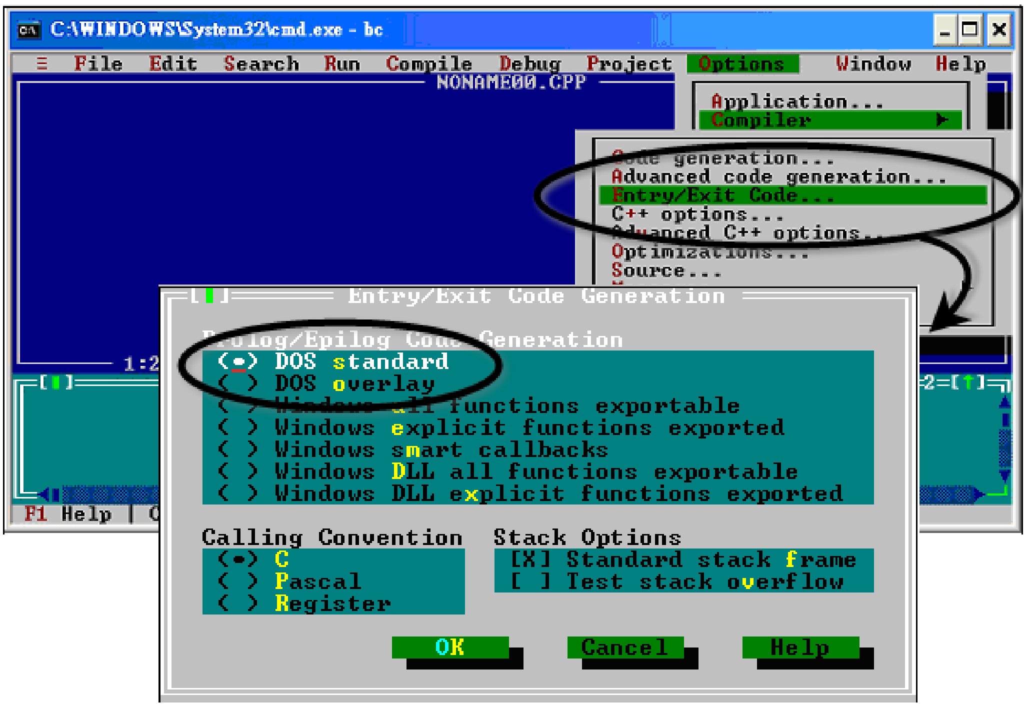
Instruction Set
( ) 8088/8086
(•) 80186
( ) 80286
( ) 80386

Options
[X] Generate underbars
[ ] Line numbers debug info
[X] Debug info in OBJs
[ ] Browser info in OBJs
[X] Fast floating point
[ ] Fast huge pointers
[ ] Generate COMDEFs
[ ] Automatic far data

Far Data Threshold 32767

OK Cancel Help

## Step 6: Set the Entry/Exit Code Generation option and setting the DOS standard



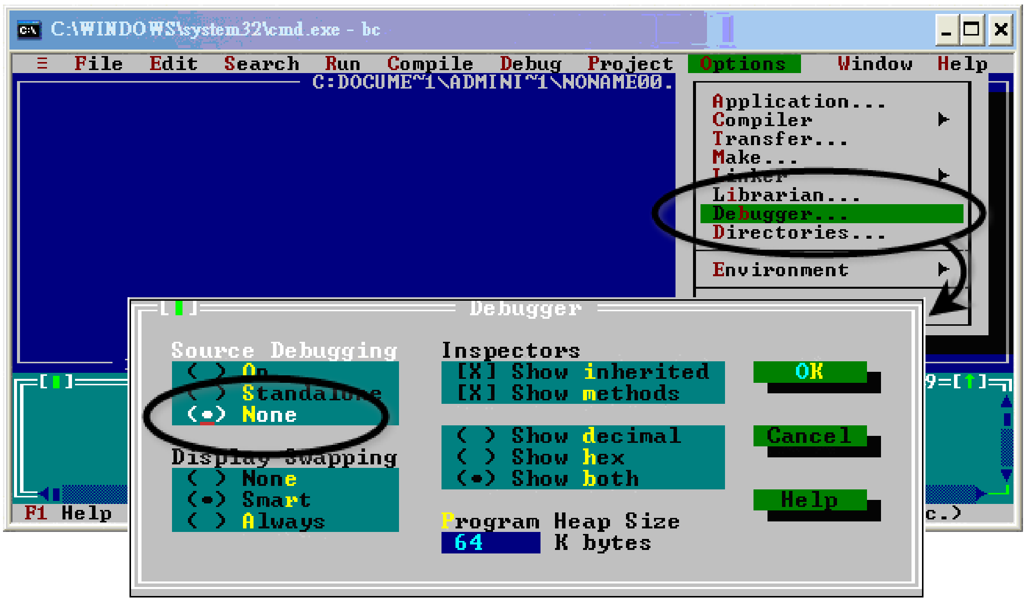
Entry/Exit Code Generation

Prolog/Epilog Code Generation
(•) DOS standard
( ) DOS overlay
( ) Windows all functions exportable
( ) Windows explicit functions exported
( ) Windows smart callbacks
( ) Windows DLL all functions exportable
( ) Windows DLL explicit functions exported

Calling Convention
(•) C
( ) Pascal
( ) Register

Stack Options
[X] Standard stack frame
[ ] Test stack overflow

## Step 7: Choosing the Debugger…and set the Source Debugging to None


C:\WINDOWS\system32\cmd.exe - bc
≡ File Edit Search Run Compile Debug Project Options Window Help
C:\DOCUME~1\ADMINI~1\NONAME00.
Application...
Compiler
Transfer...
Make...
Linker
Librarian...
Debugger...
Directories...
Environment
F1 Help

Debugger

Source Debugging
( ) On
( ) Standalone
(•) None

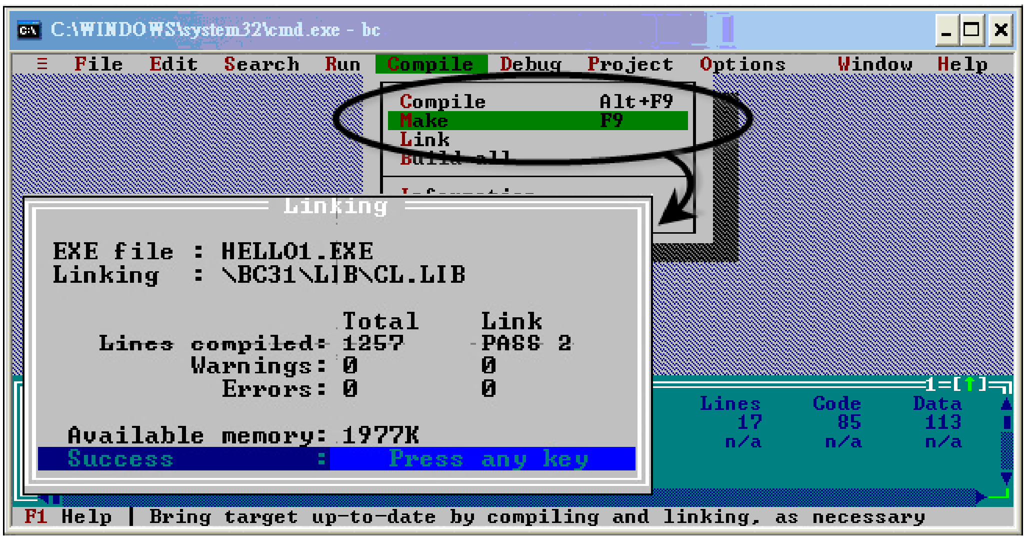
Display Swapping
( ) None
(•) Smart
( ) Always

Inspectors
[X] Show inherited
[X] Show methods

( ) Show decimal
( ) Show hex
(•) Show both

Program Heap Size
64 K bytes

### Step 8: Make the project



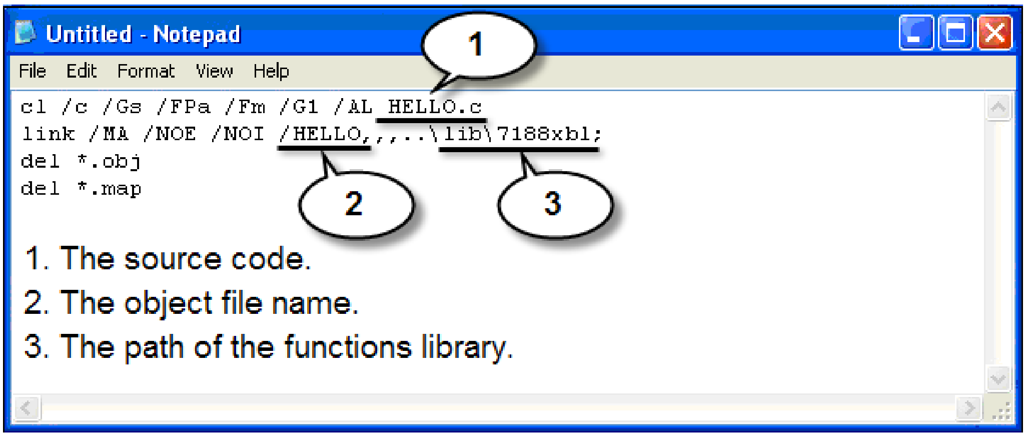
C:\WINDOWS\system32\cmd.exe - bc
≡ File Edit Search Run Compile Debug Project Options Window Help
Compile Alt+F9
Make F9
Link
Build all
Information...

```
              Linking
EXE file  : HELLO1.EXE
Linking   : \BC31\LIB\CL.LIB

                   Total     Link
   Lines compiled: 1257      PASS 2
         Warnings: 0         0
           Errors: 0         0

 Available memory: 1977K
 Success         :     Press any key
```

Lines Code Data
17 85 113
n/a n/a n/a
F1 Help | Bring target up-to-date by compiling and linking, as necessary

# D.3. MSC 6.00

**Step 1: In the source file folder, create a batch file called Gomsc.bat using the text editor**



```
cl /c /Gs /FPa /Fm /G1 /AL HELLO.c
link /MA /NOE /NOI /HELLO,,,..\lib\7188xbl;
del *.obj
del *.map
```

1. The source code.
2. The object file name.
3. The path of the functions library.

## Tip & Warnings

/C: Don't strip comments

/GS: No stack checking

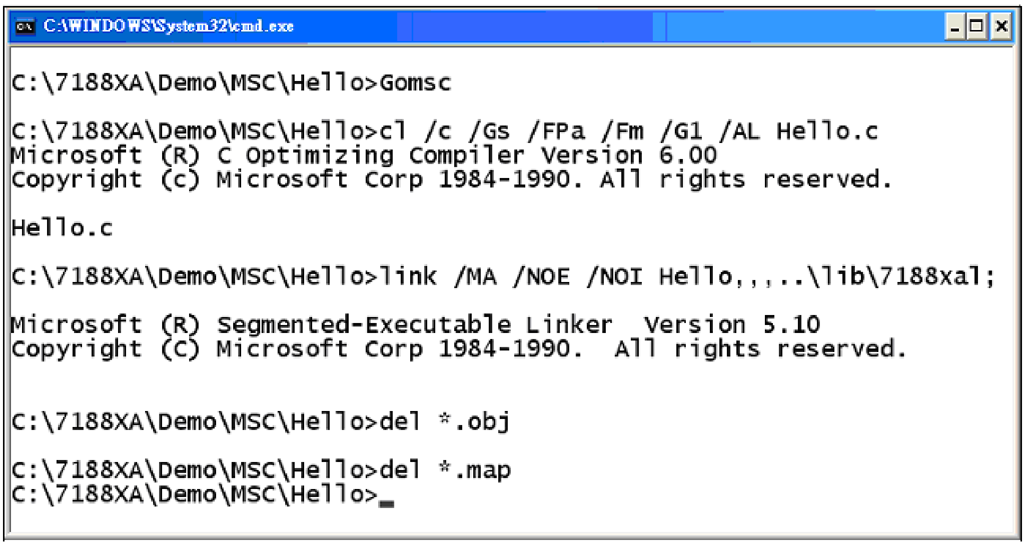
/Fpa: Calls with altmath

/Fm: [map file]

/G1: 186 instructions

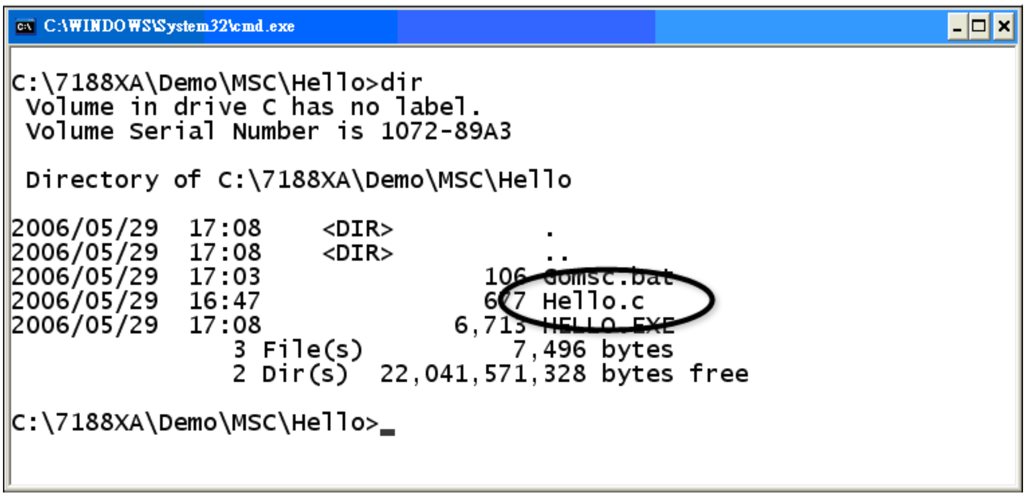
/AL: Large model

### Step 2: Run the Gomsc.bat file



```
C:\7188XA\Demo\MSC\Hello>Gomsc

C:\7188XA\Demo\MSC\Hello>cl /c /Gs /FPa /Fm /G1 /AL Hello.c
Microsoft (R) C Optimizing Compiler Version 6.00
Copyright (c) Microsoft Corp 1984-1990. All rights reserved.

Hello.c

C:\7188XA\Demo\MSC\Hello>link /MA /NOE /NOI Hello,,,..\lib\7188xal;

Microsoft (R) Segmented-Executable Linker  Version 5.10
Copyright (C) Microsoft Corp 1984-1990.  All rights reserved.


C:\7188XA\Demo\MSC\Hello>del *.obj

C:\7188XA\Demo\MSC\Hello>del *.map
C:\7188XA\Demo\MSC\Hello>_
```

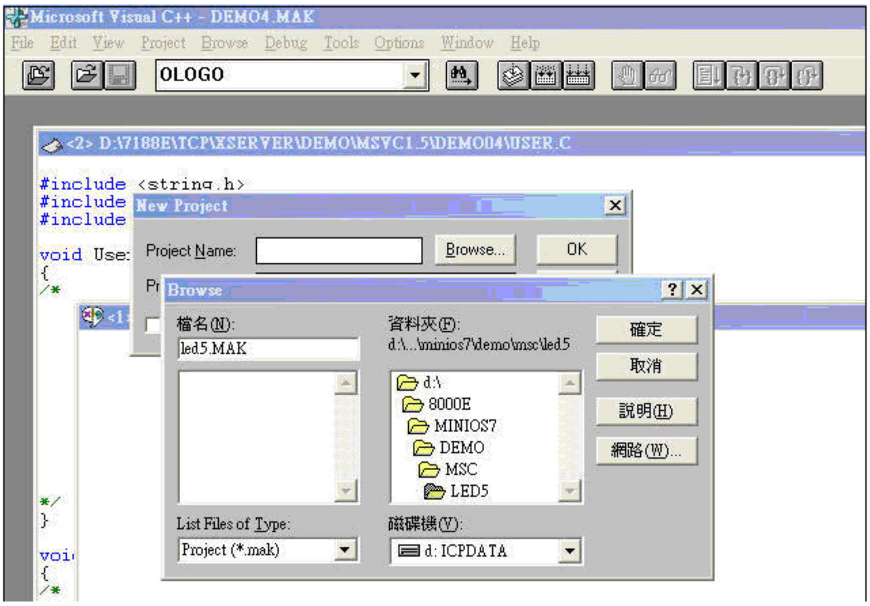
### Step 3: A new executable file will be created if it is successfully compiled



```
C:\7188XA\Demo\MSC\Hello>dir
 Volume in drive C has no label.
 Volume Serial Number is 1072-89A3

 Directory of C:\7188XA\Demo\MSC\Hello

2006/05/29  17:08    <DIR>          .
2006/05/29  17:08    <DIR>          ..
2006/05/29  17:03               106 Gomsc.bat
2006/05/29  16:47               677 Hello.c
2006/05/29  17:08             6,713 HELLO.EXE
               3 File(s)          7,496 bytes
               2 Dir(s)  22,041,571,328 bytes free

C:\7188XA\Demo\MSC\Hello>_
```

# D.4. MSVC 1.50

**Step 1: Run MSVC.exe**


Microsoft Visual C++ - DEMO4.MAK
File Edit View Project Browse Debug Tools Options Window Help
OLOGO
<2> D:\7188E\TCP\XSERVER\DEMO\MSVC1.5\DEMO04\USER.C
New Project
Project Name:
Browse...
OK
Browse
確定
取消
說明(H)
網路(W)...
d:\
8000E
MINIOS7
DEMO
MSC
LED5
List Files of Type:
磁碟機(V):

```
#include <string.h>
#include
#include

void Use:
{
/*
```

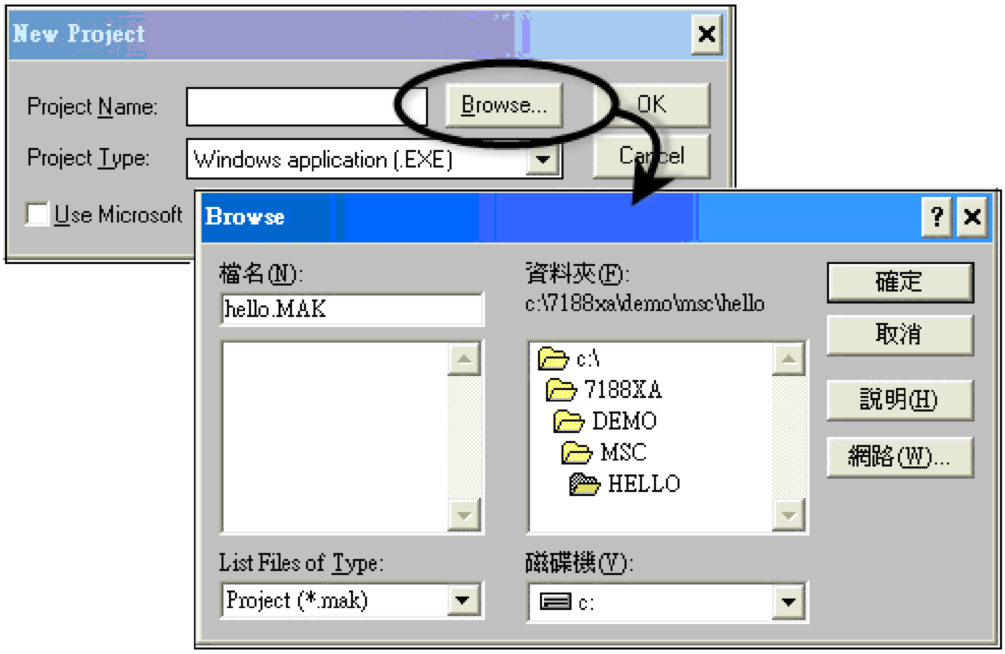
檔名(N): led5.MAK

資料夾(F): d:\...\minios7\demo\msc\led5

List Files of Type: Project (*.mak)

磁碟機(V): d: ICPDATA

**Step 2: Create a new project (*.mak) by entering the name of the project in the Project Name field and then select MS-DOS application (EXE) as the Project type**



Project Type: Windows application (.EXE)

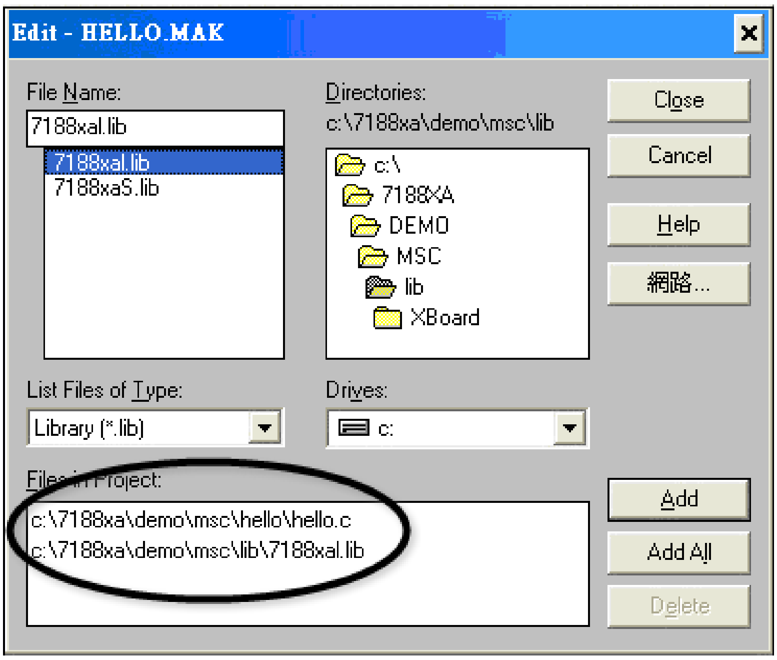
檔名(N): hello.MAK

資料夾(F): c:\7188xa\demo\msc\hello

List Files of Type: Project (*.mak)

磁碟機(V): c:

**Step 3: Add the user's program and the necessary library files to the project**



File Name: 7188xal.lib

7188xal.lib
7188xaS.lib

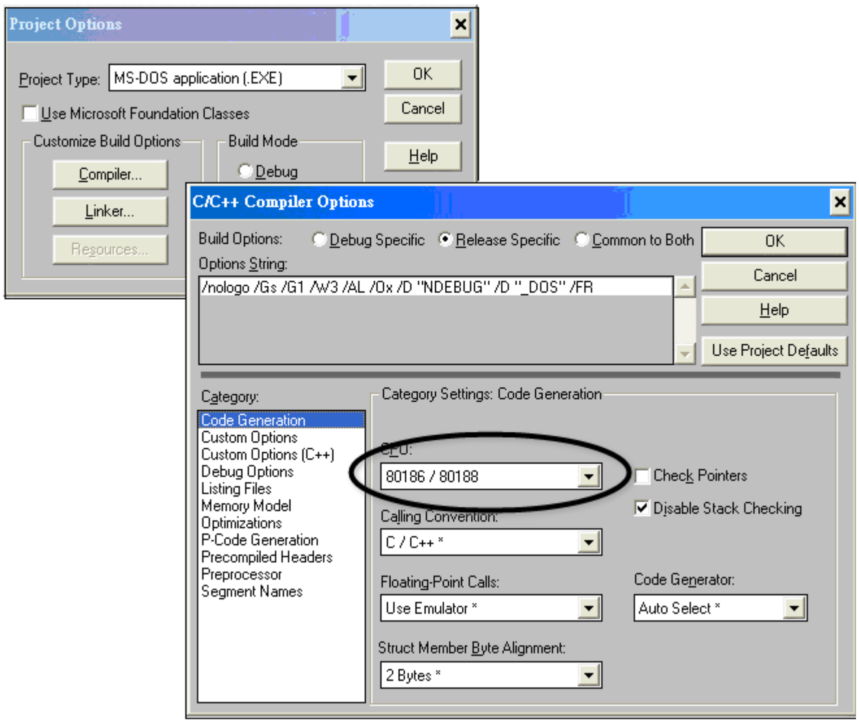
Directories: c:\7188xa\demo\msc\lib

List Files of Type: Library (*.lib)

Drives: c:

Files in Project:
c:\7188xa\demo\msc\hello\hello.c
c:\7188xa\demo\msc\lib\7188xal.lib

**Step 4: Set the Code Generation on the Compiler.**



Project Type: MS-DOS application (.EXE)

□ Use Microsoft Foundation Classes

○ Debug

Build Options: ○ Debug Specific ● Release Specific ○ Common to Both

Options String:
/nologo /Gs /G1 /W3 /AL /Ox /D "NDEBUG" /D "_DOS" /FR

Category Settings: Code Generation

CPU: 80186 / 80188

□ Check Pointers

☑ Disable Stack Checking

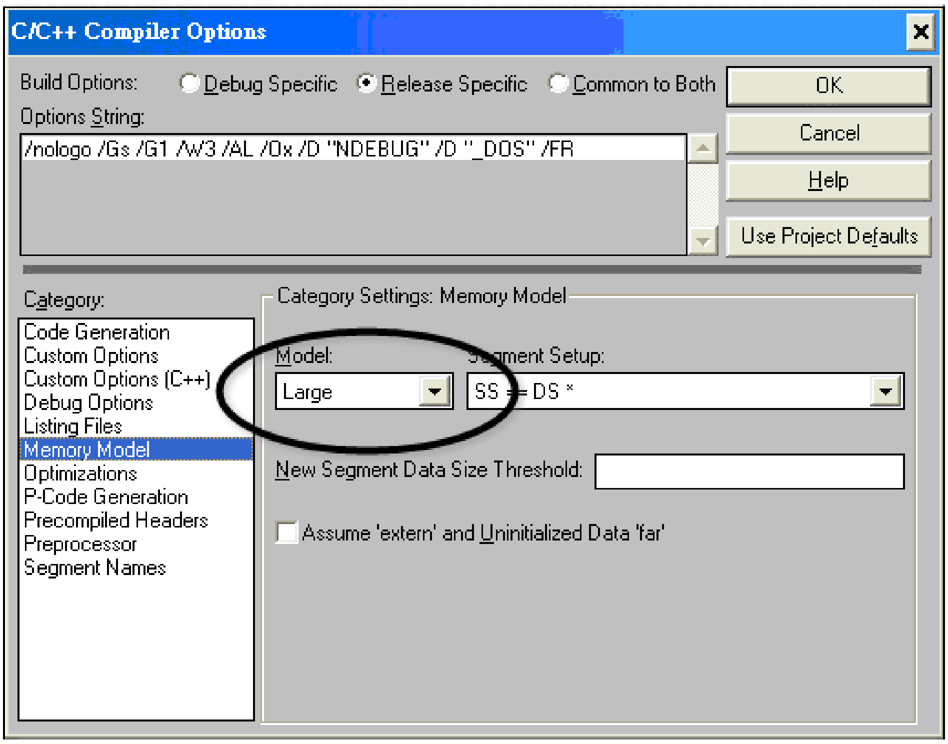
Calling Convention: C / C++ *

Floating-Point Calls: Use Emulator *

Code Generator: Auto Select *

Struct Member Byte Alignment: 2 Bytes *

### Step 5: Change the Memory model (large for 8000e.lib)



C/C++ Compiler Options

Build Options: ○ Debug Specific ● Release Specific ○ Common to Both

Options String:

```
/nologo /Gs /G1 /W3 /AL /Ox /D "NDEBUG" /D "_DOS" /FR
```

Category: Memory Model

Category Settings: Memory Model

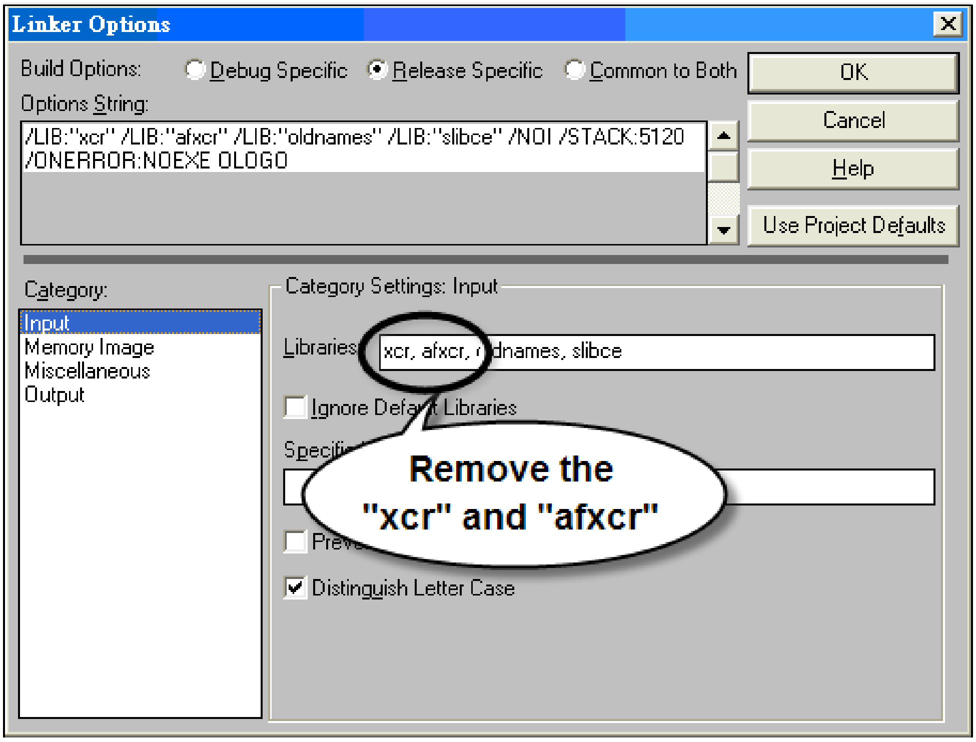
Model: Large

Segment Setup: SS == DS *

New Segment Data Size Threshold:

□ Assume 'extern' and Uninitialized Data 'far'

### Step 6: Remove the xcr, afxcr library from the Input Category



Linker Options

Build Options: ○ Debug Specific ● Release Specific ○ Common to Both

Options String:

```
/LIB:"xcr" /LIB:"afxcr" /LIB:"oldnames" /LIB:"slibce" /NOI /STACK:5120
/ONERROR:NOEXE OLOGO
```

Category: Input

Category Settings: Input

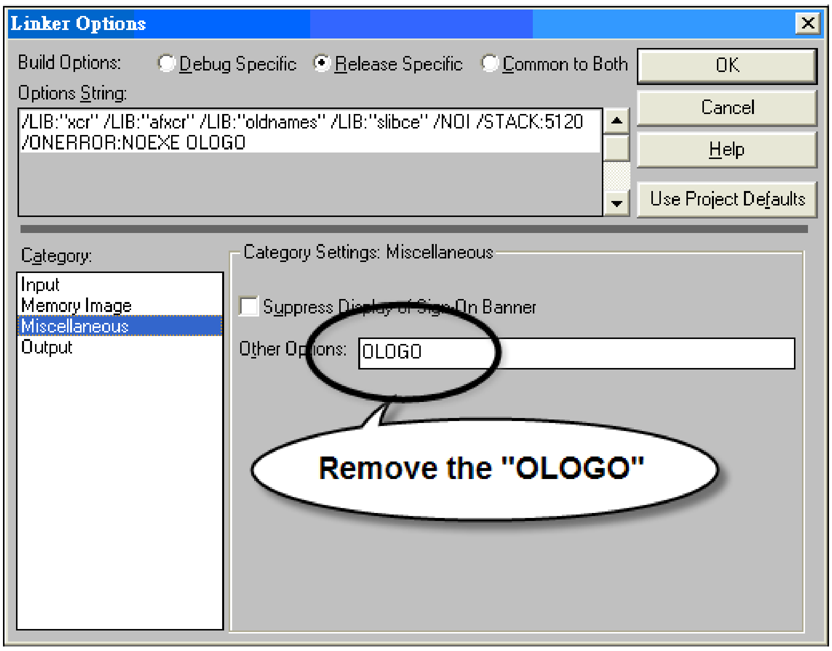
Libraries: xcr, afxcr, oldnames, slibce

□ Ignore Default Libraries

☑ Distinguish Letter Case

Remove the "xcr" and "afxcr"

### Step 7: Remove the OLOGO option from the miscellancous Category.



Build Options: ○ Debug Specific ● Release Specific ○ Common to Both

Options String:

```
/LIB:"xcr" /LIB:"afxcr" /LIB:"oldnames" /LIB:"slibce" /NOI /STACK:5120
/ONERROR:NOEXE OLOGO
```

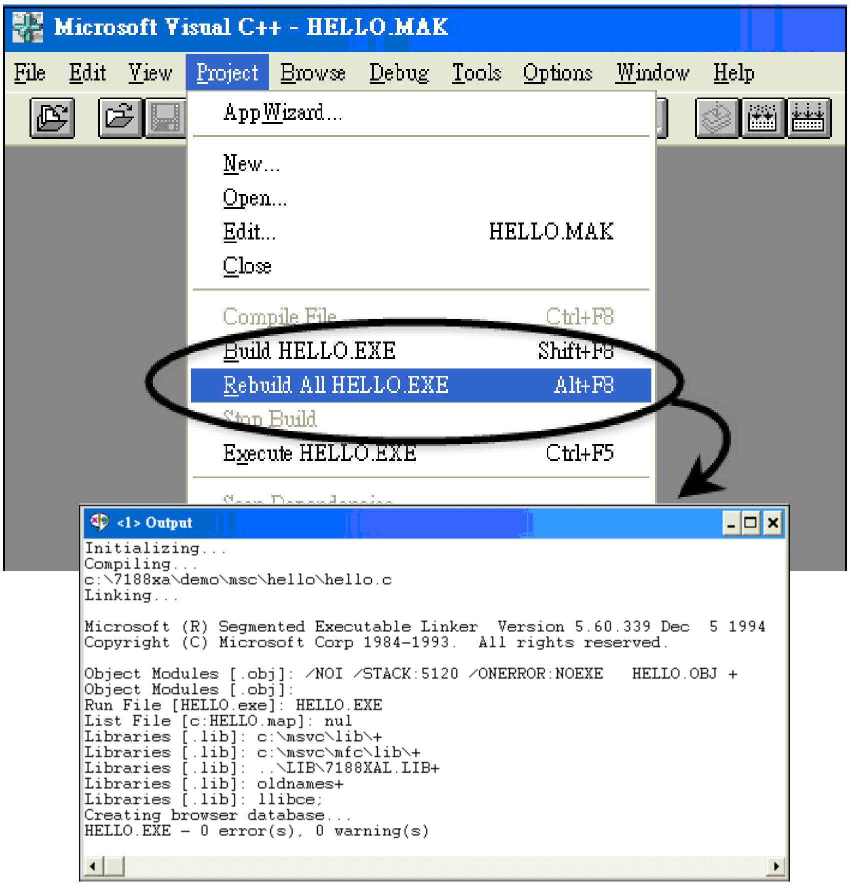
Category: Input, Memory Image, Miscellaneous, Output

Category Settings: Miscellaneous

☐ Suppress Display of Sign-On Banner

Other Options: OLOGO

### Step 8: Rebuild the project



```
Initializing...
Compiling...
c:\7188xa\demo\msc\hello\hello.c
Linking...

Microsoft (R) Segmented Executable Linker  Version 5.60.339 Dec  5 1994
Copyright (C) Microsoft Corp 1984-1993.  All rights reserved.

Object Modules [.obj]: /NOI /STACK:5120 /ONERROR:NOEXE   HELLO.OBJ +
Object Modules [.obj]:
Run File [HELLO.exe]: HELLO.EXE
List File [c:HELLO.map]: nul
Libraries [.lib]: c:\msvc\lib\+
Libraries [.lib]: c:\msvc\mfc\lib\+
Libraries [.lib]: ..\LIB\7188XAL.LIB+
Libraries [.lib]: oldnames+
Libraries [.lib]: llibce;
Creating browser database...
HELLO.EXE - 0 error(s), 0 warning(s)
```