iPAC-8000 Series

User Manual

(C Language Based PAC)

Version 1.0.4, December 2014

Service and usage information for

iP-8411

iP-8411

iP-8441/iP-8441-FD

iP-8841/iP-8841-FD

Written by Martin Hsu

Edited by Anna Huang

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 question, please feel free to contact us.

We will give you a quick response within 2 workdays.

Email: service@icpdas.com

Table of Contents

1. Introduction

The iPAC-8000 is serial/Ethernet PAC that provides a wide variety of features and capability in an extremely compact package.

designed for data acquisition, remote measurement, and control applications. It supports various connectivity including Dual 10/100 Base-TX Ethernet ports, one RS-232/485 port, one RS-485 port and two RS-232 ports, and 4/8 slots for high performance Parallel I/O modules (high profile I-8K series) and Serial I/O modules (high profile I-87K series), etc.

The iPAC-8000 is designed for industrial monitoring, measurement and controlling. It has redundant power inputs with 1 kV isolation from noise and surges, and a wide range of operating temperature (-25 °C ~ +75 °C). It can work in the harsh and rough environment.

According to the communication property, the iPAC-8000 can be divided into the following types:

- **Serial series PAC**
  iP-8411/iP-8811
- **Ethernet series PAC**
  iP-8441/iP-8841
  iP-8441-FD/iP-8841-FD

The table below lists the features and compares the differences for each iPAC-8000 unit.

<table>
<tr><th></th><th>I/O Slots</th><th>CPU</th><th>SRAM</th><th>Flash</th><th>Memory Expansion</th><th>Ethernet</th><th>RS-232/ RS-485</th></tr>
<tr><td>iP-8411</td><td>4</td><td rowspan="6">80 MHz</td><td rowspan="2">512 KB</td><td rowspan="6">512 KB</td><td rowspan="4">micro SD</td><td rowspan="2">-</td><td rowspan="2">4</td></tr>
<tr><td>iP-8811</td><td>8</td></tr>
<tr><td>iP-8441</td><td>4</td><td rowspan="4">768 KB</td><td rowspan="4">2
10-Base-T</td><td rowspan="4">3</td></tr>
<tr><td>iP-8841</td><td>8</td></tr>
<tr><td>iP-8441-FD</td><td>4</td><td rowspan="2">micro SD + 256 MB NAND Flash</td></tr>
<tr><td>iP-8841-FD</td><td>8</td></tr>
</table>

1.1. Features

iPAC-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.

Software Features

➢ MiniOS7 embedded operating system

MiniOS7 was introduced in 1996 as an MS-DOS like operating system for embedded controller developers. The features of MiniOS7 include

A. Small kernel size (64KB)
B. Fast boot speed (0.4~0.8 second)
C. Hardware diagnostic functions
D. Simple command line operation over RS-232 or Ethernet
E. Load files via RS-232 or Ethernet

➢ VxComm Technique Supported

VxComm technique is used to create virtual COM ports on PC (for windows 2K/XP) to map remote COM ports of PDS-700, I-7188E, I-8000 and iPAC-8000 over the Ethernet. Using the technique, RS-232/485 software can access devices locally (via the physical RS-232/485 bus) or remotely (via the Ethernet). The RS-232/485 software only needs to change COM port number from the physical COM port to virtual COM port.

➢ Easy-Use Software Development Template (Xserver) for TCP/IP Application

To simplify the TCP/IP software developing process, we designed a software develop template, called XServer. 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 functionalities and greatly shorten the developing time.

➢ Redundant Ethernet Communication (for Ethernet series PAC only)

With the dual LAN features of iPAC-8000, user's software on PCs or other controllers can implement redundant Ethernet communication. With VxComm technique, the redundant Ethernet communication is ready. One virtual COM port on PC can map to one COM port of iPAC-8000 via two IP address. When the communication is failed (or timeout), the VxComm driver can automatically and quickly switch the virtual COM port mapping to another IP address to keep the communication.

➢ Slave I/O firmware options (for DCON or Modbus/TCP protocol)

In some simple Ethernet I/O applications, users just want to know how to send a command to the I/O to get back a response. They don’t want to develop a firmware. That is too difficult to them. Thus, we also provide two firmware for this purpose.

A. DCON firmware

DCON firmware supports an ASCII string based command set, called DCON protocol

B. Modbus firmware

Modbus firmware supports the standard Modbus/TCP protocol. SCADA software can easily access the I/O module plugged in the iPAC-8000.

➢ 80186 CPU (16bit and 80MHz) with 512KB Flash and 768KB SRAM

The 512KB flash is for storing files, and the 768KB SRAM is for running programs.

➢ 64-bit Hardware Serial Number

The 64-bit hardware serial number is unique and individual. Every serial number of iPAC-8000 is different. Users can add a checking mechanism to their AP to prevent software from pirating.

➢ Dual Battery Backup SRAM (512KB)

To maintain important data while power off, non-volatile memory is the ideal design. The iPAC-8000 equips a 512KB SRAM with two Li-batteries to maintain data while power off. The two Li-batteries can continually supply power to the 512KB SRAM to retain the data for 5 years; and the dual-battery design can avoid data lost while replacing a new battery.

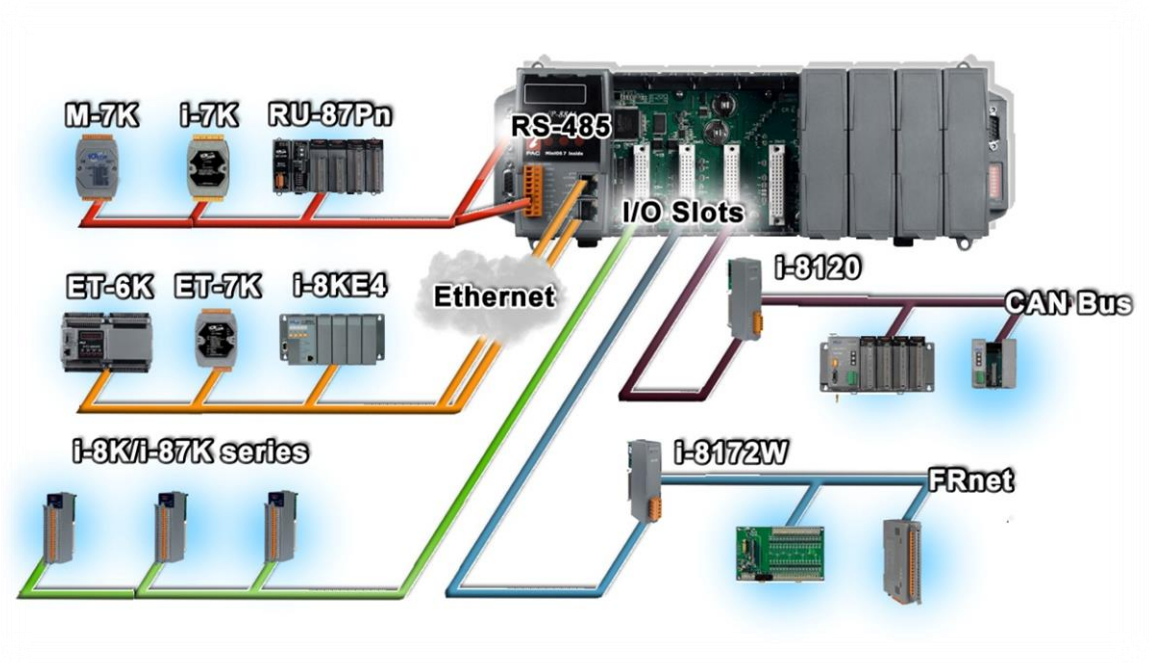
➢ Dual Ethernet Ports (for Ethernet series PAC only)

iPAC-8000 provides two Ethernet ports. The two Ethernet ports can be used to implement redundant Ethernet communication and separate Ethernet communication (one for global Internet, one for private Ethernet).

➢ Redundant Power Inputs

To prevent theiPAC-8000 from failing by the power loss, the power module is designed with two inputs. The iPAC-8000 can keep working even one power input fails, and mean while there is a relay output for informing the power failure.

➢ **Rich I/O Expansion Ability (RS-232/485, Ethernet, FRnet, CAN)**



Beside the local I/O slots, iPAC-8000 also equips several RS-232/485 ports, two Ethernet ports to connect serial I/O and Ethernet I/O. And with FRnet and CAN communication module in local slot, FRnet I/O and CAN devices are easy to be integrated.

➢ **Ventilated Housing Design Allows Operation Between -25 ~ +75 ℃**

Each iPAC-8000 is housed in a plastic-based box with a column-like ventilator that can help to cool the working environment inside the box and allow the iPAC-8000 operating between -25 °C and +75 °C.

1.2. Specification

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

Specification

Models	I-8411	I-8811	I-8431	I-8831	I-8431-80	I-8431-80
CPU Module						
CPU	80188 or compatible (8-bit and 40 MHz)				80186 or compatible (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						
Digit 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					
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

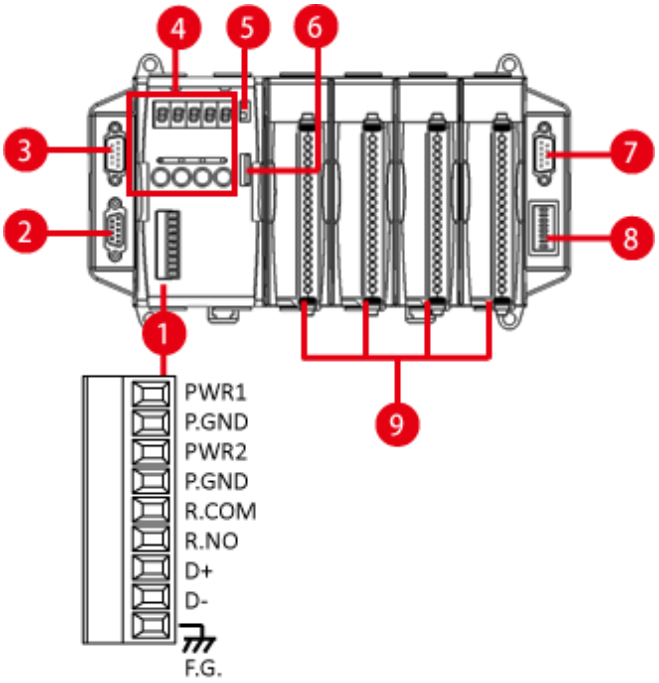
Models	
DP-660	AC 85 ~ 270 V input, DC 24 V/2.5 A and 5 V/0.5 A output power supply
DP-665	AC 85 ~ 270 V input, DC 24 V/1.7 A and 5 V/0.5 A output power supply
DP-1200	AC 85 ~ 270 V input, DC 24 V/5.0 A output power supply
MDR-60-24	AC 85 ~ 264 V input, DC 24 V/2.5 A Output Industrial DIN Rail Power Supply
I-7560 CR	USB to RS-232 Converter (RoHS)
3LMSD-2000	2 GB microSD card

1.3. Overview

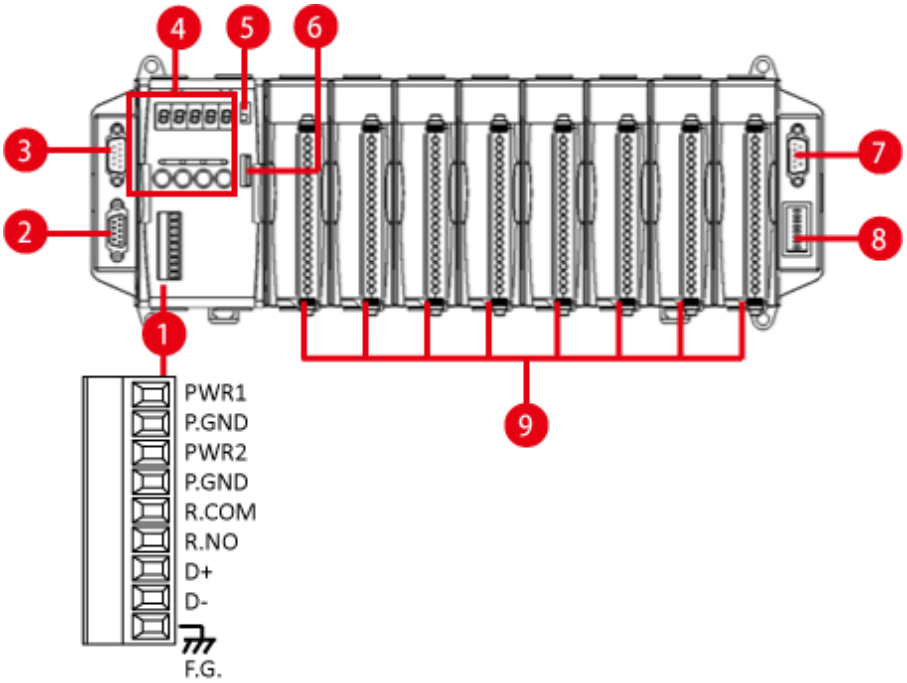
iPAC-8000 consists of several different components that integrate with ICP DAS system. Here is an overview of the components and its descriptions. The following list shows the details of the components:

- **Serial series PAC**

iP-8411



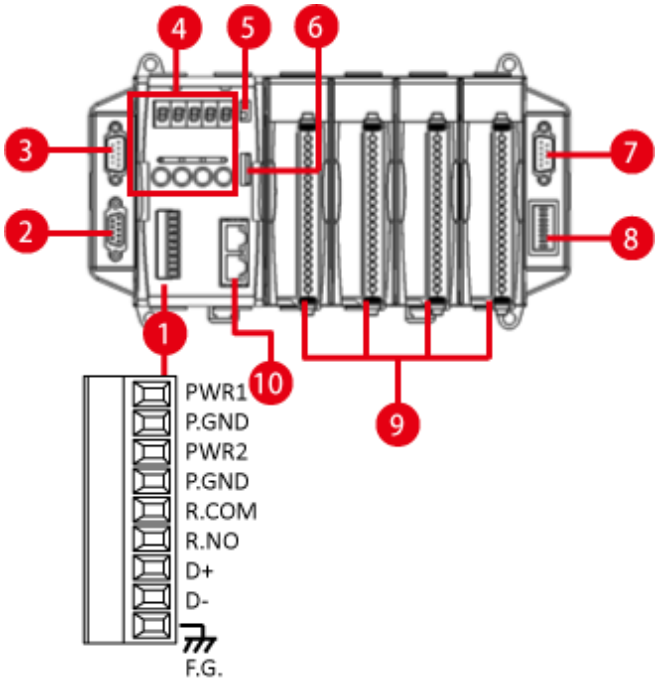
iP-8811



➢ Ethernet series PAC

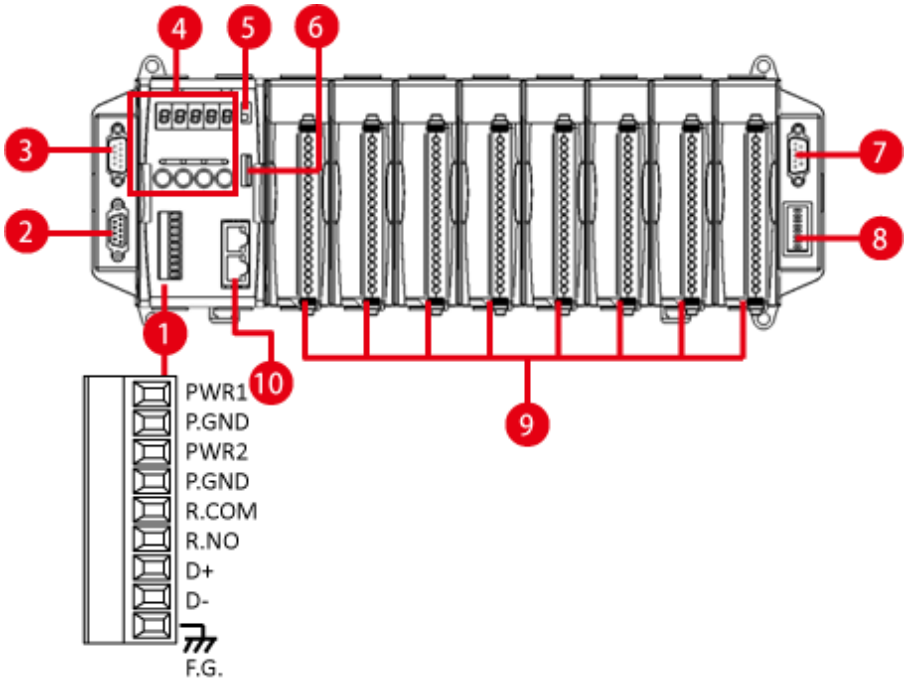
iP-8441

iP-8441-FD



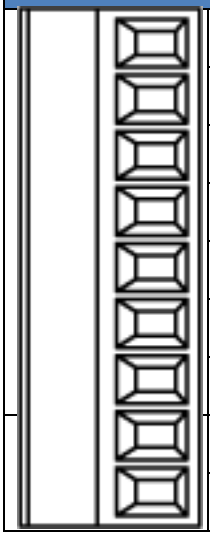
iP-8841

iP-8841-FD



1. Terminal Block

The iPAC-8000 has a 9-pins terminal block. The table below describes the terminal block designations and its functions.

Screw Terminal		Signal	Description
	1	PWR1	Power Input 1
	2	P.GND	
	3	PWR2	Power Input 2
	4	P.GND	
	5	R.COM	Relay Output
	6	R.NO	
	7	D+	COM2: RS-485
	8	D-	
	9	F.G.	Frame Ground

COM2 (2-Pins RS-485)

Baud Rate: 115200, 57600, 38400, 19200, 9600, 4800 bps

Data Bits: 7, 8

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

Stop Bits: 1, 2

FIFO: 16 bytes

2. COM1 (RS-232)

Port Type: Female

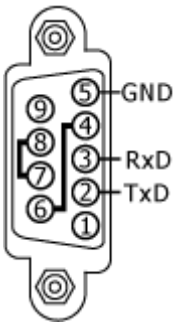
Baud Rate: 115200, 57600, 38400, 19200, 9600, 4800, 2400, 1200 bps

Data Bits: 7, 8

Parity: None, Even, Odd

Stop Bits: 1

FIFO: 1 byte



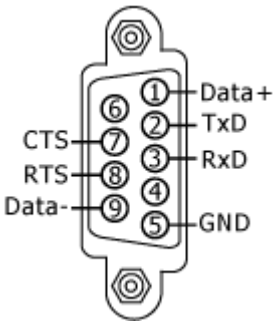
3. COM3 (RS-232/RS-485)

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; 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.

4. SMMI (Small Main-Machine Interface)

SMMI is a small control panel with 5-digital 7-Segment LED display, 4 programmable LED indicators, and 4 function buttons that allows you to quickly monitor the status of iPAC-8000.

5. DIP Switch

The DIP Switch is an operating mode selector which provides functions to configure modes of operation.

For more information about these modes of operation, see section 2.3., “Configuring the Boot Mode.”

6. microSD Expansion Slot

The microSD expansion slot is an interface used to access and download information on a microSD card to iPAC-8000. The microSD card can be used to increase memory capacity to 16 GB.

7. COM4 (RS-232)

Port Type: Male

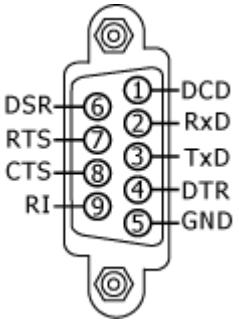
Baud Rate: 115200, 57600, 38400, 19200,
9600, 4800, 2400, 1200 bps

Data Bits: 5, 6, 7, 8

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

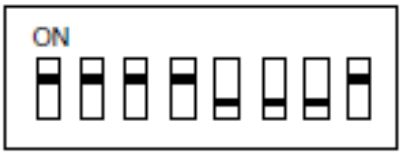
Stop Bits: 1, 2

FIFO: 16 bytes



8. 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.

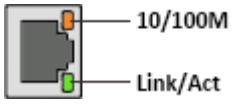


9. Expansion I/O Slots

iPAC-8000 is equipped with 4/8 expansion I/O slots, which allows it to communicate with external I/O devices.
For more information about the I/O expansion modules that are compatible with iPAC-8000, see
http://www.icpdas.com/root/product/solutions/remote_io/rs-485/i-8k_i-87k/i-8k_i-87k_selection.html

10. Ethernet Port (for Ethernet series PAC 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.

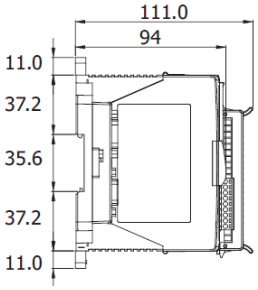
LED Indicator	State (Color)	Meaning
10/100M	ON (Orange)	Network Speed: 100 MB
	OFF	Network Speed: 10 MB
Link/Act	ON (Green)	The Link is active
	OFF	The Link is inactive
	Blinking(Green)	Network activity

1.4. Dimension

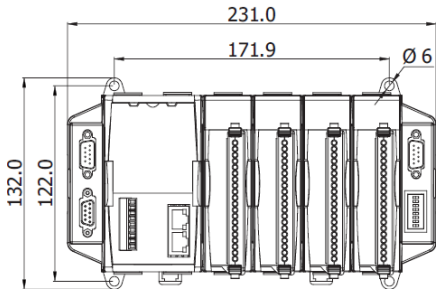
The diagrams below provide the dimensions of iPAC-8000 to use in defining your enclosure specifications. Remember to leave room for potential expansion if you are using other components in your system.

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

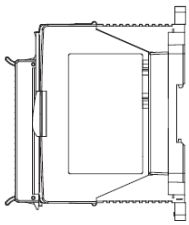
1.4.1. iP-8411/iP-8441/iP-8441-FD



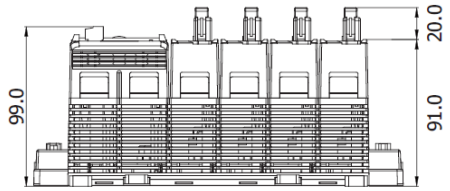
Left Side View



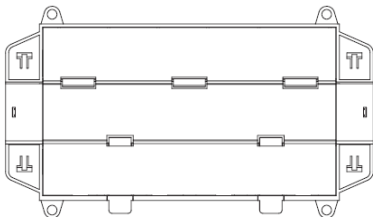
Front View



Right Side View

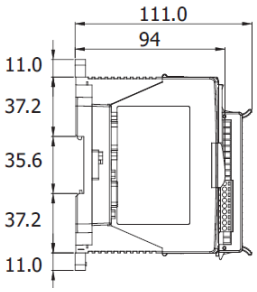


Bottom View

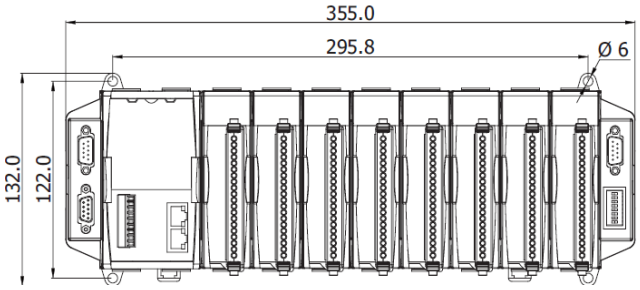


Back View

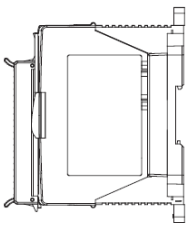
1.4.2. iP-8811/iP-8841/iP-8841-FD



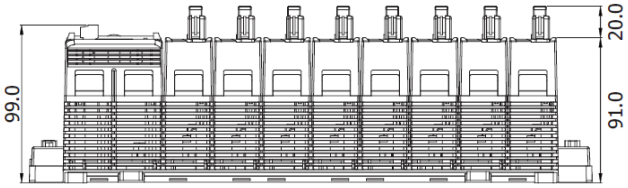
Left Side View



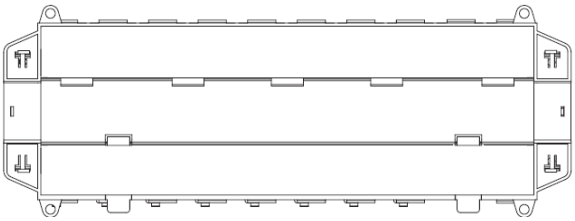
Front View



Right Side View



Bottom View



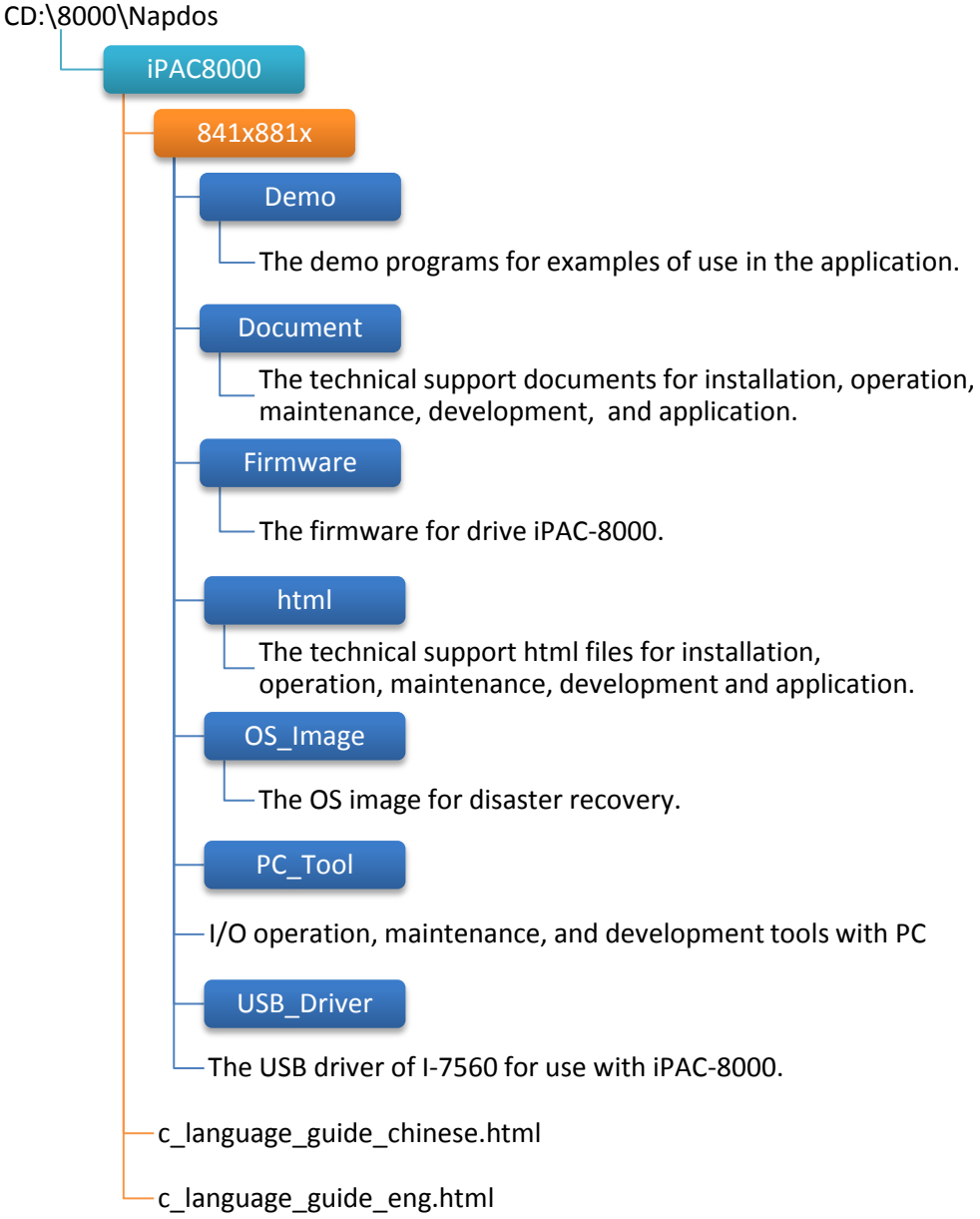
Back View

1.5. Companion CD

This package comes with a CD that provides drivers, software utility, and all of the required documentations…, etc., and the latest version of the contents of this CD that was provided by downloading from ICP DAS web site.

http://ftp.icpdas.com/pub/cd/8000cd/napdos/

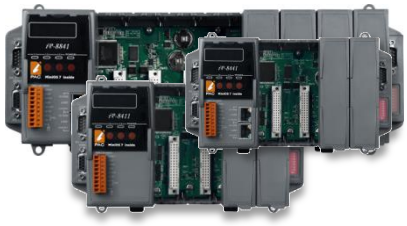
CD:\8000\Napdos



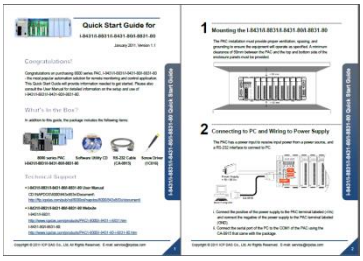
2. Getting Started

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 iPAC-8000.

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



iP-8411/iP-8811
iP-8441/iP-8841
iP-8841-FD/iP-8841-FD



Quick Start Guide

Software Utility CD

RS-232 Cable
(CA-0915)

Screw Driver
(1C016)

2.1. Mounting the Hardware

Before you work with iPAC-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.”

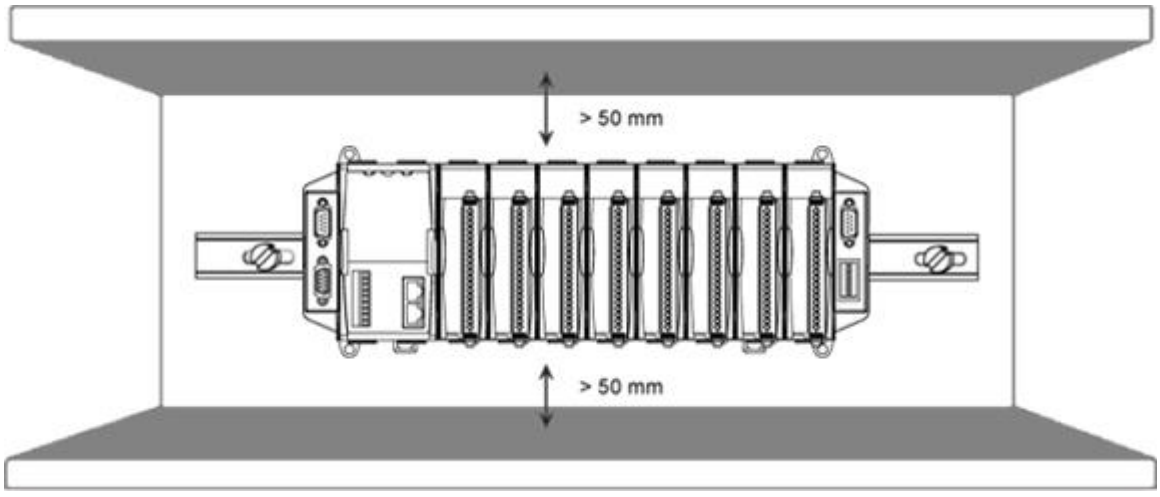
You also need to know the expansion capacities in order to choose the best expansion module for achieving maximal efficiency.

For more information about expansion modules that are compatible with the iPAC-8000, please refer to

http://www.icpdas.com/root/product/solutions/remote_io/rs-485/i-8k_i-87k/i-8k_i-87k_selection.html

2.1.1. Mounting the iPAC-8000

iPAC-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 clearance of 50mm between the iPAC-8000 and the top and bottom side of the enclosure panels.



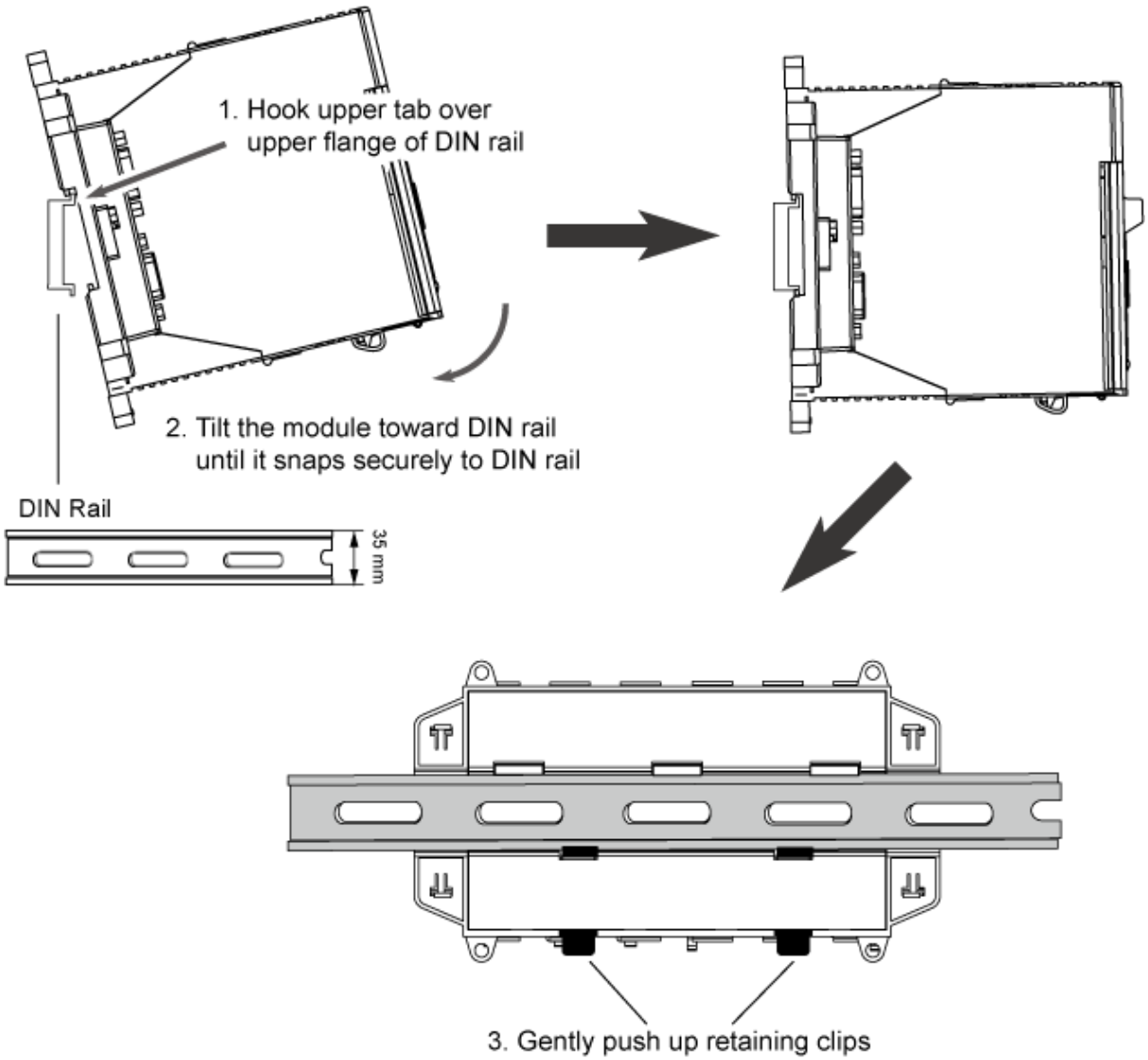
There are two ways for mounting the I-8000.

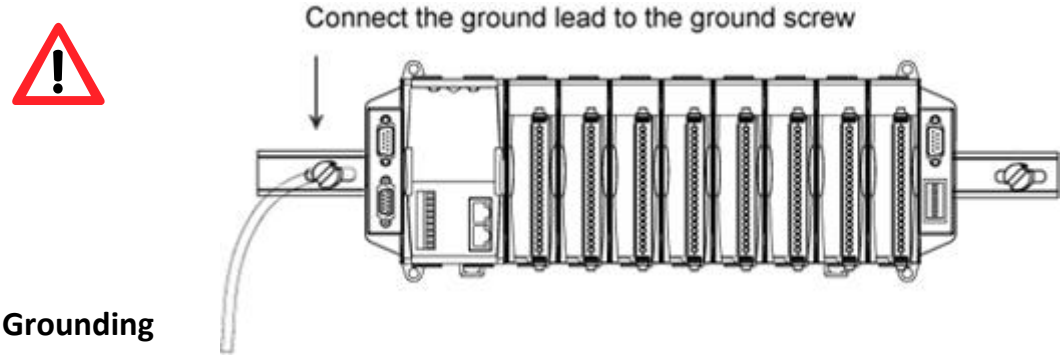
1. To mount the iPAC-8000 on a DIN rail

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





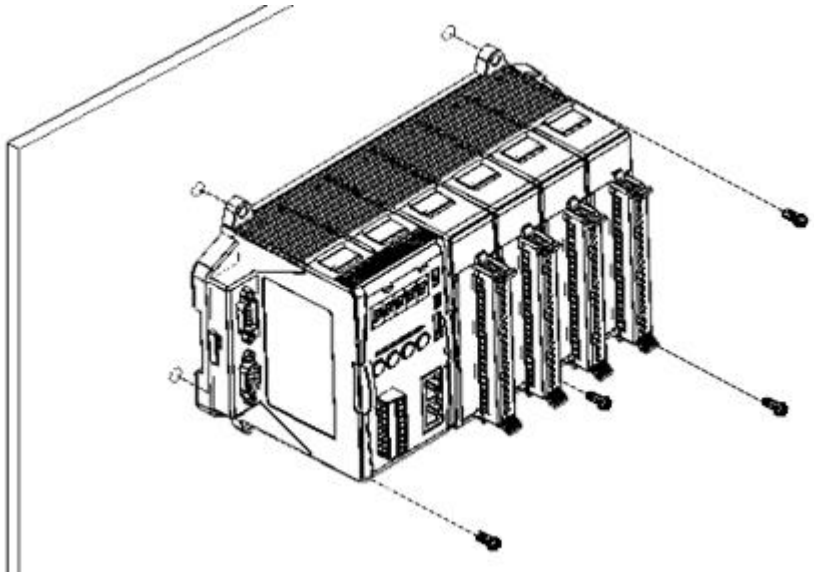
Grounding

A good common ground reference (earth ground) is essential for proper operation of the XP-8000-Atom. 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 iPAC-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 iPAC-8000

The package includes a RS-232 cable (CA-0915) for connecting the iPAC-8000 to a PC/Laptop. The iPAC-8000 has the power supply interface for supplying power from the power supply.

Step 1: Connect to the power supply

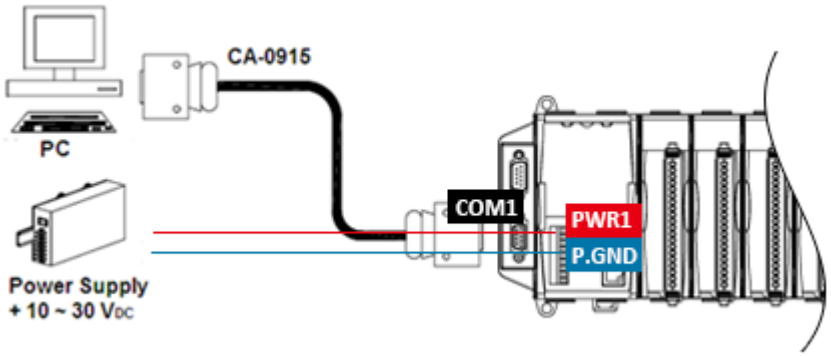
The iPAC-8000 requires a 10 to 30 V_{DC} power supply to operate.

i. Connect the +Vs of the iPAC-8000 to the positive of the power supply.

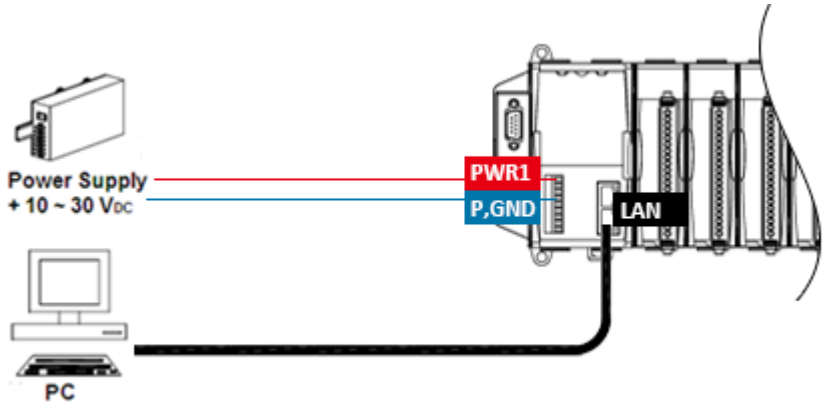
ii. Connect the GND of the iPAC-8000 to the negative of the power supply.

Step 2: Connect to PC

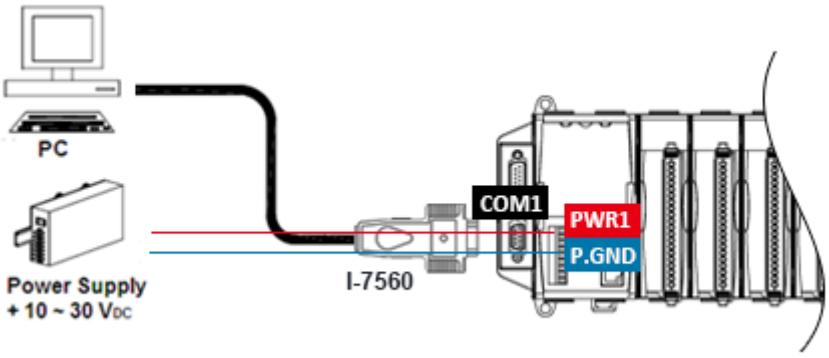
You can connect the COM1 of the iPAC-8000 to PC using a RS-232 cable (CA-0915) that was provided with the package.



For Ethernet controllers, you also can connect to PC using an Ethernet cable.



If the PC not fitted with a COM port, you can use the I-7560 (USB to RS-232 converter) for connection between iPAC-8000 and PC.

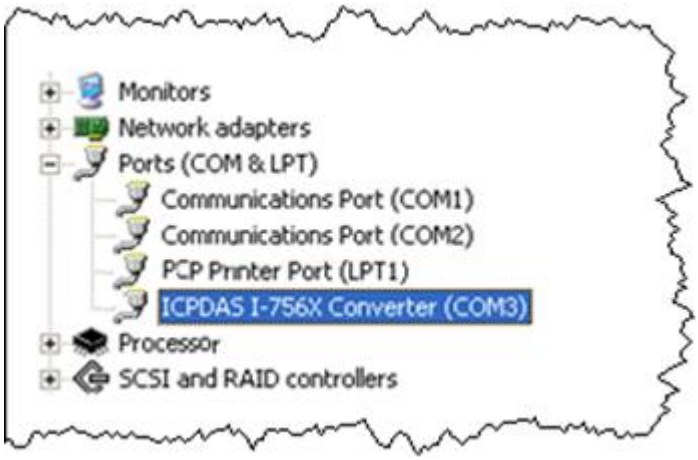


The I-7560 driver must be installed before using the I-7560 converter.

The USB driver can be can be found separately on the CD that was provided with the package or by downloading the latest version from ICP DAS web site.

CD:\8000\Napdos\7000\756x\
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/7000/756x/

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 the I-7560.



2.1.3. Inserting I/O modules

iPAC-8000 has 4/8 expansion I/O slots to expand the functions of iPAC-8000, allowing it to communicate with external I/O devices, 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.

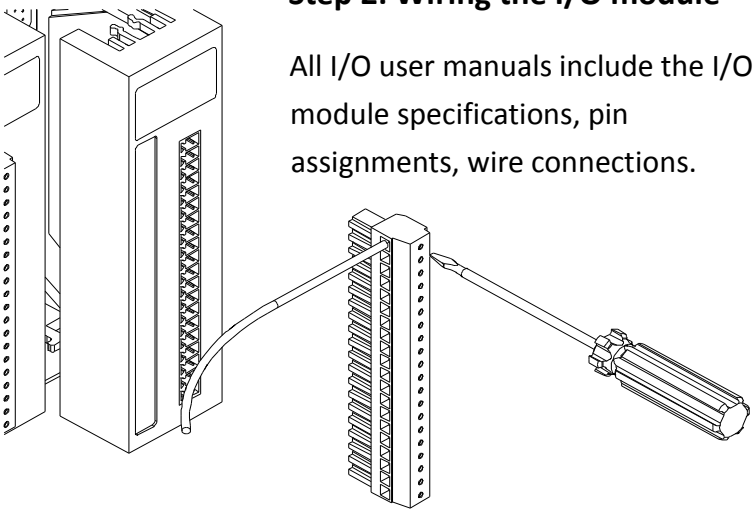
There are more than 30 high profile I/O modules available for interfacing many different measurements, including thermocouple, voltage, RTD, current, resistance, strain, digital,…, etc., and these modules have their own manuals, so if you are using them you should supplement this manual with the manual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

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.

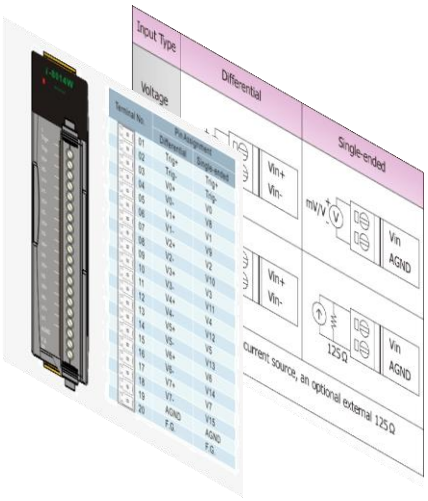
The I/O user manuals can be found separately on the CD that was provided with the package or by downloading the latest version from ICP DAS web site.

http://www.icpdas.com/root/product/solutions/remote_io/rs-485/i-8k_i-87k/i-8k_i-87k_selection.html



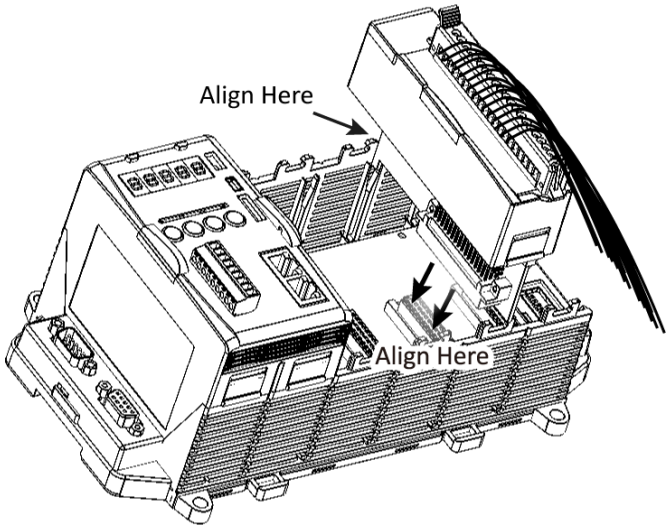
Step 2: Wiring the I/O module

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

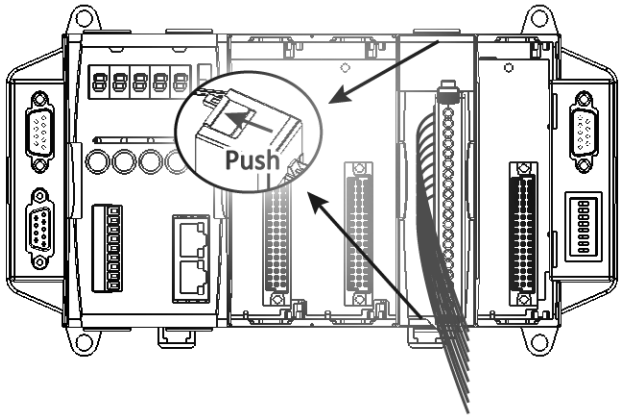


Step 3: Insert the I/O module into I-8000

i. Align circuit card with slot and press firmly to seat module into connector.



ii. Pull top and bottom locking tabs toward module face. Click indicates lock is engaged



2.2. Installing Tools and Utilities

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

2.2.1. Installing the iPAC-8000 Header and Library Files

The iPAC-8000 header and library files can be installed from the CD that was provided with the package or by downloading the latest version from ICP DAS web site.

http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/

CD: \8000\Napdos\iPAC8000\Demo\

- **Basic**
  - The header files, library files, and demo programs for general operations.
- **Framework**
  - The header files, library files, and demo programs for framework operation.
- **TCP**
  - The header files, library files, and demo programs for Ethernet applications.
- **Xserver**
  - The header files, library files, and demo programs for Xserver applications.
- Readme.txt

2.2.2. Installing the MiniOS7

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.

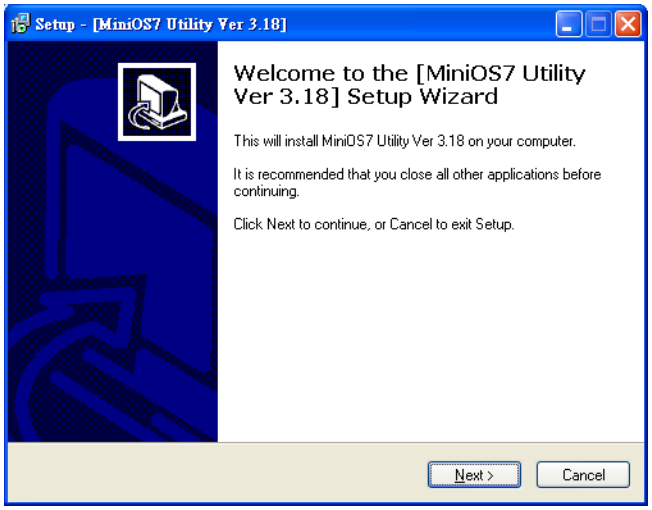
Step 1: Get the MiniOS7 Utility

The MiniOS7 Utility can be found separately on the CD that was provided with the package or by downloading the latest version from ICP DAS web site.

CD:\8000\Napdos\MiniOS7\utility\MiniOS7_utility\
http://ftp.icpdas.com/pub/cd/8000cd/napdos/minios7/utility/minios7_utility/

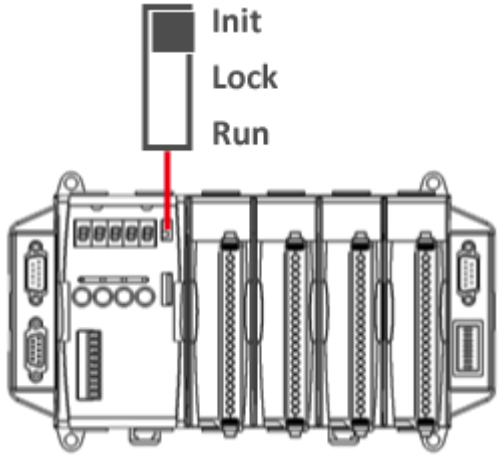
Step 2: Follow the prompts to complete the installation

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



2.3. Configuring the Boot Mode

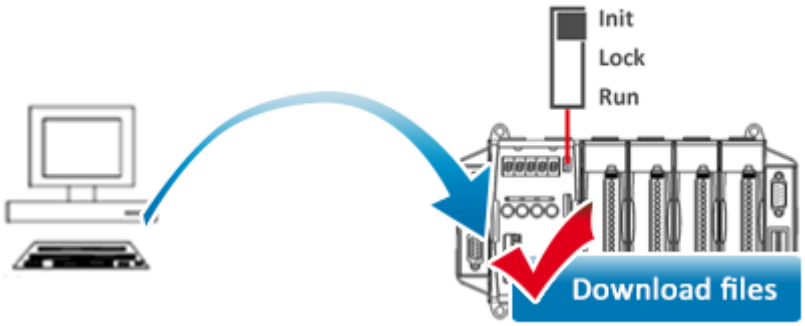
The iPAC-8000 has three standard modes of operation, initial mode, lock mode, and running mode, these modes can be configured by a DIP switch.



The table below compares the differences between these three modes of operation.

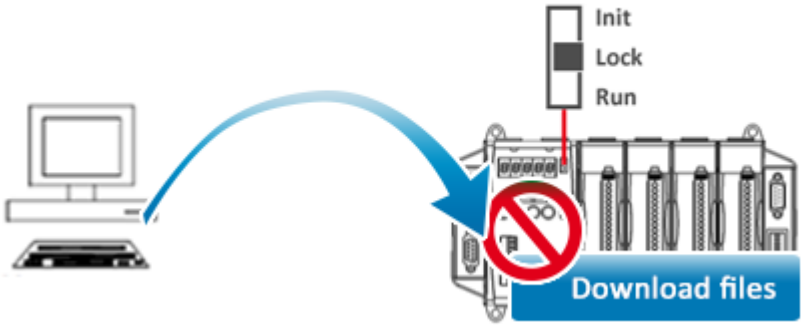
Position	Operating Mode	Description
Init	Initial Mode	OS cannot execute autoexec.bat
		Flash can be read/written
Lock	Lock Mode	OS can execute autoexec.bat
		Flash only can be read (lock)
Run	Running Mode	OS can execute autoexec.bat
		Flash can be read/written

2.3.1. Initial Mode



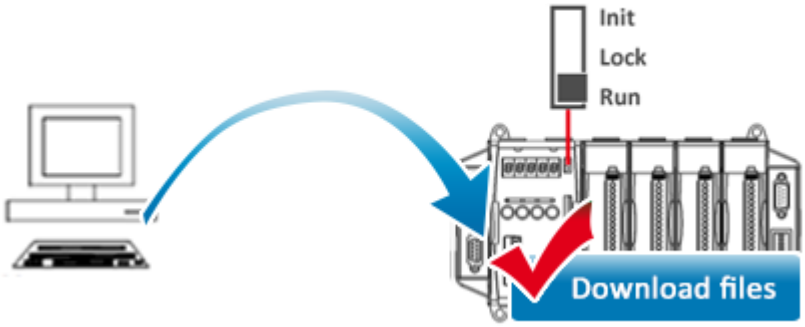
Initial mode is used to update the OS image and upgrade the firmware. When you boot into initial mode, iPAC-8000 doesn't execute the autoexec files and will enter the OS operation mode, in this case there is no program running on the iPAC-8000, the 5-digital 7-SEG LEDs will count the number as shown below, and you can download programs from a PC to the iPAC-8000 using the MiniOS7 Utility.

2.3.2. Lock Mode



Lock mode is used to prevent or protect data stored in the flash memory from being modified. When you boot into lock mode, the flash memory of the iPAC-8000 could not be written unless this mode is specifically disabled. This characteristic prevents the flash memory from be used as main memory. In this case the 5-digital 7-SEG LEDs will display the message according to the running programs, and you can't download programs to the iPAC-8000.

2.3.3. Running Mode



Running mode is used to test possible future functions as an early warning system. When you boot into running mode, iPAC-8000 can not only execute the autoexec files but also download programs at the same time. In this case the 5-digital 7-SEG LEDs will display the message according to the running programs

2.4. Assigning an IP Address (for Ethernet Series PAC only)

The iP-8441/iP-8841/iP-8441-FD/iP-8841-FD is Ethernet PAC which comes with default IP addresses. If you want to add an iPAC-8000 to your network, you must assign a new IP address, subnet mask, and gateway to your iPAC-8000.

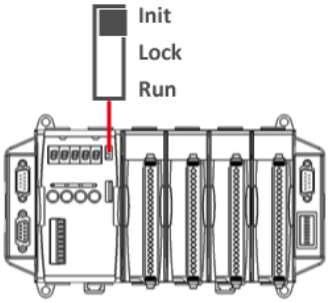
The factory default IP settings are as follows:

Item	LAN1	LAN2
IP Address	192.168.255.1	192.168.255.2
Subnet Mask	255.255.0.0	255.255.0.0
Gateway	192.168.0.1	192.168.0.1

MiniOS7 Utility can be used to configure the IP address. Before starting the configuration process, make sure that the LAN1/LAN2 are used to connect to your network.

Step 1: Reboot the iPAC-8000 into Initial mode

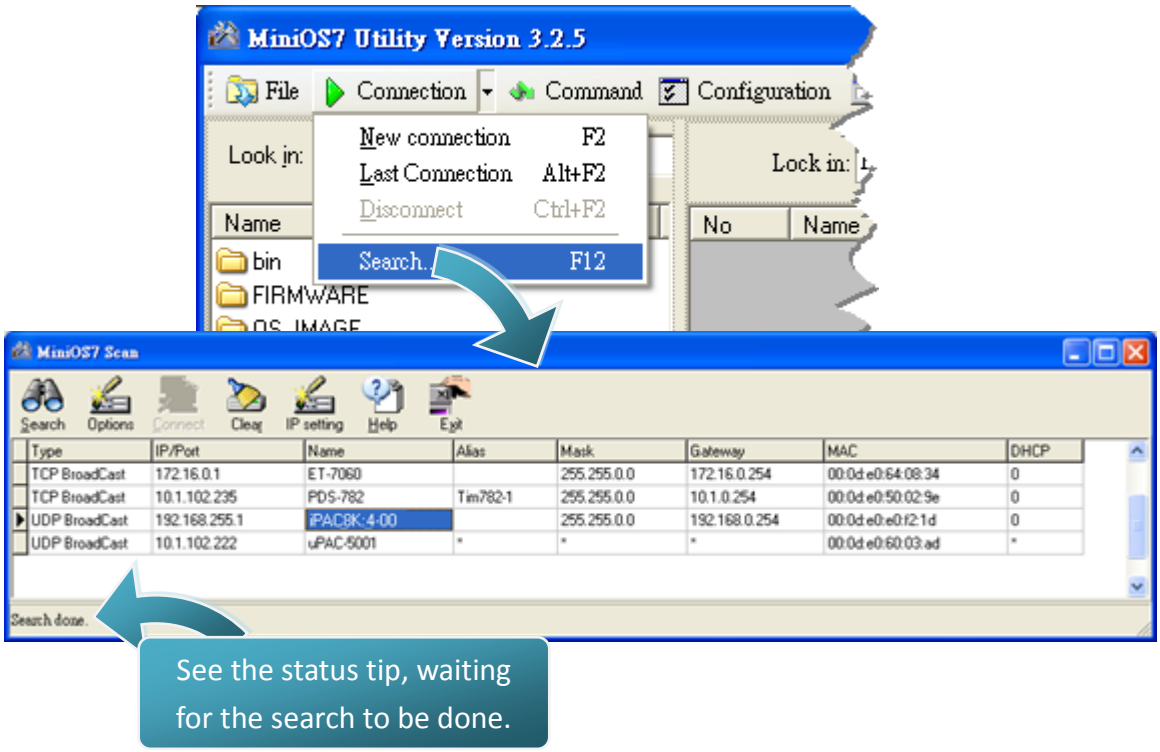
Make sure the DIP switch is in “init” position.



Step 2: Run the MiniOS7 Utility

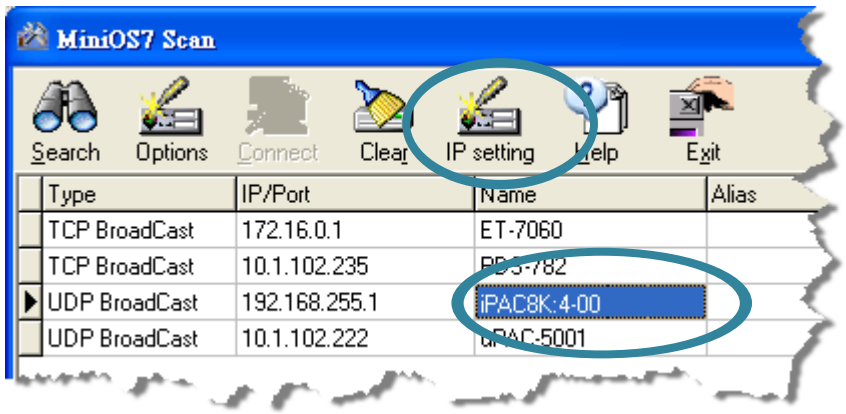
Step 3: Click the "Search" from the "Connection" menu

You need to wait for the process to be done.

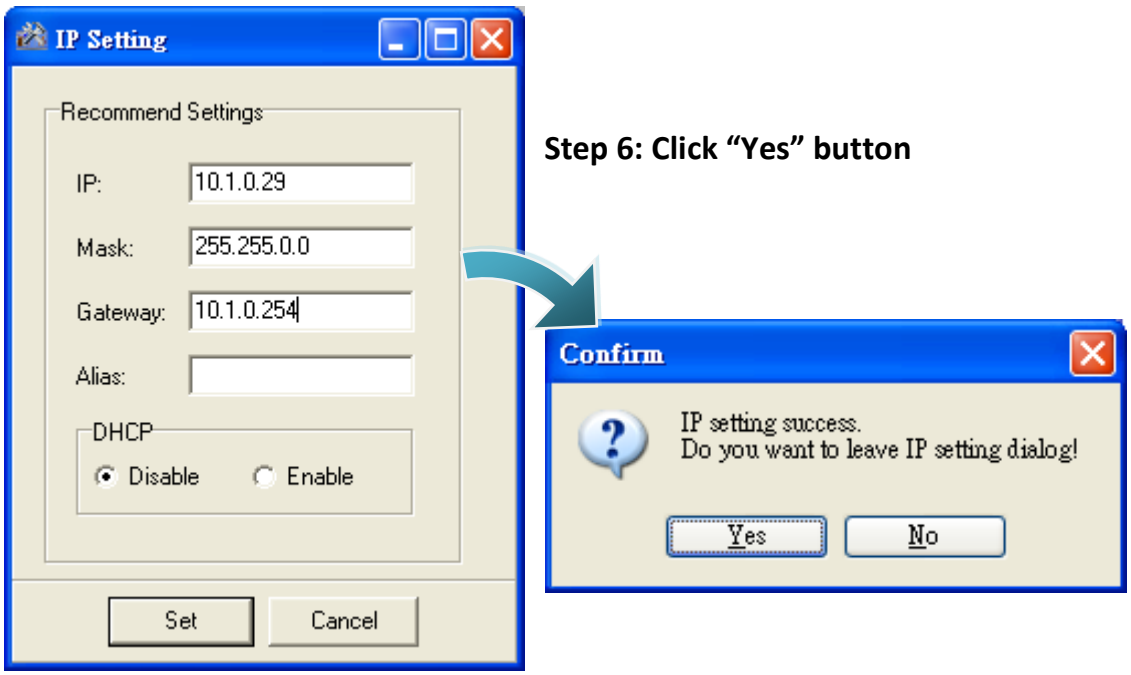


See the status tip, waiting for the search to be done.

Step 4: Choose the name of iPAC-8000 module (which comes with a default IP address "192.168.255.1/192.168.255.2") from the list and then click the "IP setting" from toolbar



Step 5: Configure the “IP” settings and then click the “Set” button



Step 6: Click “Yes” button

2.5. Uploading iPAC-8000 Programs

Before you upload programs into iPAC-8000, ensure 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.”

The upload process has the following main tasks:

1. Establishing a connection between PC and iPAC-8000 (see section 2.5.1)
2. Uploading and executing programs on iPAC-8000 (see section 2.5.2)
3. Making programs start automatically (see section 2.5.3)

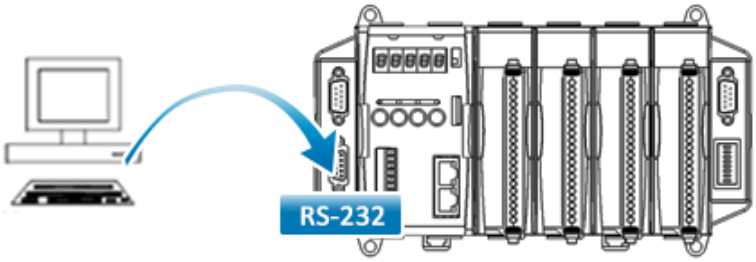
All of these main tasks will be described in detail later.

2.5.1. Establishing a connection between PC and iPAC-8000

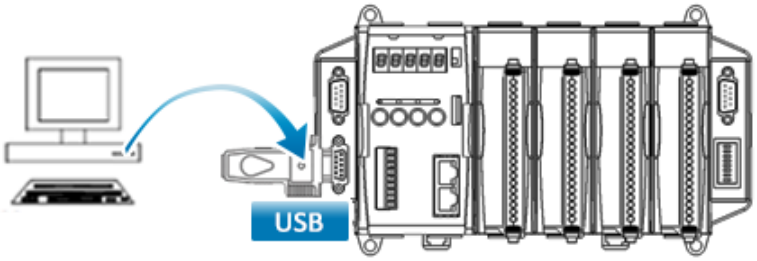
Before you use MiniOS7 Utility to upload programs, ensure that iPAC-8000 is connected to PC. For more information on how to connect iPAC-8000 to PC, see section 2.1.2., “Wiring the iPAC-8000.”

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)

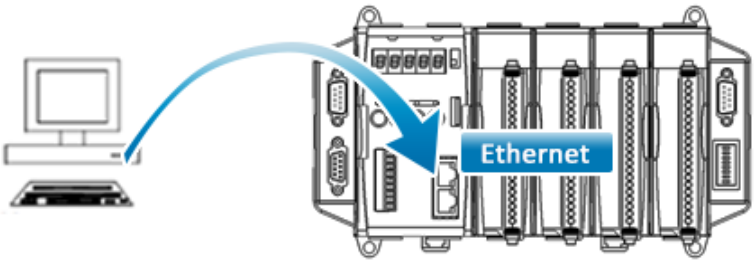


2. USB (see section 2.5.1.2)



3. Ethernet (see section 2.5.1.3)

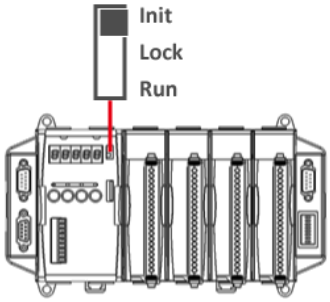
(for Ethernet controllers only)



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

2.5.1.1. Using RS-232 to Establish a Connection

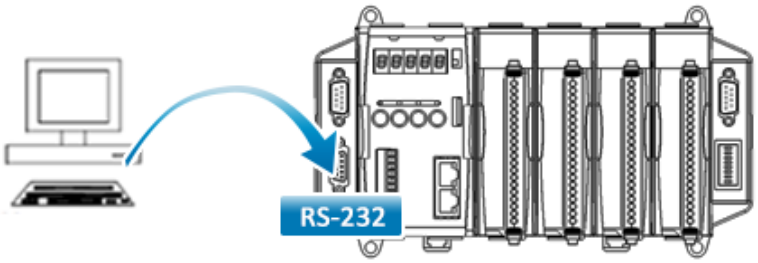
Below are step-by-step instructions on how to connect to PC using a RS-232 connection.



Step 1: Reboot the iPAC-8000 into Initial mode

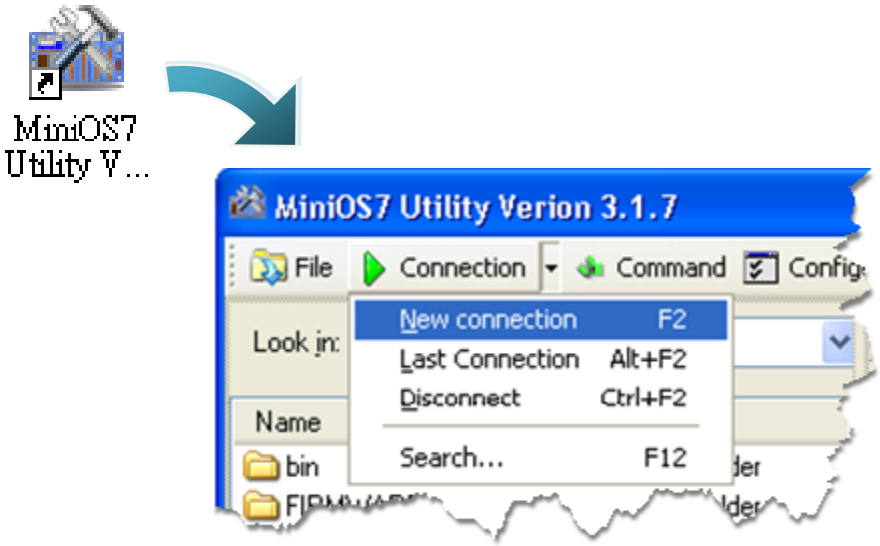
Make sure the DIP switch is in “init” position.

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

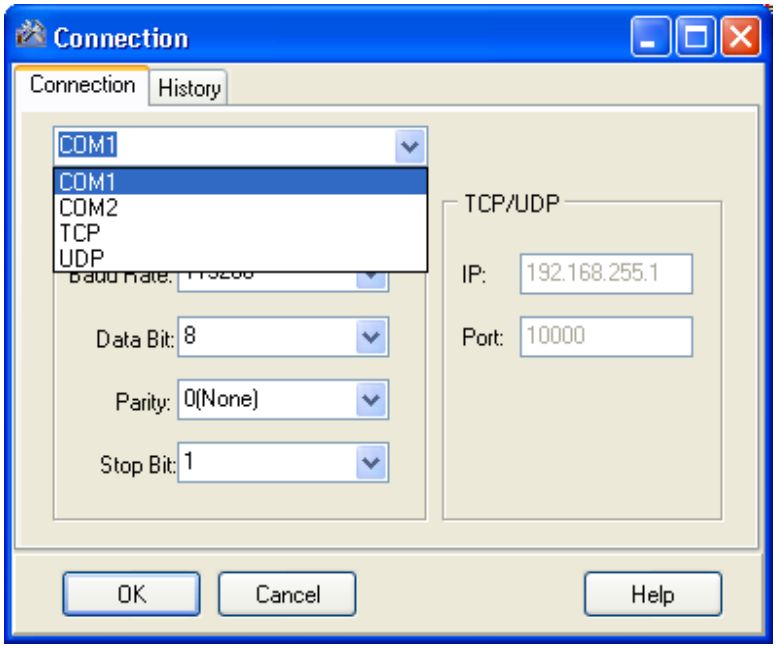


Step 3: Run the MiniOS7 Utility

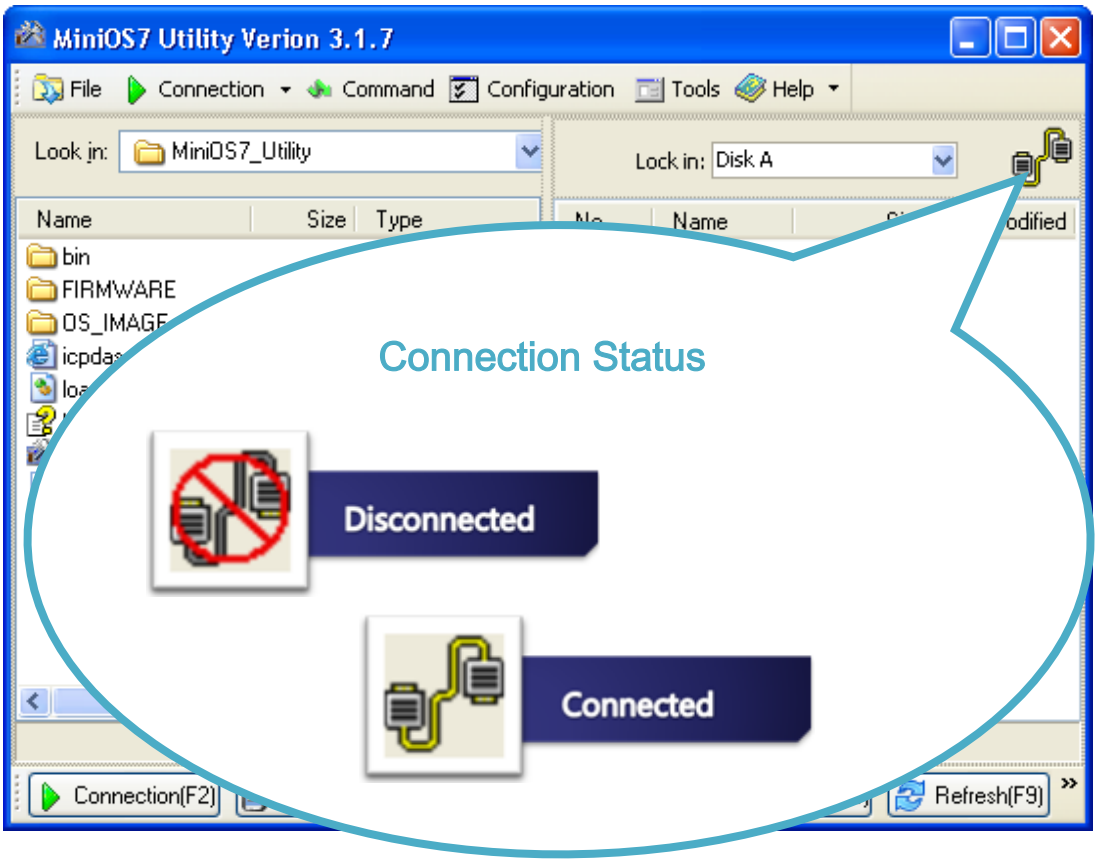
Step 4: Click the “New connection” function from the “Connection” menu



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

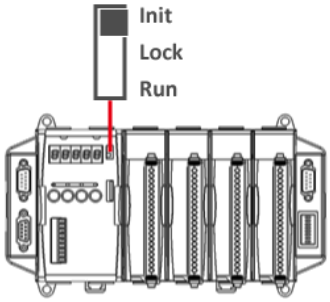


Step 6: The connection has already established



2.5.1.2. Using USB to Establish a Connection

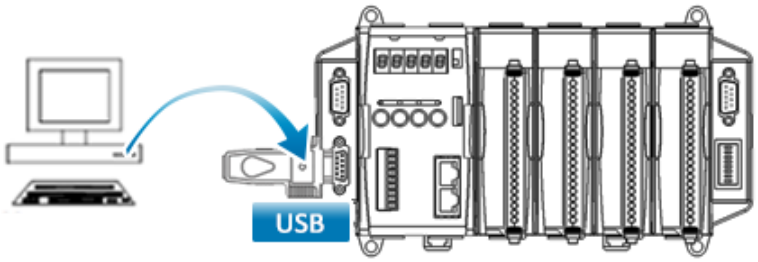
Below are step-by-step instructions on how to connect to PC using an USB connection.



Step 1: Reboot the iPAC-8000 into Initial mode

Make sure the DIP switch is in “init” position.

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

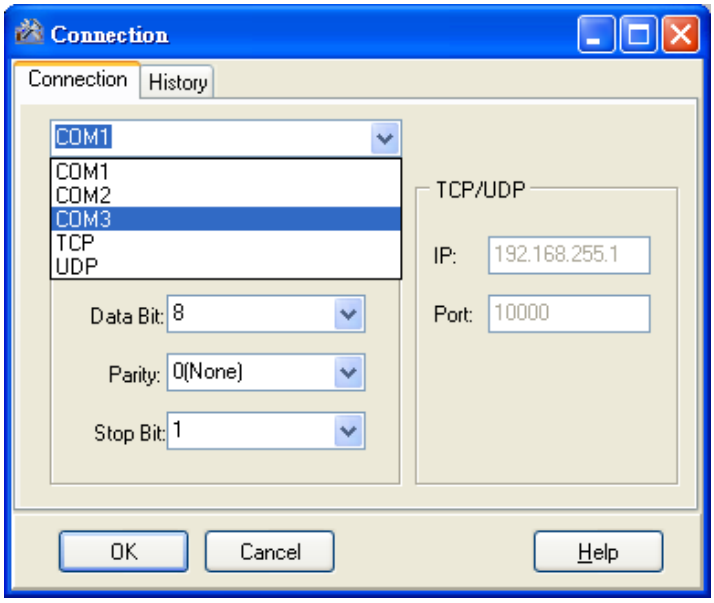


Before using the USB connection, ensure the I-7560 driver that you have installed. If they are not installed, please refer to “section 2.1.2. Wiring the iPAC-8000”.

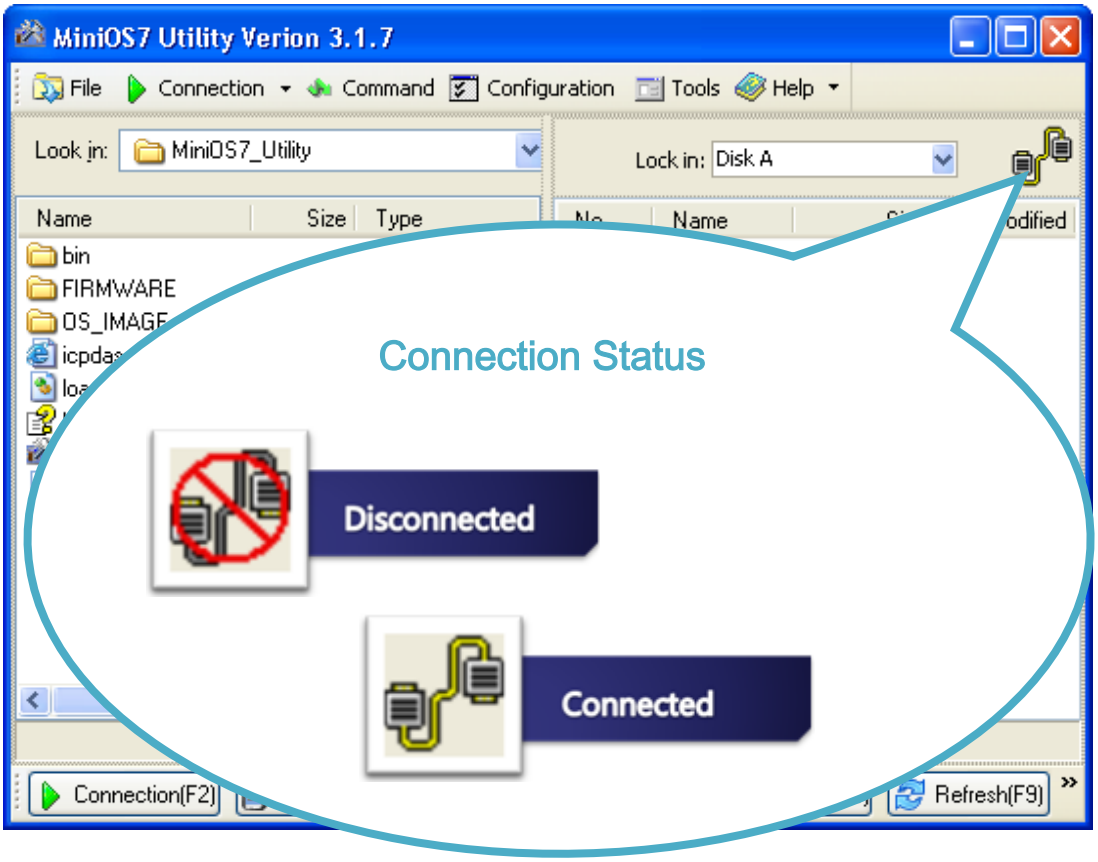
Step 3: Run the MiniOS7 Utility

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"



Step 6: The connection has already established

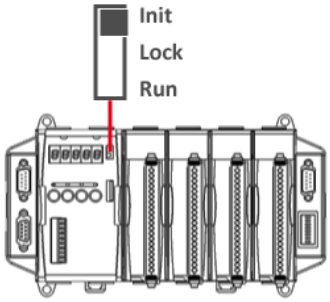


2.5.1.3. Using Ethernet to Establish a Connection

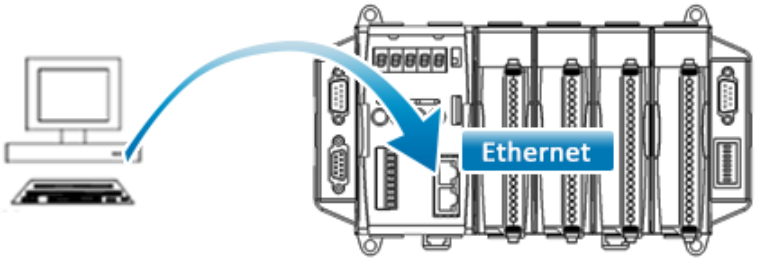
Below are step-by-step instructions on how to connect to PC using an Ethernet connection.

Step 1: Reboot the iPAC-8000 into Initial mode

Make sure the DIP switch is in “init” position.

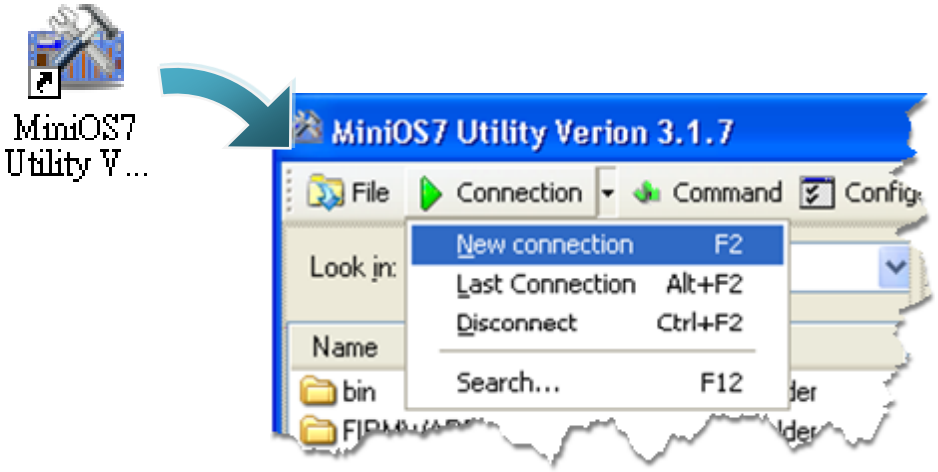


Step 2: Use the RS-232 Cable (CA-0915) to connect to PC

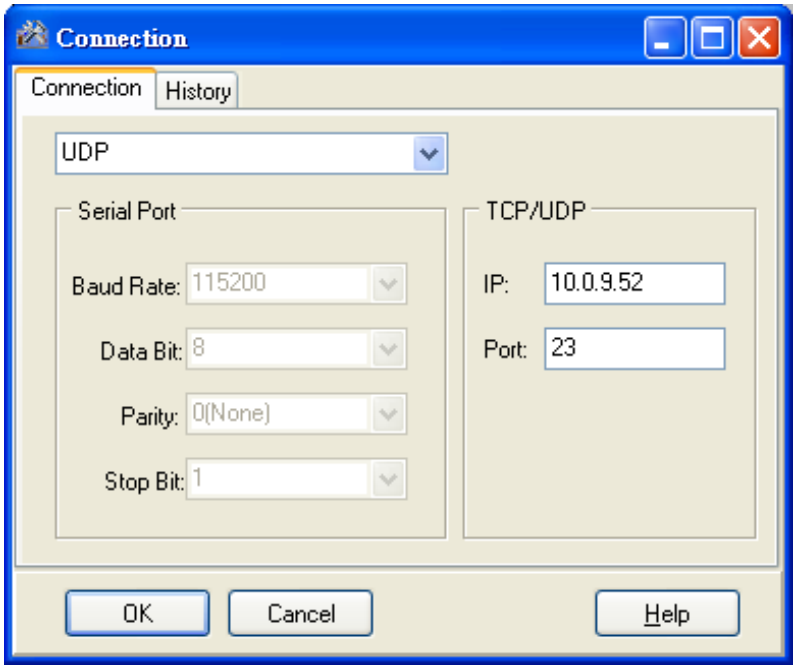


Step 3: Run the MiniOS7 Utility

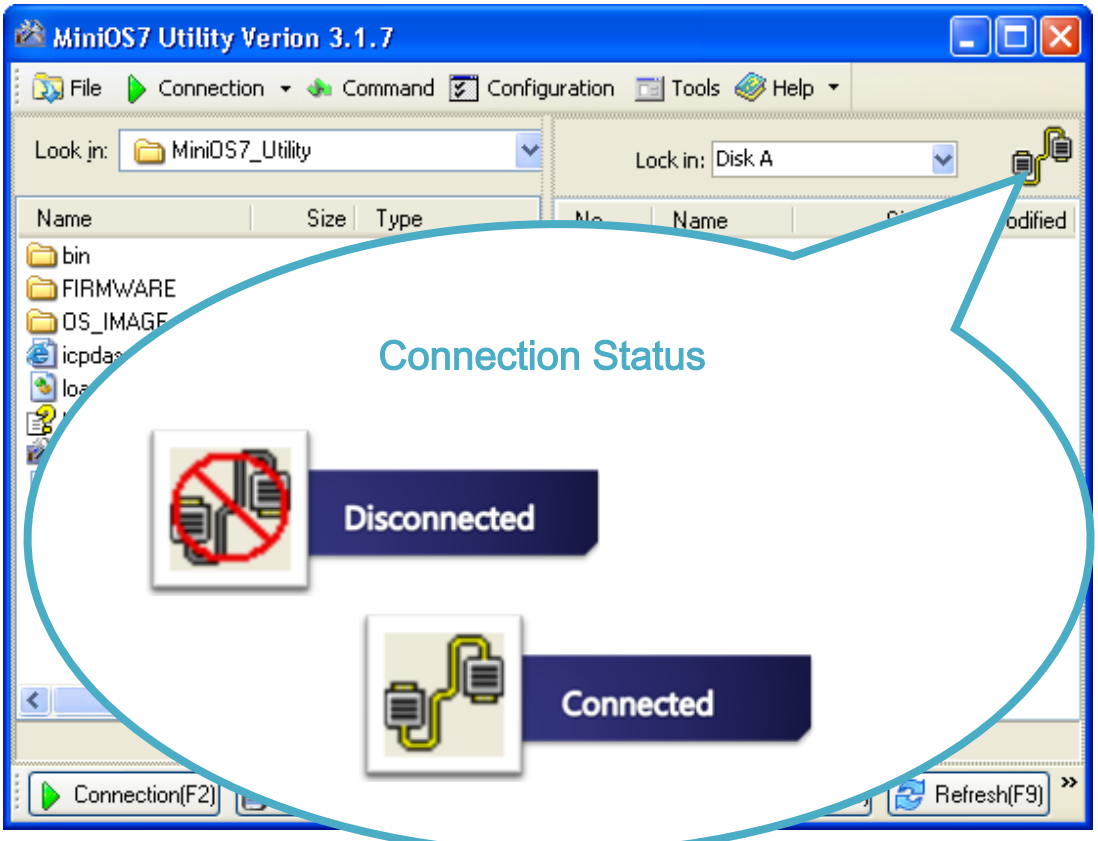
Step 4: Click the “New connection” function from the “Connection” menu



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



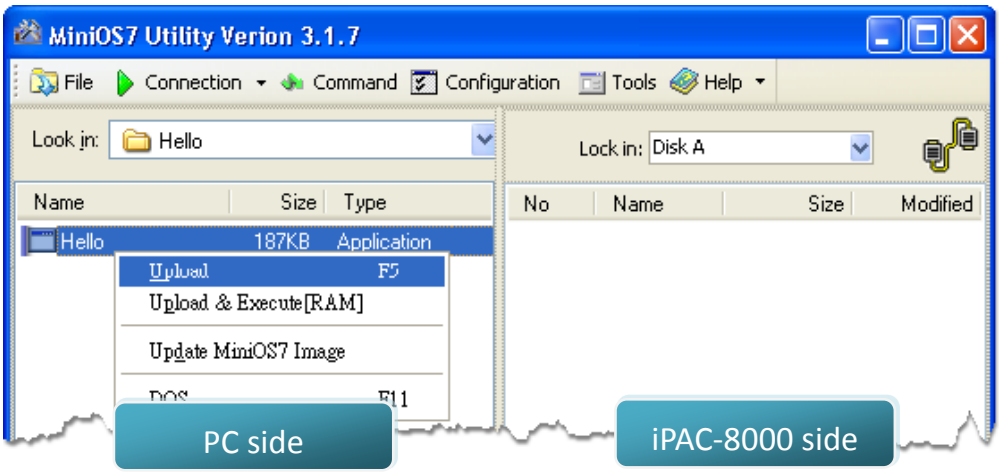
Step 6: The connection has already established



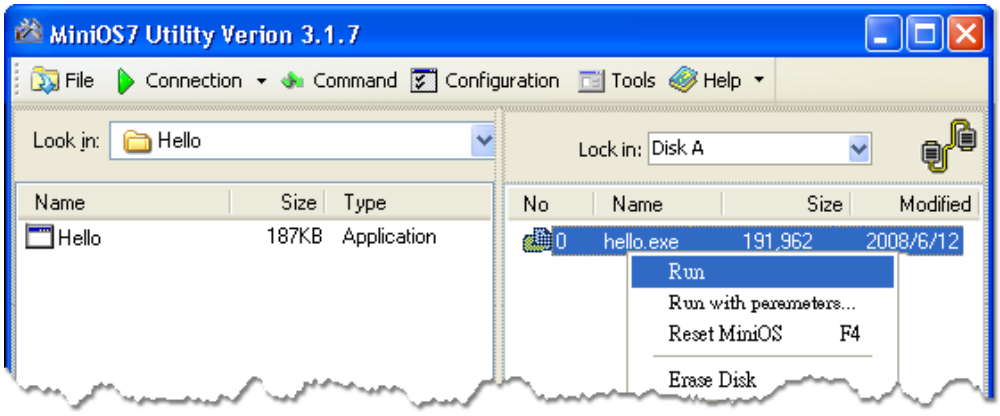
2.5.2. Uploading and Executing iPAC-8000 programs

Before uploading and executing iPAC-8000 programs, you must firstly establish a connection between PC and iPAC-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 on the iPAC-8000, if you need programs to start automatically after the iPAC-8000 start-up, it is easy to achieve it, to create a batch file called autoexec.bat and then upload it to the iPAC-8000, the program will start automatically in the next start-up.

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

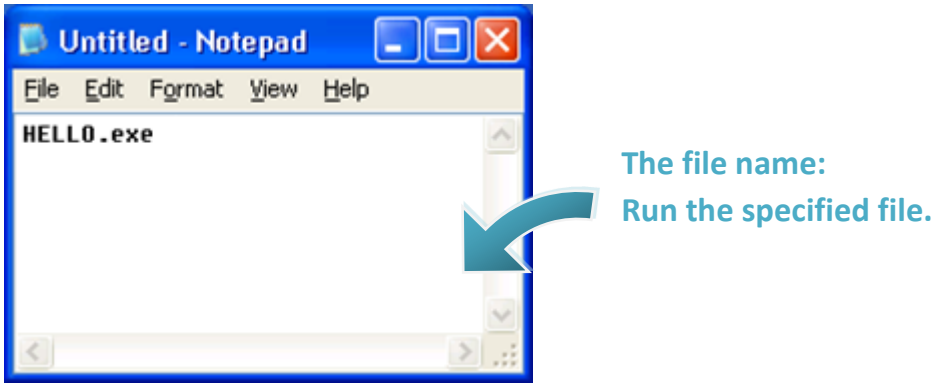
Step 1: Create an autoexec.bat file

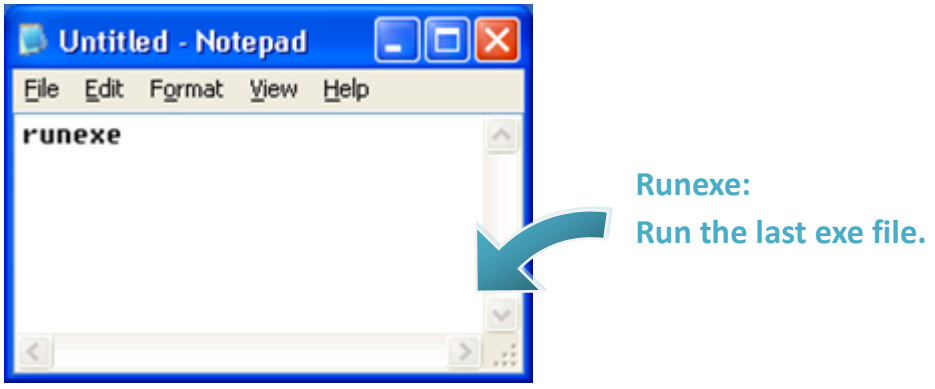
i. Open the "Notepad"

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)

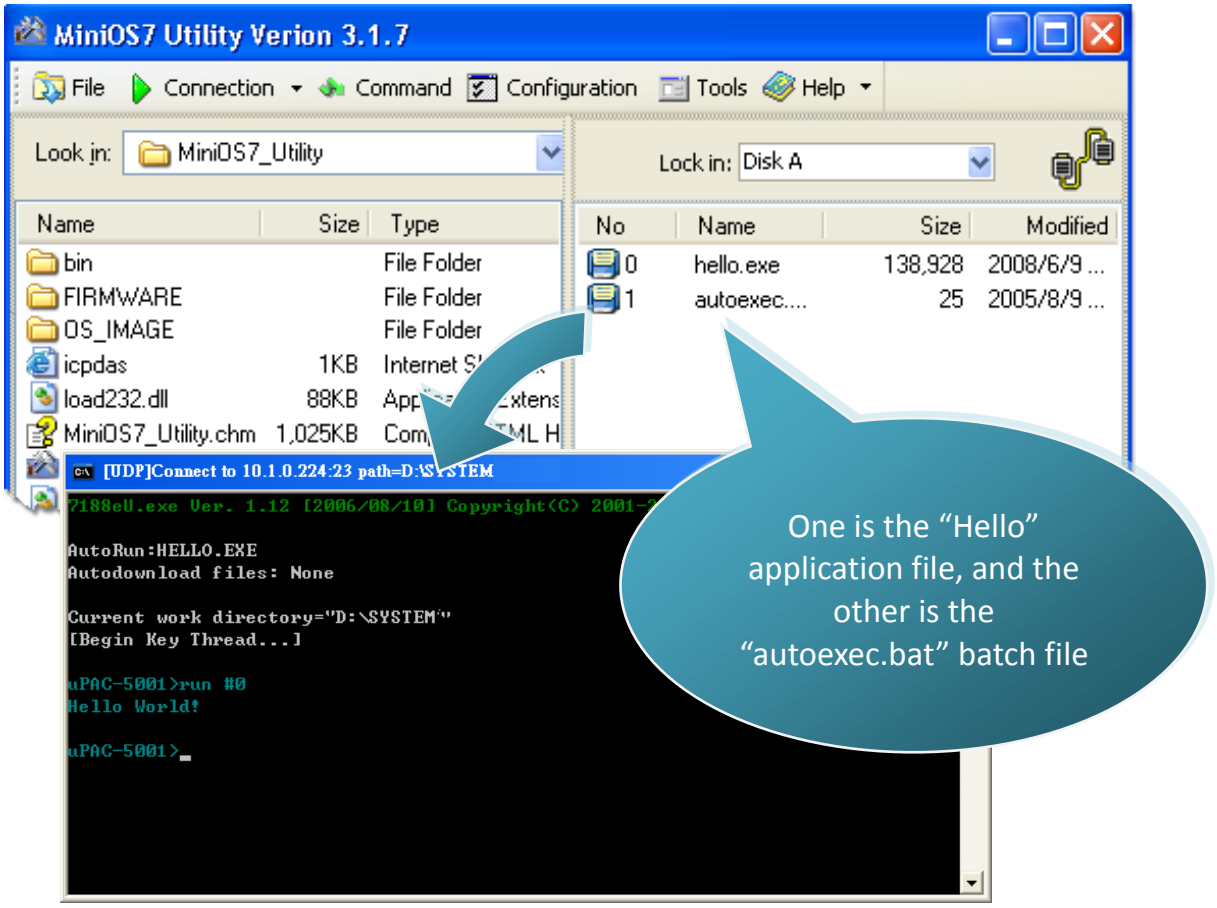
iii. Save the file as autoexec.bat





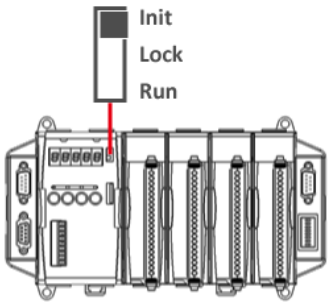
Step 2: Upload programs to iPAC-8000 using MiniOS7 Utility

For more detailed information about this process, please refer to section “2.5.2. Uploading and executing iPAC-8000 programs”



Tips & Warnings

Before restaring the iPAC-8000 for settings to take effect, you must firstly turn the switch to “Init” position.



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.

When using Windows 64-bit platform like Windows 7 or Windows 8, it can’t directly use C compiler , please refer Appendix E :How to build “MiniOS7” project on Windows 7, 8 or others 64-bit platform

3.1. Choosing a C Compiler

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

Before writing your first iPAC-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 as the libraries have been created on the companion CD.

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.

Tips & Warnings

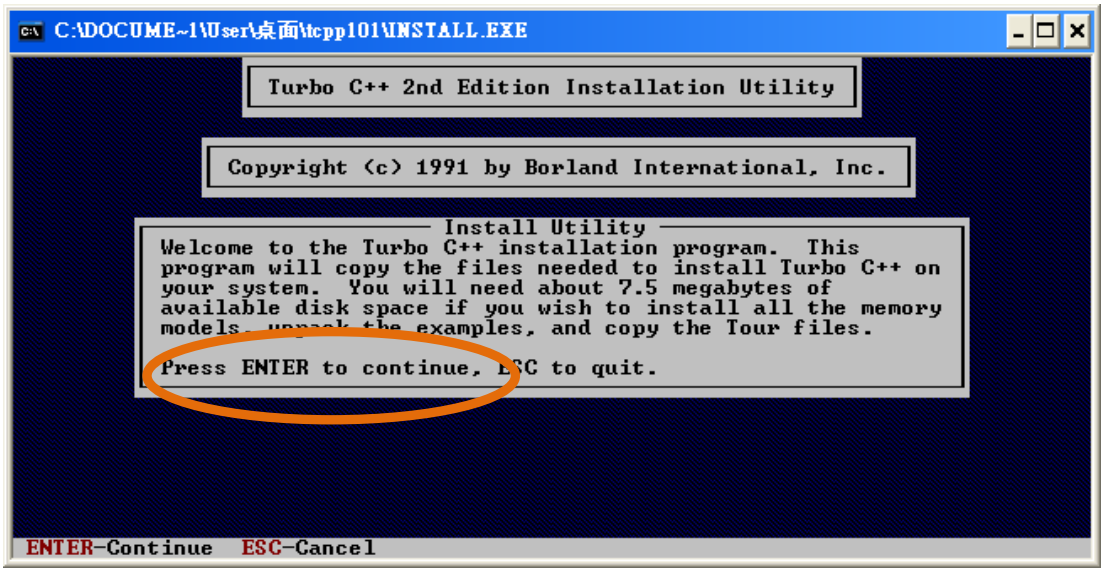
Windows 7, Windows 8, or others 64-bit Windows are incompatible with a 16-bit compiler. If we want to run the program that was compiled with Turbo C++ 1.0.1 on 64-bit Windows, please refer to Appendix E.2 How to Install the Turbo C++ on 64-bit Windows.

Step 1: Get the Turbo C++

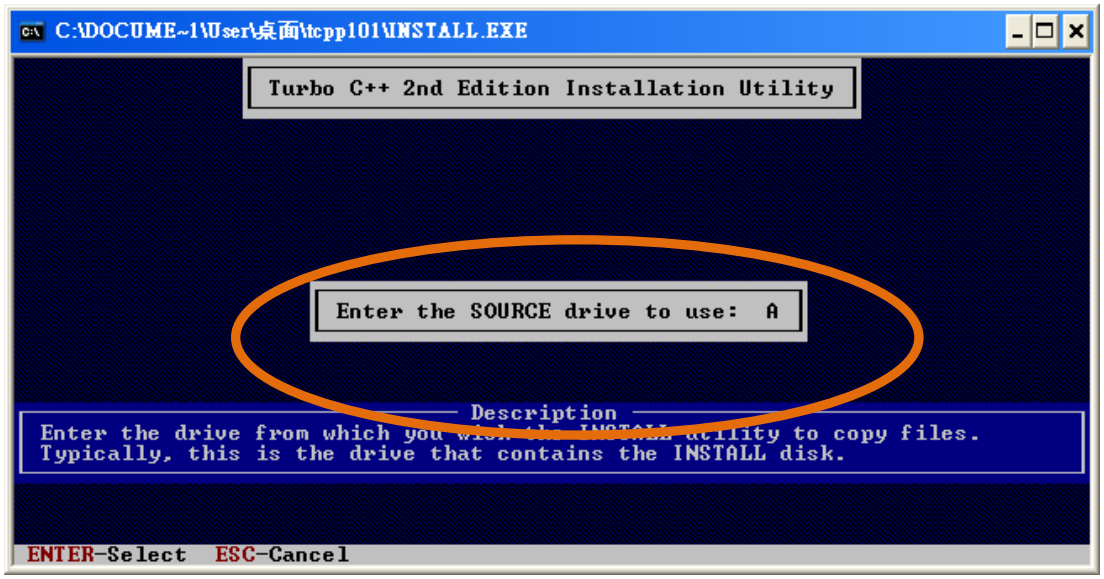
The Turbo C++ 3.0 can be obtained from the following link.
http://www.bestfreewaredownload.com/download/t-free-turbo-c--freeware-flggsdpz.html

Step 2: extract the file and then install it

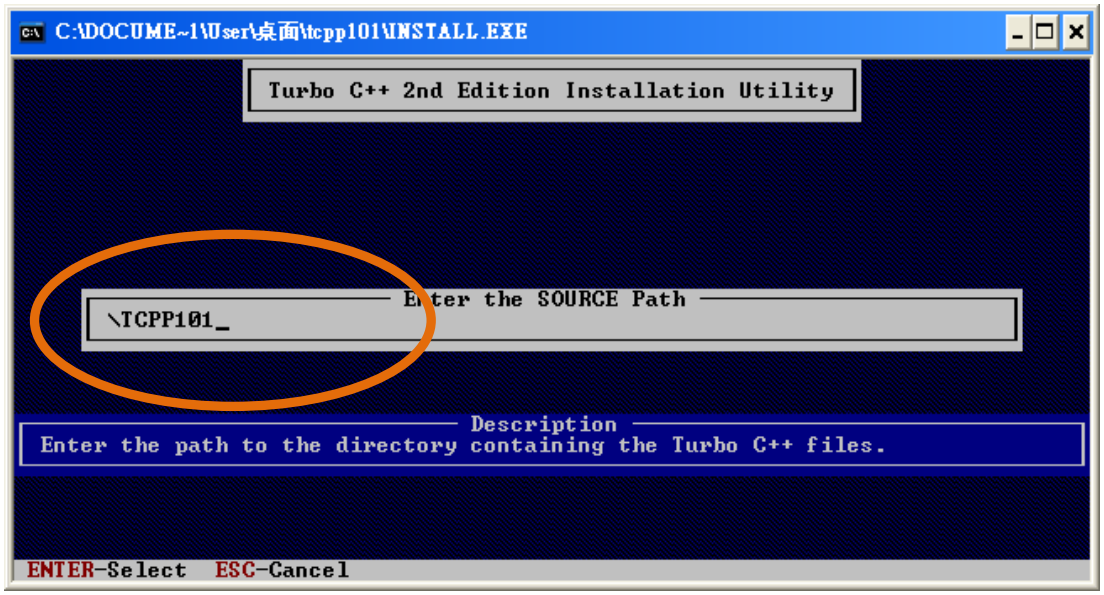
Step 3: Press “Enter” to continue



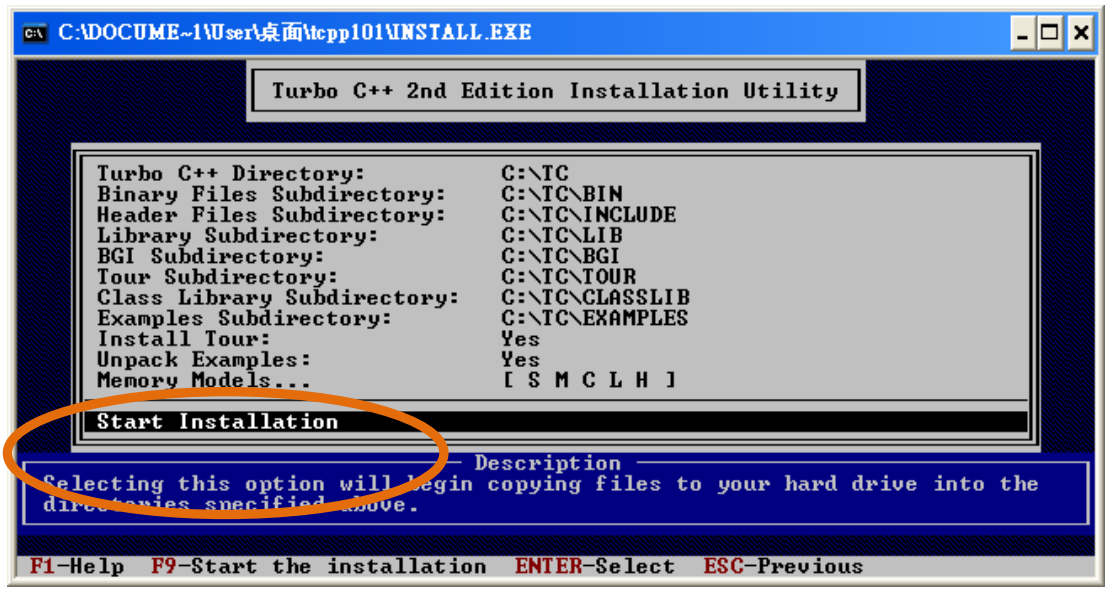
Step 4: Enter the letter of the hard drive you wish to install the software



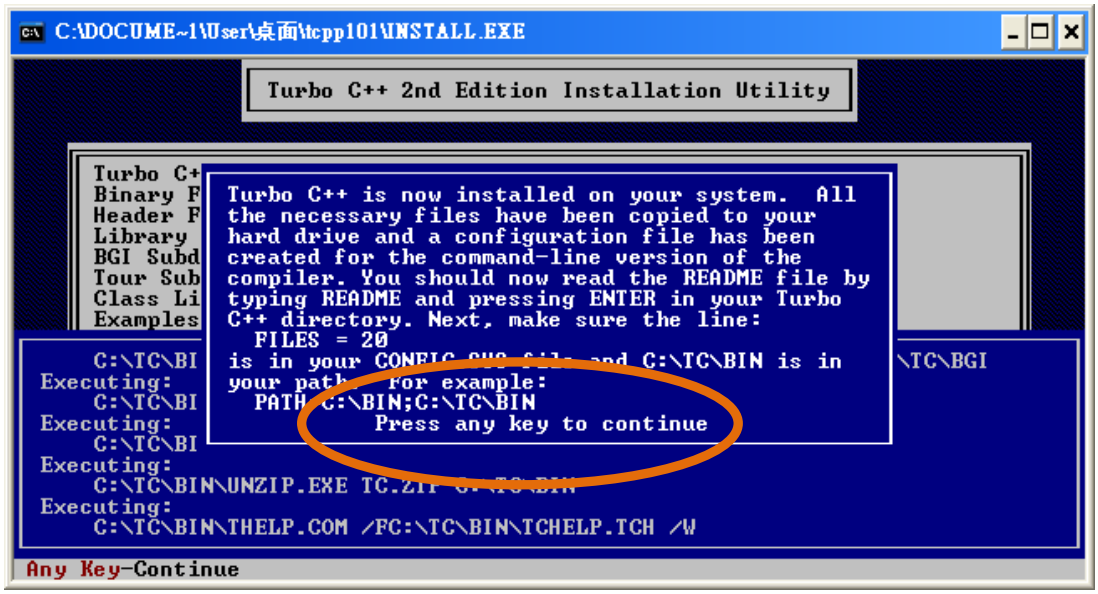
Step 5: Enter the path to the directory you wish to install files to



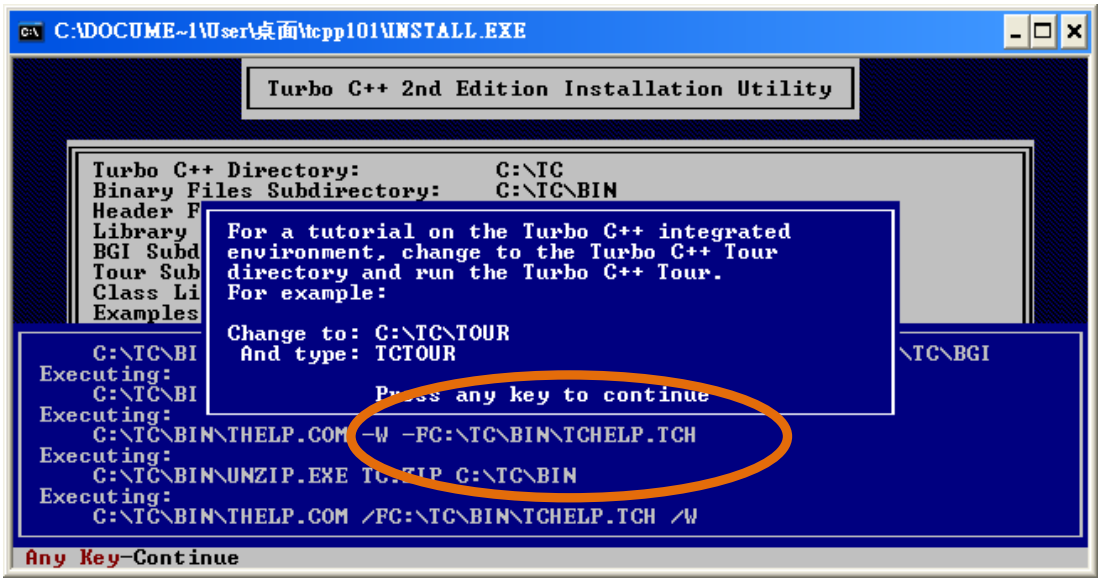
Step 6: Select "Start Installation" to begin the install process



Step 7: Press any key to continue



Step 8: Press any key to continue

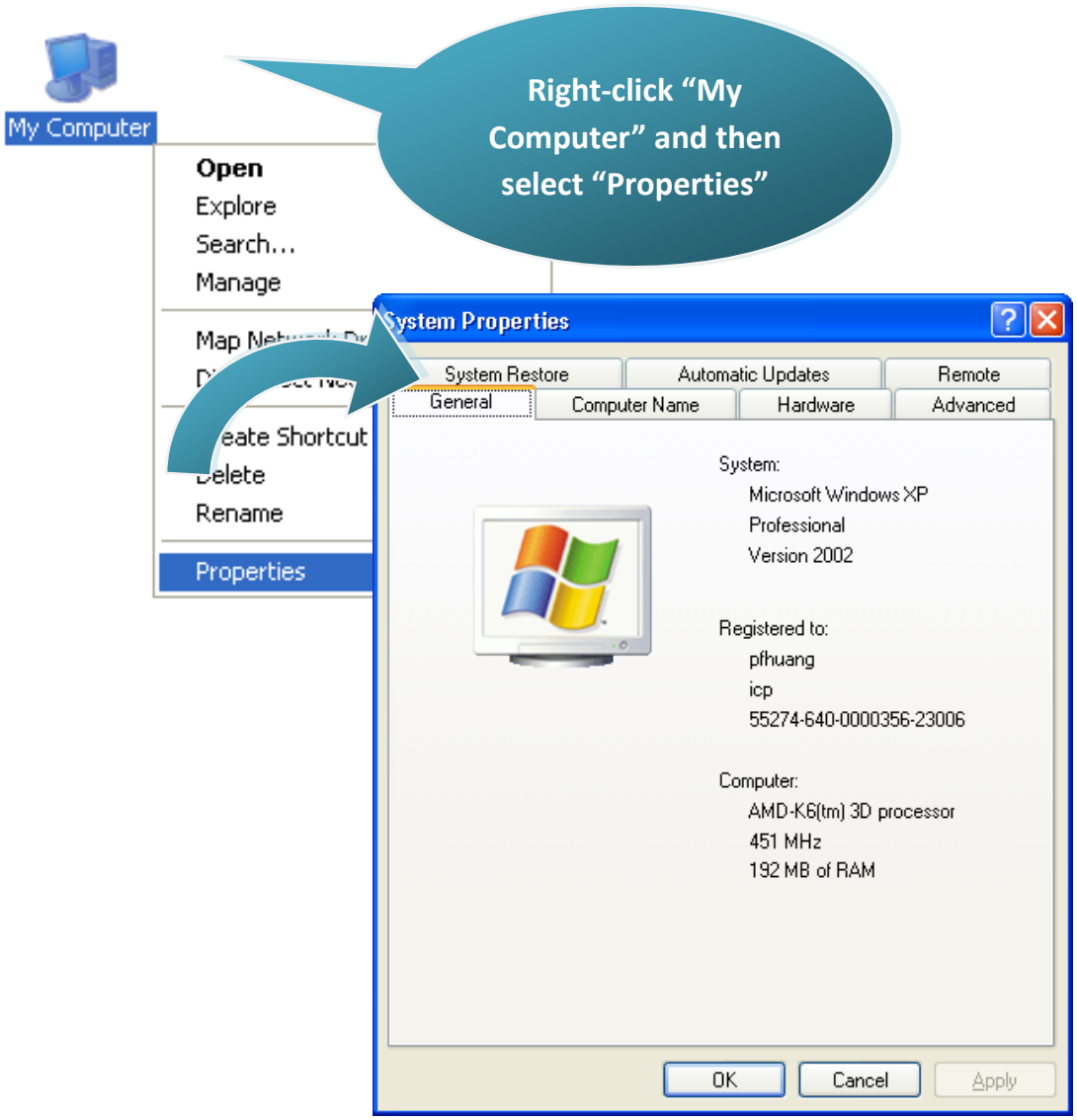


Step 9: Installation is complete

3.1.2. Setting up the environment variables

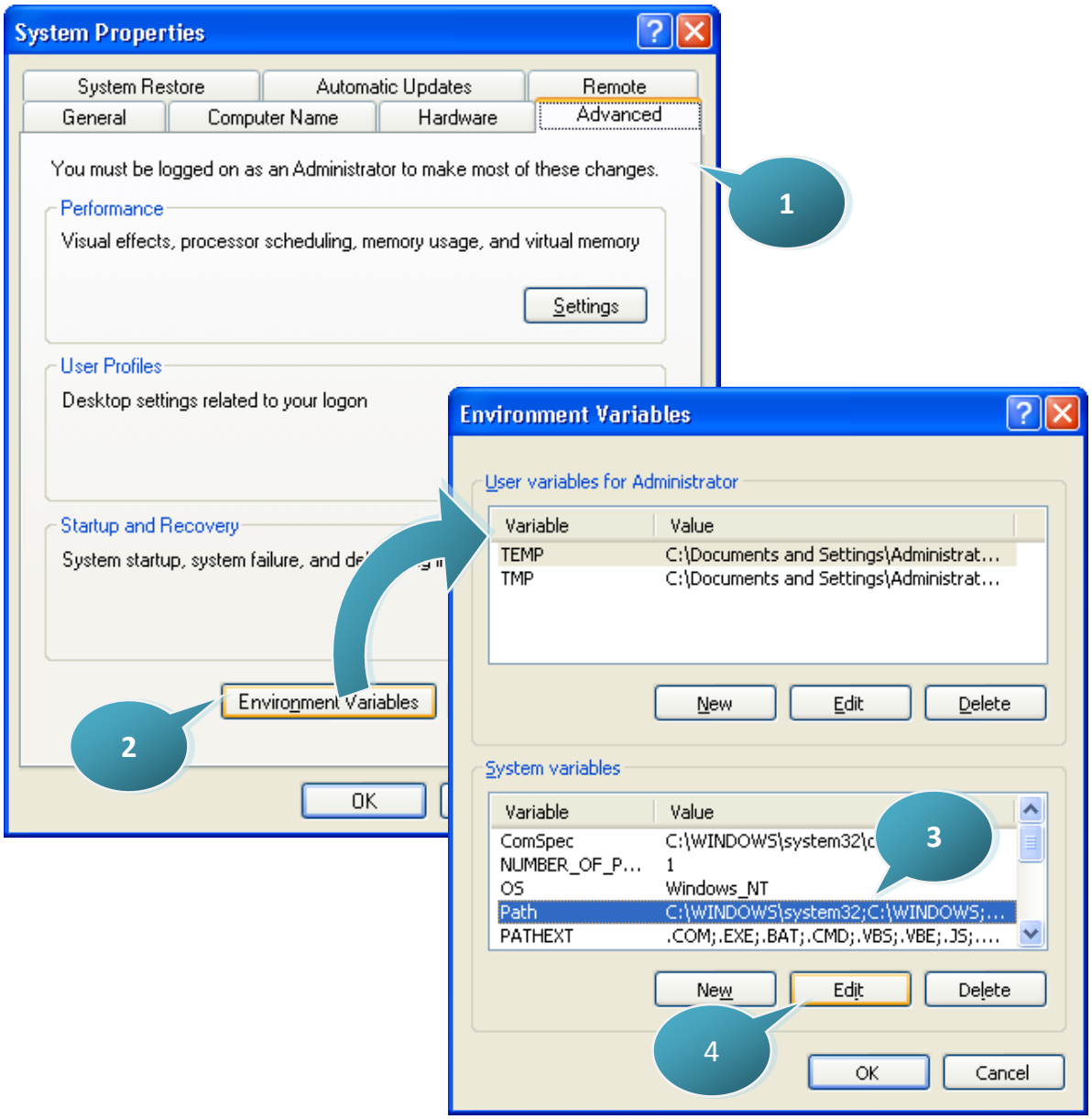
After installing the compiler, several compilers will be available from the Windows Command line. You can set the path environment variable so that you can execute this compiler on the command line by entering simple names, rather than by using their full path names.

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



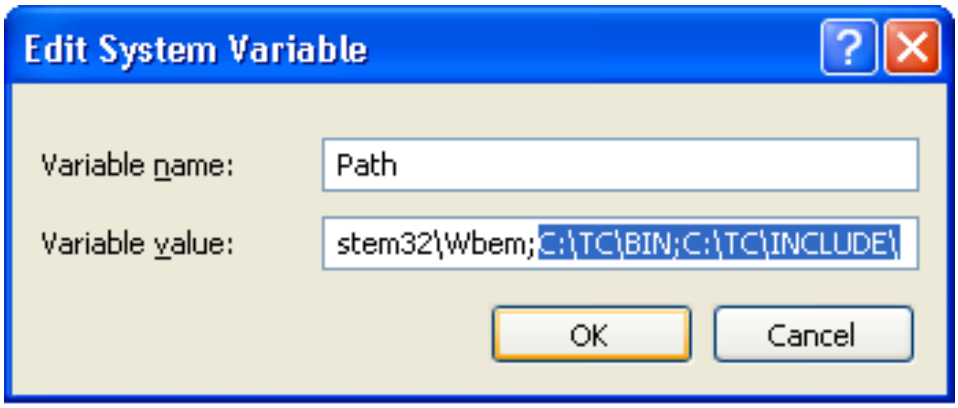
Step 2: On the "System Properties" dialog box, click the "Environment Variables" button located under the "Advanced" sheet

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



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\”



Step 5: Reboot the computer to allow your changes to take effect

3.2. Getting the iPAC-8000 APIs

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

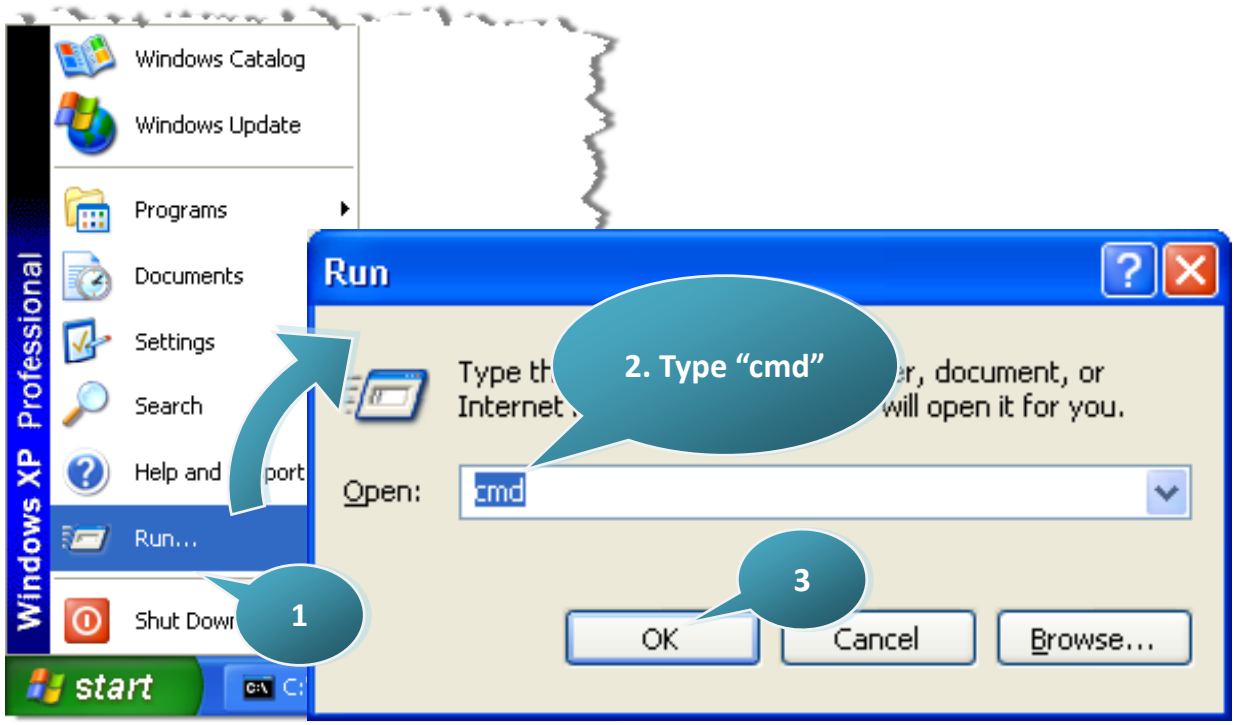
Before creating the application, ensure them that you have installed. If they are not installed, please refer to “section 2.2.1. Installing the iPAC-8000 header and libraries files”.

3.3. Creating Your First iPAC-8000 Program

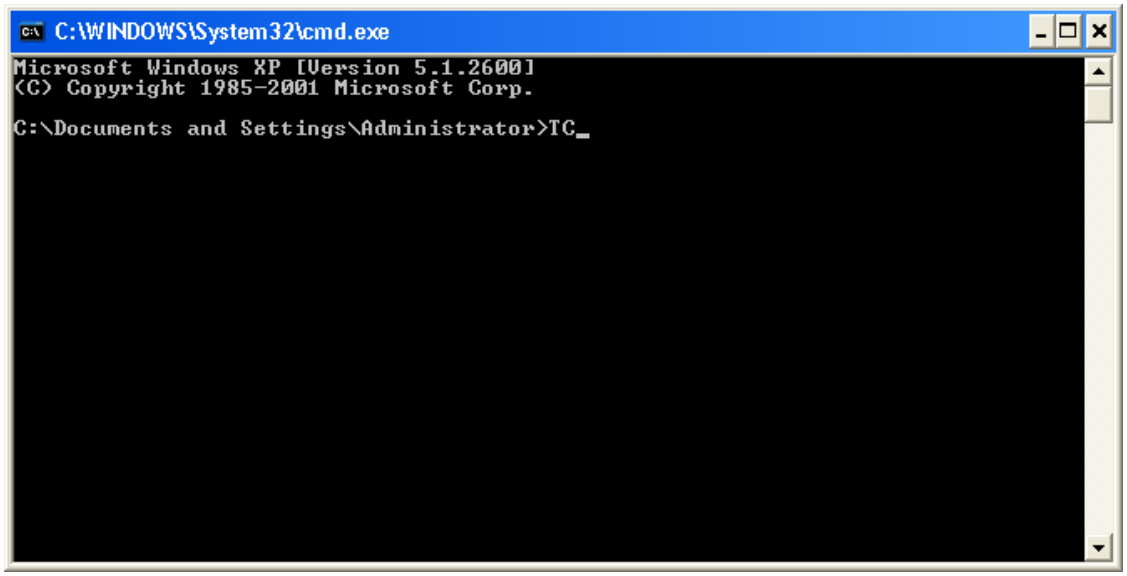
Here we assume you have installed the Turbo C++ 1.01 (as the section “3.1.1. Installing the C Compiler”) and the iPAC-8000 APIs (as the section “2.2.1. Installing the iPAC-8000 Header and Library Files”) under the C driver root folder.

Step 1: Open a MS-DOS command prompt

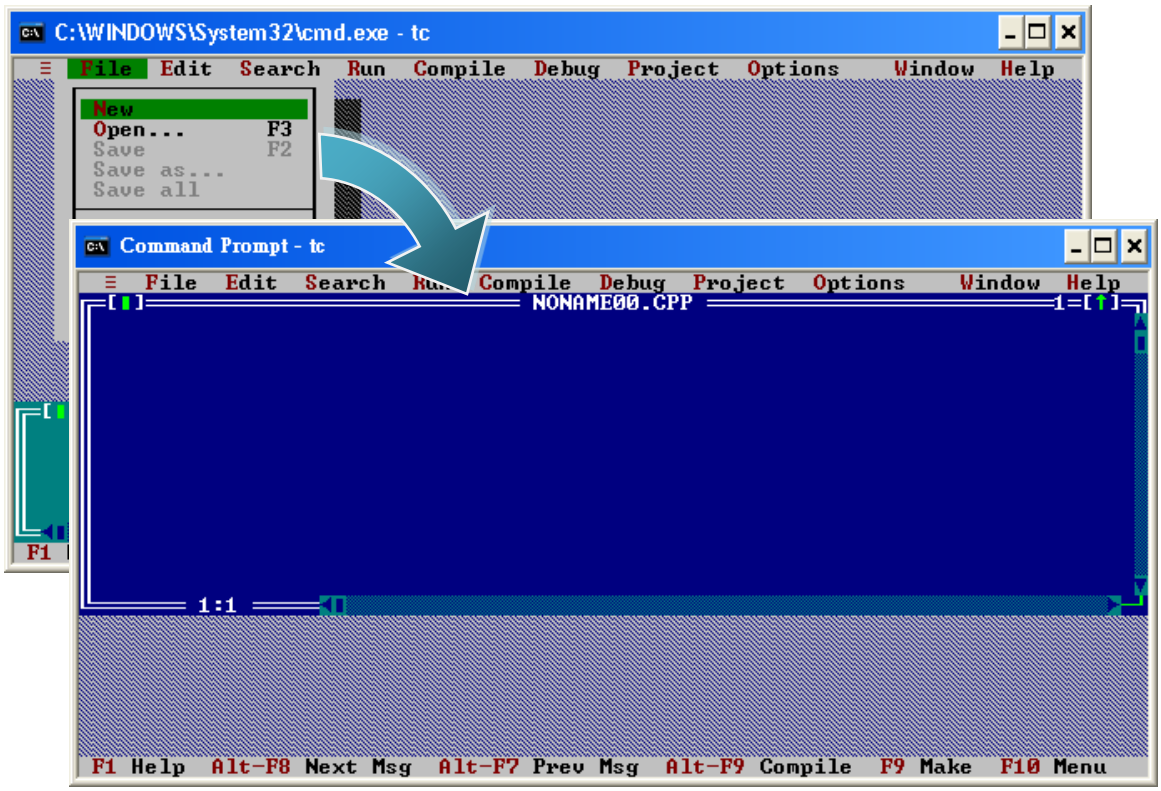
i. Select “Run” from the “Start” menu
ii. On the “Run” dialog box, type “cmd”
iii. Click the “OK” button



Step 2: At the command prompt, type “TC” and then press “Enter”



Step 3: Select “New” from the “File” menu to create a new source file



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

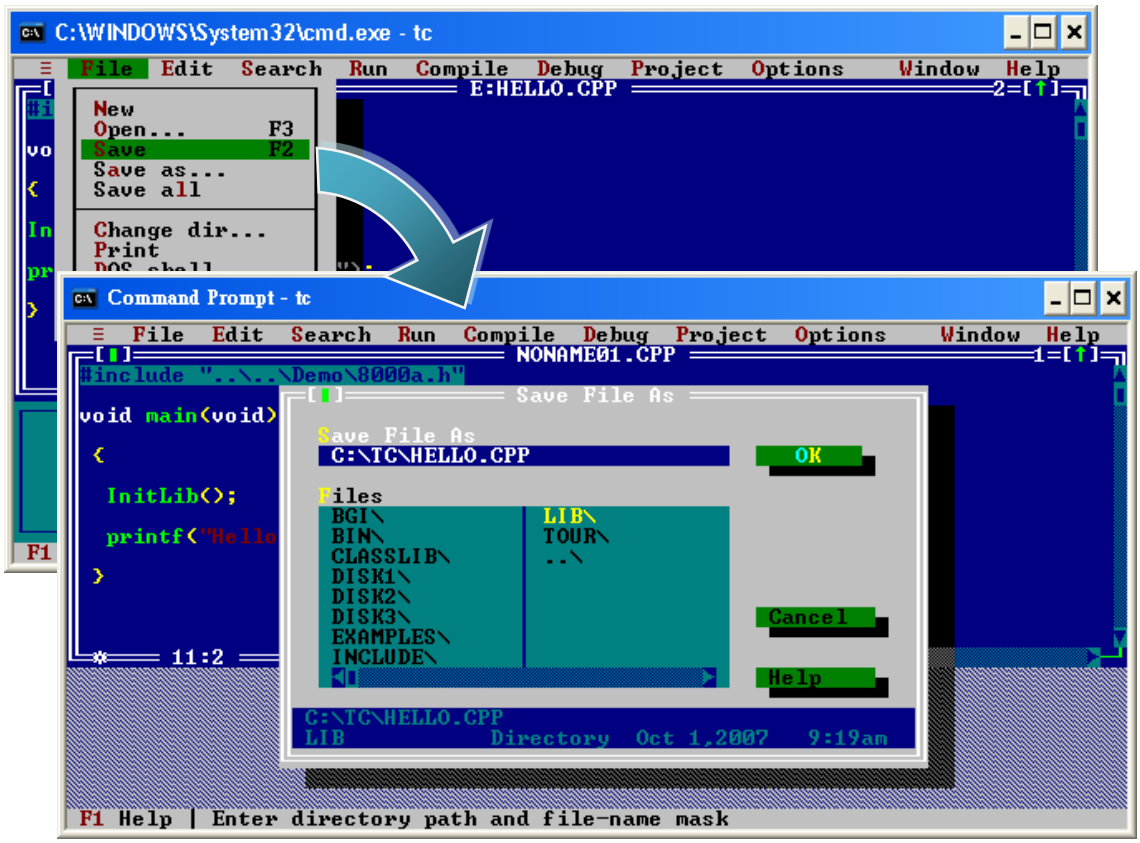
```
#include "..\..\Demo\basic\Lib\8000a.h"
/* Include the header file that allows 8000a.lib functions to be used */

void main(void)
{
  InitLib();      /* Initiate the 8000a library */
  Print("Hello 8000!\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”



Tips & Warnings

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

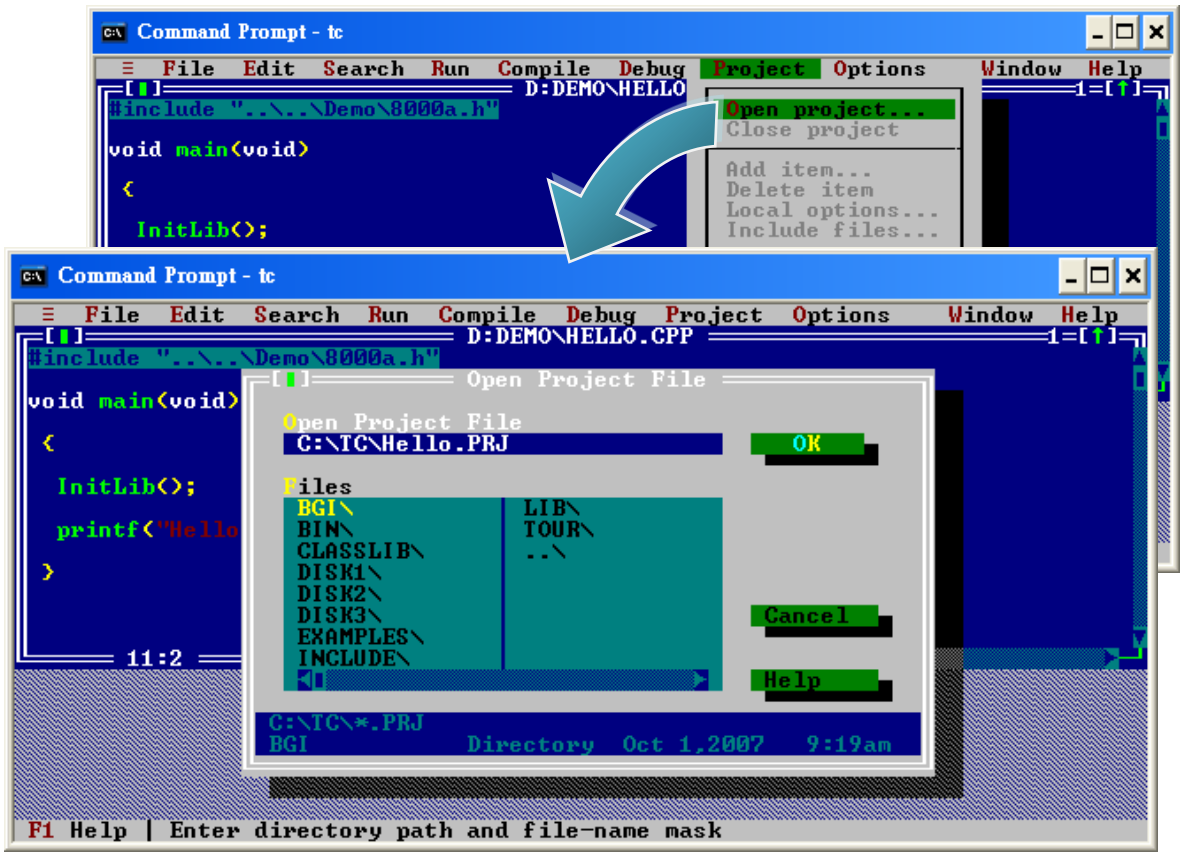
you must save the source code under a filename that terminates with the extension “C”.

Step 6: Create a project (*.prj)

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

ii. Type the project name “Hello”

iii. Select “OK”



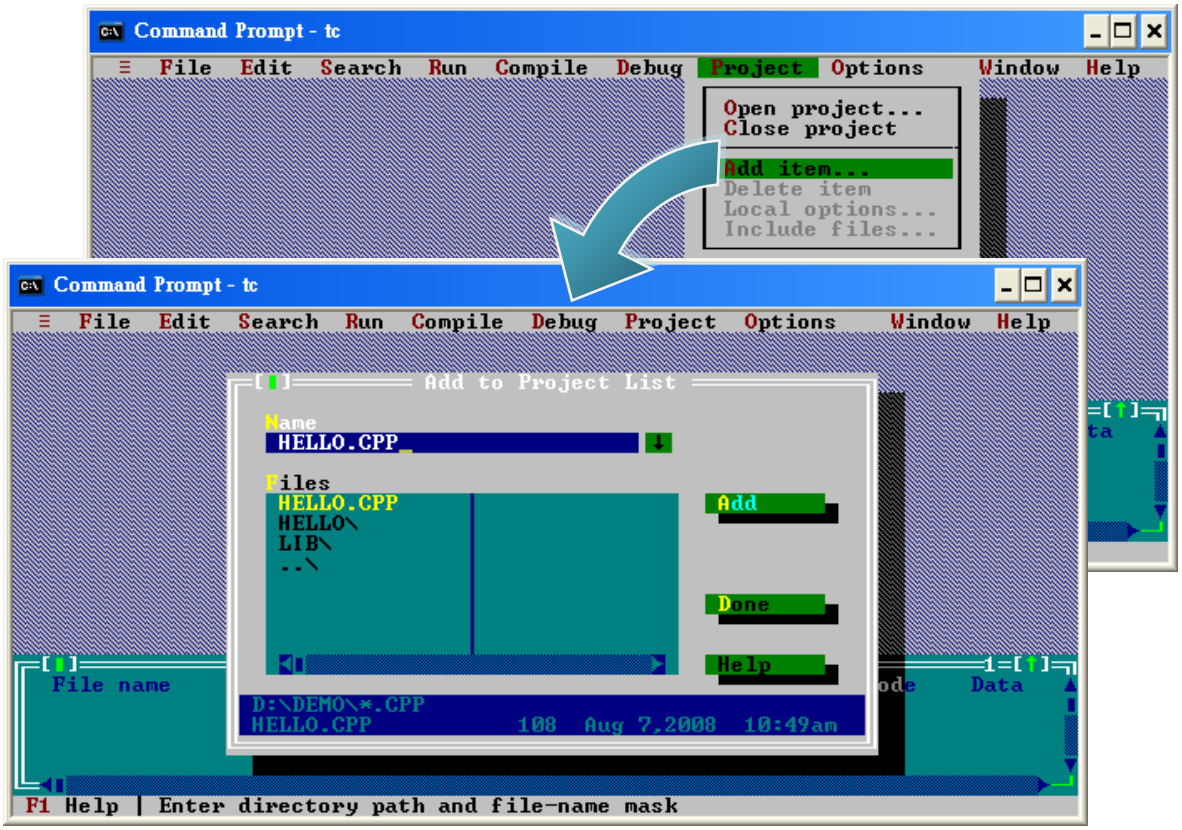
Step 7: Add the necessary function libraries to the project (*.lib)

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 (8000a.lib) and then click the “Add” button

iv. Select “Done” to exit

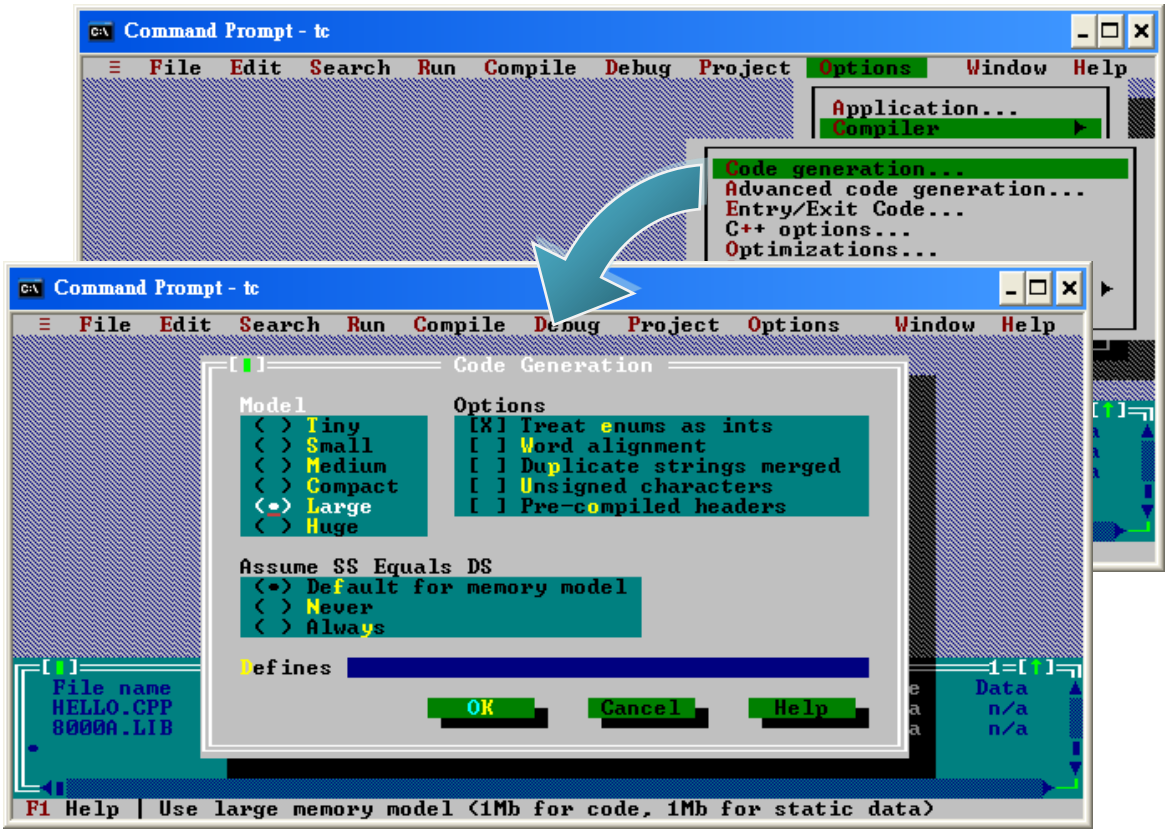


Step 8: Set the memory model to large

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

ii. On “Model” option, select “Large”

iii. Select “OK”



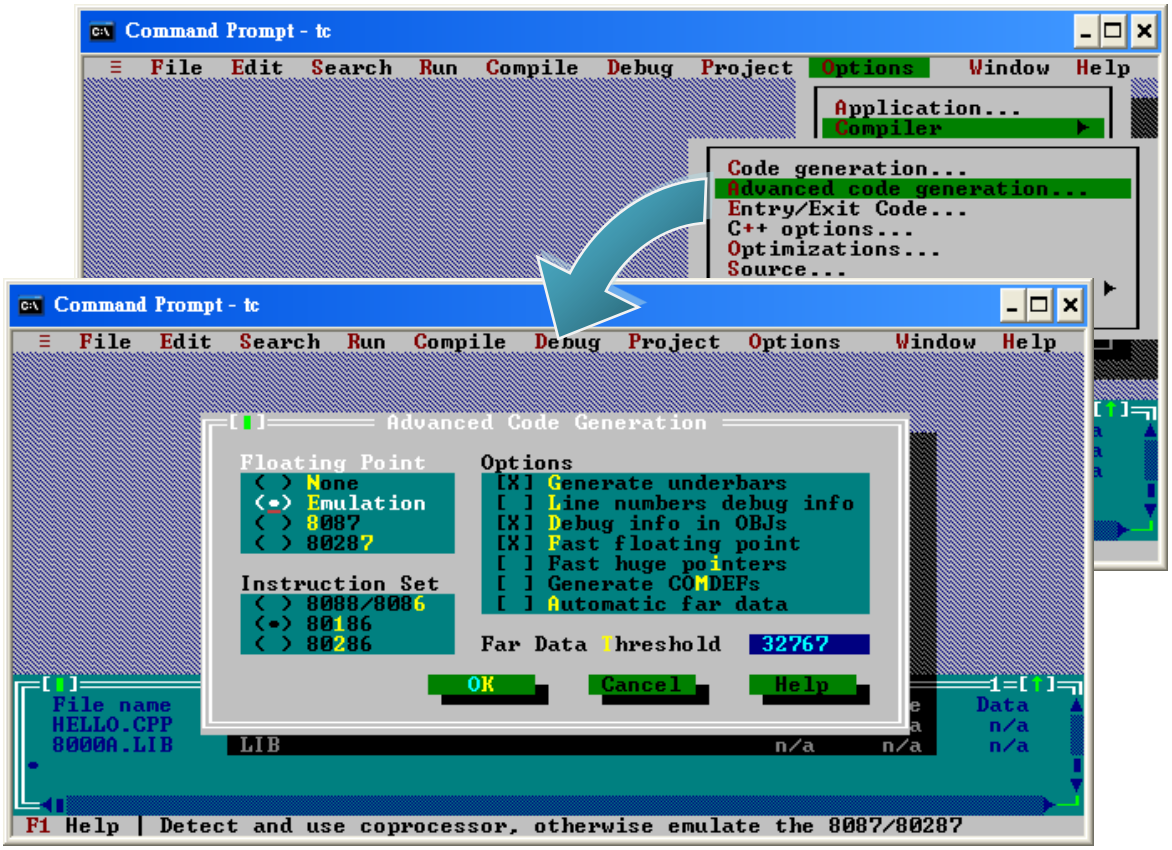
Step 9: Set the Floating Point to Emulation and the Instruction Set to 80186

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”



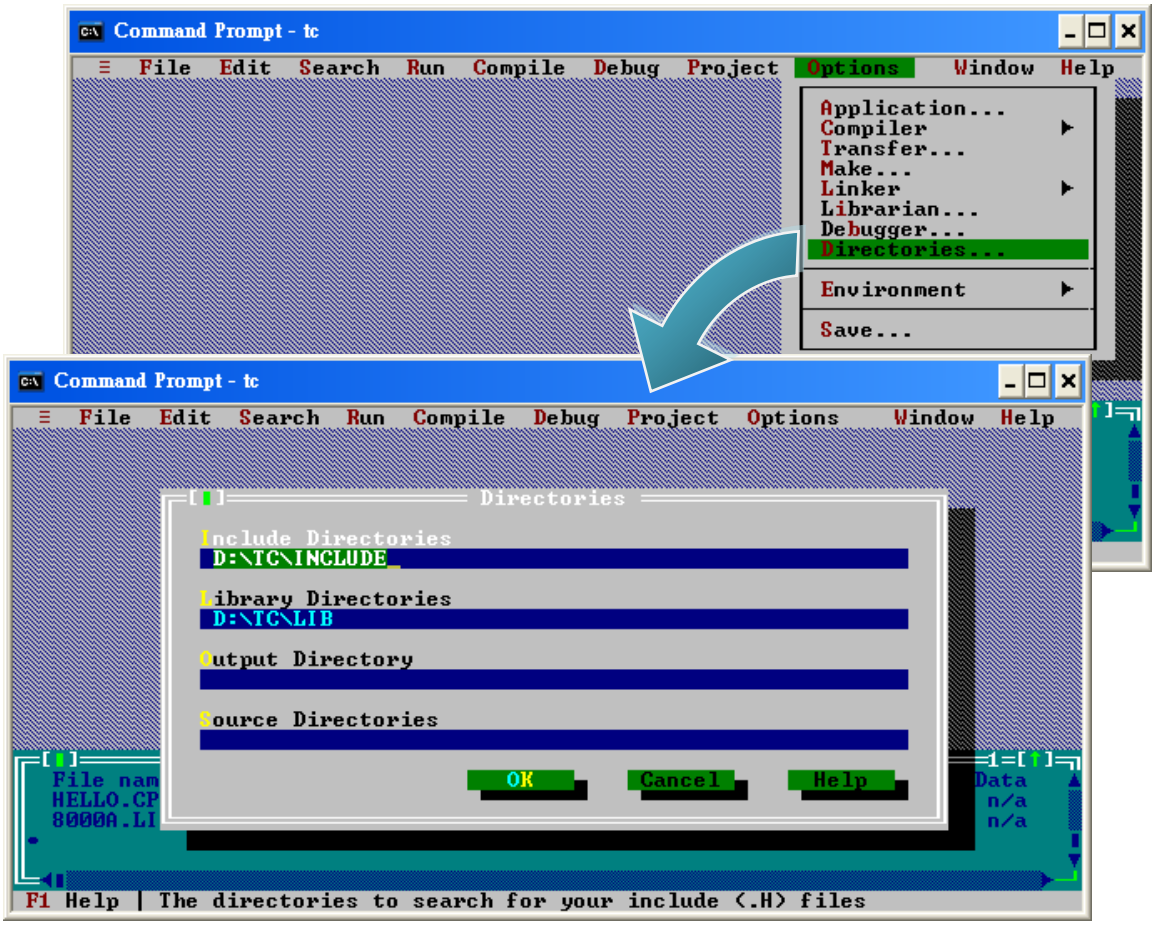
Step 10: Set the TC compiler include and library directories

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

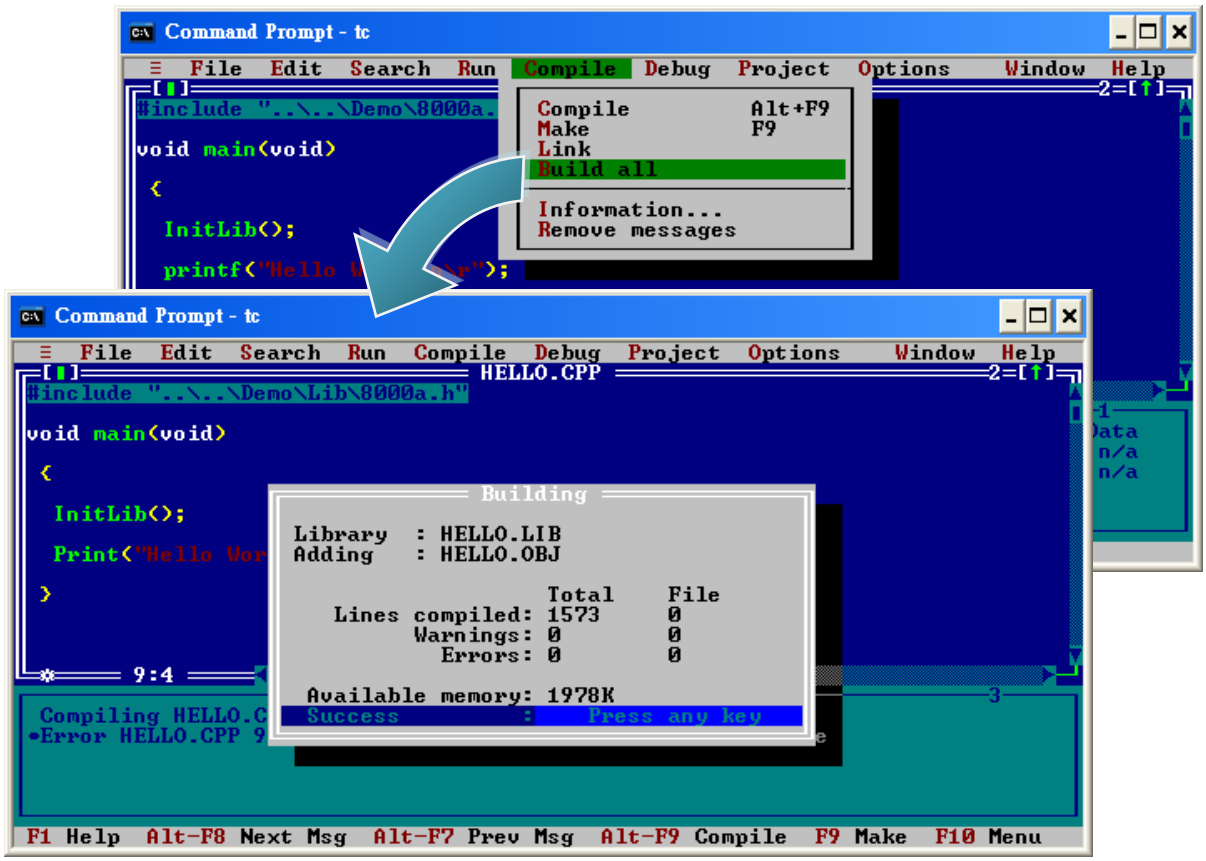
ii. On “Include Directories” option, specify the header file

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

iv. Select “OK”

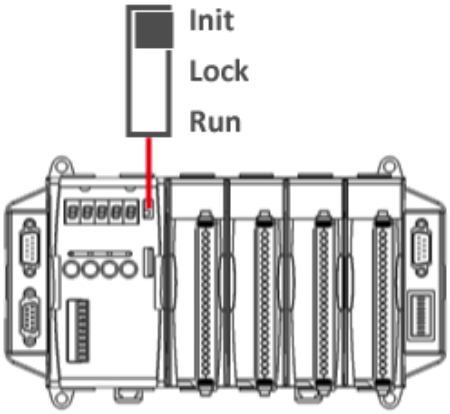


Step 11: Select “Build all” from the “Compile” menu to build the project

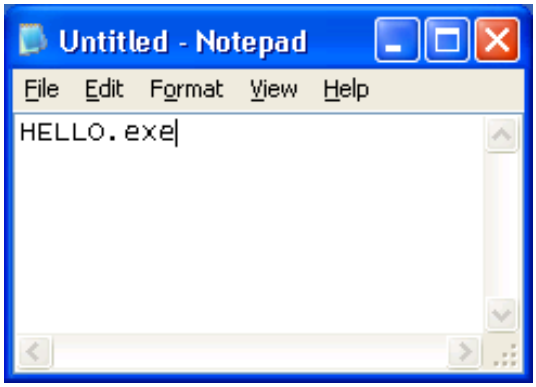


Step 12: Boot the iPAC-8000 into initial mode

Make sure the switch placed in the “Init” position.



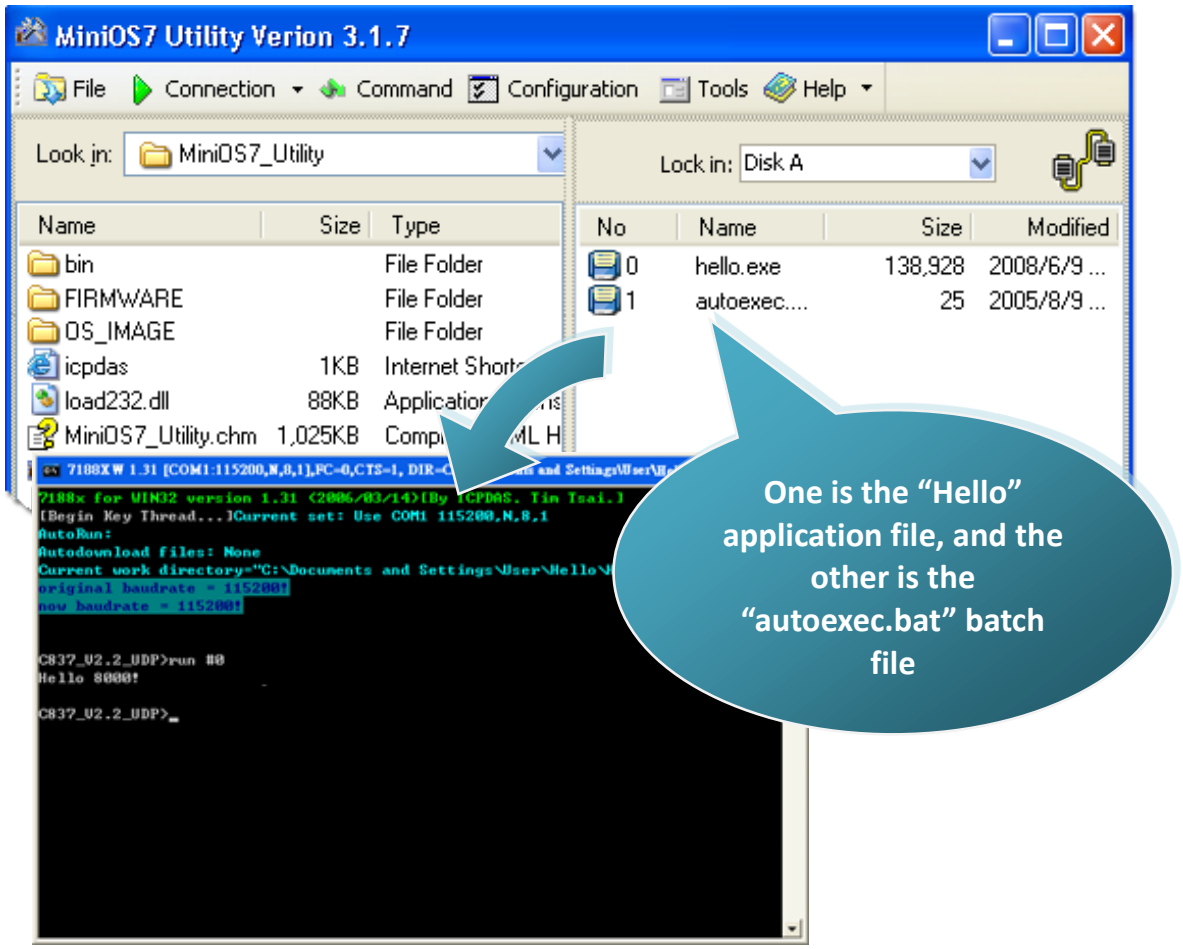
Step 13: Create an autoexec.bat file



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

Step 14: Upload programs to iPAC-8000 using MiniOS7 Utility

For more detailed information about this process, please refer to section “2.5.1. Establishing a connection”



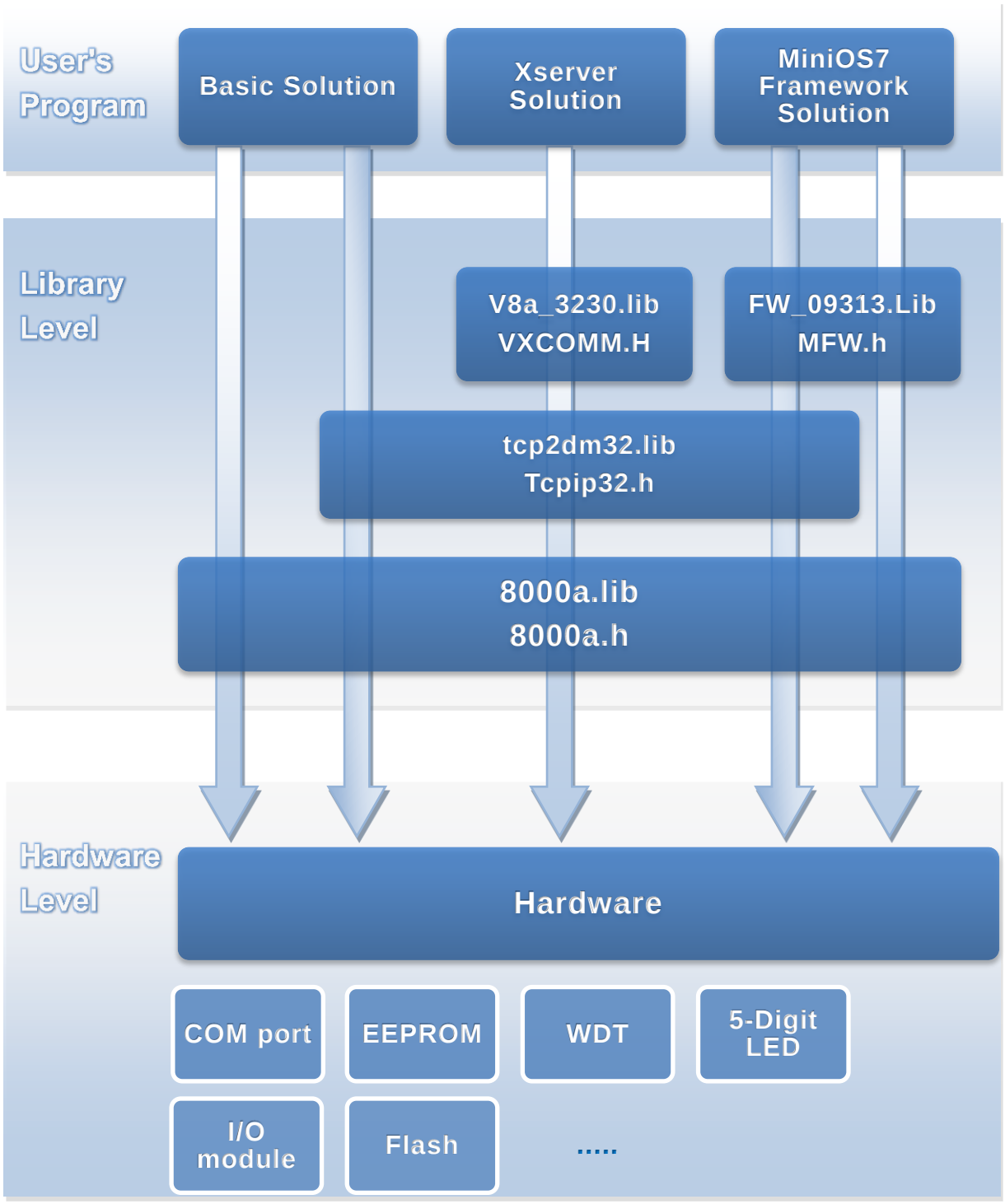
4. APIs and Demo References

There are several APIs and demo programs that have been designed for iPAC-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 iPAC-8000 APIs grouped by functional category.

Description	Header File	Library
CPU driver	8000a.h	8000a.lib
DCON driver	DCON_FUN.h	DCON_8KL.LIB
TCP/IP driver	Tcpip32.h	tcp2dm32.lib
Framework driver	MFW.h	MFW09313.LIB
Xserver driver	VXCOMM.H	V8a_3230.lib
microSD driver	microSD.h	SD_V100.LIB
Flash memory driver	MFS.h	MFS_V212.LIB

➤ System Structure



4.1. MiniOS7 API

MiniOS7 API is the core API set; it allows developers to access core iPAC-8000, it contains most of the methods and functions you would use to utilize iPAC-8000 in your application.

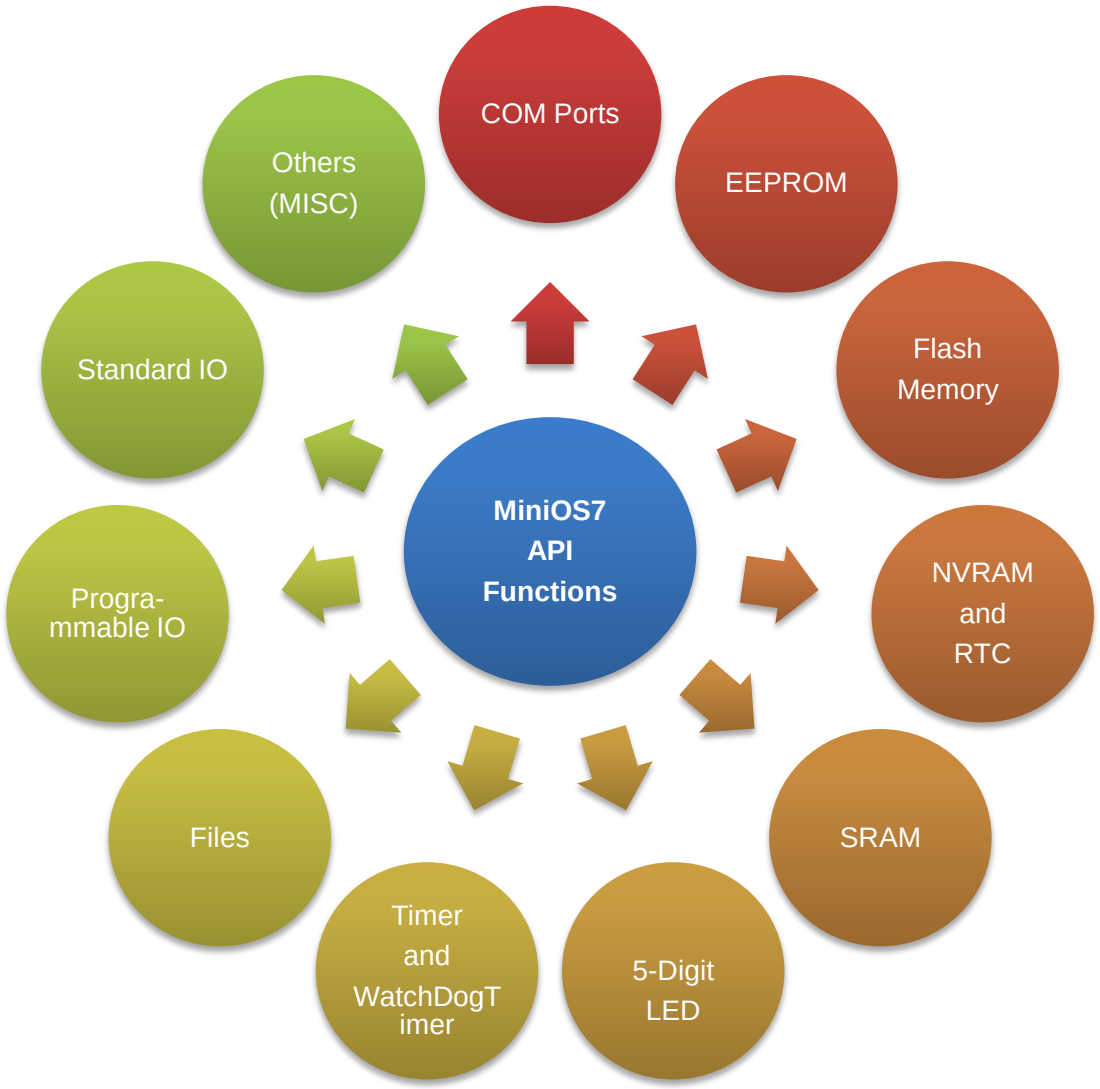
The following provides a brief introduction to the functions of MiniOS7 API.

Functions Library – 8000a.lib

This file contains the MiniOS7 API (Application Programming Interface) and has hundreds of pre-defined functions related to iPAC-8000

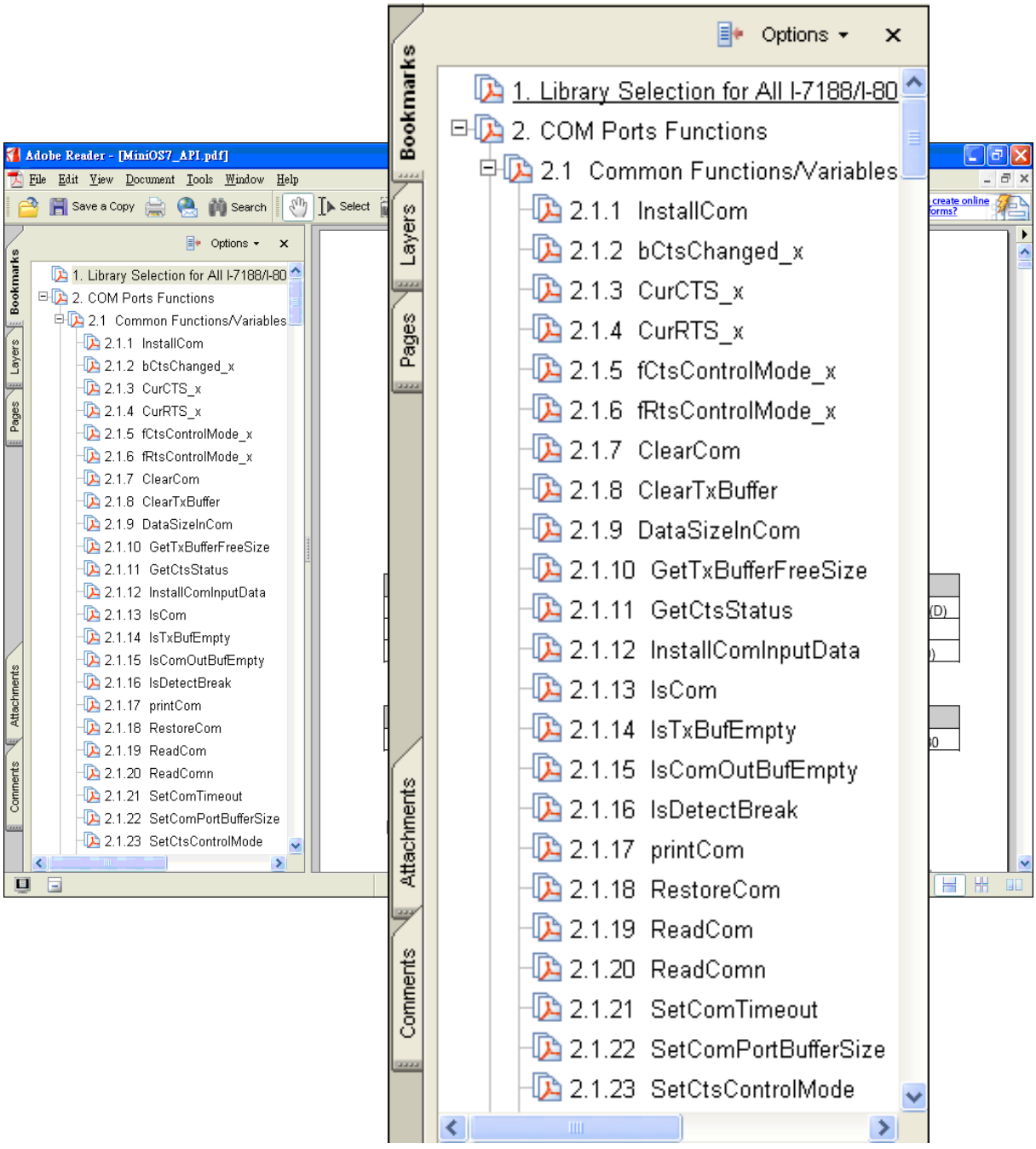
Header File – 8000a.h

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:

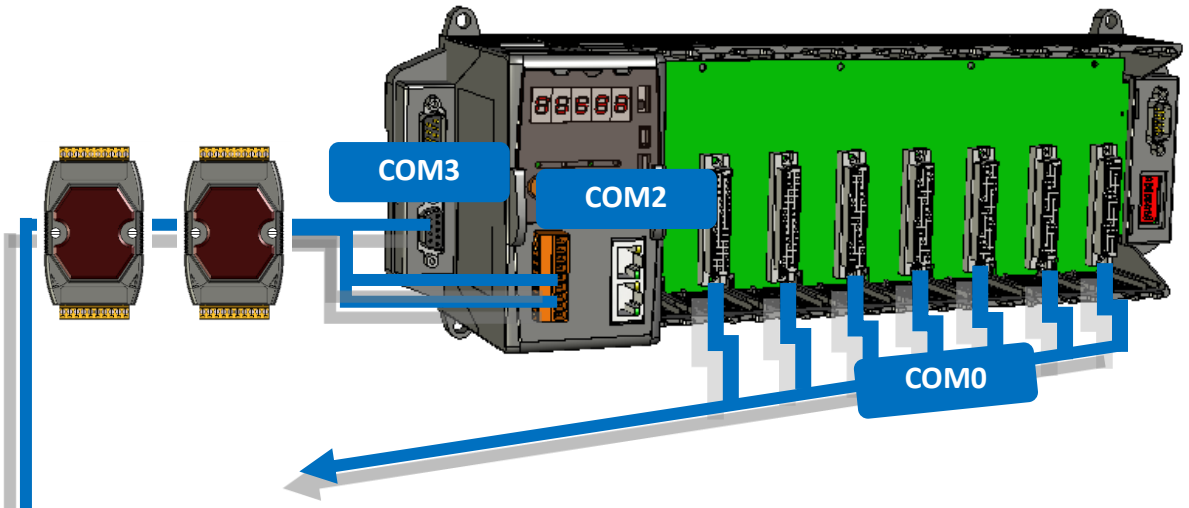
CD:\8000\Napdos\MiniOS7\Document\
http://ftp.icpdas.com/pub/cd/8000cd/napdos/minios7/document/



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

➢ Basic

Folder	Demo	Explanation
File	Config_1_Basic	Reads information from a text file (basic).
	Config_2_Advanced	Reads a config file (text file)(advanced).
Hello	Hello_C	Reads the library version and flash memory size.
	Hello_C++	
Misc	Reset	Resets the software.
	Runprog	Illustrates how to select an item and run it.
	Serial	Illustrates how to retrieve 64-bit hardware unique serial number.
	Watchdog	Enables the WDT or bypasses the enable WatchDog function.
Smmi	SystemKey	Shows how to operate the system key function simply and easily.
	Led	Shows how to control the red LED and 7-segment display.
Memory	Battery_Backup_SRAM	Shows how to read or write to the 256K/512K byte battery backup.
DateTime	DateTime	Shows how to read and write the date and time from the RTC.
Com port	C_Style_IO	(1) Shows how to write a function to input data. (2) Shows how to receive a string. (3) Shows how to use a C function: sscanf or just use Scanf()
	Receive	Receives data from COM port. Slv_COM.c is in non-blocked mode Receive.c is in blocked mode.
	Slv_COM	A slave COM Port demo for (request/reply) or (command/response) applications.
	ToCom_In_Out	Illustrates how to Read/Write byte data via COM Port.
For more information about these demo programs, please refer to: **CD:\8000\Napdos\iPAC8000\Demo\Basic** **http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/basic/**		



➢ 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

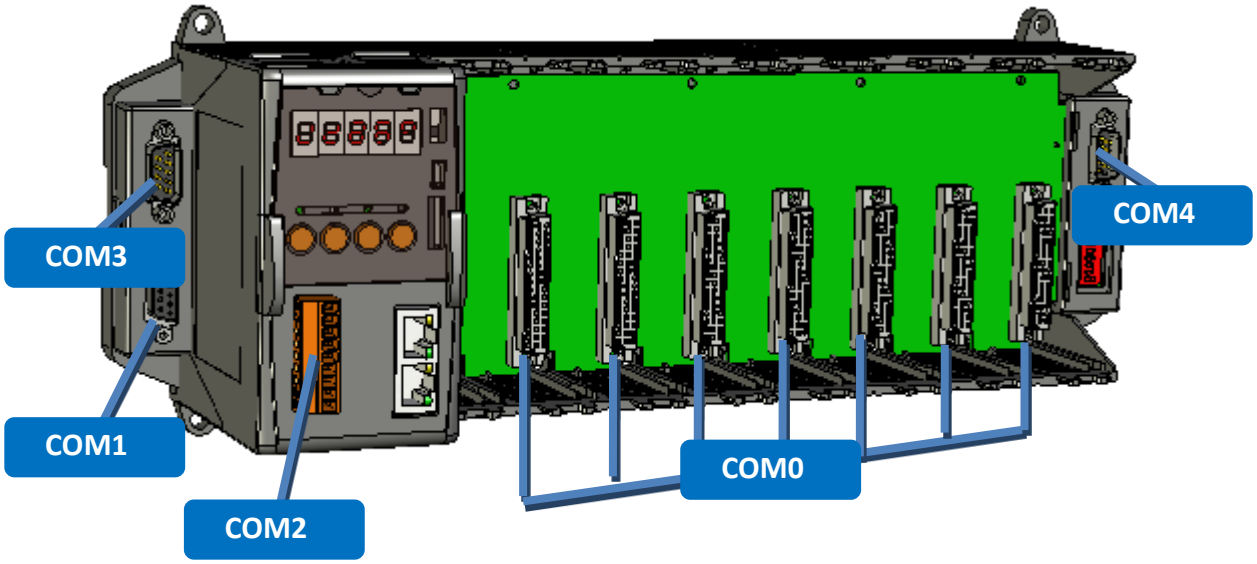
Folder	Demo	Explanation
7K 87K_for_Com	7K87K_DI_for_Com	"COM Port" can be used to connect and control I-7k or I-87k series modules. ■ For iPAC-8000 module and can use, COM2, COM3. ■ For iPAC-8000 module and (CPU 40 and 80M) can use, COM3, COM4.
	7K87K_DO_for_Com	
	7K87K_AI_for_Com	
	AO_22_26_for_Com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

For more information about these demo programs, please refer to:

CD:\8000\Napdos\iPAC8000\Demo\Basic\iP-84x1_iP-88x1\7K87K_for_COM\
http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/basic/ip-84x1_ip-88x1/7k87k_for_com/

4.1.1. MiniOS7 API for COM Port

The iPAC-8000 provides five built-in COM ports.



4.1.1.1. 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.

The table below summarizes 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() Putch()	Getch()	Print()

4.1.1.2. API for MiniOS7 COM port

API for using COM ports

1. InstallCom()

Before any COM Port can be used, the driver must be initiated 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 “8000a.h”

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

 InitLib();      /* Initiate the 8000a library */
 InstallCom(1, 115200, 8, 0, 1);      /* Initiate the COM1 */

 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 */
}
```

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 “8000a.h”

void main(void)
{
 int i;

 /* Initiate the 8000a library */
 InitLib();
 InstallCom(1, 115200, 8, 0, 1);      /* Initiate the COM1 */
 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);    /* Release the COM1 */
}
```

4.1.1.3. API for standard COM port

The standard COM port is used to upload program from PC to the iPAC-8000.

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 “8000a.h”

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

 InitLib();      /* Initiate the 8000a 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 “8000a.h”

void main(void)
{
 int i;

 /* Initiate the 8000a library */
 InitLib();
 for(i=0;i<10;i++)
 {
  Print(“Test %d\n\r”,i);
 }
}
```

4.1.1.4. Port functions Comparison

For example, learning to show the ASCII code.

MiniOS7 COM port functions	Standard COM port functions

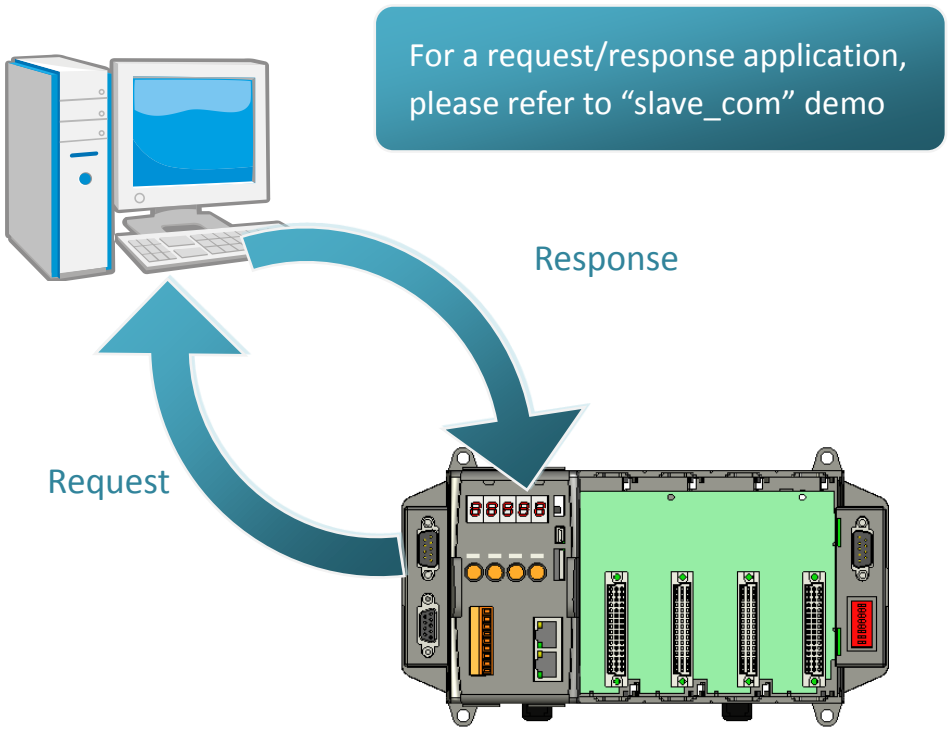
#include<stdio.h>
#include “8000a.h”
 	
void main(void)
{
unsigned char item;
 	
InitLib();
InstallCom(1, 115200, 8, 0, 1);
printCom(1,”Hits any key.\n”);
printCom(1,”Hit the ESC to exit!\n”);
 	
for(;;)
{
if(IsCom(1))
{
item=ReadCom(1);
if(item==’q’)
{
return;
}
else
{
printCom(1,”----------\n\r”);
printCom(1,”char:”);
	#include<stdio.h>
	#include “8000a.h”
	
	void main(void)
	{
	unsigned char item;
	
	InitLib();
	
	Print("Hits any key.\n");
	Print("Hits the ESC to exit !\n");
	
	for(;;)
	{
	if(kbhit())
	{
	item=Getch();
	if(item==’q’)
	{
	return;
	}
	else
	{
	Print(”----------\n\r”);
	Print(“char:“);

```
    ToCom(1,item);
    printCom(1,"\n\rASCII(%c)\n\r",item);
    printCom(1,“Hex(%02X)\n\r”,item);
   }
  }
 }
 Delay(10);
 RestoreCom(1);
}
```

```
    Putch(item);
    Print("\n\rASCII(%c)\n\r",item);
    Print(“Hex(%02X)\n\r”,item);
   }
  }
 }


}
```

4.1.1.5. Request/Response protocol define on COM port

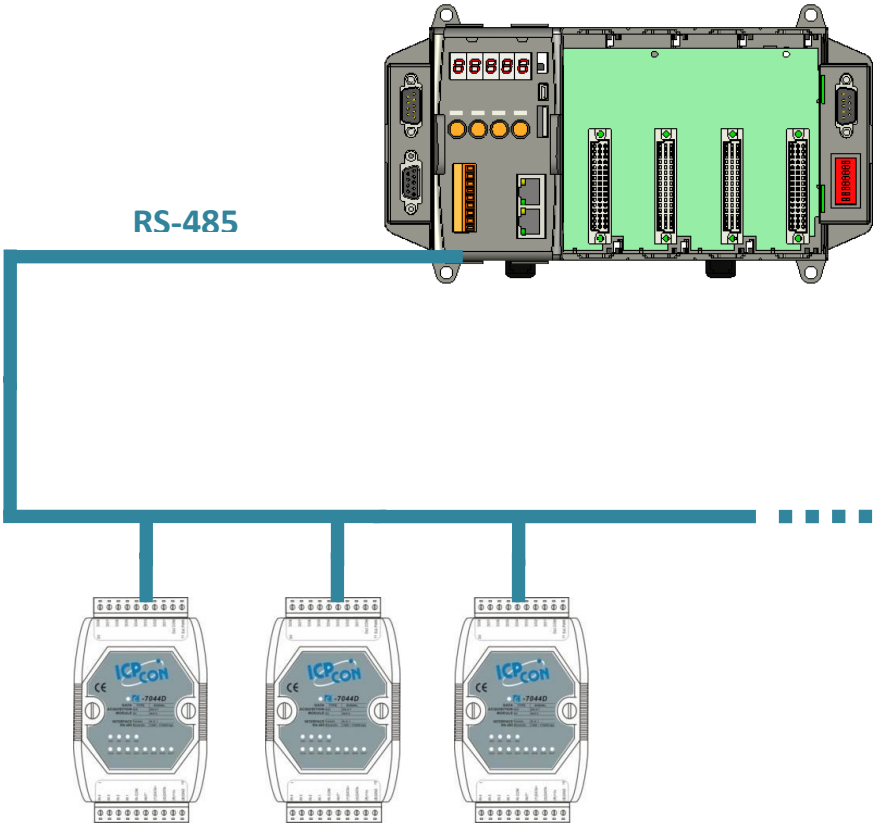
Request/Response communication is very typical protocol architecture. If you want to design a command set of communication protocol as table below, you can refer to “slave_com” demo.



Request	Response
c1	Debug information: Command1 Command1
c2	Debug information: Command2 Command2
Q	Debug information: Quick program
Other command	Debug information: Unknown command

4.1.2. MiniOS7 API for I/O Modules

The iPAC-8000 equip 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 to communicate 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.

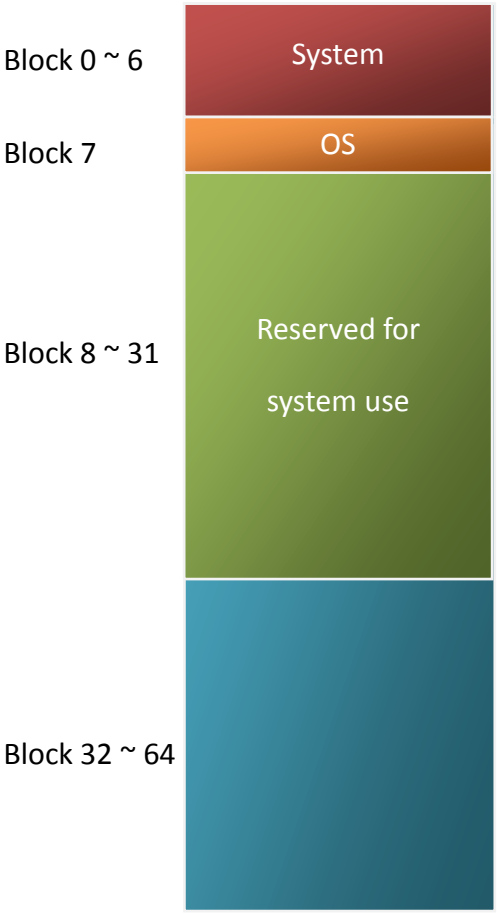
Step 4: Use RestoreCom() to restore the COM port driver

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

```
#include <stdio.h>
#include "8000a.h"

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

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

 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 */
}
```

4.1.3. MiniOS7 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 the EEPROM

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 “8000a.h”

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

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

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

4.1.4. MiniOS7 API for Flash Memory

- The iPAC-8000 module 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 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 writing data to 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 “8000a.h”

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

InitLib();      /* Initiate the 8000a 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;
}
```

4.1.5. MiniOS7 API for NVRAM

- The iPAC-8000 equip 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.

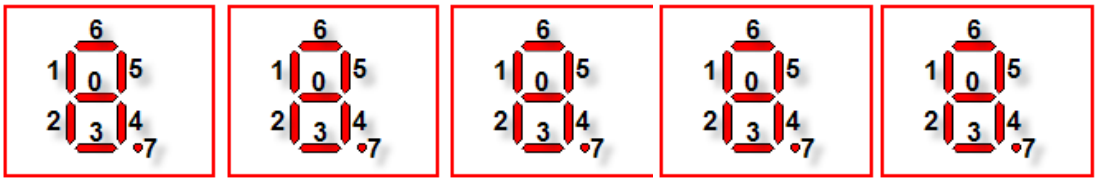
```
#include <stdio.h>
#include “8000a.h”

void main(void)
{
 int data=0x55, data2;
 InitLib();      /* Initiate the 8000a 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 “8000a.h”

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

InitLib();      /* Initiate the 8000a 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 */
}
```

4.1.6. MiniOS7 API for 5-Digital LED

The iPAC-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 “8000a.h”

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

}
```

4.1.7. MiniOS7 API for Timer

- The iPAC-8000 can support a single main time tick, 8 stop watch timers and 8 counts down timers.
- The iPAC-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 “8000a.h”

void main(void)
{
 Unsigned long time iTime;

 InitLib();      /* Initiate the 8000a 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 iPAC-8000 timer function */
}
```

4.1.8. MiniOS7 API for WatchDog Timer (WDT)

- The iPAC-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 “8000a.h”

void main(void)
{
 Unsigned long time iTime;

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

4.2. microSD API

Required library and header files:
SD_V102.LIB and microSD.h

The iPAC-8000 series can support one microSD card and the size can be 1GB or 2 GB.

- Summarize of the microSD functions:

Function	Description
pc_init	Initializes the microSD socket library
pc_open	1. Open an existing file and return a file handle 2. Creates a new file.
pc_close	Closes a file and release a file handle.
pc_read	Reads the specified file
pc_write	Writes the specified file
pc_seek	Moves the file pointer to relative offset from the current offset
pc_tell	Gets current offset of the file pointer
pc_eof	Checks whether the end-of-file is reached
pc_format	Formats the microSD card as FAT (FAT16)
pc_mkdir	Creates a directory or subdirectory
pc_rmdir	Removes an existing directory
pc_move	Renames an existing file or a directory, including the subdirectory
pc_del	Deletes the specified file
pc_deltree	Deletes the specified directory or subdirectory
pc_isdir	Checks whether the file is a directory
pc_isvol	Checks if is a volume
pc_size	Gets the size of the specified file
pc_set_cwd	Sets the current working directory
pc_get_cwd	Gets the pathname of the current working directory
pc_gfirst	Moves the pointer to the first element

Function	Description
pc_gnext	Moves the pointer to the next element
pc_gdone	Moves the pointer to the last element
pc_get_freeSize_KB	Gets the free space of the SD memory card
pc_get_usedSize_KB	Gets the used space of the SD memory card
pc_get_totalSize_KB	Gets the total size of the SD memory card
pc_get_attributes	Gets the file attributes
pc_set_attributes	Sets the file attributes
pc_get_errno	Gets the error number

API for starting microSD

1. pc_ Init()

Before using any miscroSD functions, PC_Init() must be called to initialize the microSD.

API for enabling/disabling microSD

3. pc_open()

Before writing/reading data to/from the microSD card, PC_open() must be called to open the file.

4. pc_close()

After the data has finished being written/read to/from the microSD, PC_close() must be called to close the file with a file handle.

API for writing data to the microSD

5. pc_write()

After using PC_open() to open the file, PC_write() must be called to read data from the microSD.

For example, writing data to the microSD

```
#include <string.h>
#include <stdio.h>
#include “8000a.h”
#include “microSD.h”
{
  int fd, iRet;

  InitLib();
  If(pc_init())
  {
    Print(“Init microSD ok\n\r”);
  }
  else
  {
    Print(“Init microSD failed\n\r”);
    iRet=pc_get_errno();
    switch(iRet)
    {
      case PCERR_BAD_FORMAT:      //1
          Print("Error 01: format is not FAT\n\r");
          break;
      case PCERR_NO_CARD:      //2
          Print("Error 02: no microSD card\n\r");
          break;
      default:
          Print("Error %02d: unknow error\n\r",iRet);
    }
  }
```

```
fd=pc_open("test.txt",(word)(PO_WRONLY|PO_CREAT|PO_APPEND),(word)(PS_I
WRITE|PS_IREAD));
  if(fd>=0)
  {
    pc_write(fd,"1234567890",10);      //write 10 bytes
    pc_close(fd);
  }
}
```

API for reading data from the microSD

6. pc_read()

After using PC_open() to open the file, PC_read() must be called to read data from the microSD.

For example, reading data from the microSD:

```
#include <string.h>
#include <stdio.h>
#include “8000a.h”
#include “microSD.h”
{
  int fd, iRet;
  unsigned char Buffer[80];

  InitLib();
  If(pc_init())
  {
    Print(“Init microSD ok\n\r”);
  }
  else
  {
    Print(“Init microSD failed\n\r”);
    iRet=pc_get_errno();
    switch(iRet)
    {
      case PCERR_BAD_FORMAT:      //1
          Print("Error 01: format is not FAT\n\r");
          break;
      case PCERR_NO_CARD:      //2
          Print("Error 02: no microSD card\n\r");
          break;
      default:
          Print("Error %02d: unknow error\n\r",iRet);
```

```
    }
  }
  fd=pc_open("test.txt",(word)(PO_RDONLY),(word)(PS_IWRITE|PS_IREAD));
  if(fd>=0)
  {
    iRet=pc_read(fd,Buffer,10);      //reads 10 bytes
    Buffer[10]=0;      //adds zero end to the end of the string.
    pc_close(fd);
    Print("%s",Buffer);
  }
}
```

For more demo program about the microSD, please refer to:

CD:\8000\Napdos\iPAC8000\Demo\Basic\iP-84x1_iP-88x1\microsd\

http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/basic/ip-84x1_ip-88x1/microsd/

4.3. MFS API (For iPAC-8000-FD series only)

Required library and header files:
MFS_V212.LIB and MFS.h

The iPAC-8000-FD series products equip an extra 64MB flash memory, the MFS is designed to read/write file from/to the 64MB flash memory.

For full usage information regarding the hardware supported, applications, and the specification, please refer to section “Appendix C. What is MiniOS7 File System (MFS)”

- Summarize of the MFS functions:

Function	Description
mfs_Init	Initialize the file system.
mfs_Stop	Allocated buffers are freed upon closing.
mfs_ResetFlash	Initialize the file system. All files will lose.
mfs_X600Fs_GetLibVersion	Gets the version number of function library.
mfs_GetLibDate	Gets the create date of function library.
mfs_GetFileNo	Gets the total number of files stored in the NAND Flash.
mfs_GetFreeSize	Gets the size of available space that can be used to append file.
mfs_GetBadSize	Gets the size of non-available space.
mfs_GetUsedSize	Gets the size of used space.
mfs_GetFileSize	Gets the size of file stored in the NAND Flash.
mfs_GetFileInfoByName	Uses the specified filename to retrieve file information.
mfs_GetFileInfoByNo	Uses the file number index to retrieve file information.
mfs_DeleteAllFiles	Delete all files stored in the NAND Flash.
mfs_DeleteFile	Delete one selected file that has been written to the NAND Flash.

Function	Description
mfs_OpenFile	1. Opens a file with a file name. 2. Creates a new file.
mfs_CloseFile	Closes a file with a file handle. All buffers associated with the stream are flushed before closing.
mfs_ReadFile	Reads specified bytes of data from a file.
mfs_WriteFile	Appends specified bytes of data to a file.
mfs_Getc	Gets a character from a file.
mfs_Putc	Outputs a character data to the file.
mfs_Gets	Gets a string from a file.
mfs_Puts	Outs a string a file.
mfs_EOF	Macro that tests if end-of-file has been reached on a file.
mfs_Seek	Repositions the file pointer of a file.
mfs_Tell	Returns the current file pointer.
mfs_EnableWriteVerify	Enable the data verification. By default, the data verification is enabling.
mfs_DisableWriteVerify	Disable the data verification.

5. iPAC-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.

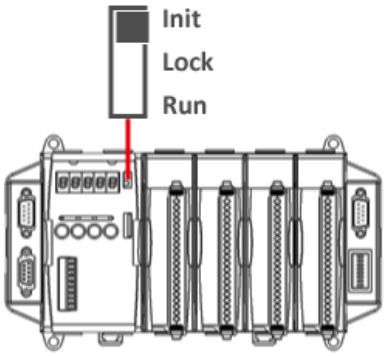
Both the OS updates services and SDK updates services can be found separately on the CD that was provided with the package or by downloading the latest version from ICP DAS web site.

5.1. Updating the OS image

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 iPAC-8000 into Initial mode

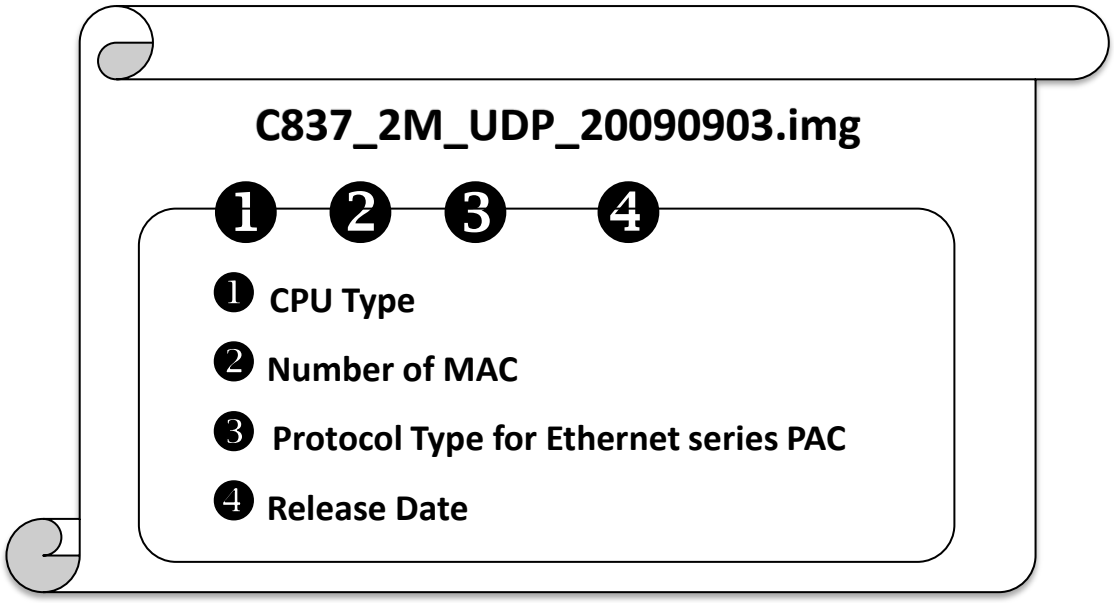
Make sure the switch placed in the "Init" position



Step 2: Get the latest version of the iPAC-8000 OS image

The OS image of Serial series PAC and Ethernet series PAC are the same. The iPAC-8000 OS image can be found separately on the CD that was provided with the package or by downloading the latest version from ICP DAS web site.

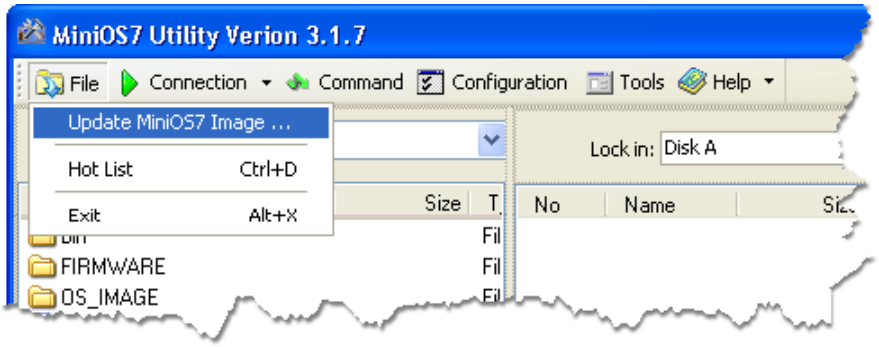
CD:\8000\Napdos\iPAC8000\OS_Image\iP-84x1_iP-88x1\
http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/os_image/ip-84x1_ip-88x1/



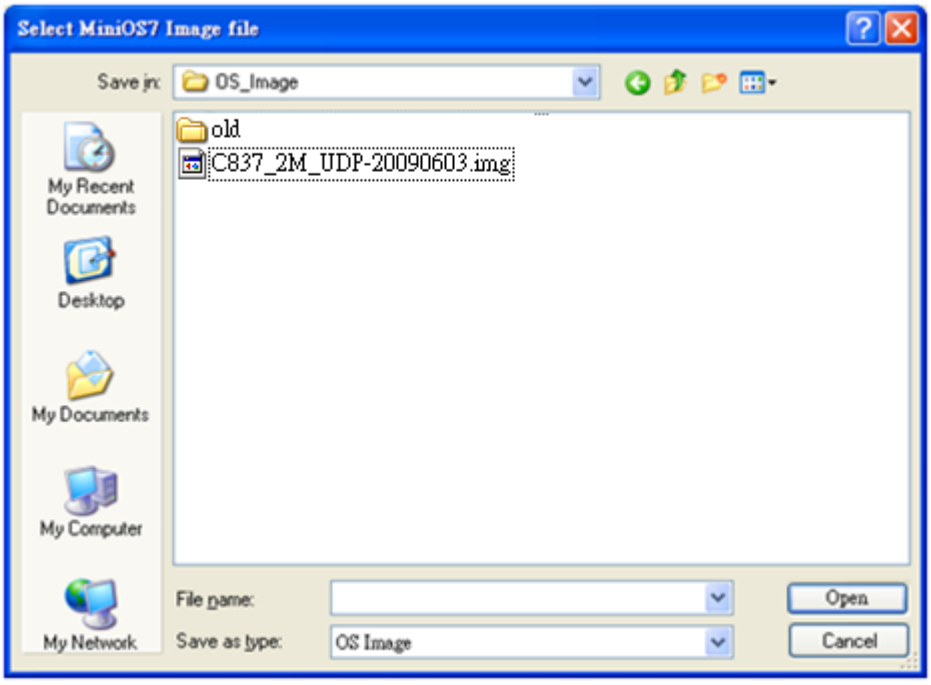
Step 3: Establish a connection

For more detailed information about this process, see section “2.5.1. Establishing a connection”

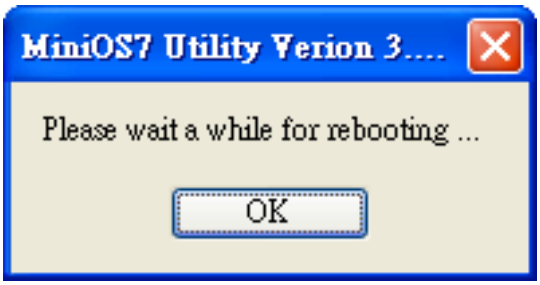
Step 4: Click the “Update MiniOS7 Image …” from the “File” menu



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



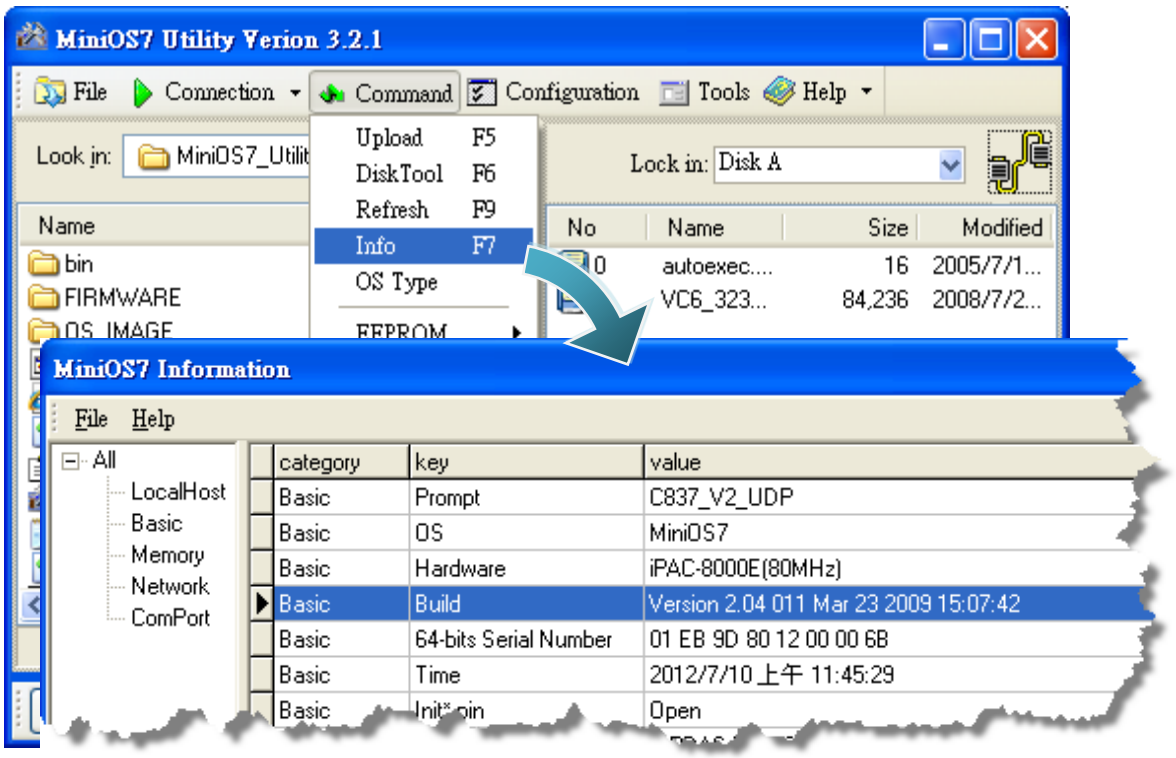
Step 6: Click “OK” button



Step 7: Reboot the MiniOS7 Utility and then establish a connection

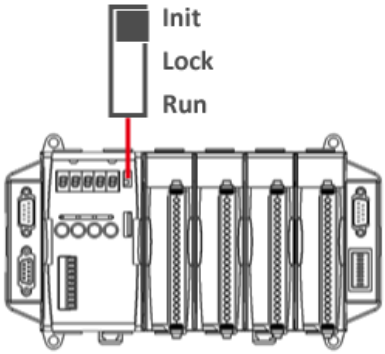
For more detailed information about this process, please refer to section “2.5.1. Establishing a connection”

Step 8: Click the “Info” from the “Command” menu to check the version of the OS image



5.2. Updating the Firmware

The firmware is stored in flash memory and can be updated to add even more useful features and support more protocols.



Step 1: Boot the iPAC-8000 into Initial mode

Make sure the switch placed in the “Init” position

Step 2: Get the latest version of the firmware

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

CD:\8000\Napdos\iPAC8000\Firmware\
http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/firmware/

Step 3: Establish a connection between PC and iPAC-8000

For more detailed information about this process, see section “2.5.1. Establishing a connection”

Step 4: upload and run the iPAC-8000 firmware

For more detailed information about this process, please refer to section “2.5.2. Uploading and Executing iPAC-8000 programs”

Appendix A. What is MiniOS7?

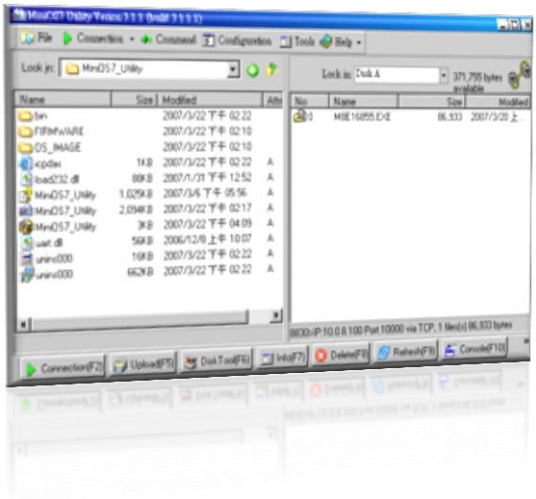
MiniOS7 is an embedded operating system design by ICP DAS. It is functionally equivalent to other brands of DOS, and can run programs that are executable under a standard DOS.

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?



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

- Supported connection ways
  1. COM port connection (RS-232)
  2. Ethernet connection (TCP & UDP)
     (Supported since version 3.1.1)
- Maintenance
  1. Upload file(s)
  2. Delete file(s)
  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.

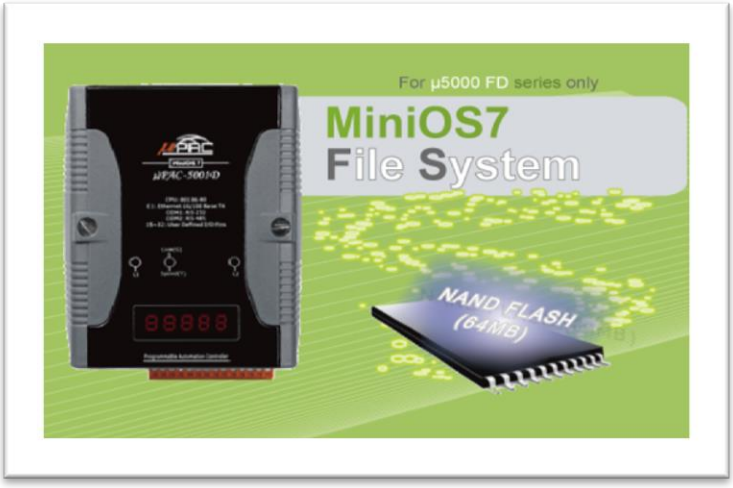
Including frequently used tools

a. 7188XW
b. 7188EU
c. 7188E
---
d. SendTCP
e. Send232
f. VxComm Utility

Upload location:

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

Appendix C. What is MiniOS7 File System (MFS)?



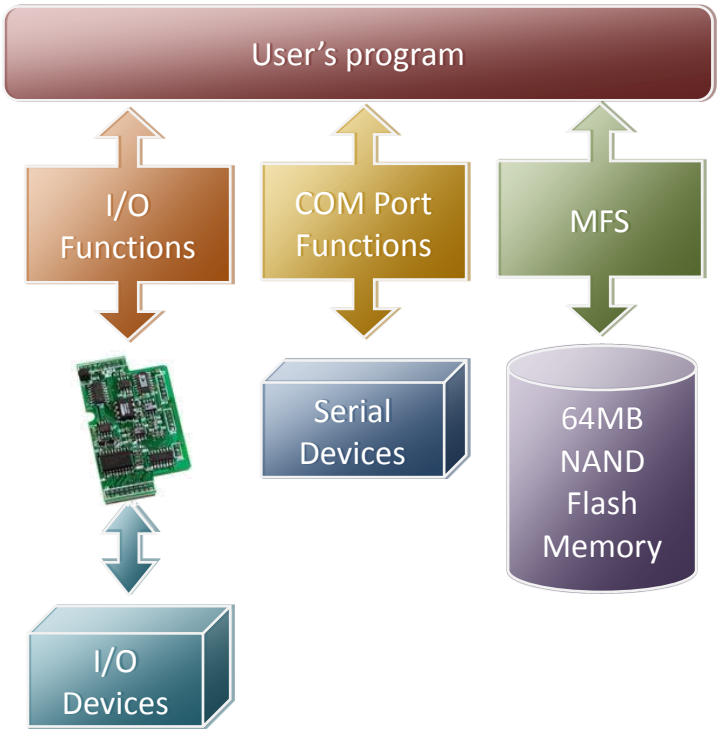
MiniOS7 file system, MFS, offers a rugged alternative to mechanical storage systems. Designed for the 64MB NAND flash memory, MFS implements a reliable file system with C language API for embedded data logger applications on MiniOS7.

Hardware Supported

iPAC-8000-FD (With 64MB Flash Memory), NVRAM: all of the 31 bytes.

Applications

Log data with timestamp, Log data and forward via the Ethernet



MFS Specifications

Item	Description
Disk size	1/2 size of the flash memory size
File number	456 files max. for each disk
File size	Disk Size max. for each file
File name	12 bytes max (case sensitive)
File operation modes	1. Read only 2. Write only: Creates a new file to write data, or overwrite a file (if the file is already exit). 3. Append: appends data to a file.
File handle	10 max for each disk. For read mode: the 10 file handles can all be used for reading operation on each disk. Total 20 files can be opened for reading mode. For write and append mode: only 1 file handle can be used for writing operation on all disks.
Writing verification	Yes. Default is enabled. Calling mfs_EnableWriteVerification and mfs_DisableWriteVerification can change the setting.
Automate file system recovery	Yes. If an unexpected reset or power loss occurs, closed files, and files opened for reading are never at risk. Only data written since the last writing operation (mfs_WriteFile,) might be lost. When the file system reboots, it restores the file system to its state at the time of the last writing operation.
Writing speed	mfs_WriteFile: 147.5 KB/Sec (verification enabled) (default) 244.0 KB/Sec (verification disabled) mfs_Puts: 142.1 KB/Sec (verification enabled) (default) 229.5 KB/Sec (verification disabled)
Reading speed	mfs_ReadFile: 734.7 KB/Sec mfs_Gets: 414.2 KB/Sec
Max. length of writing data	32767 bytes.

Max. length of reading data	32767 bytes.

Resources upload

- **MFS SDKs:**

http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/basic/ip-84x1_ip-88x1/lib/

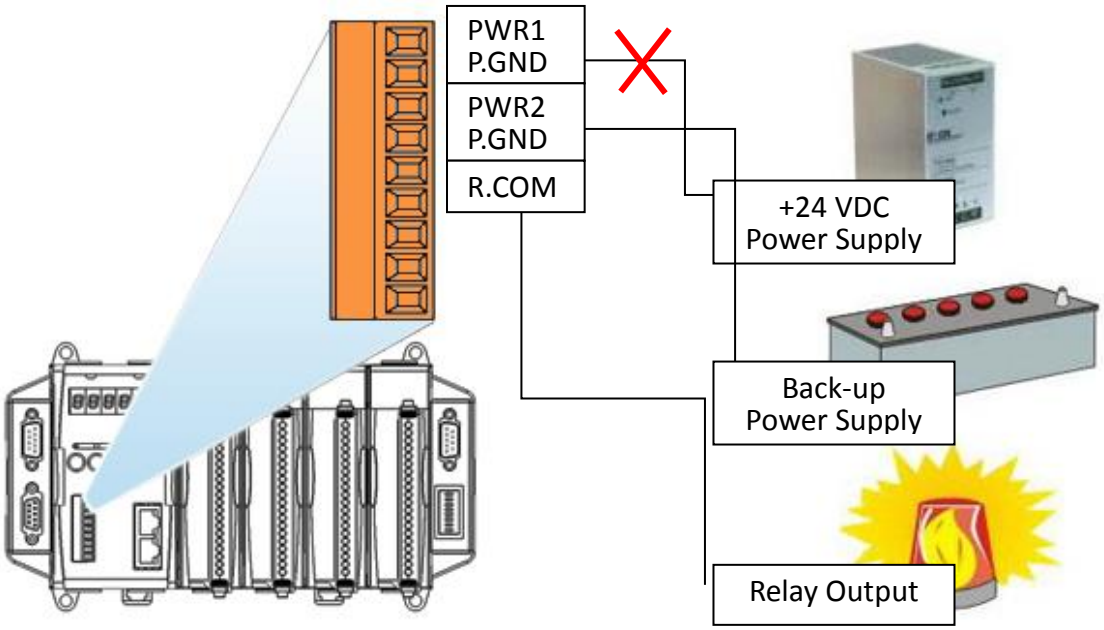
- **MFS Demos:**

http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/basic/ip-84x1_ip-88x1/256mb_flash/

Appendix D. Redundant Power

The iPAC-8000 provides two power inputs that can be connected simultaneously to live DC power sources. If one of the power inputs fails, the other live source acts as a backup to automatically support the power of iPAC-8000 needs.

The iPAC-8000 provides relay contact outputs to warn technicians on the shop floor when the power fails.



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

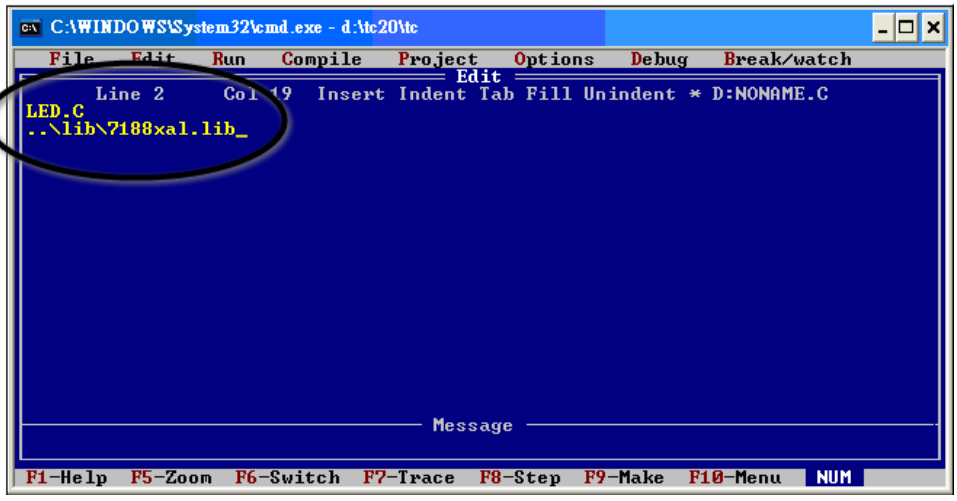
CD:\ APDOS\iPAC8000\Demo\Basic\iP-84x1_iP-88x1\Hello\Hello_C\gotc.bat
tcc -Ic:\tc\include -Lc:\tc\lib hello.c ..\..\Demo\basic\ iP-84x1_iP-88x1\lib\8000a.lib

2: Using the TC Integrated Environment

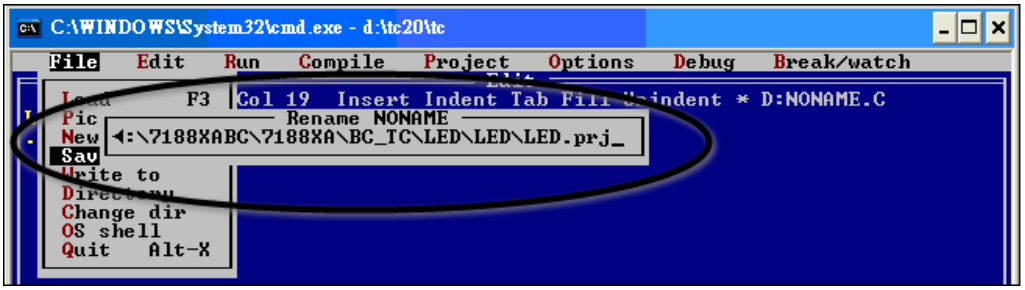
Step 1: Executing the TC 2.01

Step 2: Editing the Project file

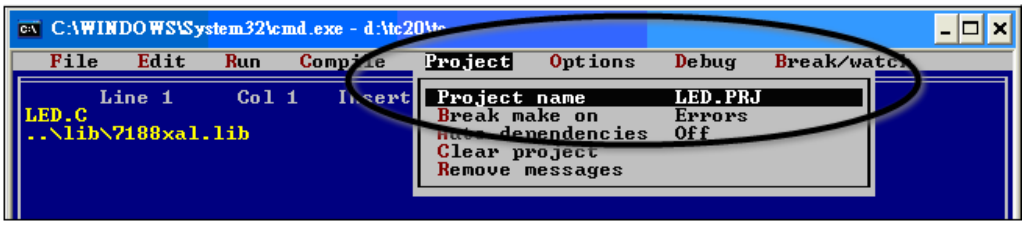
Adding the necessary library and file to the project



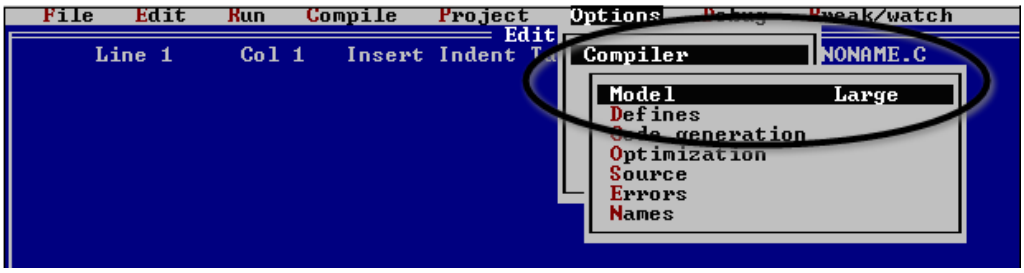
Step 3: Save the project and entering a name, such as LED.prj

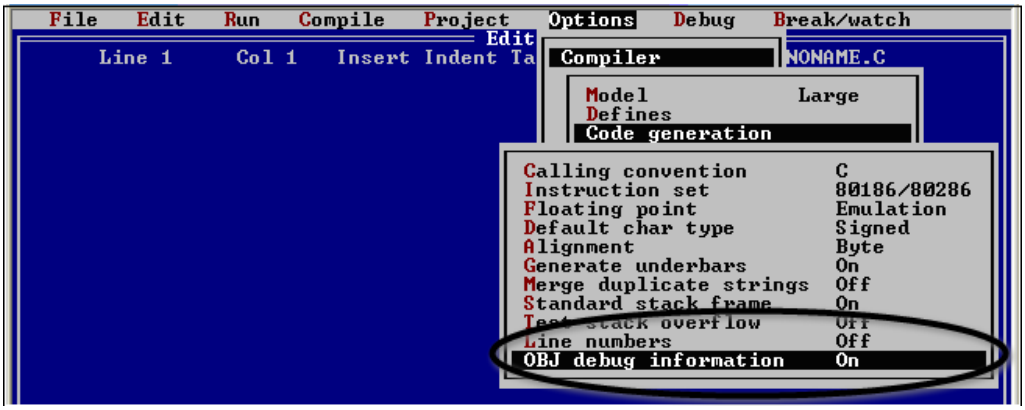


Step 4: Load the Project

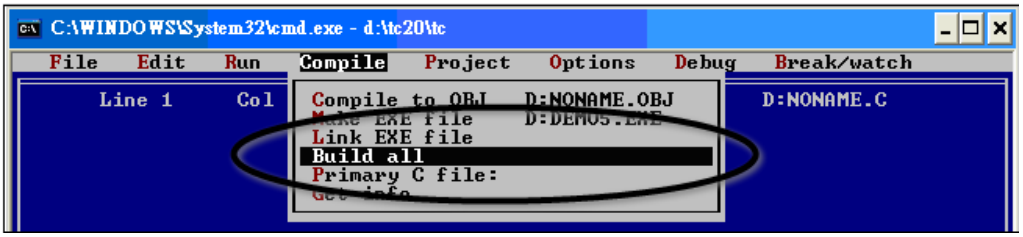


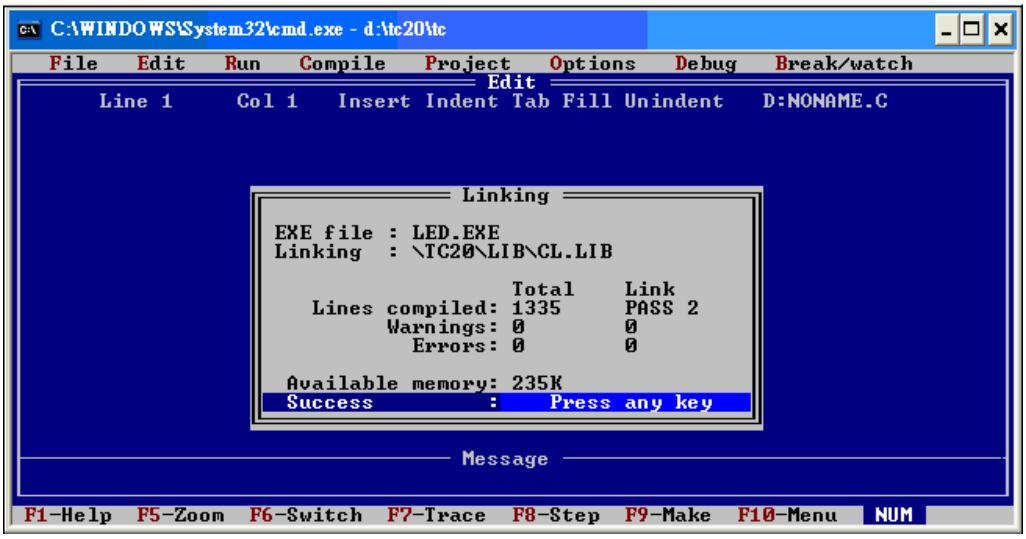
Step 5: Change the Memory model (Large for iPAC-8000.lib) and set the Code Generation to 80186/80286





Step 6: Building the project

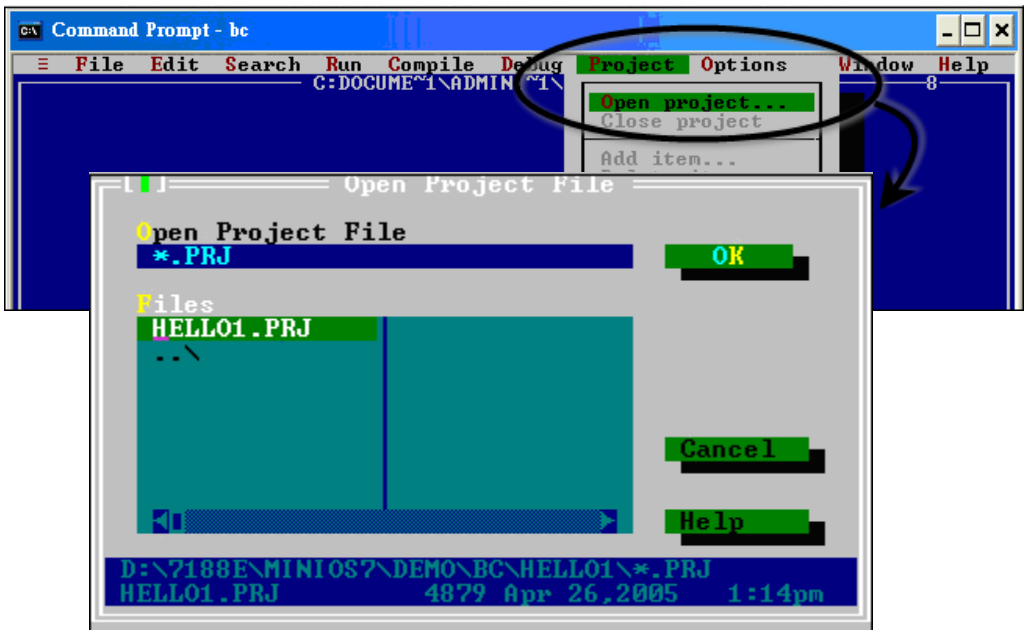




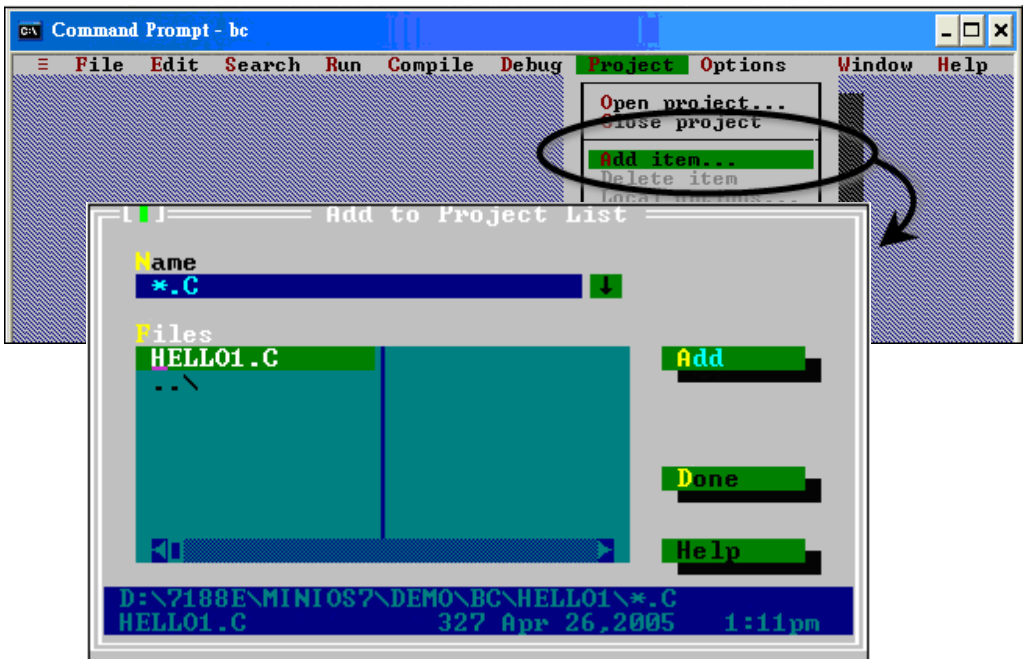
D.2. BC++ 3.1. IDE

Step 1: Executing the Borland C++ 3.1

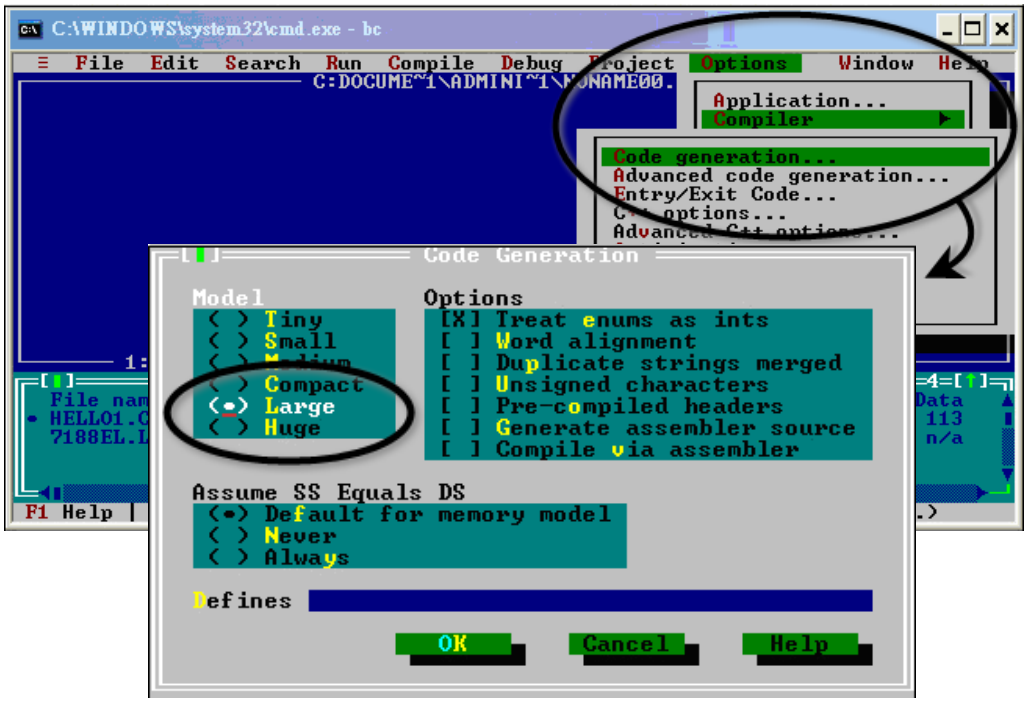
Step 2: Creating a new project file (*.prj)



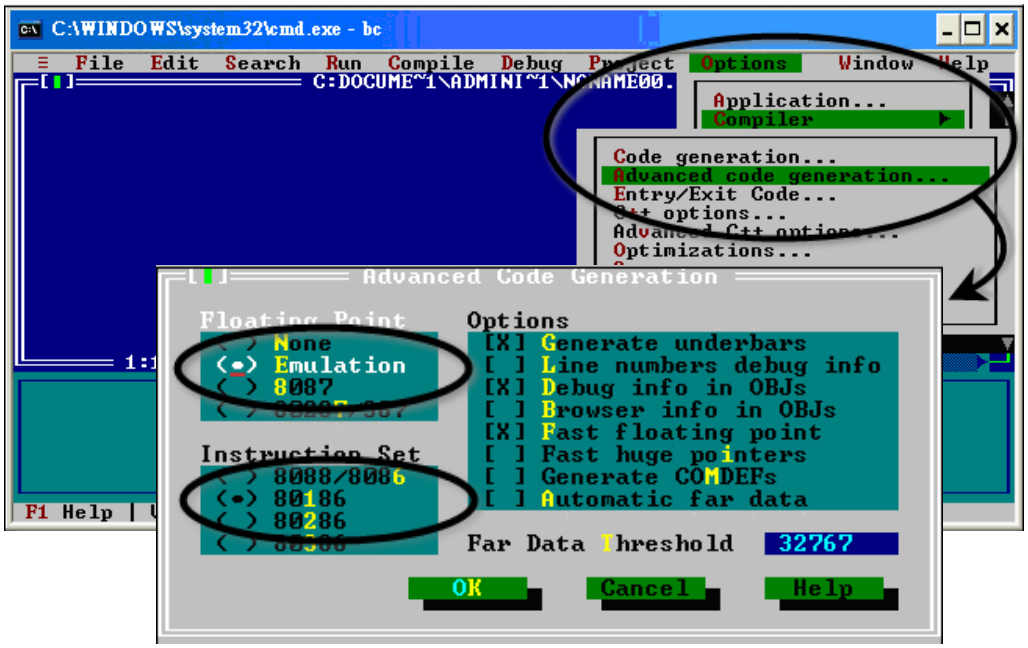
Step 3: Add all the necessary files to the project



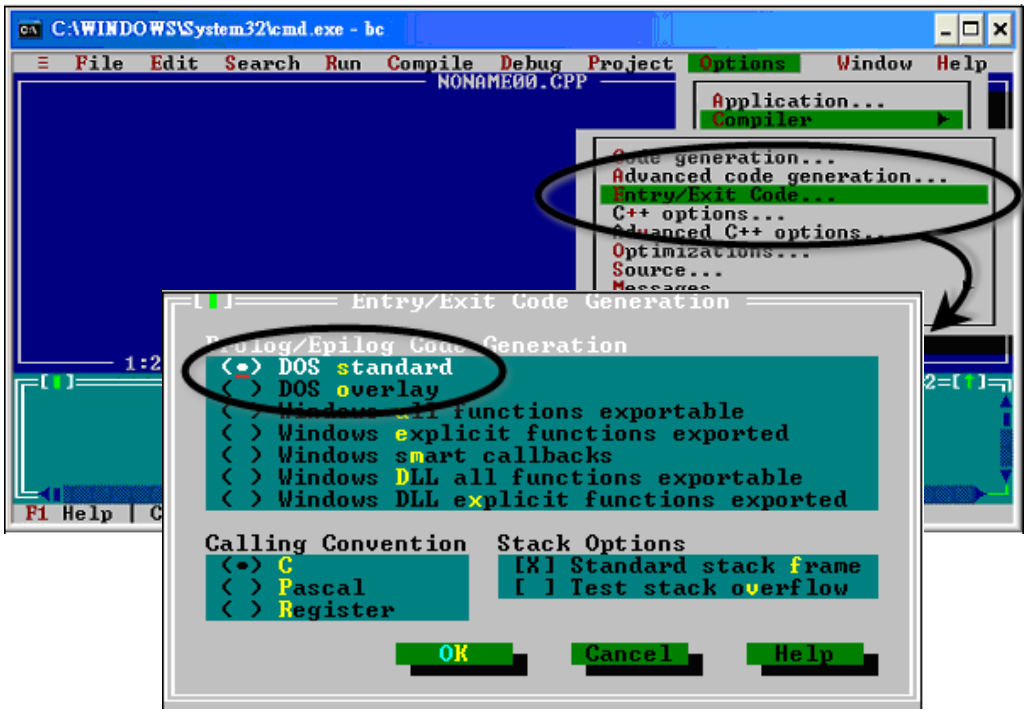
Step 4: Change the Memory model (Large for iPAC-8000.lib)



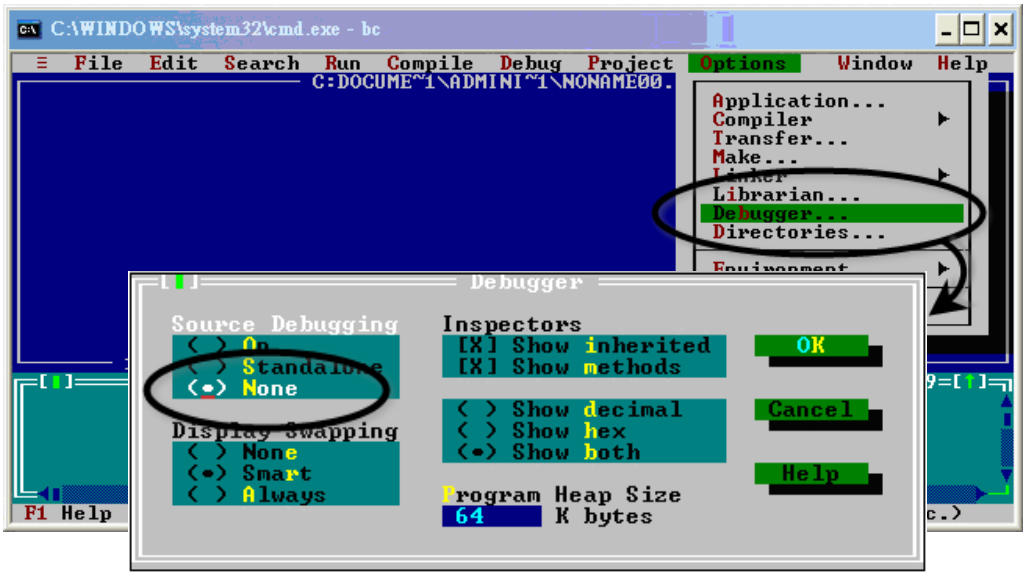
Step 5: Set the Advanced code generation options and Set the Floating Point to Emulation and the Instruction Set to 80186



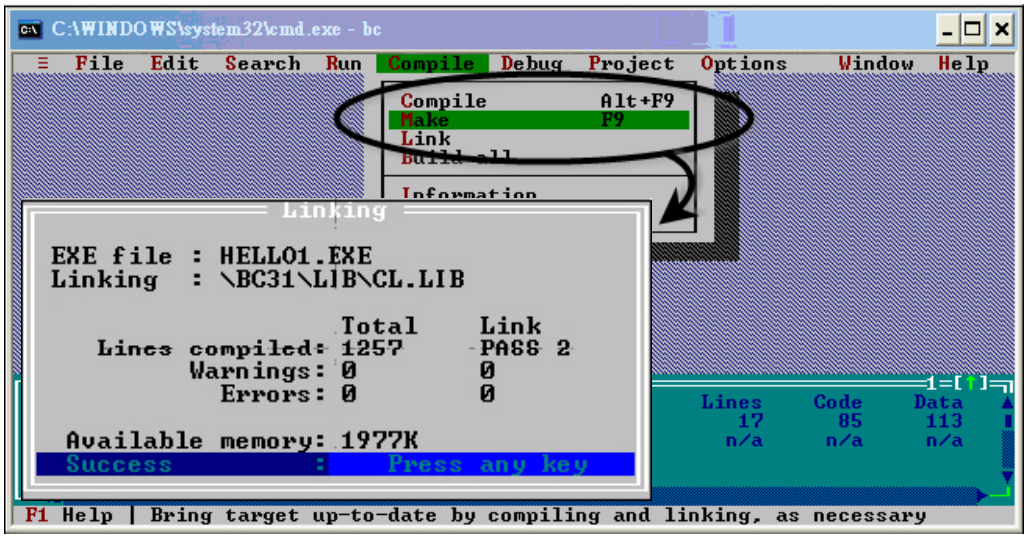
Step 6: Set the Entry/Exit Code Generation option and setting the DOS standard



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

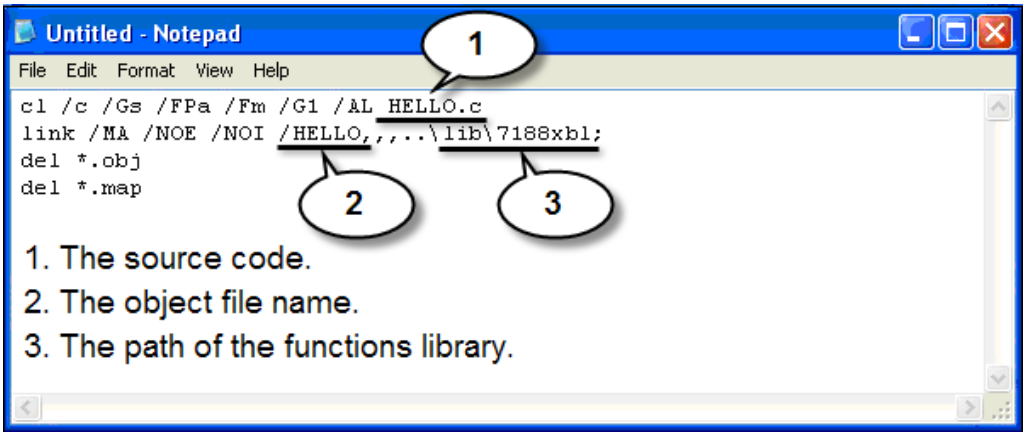


Step 8: Make the project



D.3. MSC 6.00

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



Tip & Warnings

/C: Don't strip comments

/GS: No stack checking

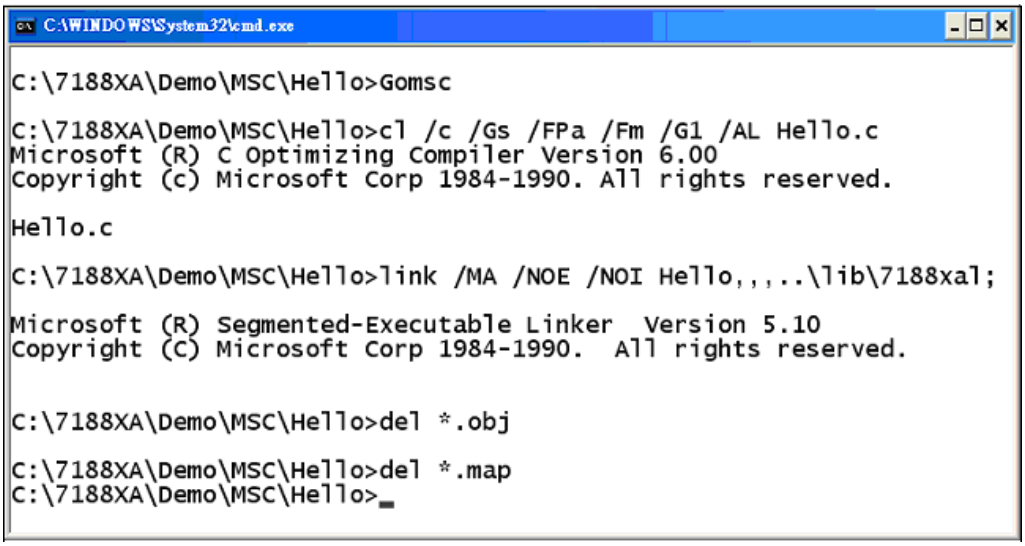
/Fpa: Calls with altmath

/Fm: [map file]

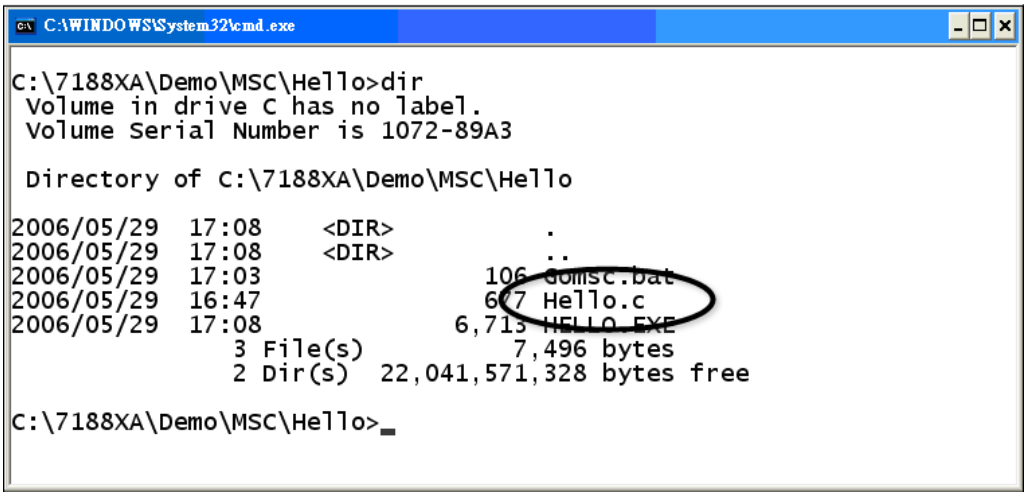
/G1: 186 instructions

/AL: Large model

Step 2: Run the Gomsc.bat file

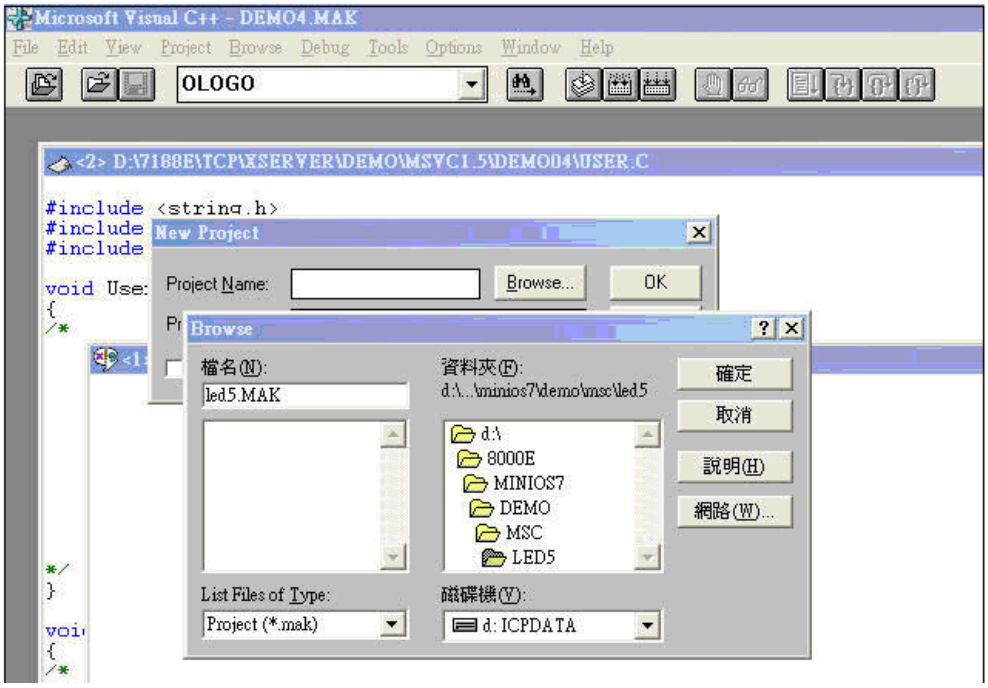


Step 3: A new executable file will be created if it is successfully compiled

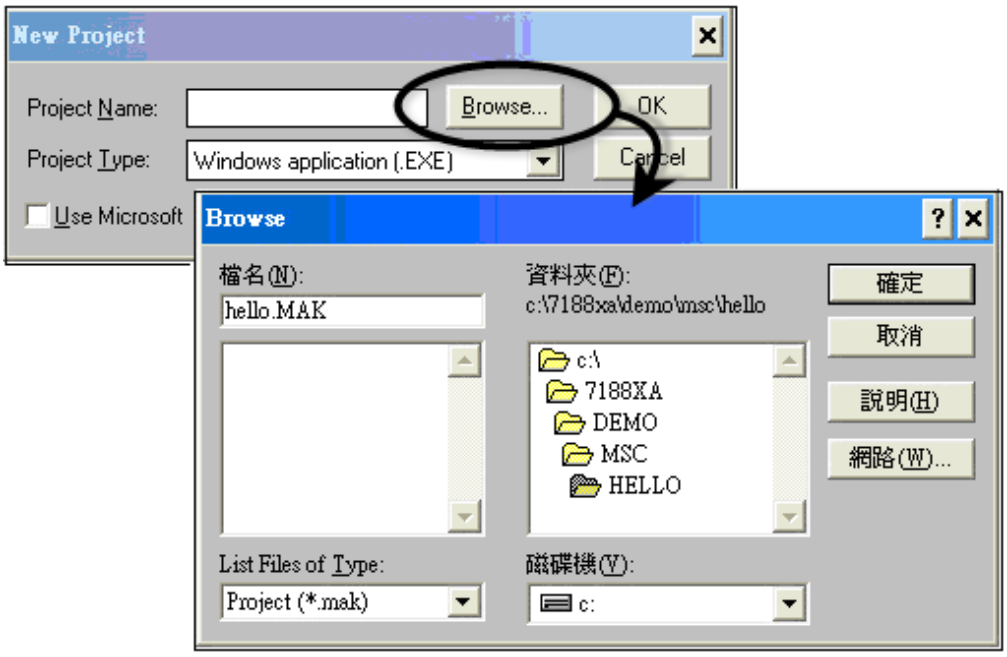


D.4. MSVC 1.50

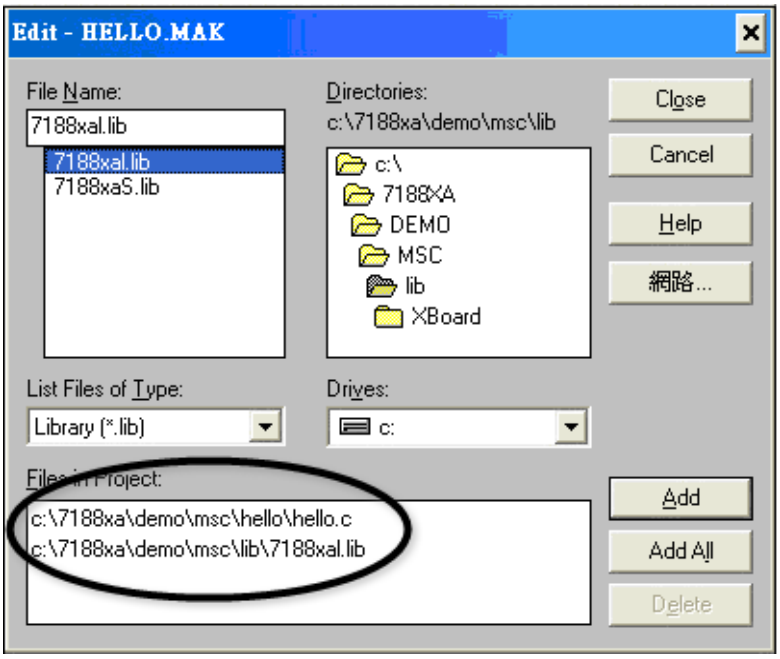
Step 1: Run MSVC.exe



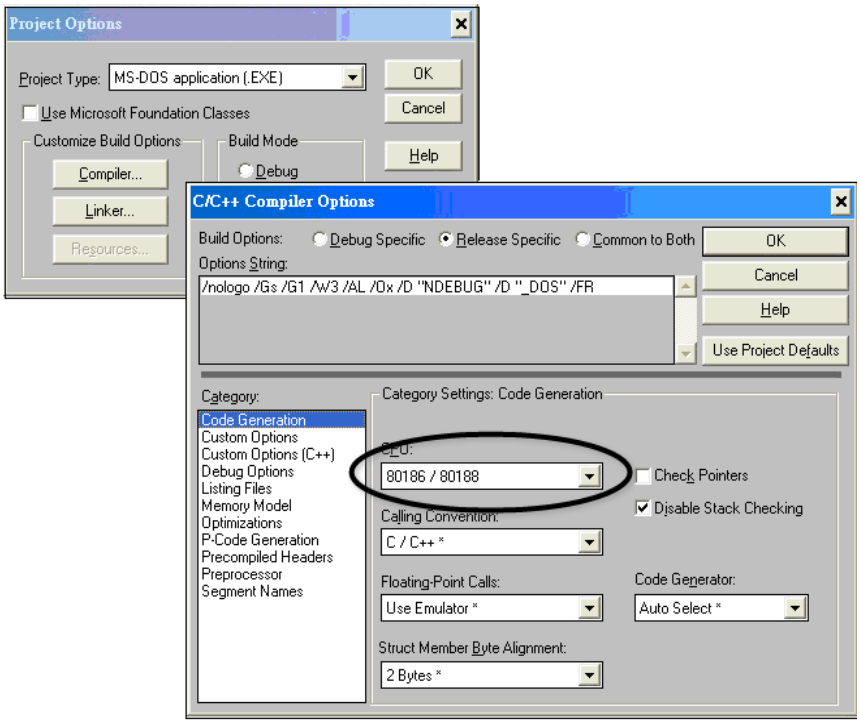
Step 2: Create a new project (*.mak) by entering the name of the project in the Project Name field and then select MS-DOS application (EXE) as the Project type



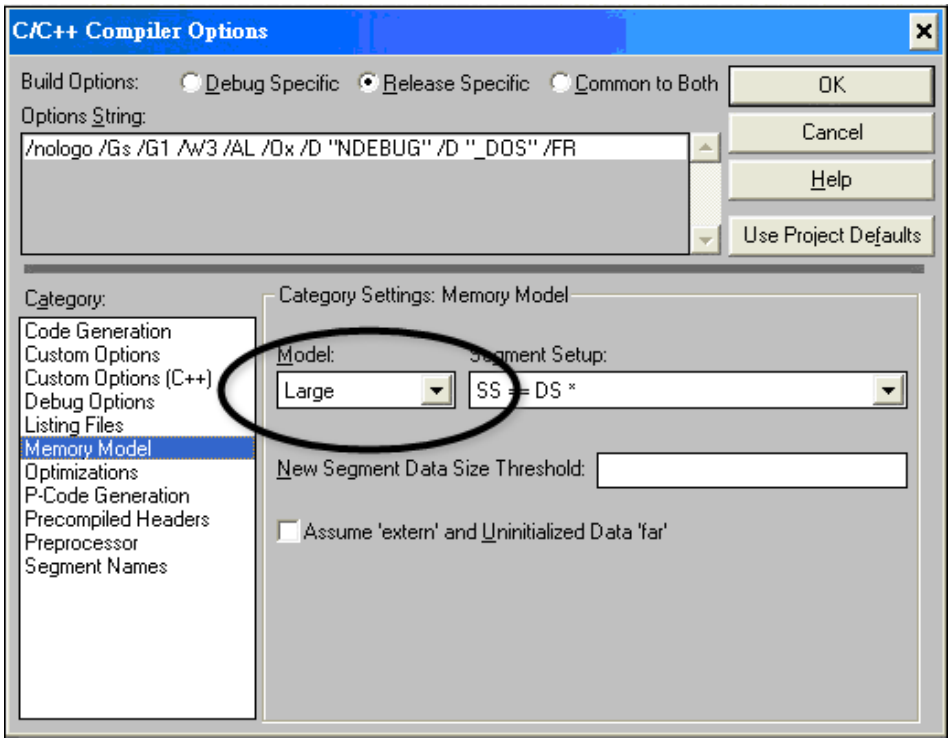
Step 3: Add the user's program and the necessary library files to the project



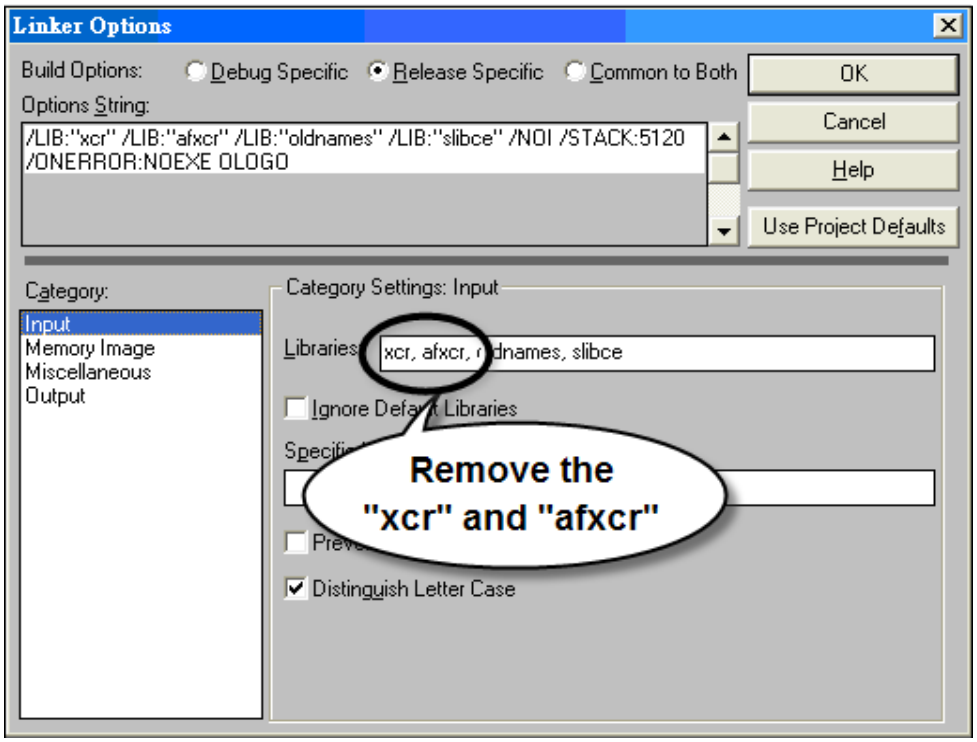
Step 4: Set the Code Generation on the Compiler.



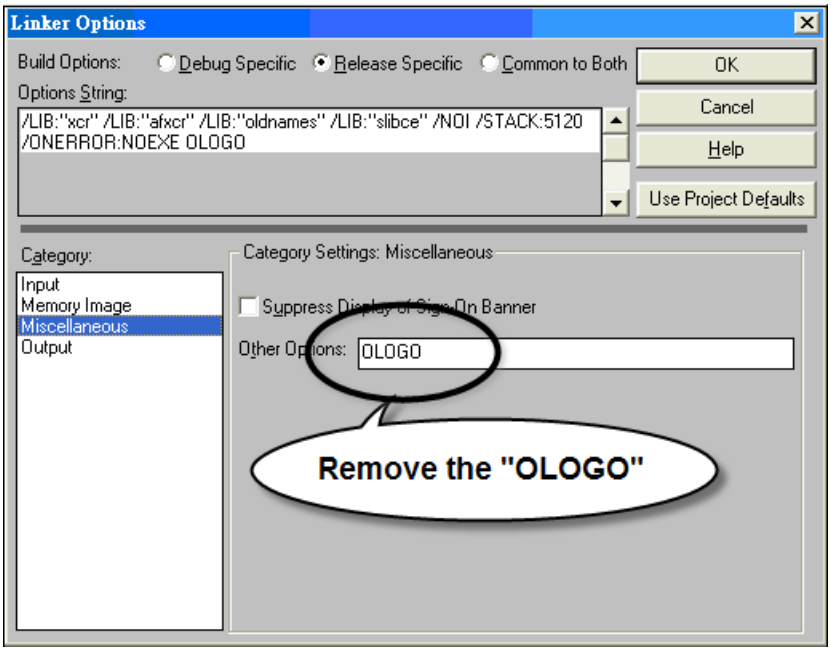
Step 5: Change the Memory model (large for iPAC-8000.lib)



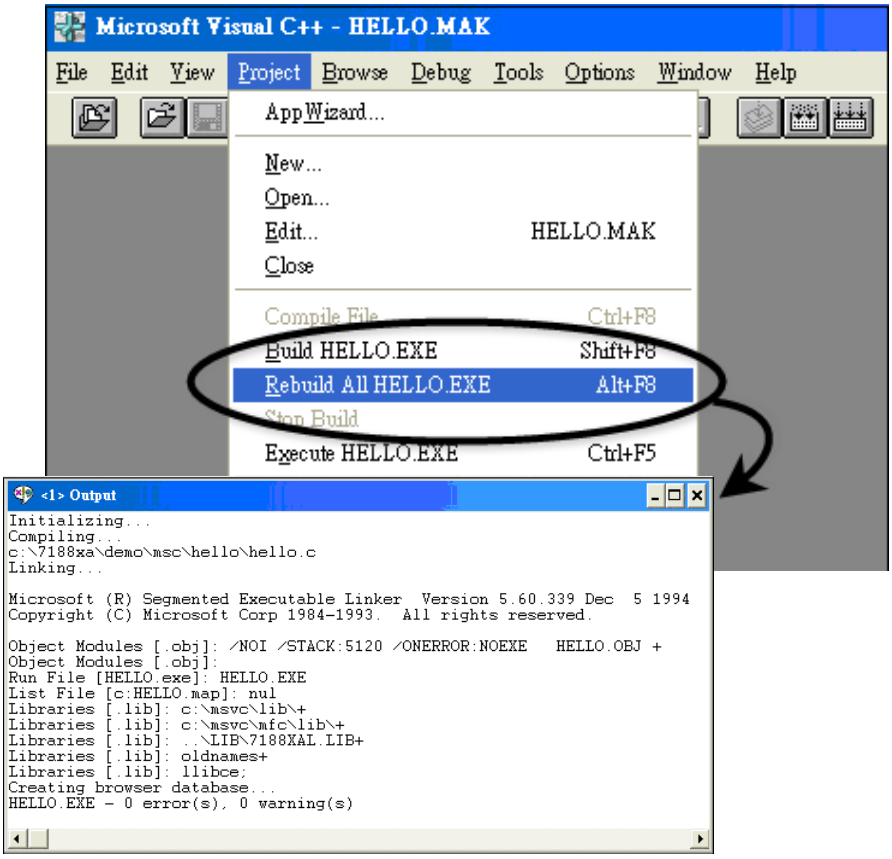
Step 6: Remove the xcr, afxcr libraries from the Input Category



Step 7: Remove the OLOGO option from the miscellaneous Category.

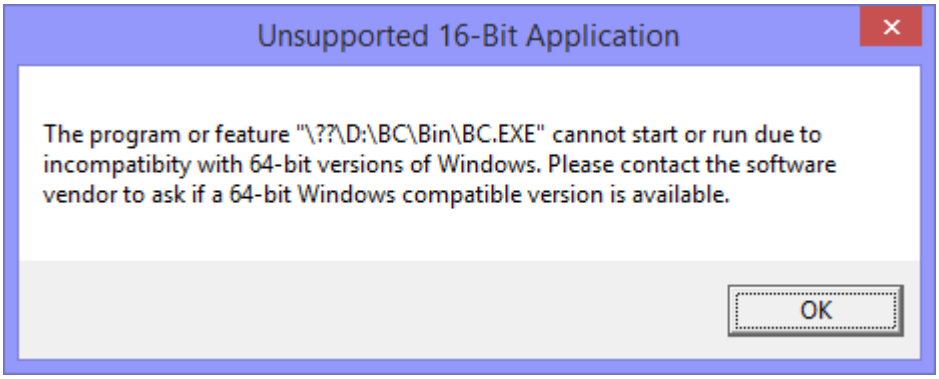


Step 8: Rebuild the project



Appendix E. How to Run a 16-bit Program on 64-bit Windows

Windows 7, Windows 8, or others 64-bit Windows do not support 16-bit programs, 16-bit processes, or 16-bit components. Therefore, if a 16 bit compiler like Borland C++ 3.1 or Turbo C++ 3.0, or a program that was compiled with a 16-bit compiler is running on 64-bit Windows, it may receive an error message like the one below.



The ultimate solution to run 16-bit programs, 16-bit processes, or 16-bit components on PC with Windows 7, Windows 8, or others 64-bit Windows is DOSBox. DOSBox emulates DOS and the environment in which DOS ran in the past (like an old pc), including memory management and sound configuration, but with the power of your computer today.

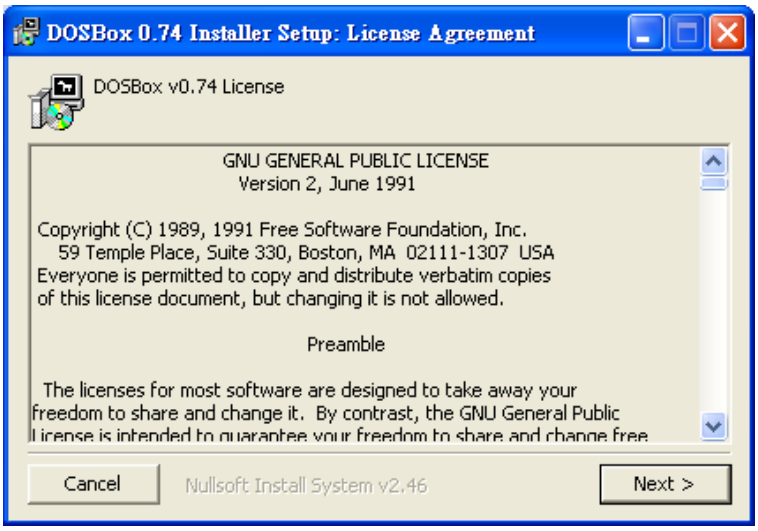
On this appendix we will show you how to make the 16-bit compiler or any 16-bit program work.

Step 1: Get the DOSBox Installer

The latest version of the DOSBox installer can be obtained from SourceForge web site.
http://sourceforge.net/projects/dosbox/

Step 2: Install the DOSBox

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

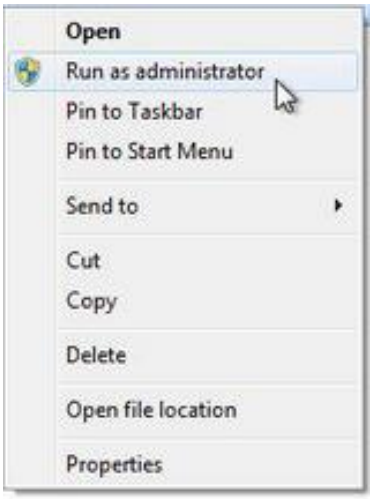


Step 3: Run the DOSBox

Tips & Warnings

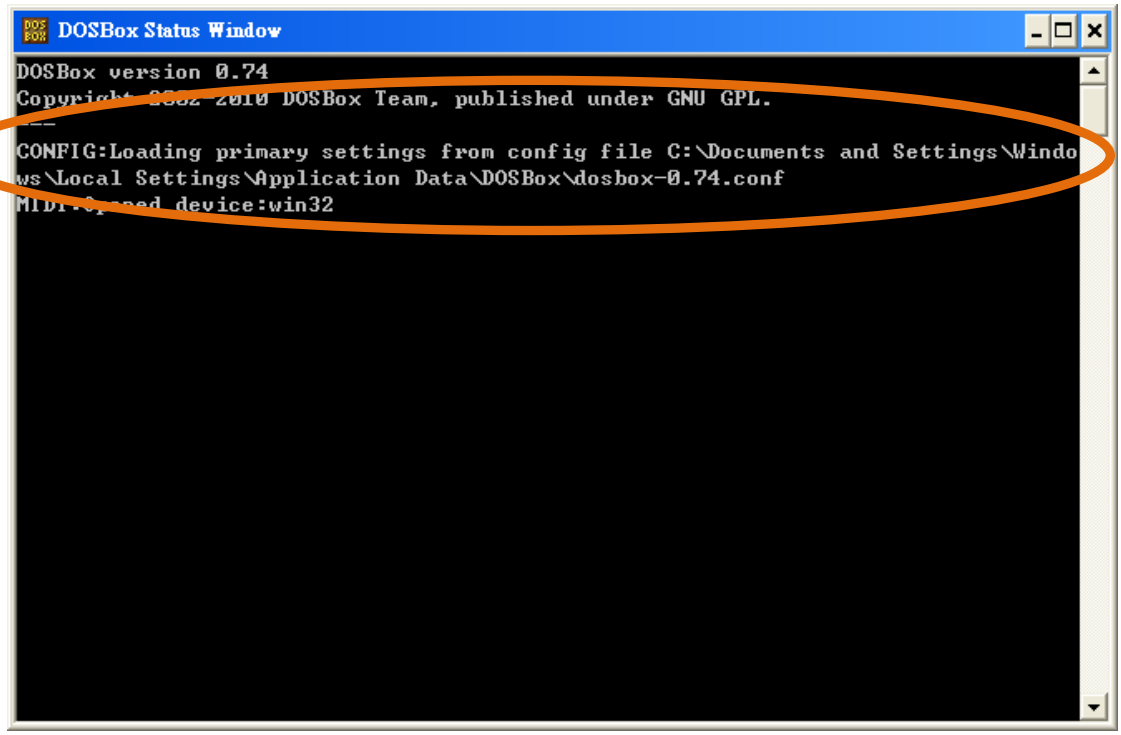
If you're running Windows Vista or later OS, we would suggest you to run the DOSBox emulator with administrator privileges.

Right-click the DOSBox program icon or shortcut, and then click Run as administrator.



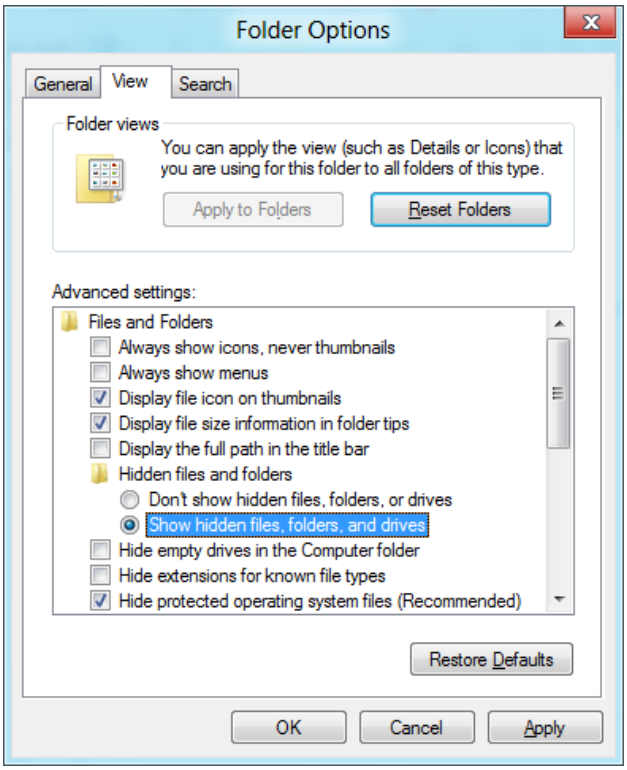
Step 4: On the DOSBox Status Window, check the path of the DOSBox configuration file, and then open up this file with a text-editor like Notepad

The DOSBox configuration file is named dosbox-number.conf where number is the version number of DOSBox.



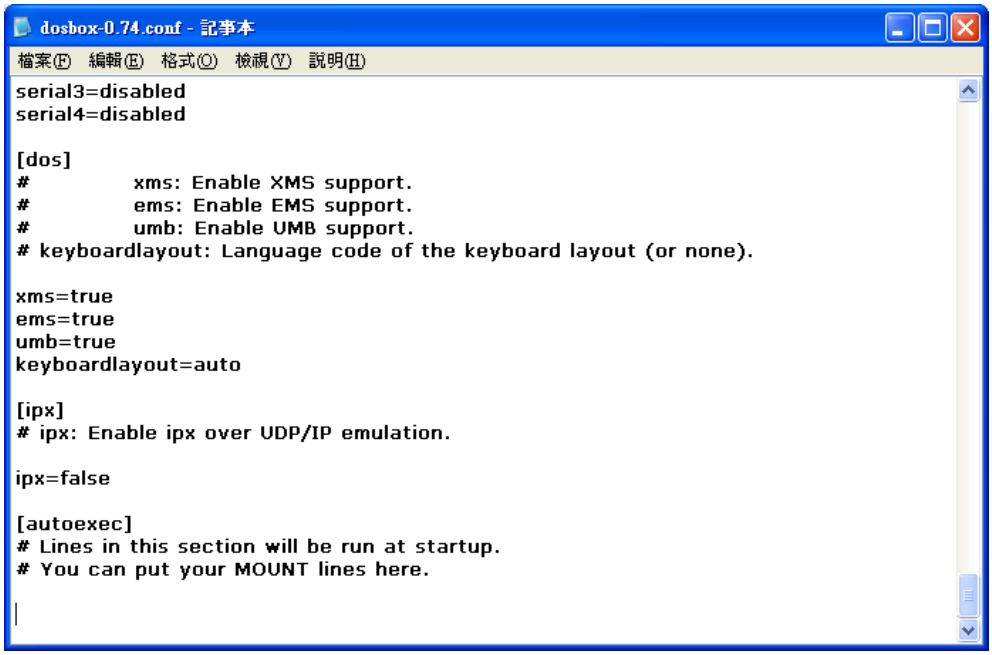
Tips & Warnings

If you can't find the dosbox-number.conf file in the location, you'll have to enable the show hidden files option in the window properties to display the hidden files.



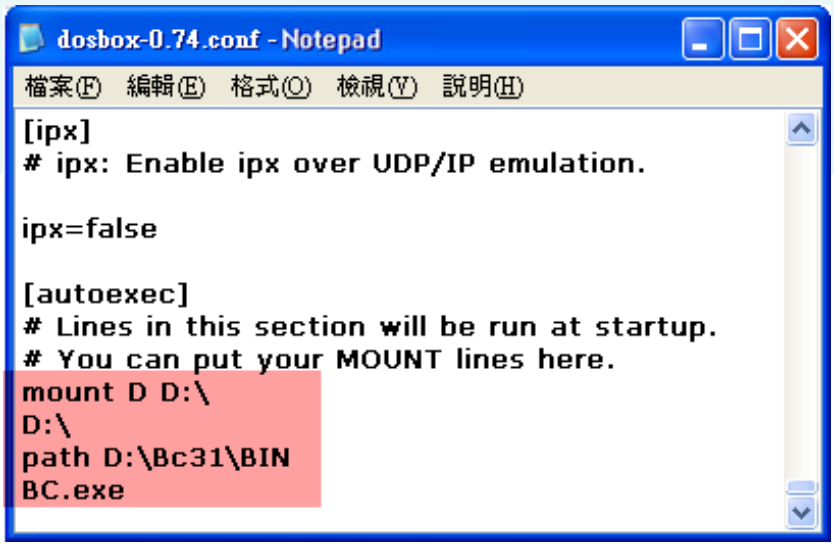
Step 5: Scroll down to the bottom of the configuration file to where it ends with:

```
[autoexec]
# Lines in this section will be run at startup.
# You can put your MOUNT lines here.
```

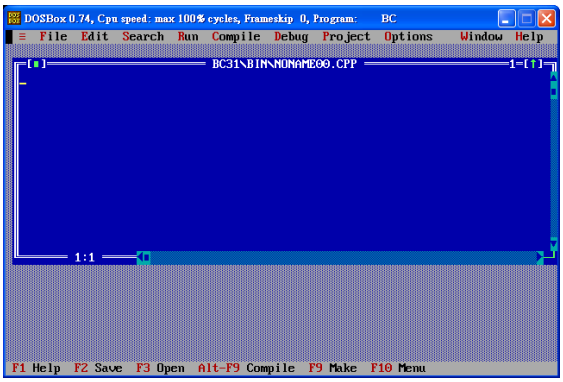


Step 6: Enter the following commands to add the drive letter of the folder that you want to start in and the Borland executable to the DOSBox path

```
[autoexec]
# Lines in this section will be run at startup.
# You can put your MOUNT lines here.
mount D D:\
D:\
path D:\Bc31\BIN
BC.exe
```



Step 7: Restart the DOSBox, and then the BC++ compiler will open in DOSBox

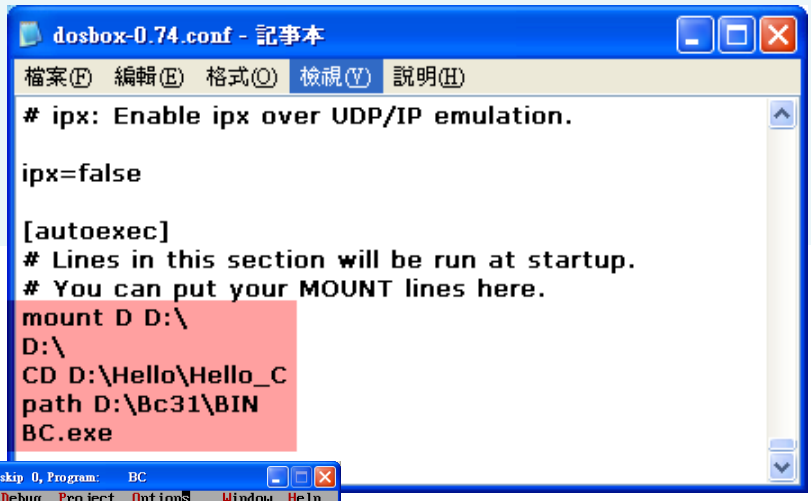


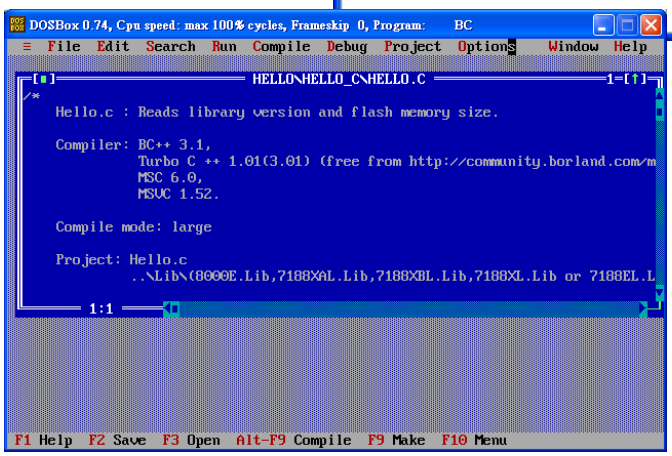
Tips & Warnings

If you want to automatically start a project, add the commands to change the directory to the project folder before the commands to set up the BC++ compiler in the DOSBox configuration file.

For example, if you want to start in the /Hello/Hello_C directory, you would add CD:\Hello\Hello_C to change the directory as follows:

```
[autoexec]
# Lines in this section will be run at startup.
# You can put your MOUNT lines here.
mount D D:\
D:\
CD D:\Hello\Hello_C
path D:\Bc31\BIN
BC.exe
```





Appendix F. Revision History

The table below shows the revision history.

Revision	Date	Description
1.0.1	July 2011	Initial issue
1.0.2	August 2012	1. Modified the basic information of iPAC-8000 in chapter 1. 2. Modified the basic operation of iPAC-8000 in chapter 2. 3. Added the update information of iPAC-8000 in chapter 5.
1.0.3	March 2014	1. Added the tip about programming with 64-bit Windows in section 3.1.1. Installing the C Compiler 2. Added the appendix about how to run 16-bit programs on 64-bit Windows in Appendix E. How to Run 16-bit Programs on 64-bit Windows
1.0.4	December 2014	Modified the information about how to automatically mount 16-bit C/C++ compiler in DOSBox in Appendix E. How to Automatically Mount 16-bit C/C++ Compiler in DOSBox