iPAC-8000 Series

User Manual

(C Language Based PAC)

v1.0.6, May 2020

Service and usage information for

iP-8411

iP-8411

iP-8441/iP-8441

iP-8841/iP-8841

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 PACthat 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

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="4">80 MHz</td><td rowspan="2">512 KB</td><td rowspan="4">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="2">768 KB</td><td rowspan="2">2
10-Base-T</td><td rowspan="2">3</td></tr>
<tr><td>iP-8841</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 EthernetCommunication(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.

5. **DCON firmware**

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

B. Modbus firmware

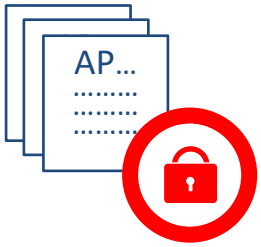
Modbus firmware supports the standard Modbus/TCP protocol. SCADA software can easily access the I/O module plugged in the iPAC-8000.

➢ 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.

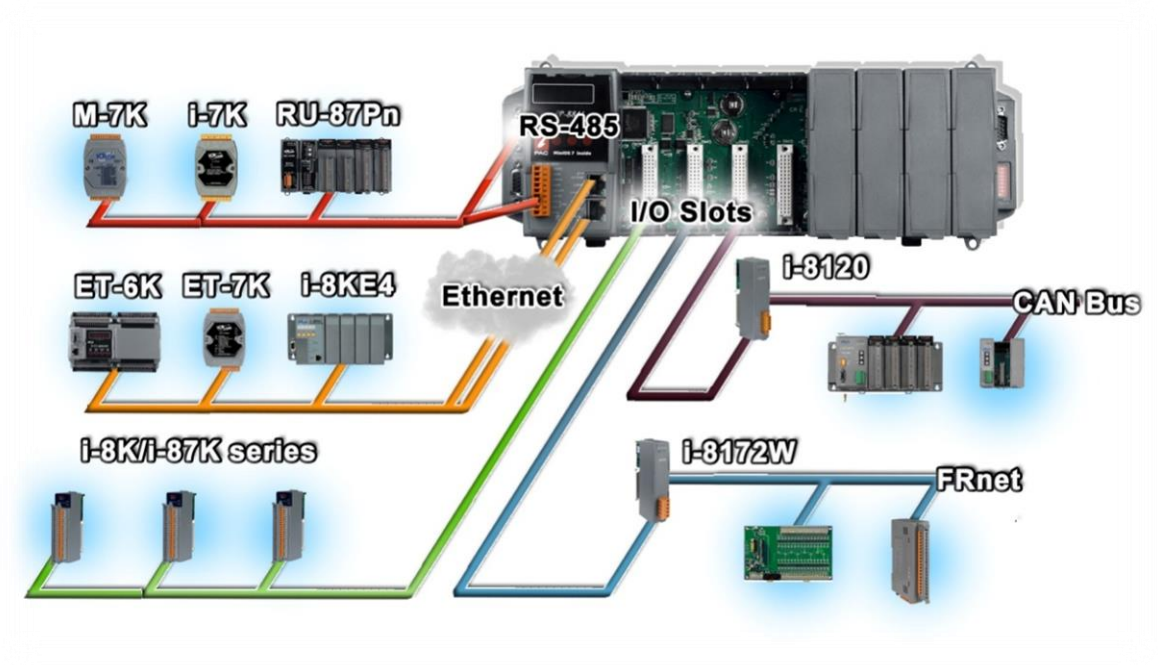
➢ Dual Ethernet Ports(for Ethernet series PAC only)

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.

Models	iP-8411	iP-8811	iP-8441	iP-8841
System Software				
CPU	MiniOS7 (DOS-like embedded system)			
Program Download Interface	RS-232 (COM1) or Ethernet			
Programming Language	C Language			
Compilers to create.exe files	TC++ 1.01, TC 2.01, BC++3.1 ~ 5.2x, MSC 6.0, MSVC++ (before version 1.5.2)			
CPU Module				
CPU	80186 or compatible, 16-bit and 80 MHz			
SRAM	512 KB		768 KB	
Dual Battery Backup SRAM	512 KB; data valid up to 5 years			
Flash	512 KB with write protect switch			
Expansion Flash Memory	microSD socket (can support microSDHC up to 32 GB)			
Dual Battery Backup SRAM	512 KB (for 5 years data retention)			
EEPROM	16 KB			
NVRAM	31 Bytes (battery backup, data valid up to 5 years)			
RTC (Real Time Clock)	Provide second, minute, hour, date, day of week, month, year			
64-bit Hardware Serial Number	Yes, for software copy protection			
Watchdog Timers	Yes (0.8 second)			
DIP Switch	Yes (8 bits)			
Communicate Interface				
COM0	Internal communication with the high profile I-87K series modules in slots			
COM1	RS-232 (to update firmware) (RxD, TxD and GND); non-isolated			
COM2	RS-485 (Data+, Data-) with internal self-tuner ASIC; 3000 VDC isolated			
COM3	RS-232/RS-485 (RxD, TxD, CTS, RTS and GND for RS-232, Data+ and Data- for RS-485); non-isolated			
COM4	RS-232RS-232 (RxD, TxD, CTS, RTS, DSR, DTR, CD, RI and GND); non-isolated			
Ethernet Port	-		RJ-45 x 2, 10/100 Base-TX (Auto negotiating, Auto MDI/MDI-X, LED indicators)	

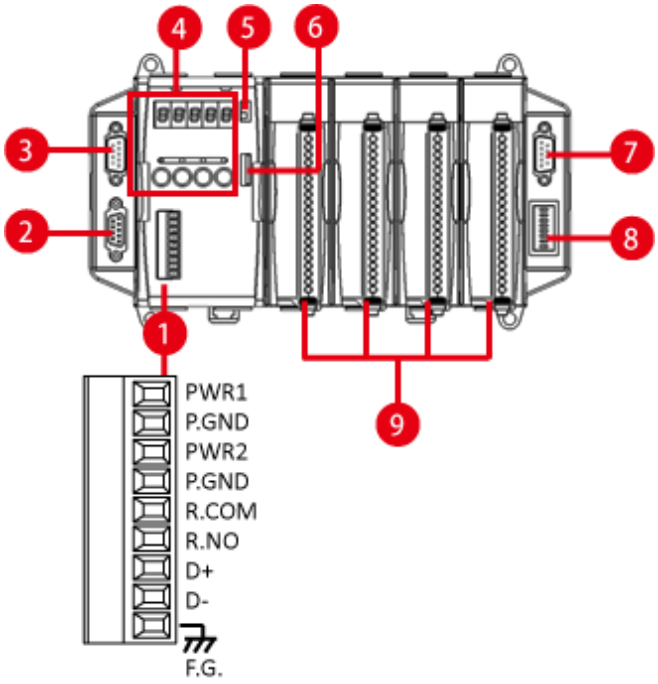
SMMI				
LED Display	Yes, 5-Digit			
Programmable LED Indicators	3			
Push Buttons	4			
Buzzer	-		Yes	
I/O Expansion Slots				
Slot Number	4 slots	8 slots	4 slots	8 slots
Mechanical				
Dimensions (W x L x H, unit: mm)	231 x 132 x 111	355 x 132 x 111	231 x 132 x 111	355 x 132 x 111
Installation	DIN-Rail or Wall Mounting			
Models	**iP-8411**	**iP-8811**	**iP-8441**	**iP-8841**
Operating Environment				
Operating Temperature	-25 ~ +75 ° C			
Storage Temperature	-30 ~ +80 ° C			
Humidity	10 ~ 90 % RH, non-condensing			
Power				
Input Range	+10 ~ +30 V_{DC}			
Isolation	1 kV			
Redundant Power Inputs	Yes, with one power relay (1 A @ 24 V_{DC}) for alarm			
Capacity	30 W			
Consumption	6.7 W	7.2 W	6.7 W	7.2 W

1.3. Overview

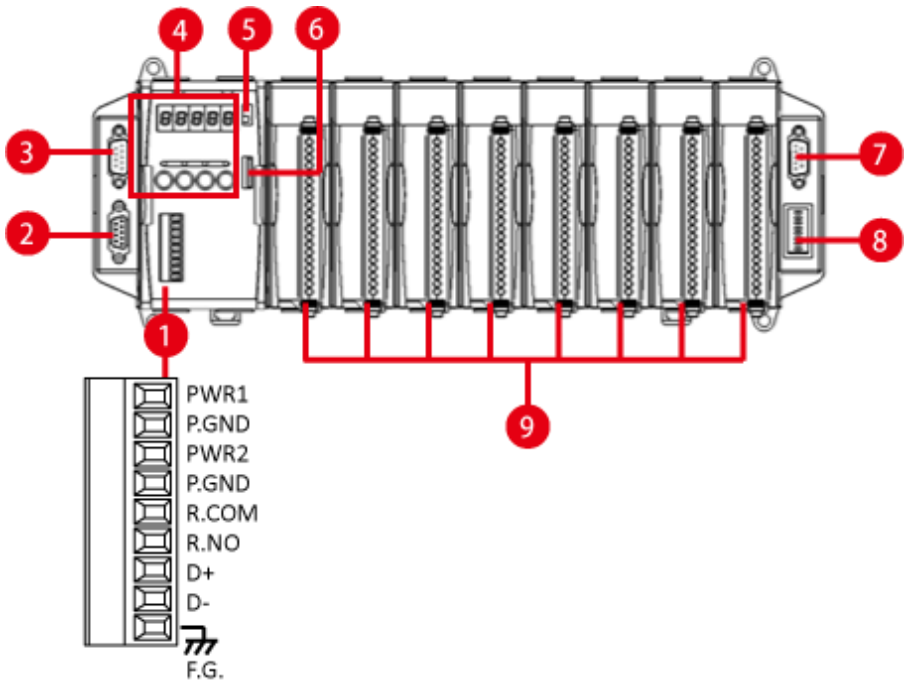
iPAC-8000 consists of several different components that integrate with ICP DAS system. Here is an overview of the components and its descriptions. The following list shows the details of the components:

- **Serial series PAC**

iP-8411

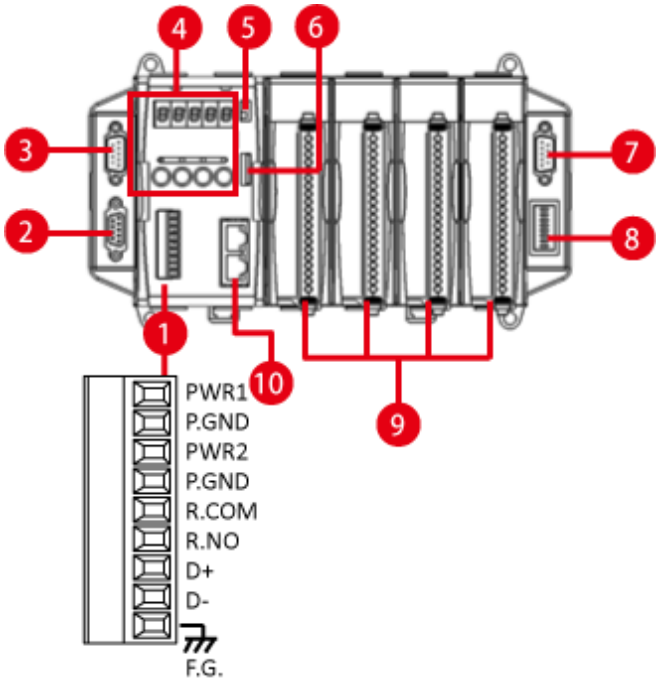


iP-8811

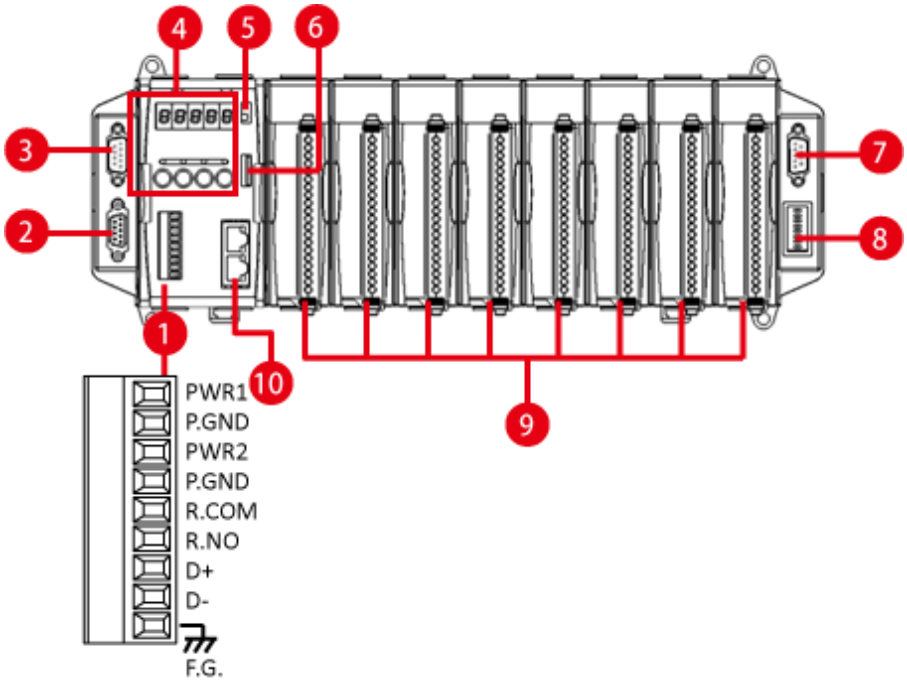


➢ Ethernetseries PAC

iP-8441

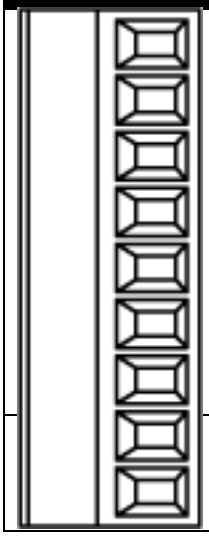


iP-8841



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

COM1 (RS-232)

Port Type: Female

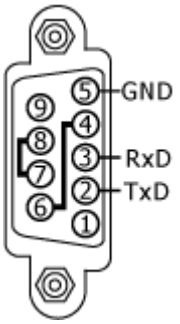
Baud Rate: 115200, 57600, 38400, 19200, 9600, 4800, 2400, 1200 bps

Data Bits: 7, 8

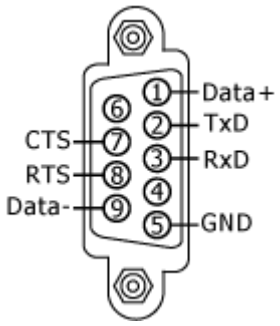
Parity: None, Even, Odd

Stop Bits: 1

FIFO: 1 byte



COM3 (RS-232/RS-485)



Port Type: Male

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

Data Bits: 5, 6, 7, 8

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

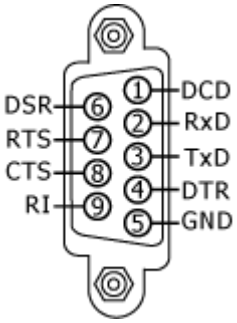
COM3 can be configured as either RS-232 or RS-485 that only can select one at a time; its configuration depends on the pin connections as follows:

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

RS-485 (Data+ and Data-)

There is no software configuration or hardware jumper needed.

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

SMMI (Small Main-Machine Interface)

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

Operating Mode Selector

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

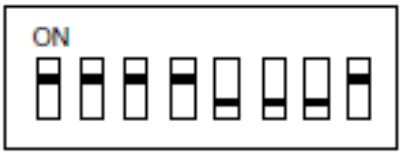
For more information about these modes of operation, see section 2.3., “Configuring the Boot Mode.”

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 32 GB.

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. For more information about the address settings about the iPAC-8000, see section“Appendix E. How to Set a Network Address”



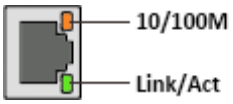
Expansion I/O Slots

iPAC-8000is equipped with 4/8expansion 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 https://www.icpdas.com/en/product/guide+Remote_I_O_Module_and_Unit+PAC_%EF%BC%86amp;_Local_I_O_Modules+I-8K_I-87K_Series_(High_Profile)

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.

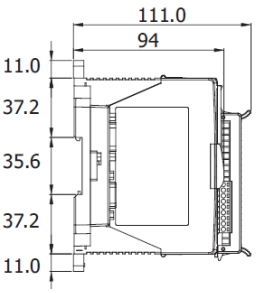
LED Indicator	State (Color)	Meaning
10/100M	ON (Orange)	Network Speed: 100 MB
	OFF	Network Speed: 10 MB
Link/Act	ON (Green)	The Link is active
	OFF	The Link is inactive
	Blinking(Green)	Network activity

1.4. Dimension

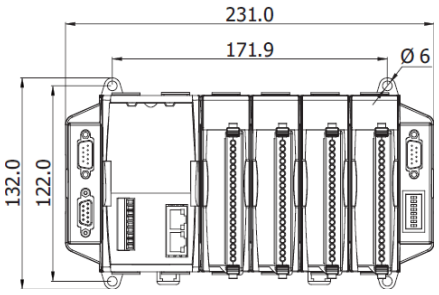
The diagrams below provide the dimensions of iPAC-8000 to use in defining your enclosure specifications. Remember to leave room for potential expansion if you are using other components in your system.

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.

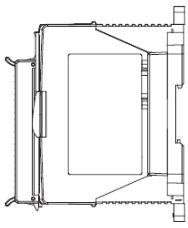
1.4.1. iP-8411/iP-8441



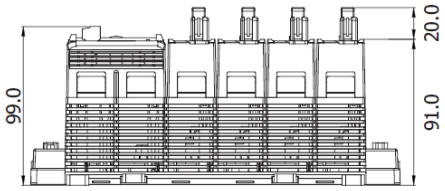
Left Side View



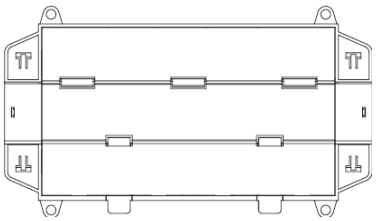
Front View



Right Side View

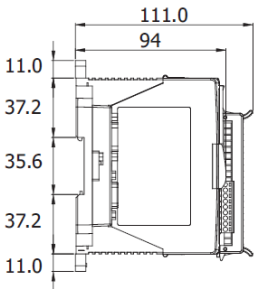


Bottom View

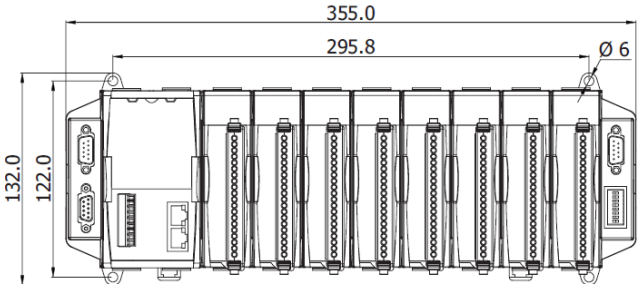


Back View

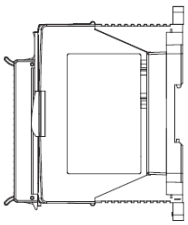
1.4.2. iP-8811/iP-8841



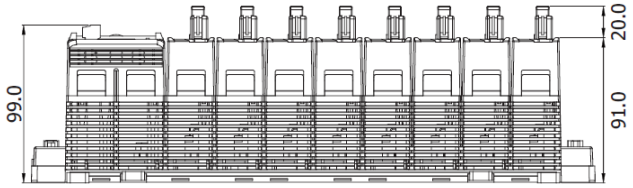
Left Side View



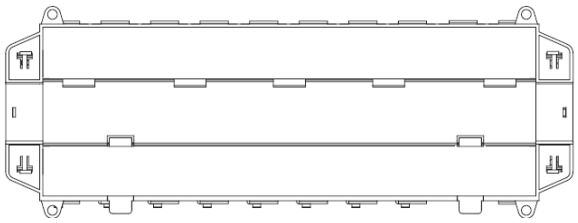
Front View



Right Side View



Bottom View



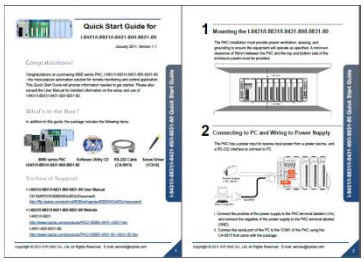
Back View

2. Getting Started

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



Quick Start Guide

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.”

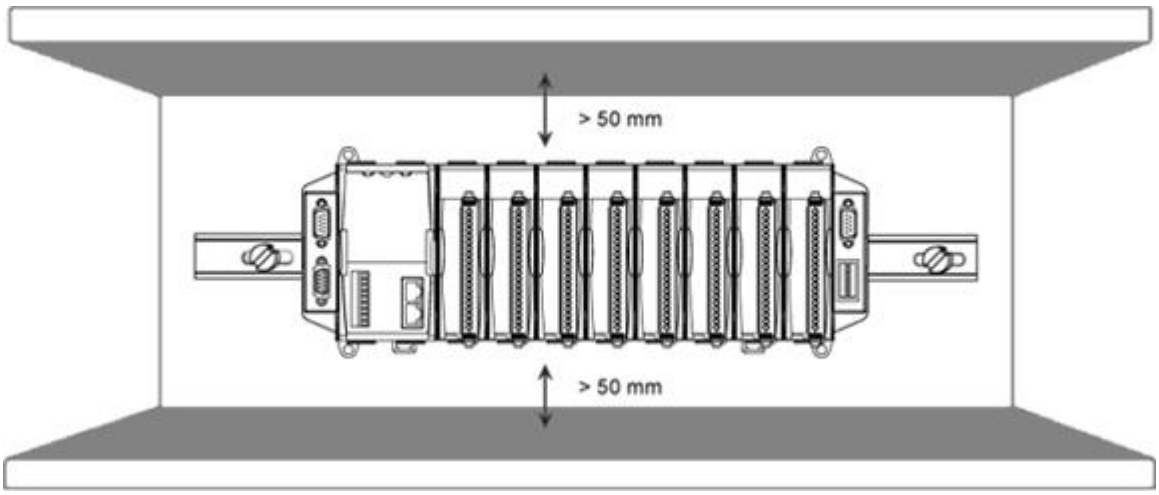
You also need to know the expansion capacities in order to choose the best expansion module for achieving maximal efficiency.

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

https://www.icpdas.com/en/product/guide+Remote_I_O_Module_and_Unit+PAC_%EF%BC%86amp;_Local_I_O_Modules+I-8K_I-87K_Series_(High_Profile)

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.



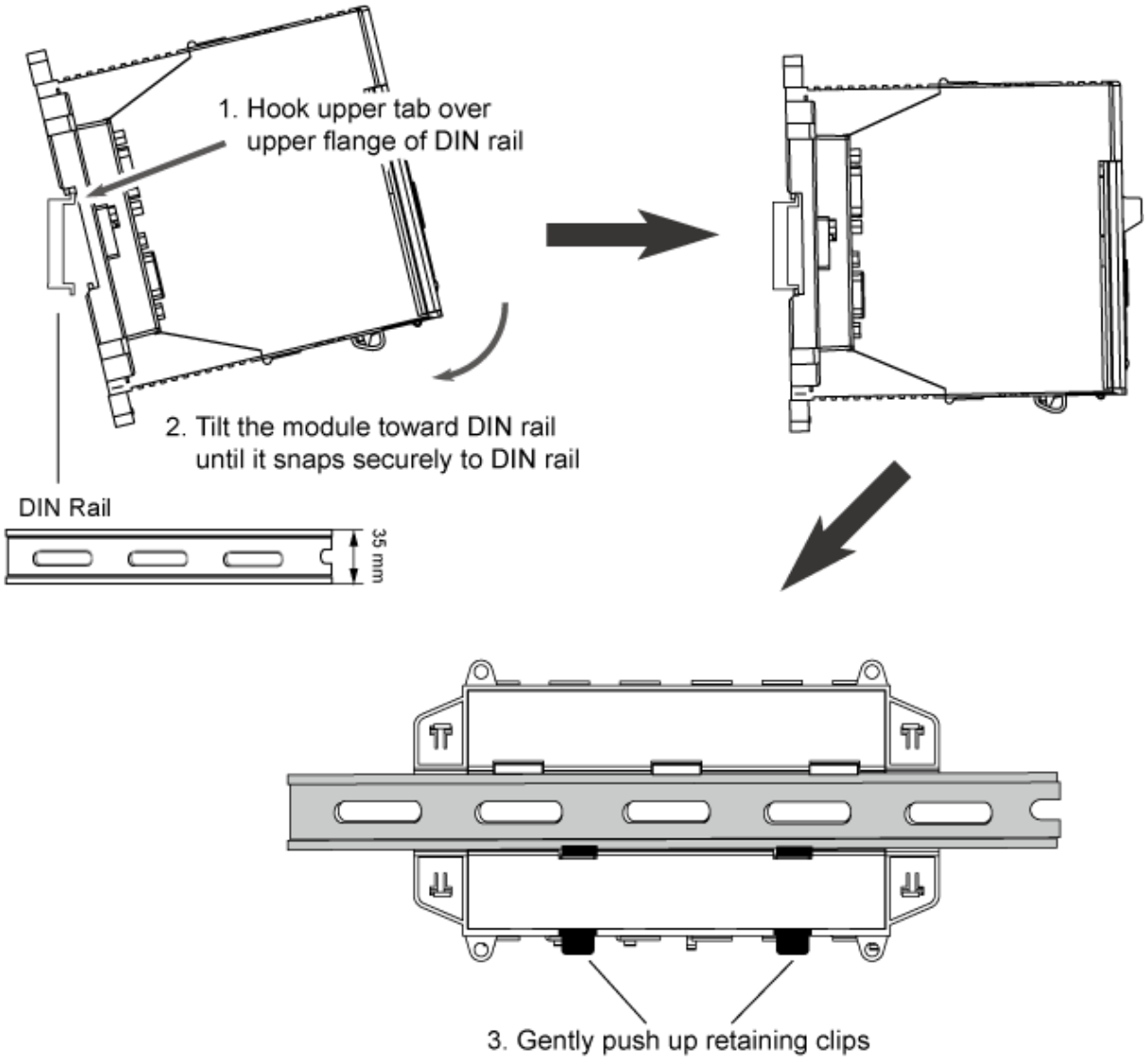
There are two ways for mounting the iPAC-8000.

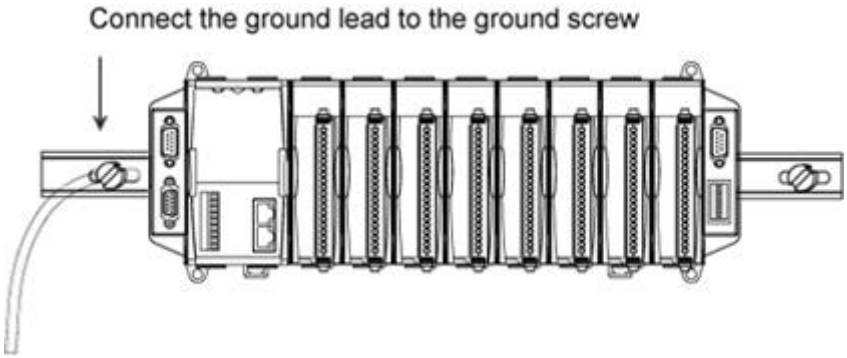
To mount the iPAC-8000 on a DIN rail

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





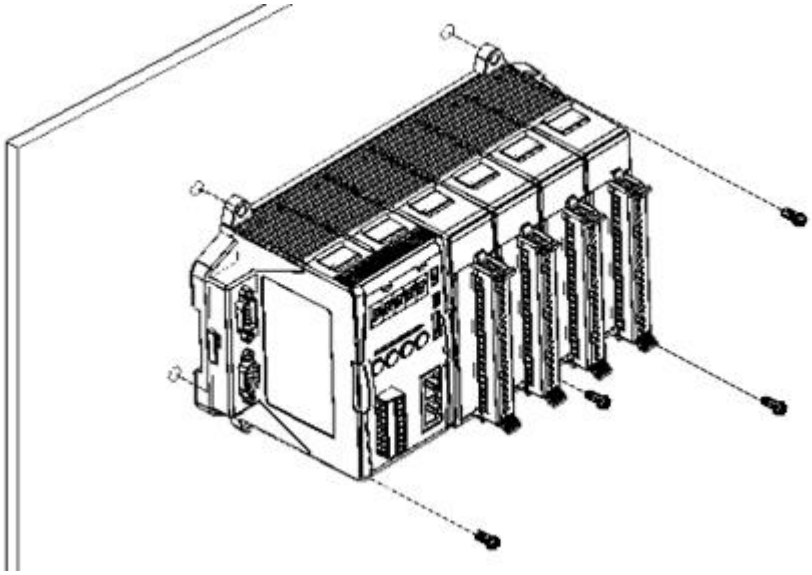
Grounding

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.

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

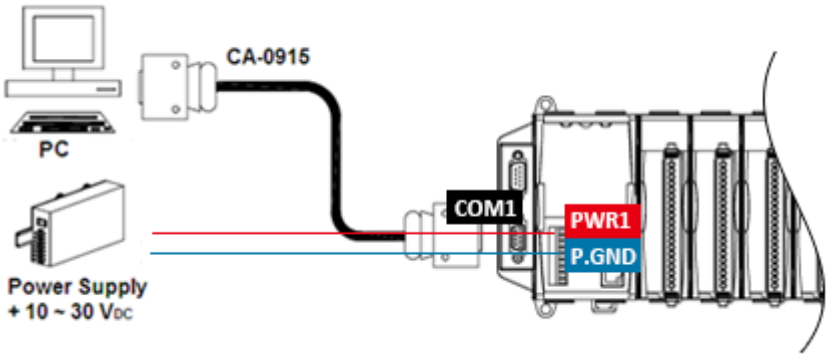
The iPAC-8000 requires a 10 to 30 V_{DC} power supply to operate.

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

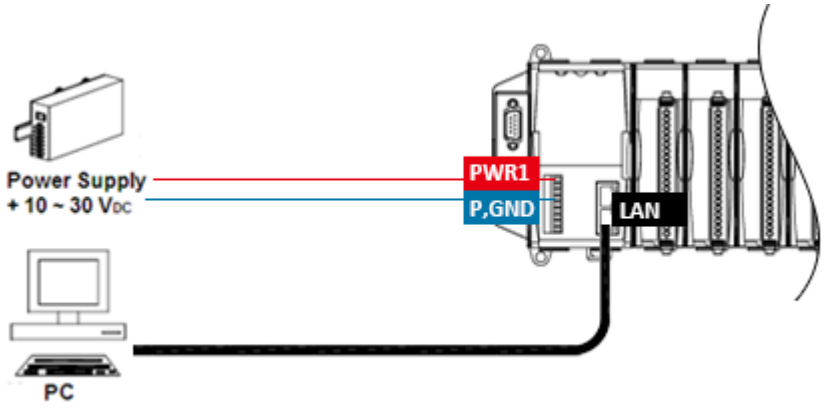
ii. Connect the GND of the iPAC-8000 to the negative of the power supply.

Step 2: Connect to PC

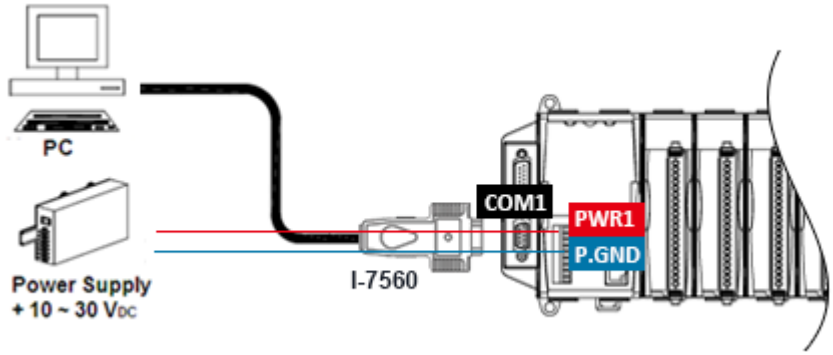
You can connect the COM1 of the iPAC-8000 to PC using a RS-232 cable (CA-0915) that was provided with the package.



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.

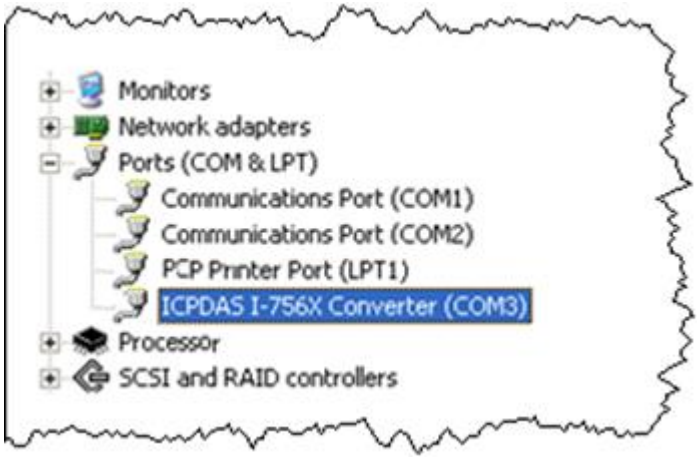


The I-7560 driver must be installed before using the I-7560 converter.

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.

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

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

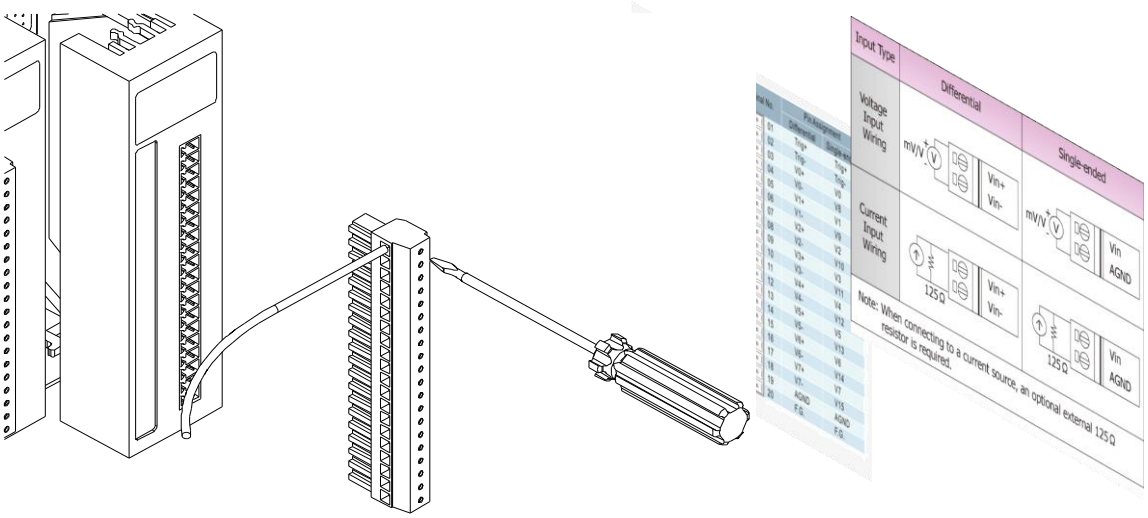
These modules have their own manuals, so if you are using them you should supplement this manual with the manual specifically designed for the special module.

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.

https://www.icpdas.com/en/product/guide+Remote_I_O_Module_and_Unit+PAC_%EF%BC%86amp;_Local_I_O_Modules+I-8K_I-87K_Series_(High_Profile)

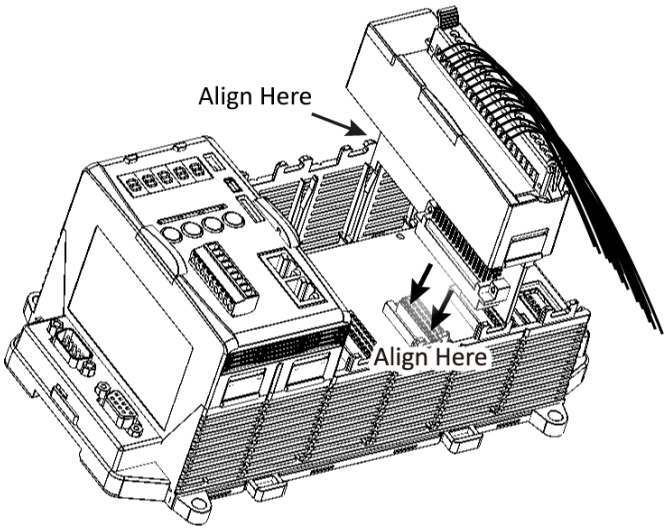
Step 2: Wiring the I/O module

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

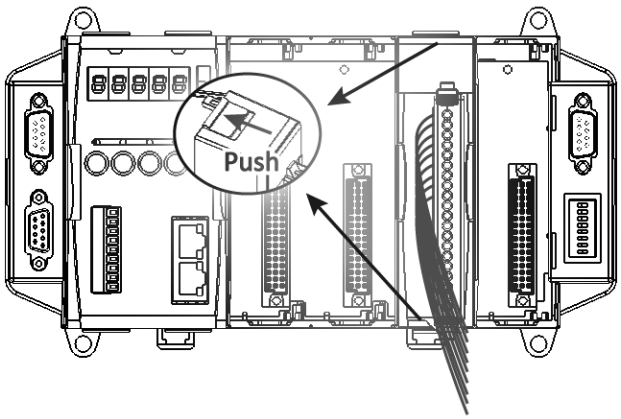


Step 3: Insert the I/O module into iPAC-8000

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



2. 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 libraries files can be downloaded from ICP DAS web site.

https://www.icpdas.com/en/download/show.php?num=3065&model=iP-8411

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.

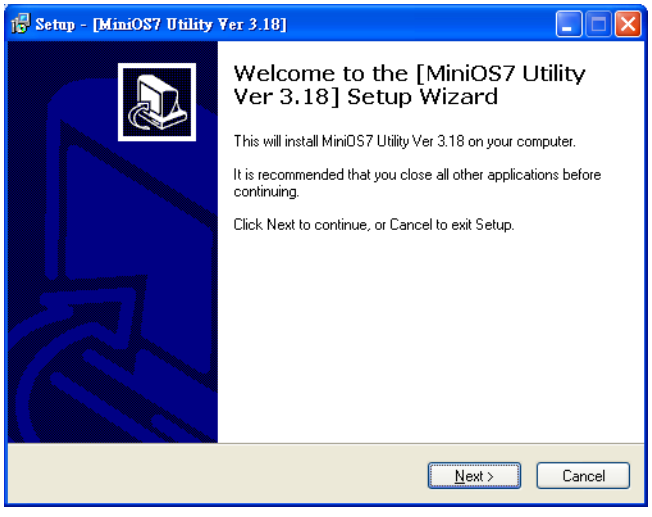
Step 1: Get the MiniOS7 Utility

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.

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

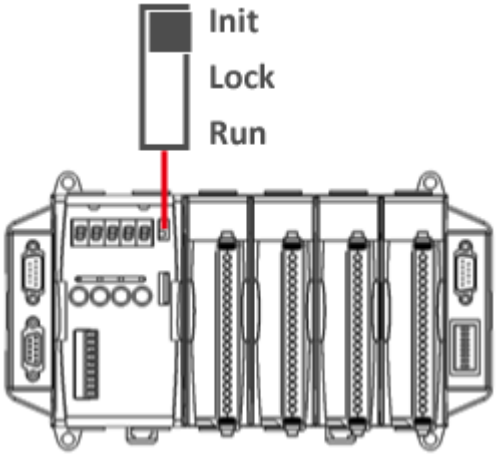
Step 2: Follow the prompts to complete the installation

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



2.3. Configuring the Boot Mode

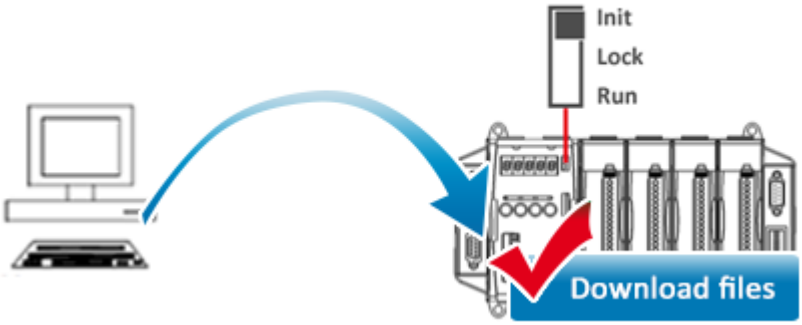
The iPAC-8000 has three standard modes of operation, initial mode, lock mode, and running mode, these modes can be configured by a DIP switch.



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

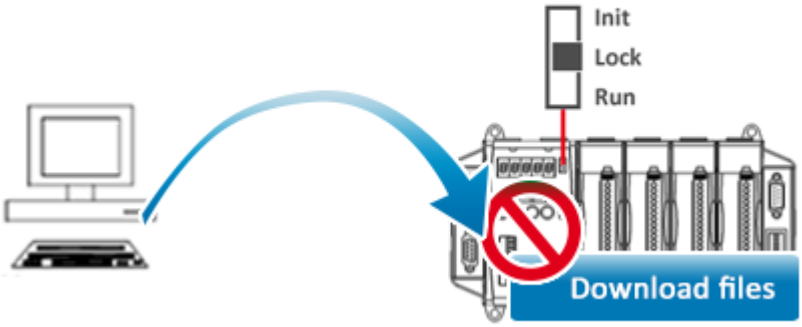
Position	Operating Mode	Description
Init	Initial Mode	OS cannot execute autoexec.bat
		Flash can be read/written
Lock	Lock Mode	OS can execute autoexec.bat
		Flash only can be read (lock)
Run	Running Mode	OS can execute autoexec.bat
		Flash can be read/written

2.3.1. Initial Mode



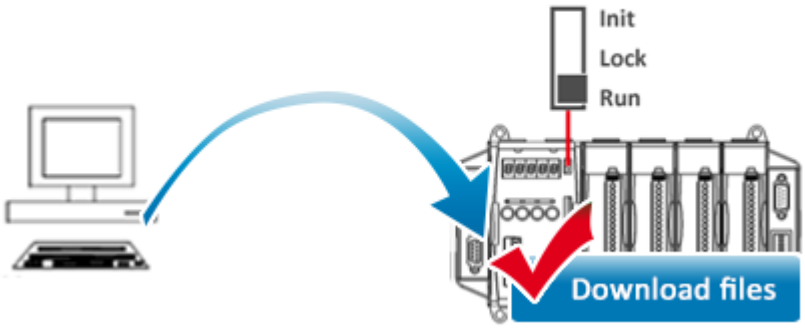
Initial mode is used to update the OS image and upgrade the firmware. When you boot into initial mode, iPAC-8000 doesn't execute the autoexec files and will enter the OS operation mode, in this case there is no program running on the iPAC-8000, the 5-digital 7-SEG LEDs will count the number as shown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

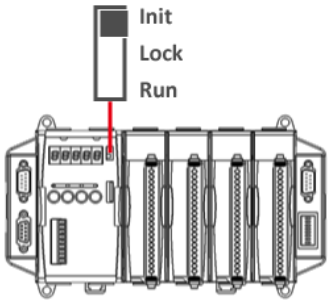
The factory default IP settings are as follows:

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

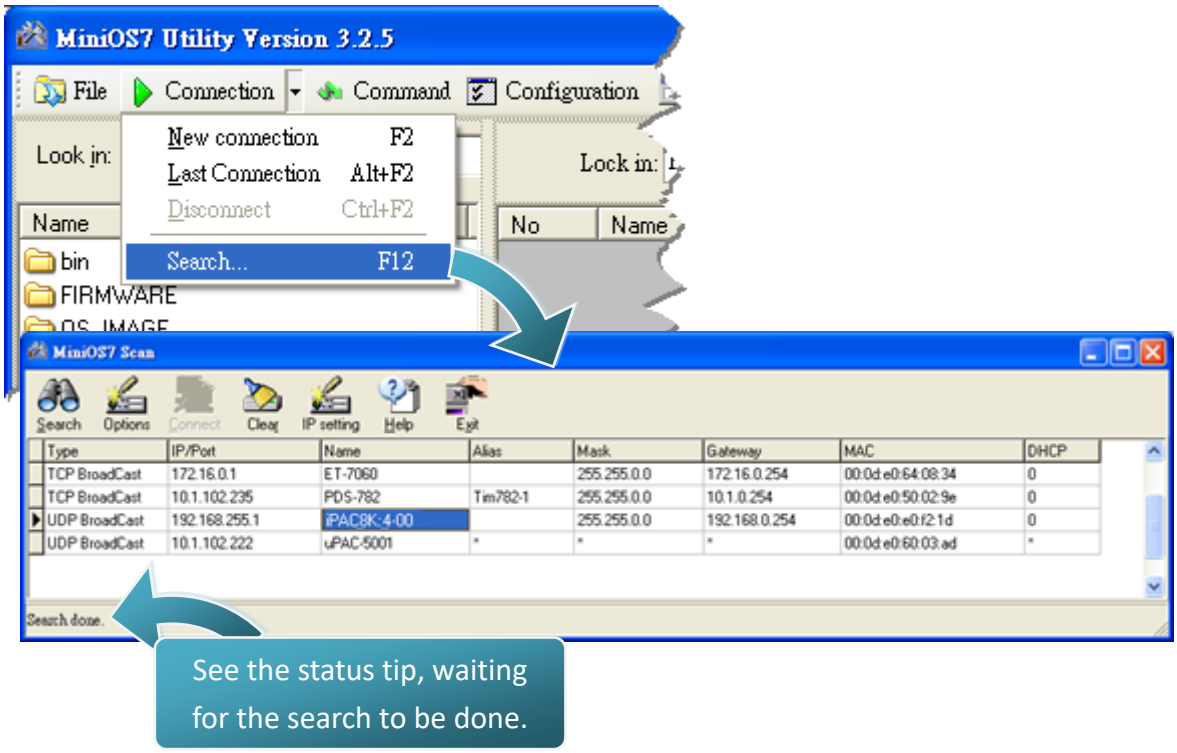
Make sure the DIP switch is in “init” position.



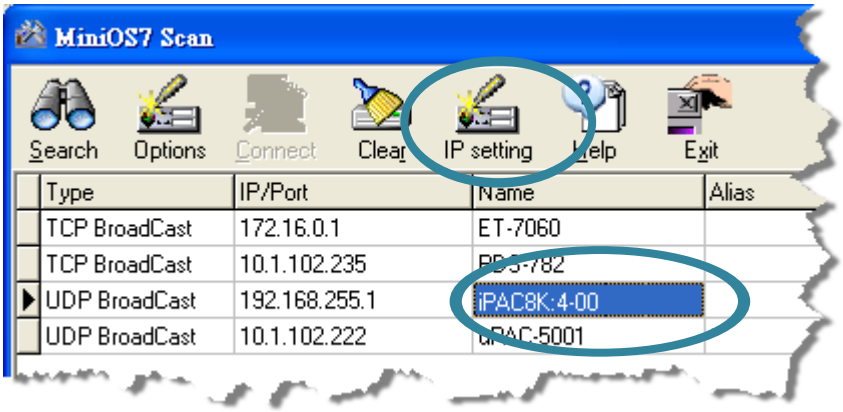
Step 2: Run the MiniOS7 Utility

Step 3: Click the “Search” from the “Connection” menu

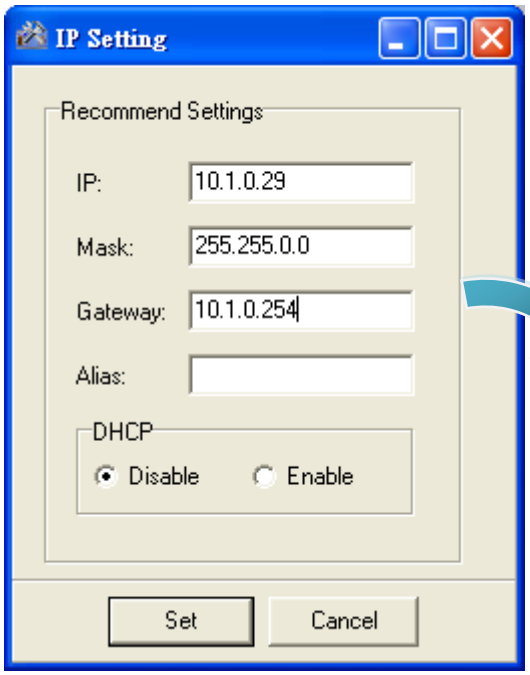
You need to wait for the process to be done.



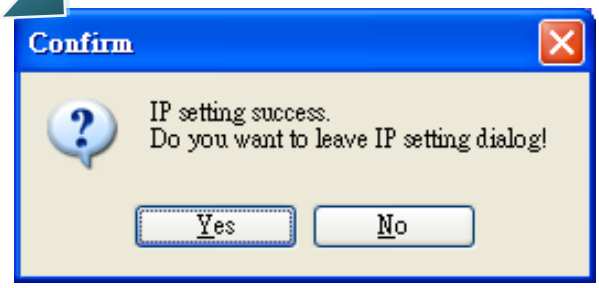
Step 4: Choose the name of iPAC-8000 module (which comes with a default IP address “192.168.255.1/192.168.255.2”) from the list and then click the “IP setting” from toolbar



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



Step 6: Click “Yes” button



2.5. Uploading iPAC-8000 Firmware/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)

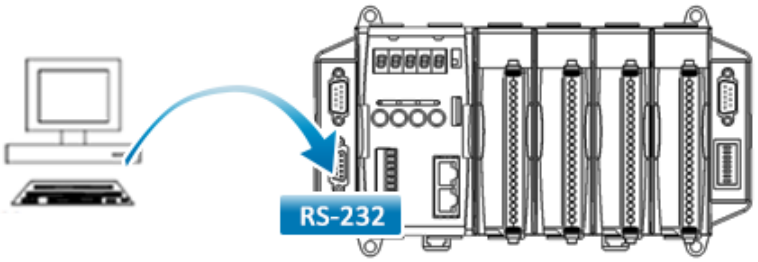
All of these main tasks will be described in detail later.

2.5.1. Establishing a connection between PC and iPAC-8000

Before you use MiniOS7 Utility to upload programs, ensure thatiPAC-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.”

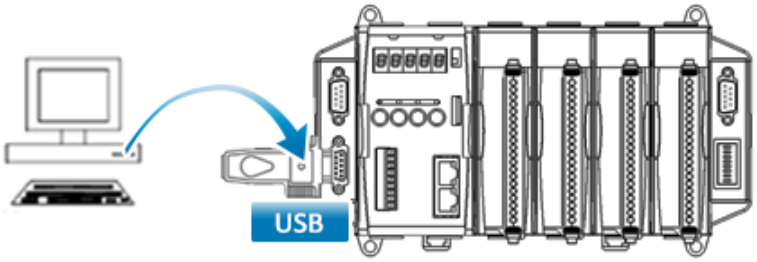
The connection can be divided into the following three types according to the type of wire:

1. RS-232 (see section 2.5.1.1)



---

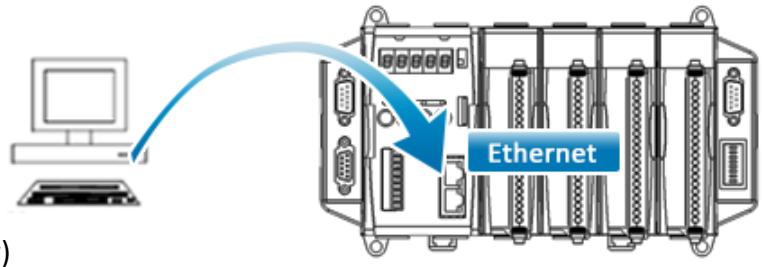
2. USB (see section 2.5.1.2)



---

3. Ethernet (see section 2.5.1.3)

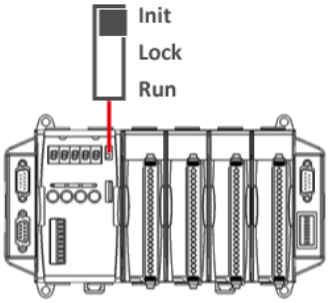
   (forEthernet controllers only)



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

2.5.1.1. Using RS-232 to Establish a Connection

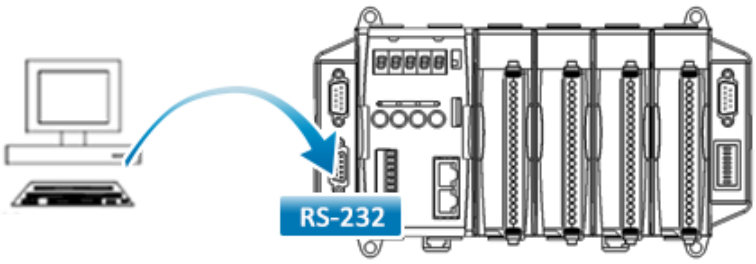
Below are step-by-step instructions on how to connect to PC using a RS-232 connection.



Step 1: Reboot the iPAC-8000 into Initial mode

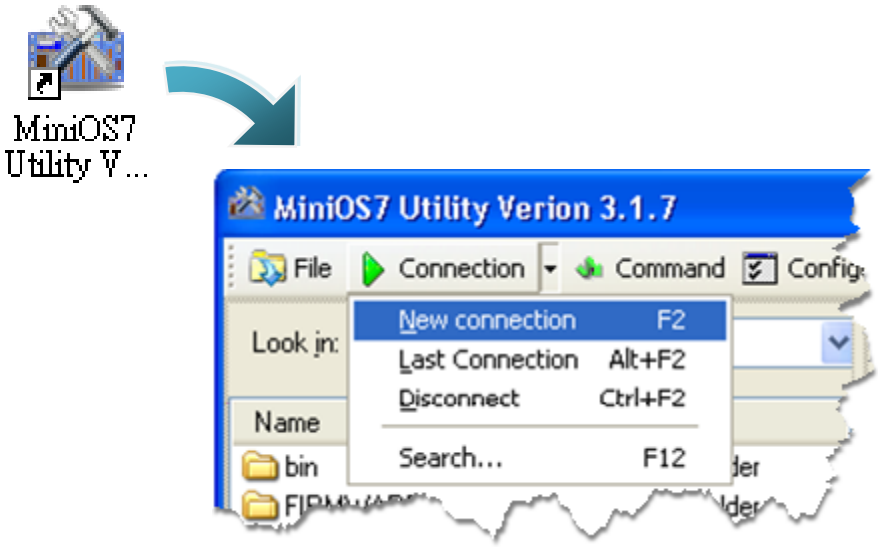
Make sure the DIP switch is in “init” position.

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

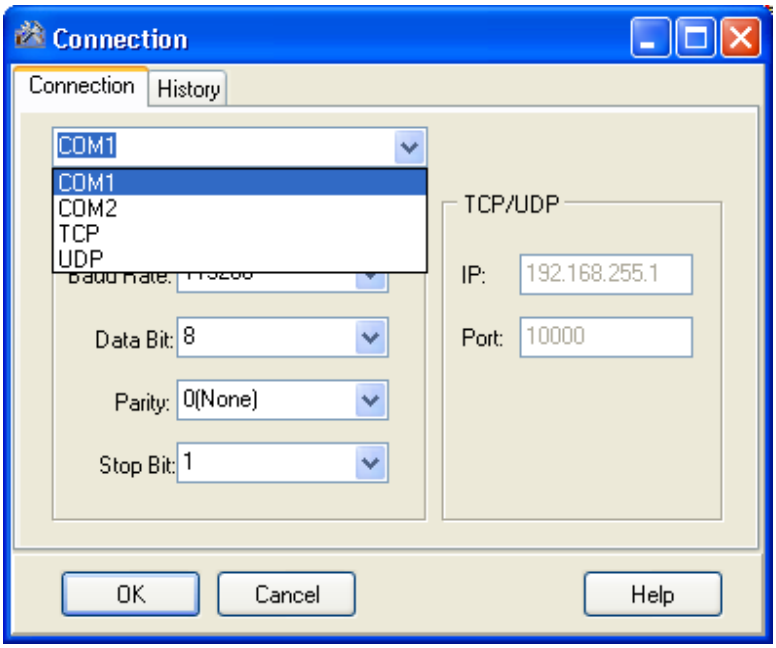


Step 3: Run the MiniOS7 Utility

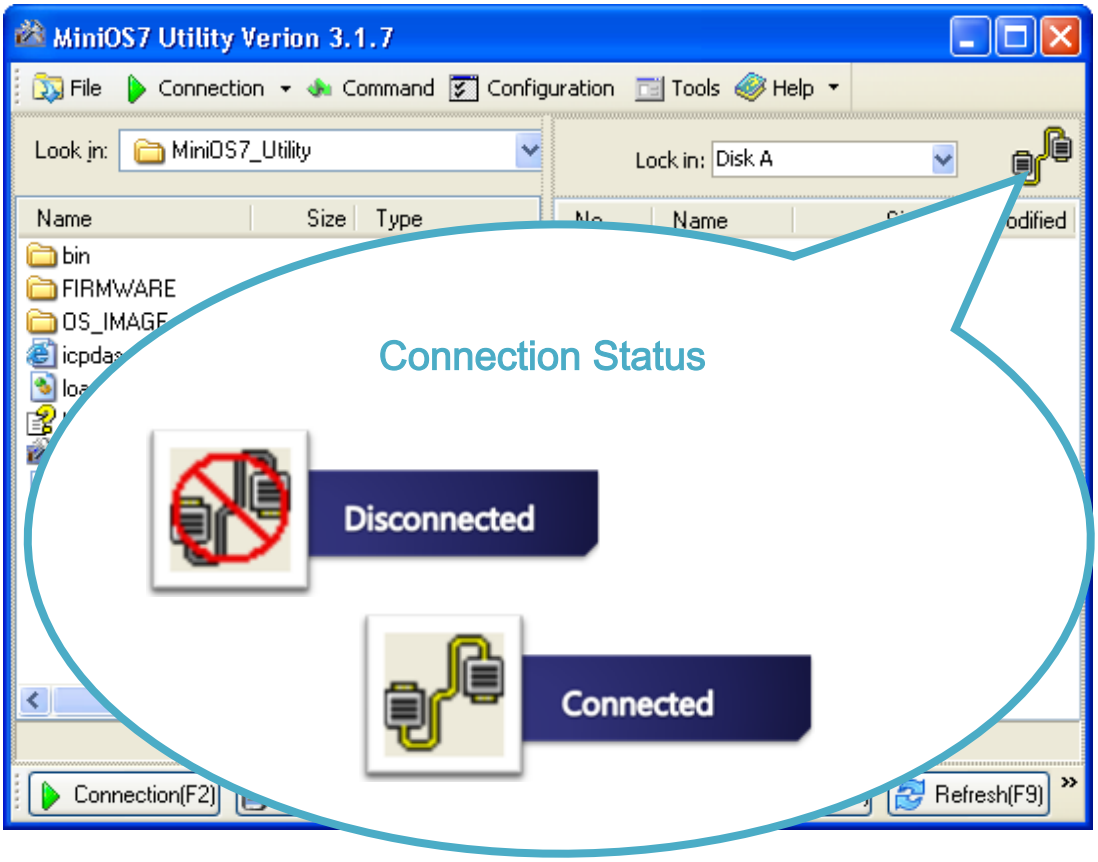
Step 4: Click the “New connection” function from the “Connection” menu



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

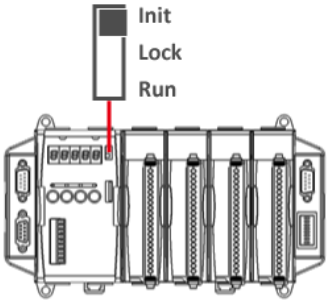


Step 6: The connection has already established



2.5.1.2. Using USB to Establish a Connection

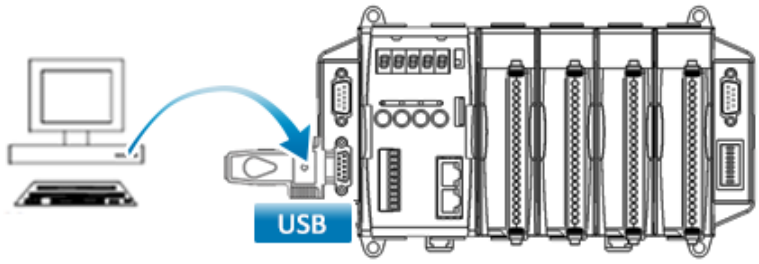
Below are step-by-step instructions on how to connect to PC using an USBconnection.



Step 1: Reboot the iPAC-8000 into Initial mode

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

Step 2: Use I-7560(Convertor) to connect to PC

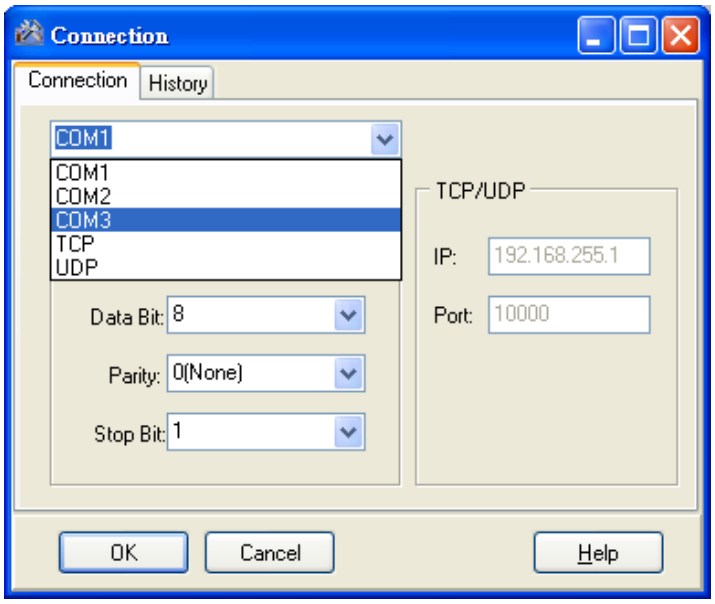


Before using the USB connection, ensure the I-7560 driver that you have installed. If they are not installed, please refer to “section 2.1.2. Wiring the iPAC-8000”.

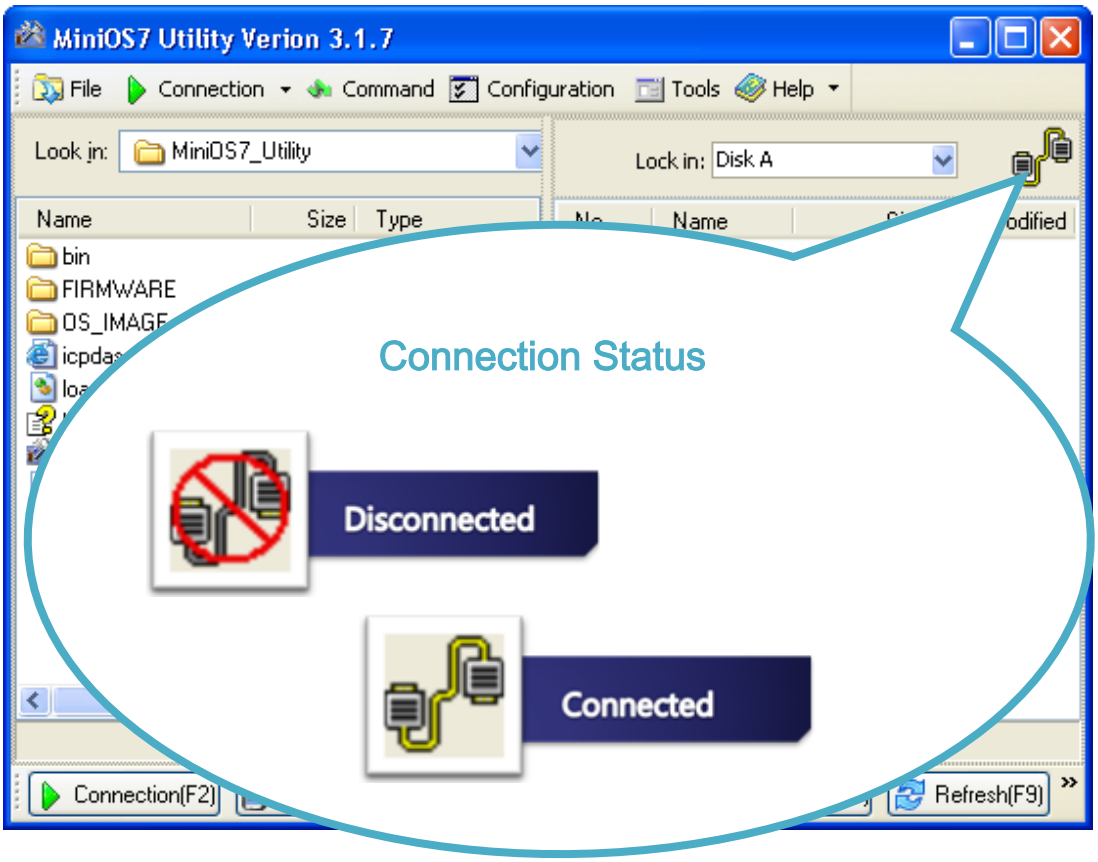
Step 3: Run the MiniOS7 Utility

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”

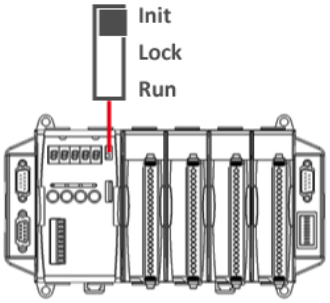


Step 6: The connection has already established



2.5.1.3. Using Ethernet to Establish a Connection

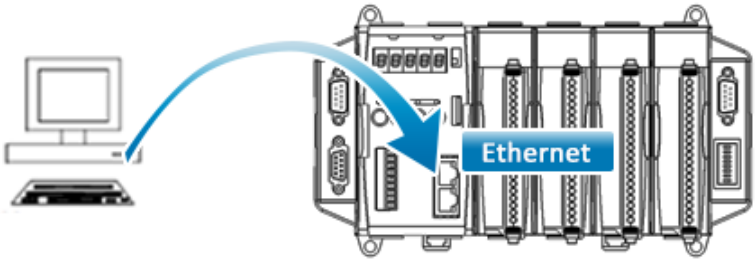
Below are step-by-step instructions on how to connect to PC using an Ethernet connection.



Step 1: Reboot the iPAC-8000 into Initial mode

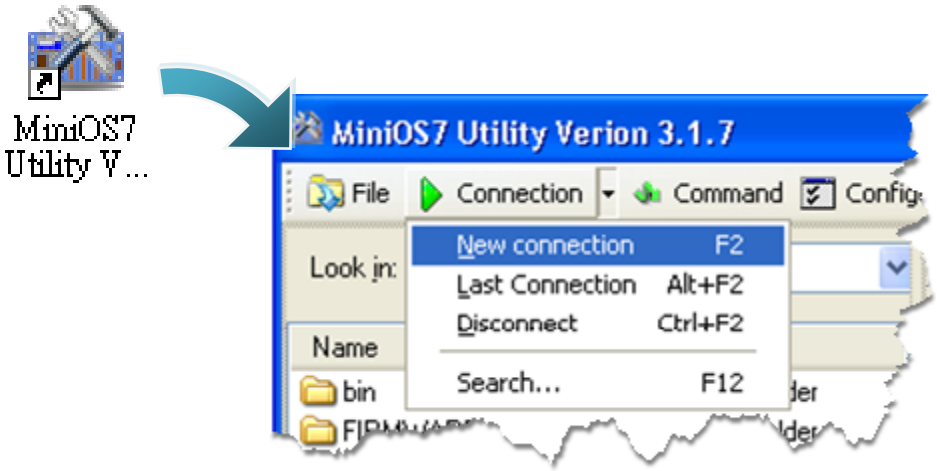
Make sure the DIP switch is in “init” position.

Step 2: Use the Ethernet to connect to PC

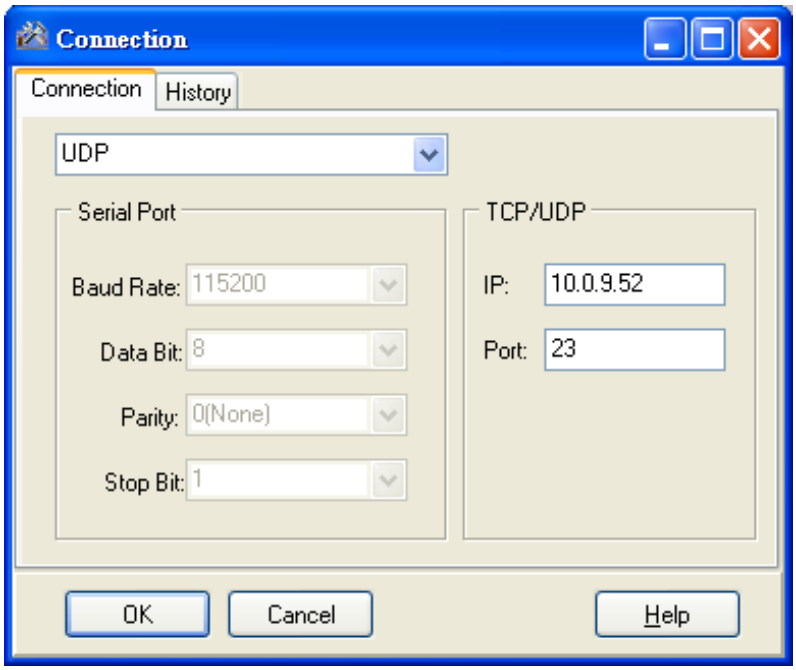


Step 3: Run the MiniOS7 Utility

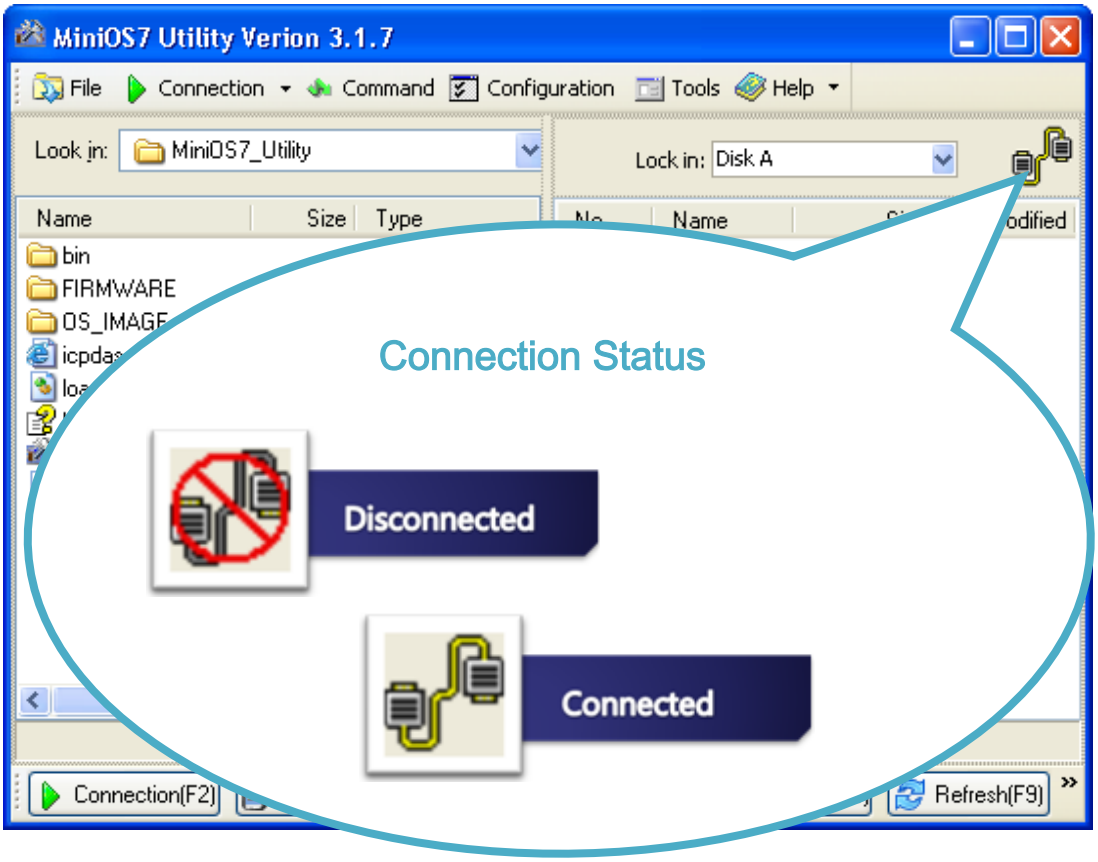
Step 4: Click the “New connection” function from the “Connection” menu



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”



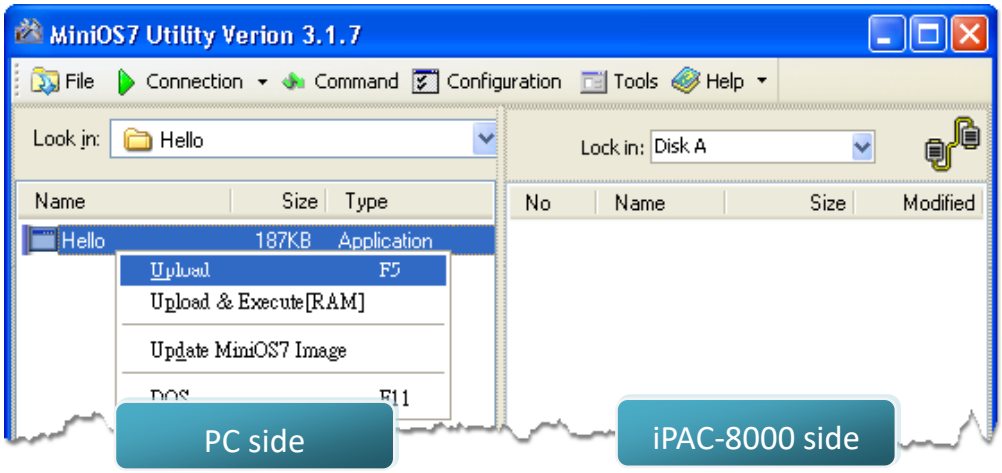
Step 6: The connection has already established



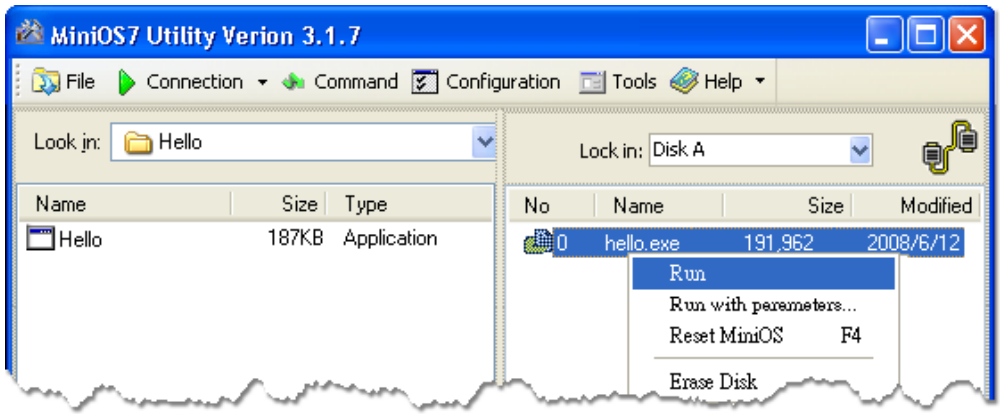
2.5.2. Uploading and Executing iPAC-8000 programs

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.

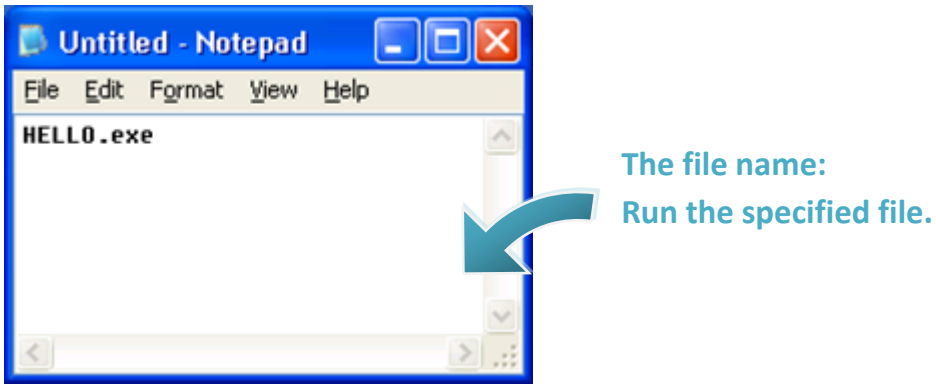
Step 1: Create an autoexec.bat file

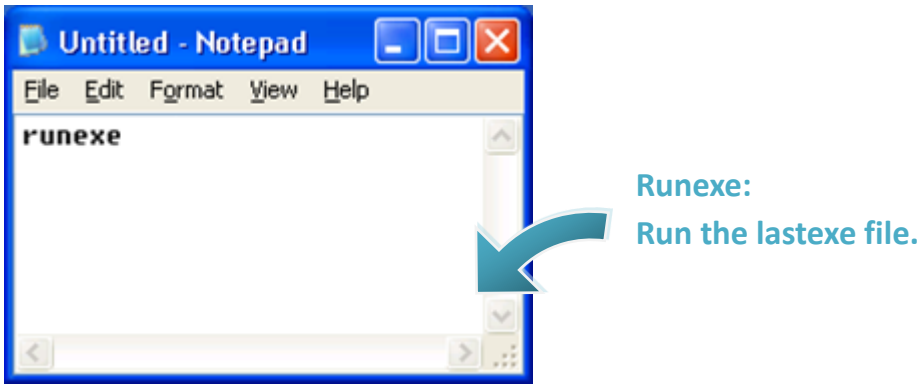
5. Open the "Notepad"

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)

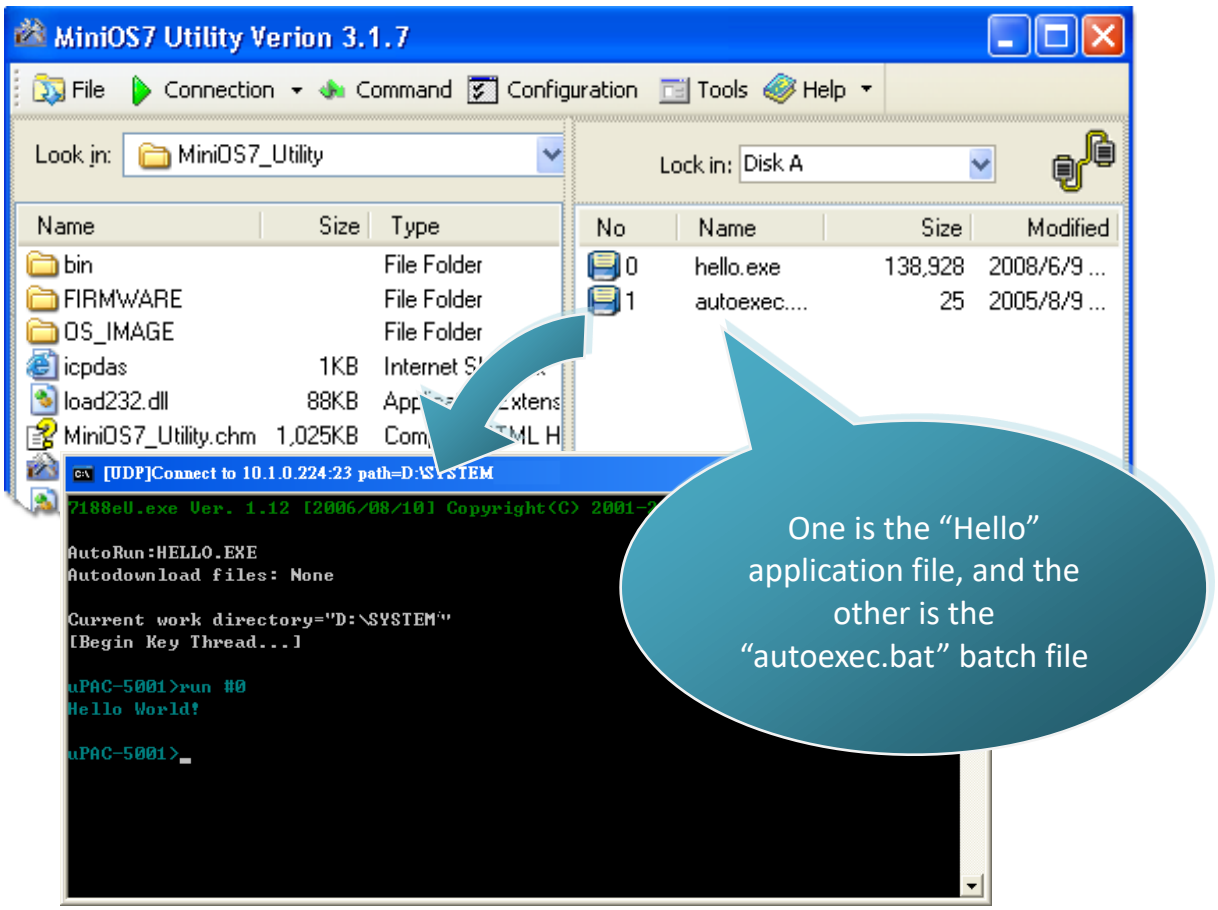
5. Save the file as autoexec.bat





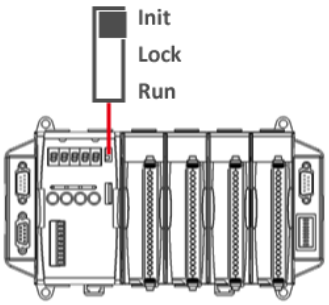
Step 2: Upload programs to iPAC-8000 using MiniOS7 Utility

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. Build First Program – Hello World

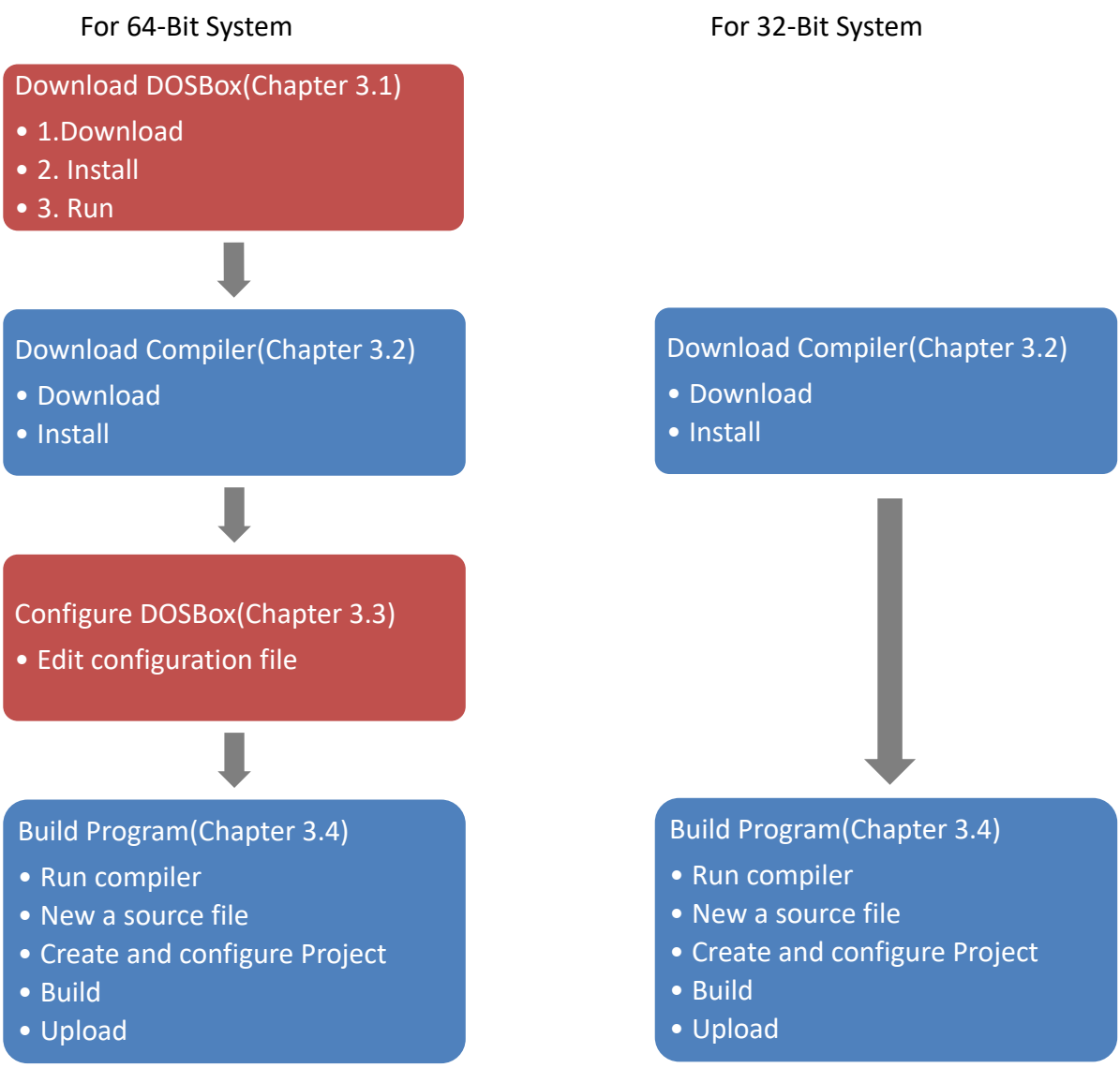
This section will tell you how to build a program for IP-8000 series controller.

Before start building the program, we need to figure out that which Operating System is being used on your computer, it is related to the process of developing the application.

If you are using a 32-bit Windows system such as Windows XP, please go to Chapter 3.2 to select the compiler and continue reading.

However, if you are using any 64-bit Windows system like Windows 7, Windows 8 or Windows 10, you will need an emulator, because the C compiler cannot be executed on these 64-bit systems directly.

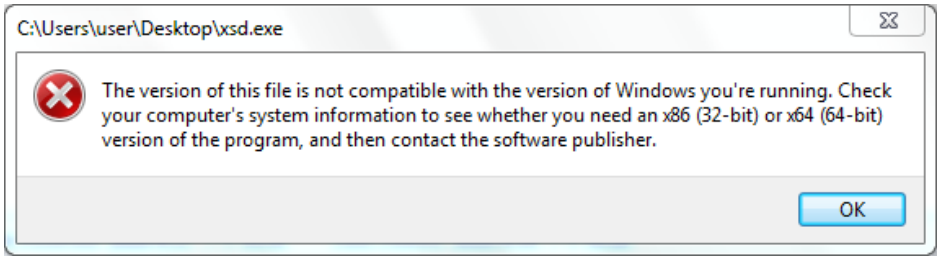
Please check the following flow chart to understand the process of developing the application.



3.1. Download DOSBox

64-bit systems do not support 16-bit programs, 16-bit processes or 16-bit components.

Therefore, if you are running a 16-bit compiler on 64-bit Windows (such as Borland C ++ 3.1, Turbo C ++ 3.0 or a program compiled by a 16-bit compiler), you may get an error message like below photo.



DOSBox is an excellent solution for running the C compiler on 64-bit systems.

DOSBox simulates the environment in which DOS and DOS used to run, including memory management and sound configuration, but has the functions of today's computers.

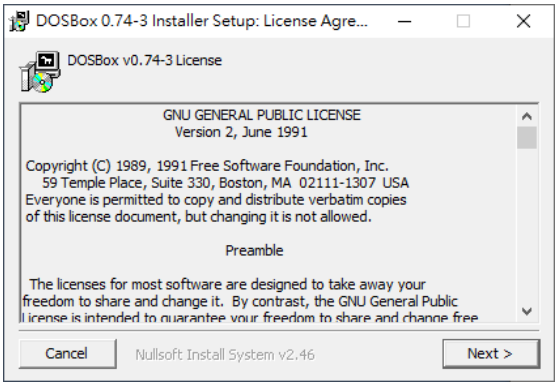
On this appendix we will show you how to execute the 16-bit compiler.

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/

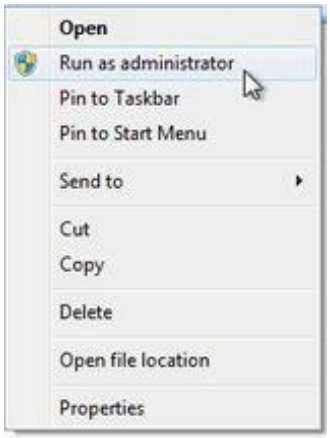
Step 2: Install the DOSBox

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



Step 3: Run the DOSBox

Right-click the DOSBox, and then click “**Run as administrator**”.



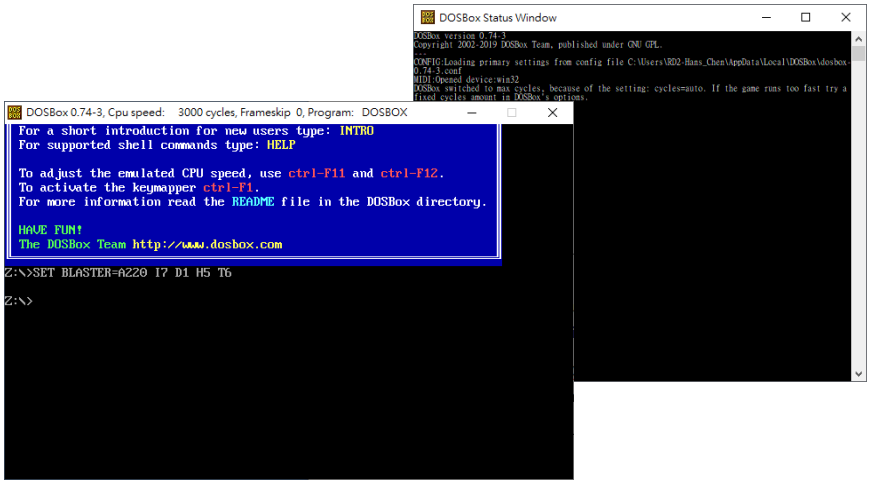
Tips & Warnings

If you do not execute DOSBox as an administrator, the installation of the compiler will fail.

Two windows will be displayed.

One is the DOSBox Status Windows,

And another one is the main execution window, which virtualizes a DOS operating environment.



3.2. Download 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.

Step 1: Download compiler

- Turbo C / C++(Borland Turbo C++ 1.01 (3.5))

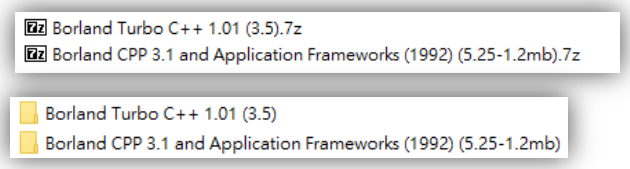
  https://winworldpc.com/product/turbo-c/1x

- Borland C / C++ (Borland CPP 3.1 and Application Frameworks (1992) (5.25-1.2mb))

  https://winworldpc.com/product/borland-c/30

Step 2: Extract the file and install compiler

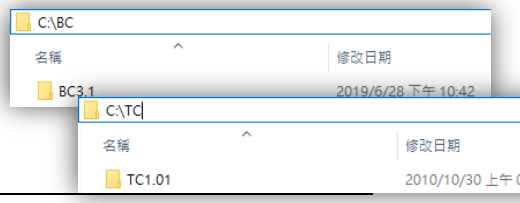
After you download the file, unzip it



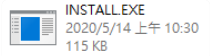
Tips & Warnings

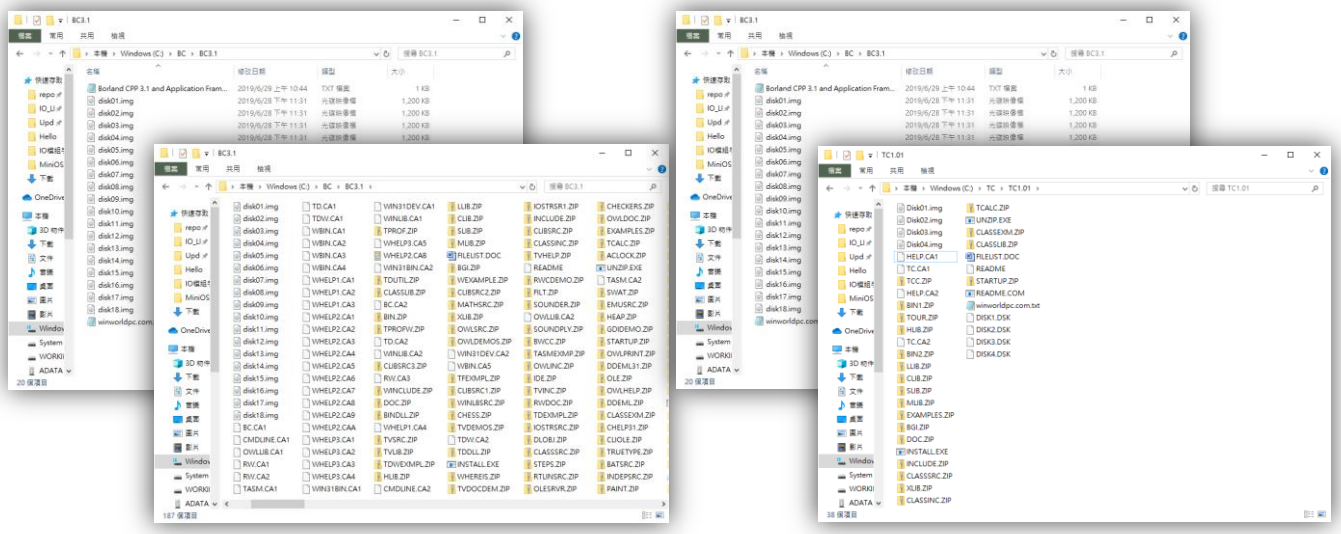
I am used to renaming the folder and moving it to disk c.

It will make the following installation steps easier.



Extract all .img files, there will be an “**install.exe**”.





For 32-bit system, you can just execute the “**install.exe**” and start to install the compiler.

But, for 64-bit system, you have to run this program in DOSBox, check the step 3 for more detail.

The following steps will use “Borland C” as an example.

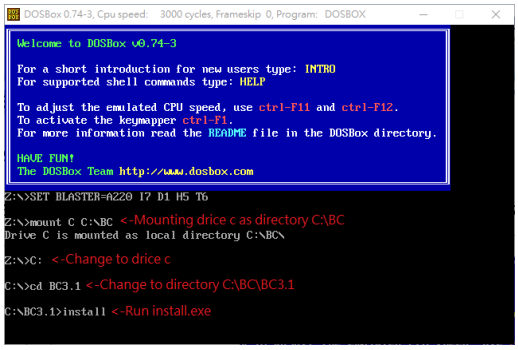
Step 3: Run the install.exe

For 32-bit system, run the install.exe will be fine.

Therefore, for 64-bit system, you need to run DOSBox and mounting a drive as a specific directory where you extract the files, then execute the “install.exe”.

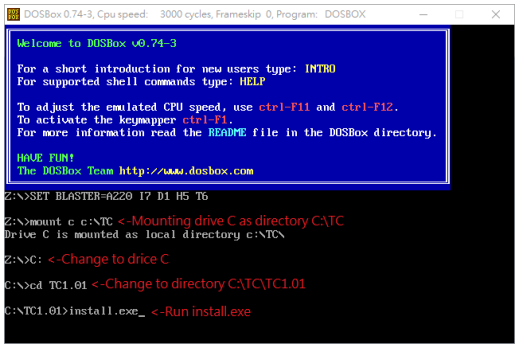
To install Borland C++, please type the commands below.

```
mount C C:\BC
C:
cd BC3.1
install.exe
```



To install Turbo C++, please type the commands below.

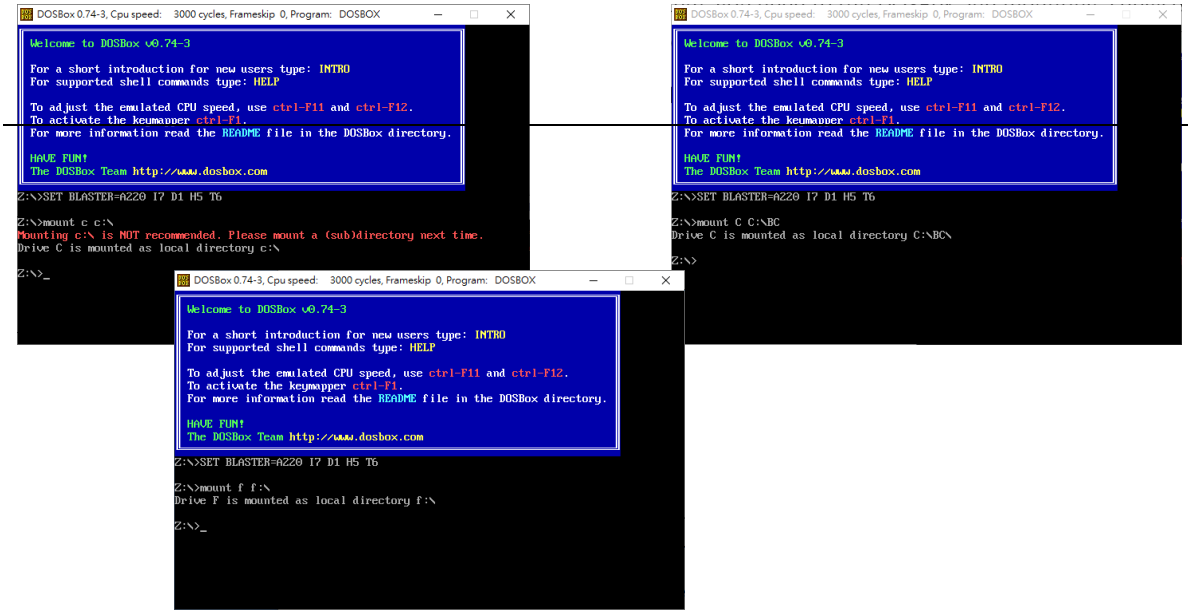
```
mount C C:\TC
C:
cd TC1.01
install.exe
```



Tips & Warnings

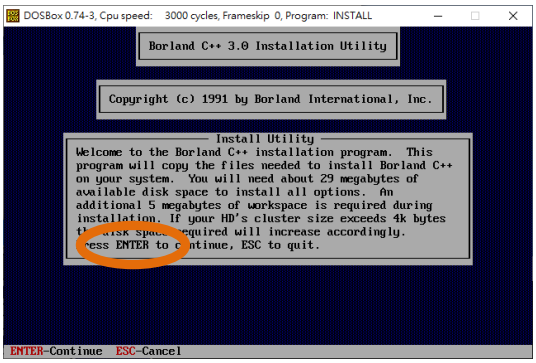
In some cases, the path “C:\” may not able to be mounted.

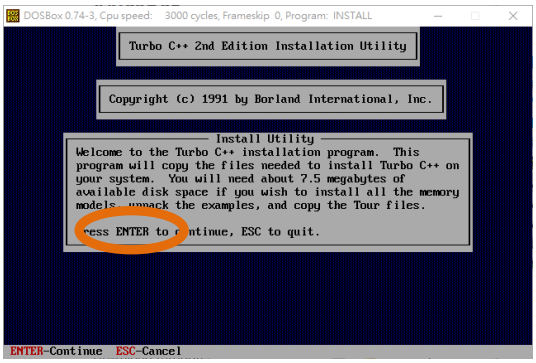
Try to type a different path like “C:\xxx”, or mount another disk will be fine.



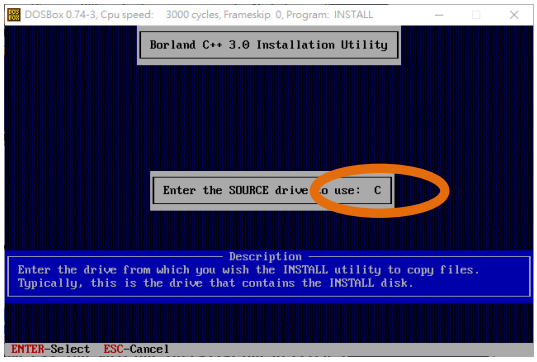
The following left side photos are screenshots of the Borland C++ installation process.
And the right side photos are screenshots of the Turbo C++ installation process.

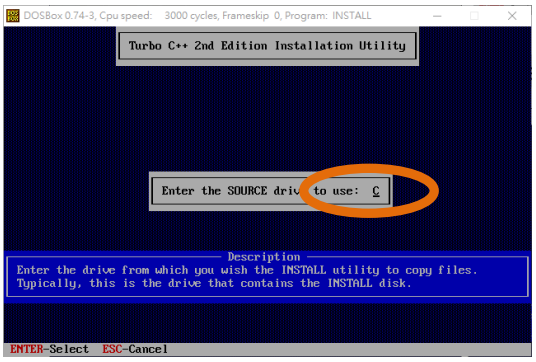
Step 3: Press “Enter” to continue



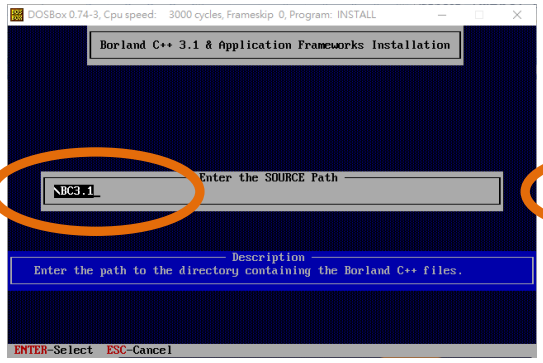


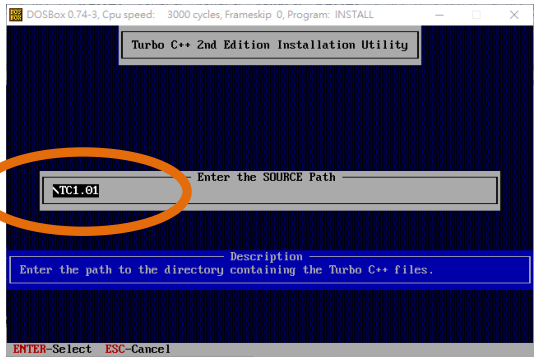
Step 4: Type the drive which the files are located



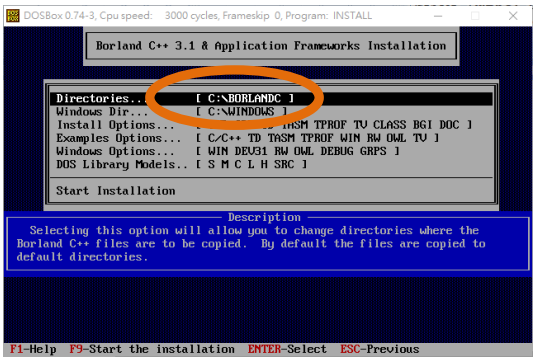


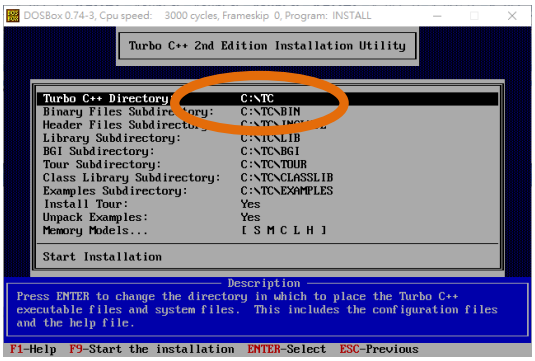
Step 5: Enter the path of the directory that the files are located

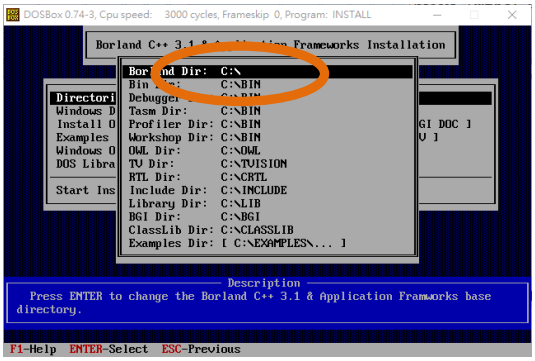


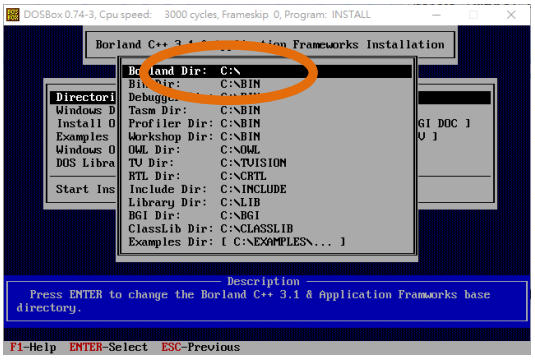


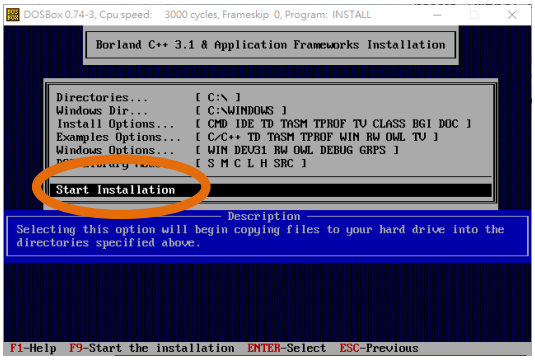
Step 6: Change directory to C:\ and Start installation

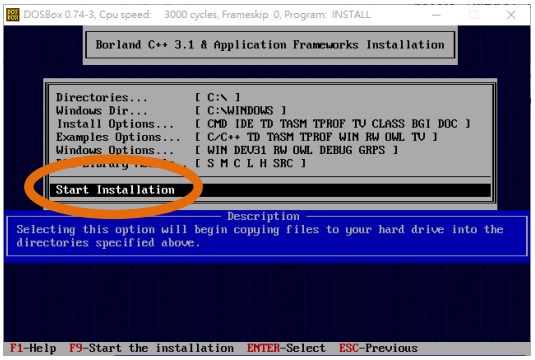




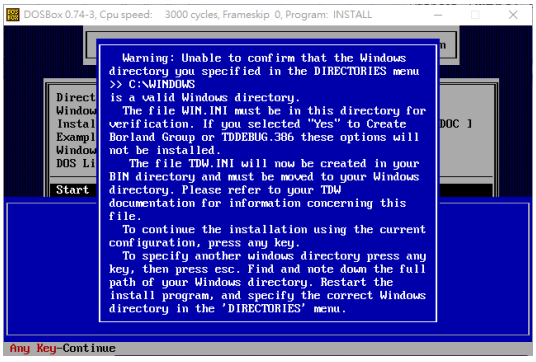


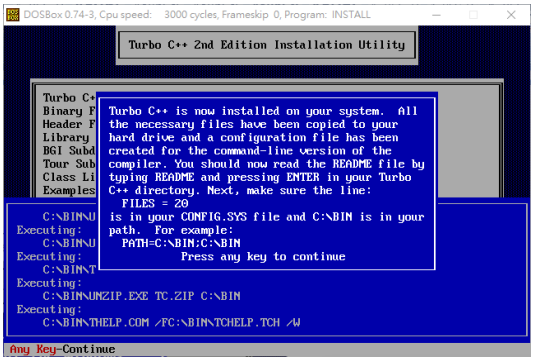


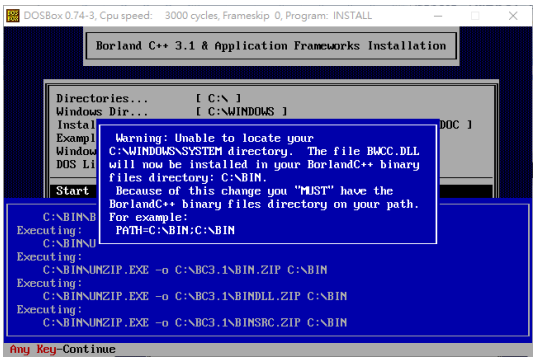


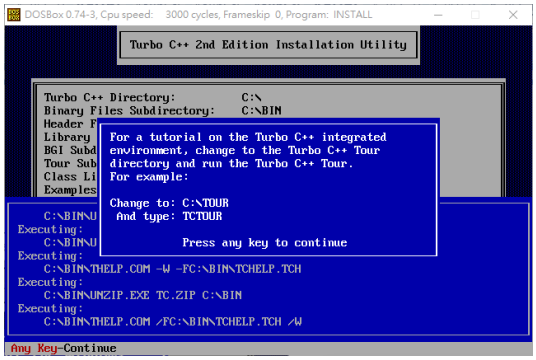


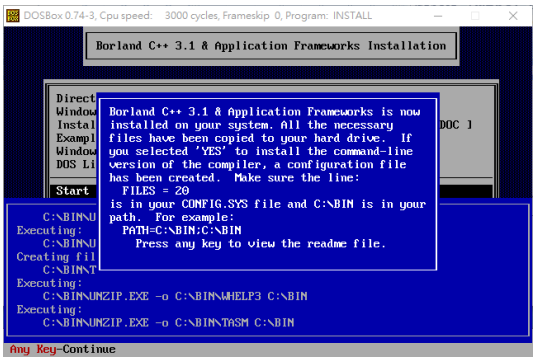
Step 7: Press any key to continue

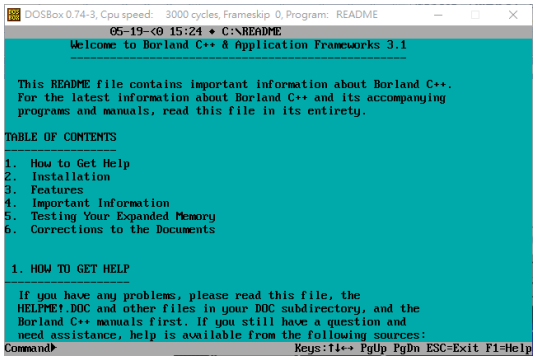




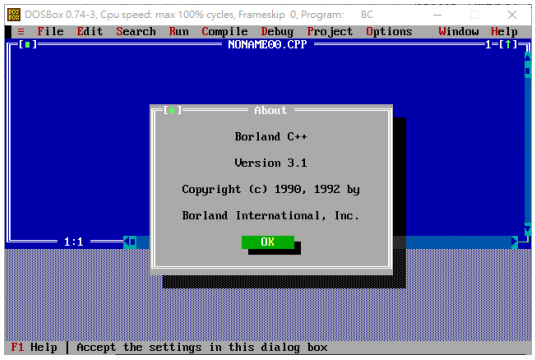


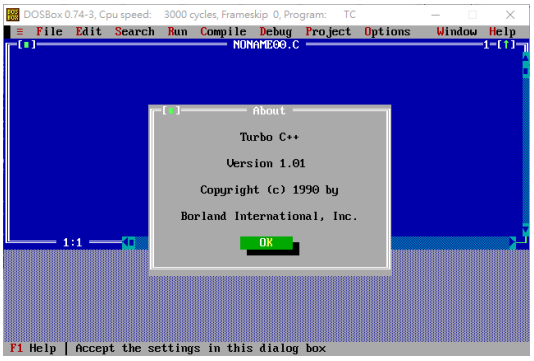






Step 8: Installation is complete





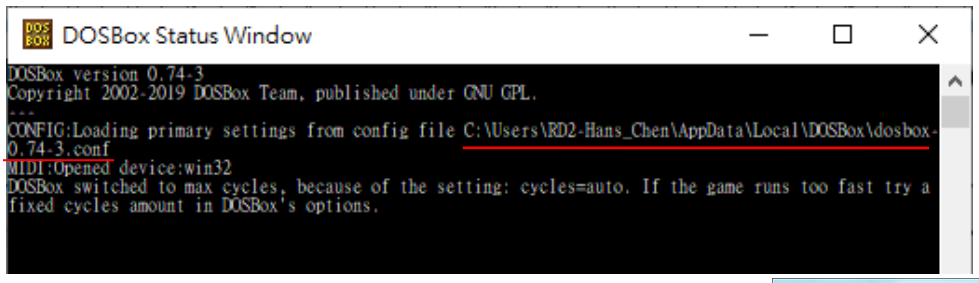
3.3. Configure DOSBox

Edit the configuration file to automatically run the compiler every time the DOSBox is executed.

The following steps will take Borland C++ as an example.

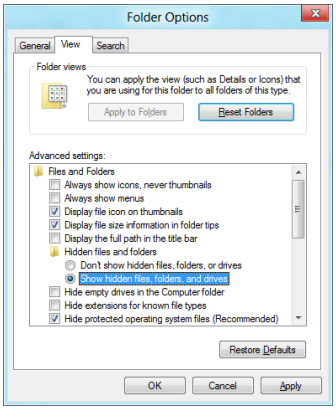
Step 1: Edit the configuration file of the DOSBox

The path of configuration file can be found in the Status Windows.



Tips & Warnings

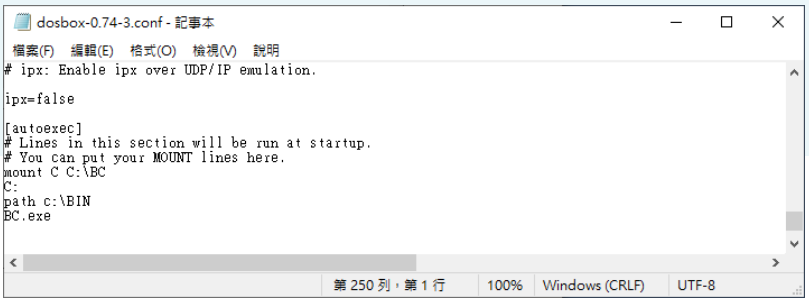
If you can't find the configuration file in the location, you'll have to enable the "**show hidden files**" option in the window properties to display the hidden files.



Step 2: Scroll down to the bottom of the configuration file and Type the commands to mount the drive and set path

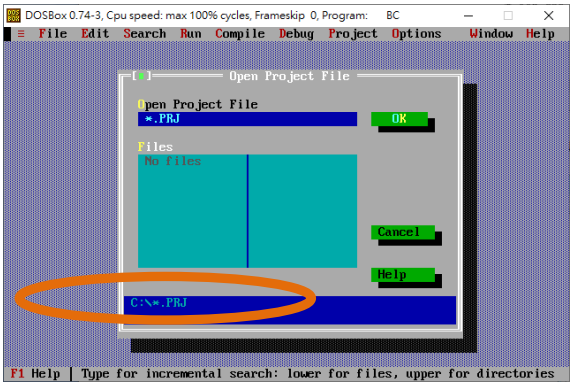
For example:

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



Save the configuration file and restart DOSBox.

It will automatically execute the compiler.

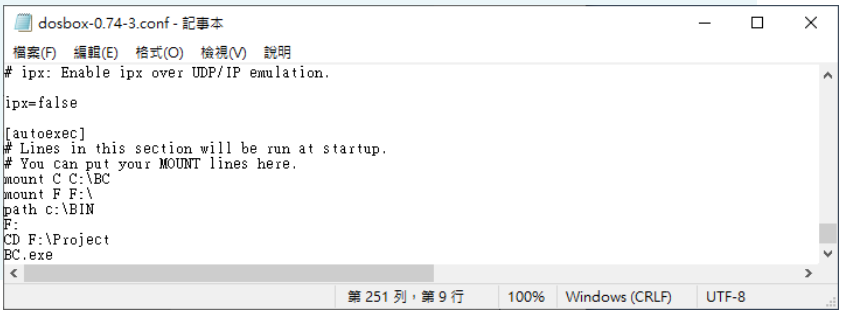


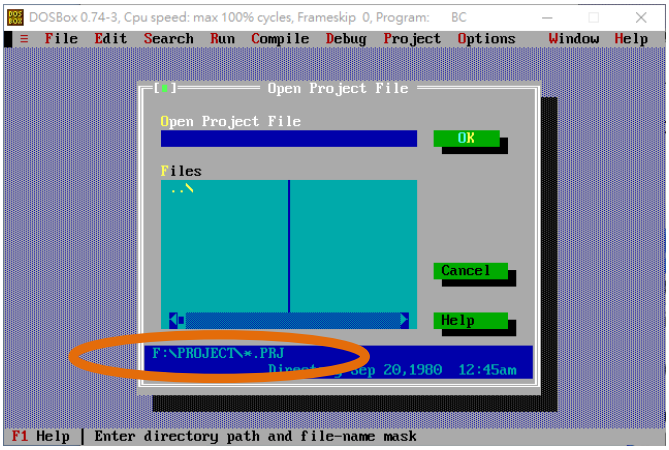
Tips & Warnings

If you want to specify the directoey every time you start the DOSBox.

Add “CD” command to change the directory.

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





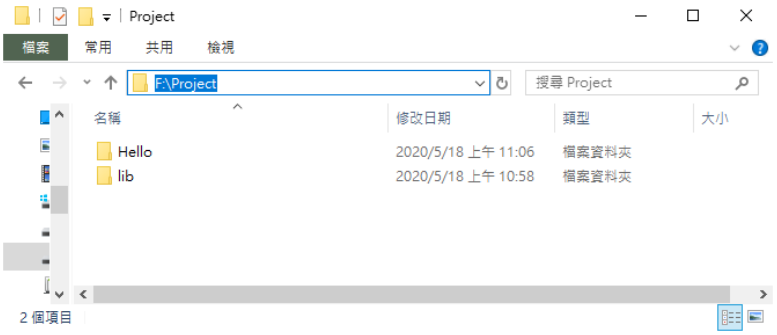
3.4. Build Program

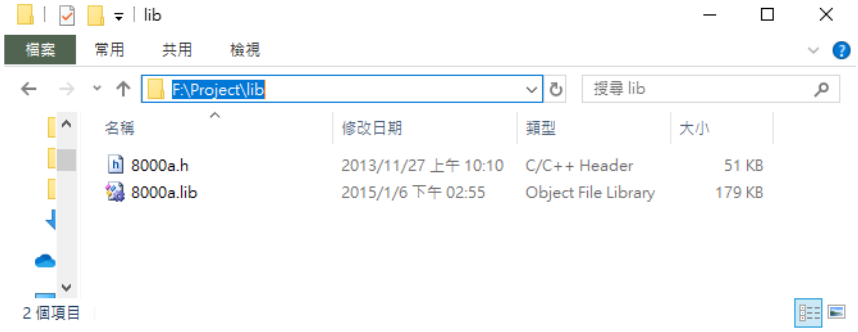
There are several APIs for customizing the standard features and integrating with other applications, devices and services.

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”.

Here we assume you have installed the Borland C++ under the C driver root folder and the iPAC-8000 APIs under the F driver root folder.

Then create a “Hello” folder for the following demonstrate.

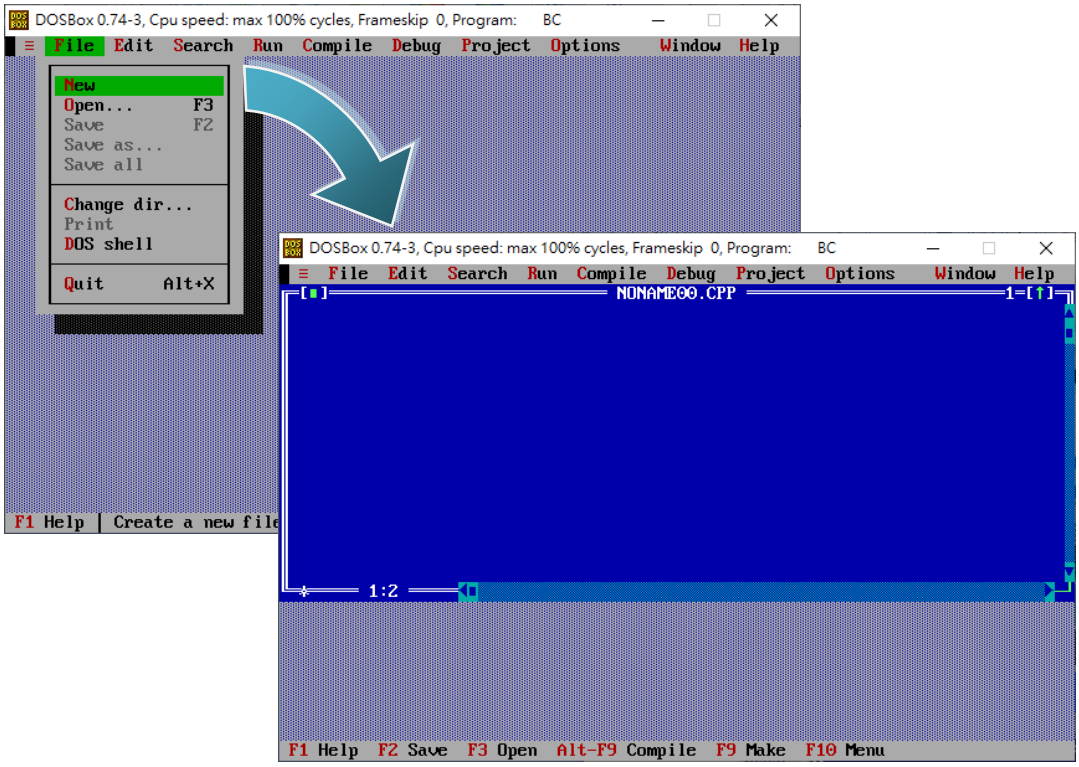




Step 1: Run DOSBox and Compiler

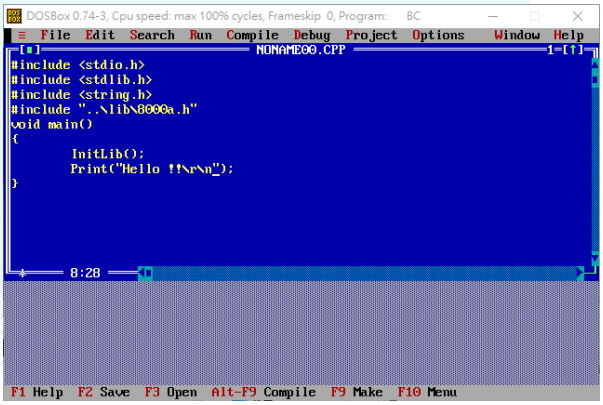
After edit the configuration file, the compiler should be executed automatically.

Step 2: Select “New” from the “File” menu to create a new source file



Step 3: Type the following code. Note that the code is case-sensitive

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include “..\lib\8000a.h”
/* Include the header file that allows 8000a.lib functions to be used */
void main(void)
{
      /* Initiate the 8000a library */
      InitLib();
      /* Print the message on the screen */
      Print(“Hello !!\r\n”);
}
```



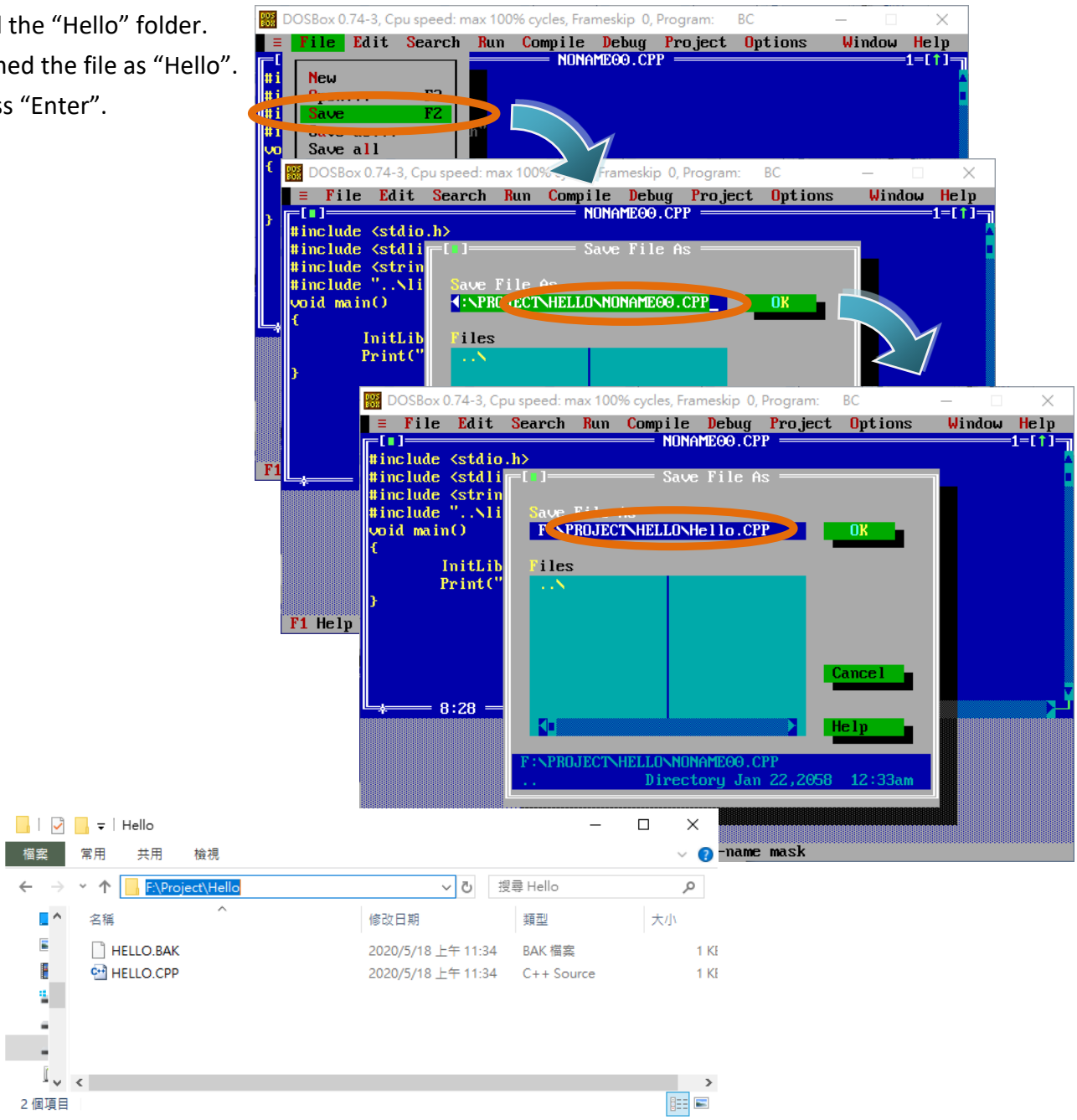
Step 4: Save the source file in the Hello folder

(1) Select “Save” from the “File” menu.

(2) Find the “Hello” folder.

(3) Named the file as “Hello”.

(4) Press “Enter”.



Tips & Warnings

You can write the code with any text editor that you are used to use.

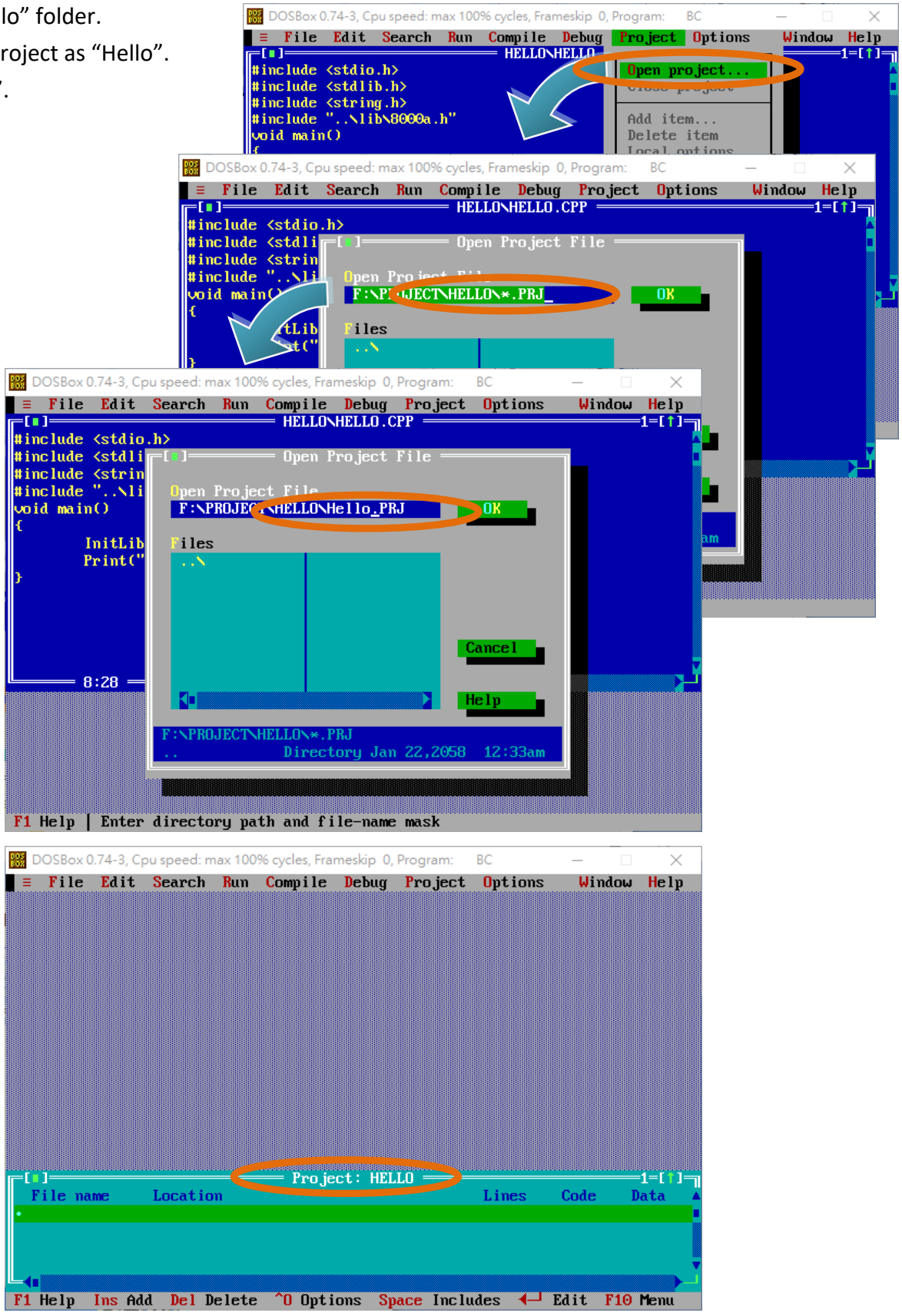
Step 5: Create a project (*.prj)

(1) Select “Open project…” from the “Project” menu.

(2) Find the “Hello” folder.

(3) Named the project as “Hello”.

(4) Press “Enter”.

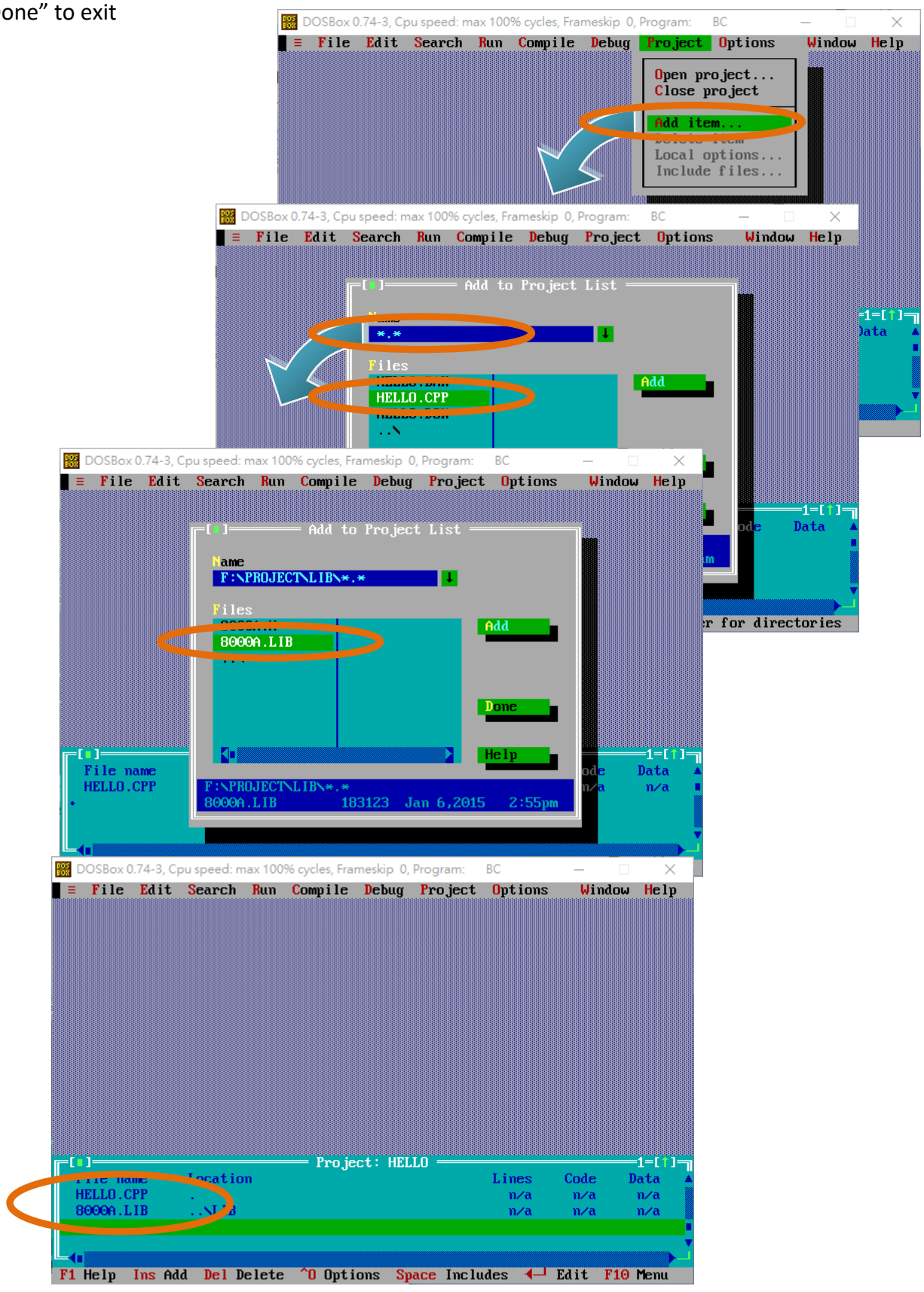


Step 6: Add the necessary function libraries to the project (*.lib)

(1) Select “Add item…” from the “Project” menu.

(2) Add the source file (hello.cpp) and library (8000a.lib).

(3) Select “Done” to exit

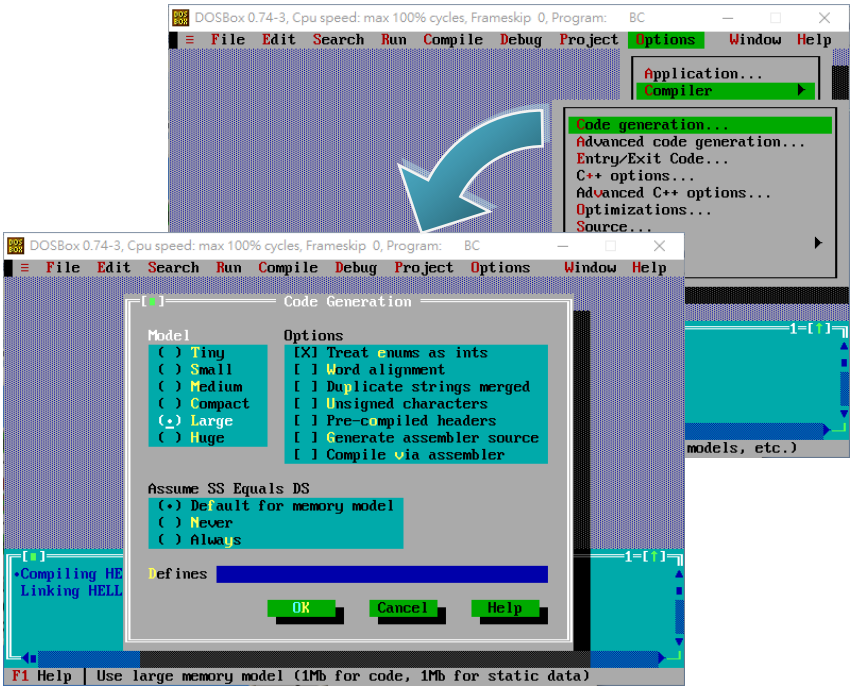


Step 7: Set the memory model to large

(1) Select “Compiler” from the “Options” menu and then select “Code generation…”.

(2) On “Model” option, select “Large”.

(3) Select “OK”.



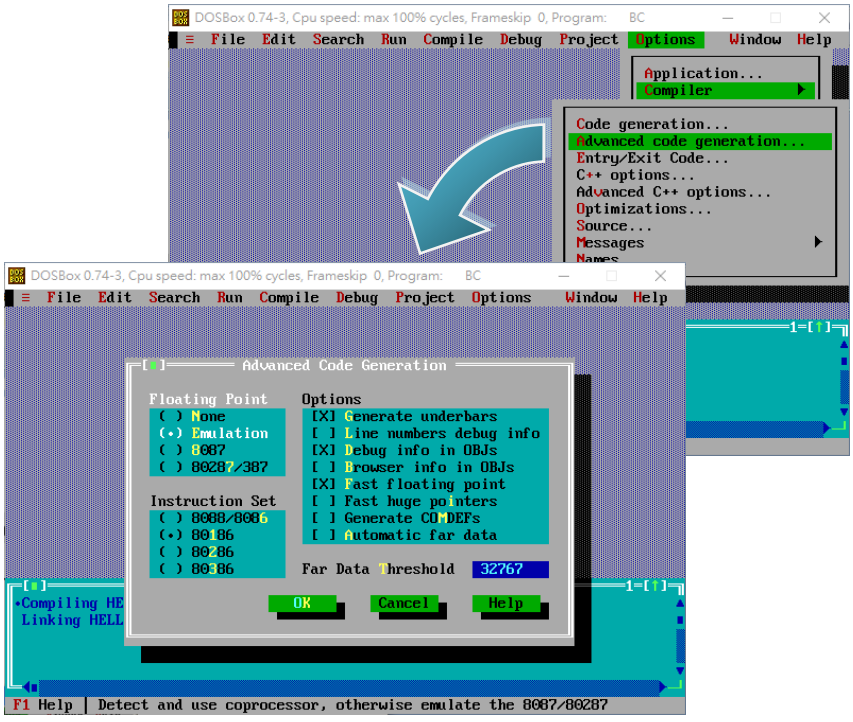
Step 8: Set the Floating Point to Emulation and the Instruction Set to 80186

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

(2) On “Floating Point” option, select “Emulation”.

(3) On “Instruction Set” option, select “80186”.

(4) Select “OK”.



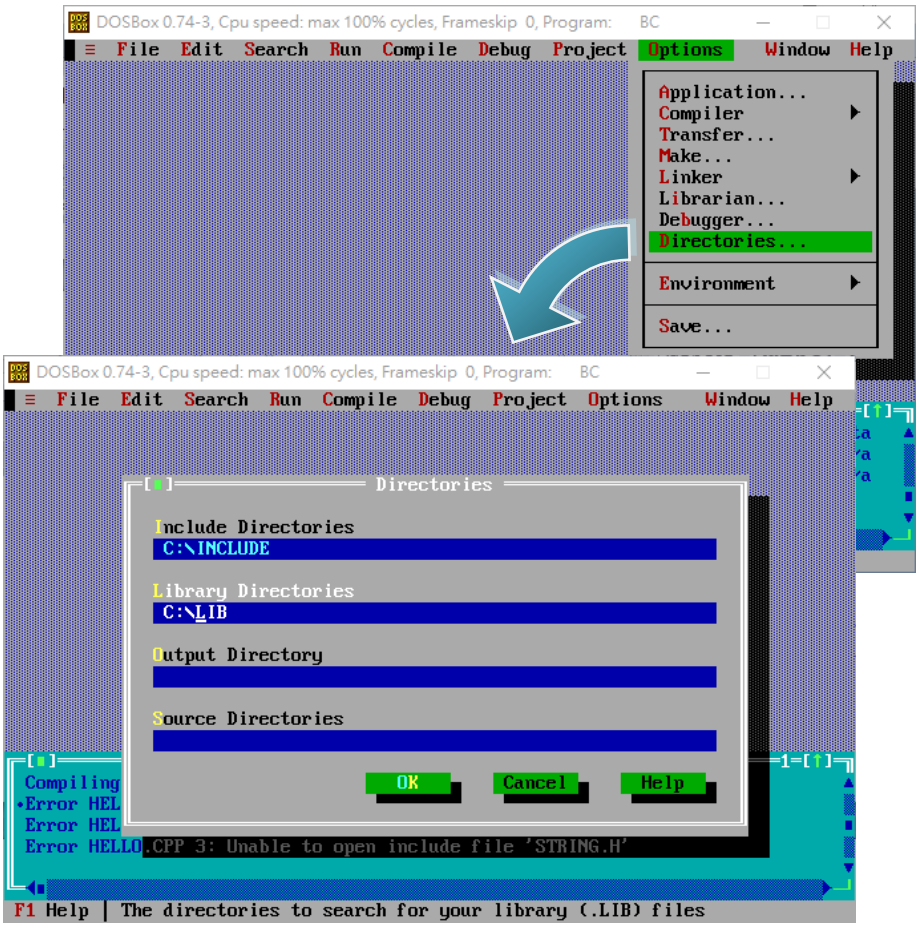
Step 9: Set the include and library directories

(1) Select “Directories…” from the “Options” menu .

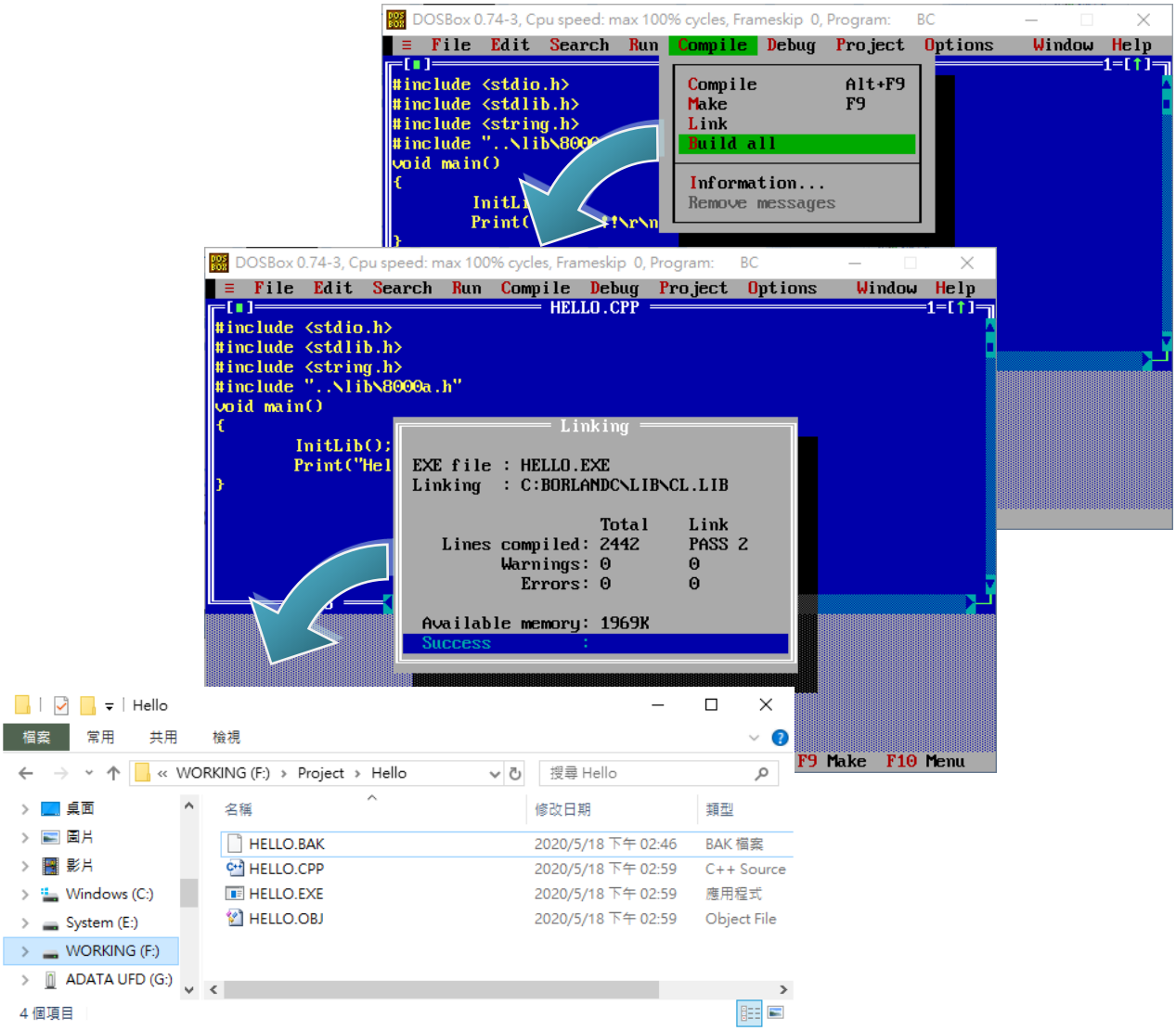
(2) On “Include Directories” option, specify the header file.

(3) On “Library Directories” option, specify the function library file.

(4) Select “OK”.



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



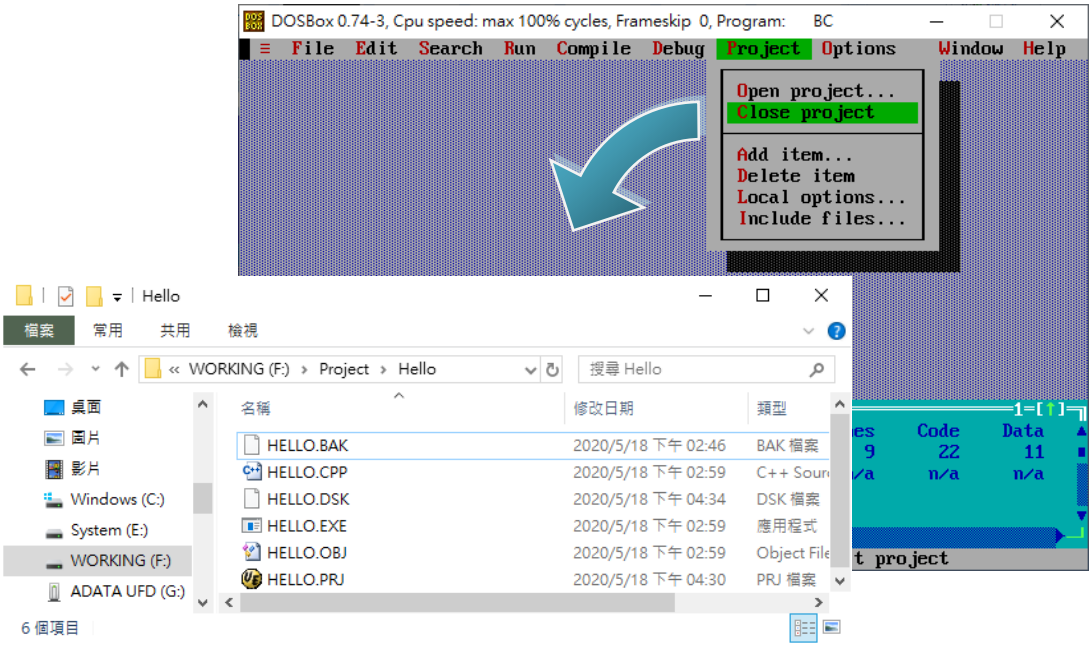
Tips & Warnings

The execute file will be named the same as the project name.

Step 12: Select “Close project” from the “Project” menu to close the project

Close the project before exit DOSBox or Compiler

The project file will be generated after the project is closed.



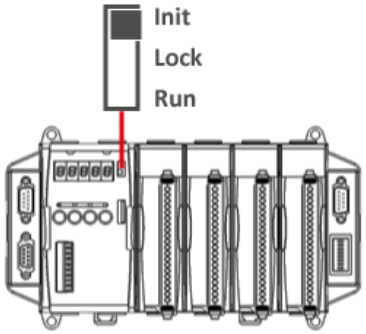
Tips & Warnings

Do not click the “X” at the right-top corner of the window before you close the project.

Otherwise, the project will not be generated or saved.

Step 12: Boot the iPAC-8000 into initial mode

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

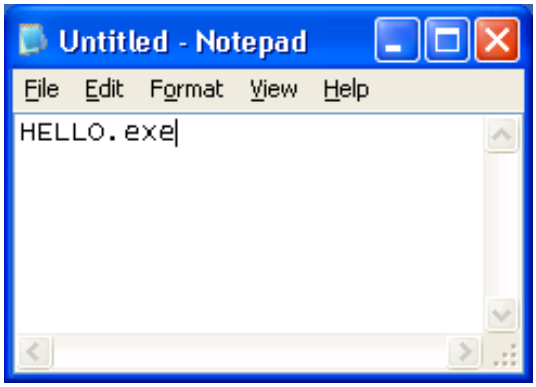


Step 13: Create an autoexec.bat file

(1) Open the “Notepad”

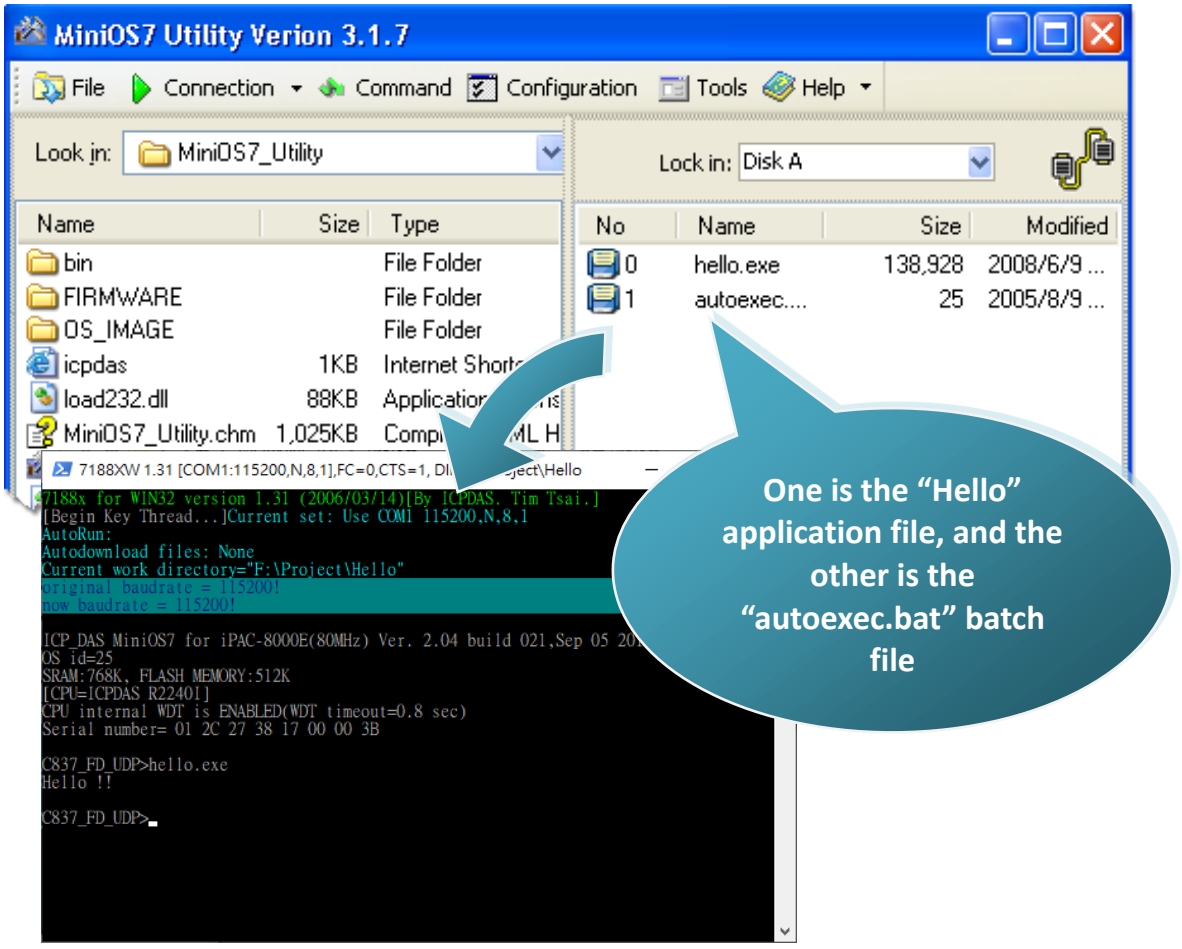
(2) Type the “HELLO.exe”

(3) Save the file and named it 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”



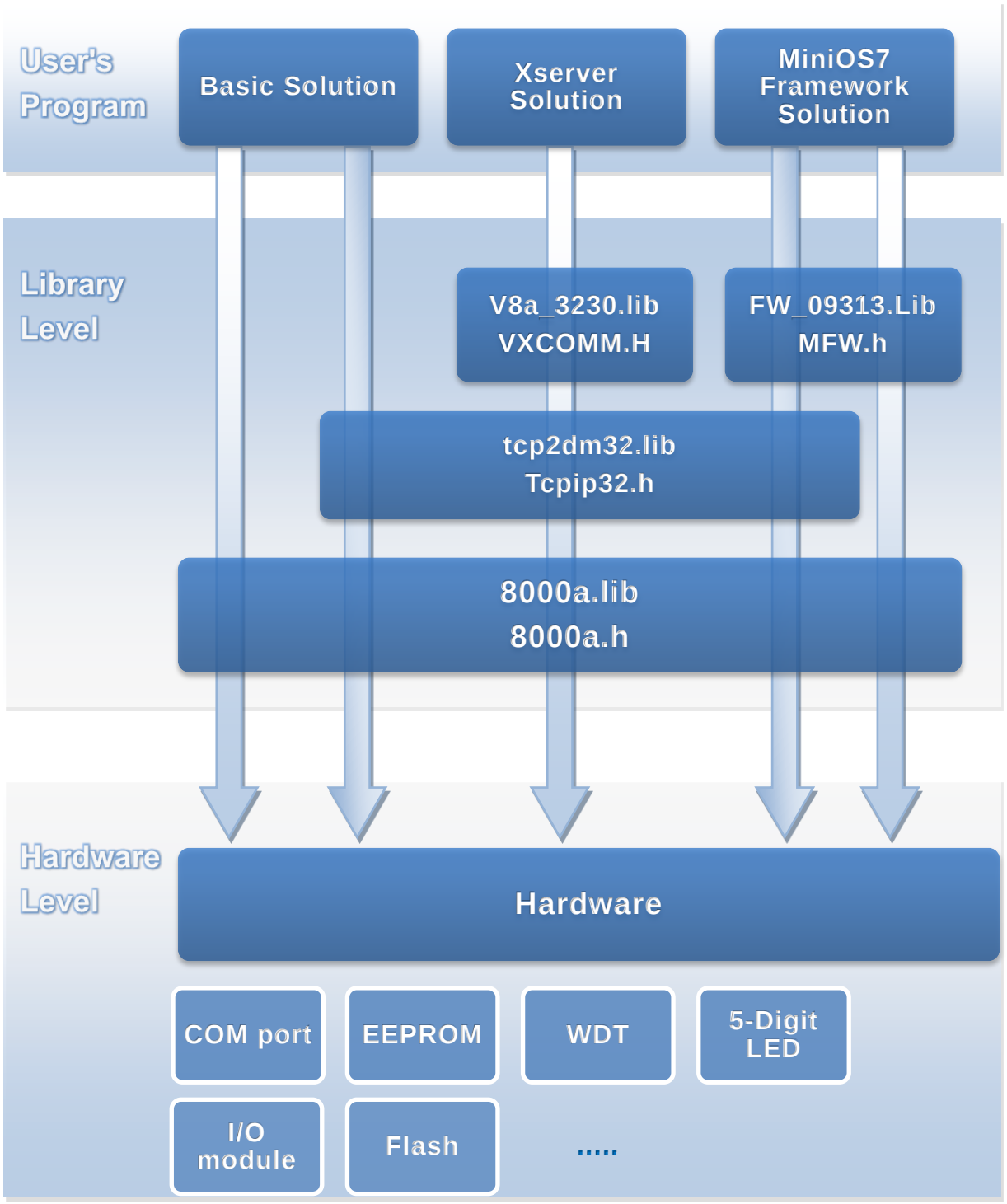
4. APIs and Demo References

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_V105.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.

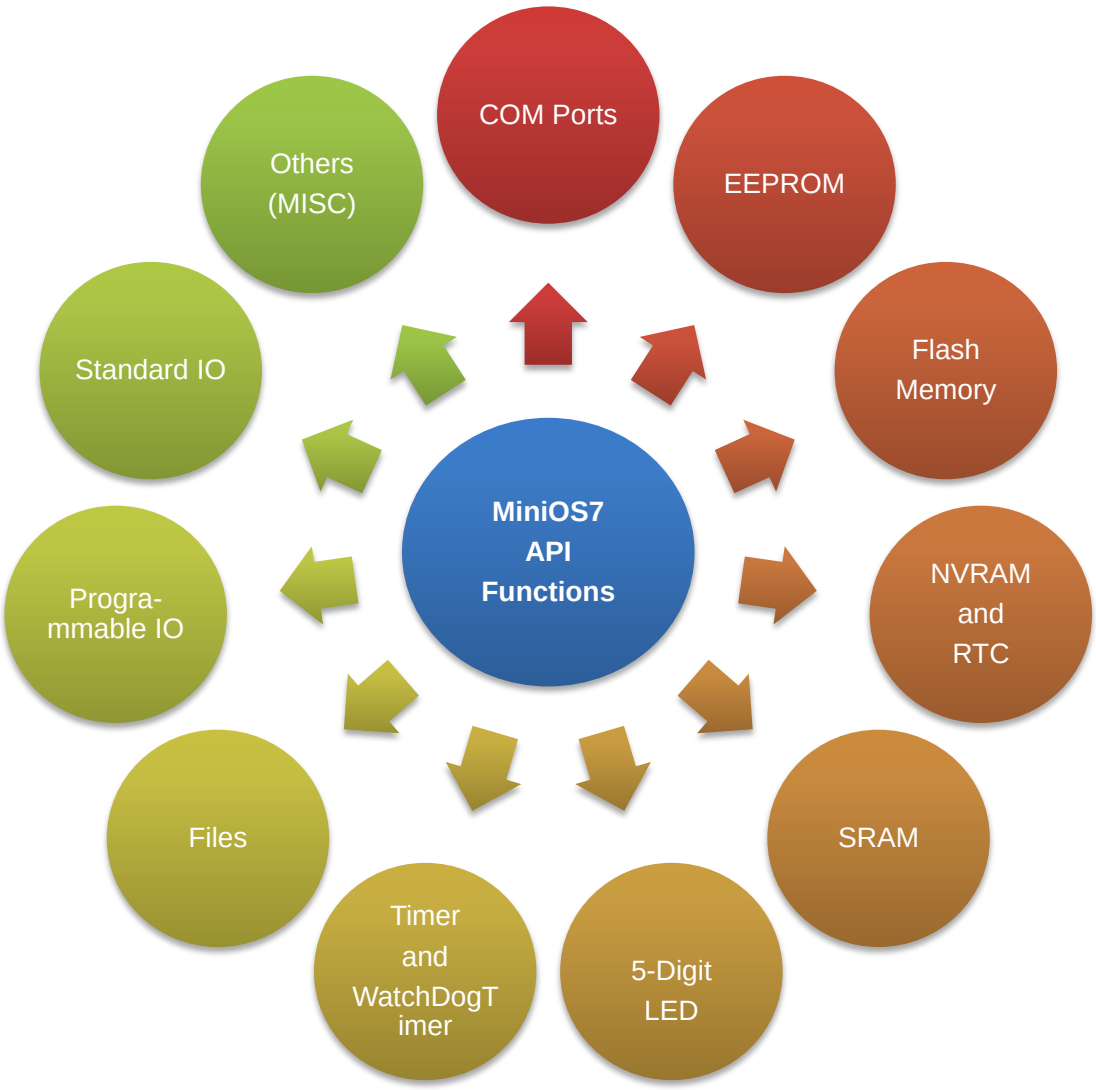
The following provides a brief introduction to the functions of MiniOS7 API.

Functions Library – 8000a.lib

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

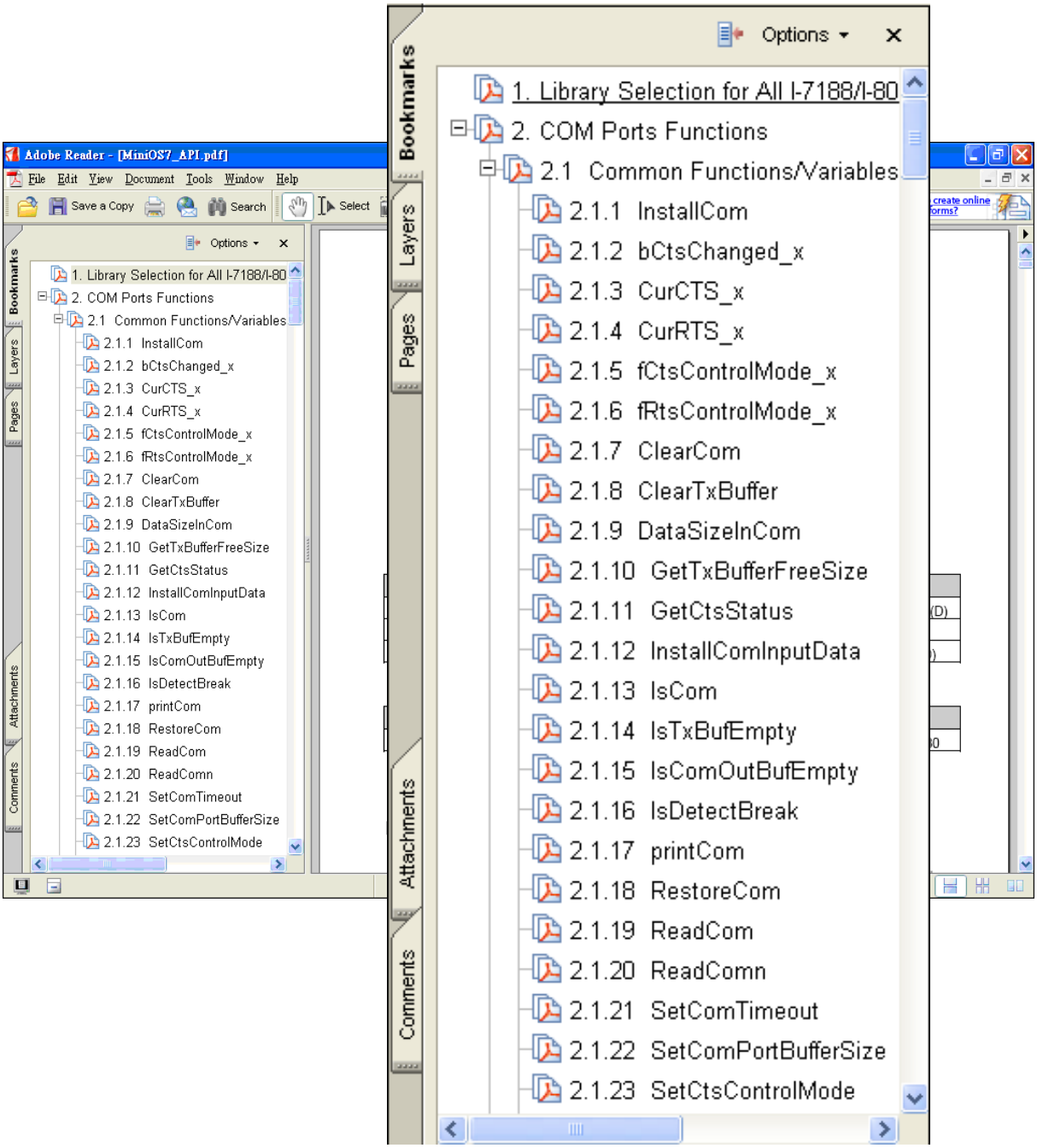
Header File – 8000a.h

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:

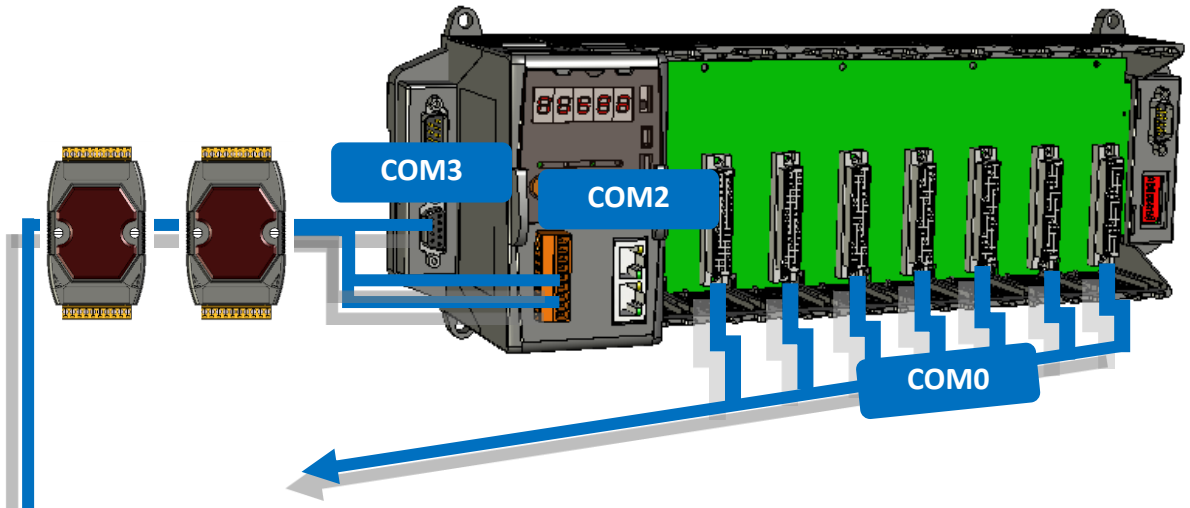
https://www.icpdas.com/en/download/show.php?num=1057&model=iP-8411



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: https://www.icpdas.com/en/download/show.php?num=3065&model=iP-8411		



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

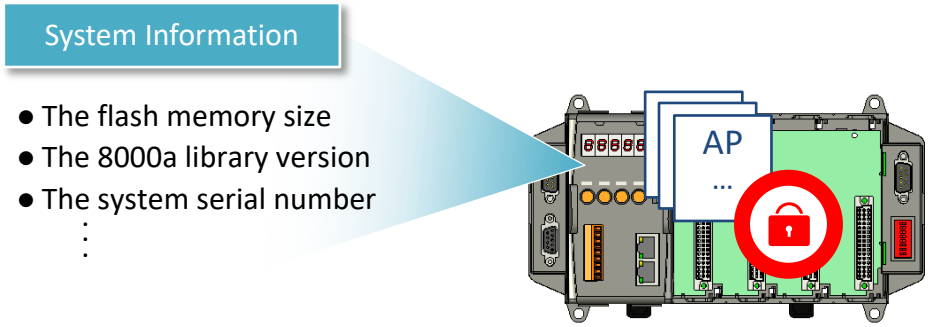
Folder	Demo	Explanation
IO_in_Slot	8K_DI	This demo program is used by 8K series DI modules, such as 8040, 8051., etc.
	8073	This demo program is used for 8073 General Functions.
	87K_DI	This demo program is used by 87K series DI modules in Com0, such as 87040, 87051, etc.
	87024	This demo program is 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 controlI-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	

4.1.1. MiniOS7 API for Library initialization and General Functions

- The MiniOS7 API provides a number of functions that you'll find useful for retrieving information about iPAC-8000 system, the hardware, and other hardware that may be running on the same network. This information is often useful for avoiding problems.
- The iPAC-8000 has a 64-bit hardware serial number that is unique and individual. The user can add a checking mechanism to their application program to prevent software from pirating.



MiniOS7 API has a variety of functions available. The following provides a brief introduction to some of the most common functions.

API for initiating the 8000a library

InitLib()

This is an important function. It must be called before any other functions are called in the 8000a library.

API for checking the flash memory size

Is8000()

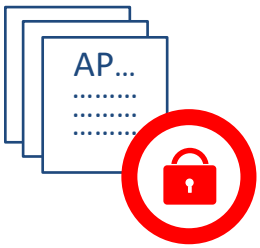
The Is8000 () have to be called in order to check the flash memory size.

API for getting the version of the 8000a library

GetLibVersion()

The GetLibVersion() have to be called in order to get the version of the 8000a library.

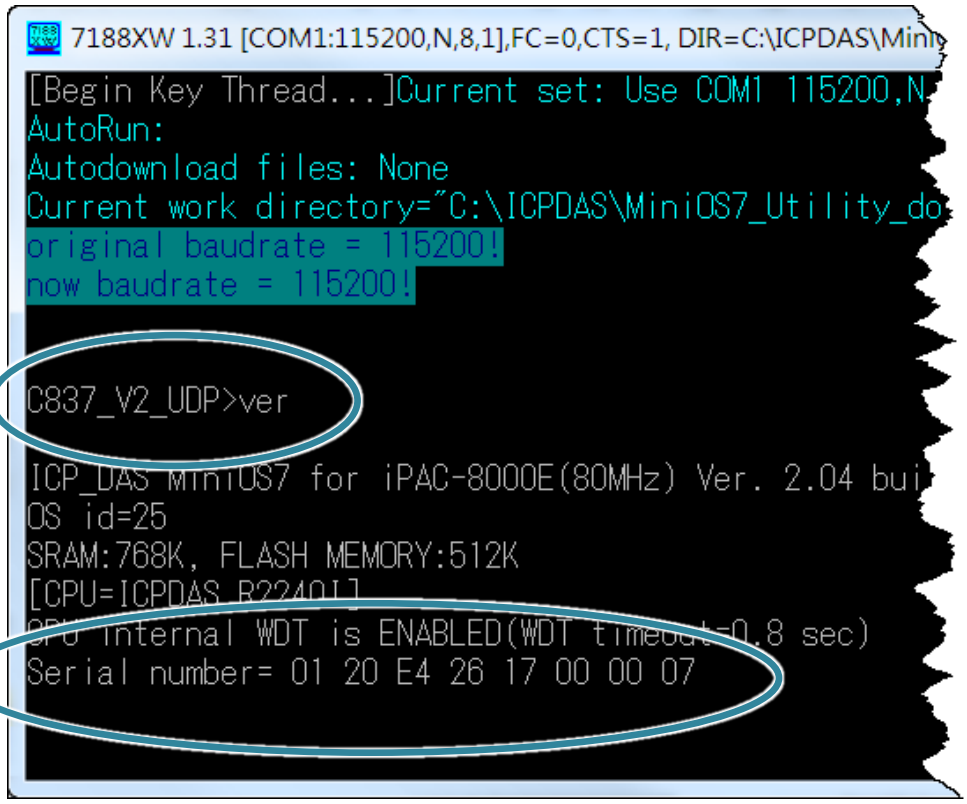
5. GetSerialNumber()



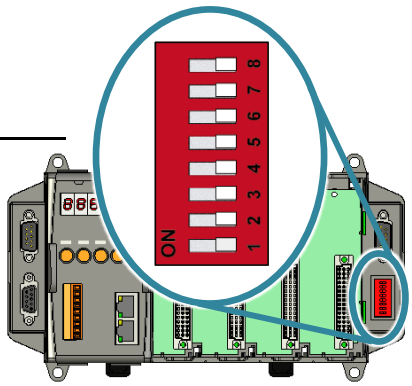
The GetSerialNumber() have to be called in order to prevent software piracy and ensure that the software is installed for an intended iPAC-8000 only.

Tips & Warnings

- The 64-bit system serial number is unique and individual. When the application program read the system serial number by calling GetSerialNumber (), the application program can only run on the first iPAC-8000.
- The 64-bit serial number can also be checked by using a MiniOS7 command “VER”.



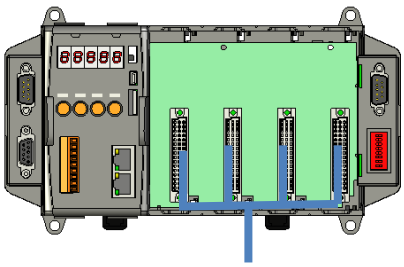
API for getting the Net ID



GetNetId()

The GetNetId () have to be called in order to get the Net ID.

API for getting number of I/O Slots

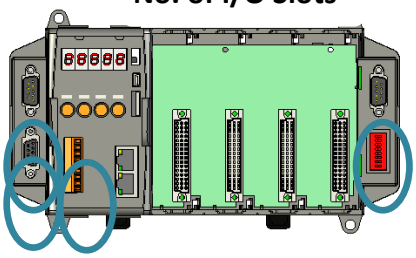


No. of I/O Slots

GetNumberOfSlot()

The GetNumberOfSlot() have to be called in order to get number of I/O slots.

API for getting number of COM ports



No. of COM ports

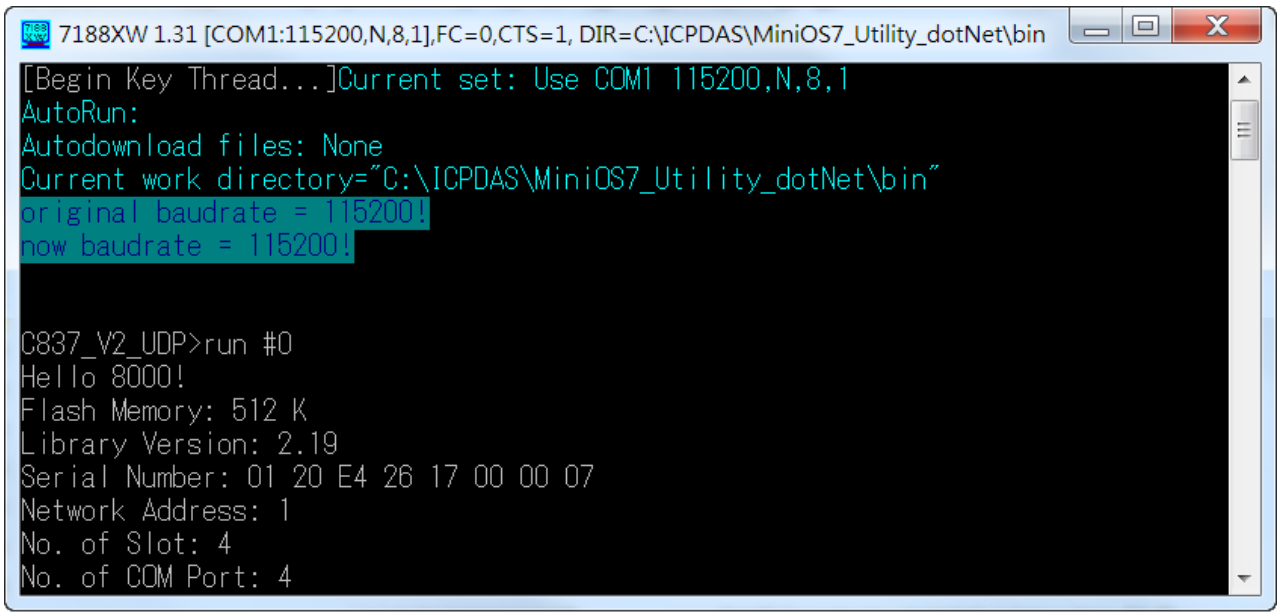
GetComportNumber()

The GetComportNumber() have to be called in order to get number of COM ports.

For example,read the system information.

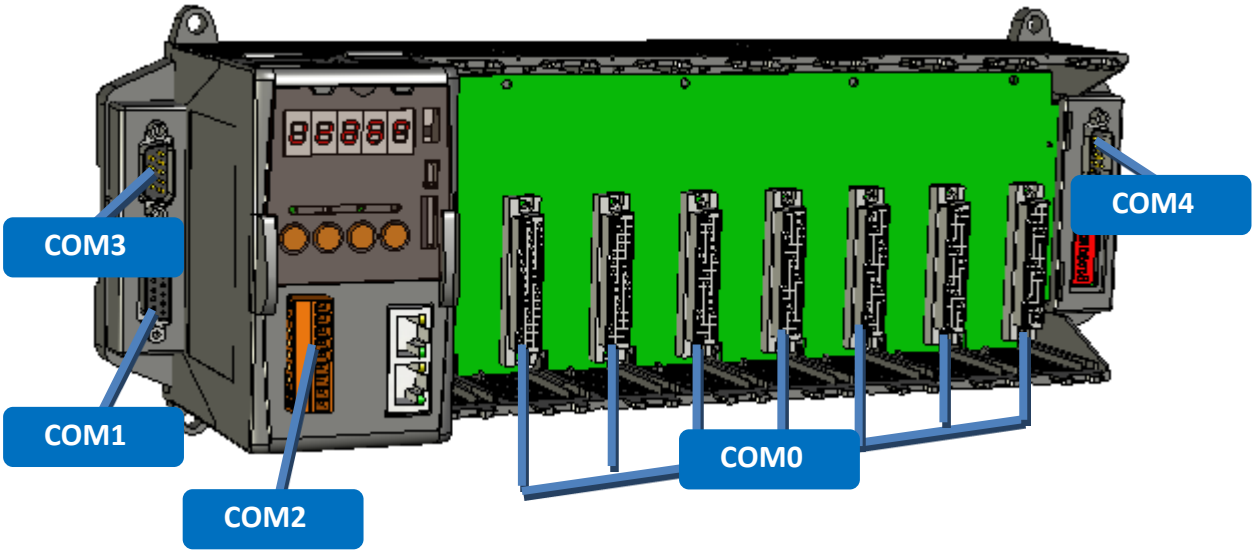
```
#include “8000a.h”
void main(void)
{
   inti,iType,iVer,iId,iSlot,iPort;
   char cSN[8];
   InitLib();
   iType=Is8000();
   if(iType){
      iVer=GetLibVersion();
      iId=GetNetId();
      iSlot=GetNumberOfSlot();
      iPort=GetComportNumber();
      Print(“Hello 8000!\n\r”);
      Print(“Flash Memory: %d K)\n\r”,iType);
      Print(“Library Version: %d.%02d\n\r”,iVer>>8,iVer&0xff);
      Print(“Serial Number: “);
      GetSerialNumber(cSN);
      for(i=0;i<8;i++){
         Print(“%02X “,cSN[i]&0xff);
      }
      Print(“\n\rNetwork Address: %d\n\r”,iId);
      Print(“No. of I/O slot: %d\n\r”,iSlot);
      Print(“No. of COM Port: %d\n\r”,iPort);
   }
   else{
      Print(“Hello PC!, this program is not run under 8000.”);
   }
}
```

The result of this example is shown in the figure below.



4.1.2. MiniOS7 API for COM Port

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



4.1.2.1. 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()

4.1.2.2. API for MiniOS7 COM port

API for using COM ports

InstallCom()

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

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

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

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

ToCom()

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

For example, read and receive 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 */
}
```

printCom()

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

For example, show 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 */
}
```

4.1.2.3. 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

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

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

Puts() – For sending a string

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

Putch( ) – For sending one character

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

Print()

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

For example, read and receive 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, show 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);
 }
}
```

4.1.2.4. Port functions Comparison

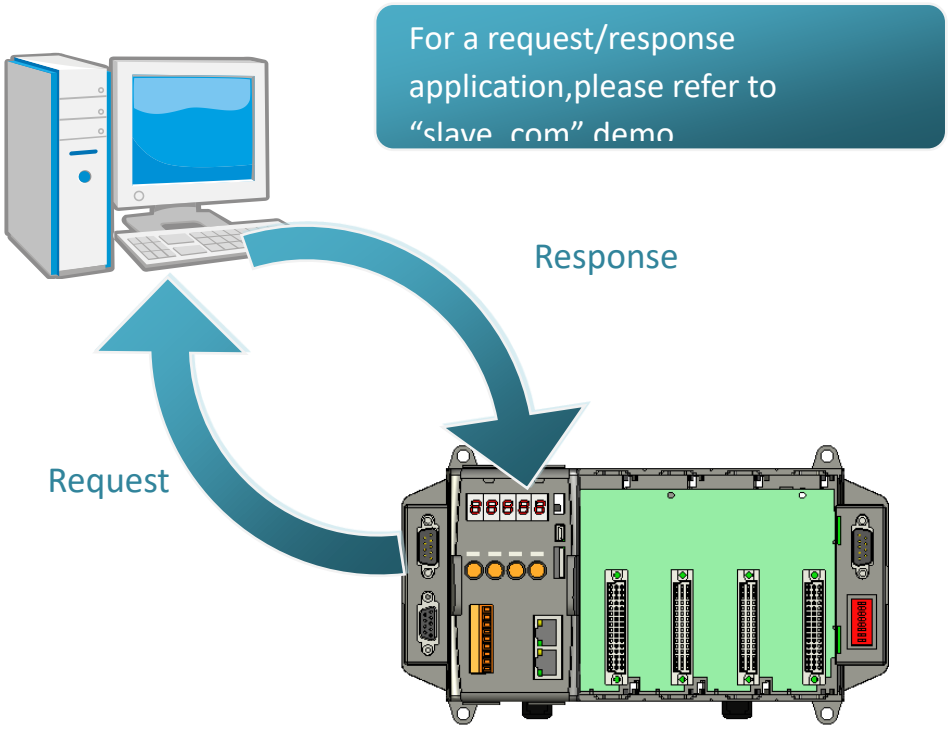
For example, show the ASCII code.

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:”);
ToCom(1,item);
printCom(1,”\n\rASCII(%c)\n\r”,item);
printCom(1,“Hex(%02X)\n\r”,item);
}
}
}
Delay(10);
RestoreCom(1);
}
	#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:“);
	Putch(item);
	Print(“\n\rASCII(%c)\n\r”,item);
	Print(“Hex(%02X)\n\r”,item);
	}
	}
	}
	}

4.1.2.5. Request/Response protocol define on COM port

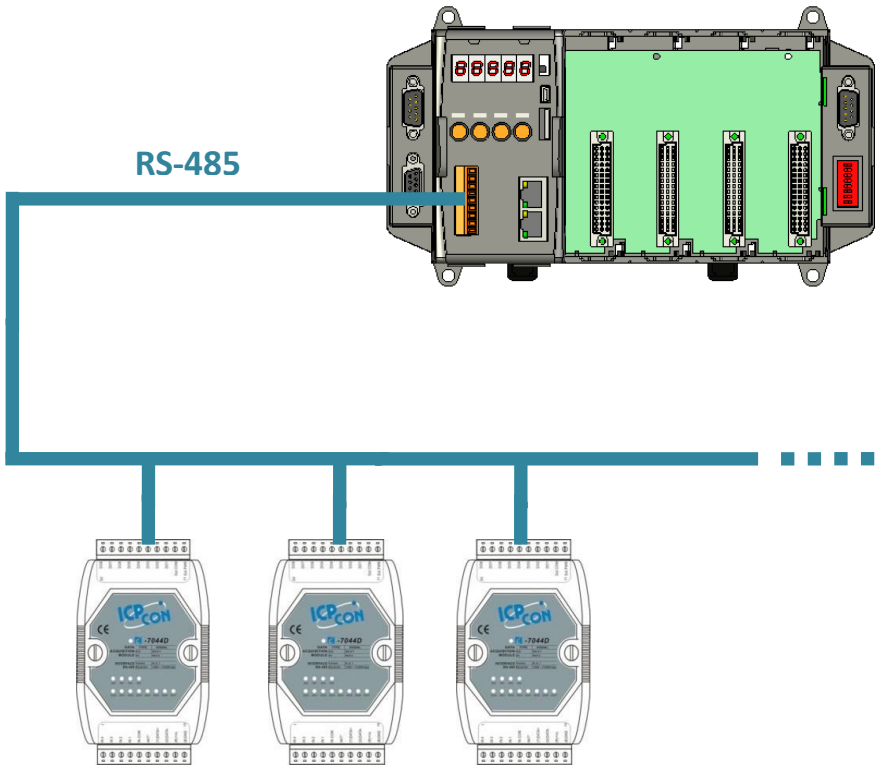
Request/Response communication is very typical protocol architecture. If you want to design a command set of communication protocol as table below, you can refer to “slave_com” demo.



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

4.1.3. 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.

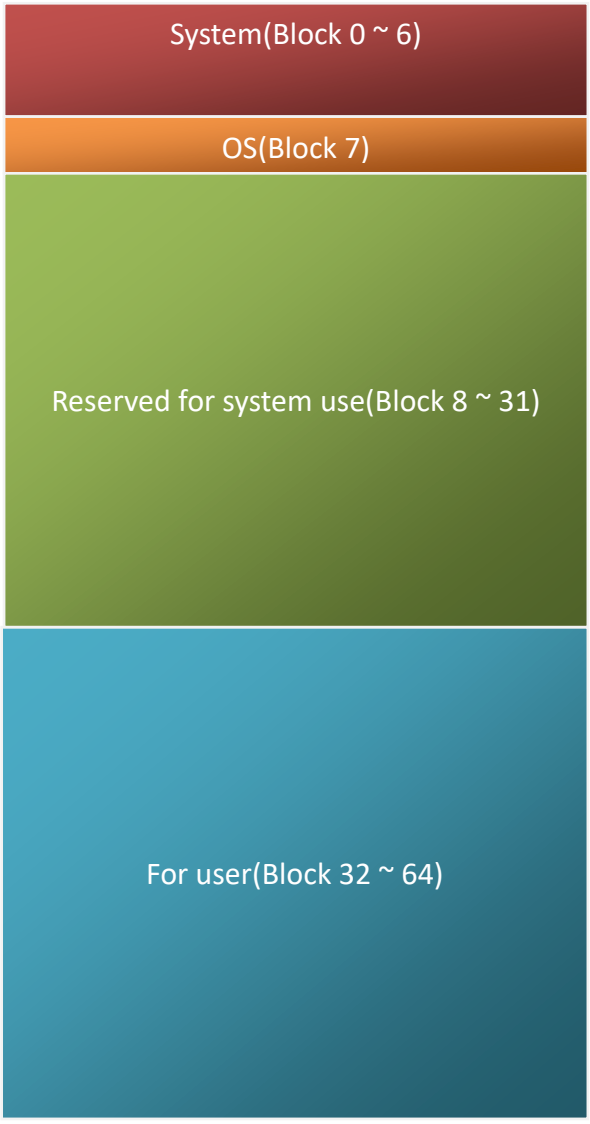
Step 4: Use RestoreCom() to restore the COM port driver

For example, senda 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 */
}
```

4.1.4. 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

EE_WriteEnable()

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

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.

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

EE_MultiRead()

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

For example, 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 */
}
```

4.1.5. 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

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

FlashWrite()

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

FlashRead()

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

For example, write an integer to segment 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;
}
```

4.1.6. 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 EEPROMboth 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

WriteNVRAM()

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

API for reading data from the NVRAM

ReadNVRAM()

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

For example, write data to the NVRAM address 0.

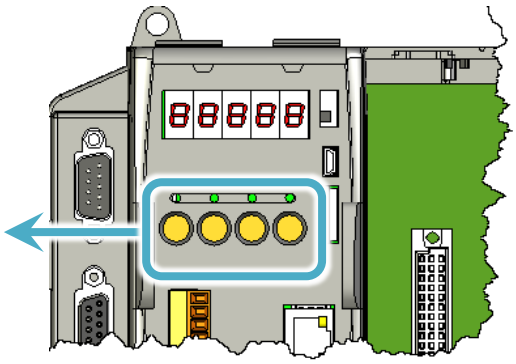
```
#include <stdio.h>
#include “8000a.h”

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

For example, 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 */
}
```

4.1.7. MiniOS7 API for SMMI

The iPAC-8000has4 programmable system keys(MODE, UP, DOWN and SET)and 3 programmable LED indicators(L1, L2 and L3), which can be set to show/play/pause the countdown timer, run the special program, and so on.



API for clearing all data in the system key input buffer

ClearSystemKey()

Before setting any system keys, the ClearSystemKey() must be called to clear all data in the system key input buffer.

API for checking there is any data in the system key input buffer.

IsSystemKey ()

Before reading data from system keys, the IsSystemKey() must be called to check whether there is any data currently in the system key input buffer.

API for reading data from system key

GetSystemKey ()

After IsSystemKey() confirms that the input buffer has data, the GetSystemKey() must be called to read the data from the system key input buffer.

API for setting the LED status

SetLedL1/2/3()

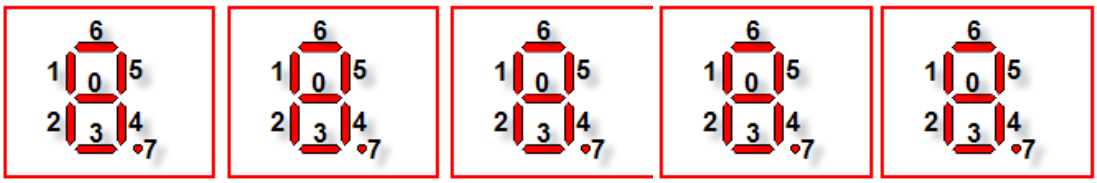
The SetLedL1/2/3()have to be called in order to set the LED status.

For example, set buttons to set LED indicators.

```
#include”8000a.h”
void main(void)
{
  int quit=0;
  InitLib();
  ClearSystemKey();
  while(!quit){
    if(IsSystemKey()){/* Check if there is any data in the system key input buffer */
      switch(GetSystemKey()){
        case SKEY_UP:
             SetLedL1(2);
             break;
        case SKEY_DOWN:
             SetLedL2(2);
             break;
        case SKEY_SET:
             SetLedL3(2);
             break;
      }
    }
    if(Kbhit()){
      if(Getch()==’q’) quit=1;
    }
  }
}
```

4.1.8. 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

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

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, 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 */

}
```

4.1.9. 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

TimerOpen()

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

API for reading the Timer

TimerResetValue()

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

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

TimerClose()

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

For example, 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 */
}
```

4.1.10. MiniOS7 API for WatchDog Timer (WDT)

- The iPAC-8000 equipsthe 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 (InterruptService Routine) refreshes it.
- The system is reset by WatchDog. The timeout period of WatchDog is 0.8 seconds.

API for refreshing WDT

EnableWDT()

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

RefreshWDT()

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

DisableWDT()

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

For example, 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_V105.LIB and microSD.h

The iPAC-8000 series can support one microSD card and supports up to 32 GB storage capacity.

- 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	Readsthe 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	Checkswhether 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
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

pc_Init()

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

API for enabling/disabling microSD

pc_open()

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

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

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”
{
intfd, iRet;
InitLib();
  If(pc_init()) {
    Print(“InitmicroSD ok\n\r”);
  }
  else {
    Print(“InitmicroSD 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_IWRITE|PS_IREAD)
  if(fd>=0) {
    pc_write(fd,”1234567890”,10);      //write 10 bytes
    pc_close(fd);
  }
}
```

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”
{
intfd, iRet;
  unsigned char Buffer[80];

InitLib();
  If(pc_init())
  {
    Print(“InitmicroSD ok\n\r”);
  }
else
  {
    Print(“InitmicroSD 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);
  }
}
```

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.

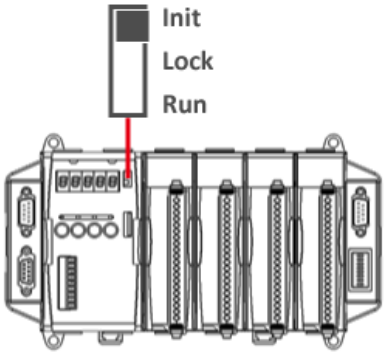
Both the OS updates services and SDK updates services can be found separately on the CD that was provided with the package or by downloading the latest version from ICP DAS web site.

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

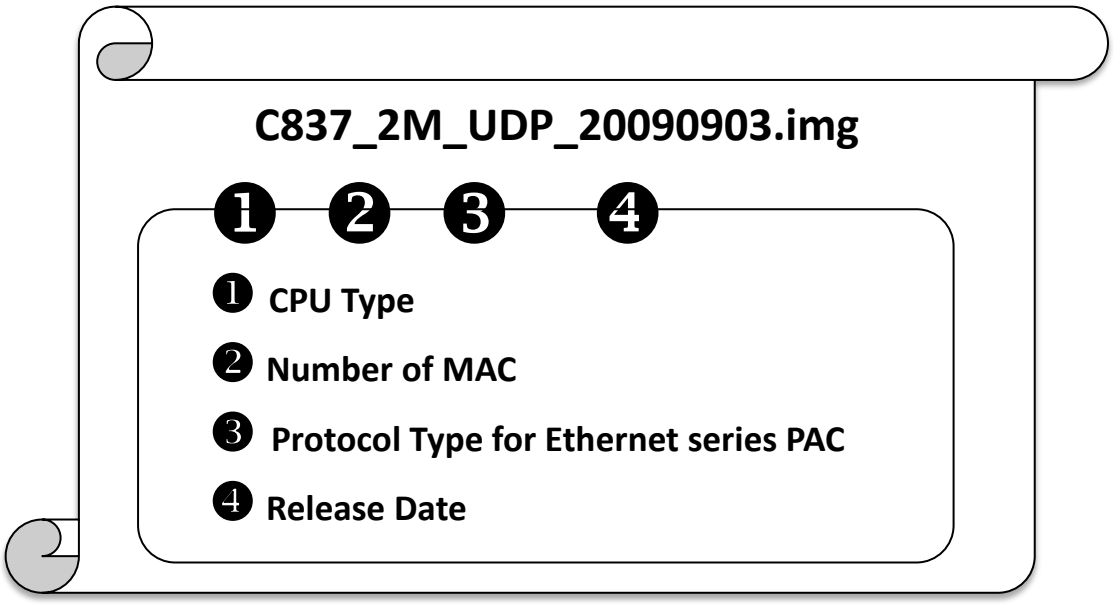
Make sure the switch placed in the "Init" position



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 from ICP DAS web site.

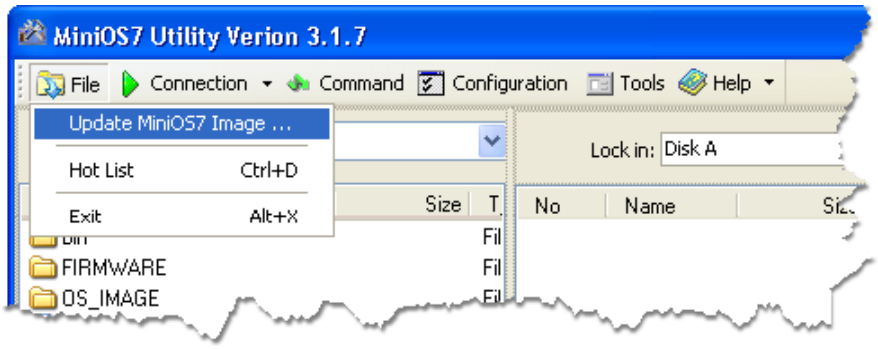
https://www.icpdas.com/en/download/show.php?num=2320&model=iP-8841



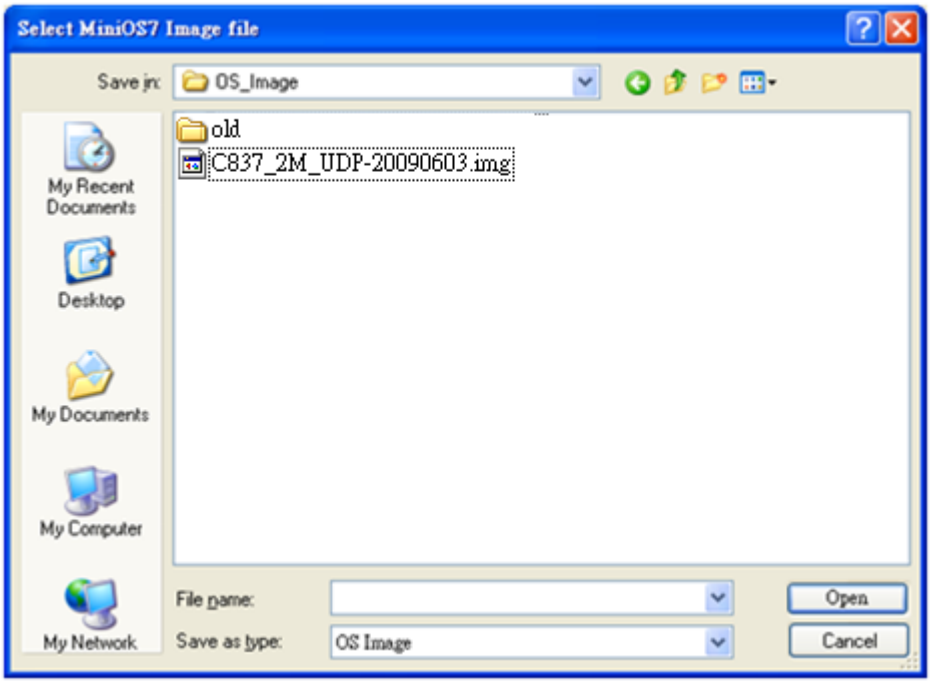
Step 3: Establish a connection

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

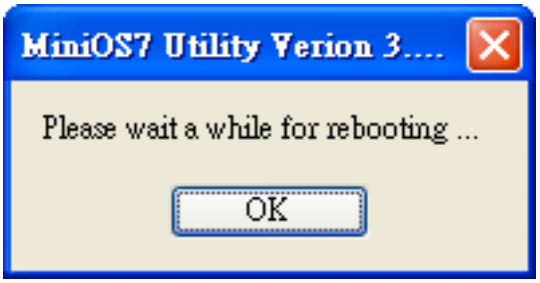
Step 4: Click the “Update MiniOS7 Image …” from the “File” menu



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



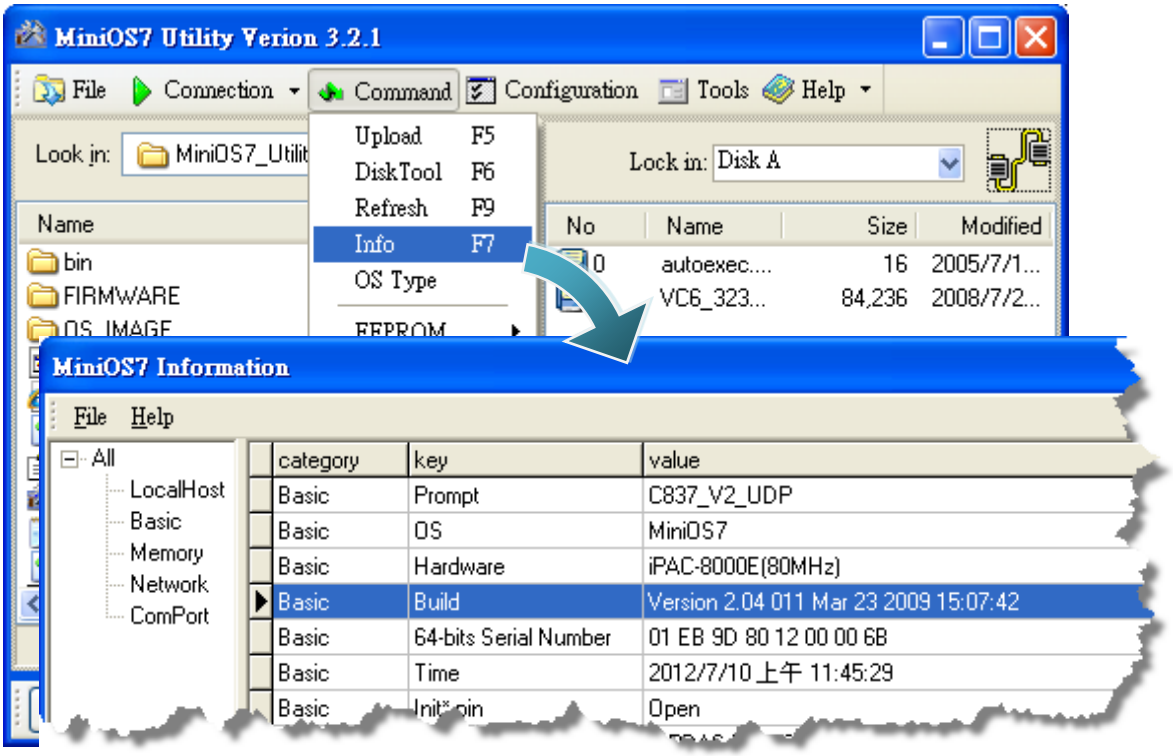
Step 6: Click “OK” button



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

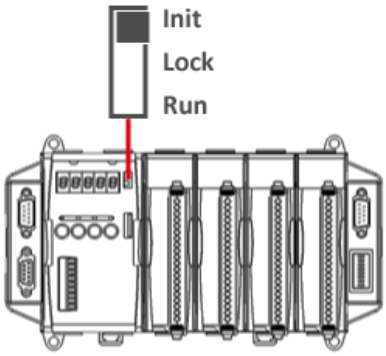


5.2. Updating the Firmware

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 from ICP DAS web site.

https://www.icpdas.com/en/download/show.php?num=2792&model=iP-8841

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?

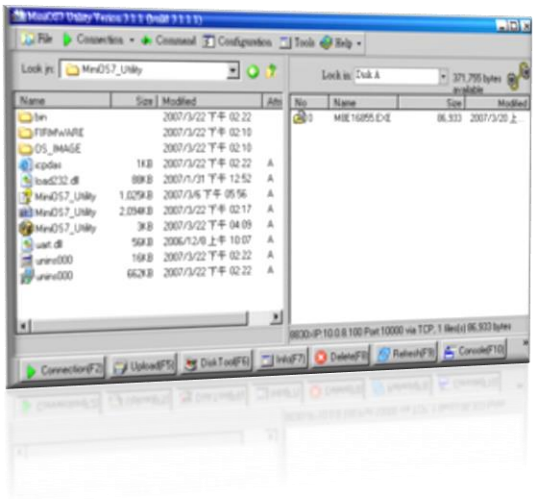
MiniOS7 is an embedded operating system design by ICP DAS. It is functionally equivalent to other brands of DOS, and can run programs that are executable under a standard DOS.

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

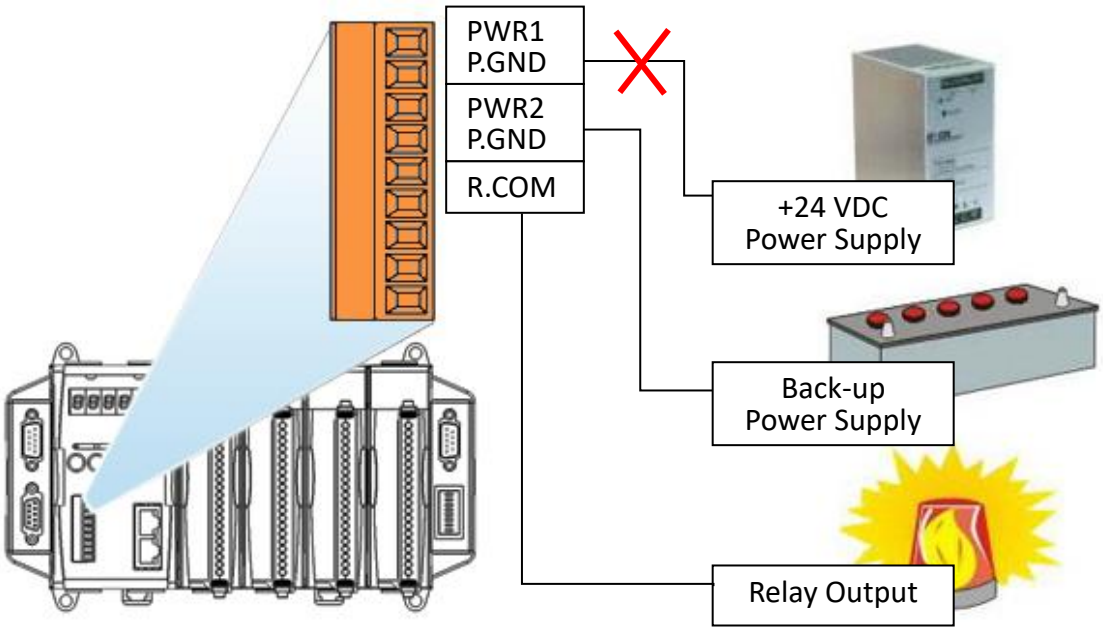
Upload location:

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

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 toautomatically support the power of iPAC-8000 needs.

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

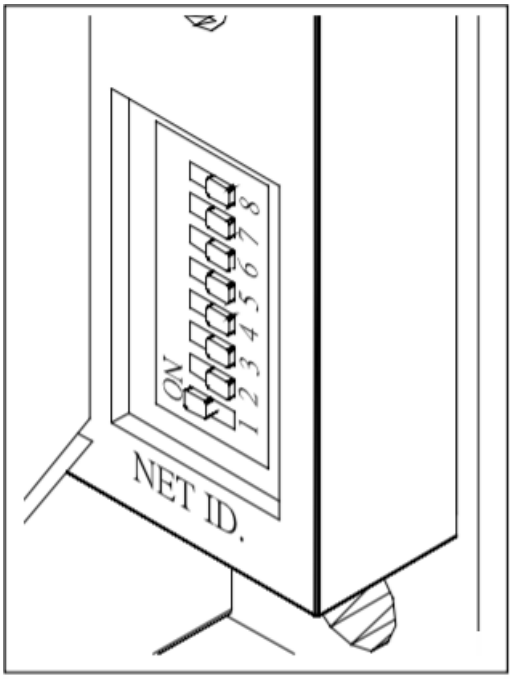


Appendix E.How to set theNet ID

All modules on the network musthave a unique address. The DIP switch Net Idis used to set the address which has the range of 1 to 255. The default address of the iPAC-8000 is 1.

If aniPAC-8000 address is set to 0, it will not participate on the network.

From DIP-8 to DIP-1 it corresponds to a binary number with 8 bits in which the highest bit is DIP-8 and the lowest bit is DIP-1. The state ON foreach bit means 1 while the state OFF means 0.



The encodes of the addresses are as follows:

Address	DIP-1	DIP-2	DIP-3	DIP-4	DIP-5	DIP-6	DIP-7	DIP-8
0	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
1	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF
2	OFF	ON	OFF	OFF	OFF	OFF	OFF	OFF
3	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF
4	OFF	OFF	ON	OFF	OFF	OFF	OFF	OFF
5	ON	OFF	ON	OFF	OFF	OFF	OFF	OFF
6	OFF	ON	ON	OFF	OFF	OFF	OFF	OFF
…..	…..							
254	OFF	ON	ON	ON	ON	ON	ON	ON
255	ON	ON	ON	ON	ON	ON	ON	ON

Appendix F.RS-485 Network Application

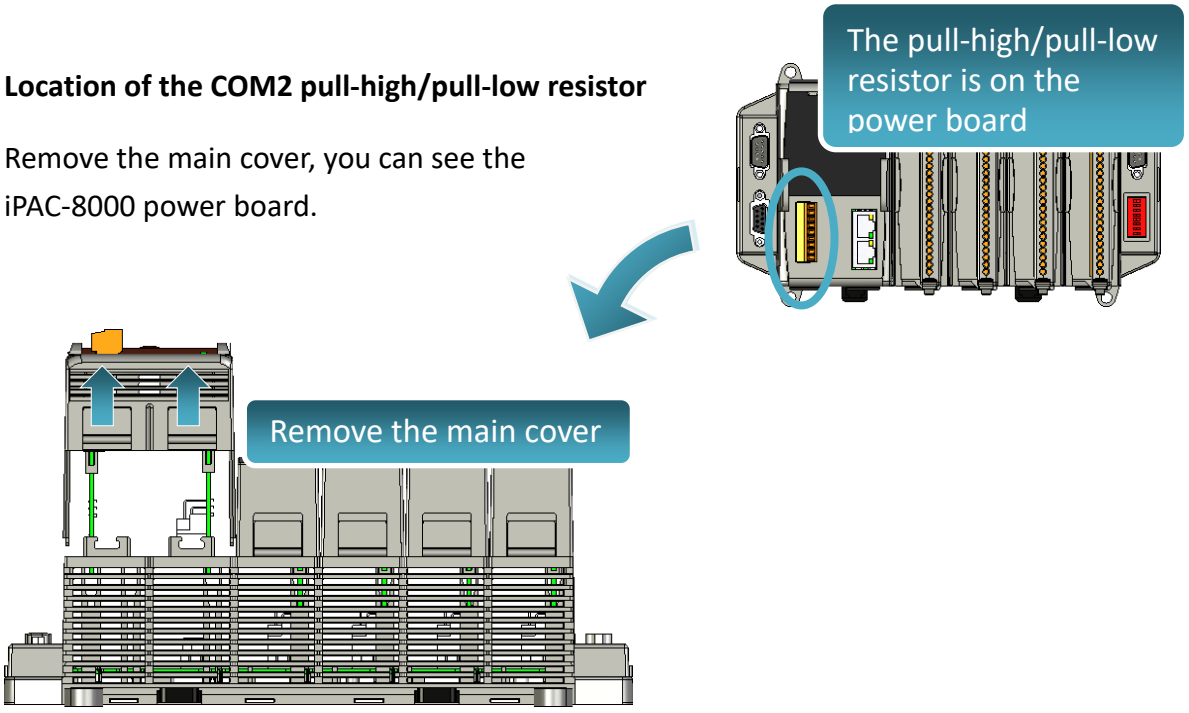
The RS-485 network is based on a Master-Slave architecture that consists of a single Master device and one or more Slave devices.

The iPAC-8000 provides two RS-485 serial ports, COM2 and COM3. Each RS-485 port includes a pull-high/pull-low resistor that can be used to set the iPAC-8000 as a Master or Slave.

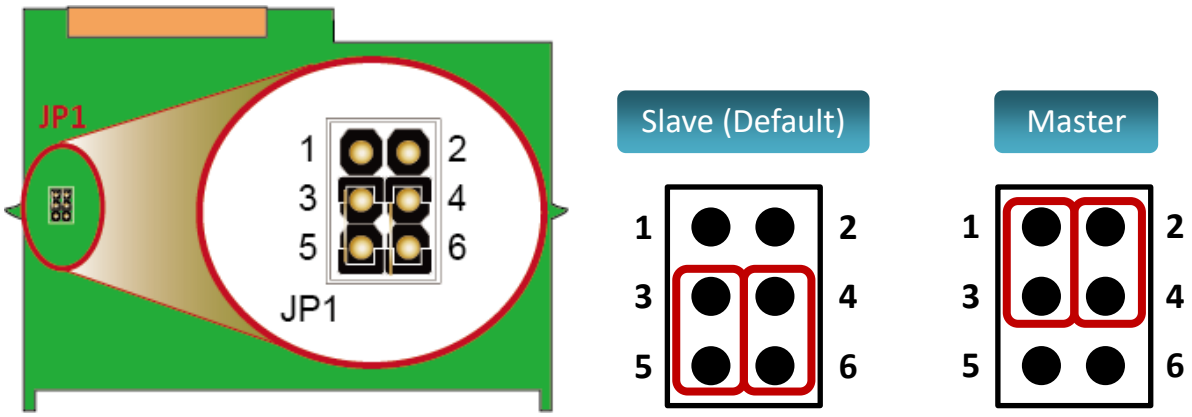
Only one of these two ports may be selected for Master operation. Here we recommend that you choose the COM2 to have protection from external signals.

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

Remove the main cover, you can see the iPAC-8000 power board.

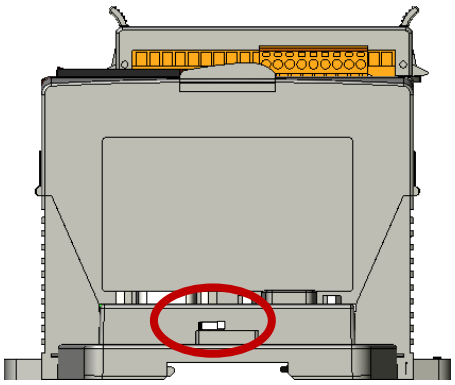


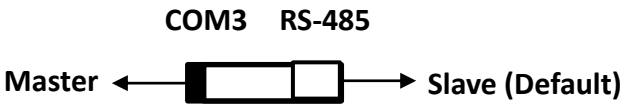
On the iPAC-8000 power board, you can see the COM2 pull-high/pull-low resistor.



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

There is a DIP switch for setting the COM3 pull-high/pull-low resistor that is as below shown.

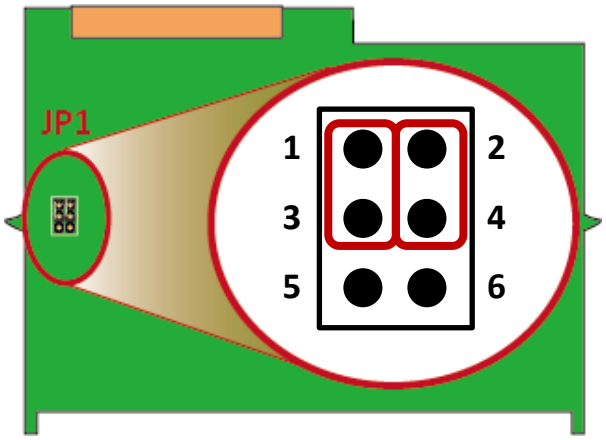




F.1. iPAC-8000 as a Master

iPAC-8000 as a master by using COM2

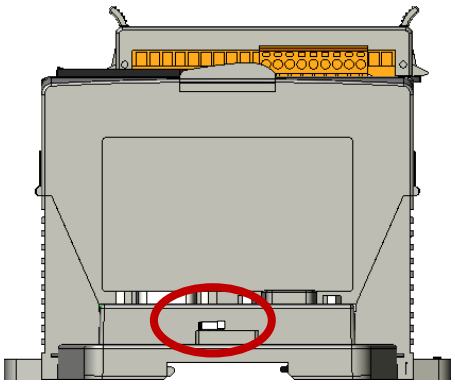
When iPAC-8000 as a master by using COM2, the status of the COM2 pull-high/pull-low resistor are as below shown.

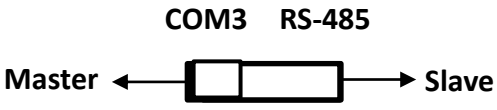


iPAC-8000 as a master by using COM2

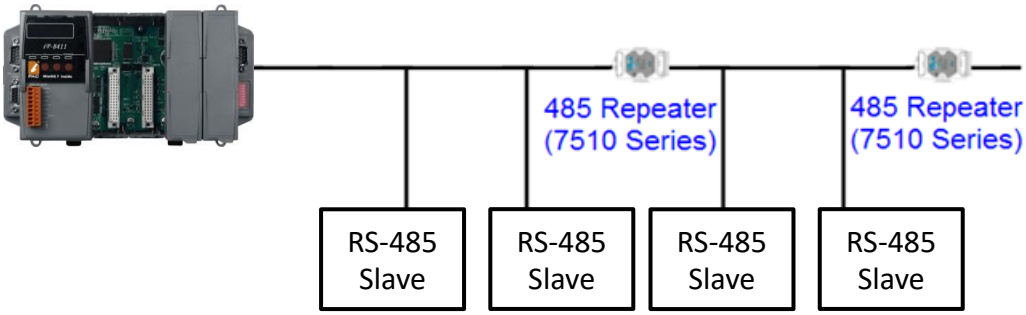
When iPAC-8000 as a master by using COM3, the status of the COM3 pull-high/pull-low are as below shown.

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





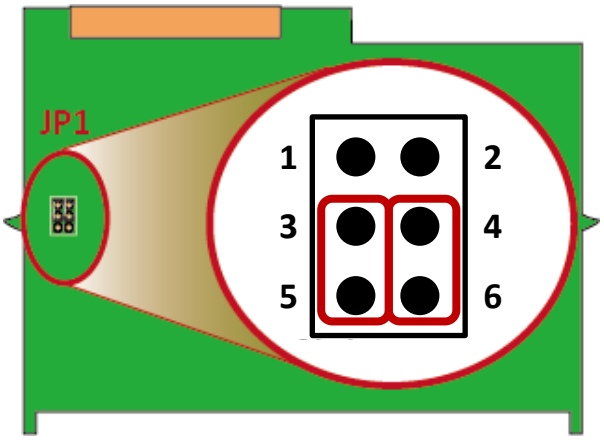
When one of iPAC-8000 is set to Master, then all of the other devices must be set to Slave. If the network is up to 1.2 KM, it will need a repeater (7510 series) to extend the network length.



F.2. iPAC-8000 as a Slave

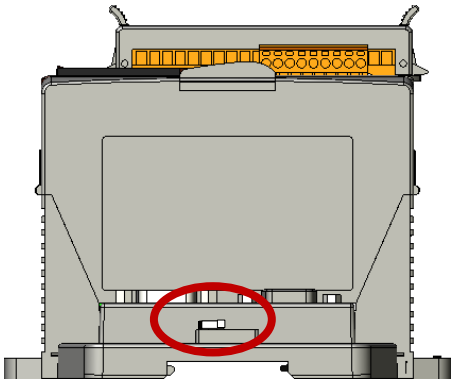
iPAC-8000 as a Slave by using COM3

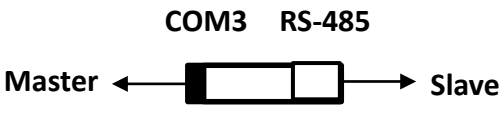
When iPAC-8000 as a slave by using COM3, the status of the COM2 pull-high/pull-low are as below shown.



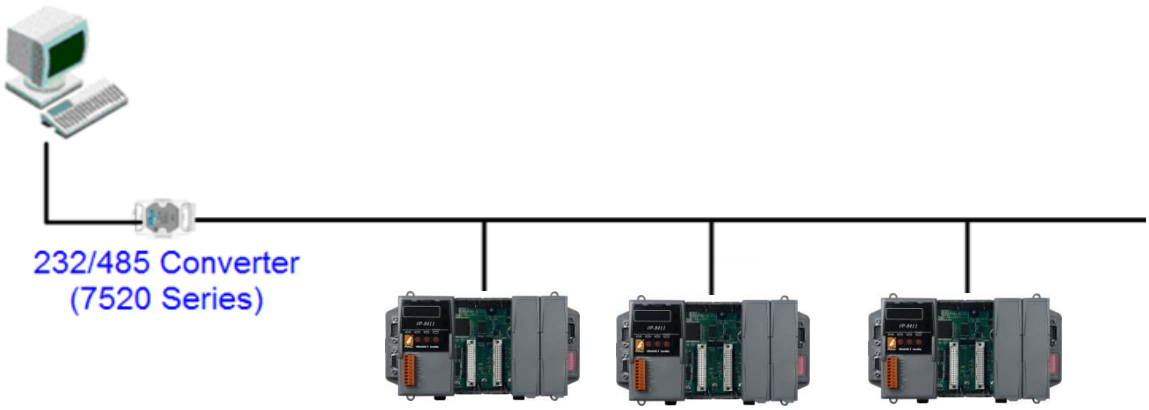
iPAC-8000 as a master by using COM3

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

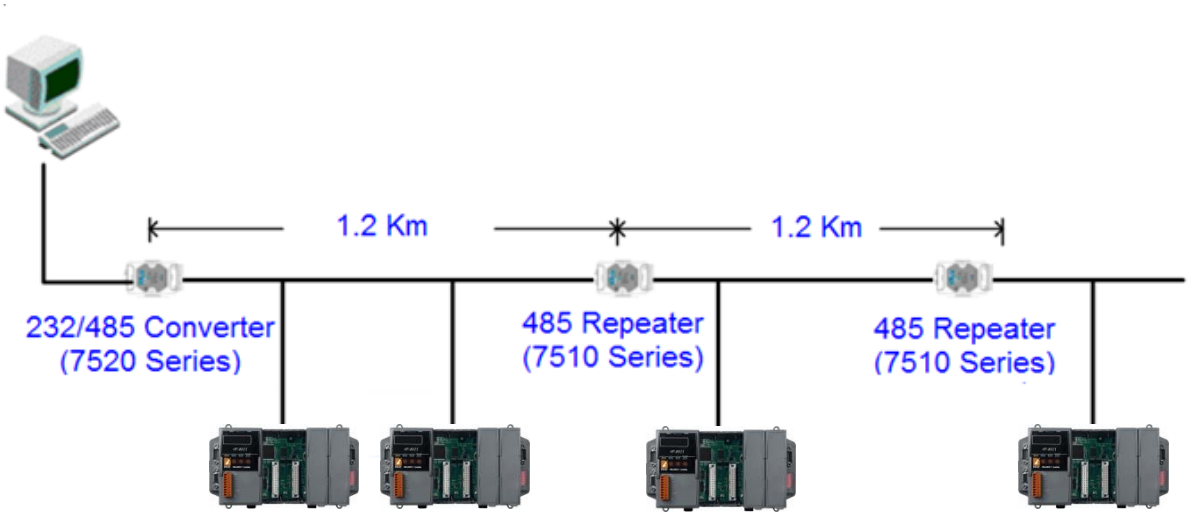




For most of application, when using one 7520 series as RS-232/485 converter, its pull-high/pull-low resistors are set to enabled. Then the iPAC-8000and all the other devices on this network must be set to slave mode.



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



Appendix G. 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

G.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

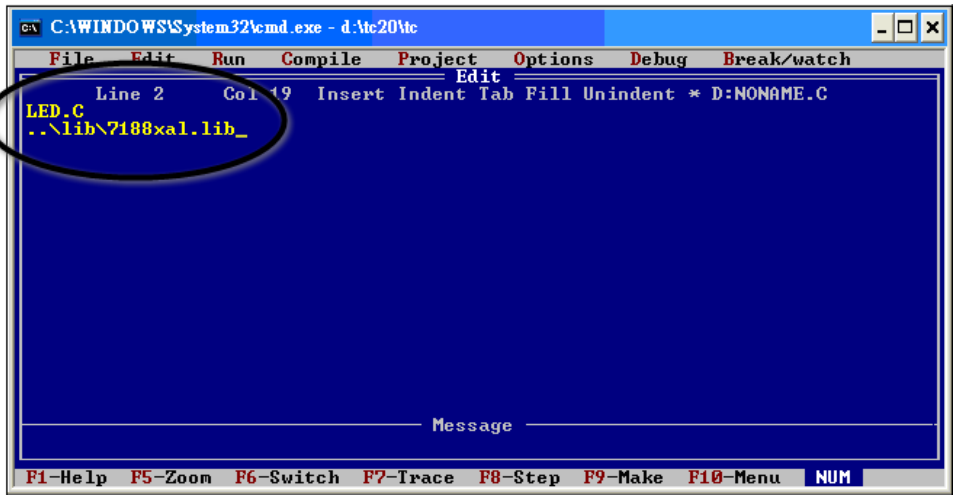
CD:\APDOS\iPAC8000\Demo\Basic\iP-84x1_iP-88x1\Hello\Hello_C\gotc.bat
tcc –Ic:\tc\include –Lc:\tc\lib hello.c ..\..\Demo\basic\ iP-84x1_iP-88x1\lib\8000a.lib

2: Using the TC Integrated Environment

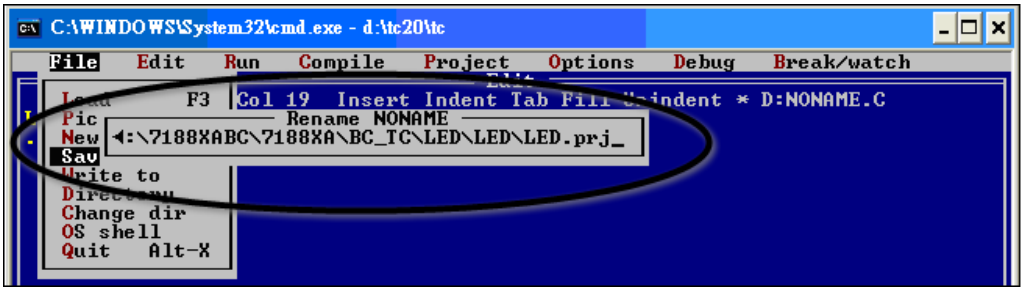
Step 1: Executing the TC 2.01

Step 2: Editing the Project file

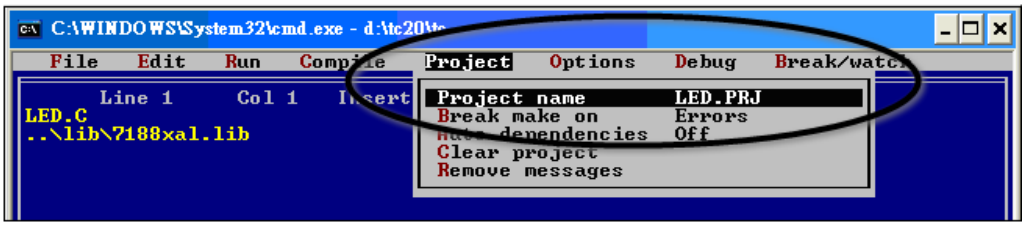
Adding the necessary library and file to the project



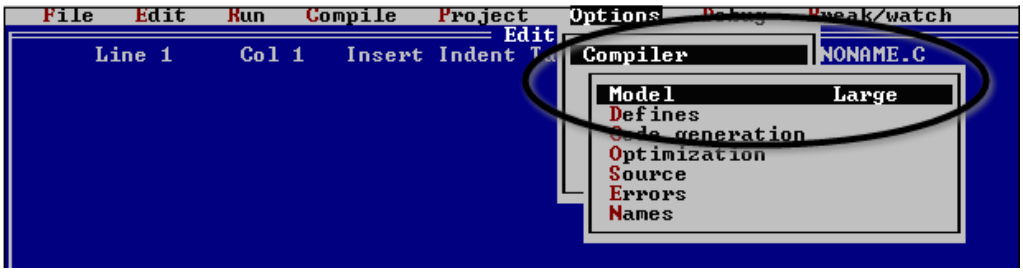
Step 3: Save the project and entering a name, such as LED.prj

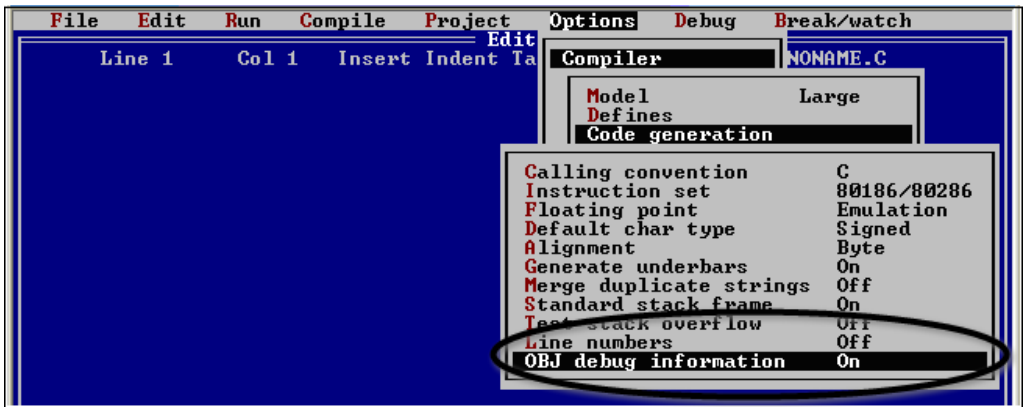


Step 4: Load the Project

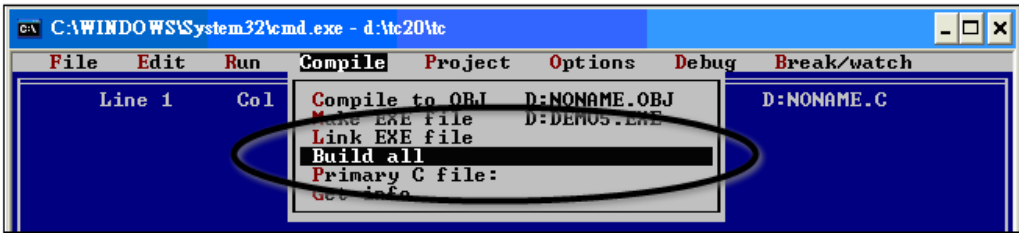


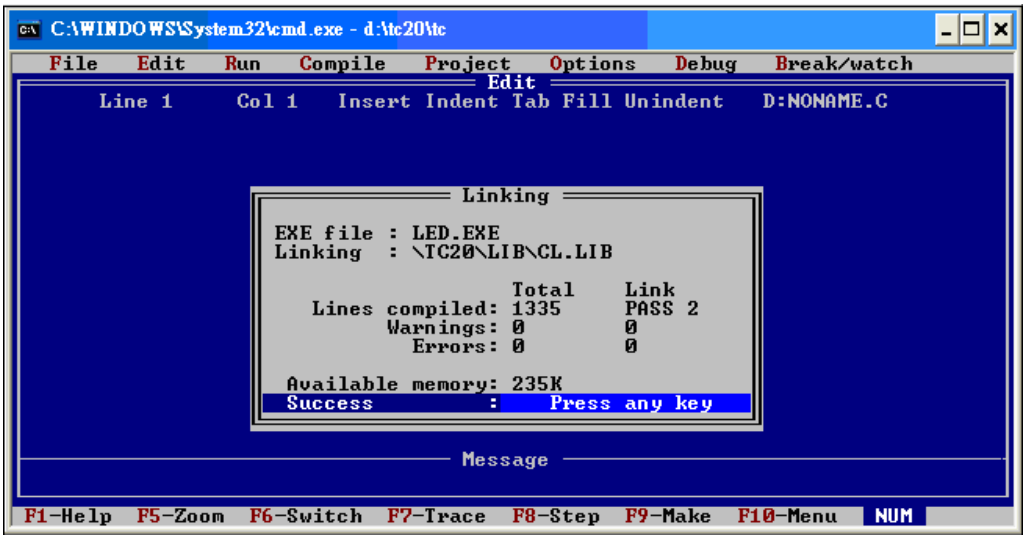
Step 5: Change the Memory model (Large for iPAC-8000.lib) and set the Code Generation to 80186/80286





Step 6: Building the project

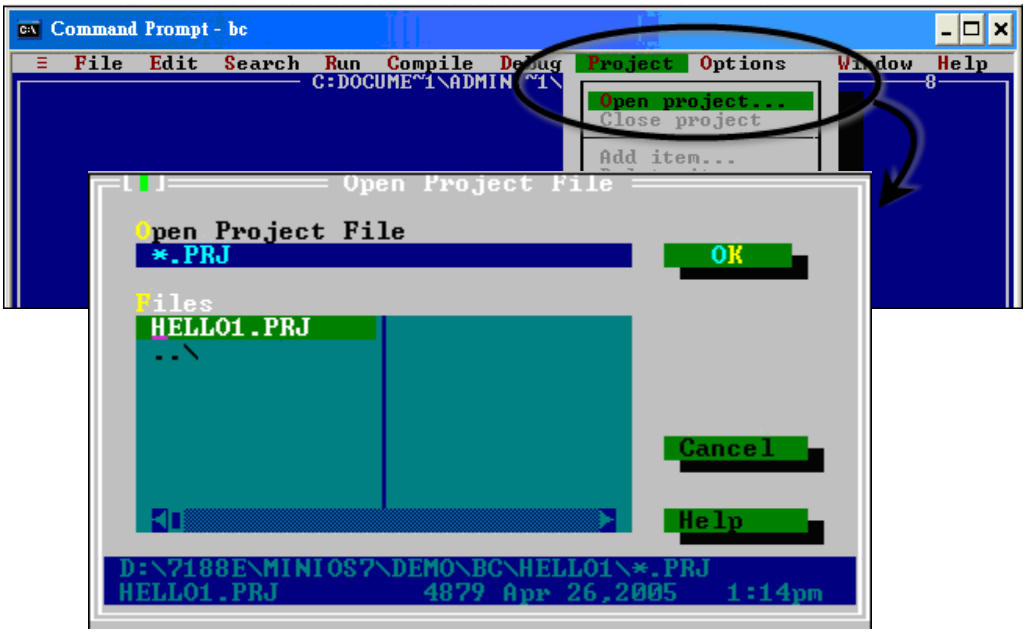




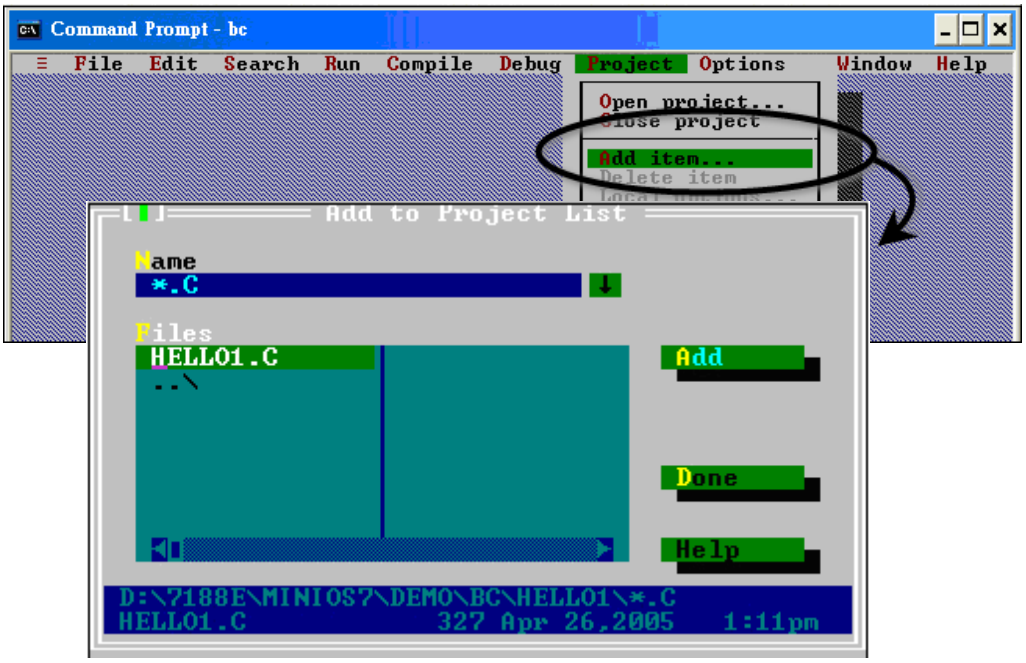
G.2. BC++ 3.1. IDE

Step 1: Executing the Borland C++ 3.1

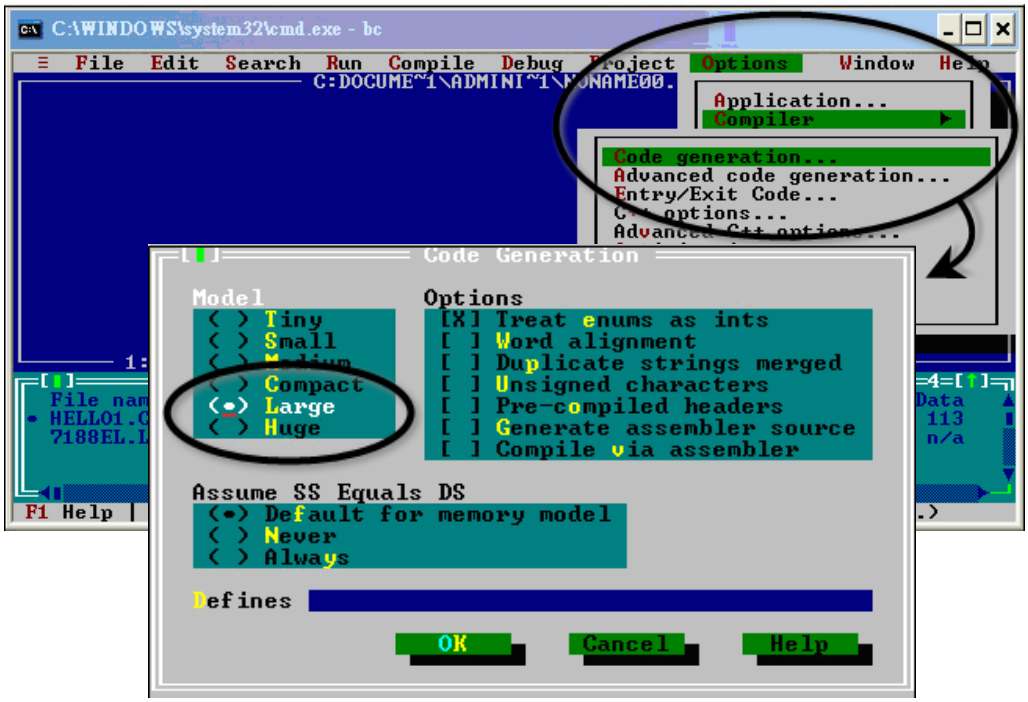
Step 2: Creating a new project file (*.prj)



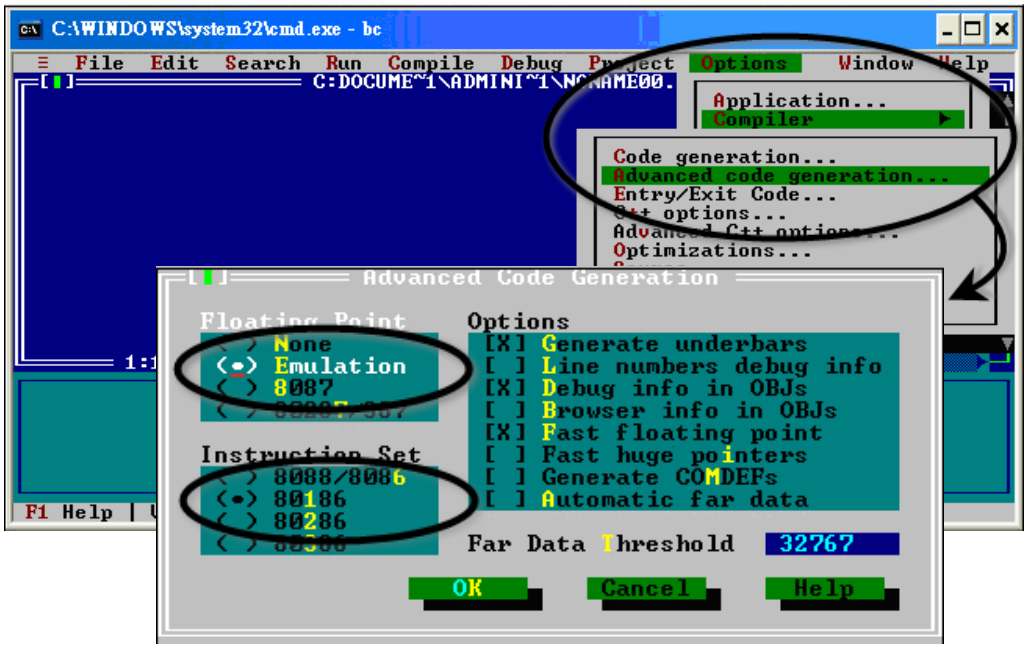
Step 3: Add all the necessary files to the project



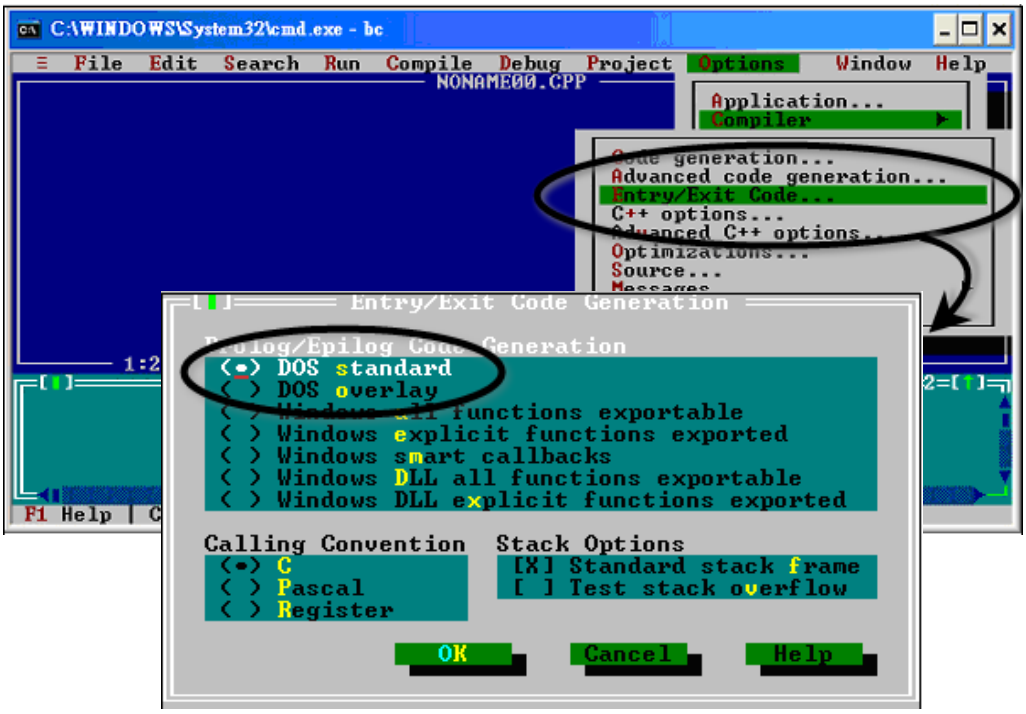
Step 4: Change the Memory model (Large for iPAC-8000.lib)



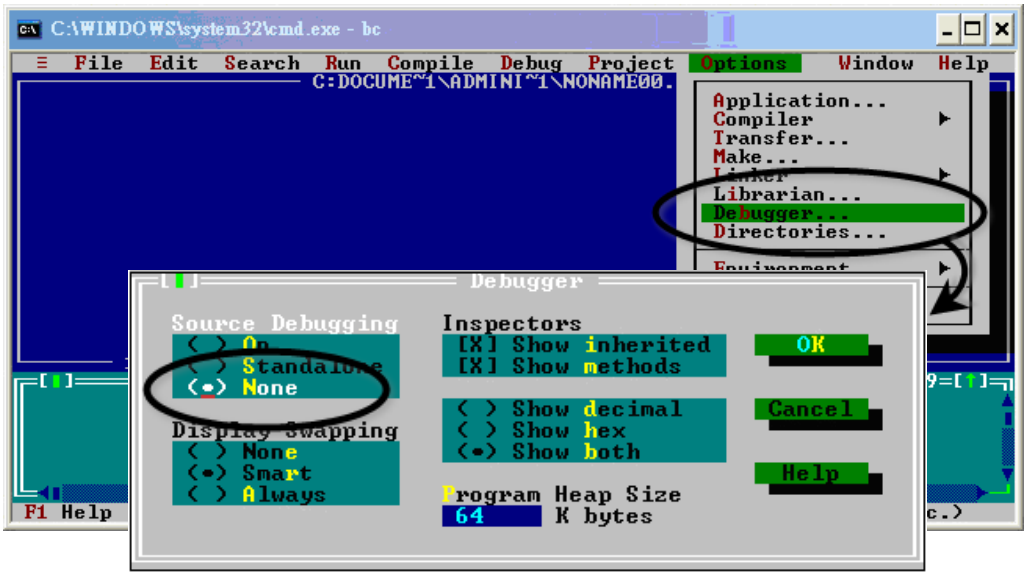
Step 5: Set the Advanced code generation options and Set the Floating Point to Emulation and the Instruction Set to 80186



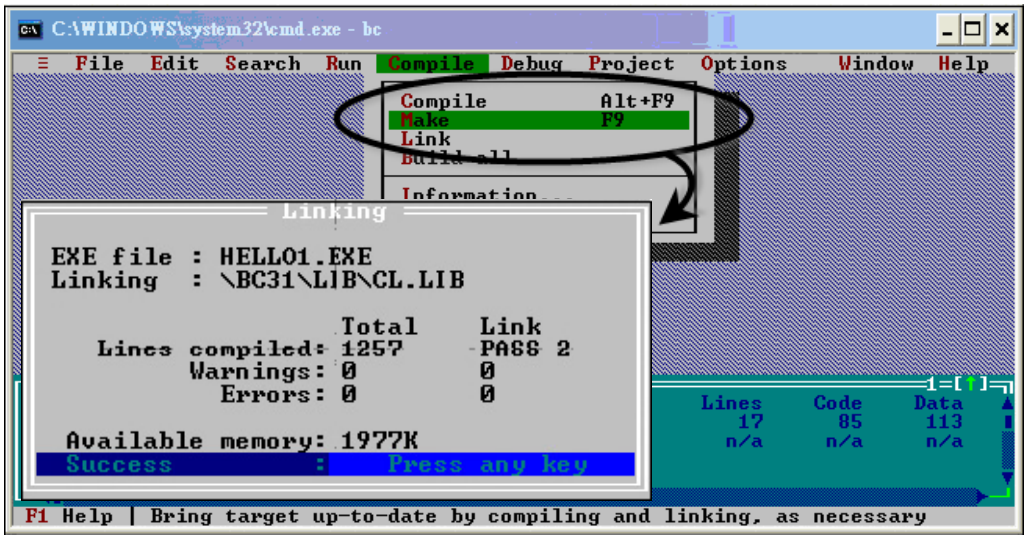
Step 6: Set the Entry/Exit Code Generation option and setting the DOS standard



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

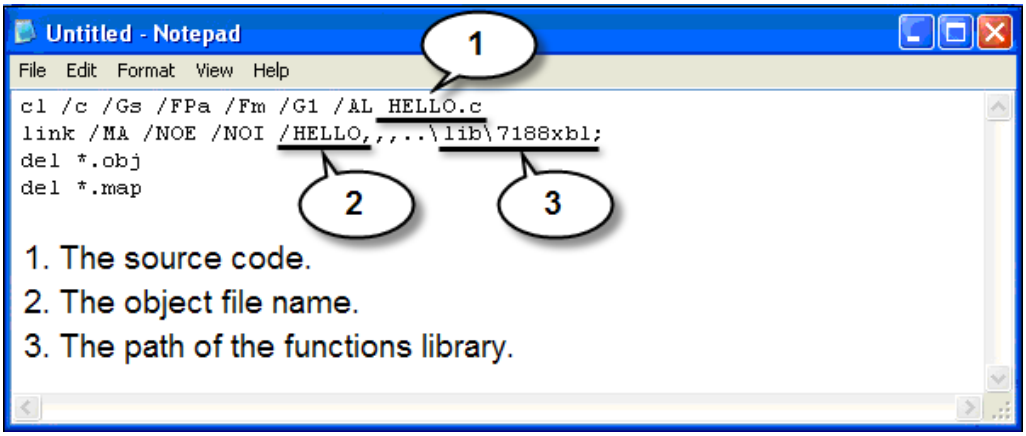


Step 8: Make the project



G.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

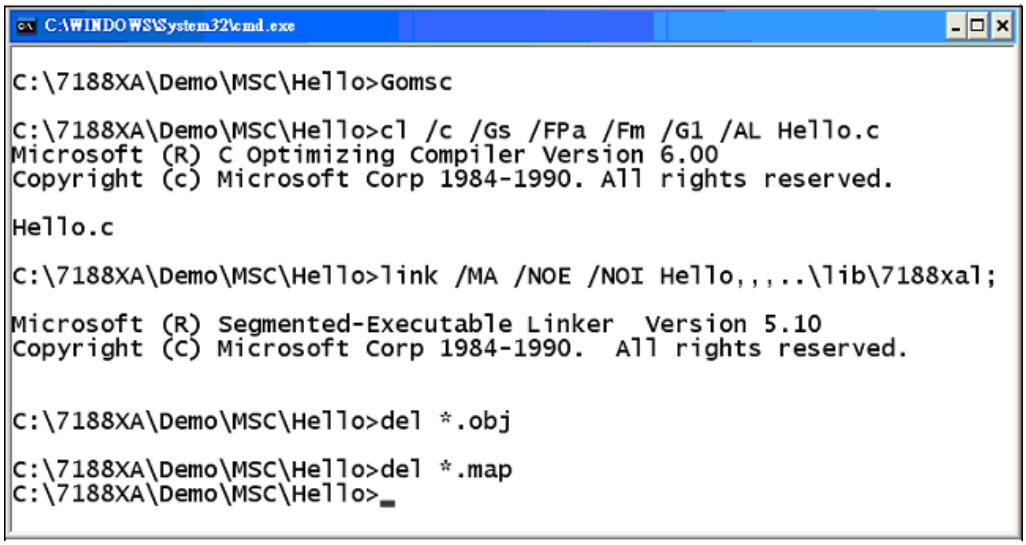
/Fpa: Calls with altmath

/Fm: [map file]

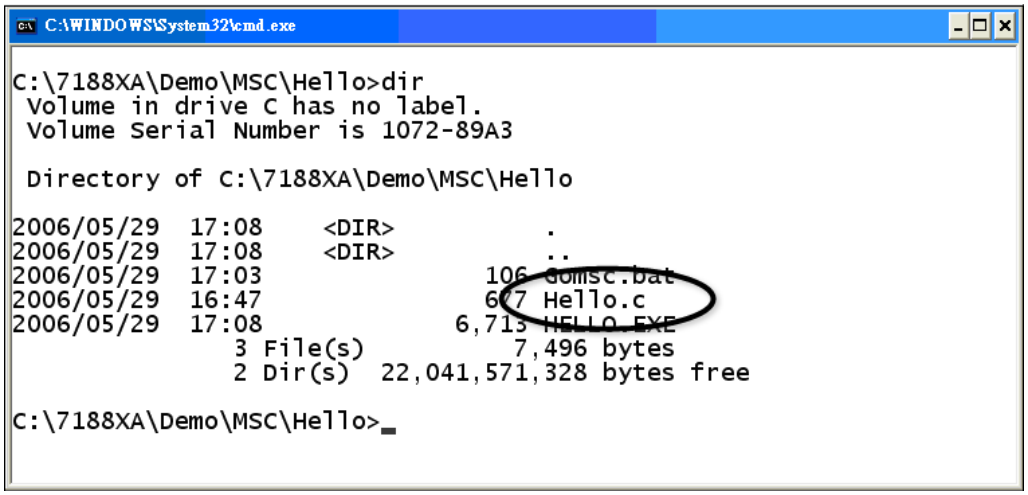
/G1: 186 instructions

/AL: Large model

Step 2: Run the Gomsc.bat file

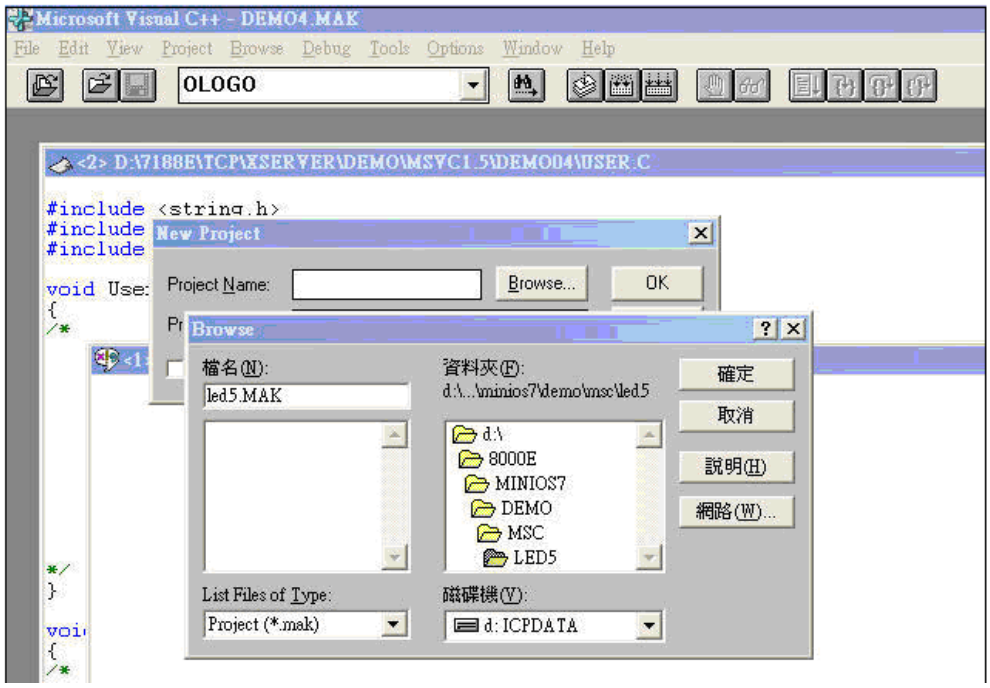


Step 3: A new executable file will be created if it is successfully compiled

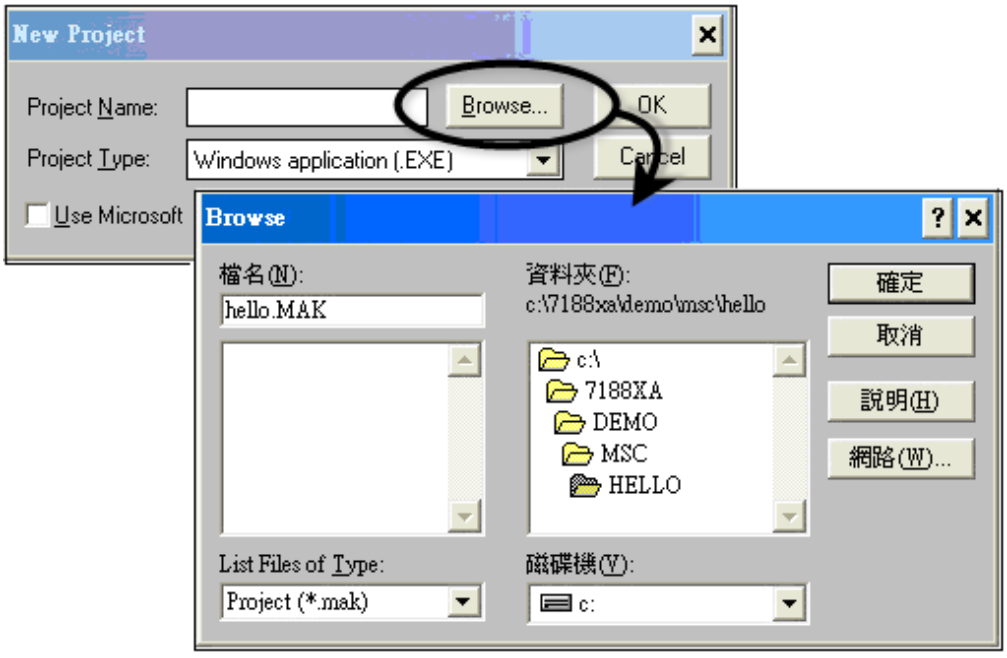


G.4. MSVC 1.50

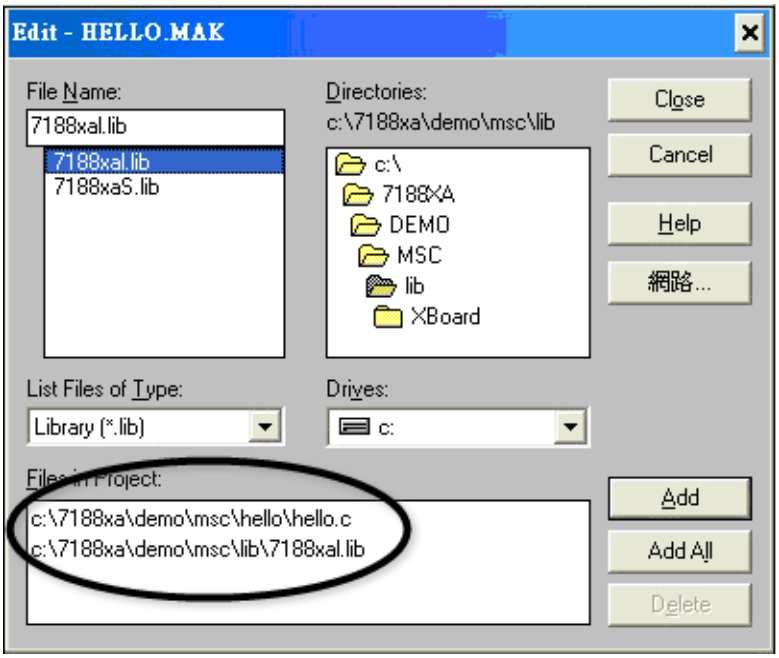
Step 1: Run MSVC.exe



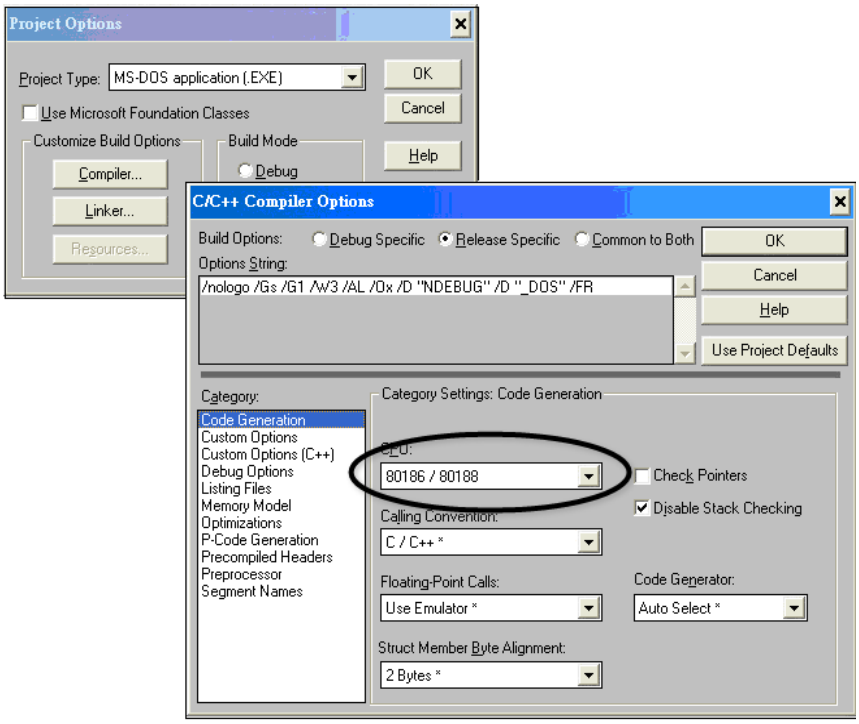
Step 2: Create a new project (*.mak) by entering the name of the project in the Project Name field and then select MS-DOS application (EXE) as the Project type



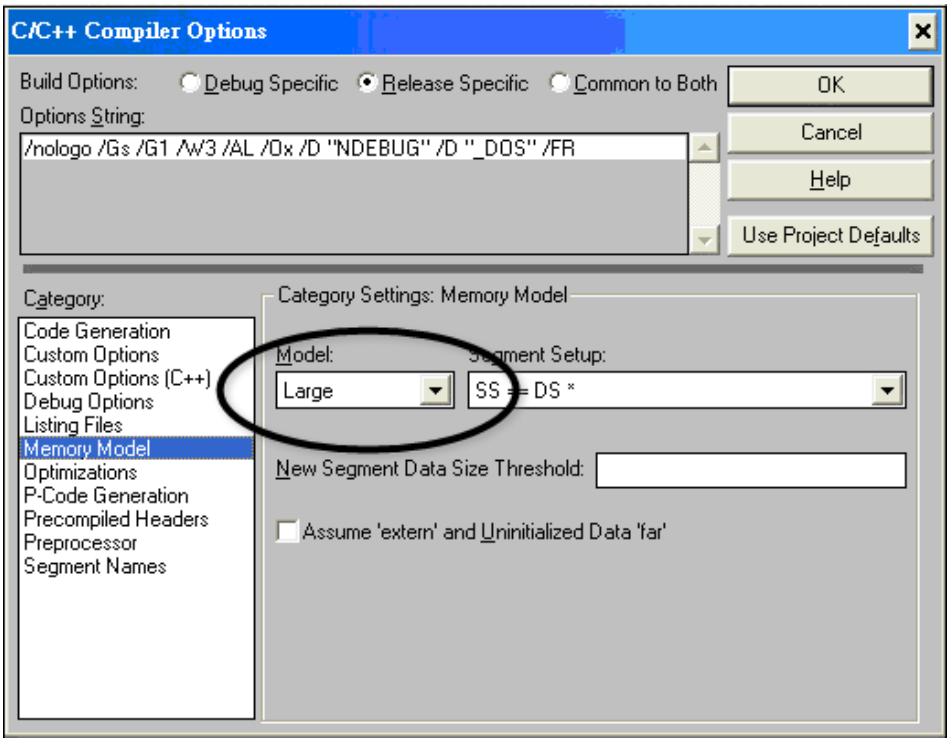
Step 3: Add the user's program and the necessary library files to the project



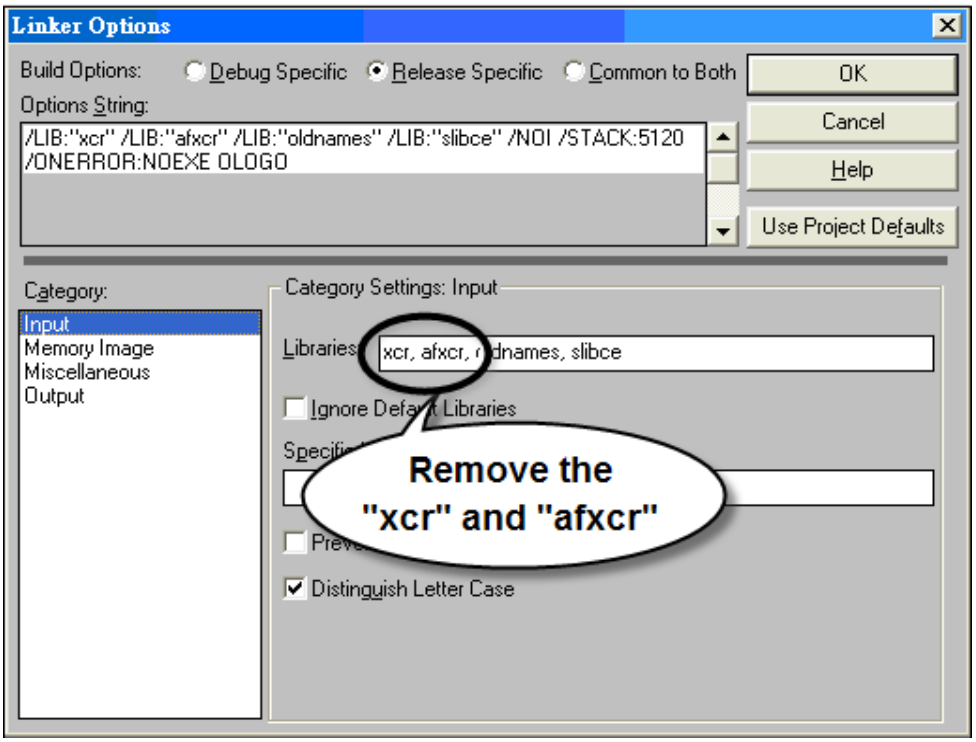
Step 4: Set the Code Generation on the Compiler.



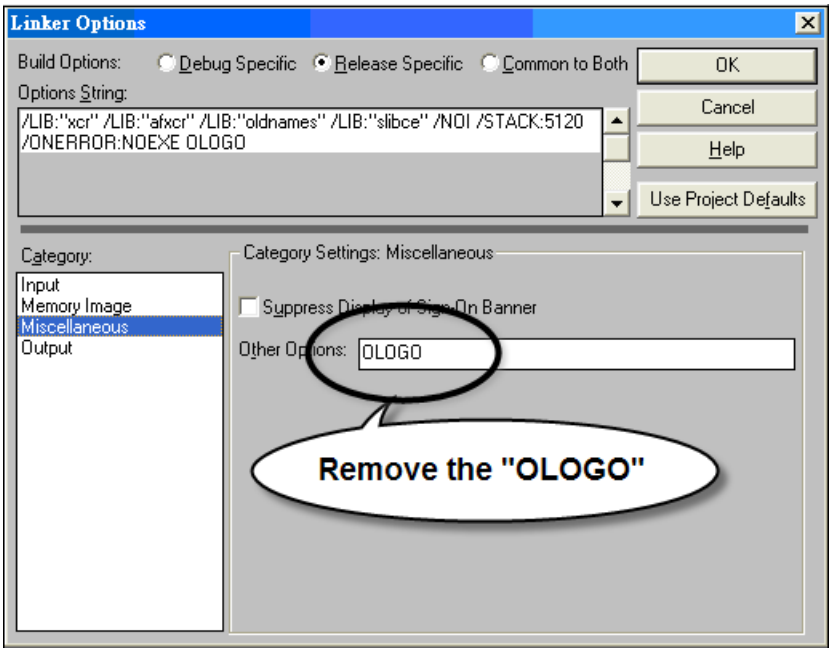
Step 5: Change the Memory model (large for iPAC-8000.lib)



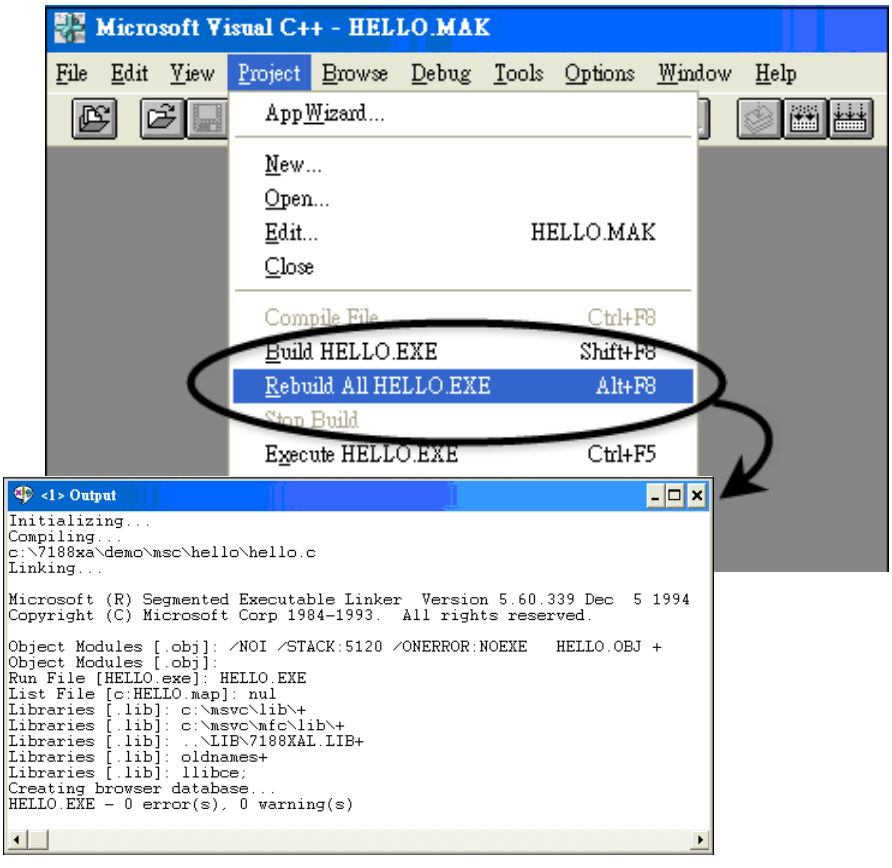
Step 6: Remove the xcr, afxcr libraries from the Input Category



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



Step 8: Rebuild the project



Appendix I. 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
1.0.5	November 2018	1. Added the information about system information functions in 4.1.1. MiniOS7 API for Library initialization and General Functions 2. Added the information about SMMI in 4.1.7. MiniOS7 API for SMMI 3. Added the information about how to set the network address in Appendix E. How to set the Network Address 4. Added the information about RS-485 Network Application in Appendix F. RS-485 Network application 5. Deleted the information about iP-8441-FD and iP-8841-FD.
1.0.6	Jan 2021	1. Combine Appendix H(how to run 16-bit programs on 64-bit Windows in Appendix) and Chapter 3 together. 2. Added more detail about how to build a 16-bit program in chapter 3. 3. Modified download link of Utilities, OS image, Firmware...etc.