iPAC-8000 User Manual

Version 1.0 beta3, February 2009

Service and usage information for

iP-8441

iP-8841

Written by Hans Chen

Edited by Anna Huang

Important Notices

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.

Table of Contents

1. Introduction

iPAC 8000 is compact size PAC (Programmable Automation Controller). It equips a 80186 CPU (16bits and 80MHz) running a MiniOS7 operating system, several communication interface (Ethernet, RS-232/485) and 4/8 slots to expand I/O modules.

The operating system, MiniOS7, can boot up in a very short time (0.4~0.8 seconds). It has a built-in hardware diagnostic function, and supports the full range of functions required to access all high profile I-8k and I-87k series I/O modules, such as DI, DO, DIO, AI, AO, Counter/Frequency, motion control modules, etc. And to simplify the TCP/IP software developing process, a software development template, X-Server, is provided. It implements 90% functionalities of Ethernet communication. Software engineer can easily finish the 10% remaining functionalities and greatly shorten the developing time.

The iPAC-8000 is designed for applications to industrial monitoring, measurement and controlling; therefore, we made it with redundant power inputs with 1KV isolation from noise and surges, and a wild range of operating temperature (-25°C~+75°C). It is toughly enough to survive the harsh and rough environment.

1.1. Features

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.

- **Redundant Ethernet Communication**

  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.

- **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 funtionalities and greatly shorten the developing time.

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

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

  A. DCON firmware

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

  B. Modbus firmware

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

Hardware Features

- **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.

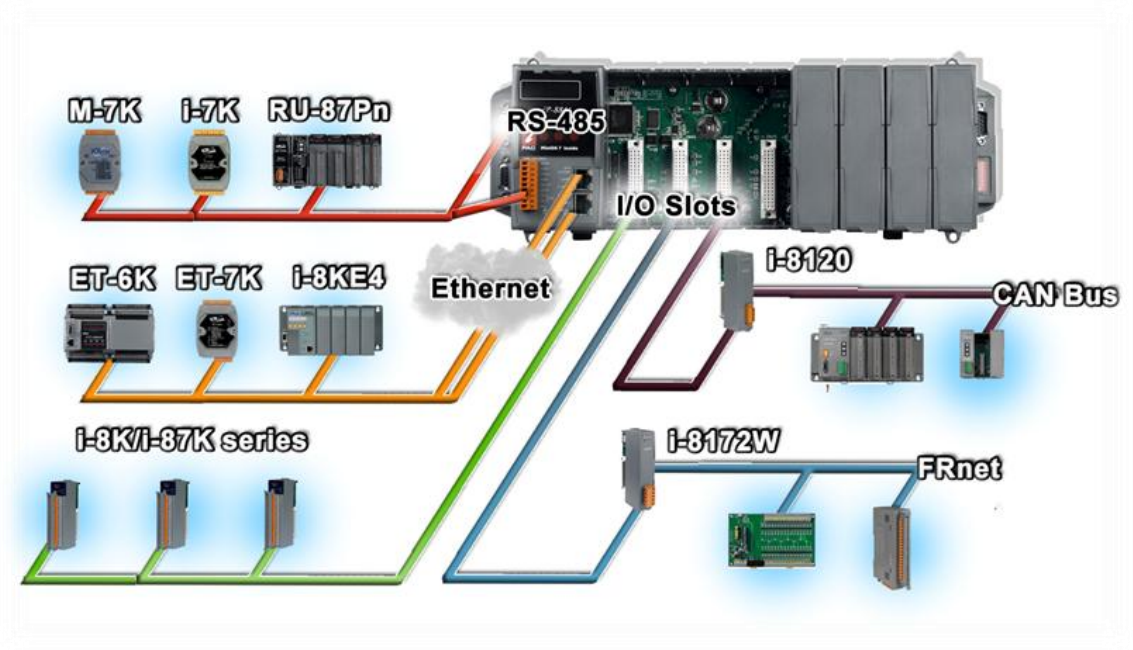
- **Dual Battery Backup SRAM (512KB)**

  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.

- **I/O Module Hot Swap Ability**

  The iPAC-8000 features hot swap which means that there is no need to power off the iPAC-8000 for replacing modules. And the OS provides a function sending plug-in and removing messages to user's applications. Using this feature, users can design its own plug-and-play applications.

- **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.

- **Dual Ethernet Ports**

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 Input**

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.

- **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. Specifications

System Software

- **OS**

  MiniOS7 (DOS-like embedded operating system)

- **Program download interface**

  RS-232(COM1) or Ethernet

- **Programming language**

  C language

- **Compilers to create .exe files**

  TC++ 1.01 (Freeware)

  TC 2.01 (Freeware)

  BC++ 3.1 ~ 5.2x

  MSC 6.0

  MSVC++ (before version 1.5.2)

CPU Module

- **CPU**

  80186 or compatible (16-bit and 80MHz)

- **SRAM**

  768K bytes

- **Flash**

  512K bytes (100,000 erase/write cycles) with Flash protection switch

- **Dual battery backup SRAM**

  512K bytes (for 5 years data retain)

CPU Module (Continue)

- **EEPROM**

  16K bytes (Data retention: 40 years; 1,000,000 erase/write cycles)

- **NVRAM**

  31 bytes (battery backup, data valid up to 5 year)

- **RTC (Real Time Clock)**

  Year-2000 compliance; seconds, minutes, hours, date of the month, year, valid up from 1980 to 2079

- **64-bit Hardware Serial Number**

  Yes

- **Watchdog Timer**

  Yes (0.8 second)

SMMI

- **5 – Digit LED Display**

Yes

- **3 – Programmable LED indicators**

Yes

- **4 – Push Buttons**

Yes

Communication Interface

- **Ehternet Port**

  RJ45 * 2 (Auto-negotiating, auto MDI/MDI-X connection, LED indicator)

  10/100 Base - TX Ethernet Controller

Communication Interface (Continue)

- **COM0**

  (Internal RS-485 interface that can communication with I-87k I/O modules)

  Non-isolation

  Baud Rate: 115200 bps

  Data bit: 8

  Parity: None, Even, Odd

  Stop bit: 1

  FIFO: 1 byte

  Note: CPU internal uart

- **COM1**

  (RS-232 used to update firmware) Non-isolation

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

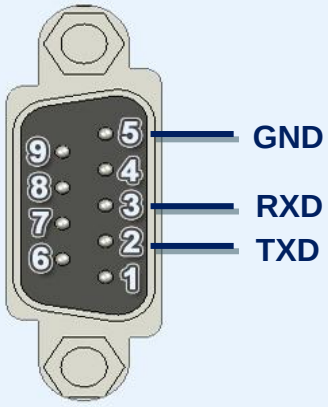
  Data bit: 7, 8

  Parity: None, Even, Odd

  Stop bit: 1

  FIFO: 1 byte

  Note: CPU internal uart



- **COM2**

  (RS-485) 3000 V_{DC} isolation; self-tuner ASIC inside

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

  Data bit: 5, 6, 7, 8

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

  Stop bit: 1, 2

  FIFO: 16 byte

  Note: 16C550 compatible

Communication Interface

- **COM3**

(RS-232/RS485)

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

Data bit: 5, 6, 7, 8

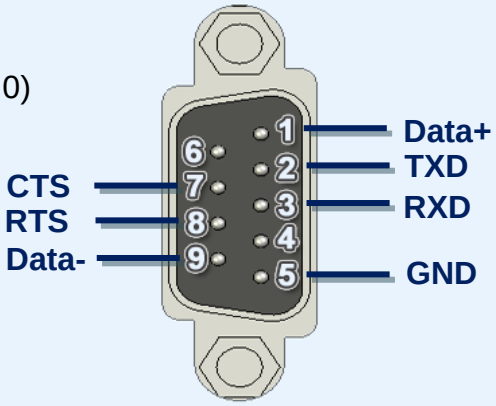
Parity: None, Even, Odd,

Mark (Always 1), Space (Always 0)

Stop bit: 1, 2

FIFO: 16 byte

Note: 16C550 compatible



COM3 can be configured as either RS-232 or RS-485, and the 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) Non-isolation

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

Data bit: 5, 6, 7, 8

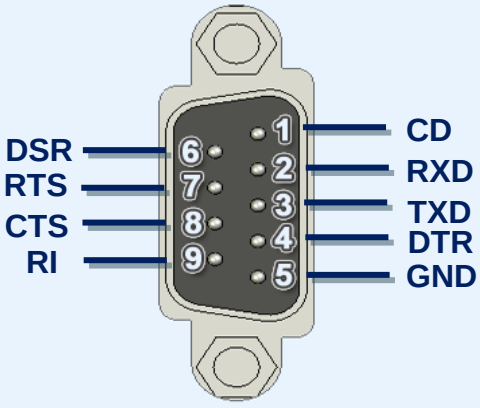
Parity: None, Even, Odd,

Mark (Always 1), Space (Always 0)

Stop bit: 1, 2

FIFO: 16 byte

Note: 16C550 compatible



I/O Expansion Slots

- **iP-8441**

  4 Slots

- **iP-8811**

  8 Slots

- **Hot Plug**

  Yes

- **Data Bus**

  8/16 bits

- **Address Bus Range**

  2K for each slot

Power Supply

- **Input Range**

  +10 ~ +30 V

- **Isolation**

  1KV

- **Capacity**

  iP-8441:

  0.85A, 5V supply to CPU, 5.51A, 5V supply to I/O expansion slots, total 30W

  iP-8841:

  0.9A, 5V supply to CPU, 5.51A, 5V supply to I/O expansion slots, total 30W

Power Supply

- **Power consumption**

  iP-8441: 6.7W (0.28A @ 24V)

  iP-8841: 7.2W (0.3A @ 24V)

Dimensions

- **iP-8441**

  230 x 132 x 111 mm

- **iP-8811**

  354 x 132 x 111 mm

Operating Environment

- **Operating Temperature**

  -25˚C ~ +75˚C

- **Storage Temperature**

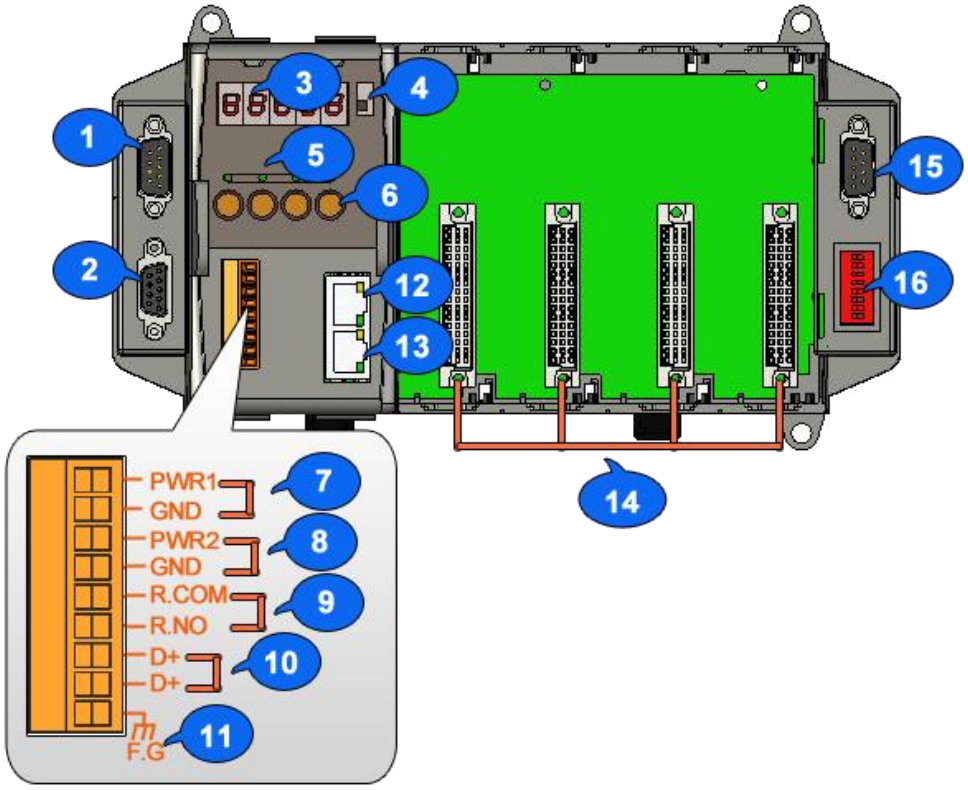
  -30˚C ~ +85˚C ,

- **Humidity**

  5 ~ 95%, Non-condensing

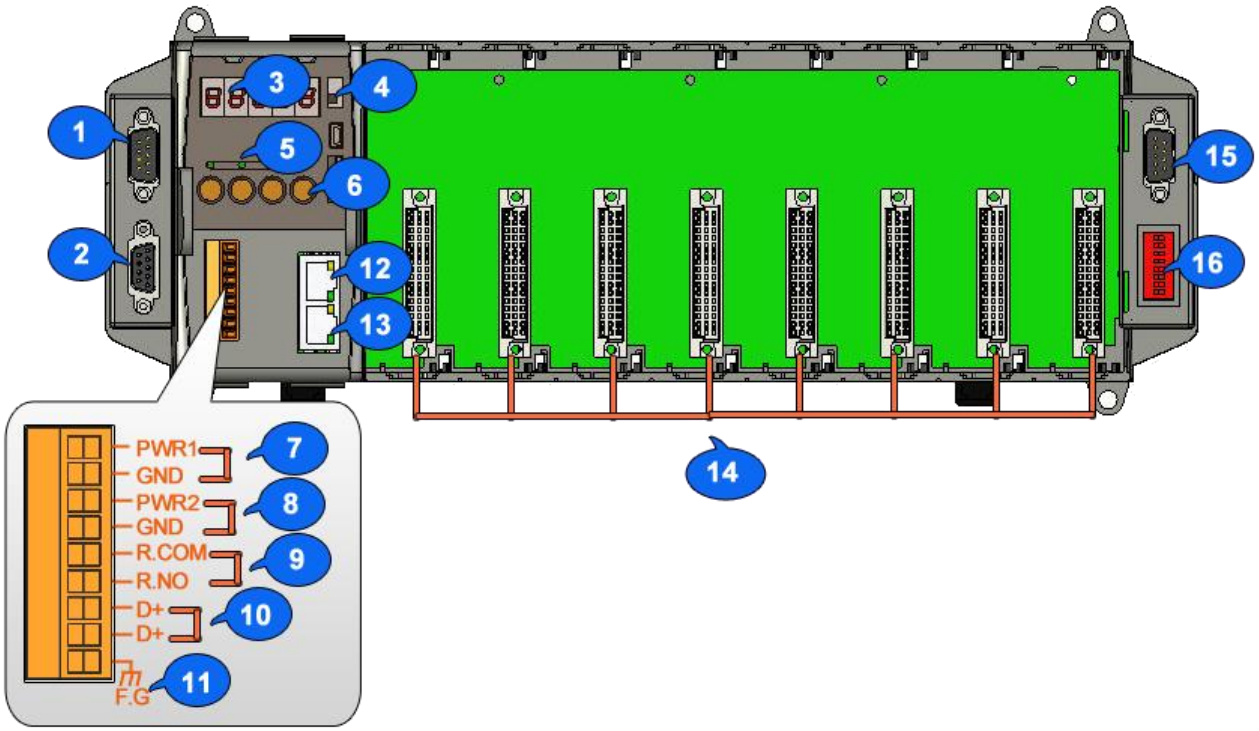
1.3. Overview

1.3.1. iP-8441



1	COM3 RS-232/RS-485	9	Relay Alarm
2	COM1 RS-232	10	COM2 RS-485
3	5-Digital LEDs	11	Frame Ground
4	Switch	12	LAN2
5	3 Programmable LED indicators	13	LAN1
6	4 Push button	14	Slot 0 ~ 3
7	PWR1	15	COM4 RS-232
8	PWR2	16	Net ID.

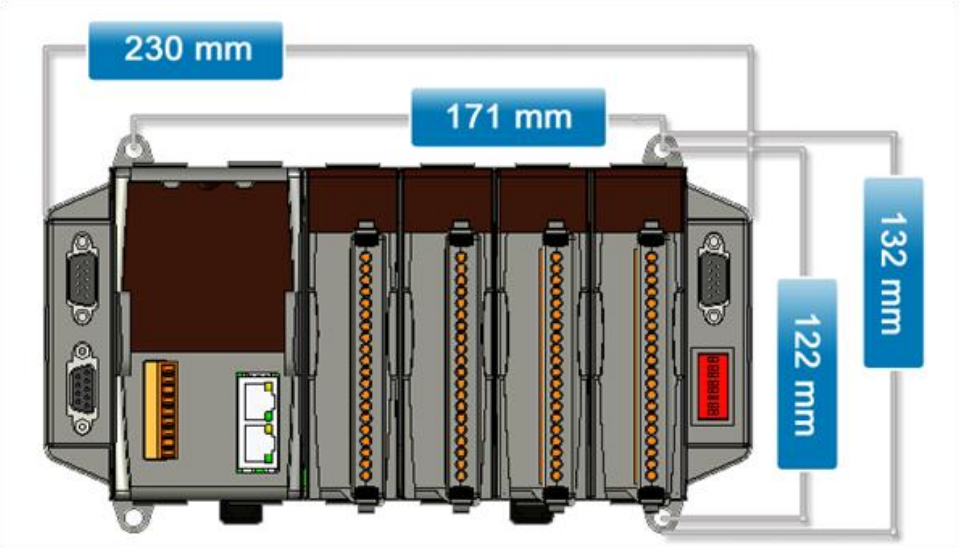
1.3.2. iP-8841



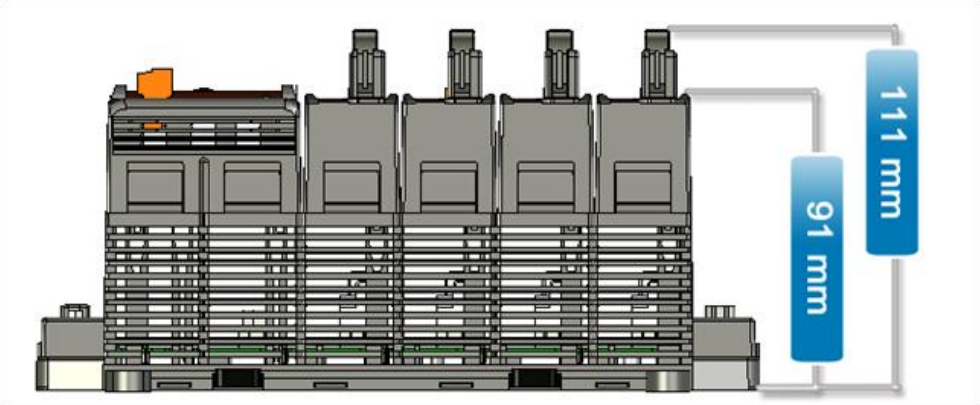
1	COM3 RS-232/RS-485	9	Relay Alarm
2	COM1 RS-232	10	COM2 RS-485
3	5-Digital LEDs	11	Frame Ground
4	Switch	12	LAN2
5	3 Programmable LED indicators	13	LAN1
6	4 Push button	14	Slot 0 ~ 7
7	PWR1	15	COM4 RS-232
8	PWR2	16	Net ID.

1.4. Dimension

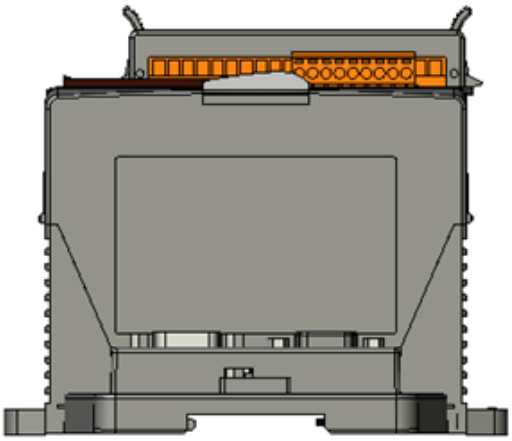
1.4.1. iP-8441



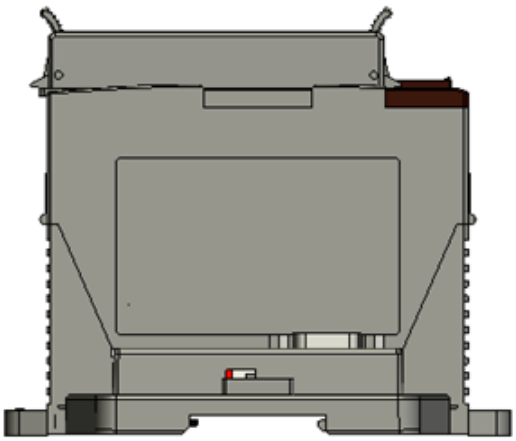
Top View



Front View

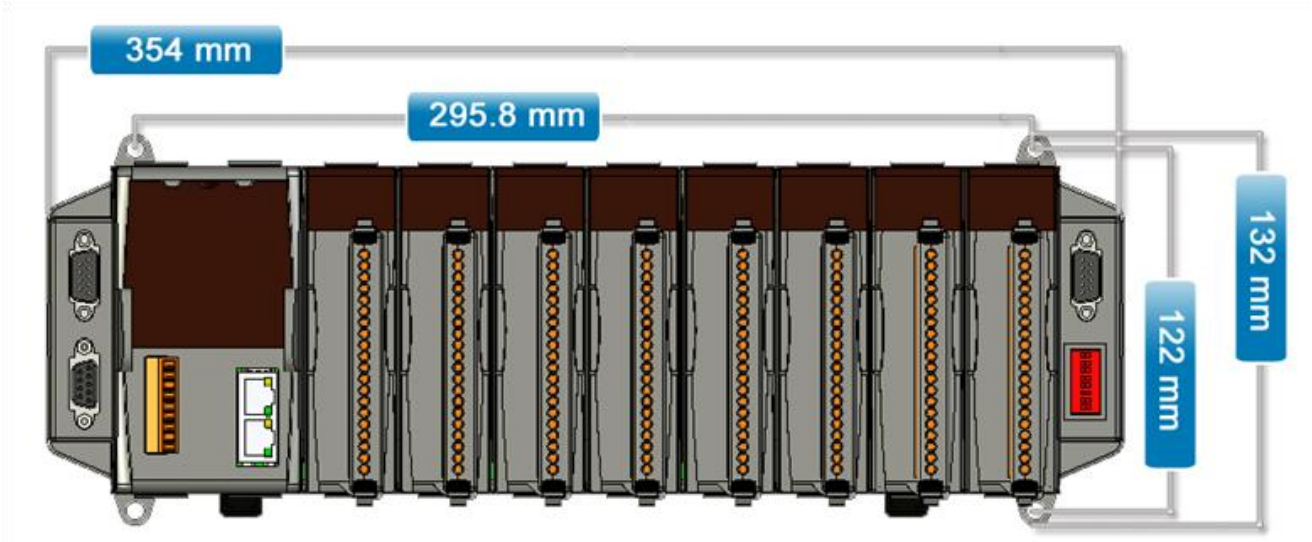


Left Side View

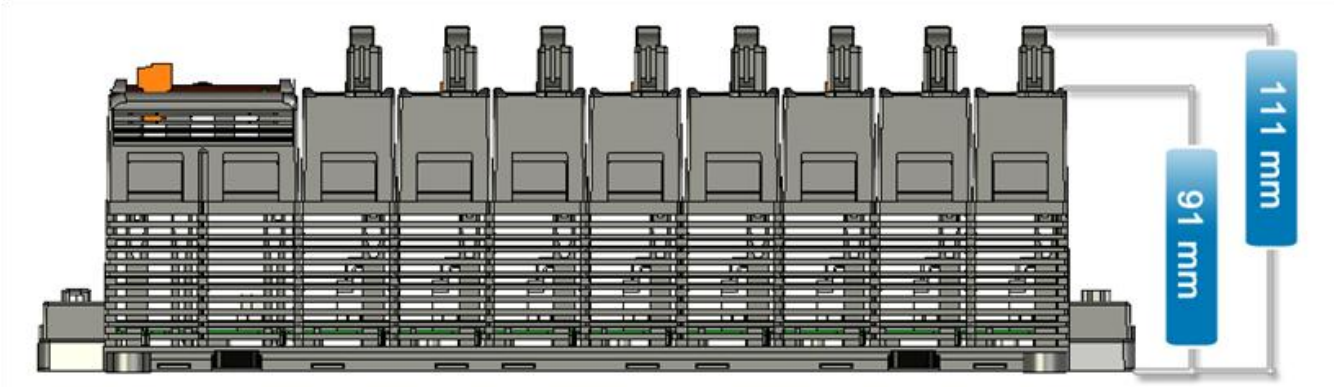


Right Side View

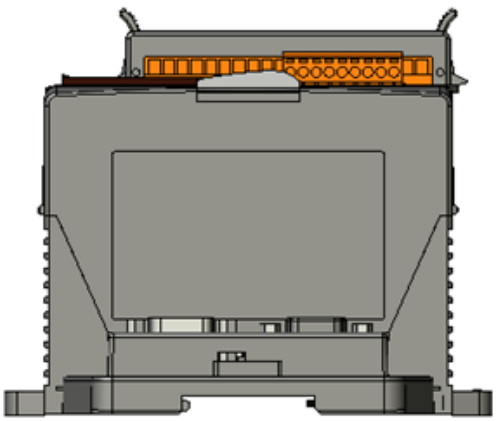
1.4.2.iP-8841



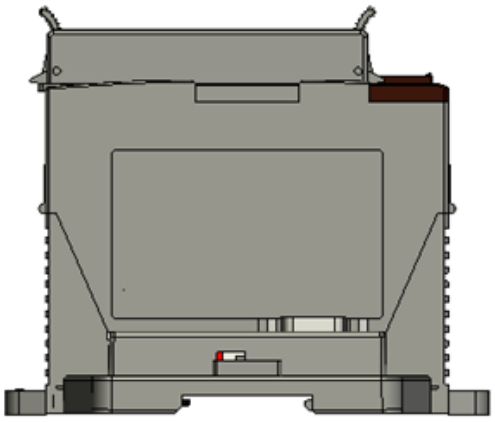
Top View



Front View



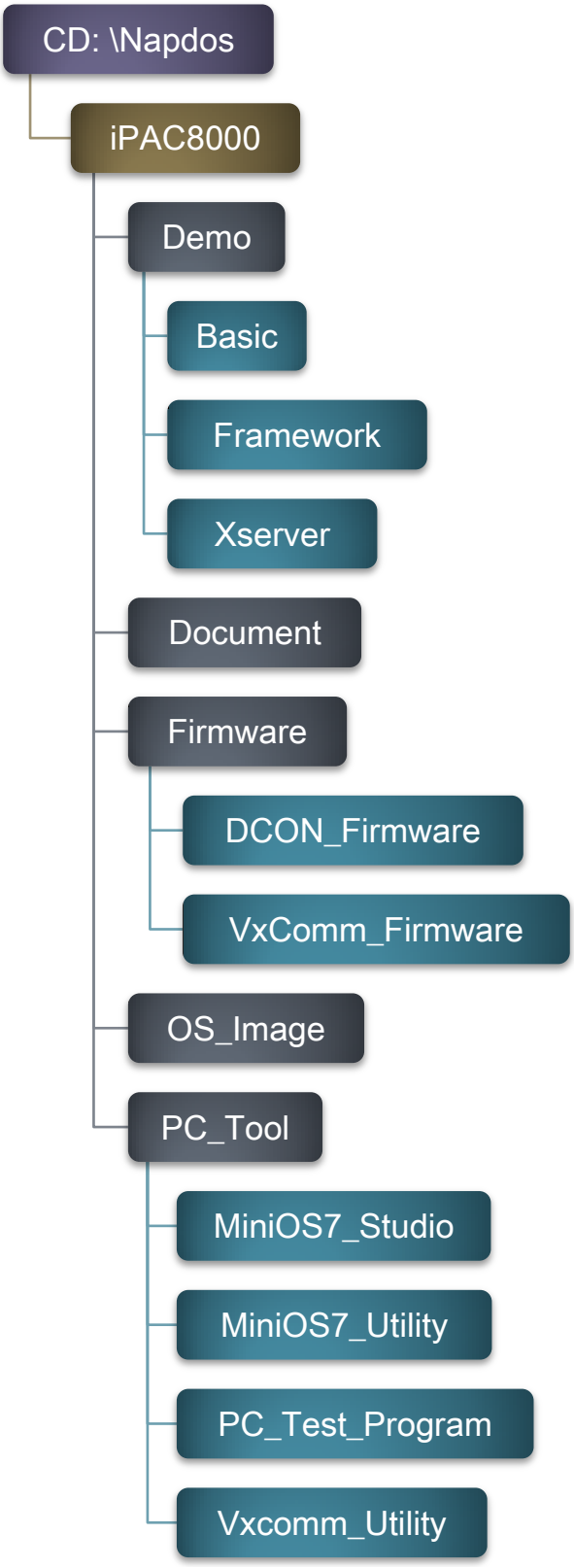
Left Side View



Right Side View

1.5. Companion CD

Thi s package comes with a CD that includes the following software and documention:



2. Quick Start

This chapter provides users with basic information needed to begin using the iPAC-8000.

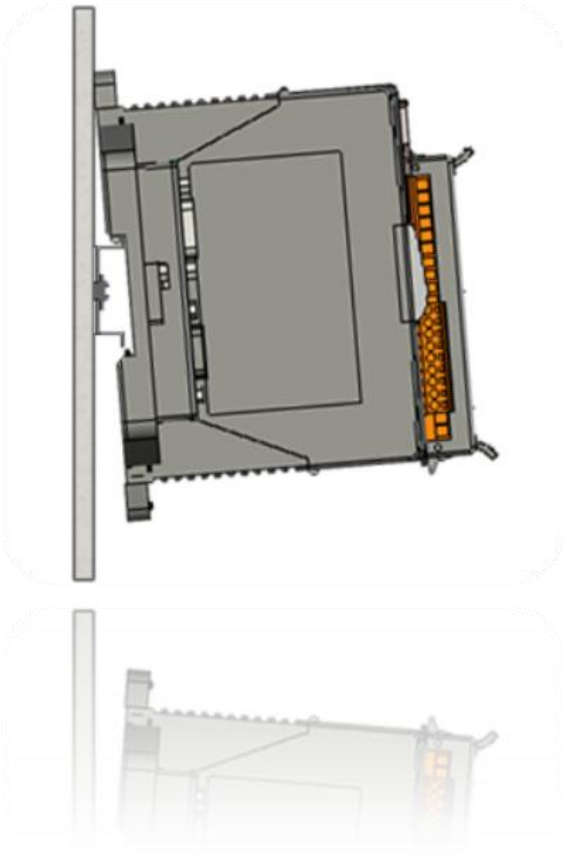
2.1. Hardware installation

2.1.1. Installing the iPAC-8000

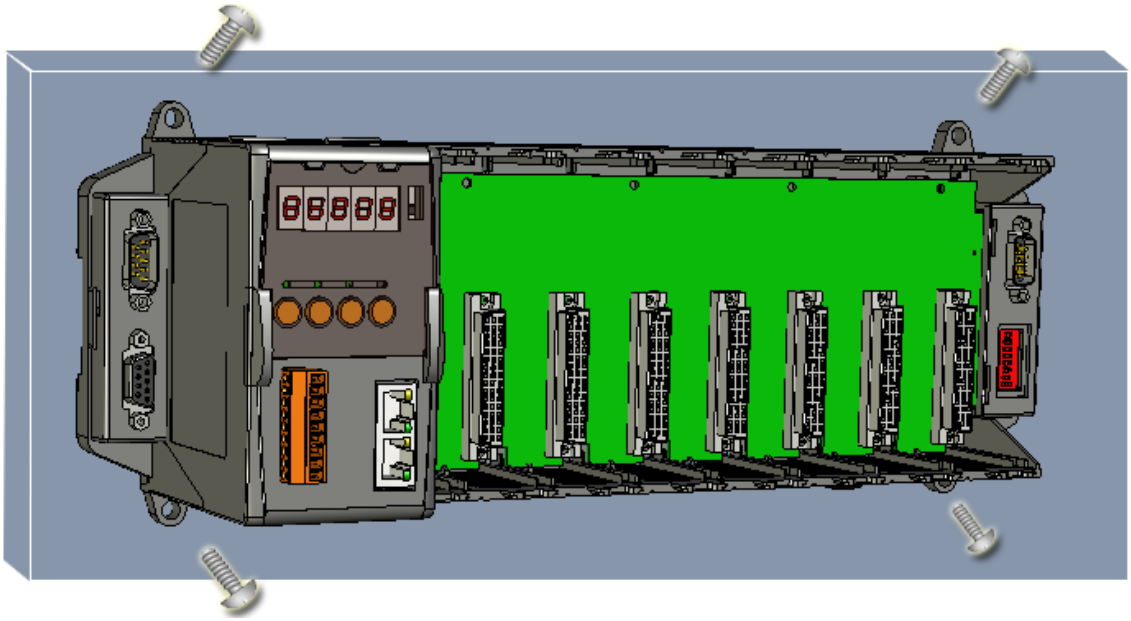
Step 1: Mounting the iPAC-8000 controller

The controller can be mounted in two different ways:

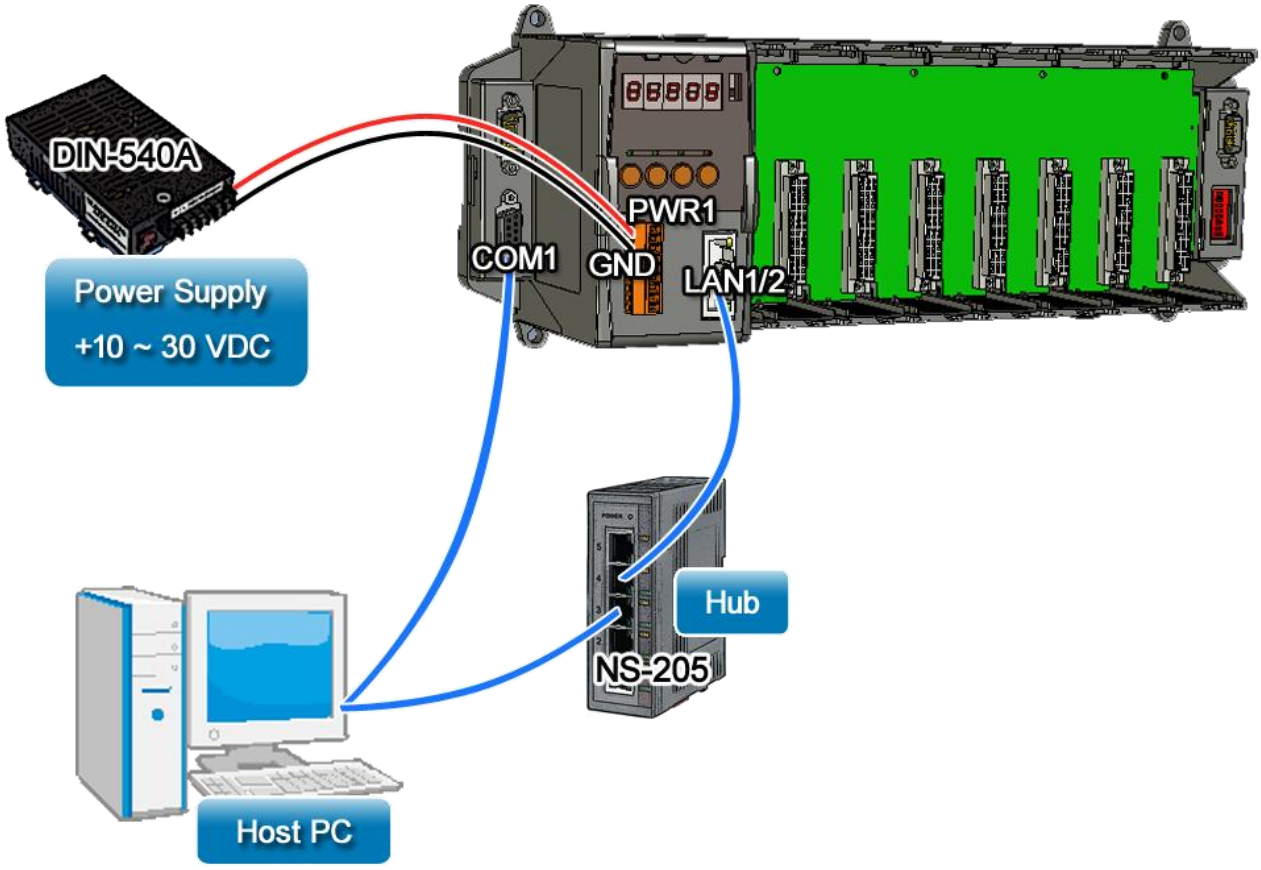
Method 1: DIN-rail mounting



Method 2: Screw panel mounting

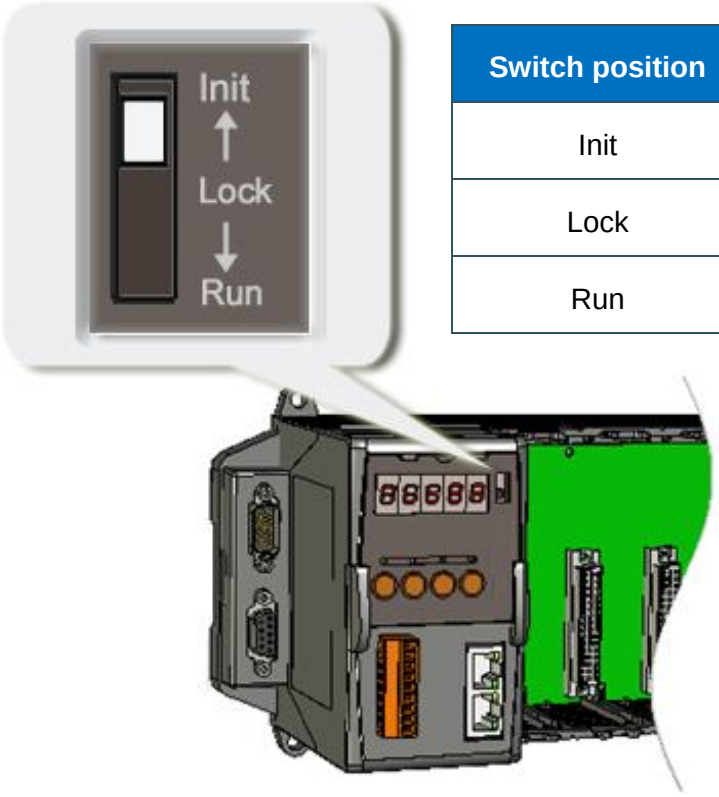


Step 2: Power the iPAC-8000



2.1.2. Operation modes of iPAC-8000

After applying power, the iPAC-8000 includes the following modes of operating for protecting the operation. Each of operation modes will be described in detail later.

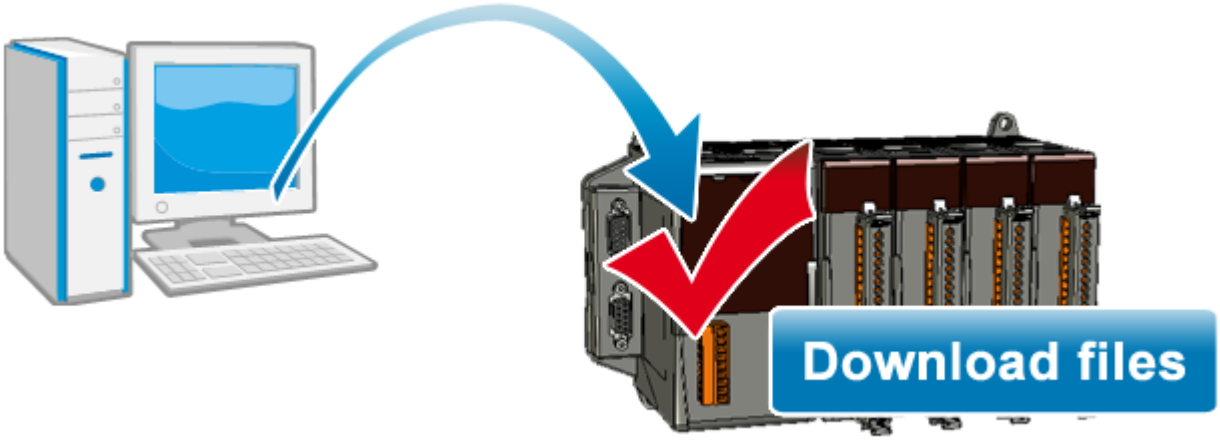


Switch position	Mode of operation
Init	Init mode
Lock	Lock mode
Run	Running mode

Operation Mode Switch	
Init	OS can not execute autoexec.bat
	Flash can be read/write
Lock	OS can execute autoexec.bat
	Flash is read only (lock)
Run	OS can execute autoexec.bat
	Flash can be read/write

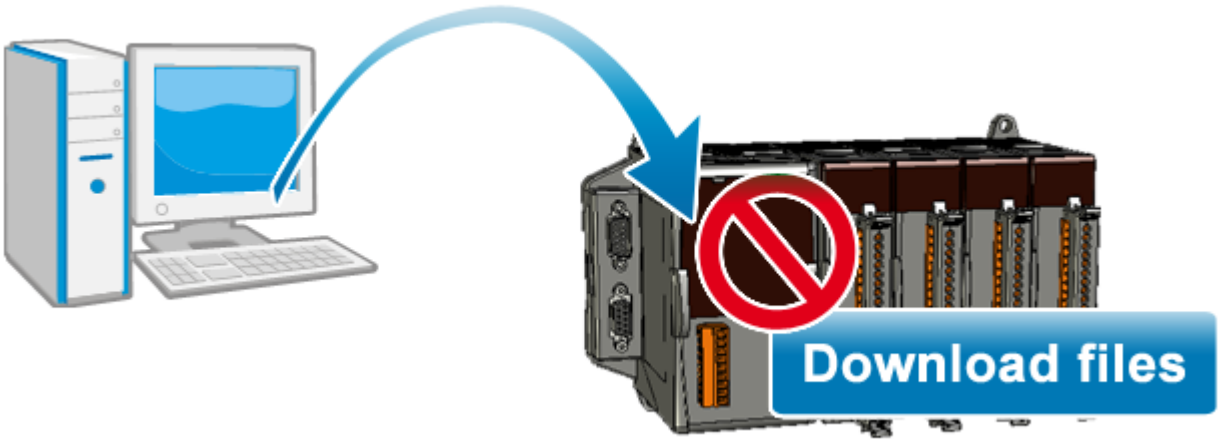
2.1.2.1. Init mode

The Init mode represents there is no program running on the iPAC-8000 and the 5-digits 7-SEG LEDs will count the number as shown below:



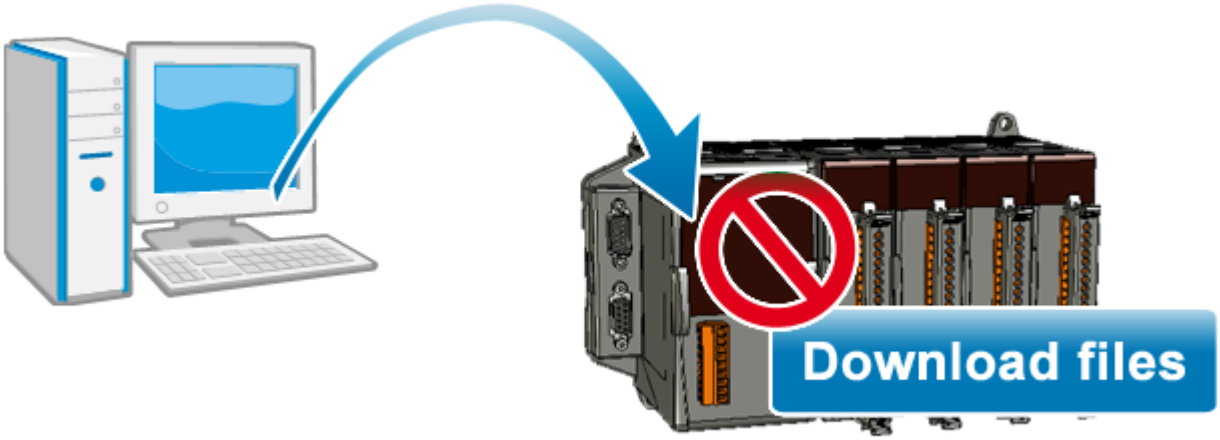
2.1.2.2. Lock mode

The lock mode prevents the data from being programmed in the memory unless an unlock sequence precedes the loading of the data that is to be programmed into the memory

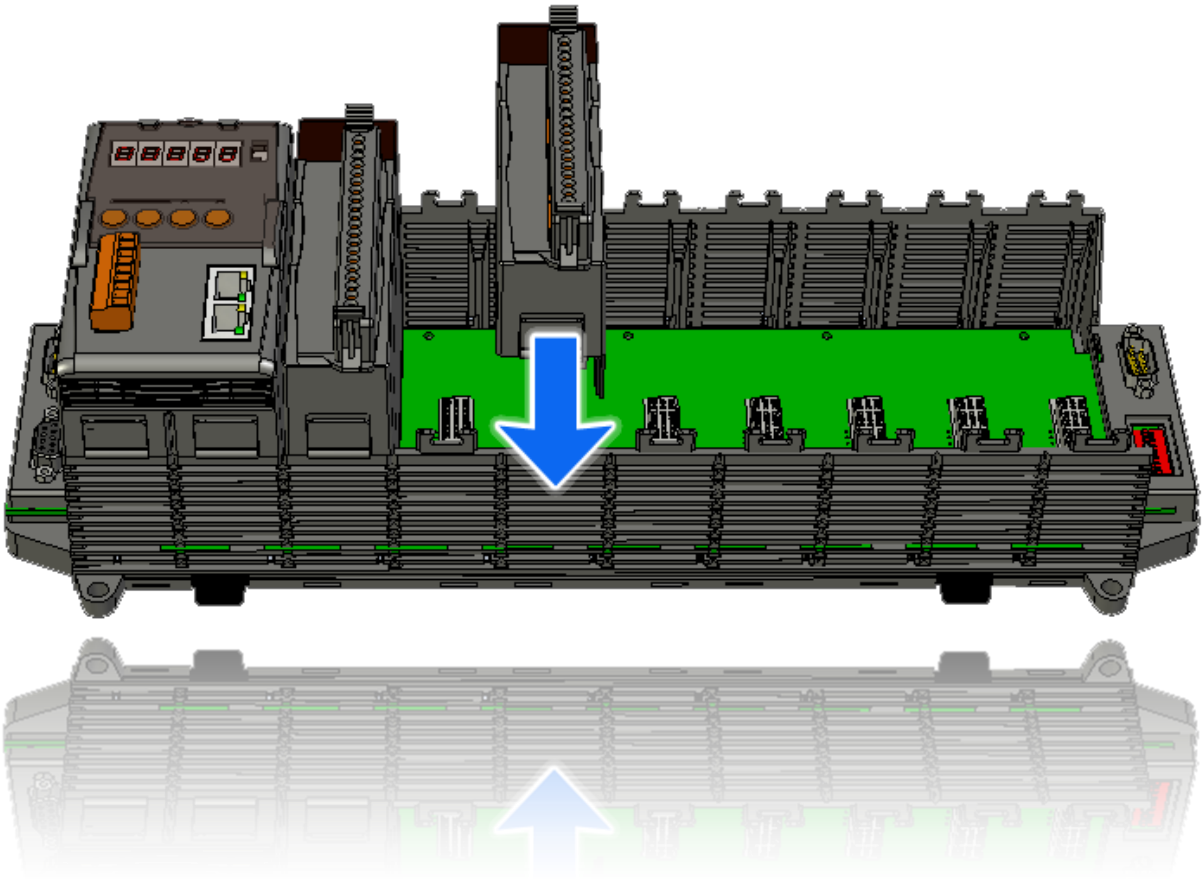


2.1.2.3. Running mode

The running mode represents there is the program running on the iPAC-8000 and the 5-digits 7-SEG LEDs will show the message according to the running program, but if during this time there is another program running on iPAC-8000, the 5-digits 7-SEG LEDs isn't managed with this program, it will stop motion at the present state.



2.1.3. Inserting the I/O module



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

http://www.icpdas.com/products/PAC/i-8000/8000_IO_modules.htm

---

By I-8K and I-87K series expansion modules, support is provided only in High Profile series.

I-8K series I/O modules (High Profile):

i-8K series I/O modules

The communication interface is parallel bus.
High profile: It can be plugged in WinPAC, LinPAC and iPAC-8000 controllers.
Low profile: It can be plugged in WinCon, LinCon and i-8000 controllers.

Type	High Profile	Low Profile	Description
Analog I/O Modules (Selection Guide)	▶ i-8017HW	i-8017H	AI modules have high sampling rate from 1K to 100Ksps (depends on software)
	NEW i-8017HS(Note 1)		
	▶ i-8024W	i-8024	
	NEW i-8037W	i-8037	
	▶ i-8040W	i-8040	
	NEW i-8040P		
	NEW i-8041W	i-8041	
	▶ i-8041AW		
	▶ i-8042W	i-8042	
	▶ i-8046W		
	▶ i-8048W	i-8048	
	▶ i-8050W	i-8050	

I-87K series I/O modules (High Profile):

i-87K series I/O modules

The communication interface is serial bus (RS-485) and the protocol is DCON. i-87K series modules are classed as
High profile: It can be plugged in WinCon, LinCon, i-8000, WinPAC, LinPAC and iPAC-8000 controllers and RU-87Pn, USB-87Pn, i-87K, RF-87K I/O expansion units.
Low profile: It can be plugged in WinCon, LinCon, i-8000 controllers and i-87K, RF-87K I/O expansion units.

Type	High Profile	Low Profile	Max. baudrate	Description
Analog I/O Modules (Selection Guide)	▶ i-87005W		115.2 Kbps	Slew rate of AO channels are programmable
	NEW i-87013W	i-87013		
	NEW i-87015			
	NEW i-87015P			
	▶ i-87016W			
	NEW i-87017W	i-87017		
	NEW i-87017W-A5			
	NEW i-87017R			
	NEW i-87017RC			
	NEW i-87018W	i-87018		
	NEW i-87018R			
	NEW i-87018Z			
	NEW i-87019R			
		i-87022		

Step 1: Read the relevant documentation

- The document for I-8k series I/O modules (High Profile) is located at:

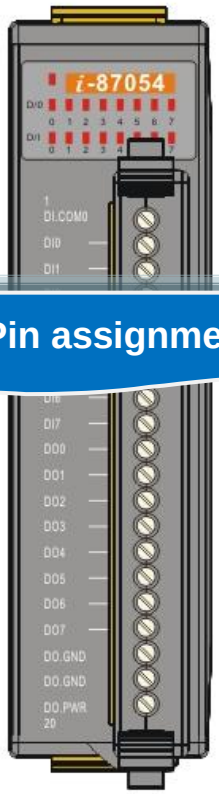
  CD:\NAPDOS\DCON\IO_Module\
  ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/dcon/io_module

- The document for I-87k series I/O modules (High Profile) is located at:

  CD:\NAPDOS\DCON\IO_Module\
  ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/dcon/io_module

  All documents includes the I/O module specifications, pin assignments and wire connections.

For example, Pin Assignments and Wiring connections for the i-87054 module are as follows:



Pin assignments

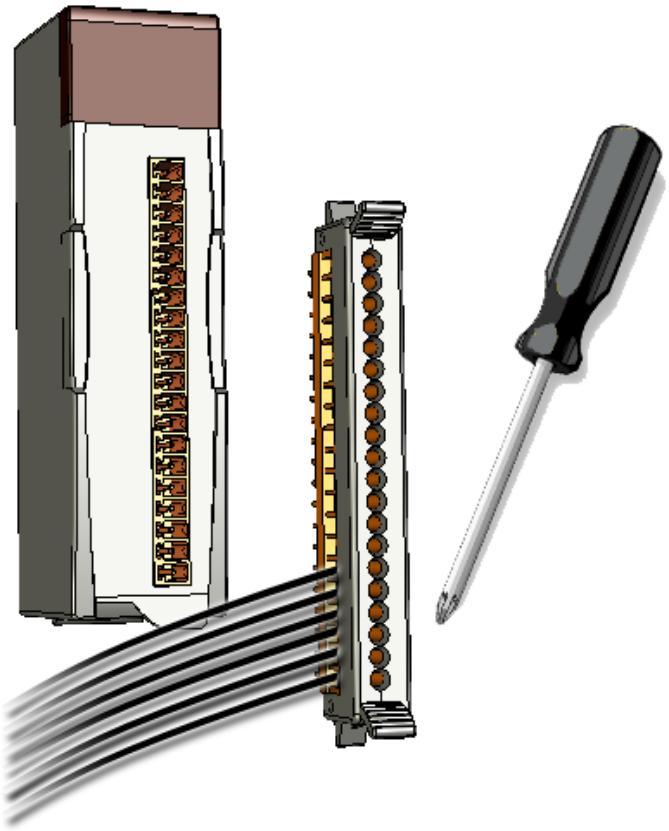
Terminal No.	Pin Assignment Name
01	DI.COM0
02	DI0
03	DI1
04	DI2
05	DI3
06	DI4
07	DI5
08	DI6
09	DI7
10	DO0
11	DO1
12	DO2
13	DO3
14	DO4
15	DO5
16	DO6
17	DO7
18	DO.GND
19	DO.GND
20	DO.PWR

Wiring connections

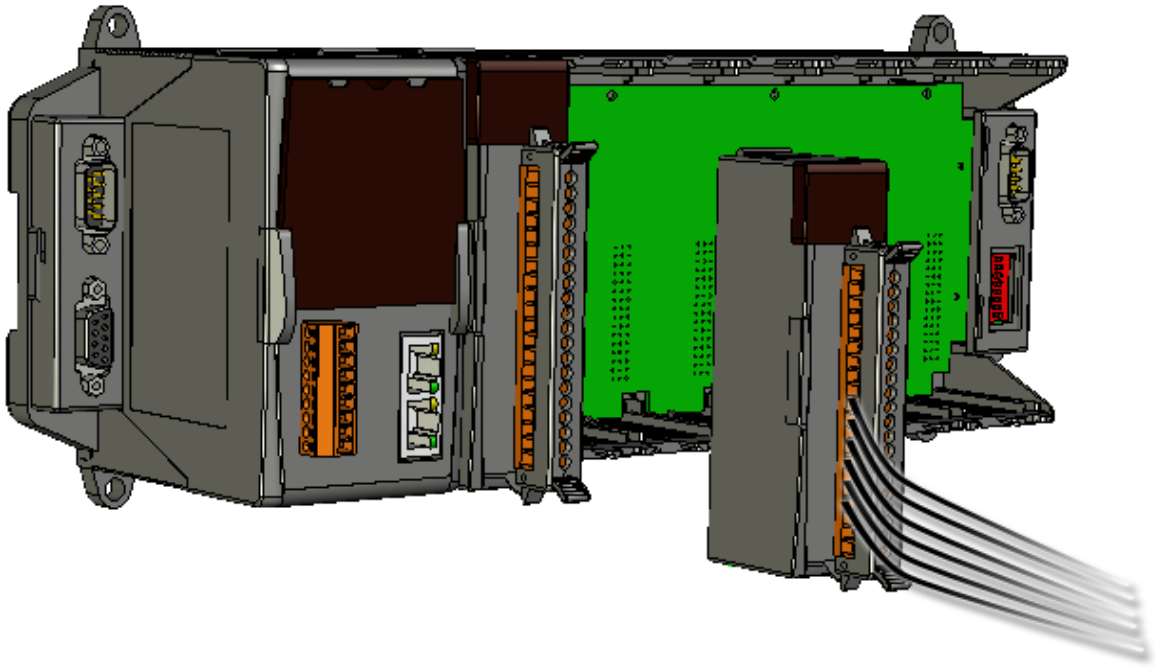
Input Type	ON State LED ON Readback as 1	OFF State LED OFF Readback as 0
Relay Contact	Relay ON Relay Close DI.COM DIx	Relay Off Relay Open DI.COM DIx
TTL/CMOS Logic	Voltage < 1V Logic Power Logic Level Low DI.COM DIx	Voltage > 3.5V Logic Power Logic Level High DI.COM DIx
NPN Output	Open Collector On ON DI.COM DIx	Open Collector Off OFF DI.COM DIx
		Open Collector Off OFF DI.COM DIx

Output Type	ON State LED ON Readback as 1	OFF State LED OFF Readback as 0
Drive Relay	Relay ON DO.PWR DOx DO.GND	Relay Off DO.PWR DOx DO.GND
Resistance Load	DO.PWR DOx DO.GND	DO.PWR DOx DO.GND

Step 2: Connect the wires



Step 3: Insert the I/O module

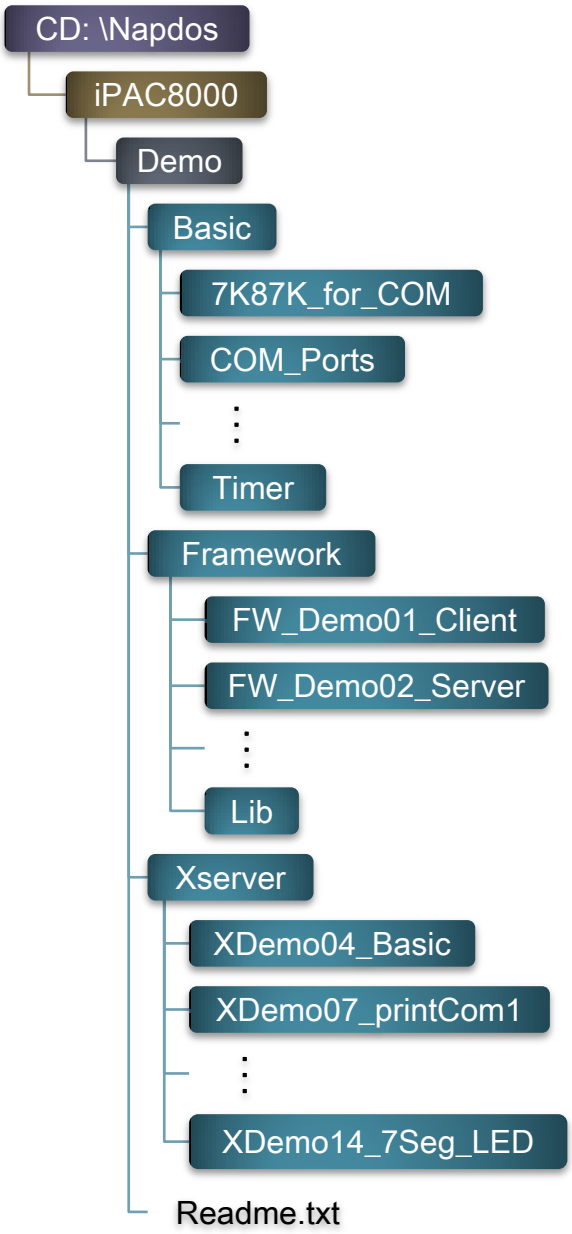


2.2. Software installation

All software resources are included on the companion CD, the following steps will help you to install the resources and software from the companion CD.

Step 1: Copy the "Demo" folder from the companion CD to the Host PC

The folder is an essential resource for users developing custom programs which contains libraries, header files, demo programs and more information as shown below:



Step 2: Install the MiniOS7 Utility

The MiniOS7 Utility is a tool that can be used to configure and upload files to the controller and is located at:

CD:\Napdos\minios7\utility\minios7_utility\

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

2.3. MiniOS7 Utility for downloading programs

Before you begin using the MiniOS7 Utility to download programs, ensure that the controller is connected to the Host PC.

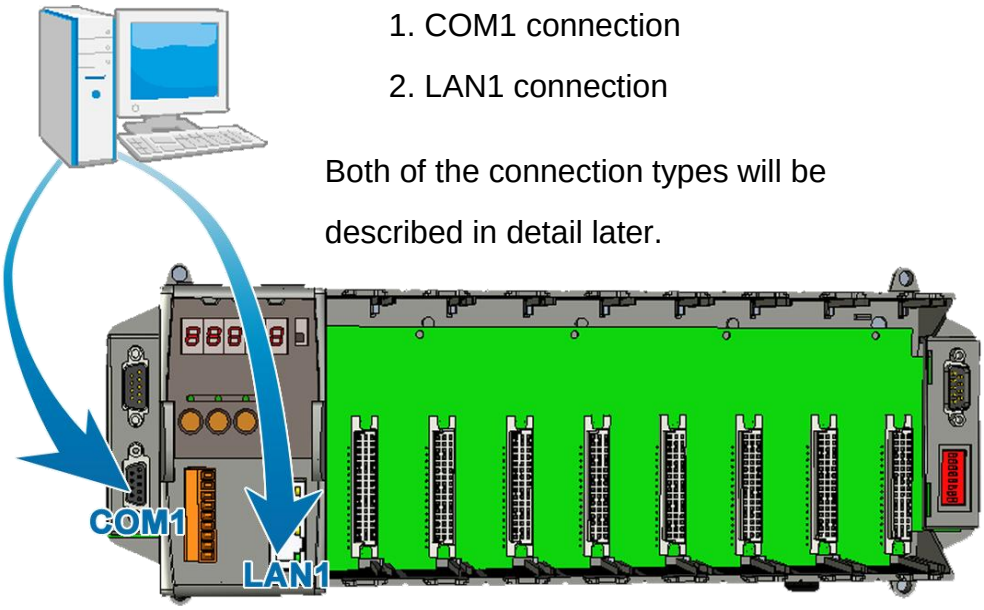
The download process has the following main steps: .

1. Establishing a connection
2. Download and executing programs on the controller
3. Making programs start automatically

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

2.3.1. Establishing a connection between the PC and the iPAC-8000

Connect the PC to the iPAC-8000 with the following connection types:

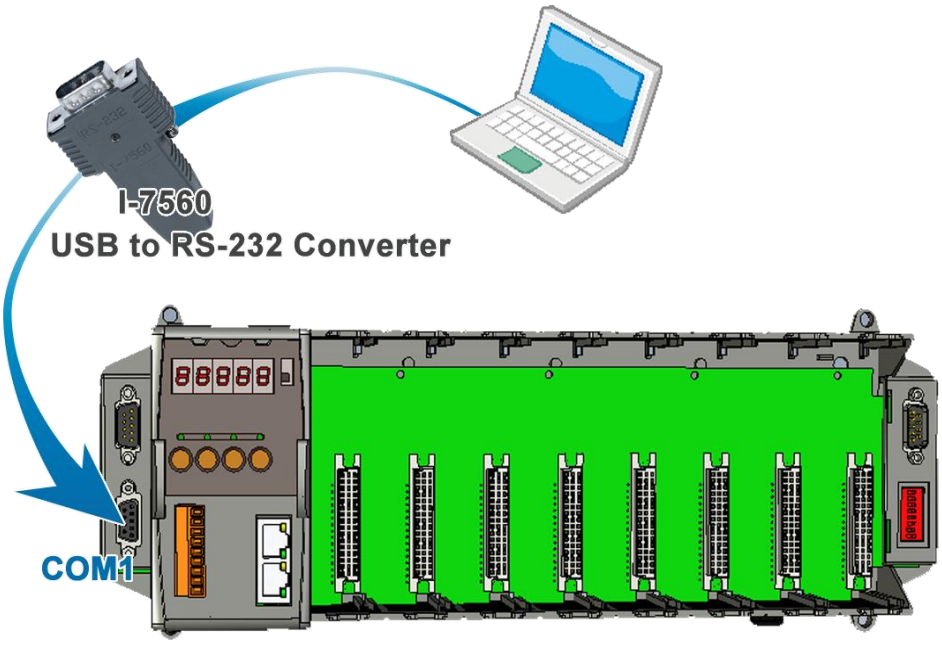


1. COM1 connection
2. LAN1 connection

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

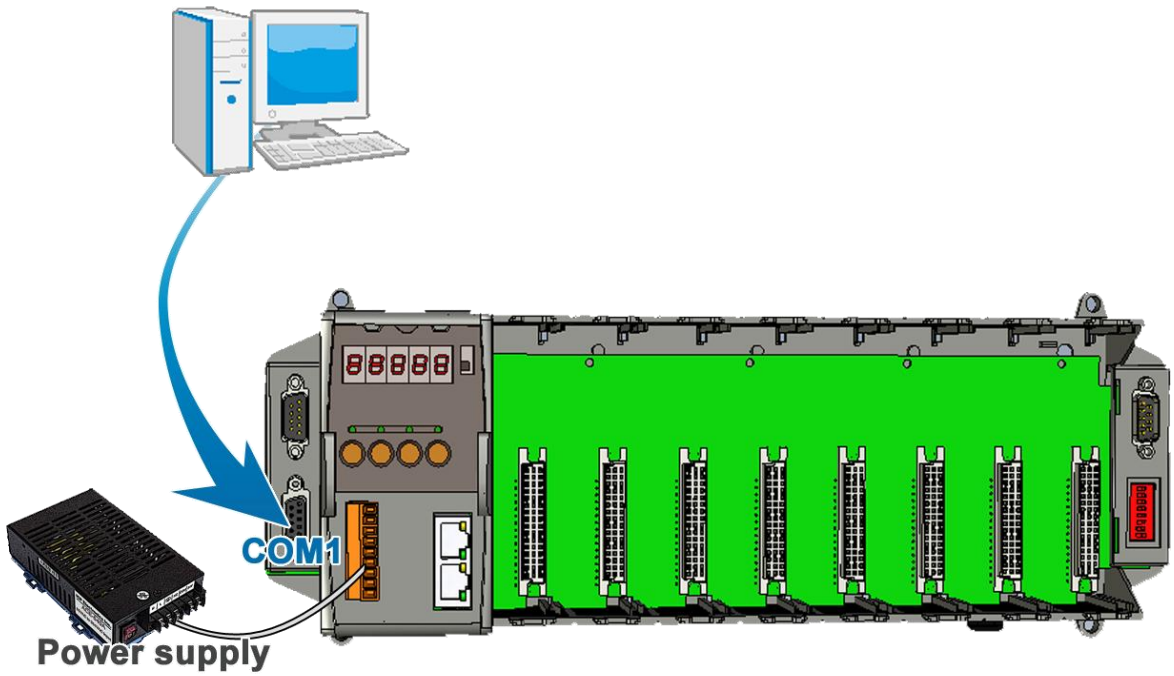
If PC or Laptop do not have a native COM port can be used making a connection, There is a simple way to use one I-7560, a USB to RS-232 converter, to convert any standard USB port into a RS-232 port.

For more detailed description on how to add a I-7560 converter to establish a USB connection, please refer to section “2.3.1.3. Steps to establish a USB connection using a I-7560 converter”

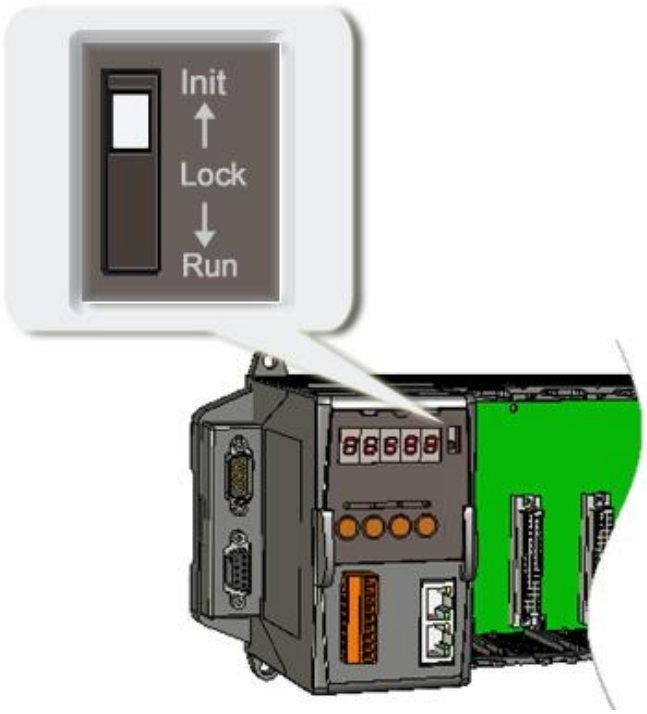


2.3.1.1. Steps to use a COM1 connection

Step 1: Connect the iPAC-8000 to the host PC using a COM1 connection

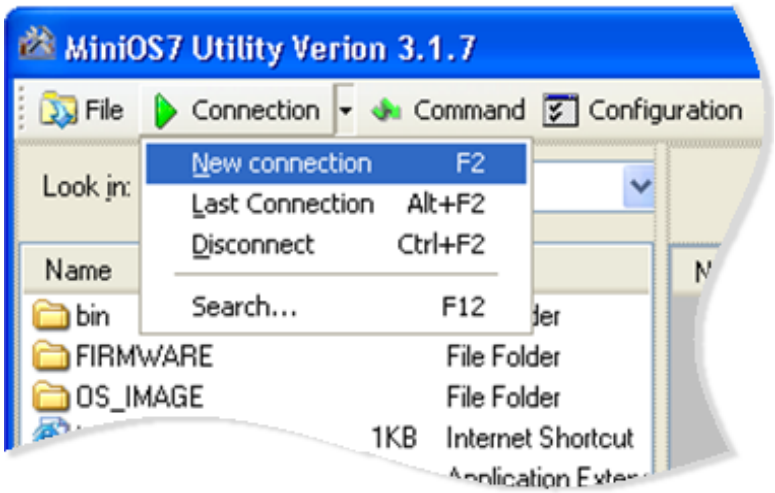


Step 2: Reboot the iPAC-8000 at Init mode

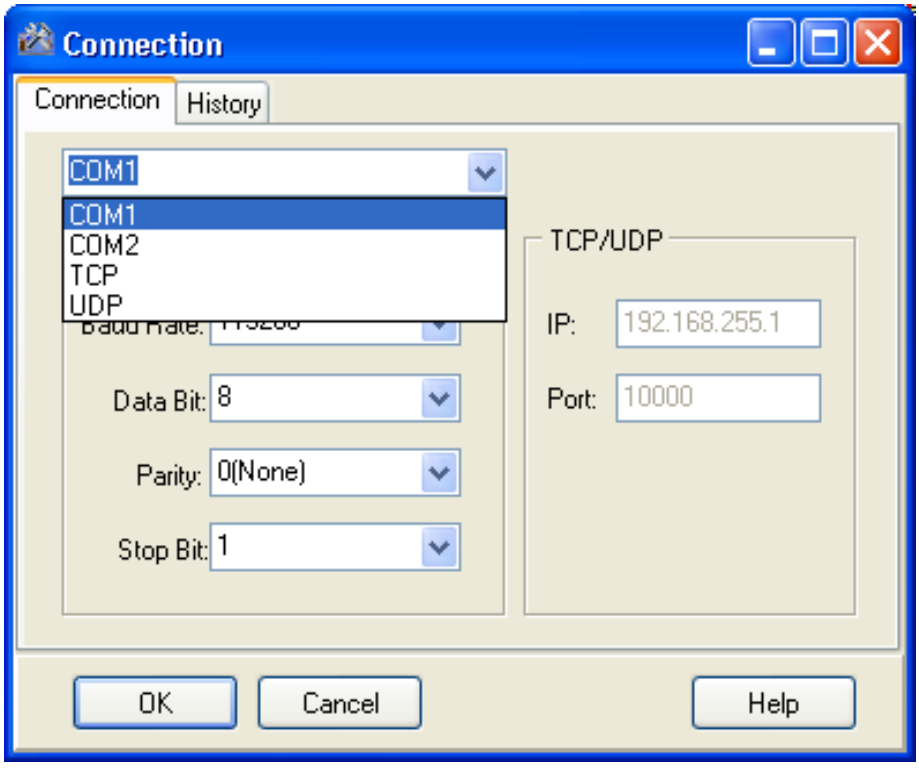


Step 3: Run the MiniOS7 Utility

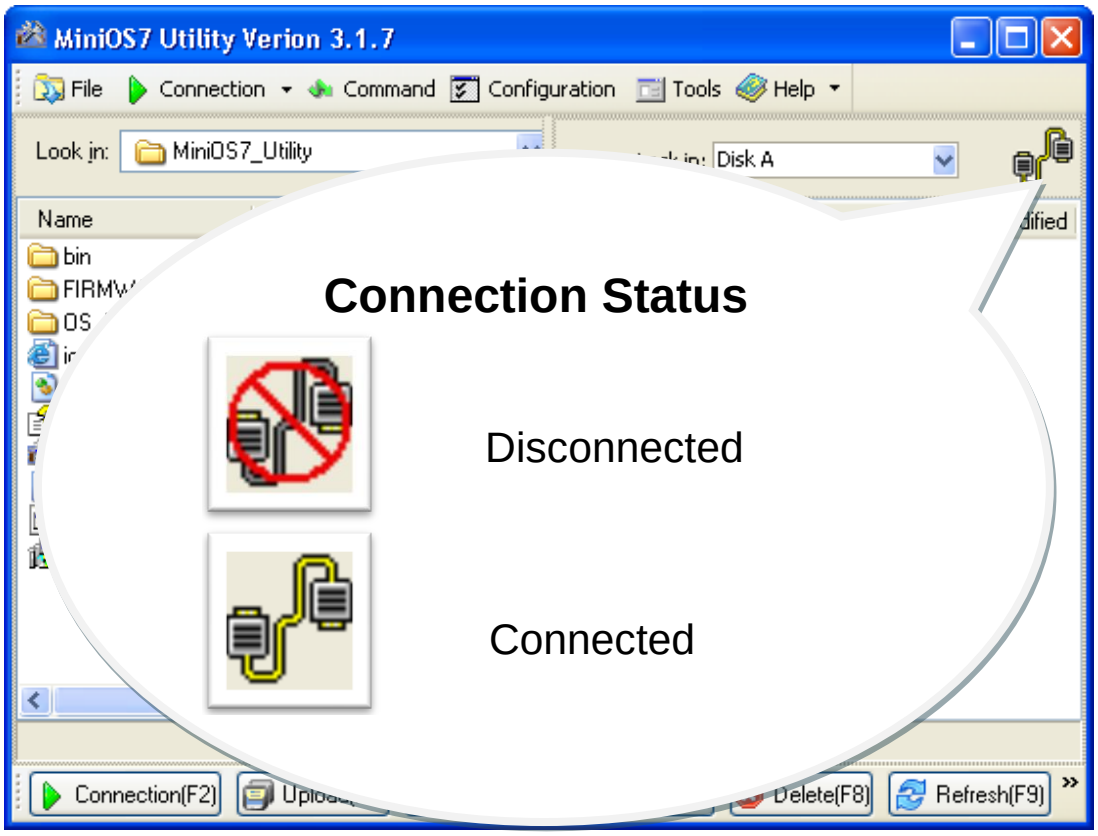
Step 4: Click the "New connection" from the "Connection" menu



Step 5: On the "Connection" dialog box, select "COM1" from the drop down list

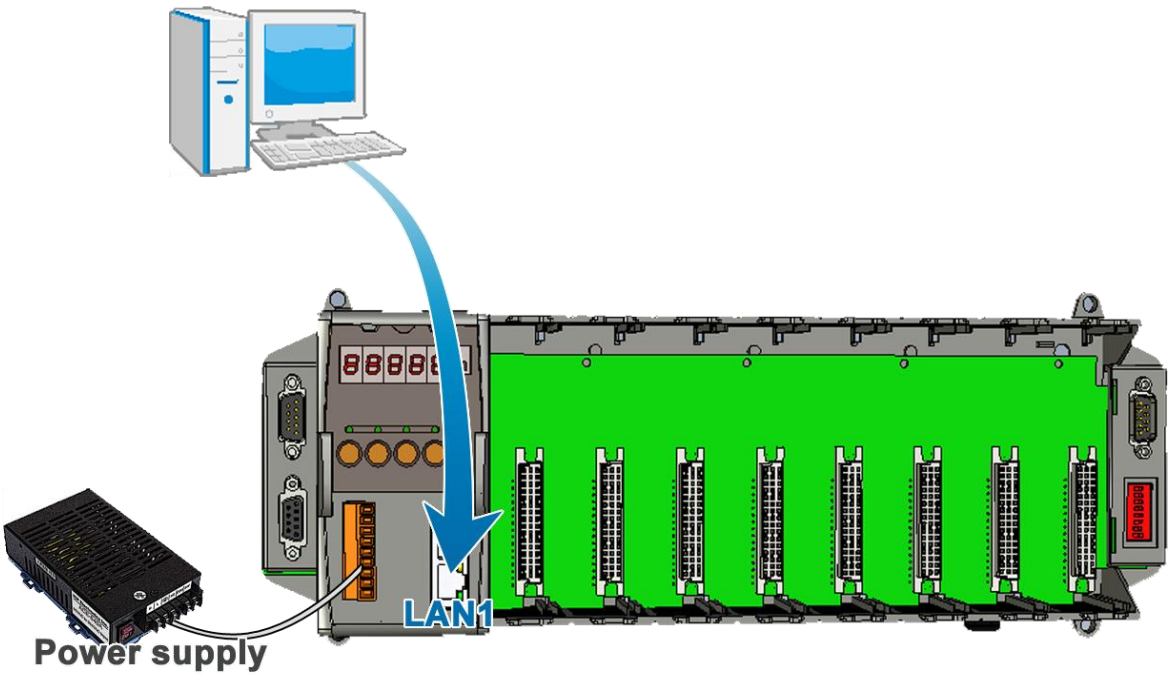


Step 6: The connection has already established

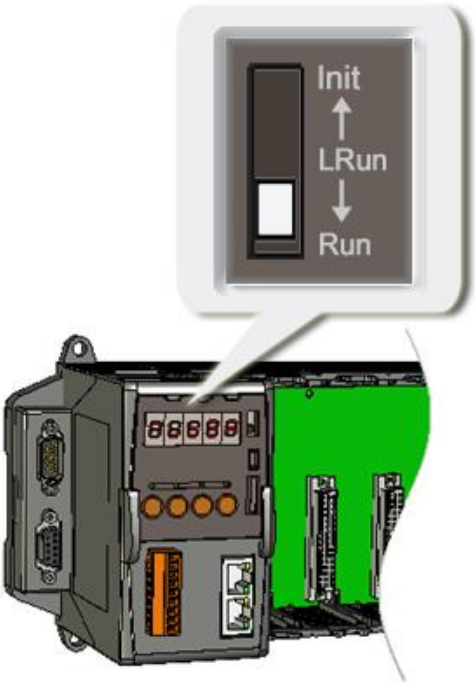


2.3.1.2. Steps to use a LAN1 connection

Step 1: Connect iPAC-8000 to the host PC using a LAN1 connection



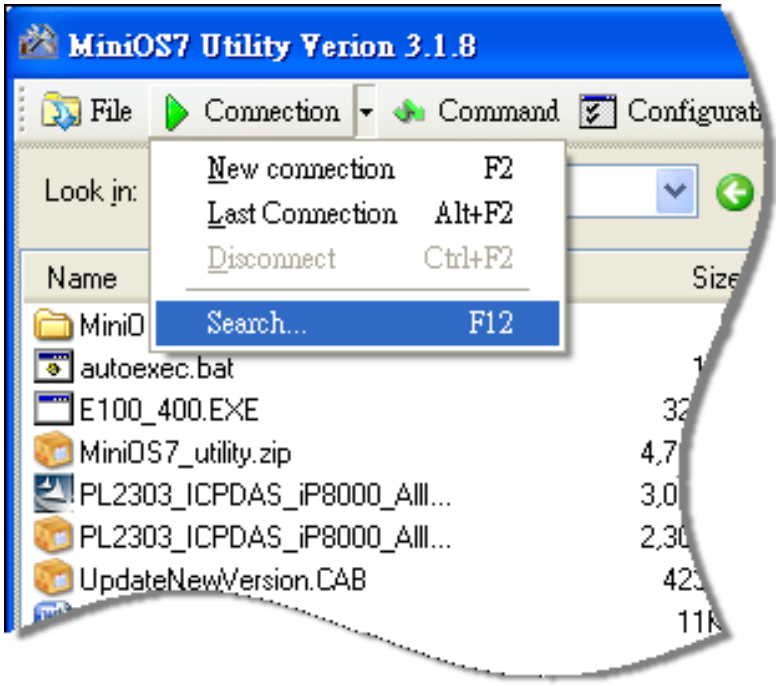
Step 2: Reboot the iPAC-8000 at Init mode



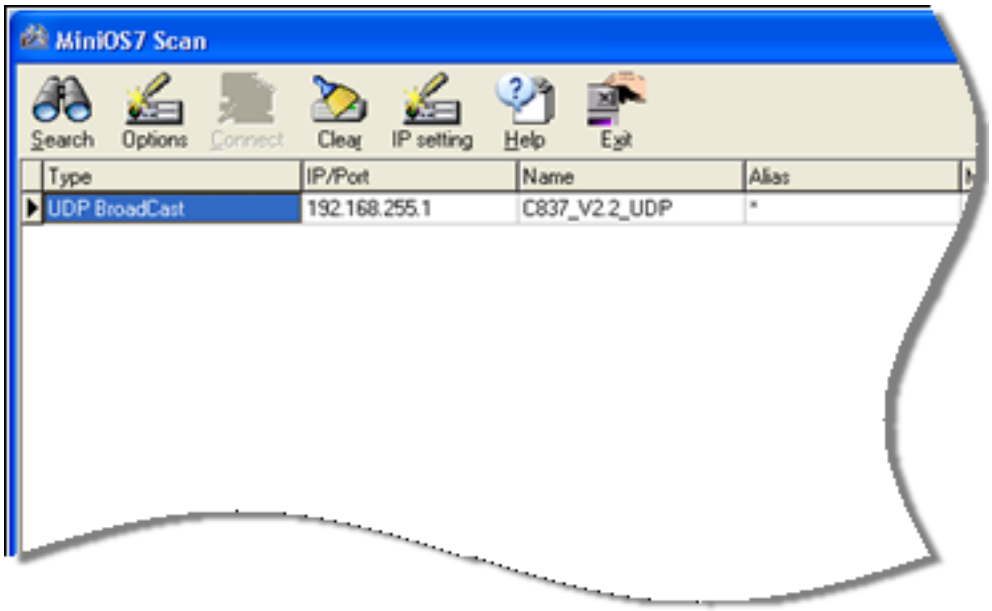
Step 3: Run the MiniOS7 Utility

If you have a DHCP server to automatically configure IP address, then you don't have to manually configure IP address, you can skip directly skip to step 8.

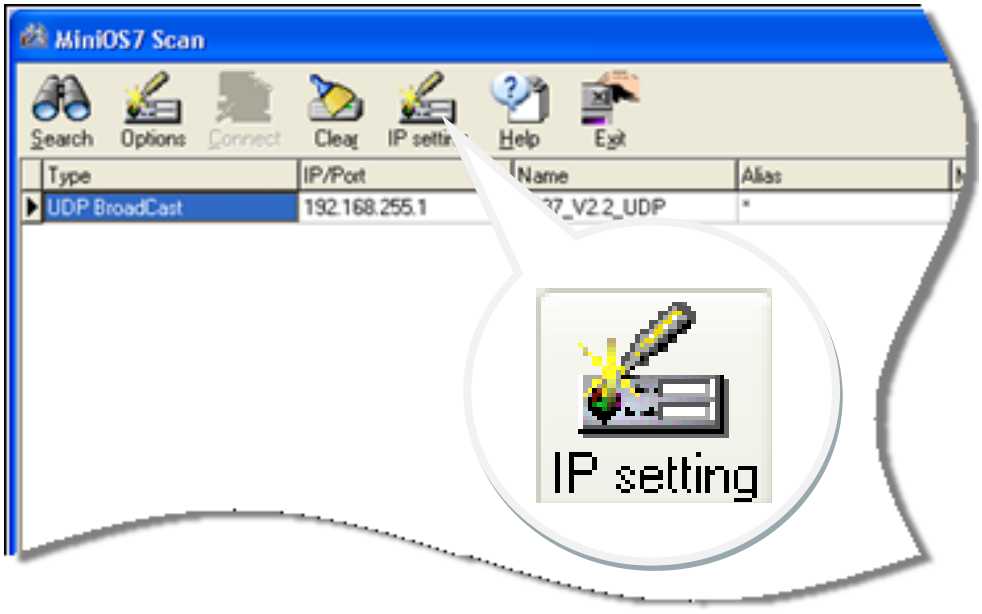
Step 4: Click the "Search" from the "Connection" menu



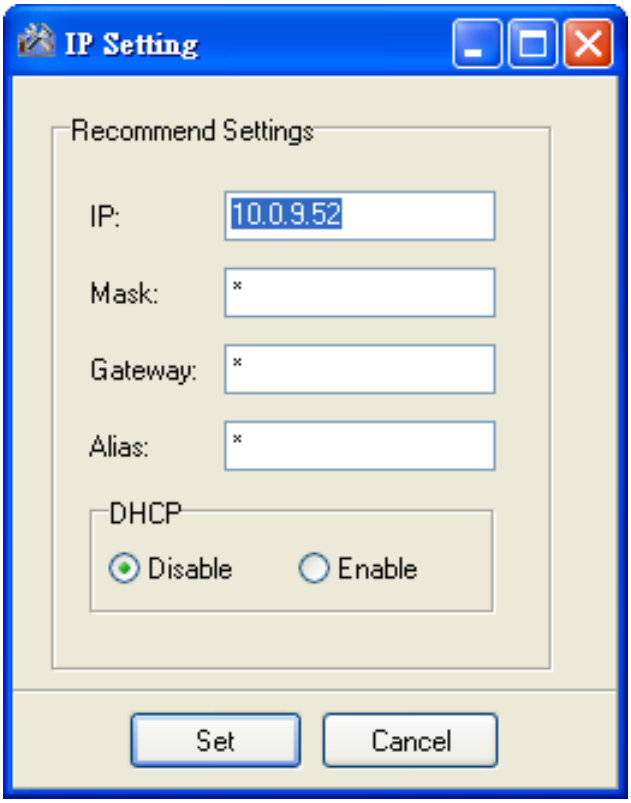
Step 5: On the “MiniOS7 Scan” dialog box, select “192.168.255.1” from the list



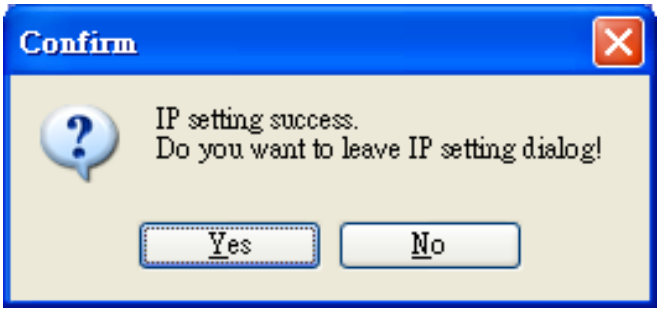
Step 5: Select “IP setting” button from the toolbar



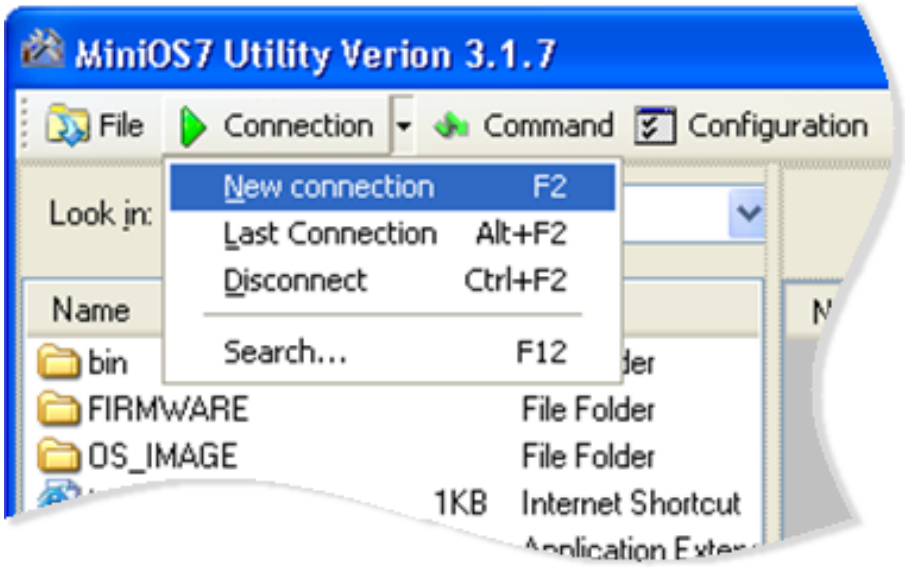
Step 6: On the “IP Setting” dialog, set the “IP” settings and then click the “Set” button



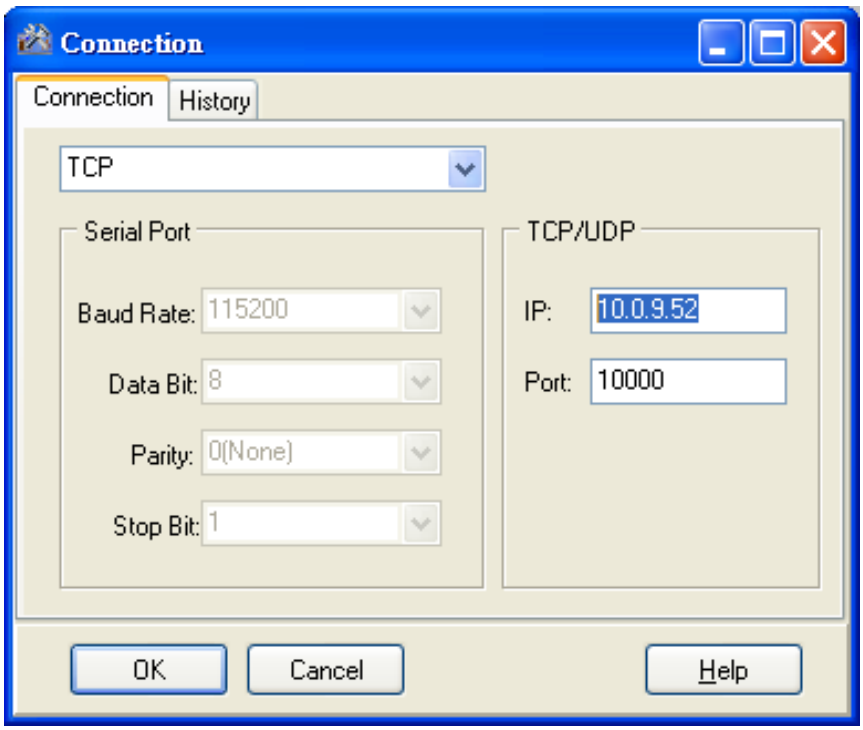
Step 7: On the “Confirm” dialog, click the “Yes” button to exit



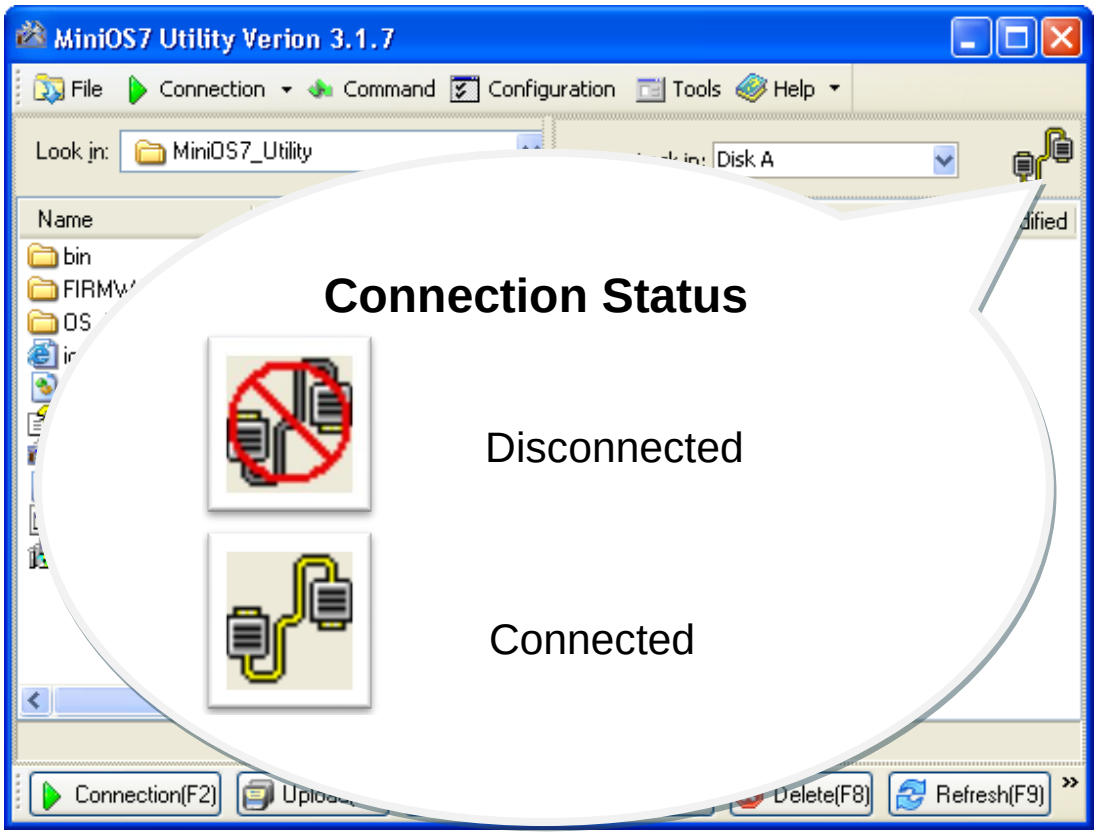
Step 8: Click the “New connection” from the “Connection” menu



Step 9: On the “Connection” dialog box, select “TCP” from the drop down list and enter the “IP” which just assigns



Step 10: The connection has already established

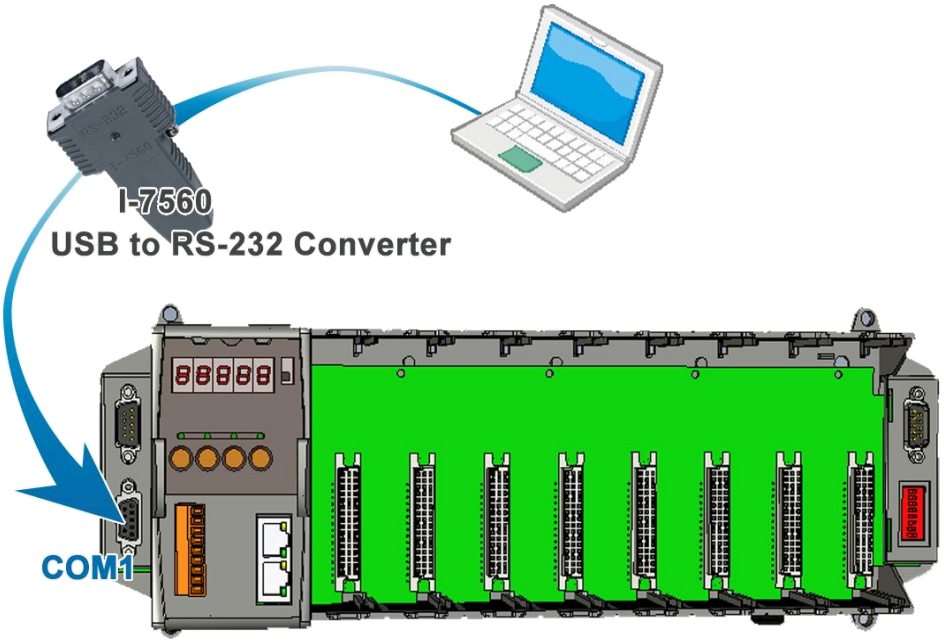


2.3.1.3. Steps to establish a USB connection using a I-7560 converter

Step 1: Connect iPAC-8000 to the PC or Laptop using a I-7560 converter

For more information about I-7560 converter, please refer to:

http://www.icpdas.com/products/Remote_IO/i-7000/i-7560.htm



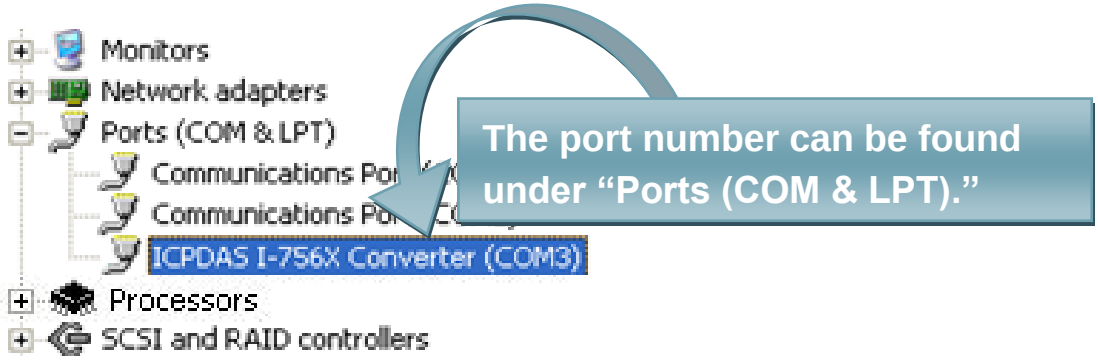
Step 2: Install the I-7560 driver

The I-7560 driver can be obtain from:

CD:\ NAPDOS\7000\756x\

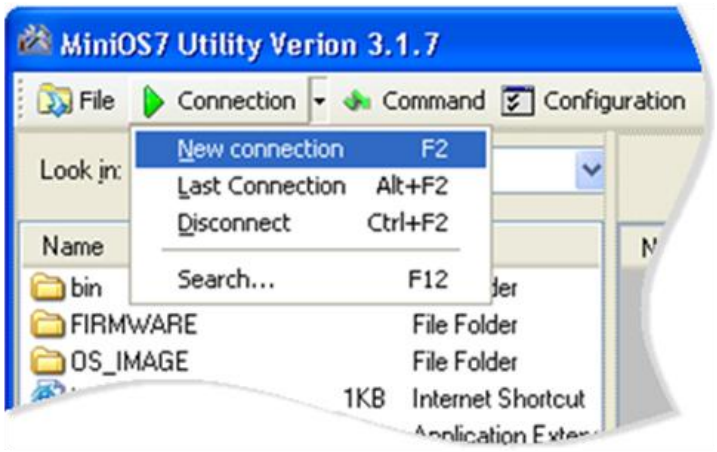
http://ftp.icpdas.com/pub/cd/8000cd/napdos/7000/756x/

Step 3: Check the port number that is assigned to the I-7560 converter

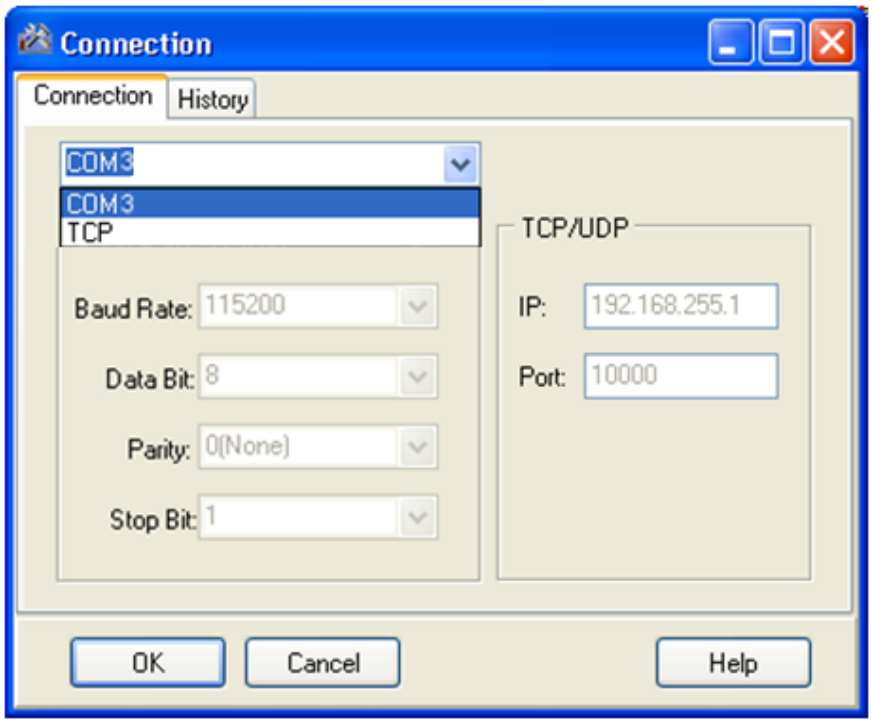


Step 4: Run the MiniOS7 Utility

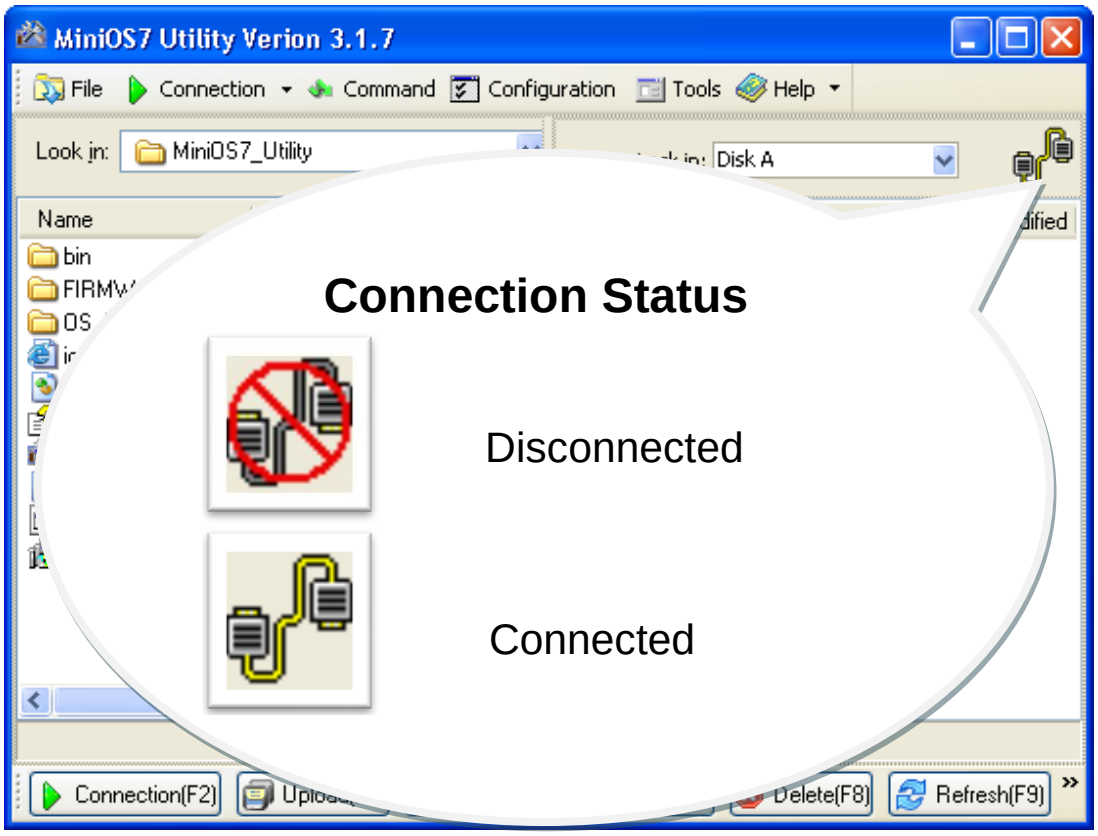
Step 5: Click the "New connection" from the "Connection" menu



Step 6: On the "Connection" dialog box, in the drop down list, select the COM port number that is assigned to I-7560 converter



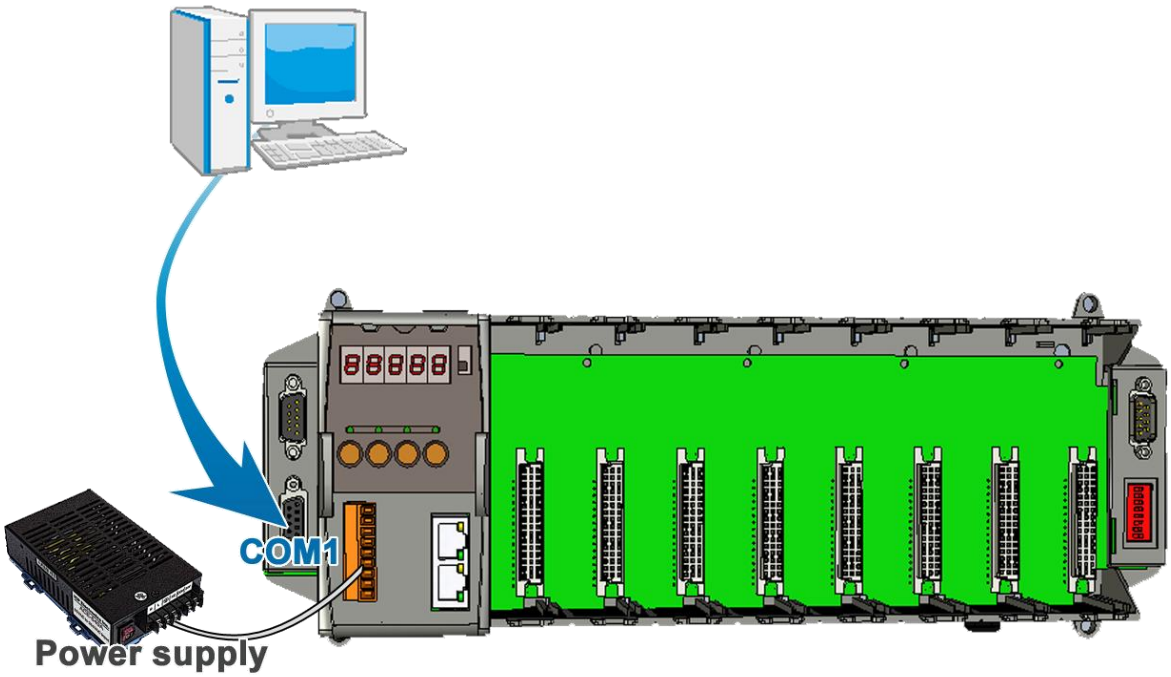
Step 7: The connection has already established



2.3.1.4. Steps to configure dual Ethernet using a COM1 connection

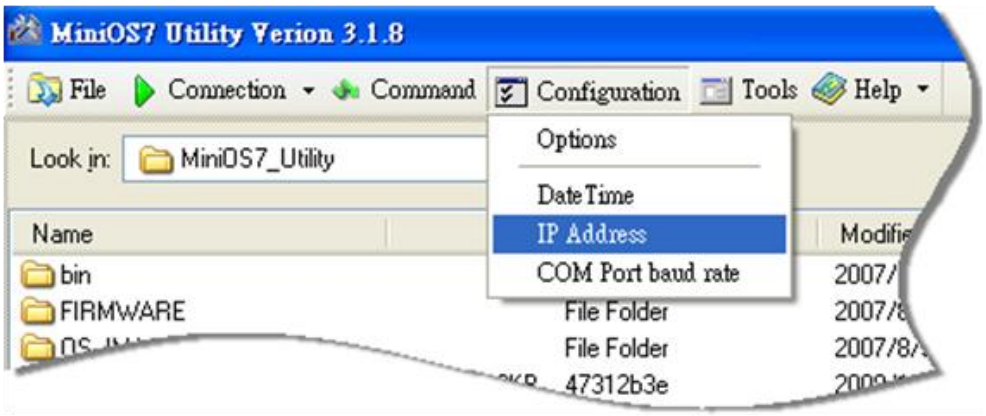
Step 1: Establish a connection using the COM1 connection

For more detailed description on how to establish a connection using the COM1 connection, please refer to section “2.3.1.1. Steps to use a COM1 connection”

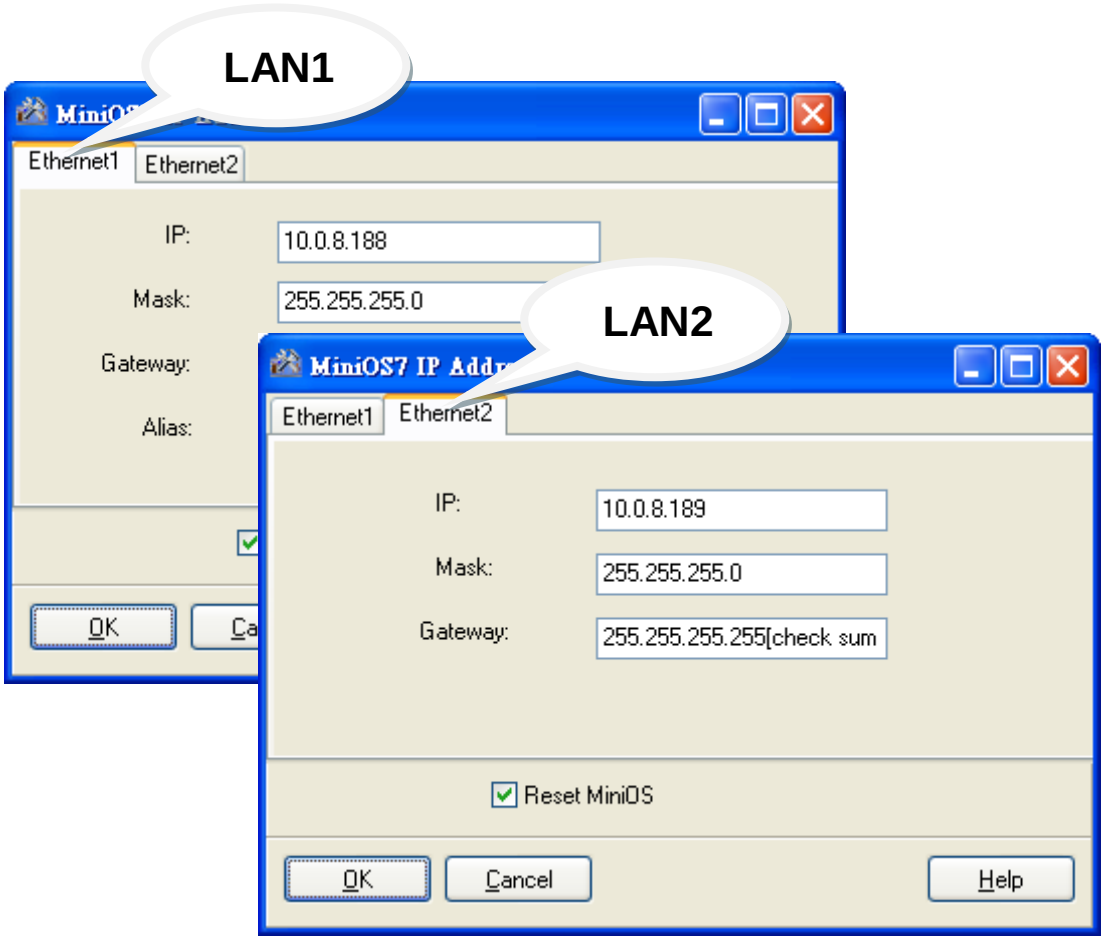


Step 2: Run the MiniOS7 Utility

Step 4: Click the "IP Address" from the "Configuration" menu

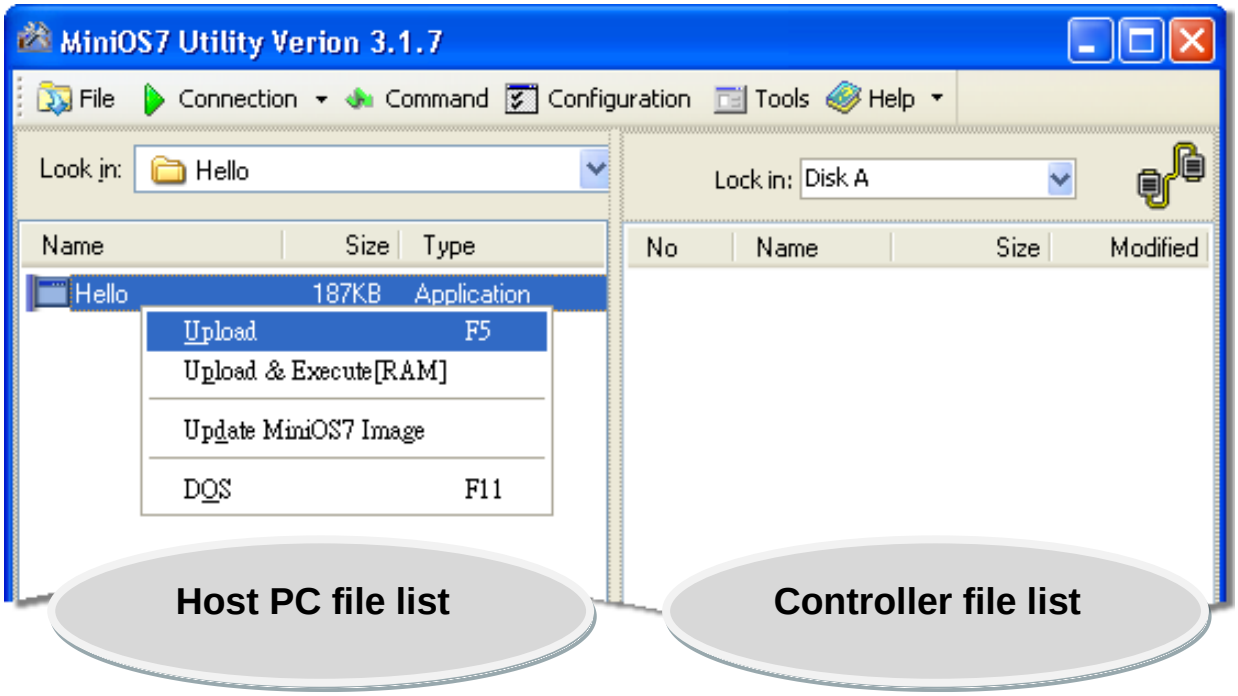


Step 5: On the "MiniOS7 IP Address" dialog box, select "Ethernet1" tab to configure LAN1, select "Ethernet2" tab to configure LAN2

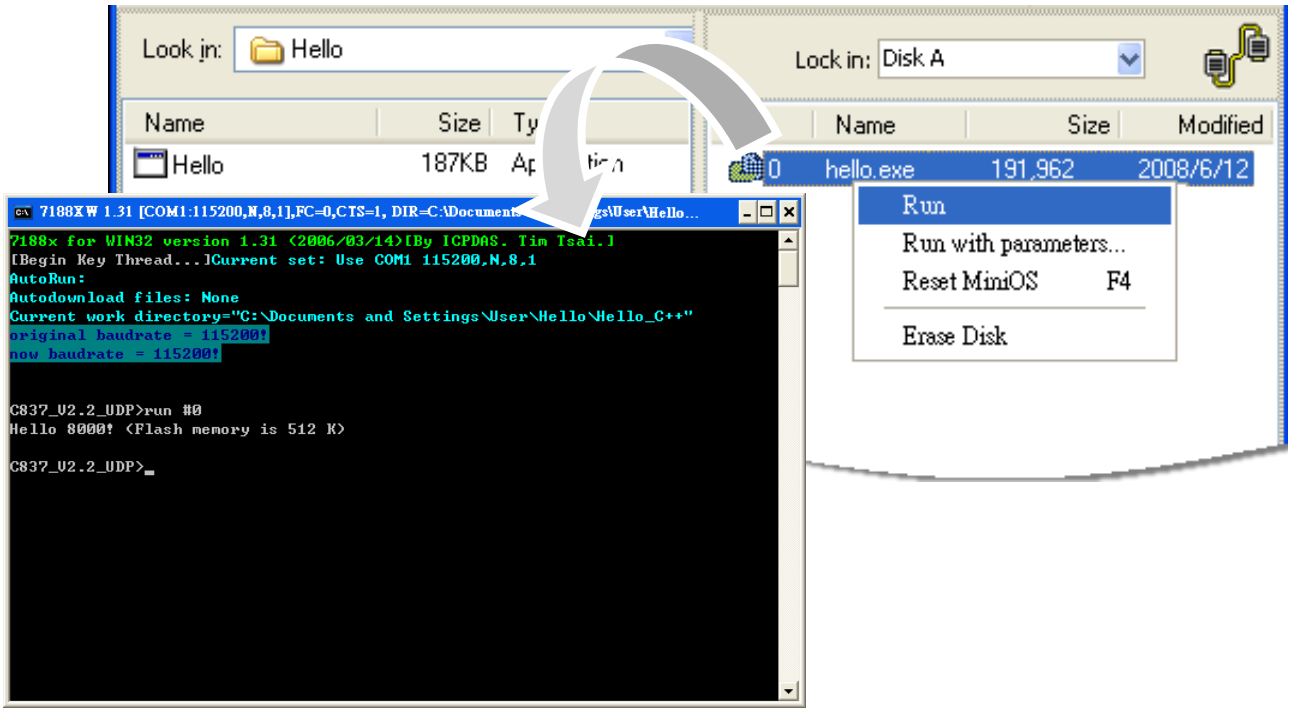


2.3.2. Uploading and executing programs on iPAC-8000

Step 1: On the host pc file list, Right click on the file name that you wish to upload and then select the “Upload” option



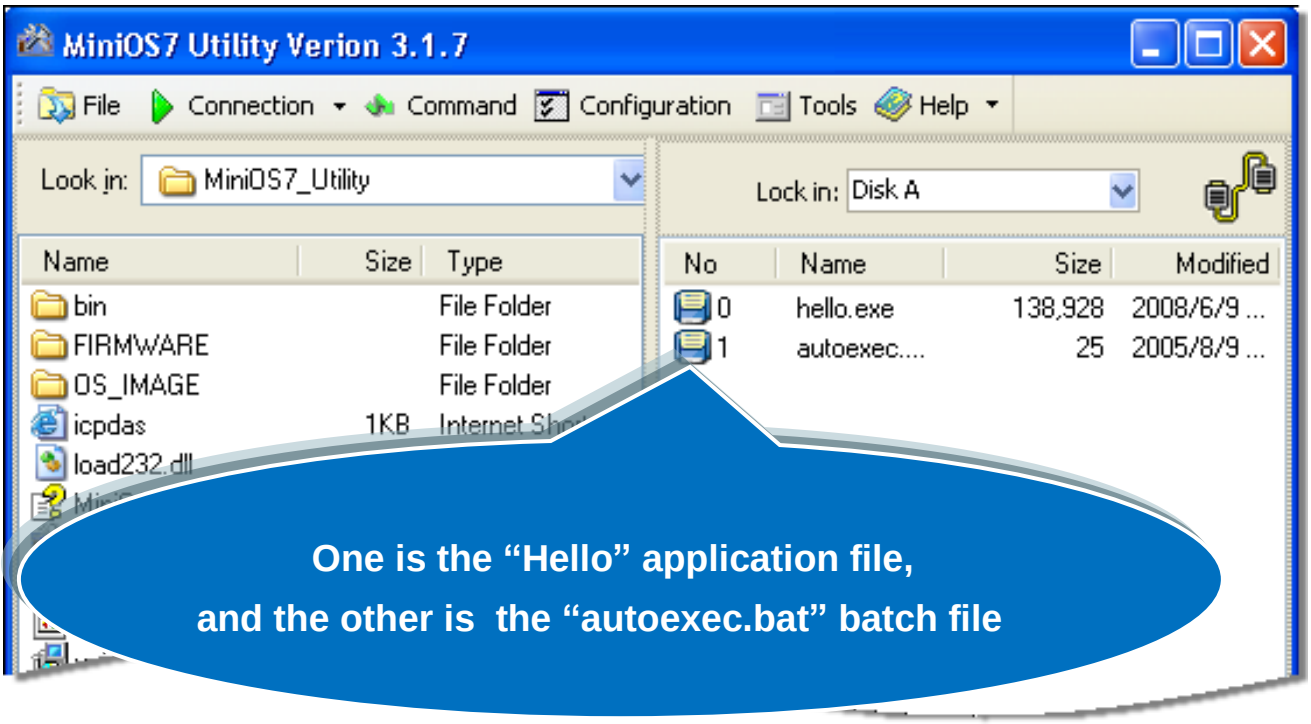
Step 2: On the controller file list, Right click on the file name that you wish to execute and then select the “Run” option

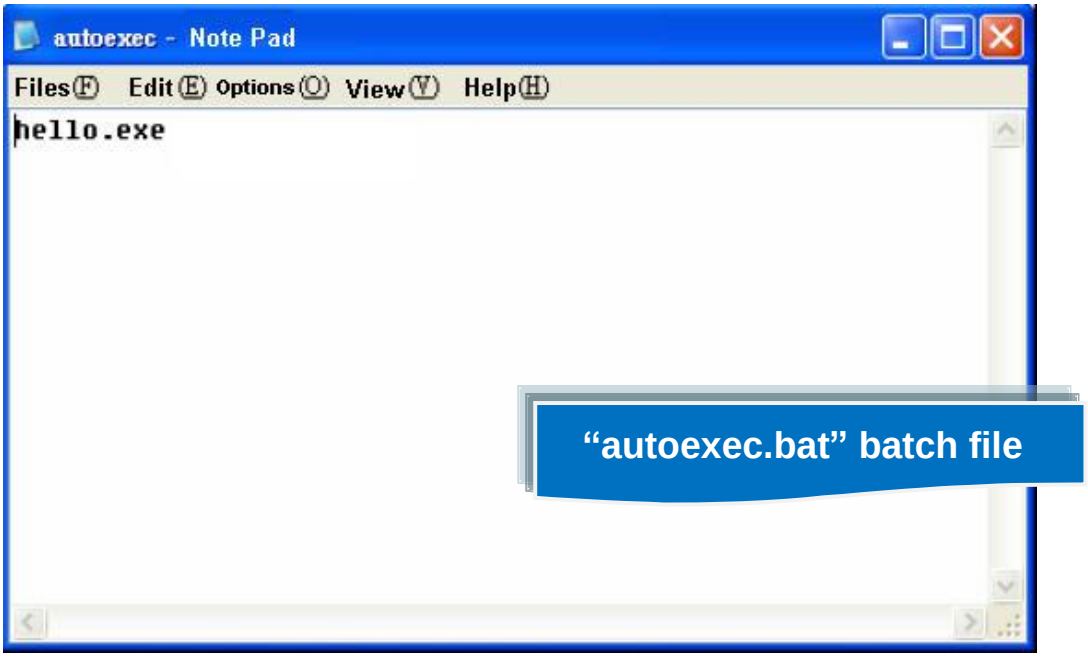


2.3.3. Making programs start automatically

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 on the iPAC-8000, the program will start automatically in the next start-up.

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





2.4. MiniOS7 Utility for updating OS image

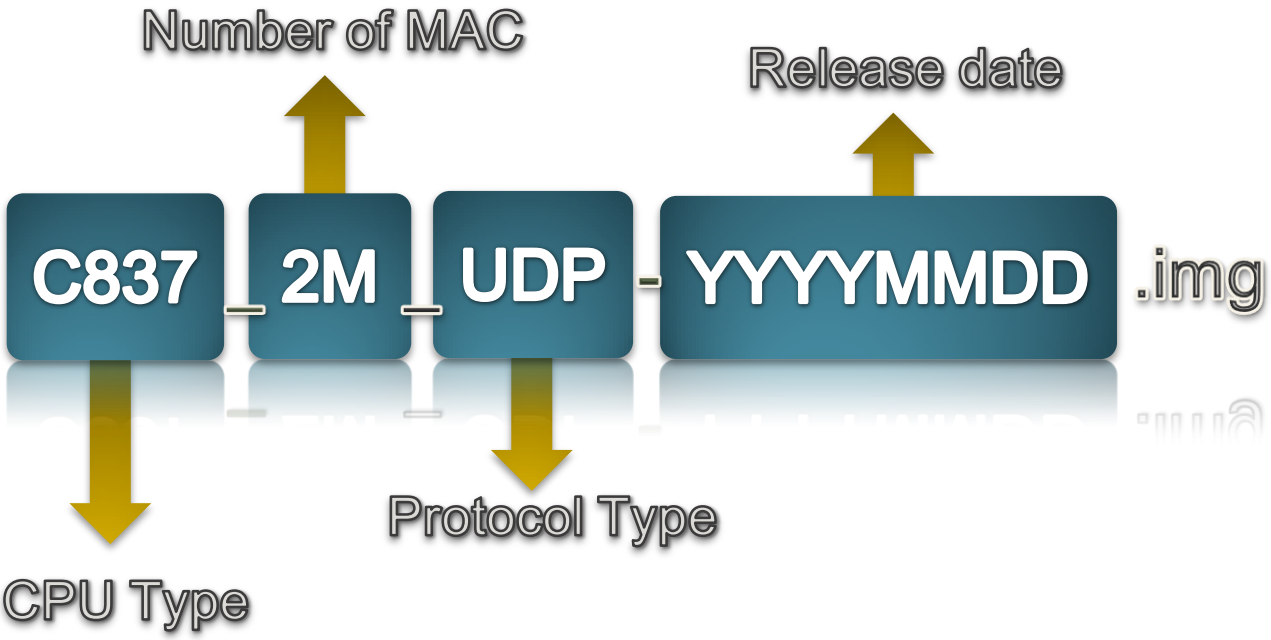
ICP DAS will continue to add additional features to MiniOS7 in the future, we advise you periodically check the ICP DAS web site for the latest update to MiniOS7.

Step 1: Get the latest version of the MiniOS7 OS image

The latest version of the MiniOS7 OS image can be obtain from:

CD:\ NAPDOS\iPAC8000\OS_Image

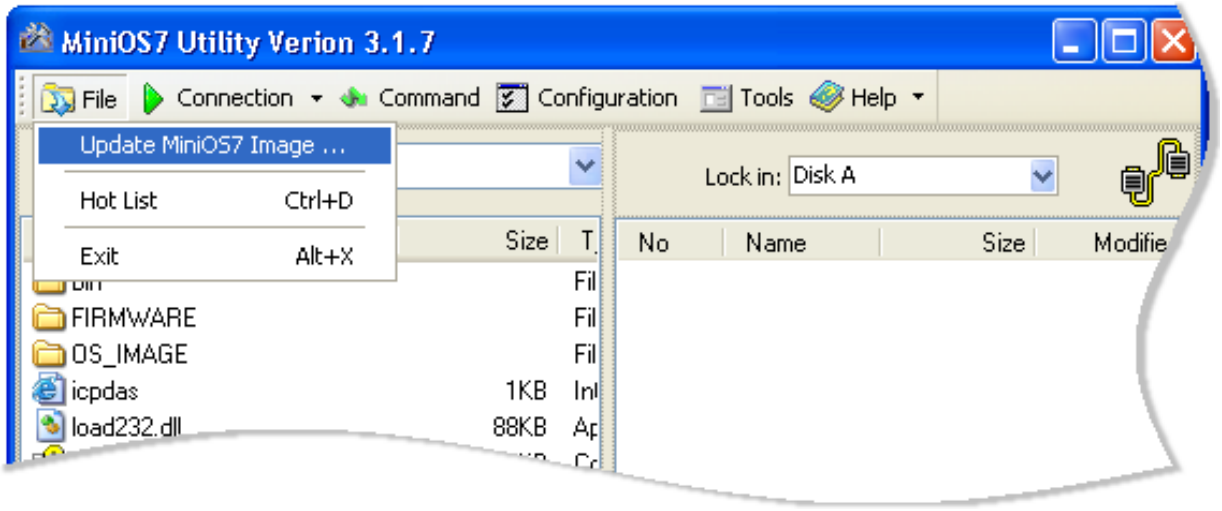
http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/os_image/



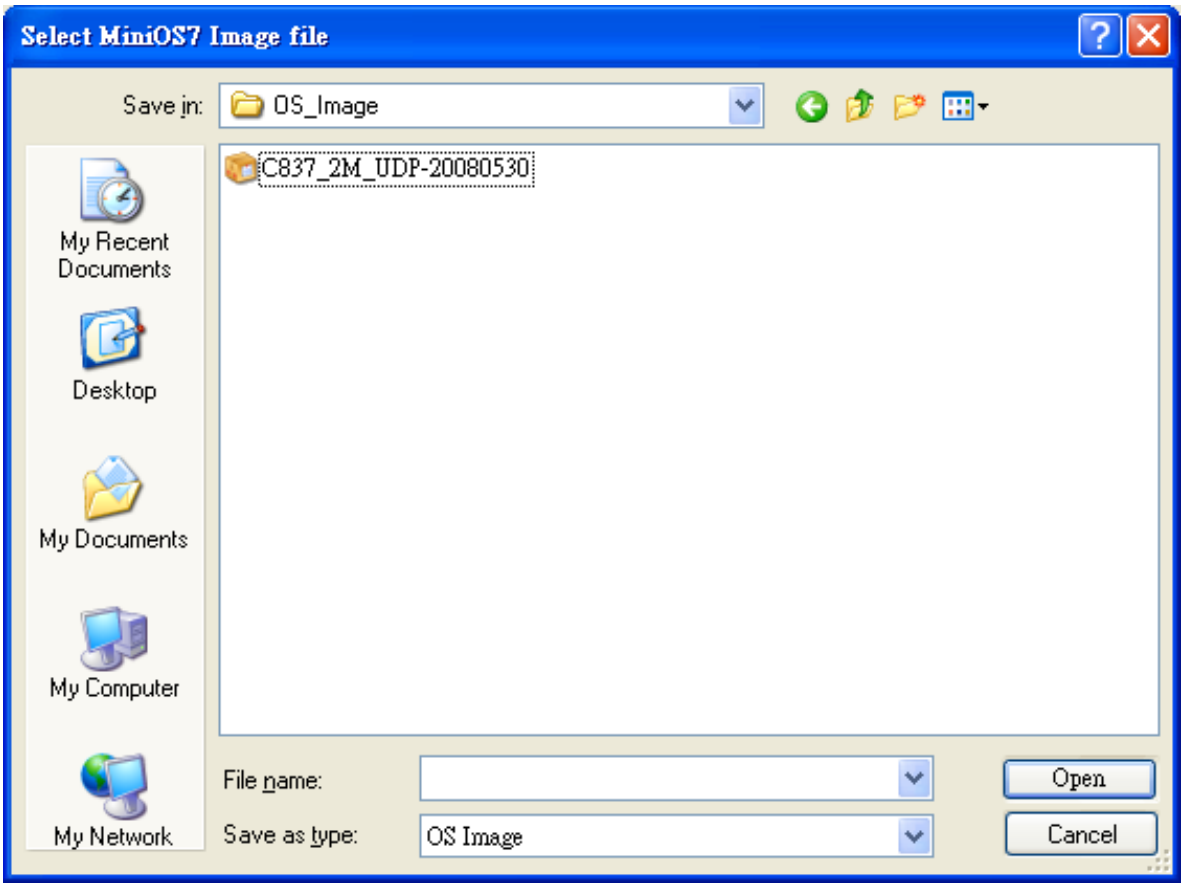
Step 2: Establish a connection

For more detailed information about this process, please refer to section "2.3.1. Establishing a connection".

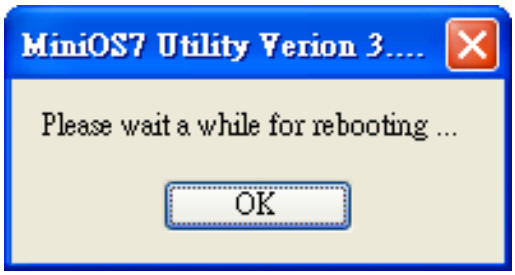
Step 3: Click on the "Update MiniOS7 Image …" from the "File" menu



Step 4: Select the latest version of the MiniOS7 OS image



Step 5: Click on the “Update MiniOS7 Image …” from the “File” menu



Step 6: Click on the “Info” buttion to check OS image version

3. Your First Program on iPAC-8000

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

3.1. Setting up the compiler

The following compilers are available for iPAC-8000.

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

ICP DAS suggests that the Borland C++ version compiler is used as the libraries provided on the companion CD have been created using this compiler.

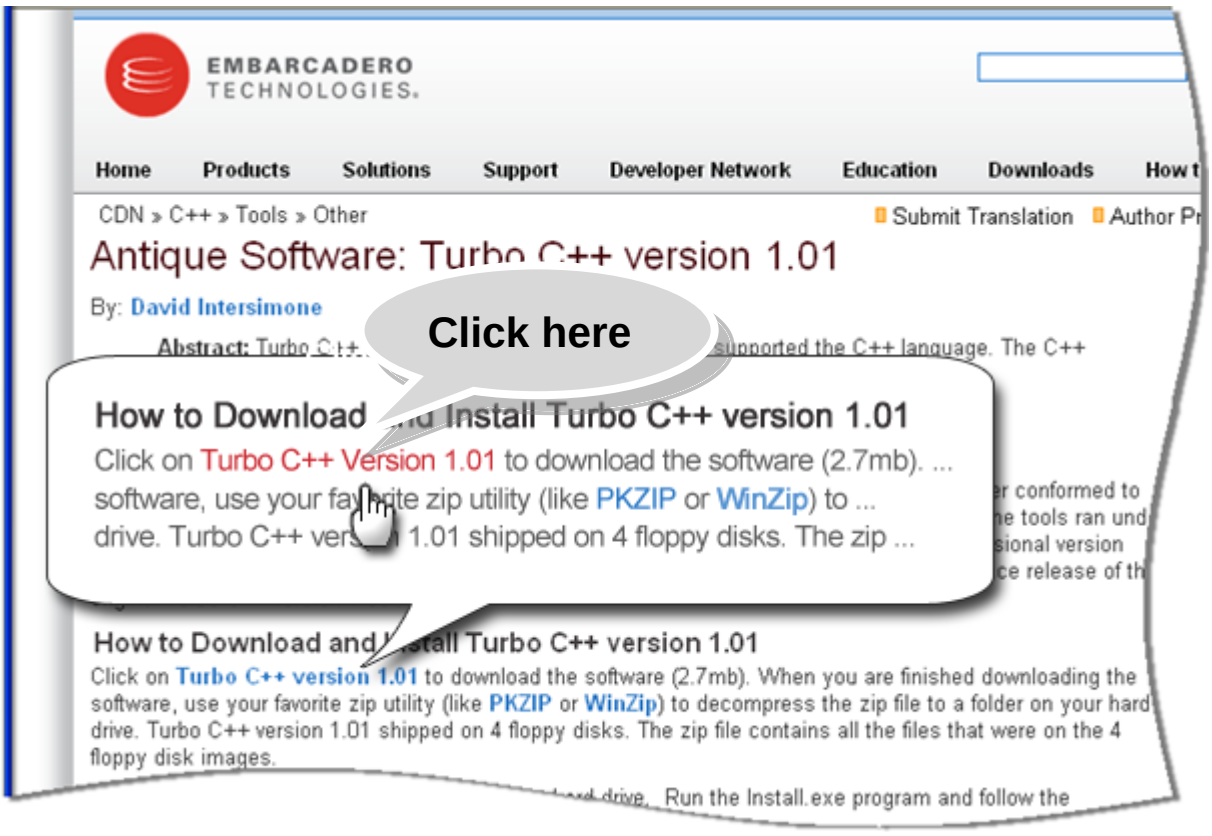
Special attention should be paid to the following items before using the compiler to develop custom applications:

- Generate a standard DOS executable program
- Set the CPU option to 80188/80186
- Set the floating point option to EMULATION if floating point computation is required. (Be sure not to choose 8087)
- Cancel the Debug Information function as this helps to reduce program size. (MiniOS7 supports this feature.).

3.1.1. Installing the Compiler

If there is no compiler currently installed on your system, installation of the compiler should be the first step. The following section guides you to install Turbo C++ Version 1.01 on your system.

Step 1: Go to the Borland web site and download Turbo C++ version 1.01



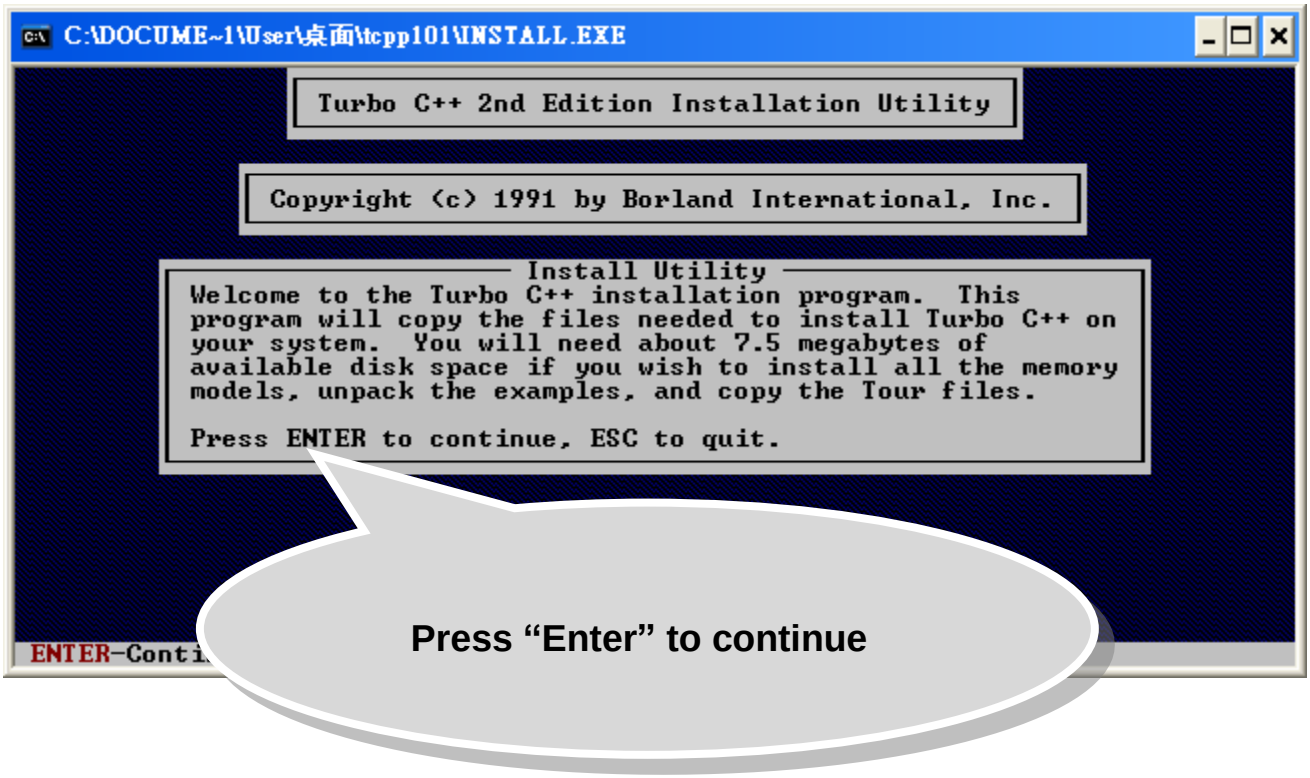
Free versions of the Turbo C++ version 1.01 and Turbo version 2.01 Compilers can be downloaded from the Borland web site.

- Turbo C++ version 1.01
  http://dn.codegear.com/article/21751
- Turbo C version 2.01
  http://dn.codegear.com/article/20841

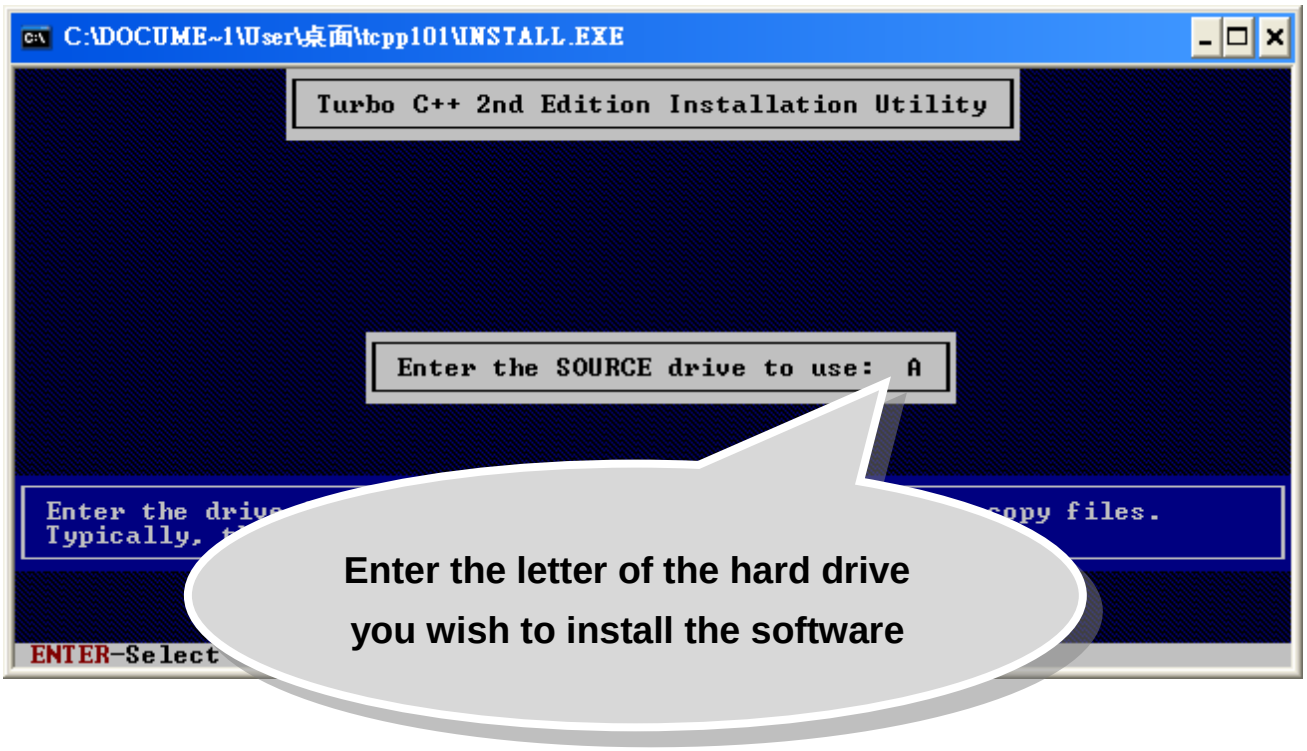
Step 2: Unzip the downloaded zip file to the temporary folder

Step 3: Double click the executable file to start setup wizard

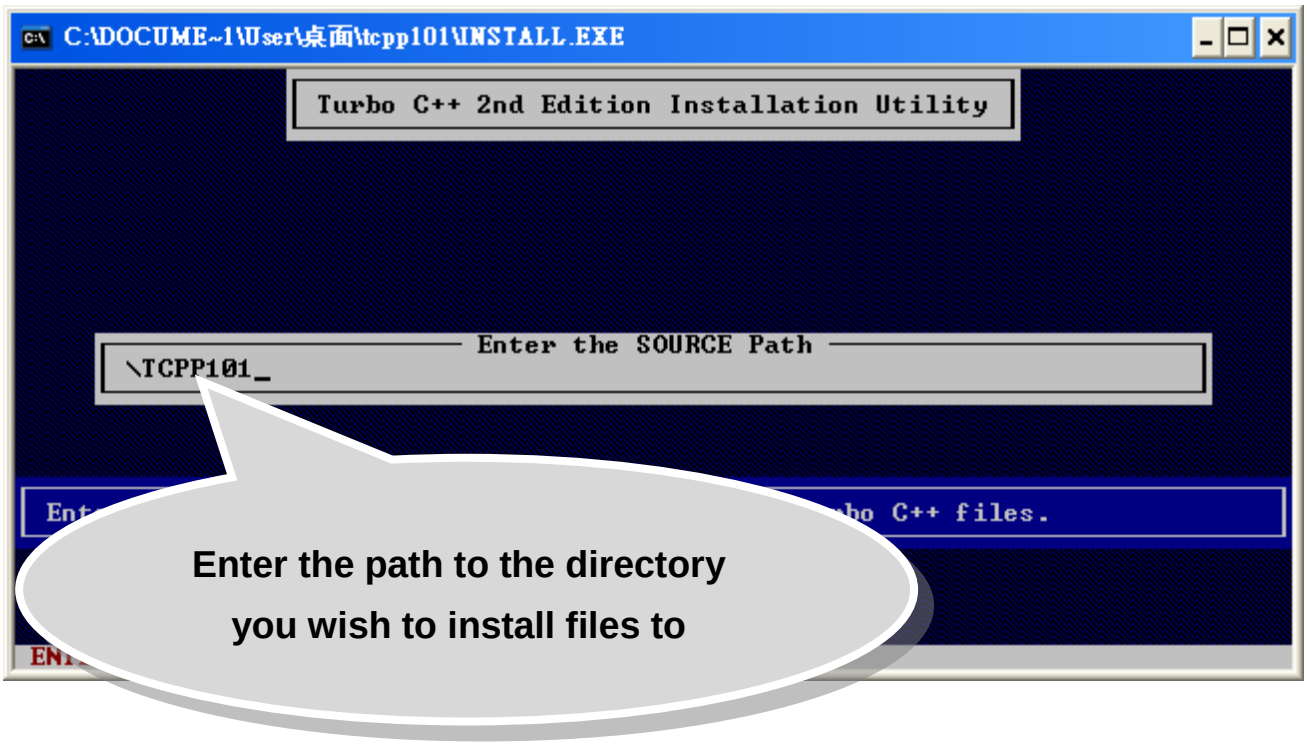
Step 4: Press “Enter” to continue



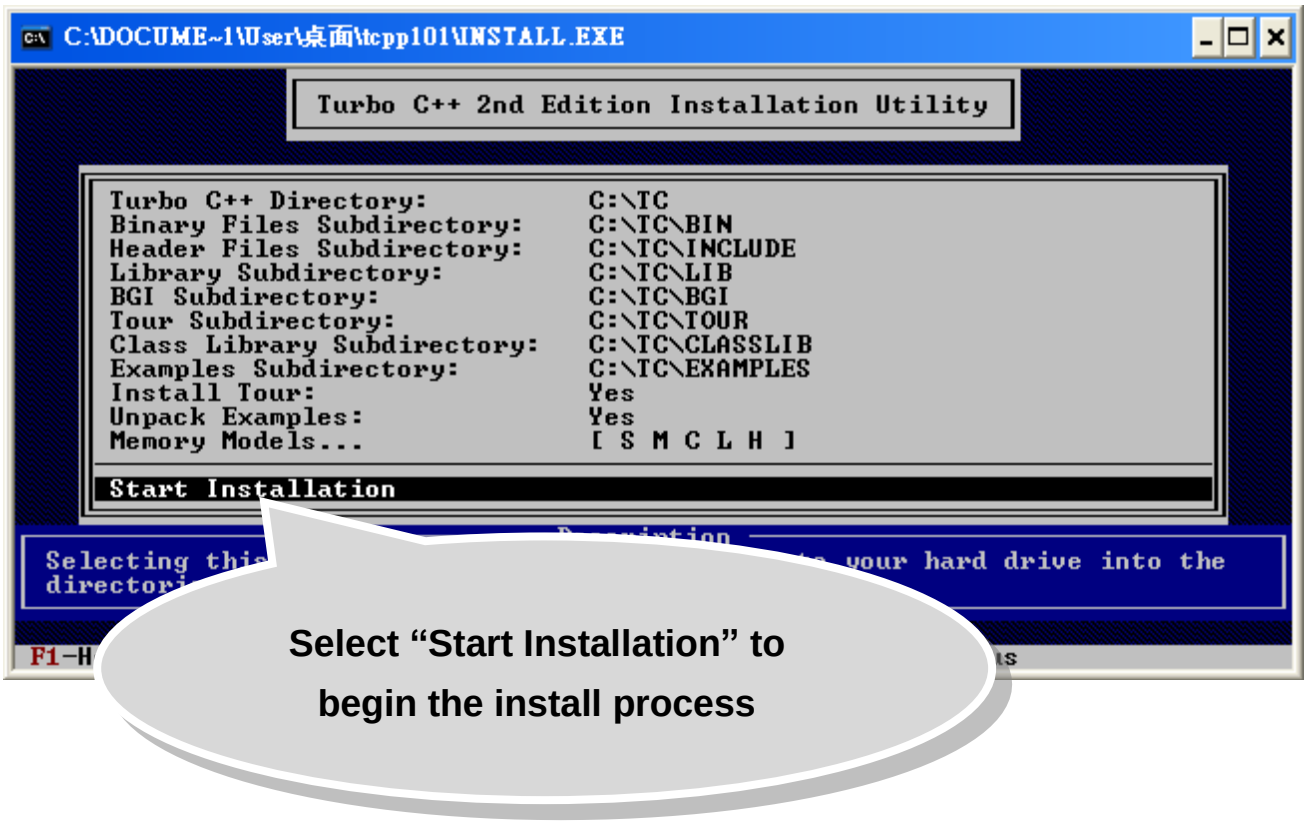
Step 5: Enter the letter of the hard drive you wish to install the software



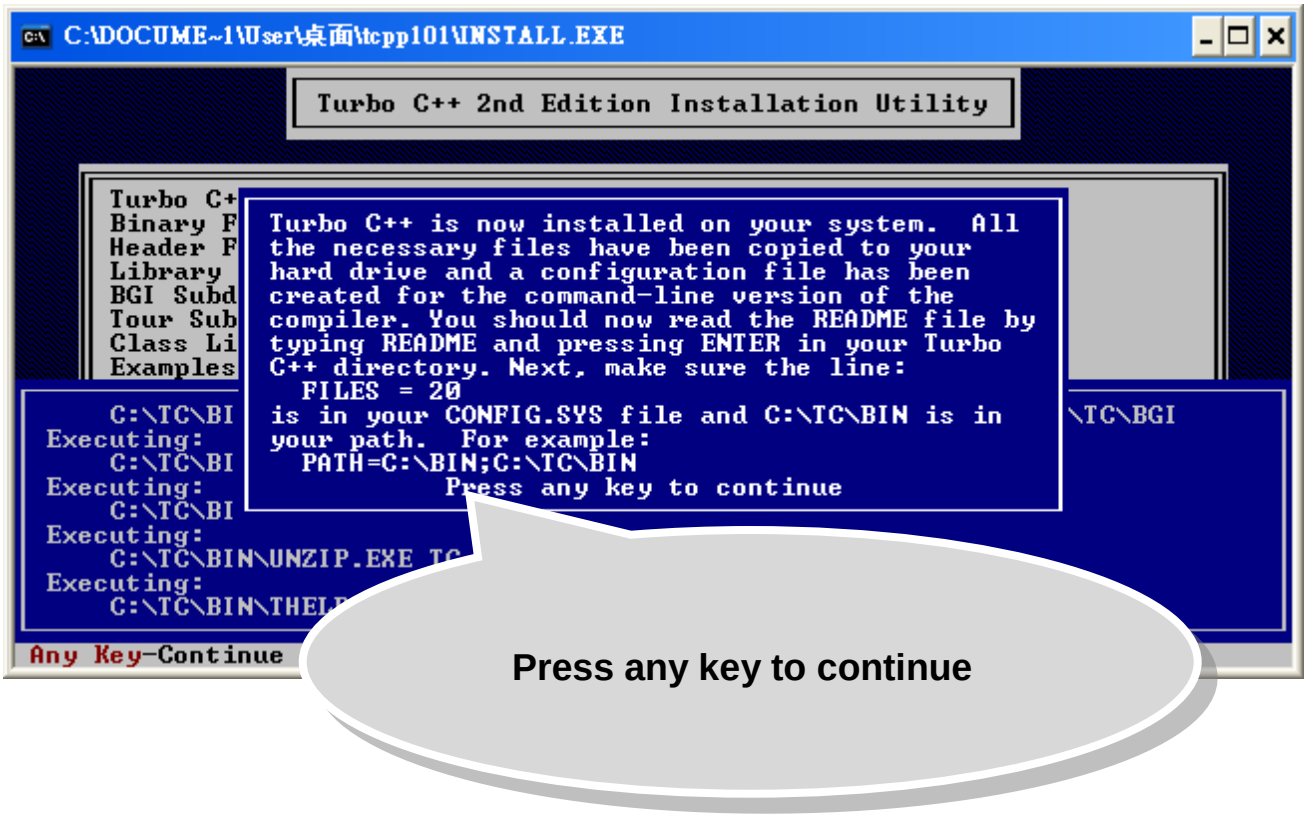
Step 6: Enter the path to the directory you wish to install files to



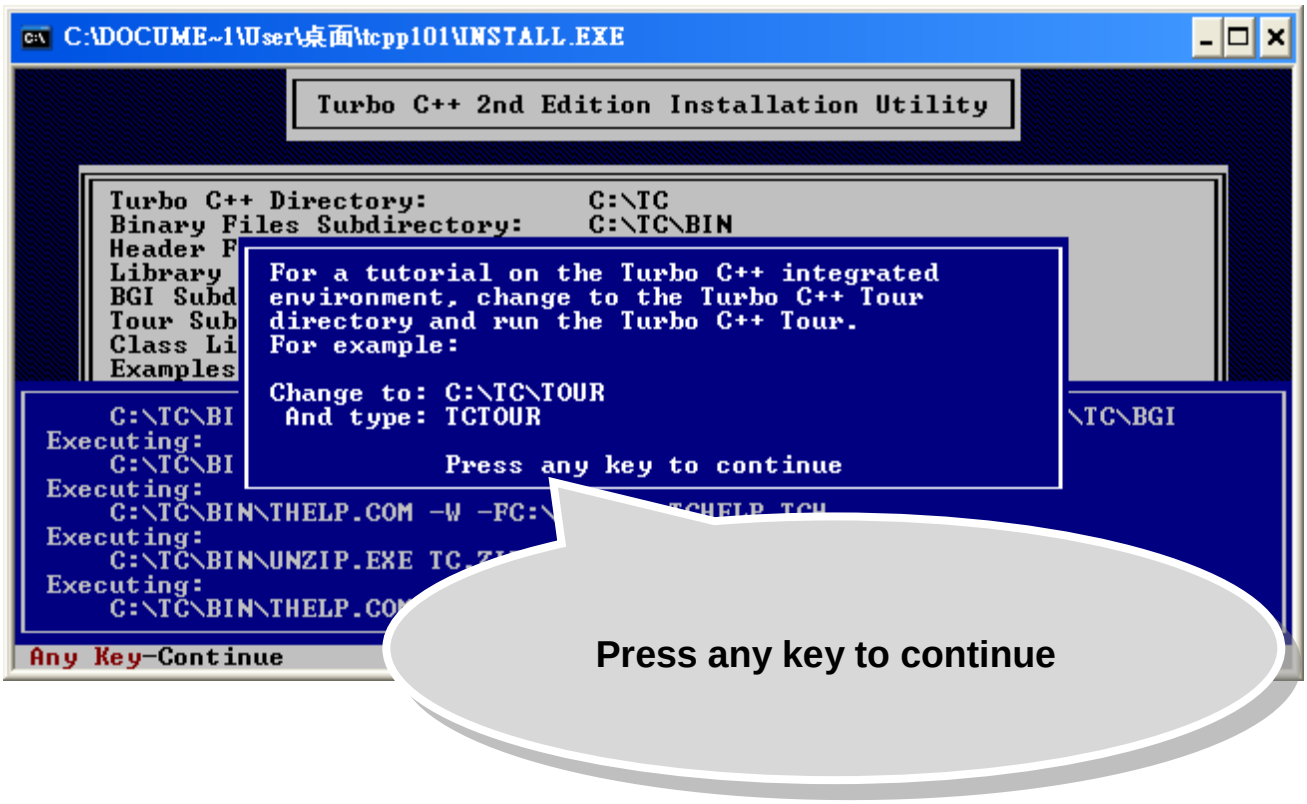
Step 7: Select “Start Installation” to begin the install process



Step 8: Press any key to continue



Step 9: Press any key to continue

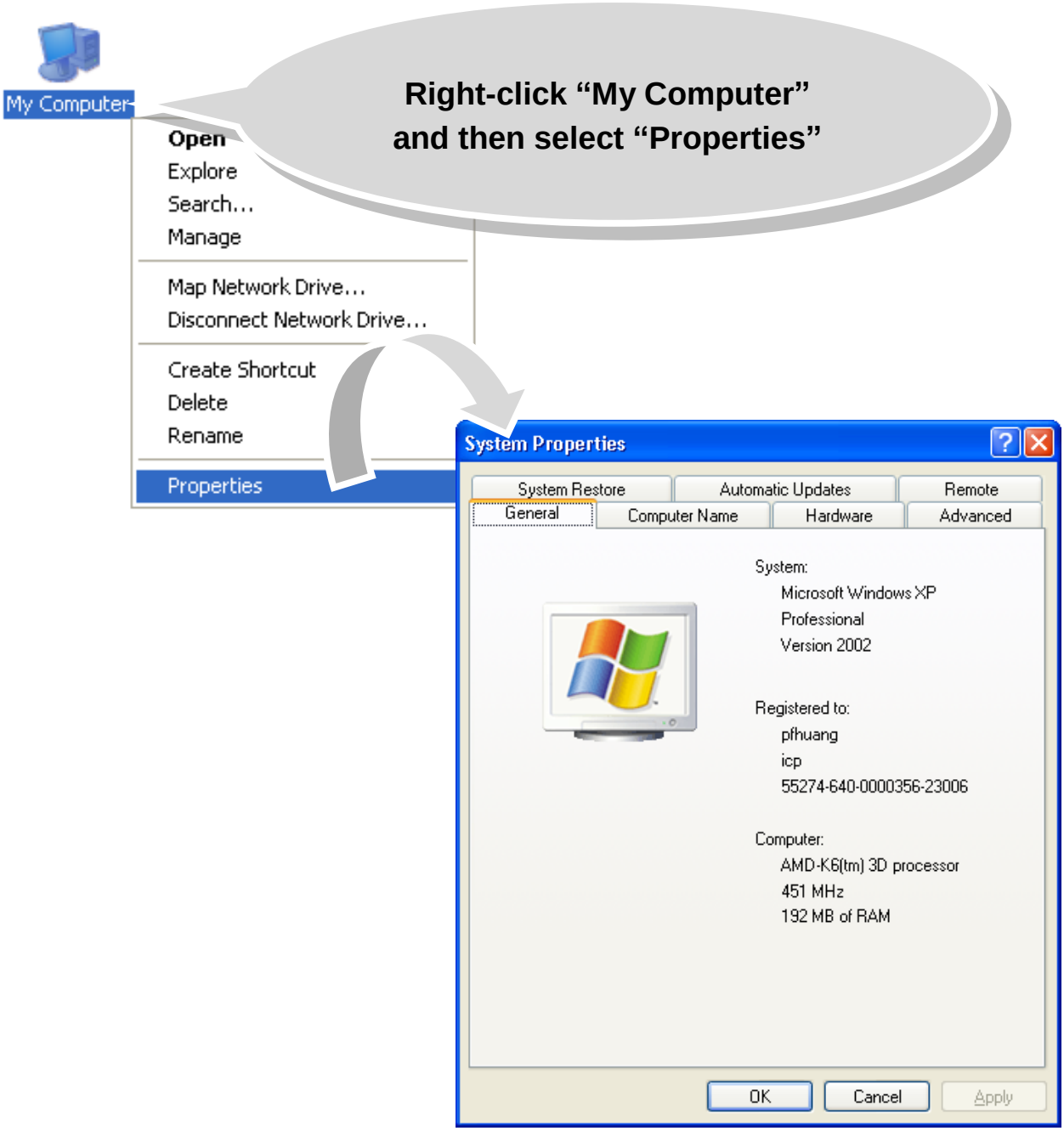


Step 10: Installation is complete

3.1.2. Setting up the environment variables

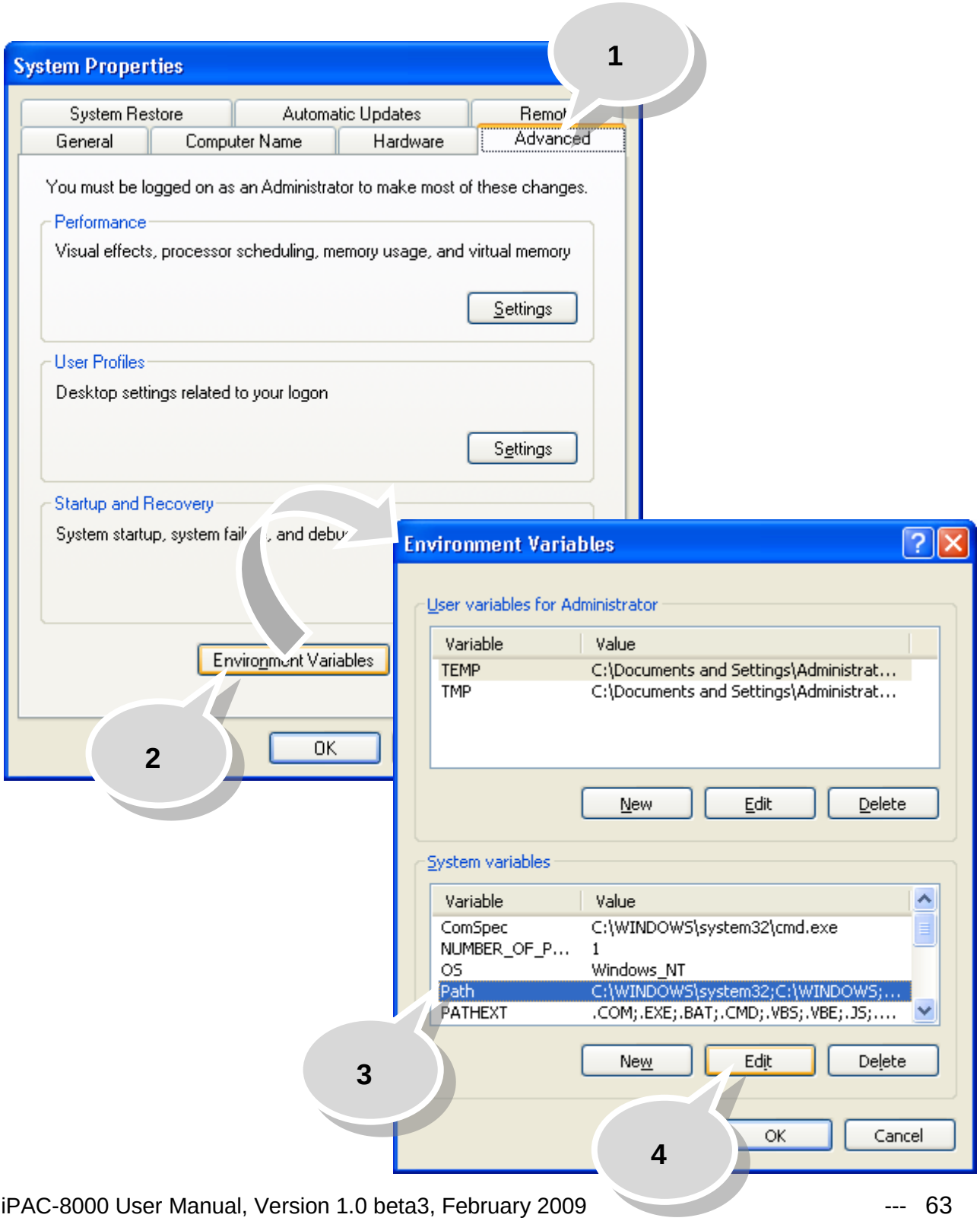
After installing the compiler, several compilers will be available from the Windows Command line. You can set the path environment variable so that you can execute This compiler on the command line by entering simple names, rather than by using Their full path names.

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



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

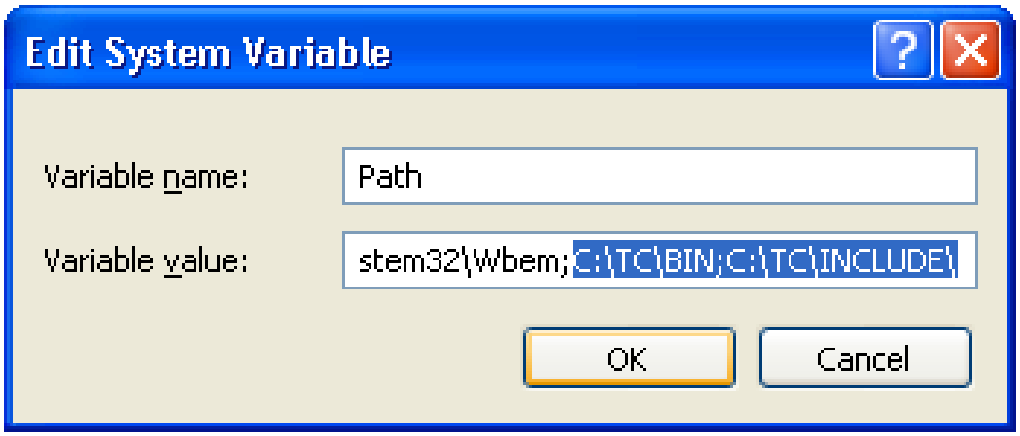
Step 3: On the “Environment Variables” dialog box, click the “Edit” button located in the “System variables” option



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

A semi-colon is used as the separator between variable values.

For example, ”;c:\TC\BIN\;c:\TC\INCLUDE\”



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

3.2. API for iPAC-8000

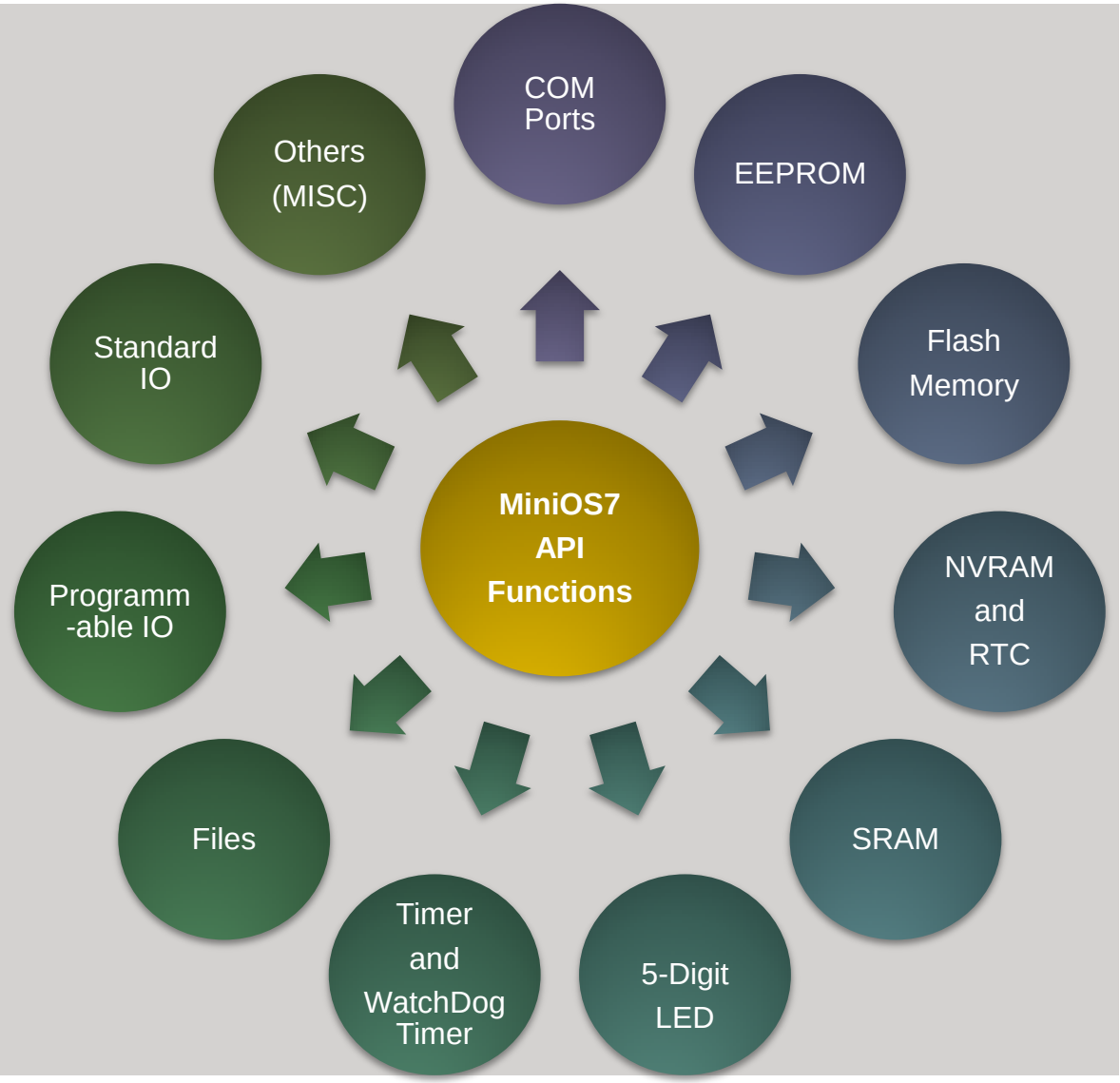
To develop a custom program, ensure that the files below are installed the Host PC. If they are not installed, refer to “section **2.2. Software Installation**”.

- **Functions Library ─ 8000a.lib**

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

- **Header File ─ 8000a.h**

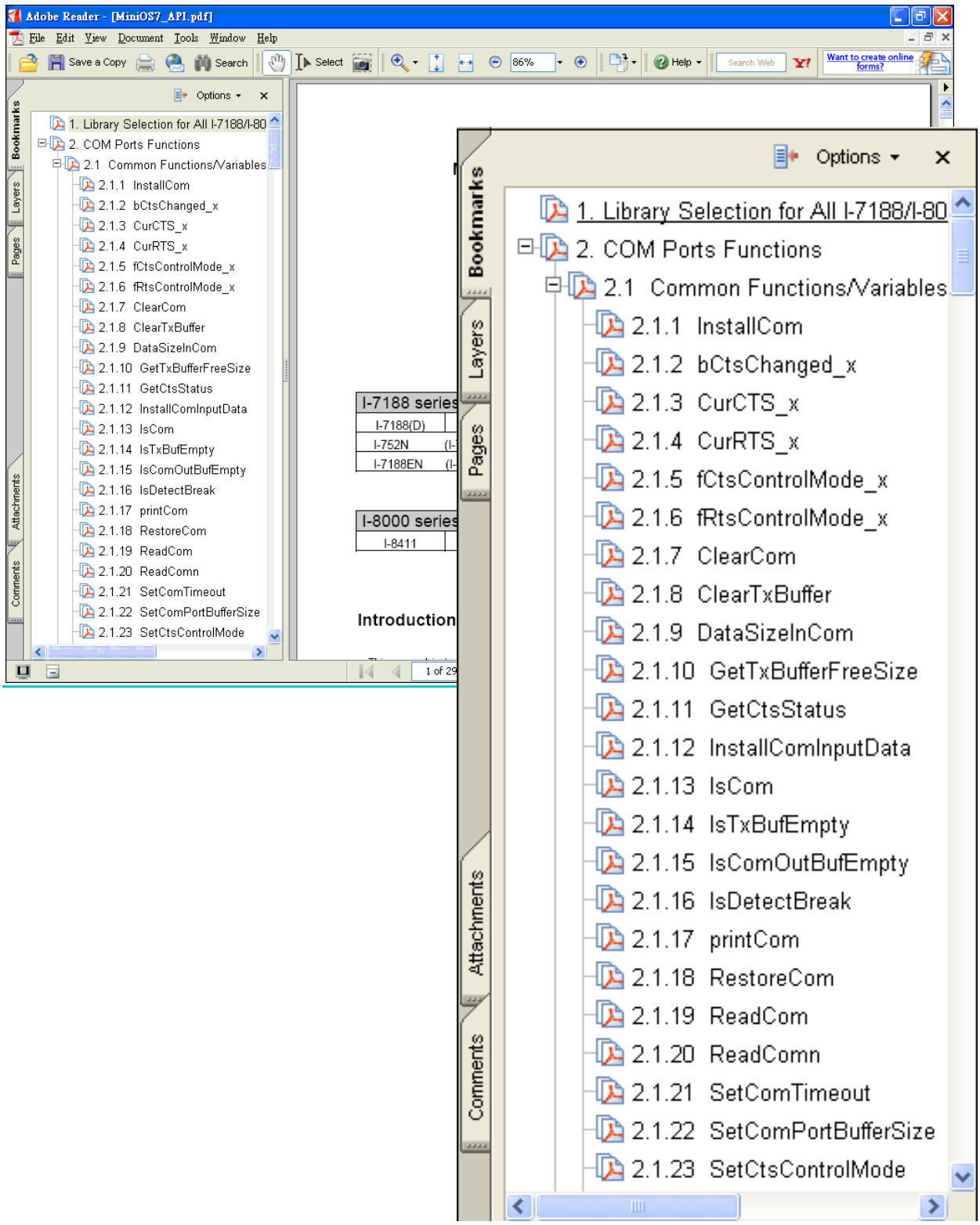
  This file contains the forward declarations of subroutines, variables, and other identifiers used for the MiniOS7 API.



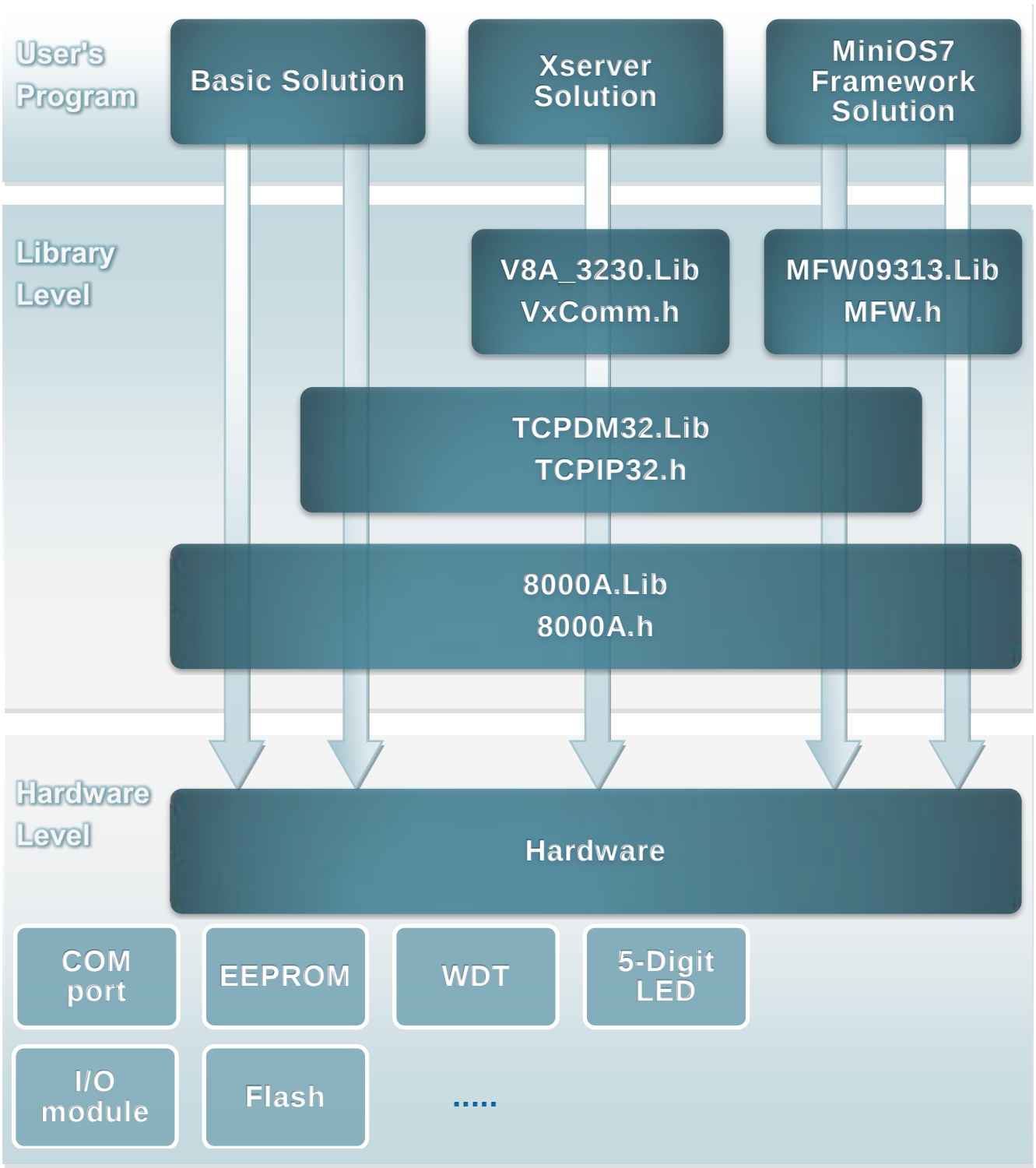
For full usage information regarding the description, prototype and the arguments of the functions, please refer to the “MiniOS7 API Functions User Manual” located at:

CD:\Napdos\MiniOS7\Document

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



- **System Structure**



3.3. Build and run your first program

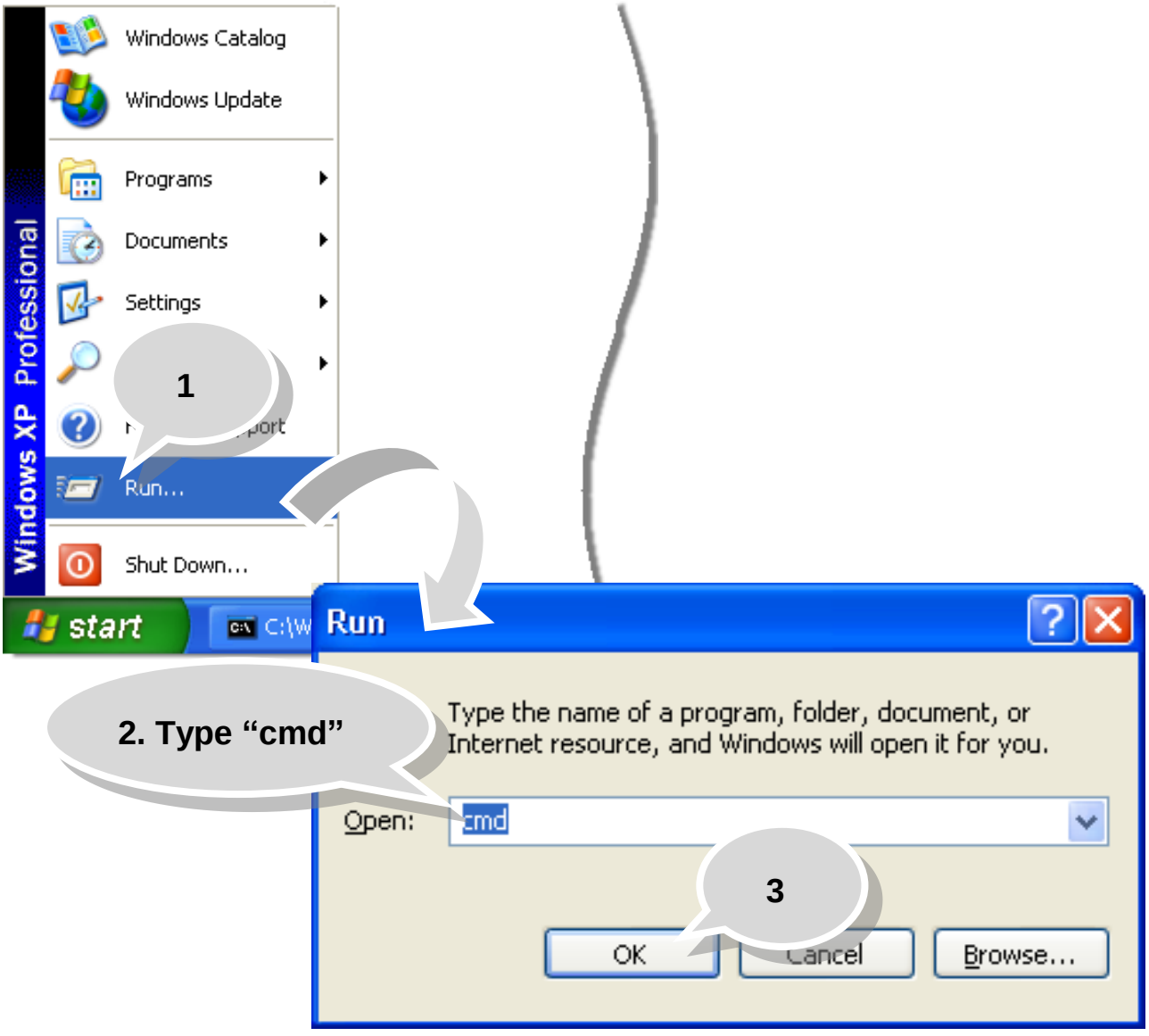
If you don't using the TC++ (Turbo C++) to write a program, please take the following steps.

Step 1: Open a MS-DOS command prompt

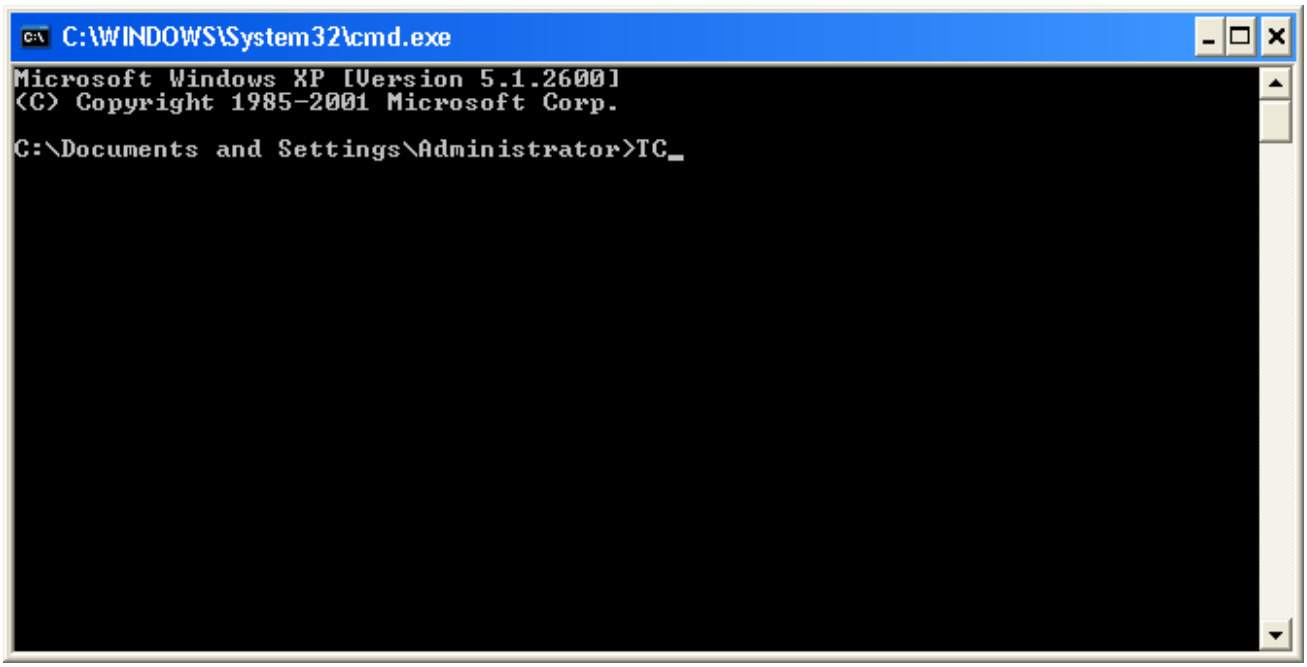
i. Select "Run" from the "Start" menu

ii. On the "Run" dialog box, type "cmd"

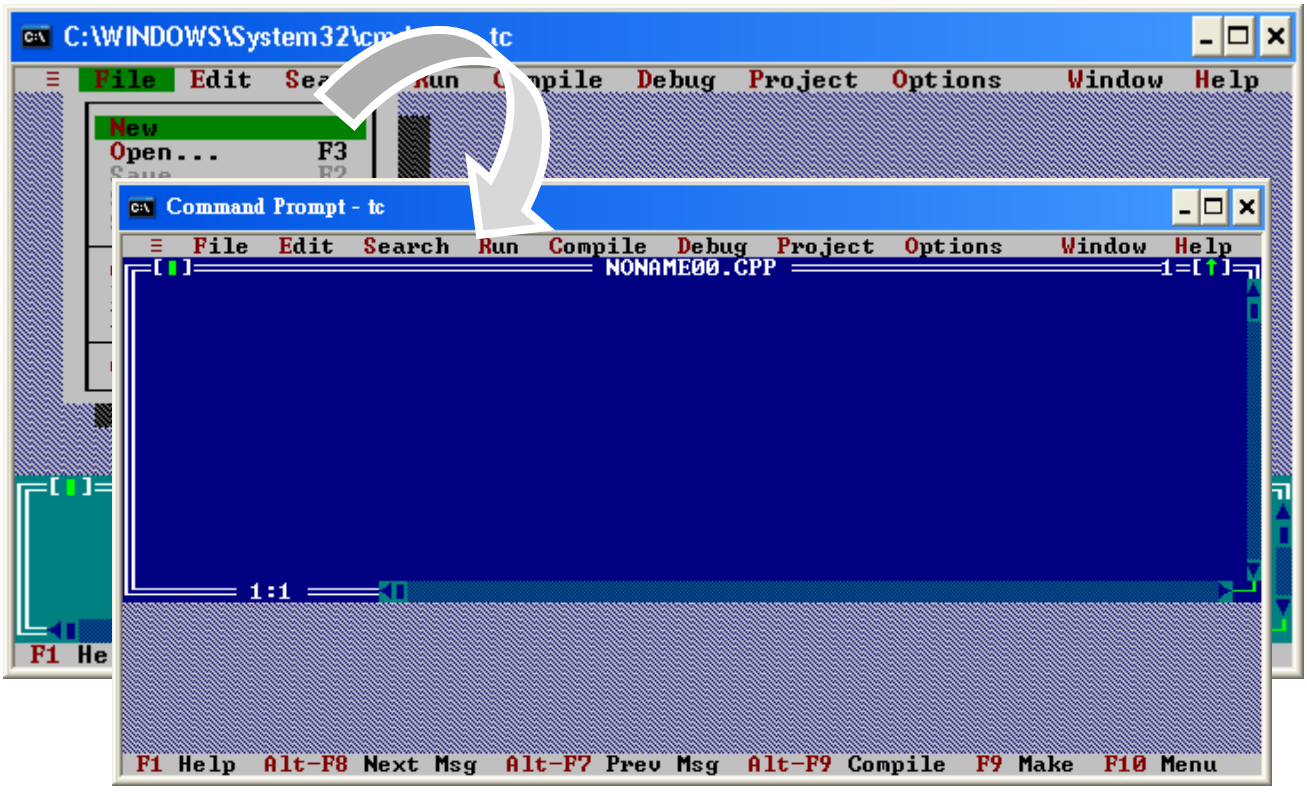
iii. click the "OK" button



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



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



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

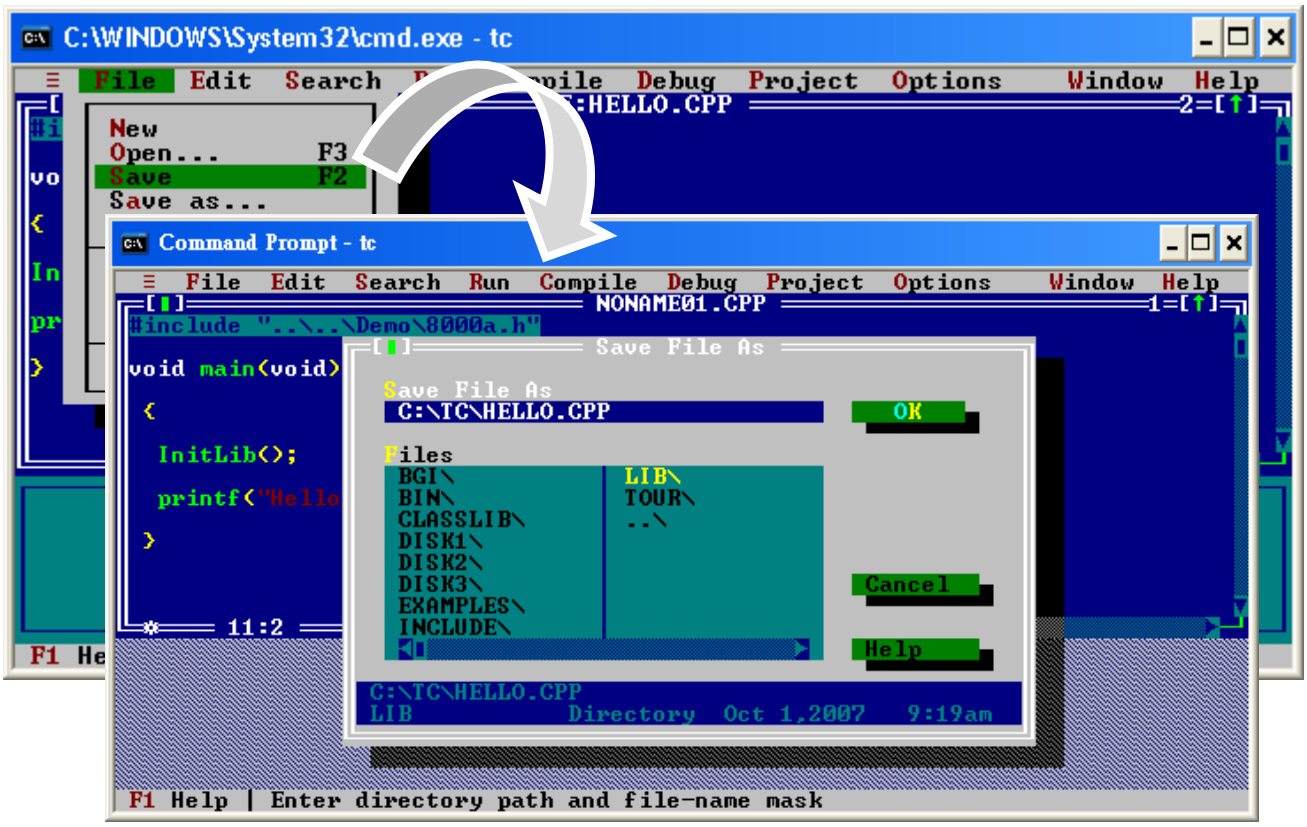
```
#include “8000A.h”
/* Include the header file that allows 8000a.lib functions to be used */
void main(void)
{
InitLib();   /* Initiate the 8000a library */

Print(“Hello world!\r\n”);   /* Print the message on the screen */
 }
```

Step 5: Save the source file

i. Select “Save” from the “File” menu

ii. Type the file name “Hello”

iii. Select “OK”



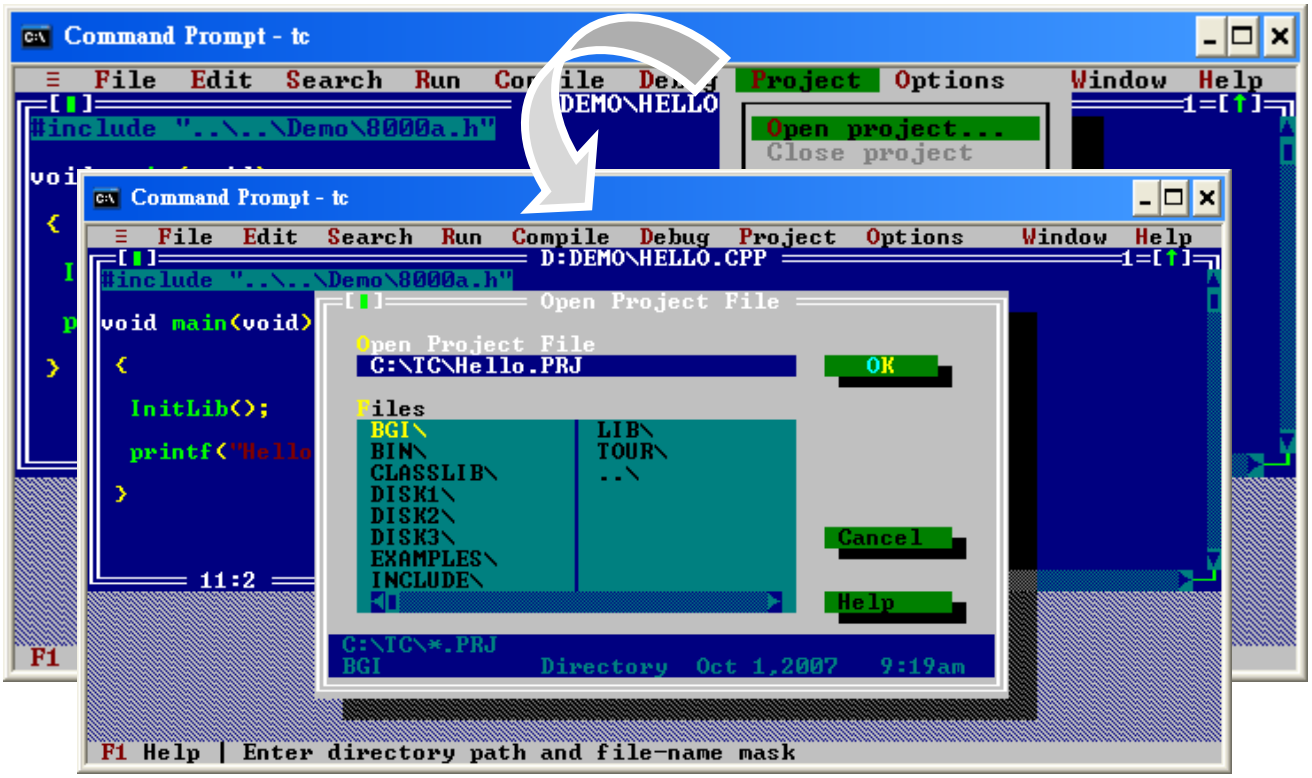
If there is a text editor you are familiar with or prefer to use such as Notepad or edit, you may use it to write the code shown above. It should be noted that a word processor application cannot be used for this purpose, as the application must save the file as plain text. C language program files should always have a ".C" extension name.

Step 6: Create a project (*prj)

i. Select "Open project…" from the "Project" menu

ii. Type the project name "Hello"

iii. Select "OK"



Step 7: Add the necessary source files to the project (*.CPP)

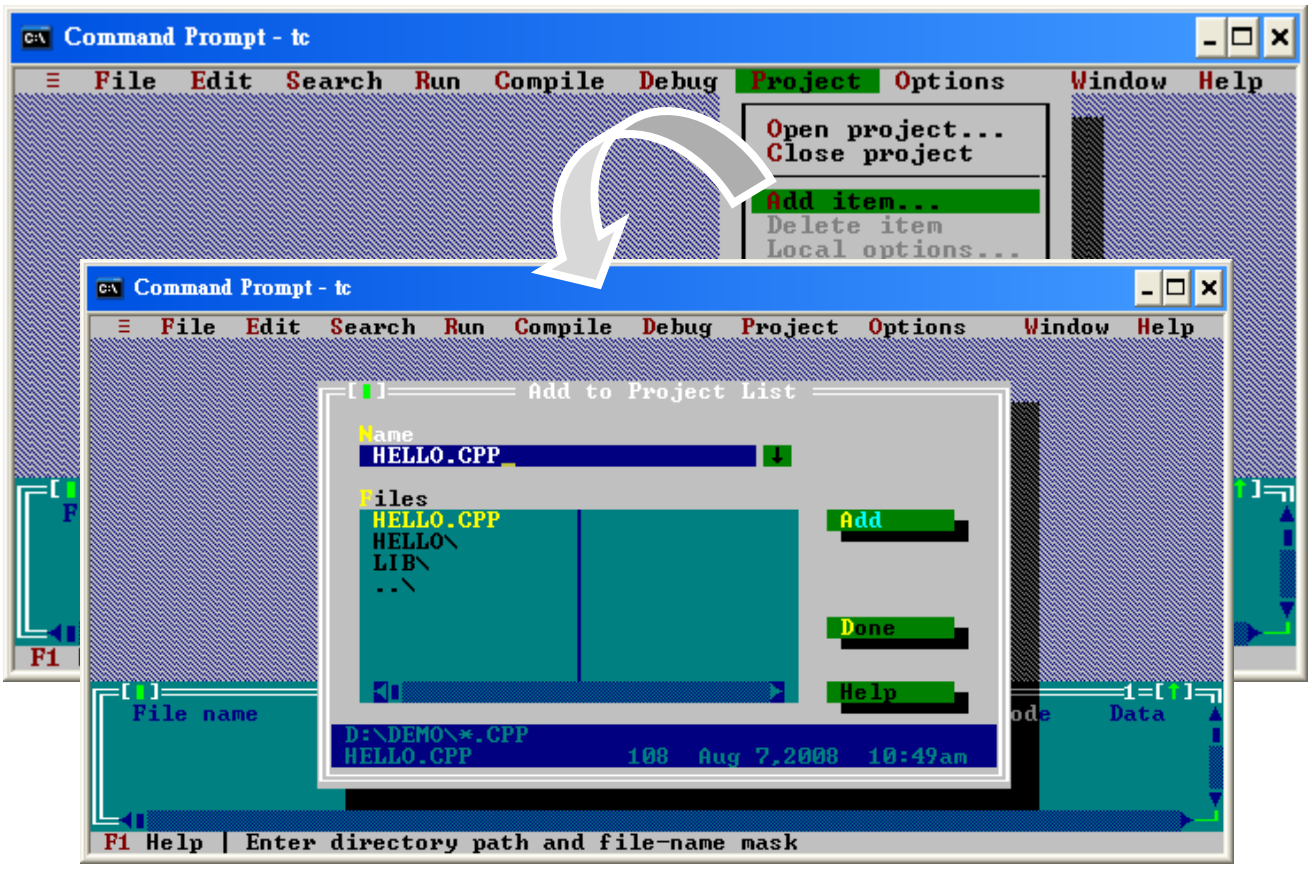
i. Select “Add item…” from the “Project” menu

ii. Type “ *.CPP ” to display a list of all available source files

iii. Choose the source files you require

iv. Select “Add”

v. Select “Done” to exit



Step 8: Add the necessary function libraries to the project (*.lib)

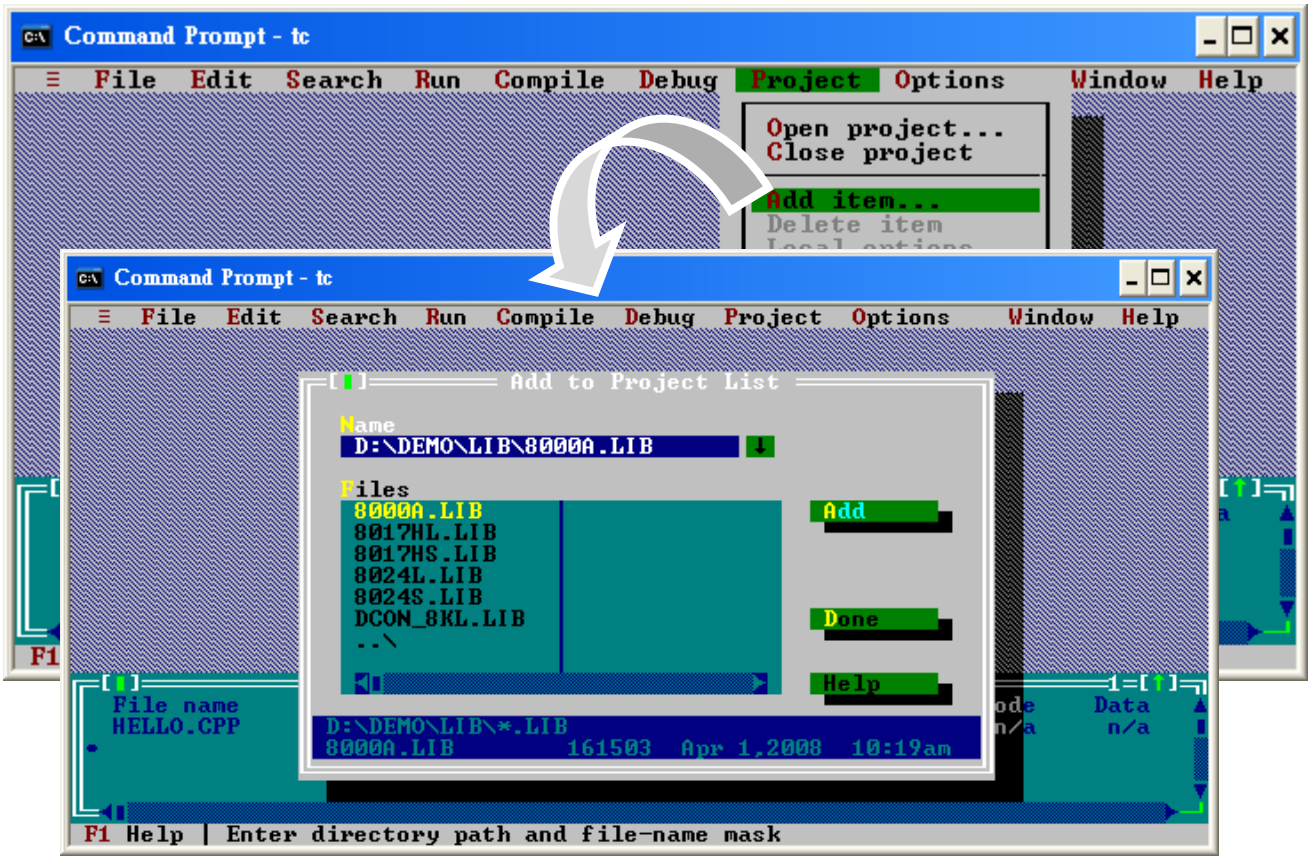
i. Select “Add item…” from the “Project” menu

ii. Type “ *.LIB ” to display a list of all available function libraries

iii. Choose the function libraries you require

iv. Select “Add”

v. Select “Done” to exit

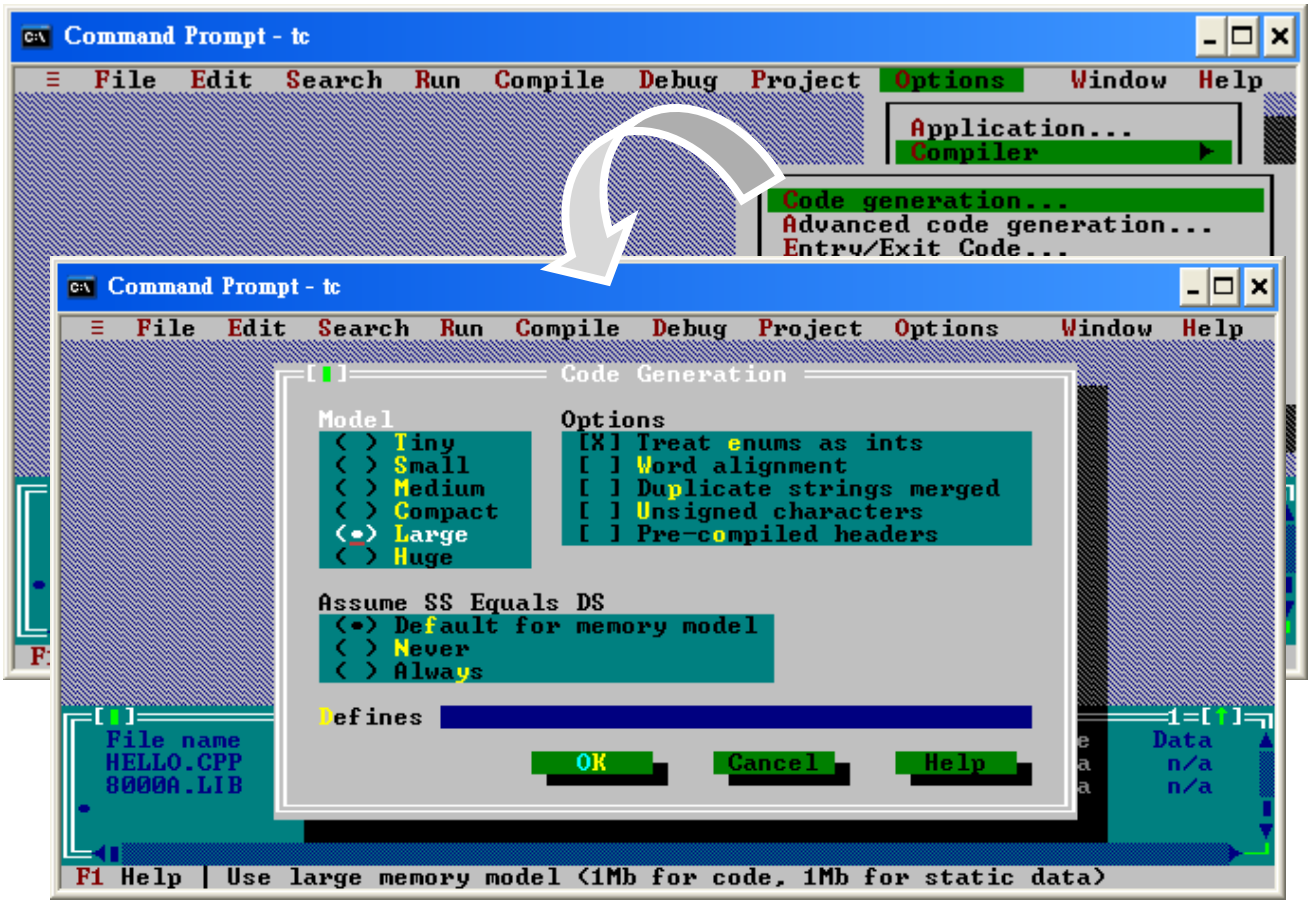


Step 9: Set the memory model to large

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

ii. On “Model” option, select “Large”

iii. Select “OK”



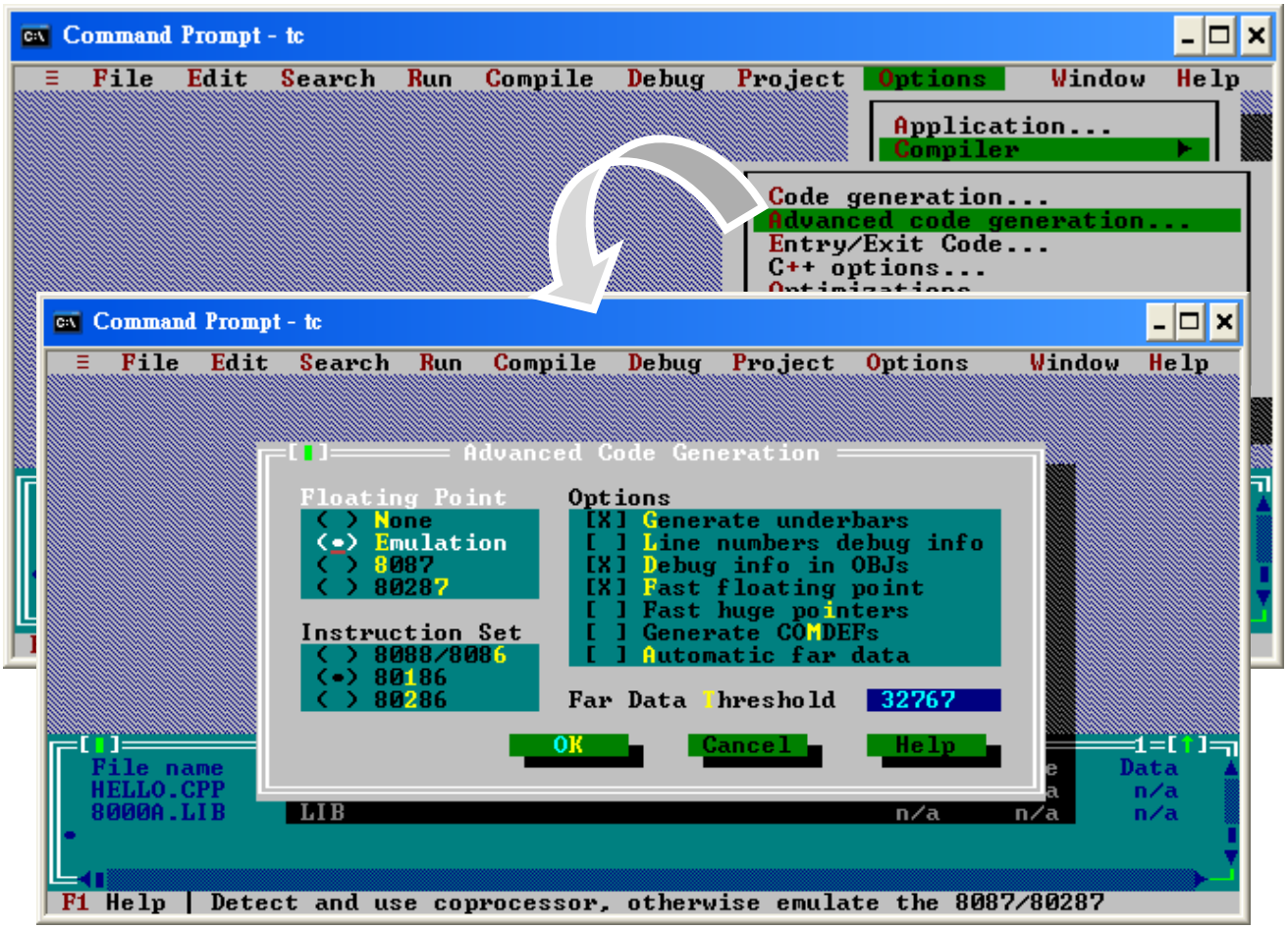
Step 10: Set the “Floating Point” to “Emulation” and “Instruction Set” to “80186”

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

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

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

iv. Select “OK”



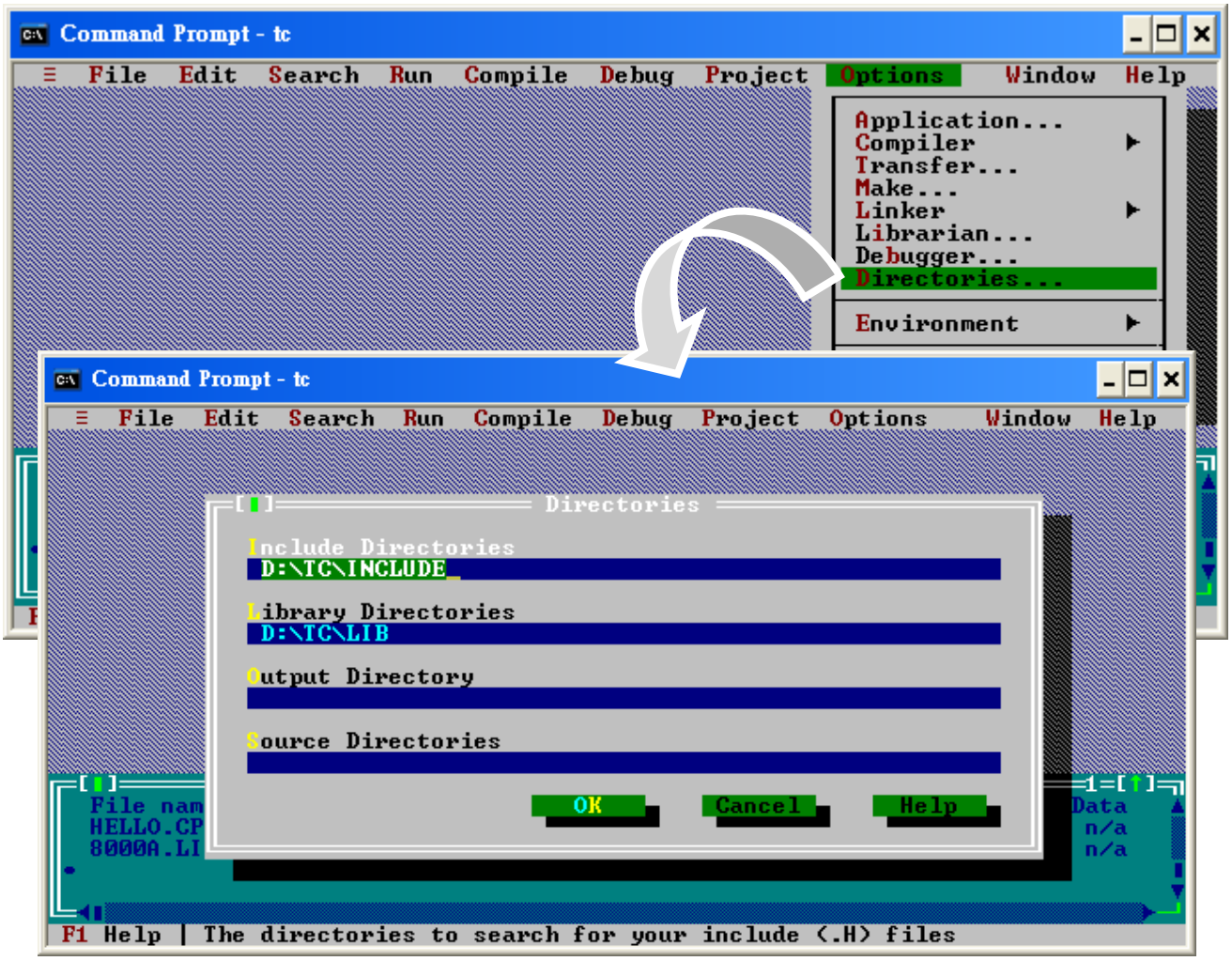
Step 11: Set the TC++1.01 include and library directories

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

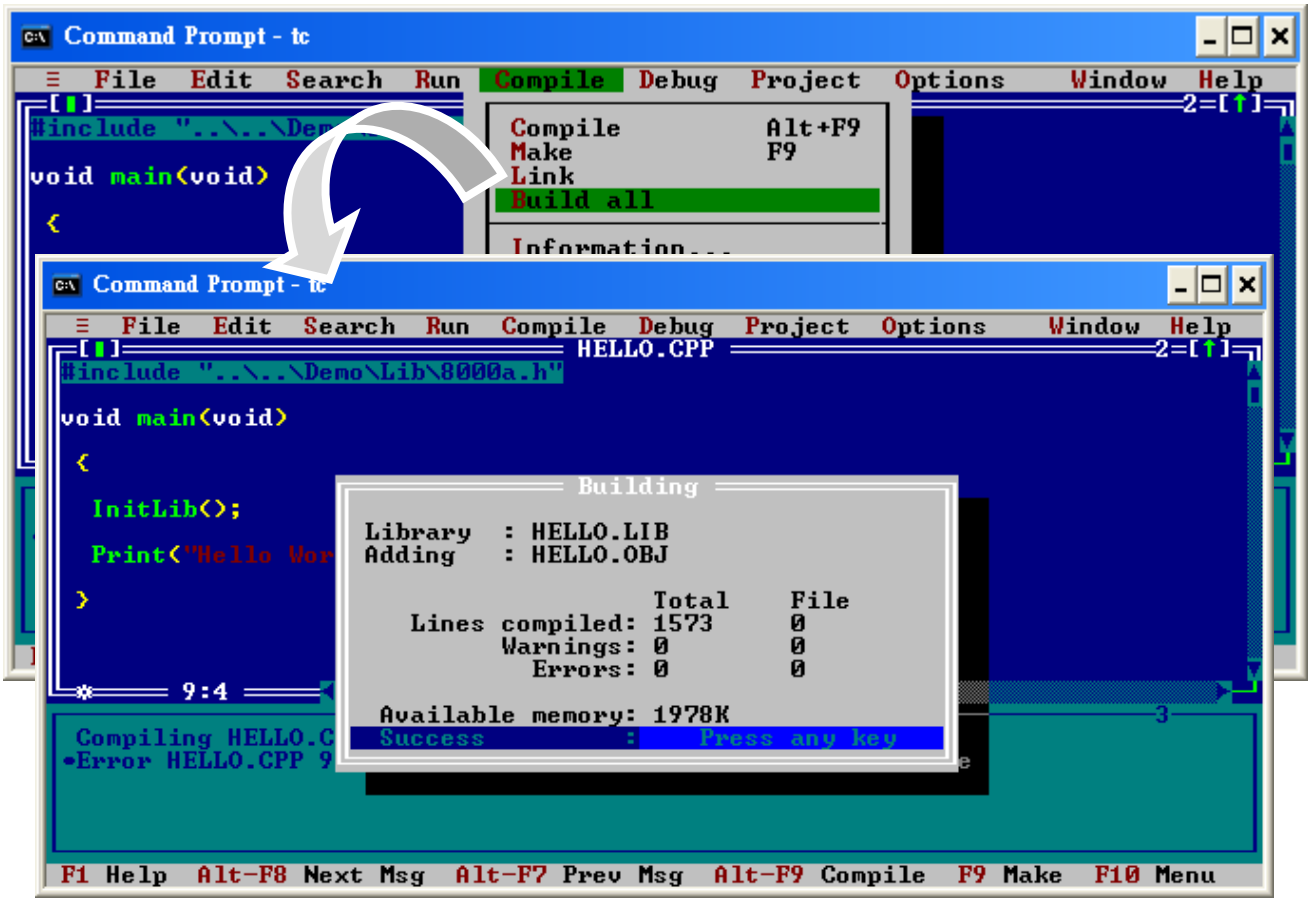
ii. On “Include Directories” option, specify the header file

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

iv. Select “OK”

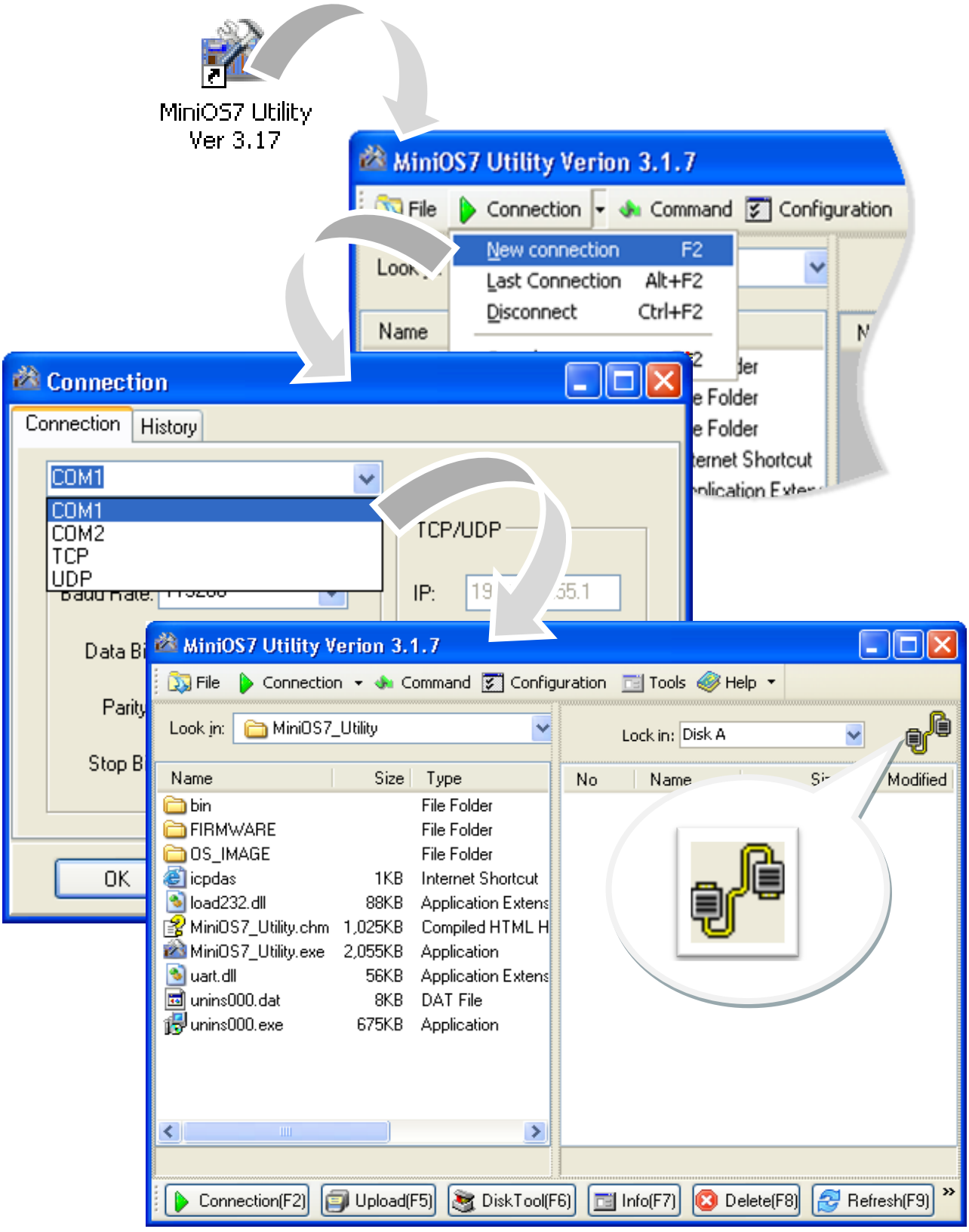


Step 12: Select “Build all” from the “Compile” menu to build the project



Step 13: Use the MiniOS7 Utility to connect the iPAC-8000

For more detailed information about this process, please refer to section "2.3.1. Establishing a connection".



Step 14: Upload and execute files

For more detailed information about this process, please refer to section “2.3.2. UPloading and executing programs on iPAC-8000

4. API and Demo Program Reference

There are several demo programs that have been designed for iPAC-8000. You can examine the demo source code, which includes numerous comments, to familiarize yourself with the MiniOS7 API, This will allow to quickly develop your own applications quickly by modifying these demo programs.

- **Basic**

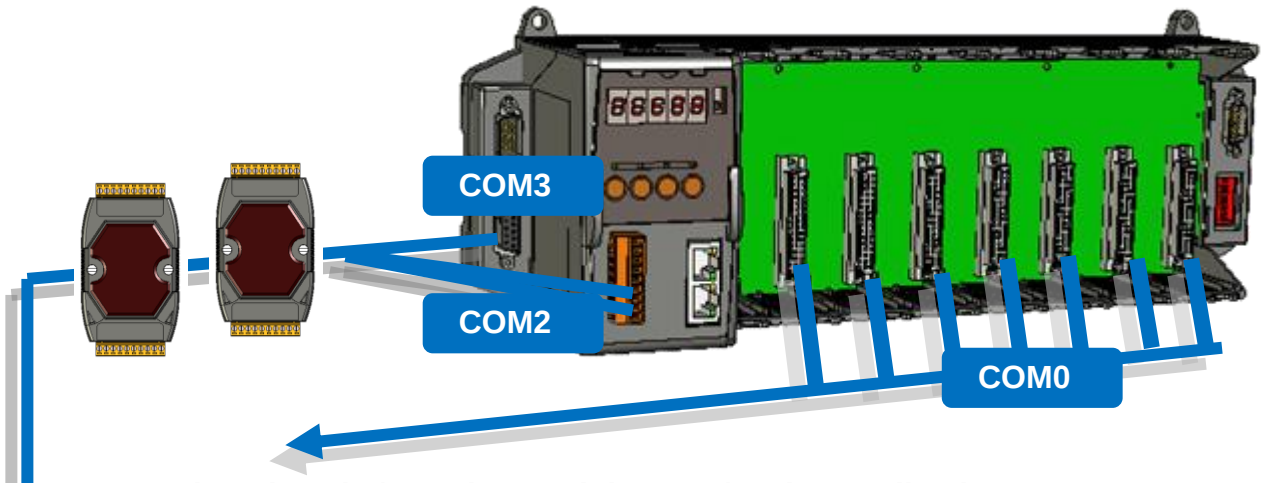
Folder	Demo	Explanation
File	Config_1_Basic	Reads information from a text file (basic).
	Config_2_Advanced	Reads a config file (text file)(advanced).
Hello	Hello_C	Reads the library version and flash memory size.
	Hello_C++	
Misc	Reset	Resets the software.
	Runprog	Illustrates how to select an item and 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 systemkey function simply and easily.
	Led	Shows how to control the red LED and 7-segment display.

Folder	Demo	Explanation
Memory	S256	Shows how to read or write to the 256K/512K byte battery backup.
DateTime	DateTime	Shows how to read and write the date and time from the RTC.
Com port	C_Style_IO	(1) Shows how to write a function to input data. (2) Shows how to receive a string. (3) Shows how to use a C function: sscanf or just use Scanf()
	Receive	Receives data from COM port. Slv_COM.c is in non-blocked mode Receive.c is in blocked mode.
	Slv_COM	A slave COM Port demo for (request/reply) or (command/response) applications.
	ToCom_In_Out	Illustrates how to Read/Write byte data via COM Port.

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

CD:\ NAPDOS\iPAC8000\ Demo\Basic\

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



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

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

► **I-7K and I-87k series modules for RS-485 Network Applications**

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

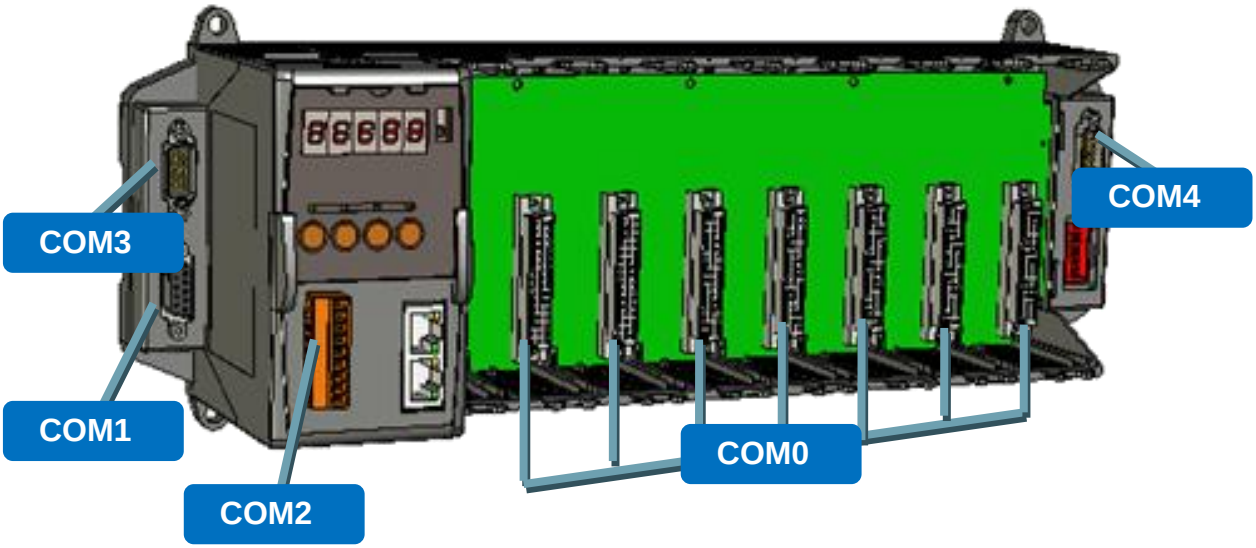
For more information about these demo programs, please refer to:

CD:\ NAPDOS\iPAC8000\ Demo\Basic\

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

4.1. API for COM port

- The iPAC-8000 includes five COM ports



4.1.1. Types of COM port functions

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

  1. MiniOS7 COM port functions

  2. (C style) Standard COM port functions

You have the alternative of MiniOS7 COM ports functions or (C style) Standard COM port functions. If you choose the ones, then the another can not be used.

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

Kinds of Functions	COM Port	Buffer		Functions			
		RX	TX	Check data	Send data	Read data	Show data
MiniOS7 COM port	0, 1, 2, etc.	1 KB	1 KB	IsCom()	ToCom()	ReadCom()	printCom()
(C style) Standard COM port	1 (Note)	512 Bytes	256 Bytes	Kbhit()	Puts() Putch()	Getch()	Print()

4.1.2. API for MiniOS7 COM port

API for using COM ports

---

1. InstallCom()

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

2. RestoreCom()

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

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

---

4. 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

---

5. 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.

6. 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);  /* Install the COM1 driver */

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

API for showing data from COM ports

---

7. 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);  /* Install the COM1 driver */

  for (i=0;i<10;i++)
  {
    printCom(1,”Test %d\n\r”, i);
  }
  Delay(10);  /* Wait for all data are transmitted to COM port */
  RestoreCom(1);
}
```

► For more demo program about the COM port, please refer to:

CD:\ NAPDOS\iPAC8000\ Demo\Basic\com_port

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

4.1.3. API for standard COM port

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

The following configurations of the standard COM port are fixed:

Baudrate=115200 bps, Data format=8 bits

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

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

1. Kbhit()

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

API for reading data from standard I/O port

2. Getch()

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

API for sending data to standard I/O port

3. Puts() – For sending a string

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

4. Putch( ) – For sending one character

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

API for showing data from standard I/O port

---

5. Print()

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

For example, read and receive data through COM1:

```
#include<stdio.h>
#include“8000E.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);
  }
}
```

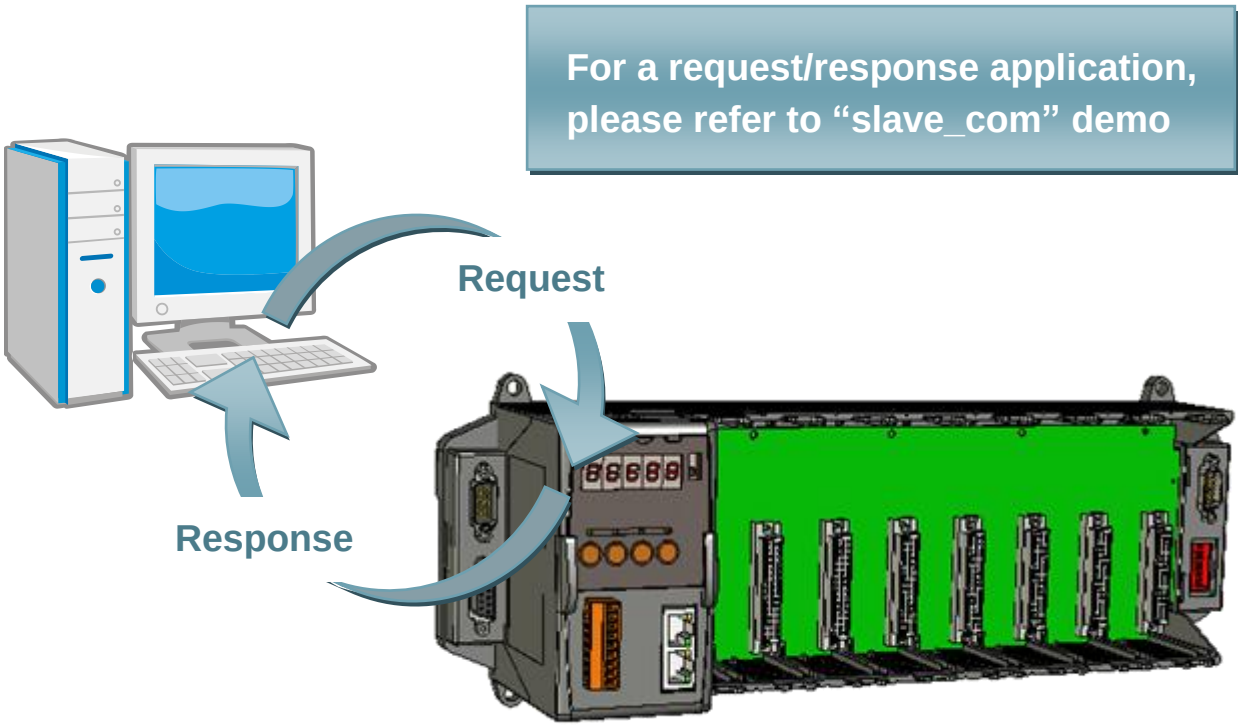
4.1.4. Comparing with MiniOS7 COM port function and Standard COM port function

For example, learn to show the ASCII code:

MiniOS7 COM port functions	Standard COM port functions
#include<stdio.h> #include"8000a.h" void main(void) { unsigned char item; InitLib(); InstallCom(1, 115200, 8, 0, 1); printCom(1,"Hits any key.\n"); printCom(1,"Hit the ESC to exit!\n"); for(;;) { if(IsCom(1)) { item=ReadCom(1); if(item=='q') { return; } else { printCom(1,"----------\n\r"); printCom(1,"char:"); 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); } } } }**

4.1.5. 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
GetCounter	>1234
SetDO1	>OK
ResetDO2	>OK
GetVersion	>V1.0.0

For more demo program about the COM port, please refer to:

CD:\ NAPDOS\iPAC8000\ Demo\Basic\com_port

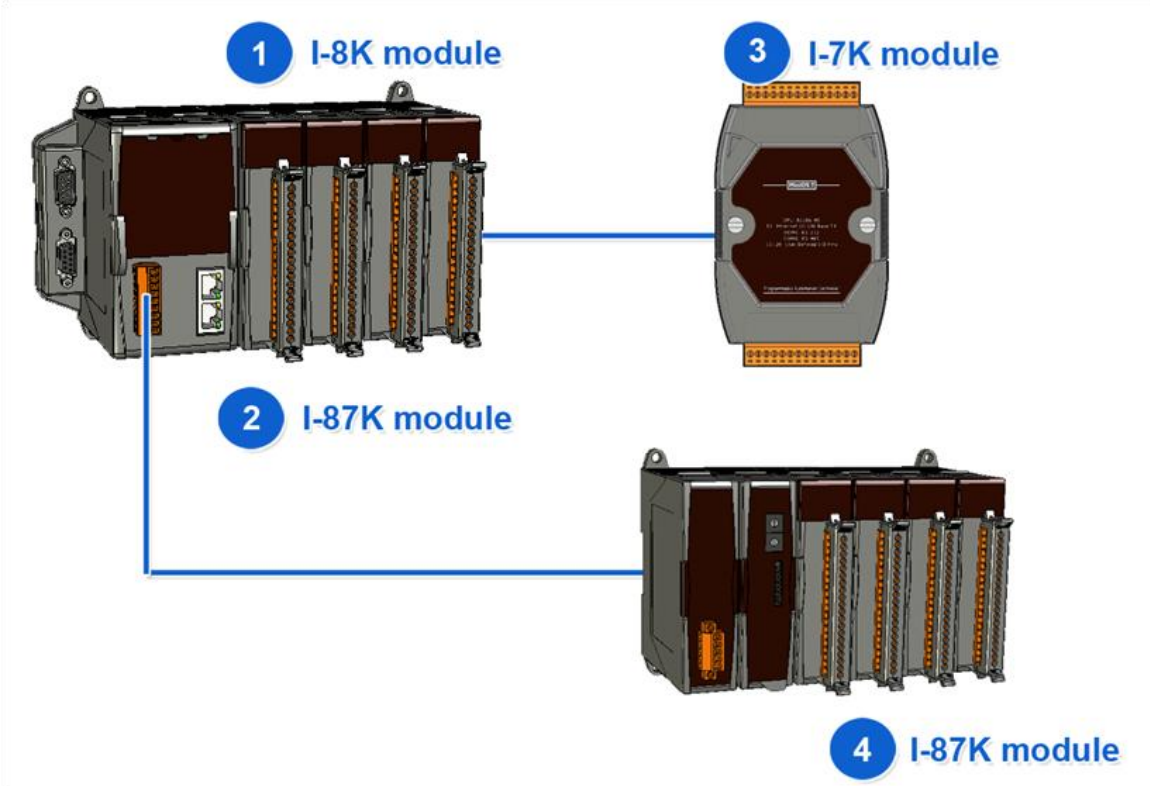
http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/basic/com_port

4.2. API for I/O modules

- The iPAC-8000 is equipped with 4/8 I/O slots to access the I-8k and I-87k series I/O modules (High profile) as shown the point 1 and point 2 in the figure below.

- The iPAC-8000 is equipped with multi-serial ports to access the I-7K series I/O modules for a wide range of RS-485 network application, as shown the point 3 in the figure below.

  The iPAC-8000 can connect to RU-87P2/4/8 to access the I-87k I/O series modules through RS-485, as shown the point 4 in the figure below.



The demo programs used for I-7K, I-8k and I-87k can be divided into the following:

► For I-8k and I-87k I/O modules in slots, please refer to:

CD:\ NAPDOS\iPAC8000\ Demo\Basic\IO_in_Slot

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

► For I-7K and I-87k I/O modules is connected to the COM ports, please refer to:

CD:\ NAPDOS\iPAC8000\ Demo\Basic\7K87K_for_COM

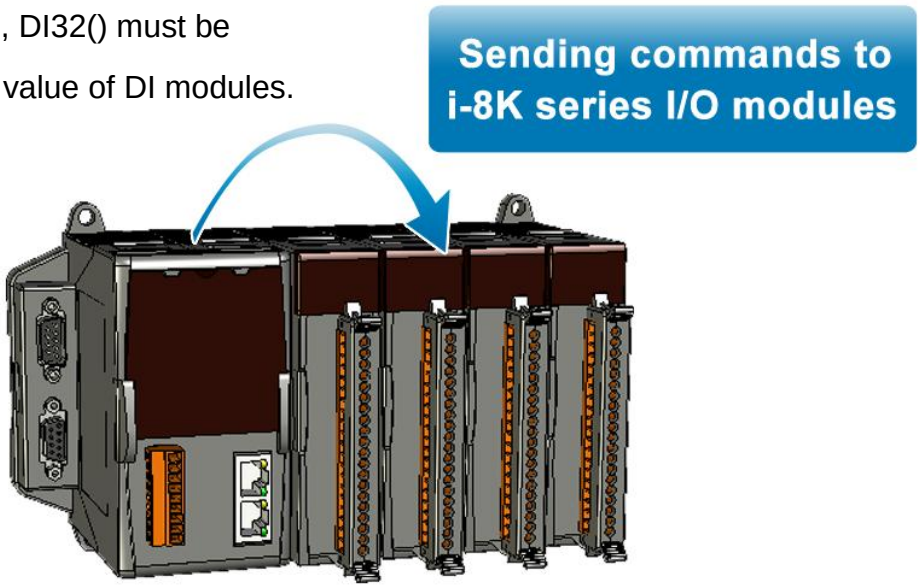
http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/basic/7k87k_for_com

4.2.1. Steps to use I-8k series I/O modules in slots

API for reading DI modules

DI_8(), DI_16(), DI_32()

The DI_8(), DI_16(), DI32() must be called to read input value of DI modules.



For example, read the input value of slot 3 DI modules:

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

 void main(void)
 {
    int DI_data, iSlot=3;
    InitLib();   /* Initiate the 8000a library */
    for(;;)
    {
    /* Read the input value of Slot 3 DI module */
    DI_data=DI_8(iSlot);
    Print(“DI Status==%x\n\r”, DI_data);
    }
  }
```

4.2.2. Steps to use I-87k series I/O modules in slots

You must perform the following steps:

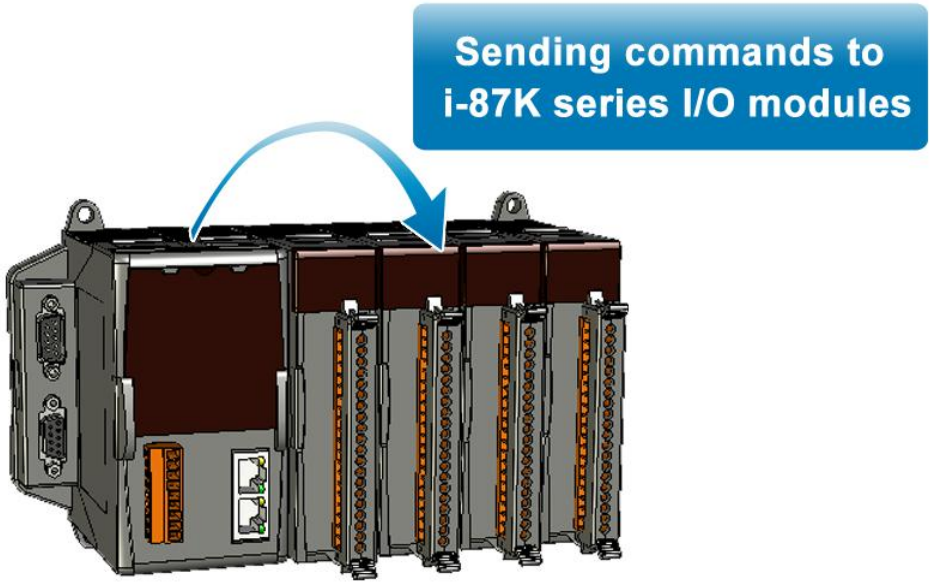
Step 1: Use **Installcom()** to install the COM port driver.

Step 2: Use **ChangeToSlot()** to assign the slot which the I-87k I/O module plugged in

Step 3: Use **SendCmdTo7000(0,…)** to send commands

Step 4: Use **ReceiveResponseFrom7000_ms()** to get the response.

Step 5: Use **RestoreCom()** to restore the COM port driver



The following settings of the COM0 are fixed:

Baudrate = 115200 bps

Data bit = 8 bits

Parity check = None

Stop bit = 1

The following settings of I-87k series I/O modules that plugged in slots are fixed:

Address = 0

Check sum = Disable

Besides, the **ChangeToSlot()** function must be called.

For example, send a command, '$00M', to the I-87k I/O module that pluged on the slot 7 of the iPAC-8000 for getting the module name:

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

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

    InstallCom(0,115200,8,0,1);   /* Install the COM0 driver */
    InstallCom(1,115200,8,0,1);   /* Install the COM1 driver */

    ChangeToSlot(7);

    SendCmdTo7000(0,"$00M",0);   /* Send a command to COM0 */

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

    printCom(1,"Module Name = %s", InBuf0);
    Delay(10);   /* Wait for all data are transmitted to COM port */
    RestoreCom(0);   /* Uninstall the COM0 driver */

    RestoreCom(1);   /* Uninstall the COM1 driver */
  }
```

4.2.3. Steps to use I-7K and I-87k series I/O modules that are connected to COM port

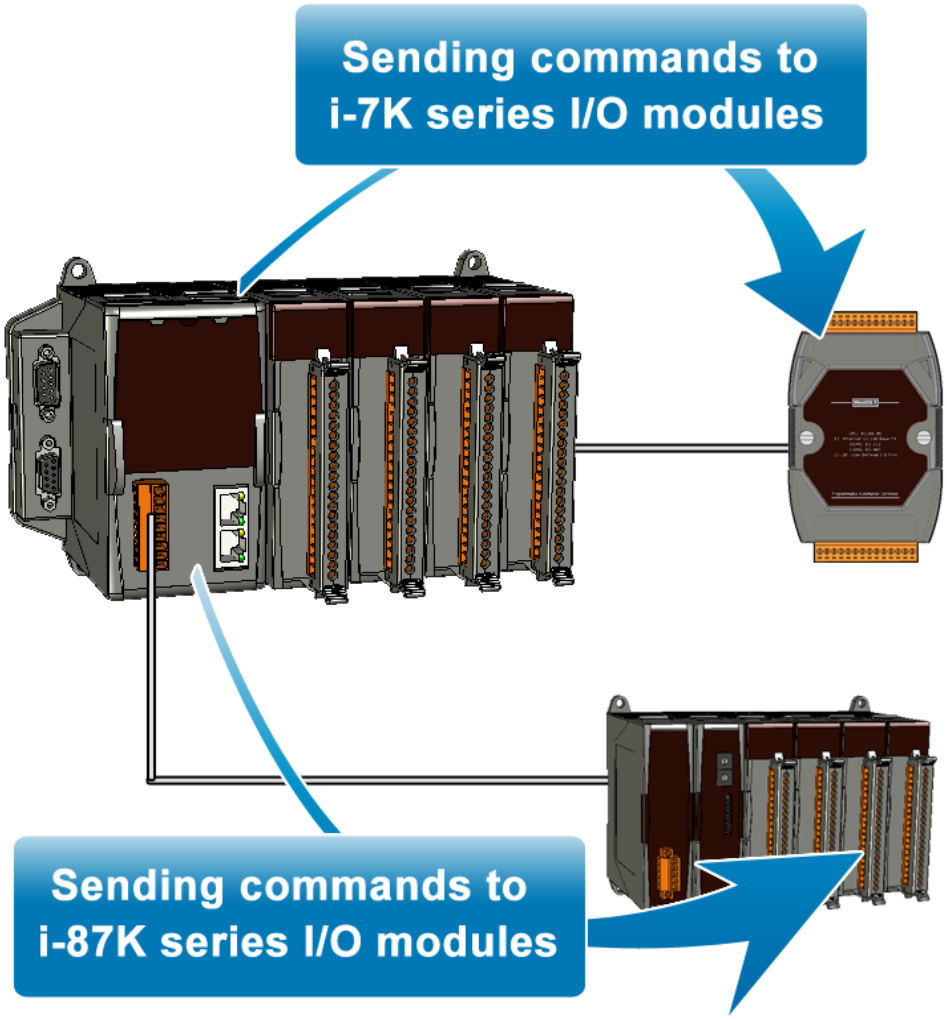
You must perform the following steps:

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

Step 2: Use **SendCmdTo7000(0,…)** to send commands

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

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

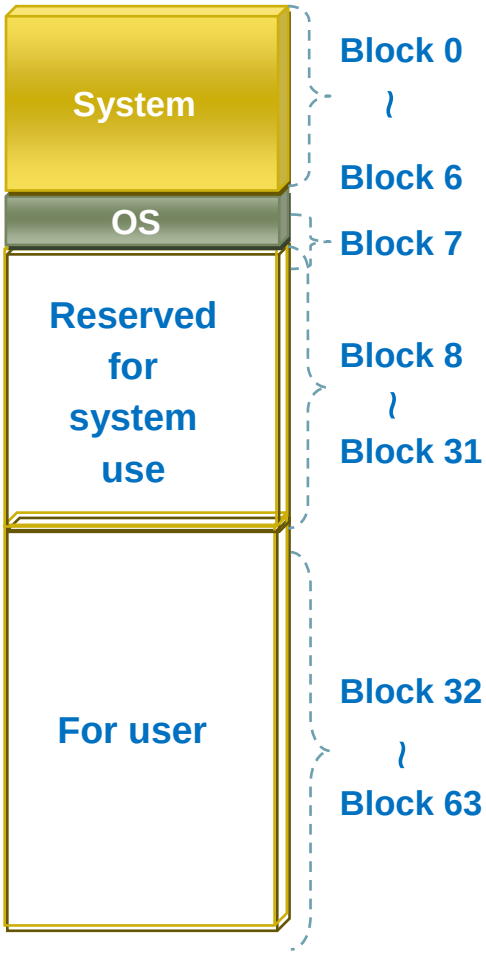


For example, send a command, '$00M', to the I-7K or I-87k I/O module that connected on the COM2 of the iPAC-8000 for getting the module name:

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

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

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

    SendCmdTo7000(2,"$00M",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);   /* Uninstall the COM1 driver */

    RestoreCom(2);   /* Uninstall the COM2 driver */
  }
```

4.3. API for EEPROM

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



API for writing data to the EEPROM

1. EE_WriteEnable()

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

2. EE_WriteProtect()

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

3. EE_MultiWrite()

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

API for reading data from the EEPROM

4. EE_MultiRead()

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

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

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

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

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

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

To write an integer to the EEPROM, the EE_WriteEnable() function must be called twice, in the same manner as writing data to the NVRAM

► For more demo program about the EEPROM, please refer to:

CD:\ NAPDOS\iPAC8000\ Demo\Basic\misc

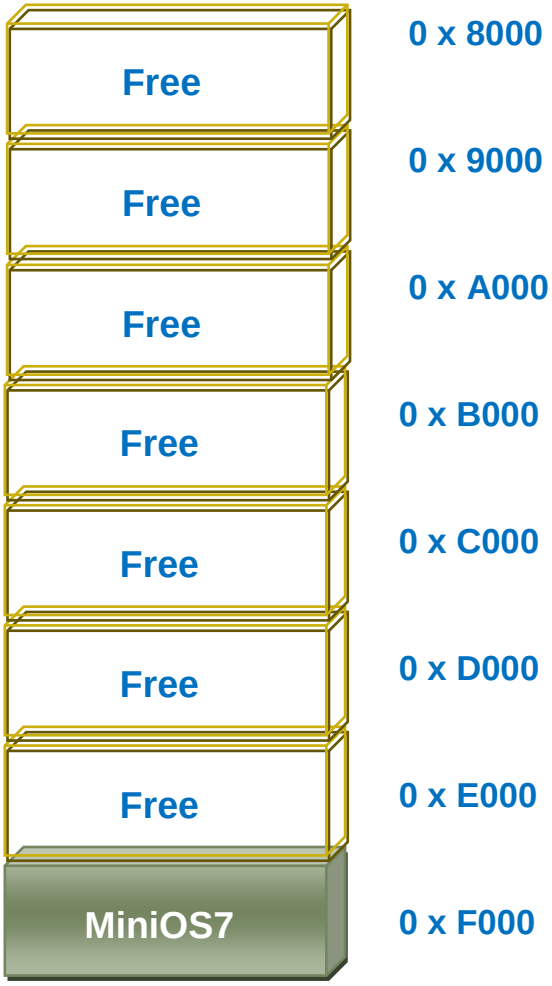
http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/basic/misc

4.4. API for Flash Memory

Free: 448 K bytes

MiniOS7: 64 K bytes

Total Size: 512 K bytes



- The iPAC-8000 module contains 512K bytes 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 their is completed, new data can be written.

API for writing data to the Flash Memory

1. FlashWrite()

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

API for reading data from the Flash Memory

2. FlashRead()

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

For example, to 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 */
    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;
  }
```

► For more demo program about the Flash memory, please refer to:

CD:\ NAPDOS\iPAC8000\ Demo\Basic\misc

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

4.5. API for NVRAM and RTC

- The iPAC-8000 is equipped with an RTC (Real Time Clock), and 31 bytes of NVRAM memory can be used to store data.
- NVRAM is the same as SRAM, but it uses a battery to retain the data, so the data. store in the NVRAM is not lost when the module is powered off and can be used for 10 years.
- NVRAM has no limit on the number of times the data can be written. (Both Flash and EEPROM both have a limit on the numbers of data can be re-written.)

API for writing data to the NVRAM

---

1. WriteNVRAM()

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

API for reading data from the NVRAM

---

2. ReadNVRAM()

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

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

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

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

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

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

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

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

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

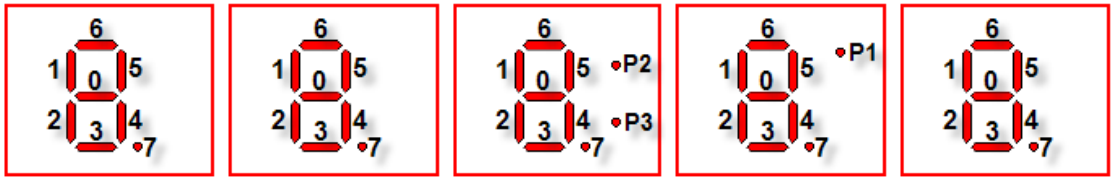
► For more demo program about the NVRAM and RTC, please refer to:

CD:\ NAPDOS\iPAC8000\Demo\Basic\misc

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

4.6. API for 5-Digit LED

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



API for controlling the 5-Digit 7-SEG LED

---

1. Init5DigitLed()

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

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

---

2. Show5DigitLed()

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

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

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

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

    Init5DigitLed();

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

  }
```

► For more demo program about the 5-digit 7-SEG LEDs, please refer to:

CD:\ NAPDOS\iPAC8000\ Demo\Basic\smmi
http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/basic/smmi

4.7. API for Timer and WatchDogTimer

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

API that can be used to control the Timer

---

1. TimerOpen()

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

API for reading the Timer

---

2. TimerResetValue()

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

3. TimerReadValue()

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

API for stopping the Timer

---

4. TimerClose()

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

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

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

 void main(void)
 {
 Unsigned long time iTime;

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

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

► For more demo program about the timer, please refer to:

CD:\ NAPDOS\iPAC8000\ Demo\Basic\timer
http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/basic/timer

4.8. API for WatchDog Timer (WDT)

- The default WatchDog timer (WDT) value for the iPAC-8000 module is fixed at 0.8 seconds for MiniOS7 version 2.0.
- When the iPAC-8000 is first powered on, the WatchDog Timer will always be enabled.
- The MiniOS7 for the iPAC-8000 will automatically refresh the WatchDog Timer after being powered on. The software driver can be called by a user program to prevent the MinOS7 from refreshing the WatchDog Timer.

API for refreshing WDT

---

1. EnableWDT()

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

2. RefreshWDT()

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

3. DisableWDT()

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

For example, to refresh the Watchdog Timer:

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

 void main(void)
 {
 Unsigned long time iTime;

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

► For more demo program about the WatchDog Timer, please refer to:

CD:\ NAPDOS\iPAC8000\ Demo\Basic\Misc
http://ftp.icpdas.com/pub/cd/8000cd/napdos/ipac8000/demo/basic/misc

Appendix A. Frame Ground

Electronic circuits are constantly vulnerable to Electro-Static Discharge (ESD), which become worse in a continental climate area. Some I-7000 ,M-7000 and I-8000 series modules feature a new design for the frame ground, which provides a path for bypassing ESD, allowing enhanced static protection (ESD) capability and ensures that the module is more reliable.

The following options will provide a better protection for the module:

The iPAC-8000 controller has a metallic board attached to the back of the plastic basket as shown in the Figure 2-1 below. When mounted to the DIN rail, connect the DIN rail to the earth ground because the DIN rail is in contact with the upper frame ground as shown in the Figure 2-2 below.

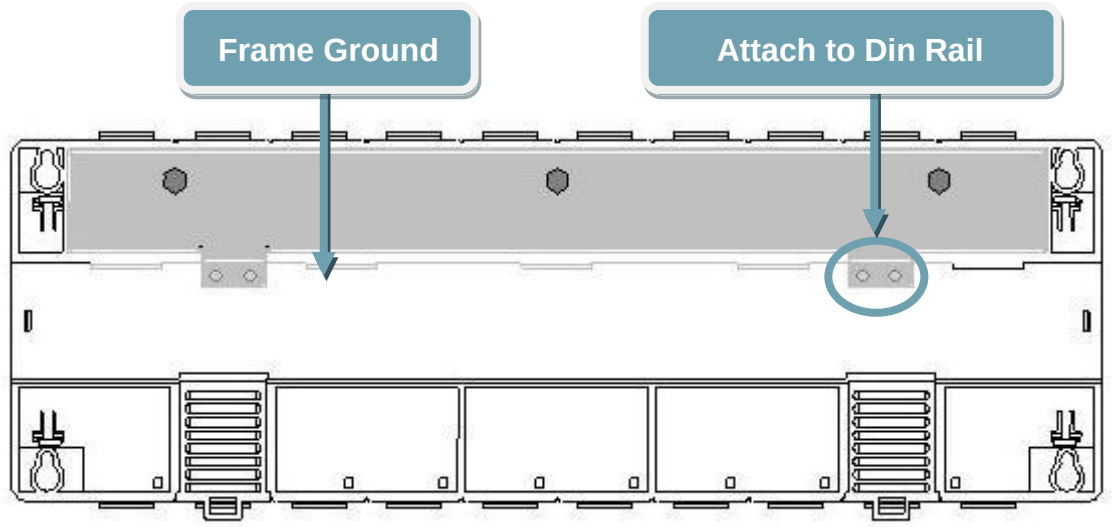


Figure B-1

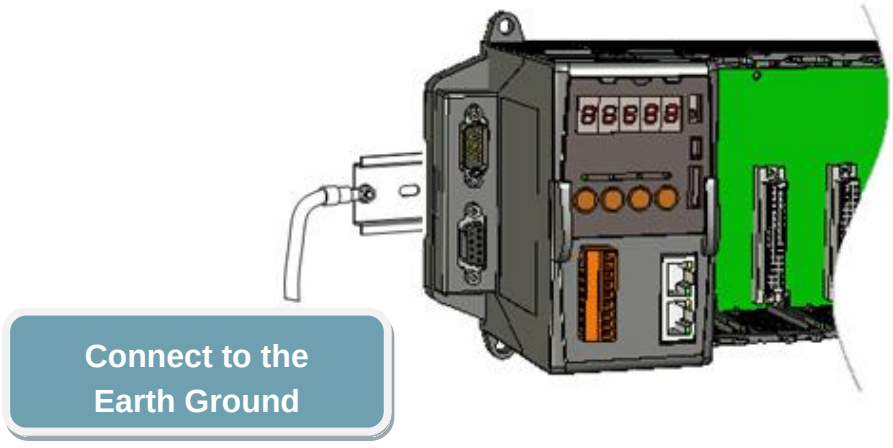
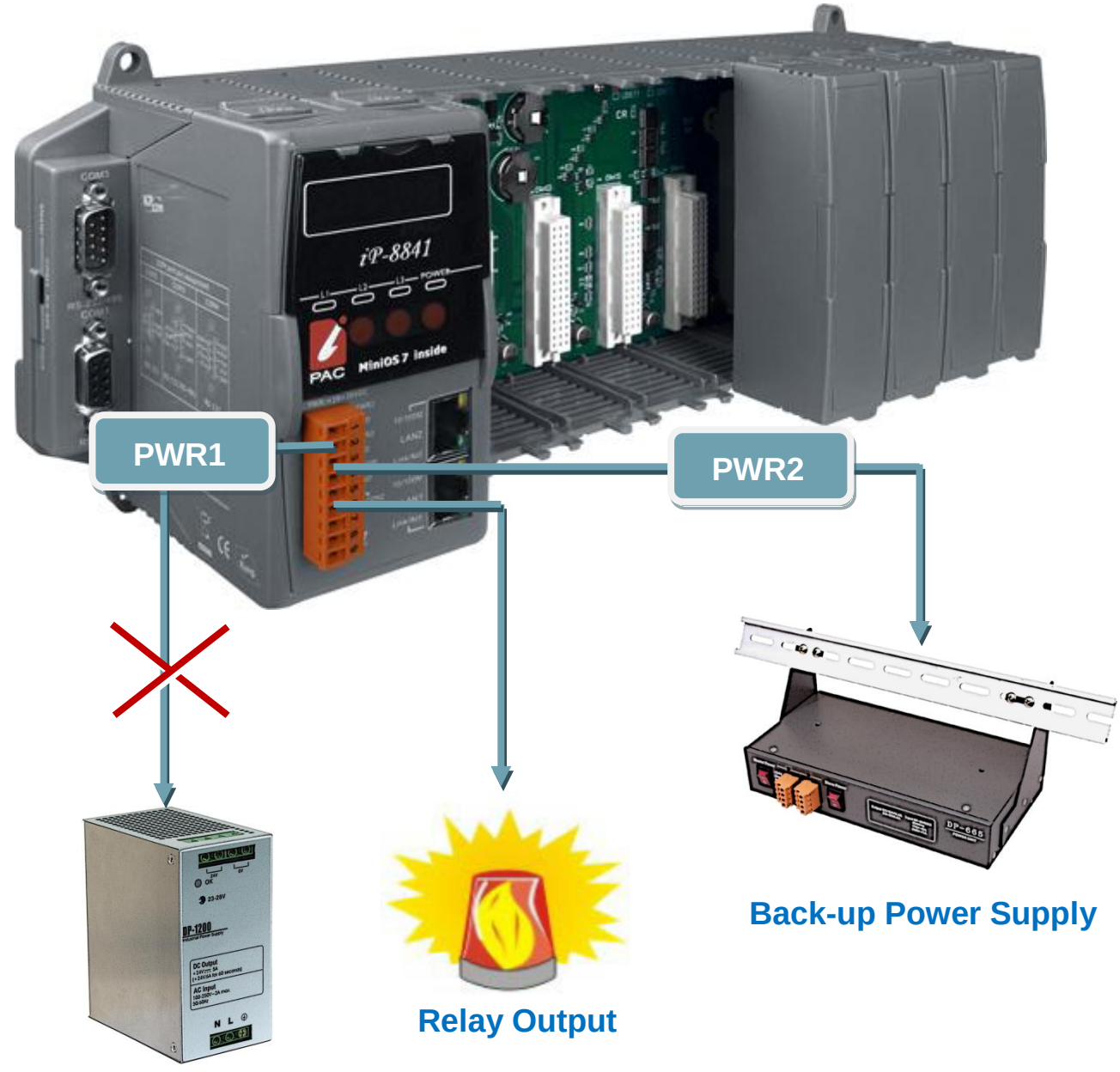


Figure B-2

Appendix B. Redundant Power

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

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



Appendix C. I-8k and I-87k serial modules

There are 1/4/8 slot options to expand local I/O. And the I/O modules can be parallel bus type (high profile I-8k series) and serial bus type (high profile I-87k series).

The difference between them is

Item	I-8k Series	I-87k Series
Microprocessor	No	Yes (8051)
Communication interface	Parallel bus	Serial bus
Communication speed	Fast	Slow
DI latched function	No	Yes
Counter input (for digital input module)	No	Yes (100 Hz)
Power on value	No	Yes
Safe value	No	Yes
Programmable slew-rate for AO module	No	Yes

Appendix D. What is MiniOS7

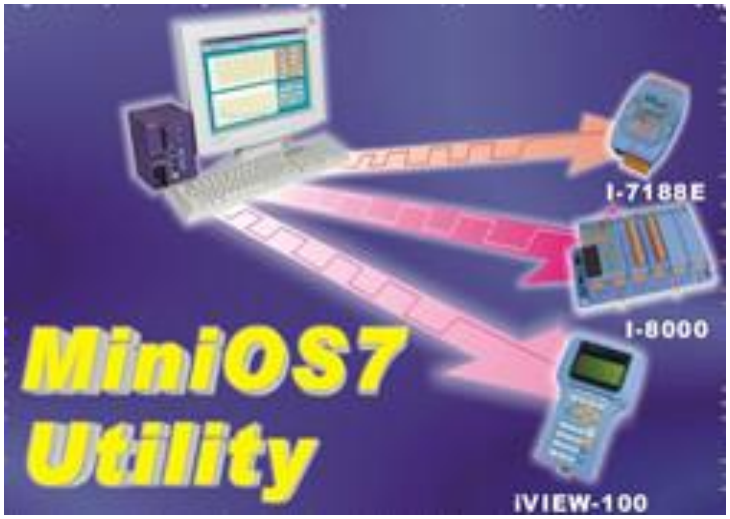
MiniOS7 is an embedded ROM-DOS operating system design by ICP DAS. It is functionally equivalent to other brands of DOS, and can run programs that are executable under a standard DOS. Photo Shop + office 2007yji4

DOS (whether PC-DOS, MS-DOS or ROMDOS) 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 compares the features between MiniOS7 and ROM-DOS：

Feature	MiniOS7	ROM-DOS
Power-up time	0.1 sec	4 ~ 5 sec
More compact size	< 64 K bytes	64 K bytes
Support for I/O expansion bus	Yes	No
Support for ASIC key	Yes	No
Flash ROM management	Yes	No
O.S. update (Download)	Yes	No
Built-in hardware diagnostic functions	Yes	No
Direct control of 7000 series modules	Yes	No
Customer ODM functions	Yes	No
Free of charge	Yes	No

Appendix E. What is MiniOS7 Utility



MiniOS7 Utility is a tool for configuring, uploading files to all products embedded with ICPDAS MiniOS7 with easiness and quickness.

Note：Since version 3.1.1, the Utility can allow users remotely access the controllers (7188E,8000E,…ect) 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

Including Frequently Used Tools

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

PC System Requirements

1. IBM compatible PC
2. Windows 95 /98/NT/2000/XP

Supported Products

1. 7188XA
2. 7188XB
3. 7188XC
4. 7188EX series
5. All i-8000 series
6. iView100
7. uPAC-7186XB
8. uPAC-7186EX
9. ET-6000 series
10. ET-7000 series

Download location：

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

Appendix F. What is VxComm Utility

The VxComm Driver creates COM port(s) and maps them to the Ethernet port(s) of the PDS/8000E/7188E.

The user's RS-232 client programs need only to change to the different COM port to get the access of serial devices that are allocated in the Internet or Ethernet network via the PDS/8000E/7188E.

The VxComm Driver supports Windows NT 4.0, 2000/XP/2003 and 32-bit Vista (Vista32), and is totally free for users using ICP DAS PDS/8000E/7188E… series products.

For downloading and more information, please refer to the following link:

http://www.icpdas.com/products/Software/VxComm/vxcomm.htm

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:\8000\NAPDOS\8000\841x881x\Demo\hello\Hello_C\gotc.bat

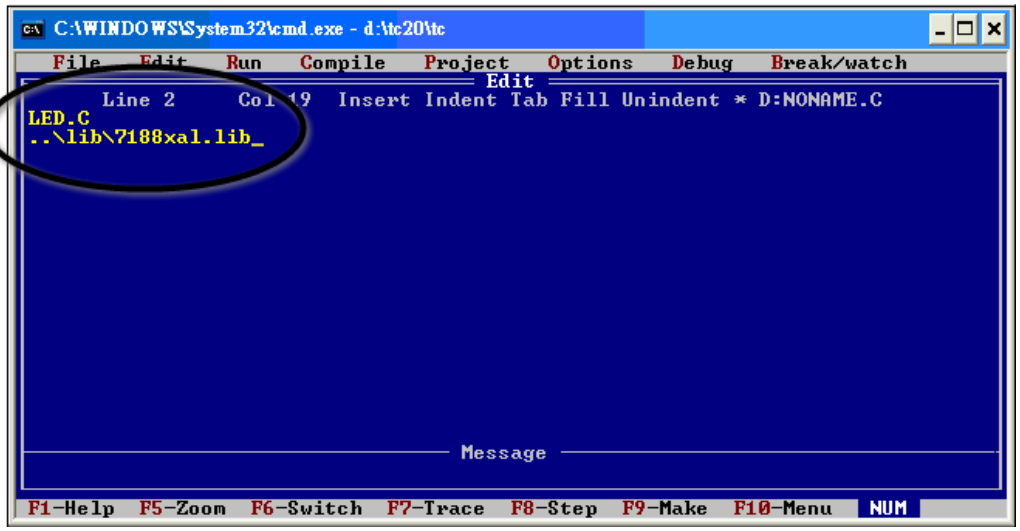
tcc -Ic:\tc\include -Lc:\tc\lib hello1.c ..\..\lib\8000e.lib

2：Using the TC Integrated Environment

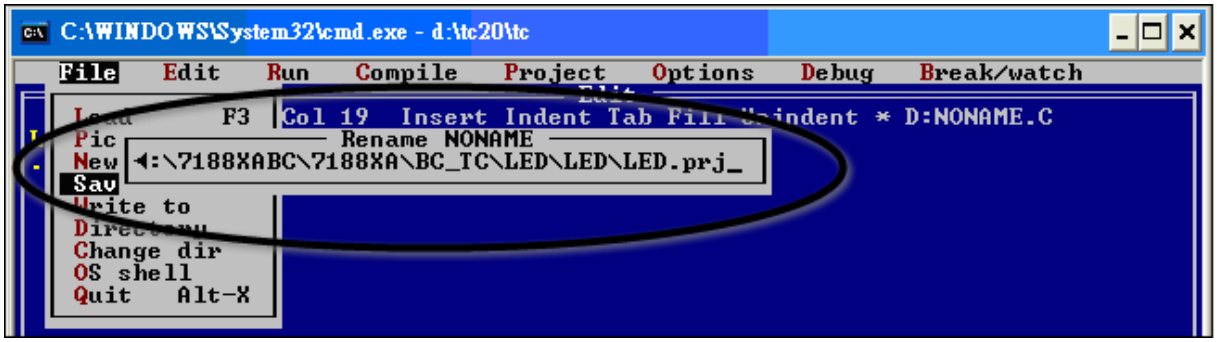
Step 1: Executing the TC 2.01

Step 2: Editing the Project file

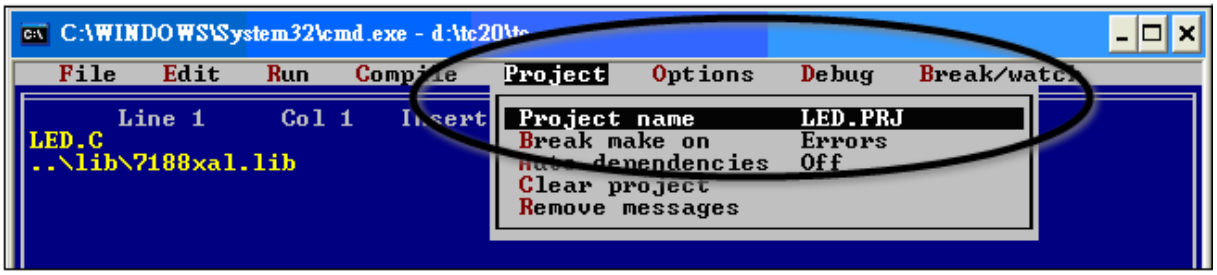
Adding the necessary library and file to the project



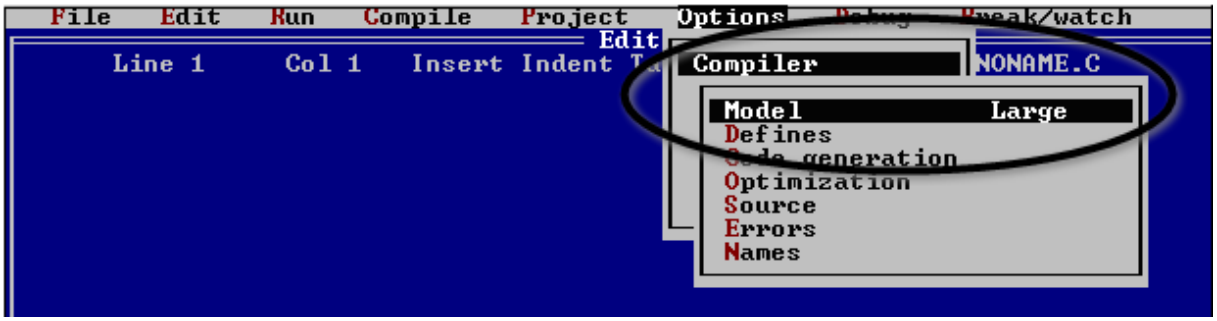
Step 3: Save the project and entering a name, such as LED.prj

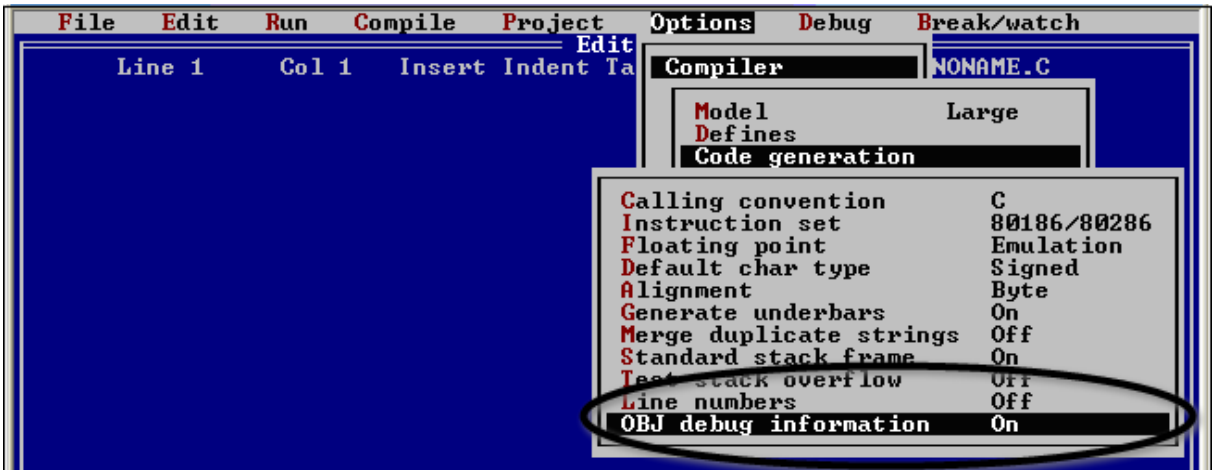


Step 4: Load the Project

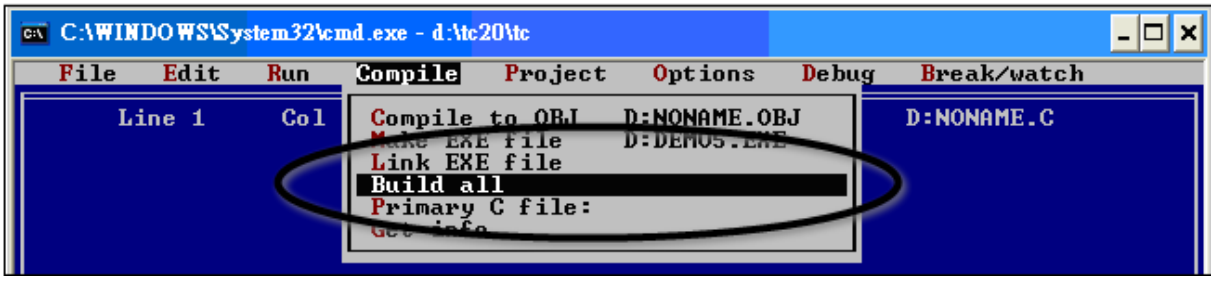


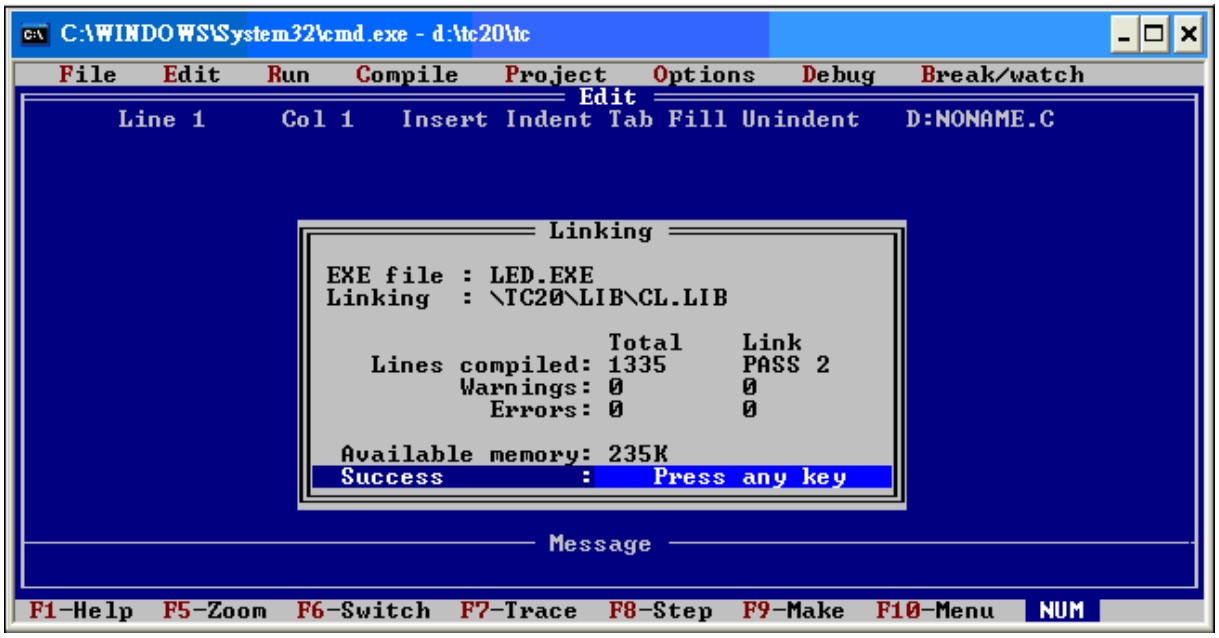
Step 5: Change the Memory model (Large for 8000e.lib) and set the Code Generation to 80186/80286





Step 6: Building the project

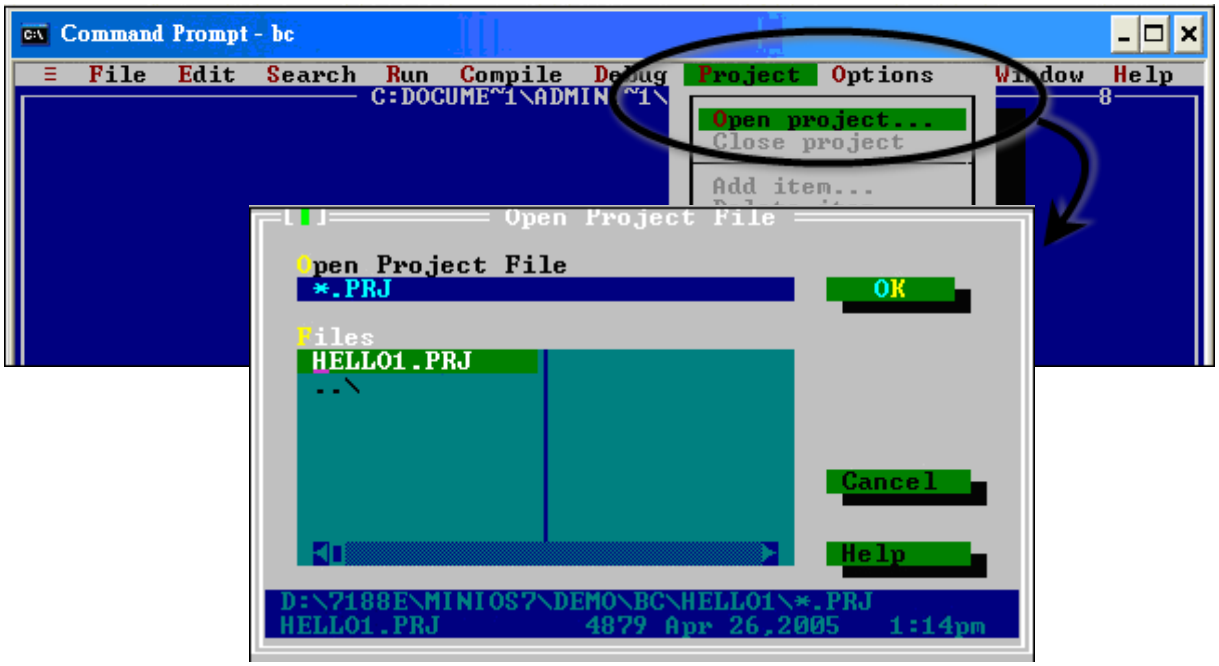




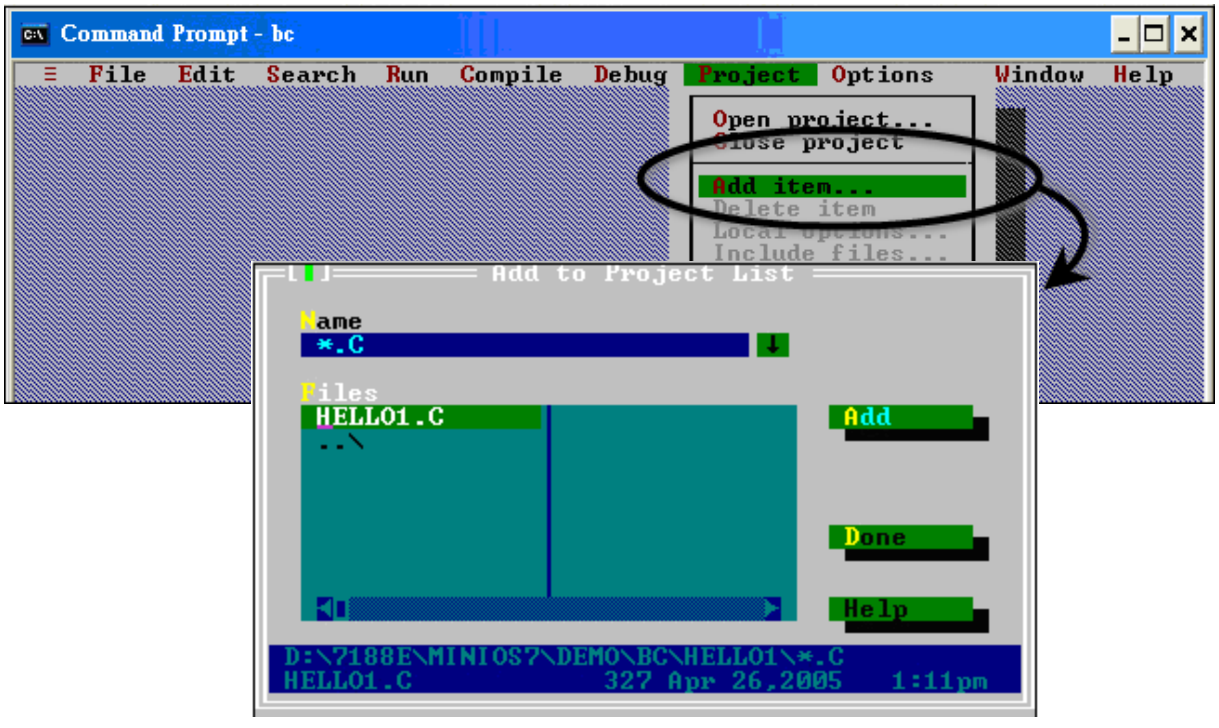
G.2. BC++ 3.1. IDE

Step 1: Executing the Borland C++ 3.1

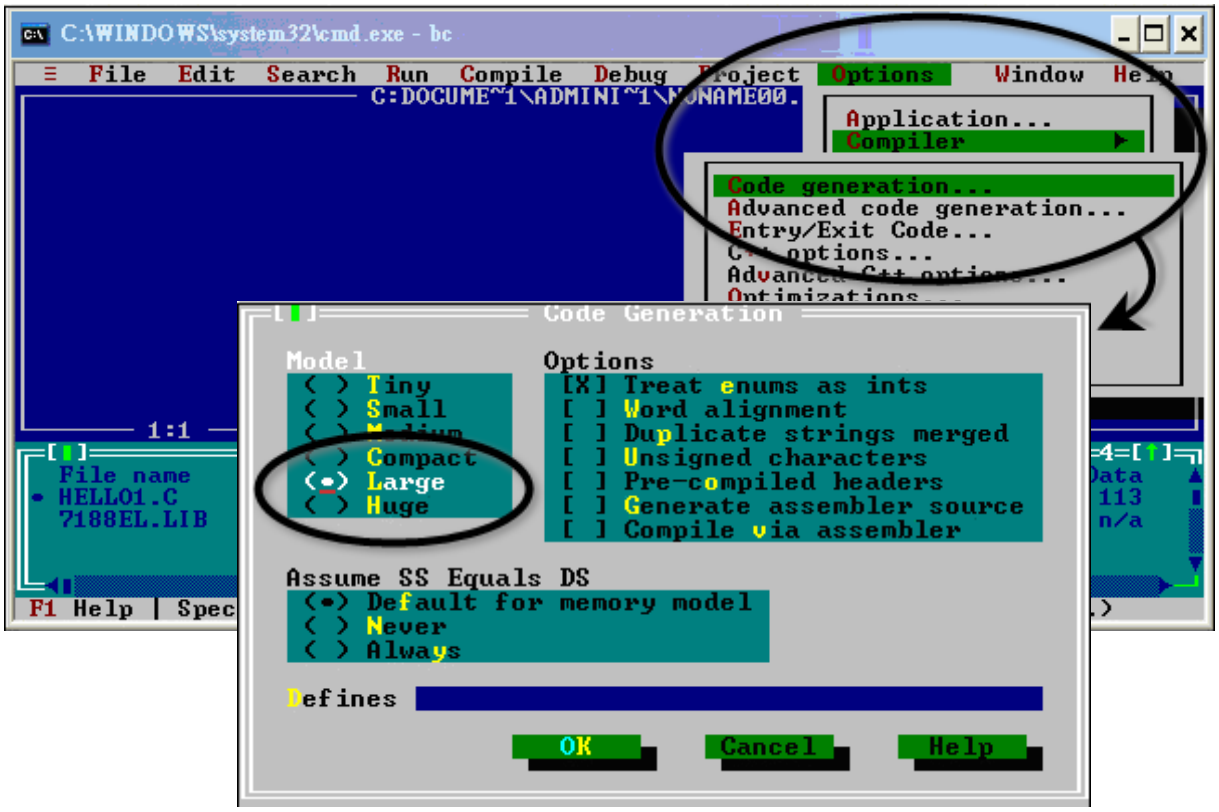
Step 2: Creating a new project file (*.prj)



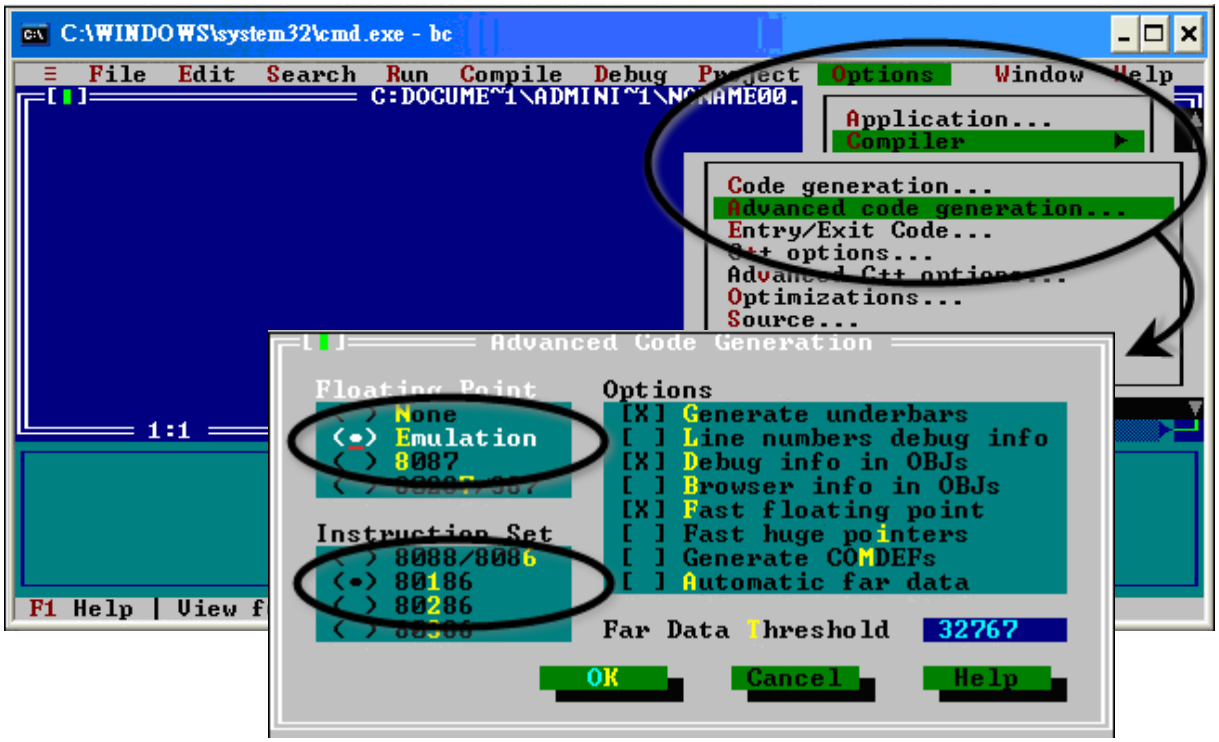
Step 3: Add all the necessary files to the project



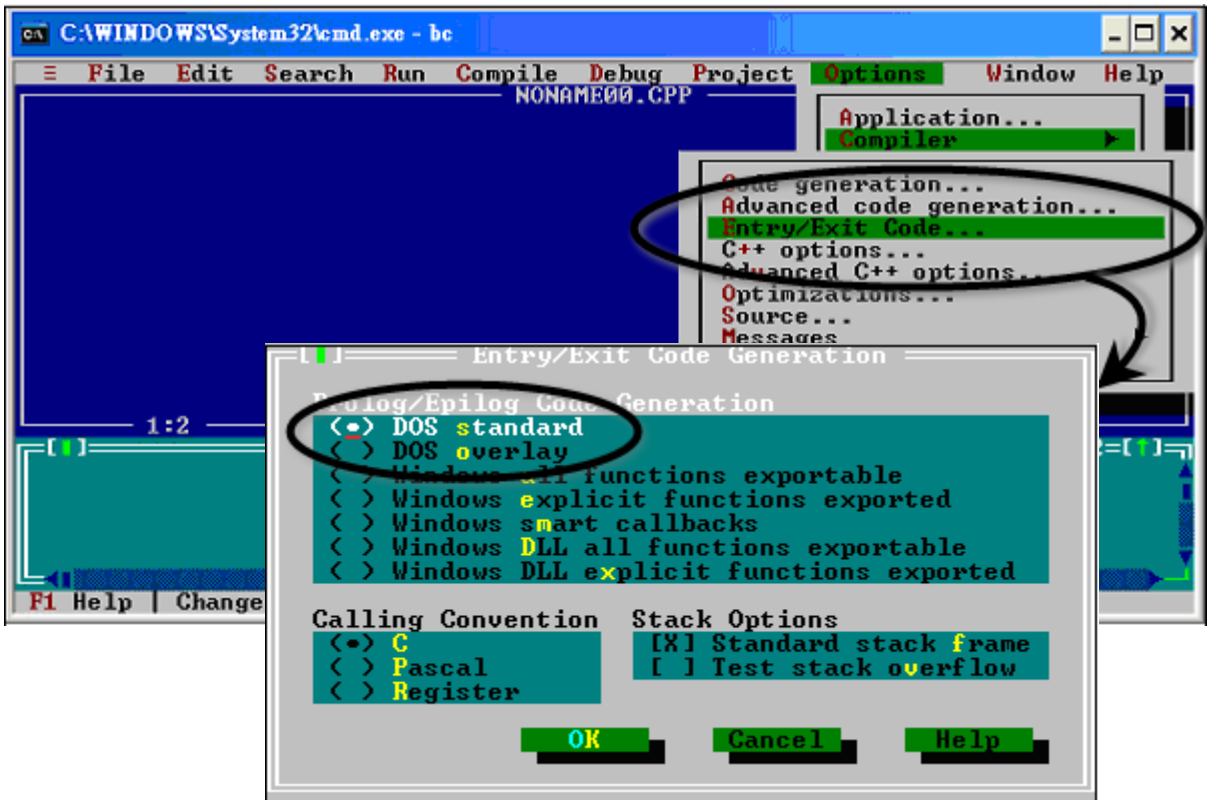
Step 4: Change the Memory model (Large for 8000e.lib)



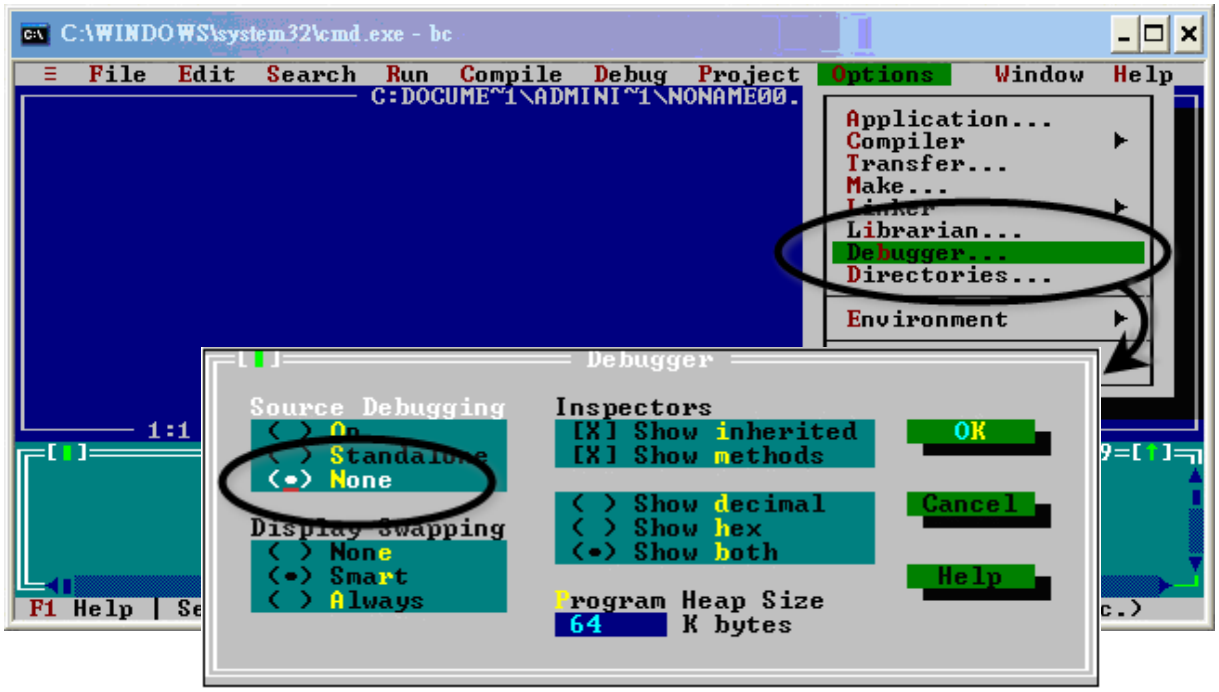
Step 5: Set the Advanced code generation options and Set the Floating Point to Emulation and the Instruction Set to 80186



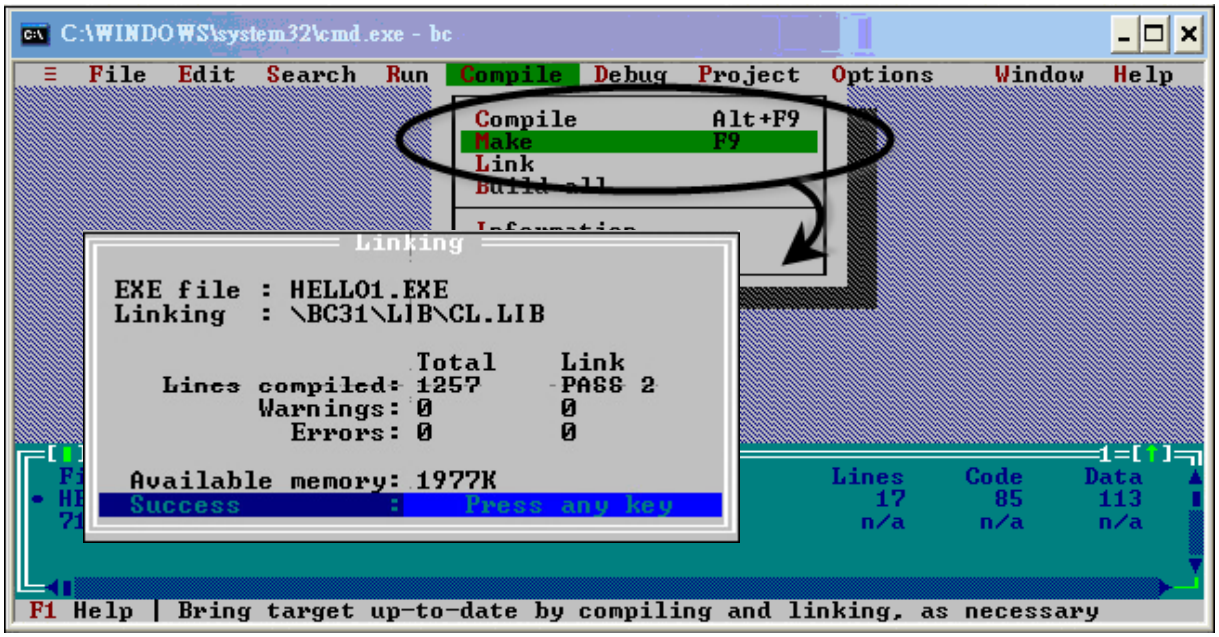
Step 6: Set the Entry/Exit Code Generation option and setting the DOS standard



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

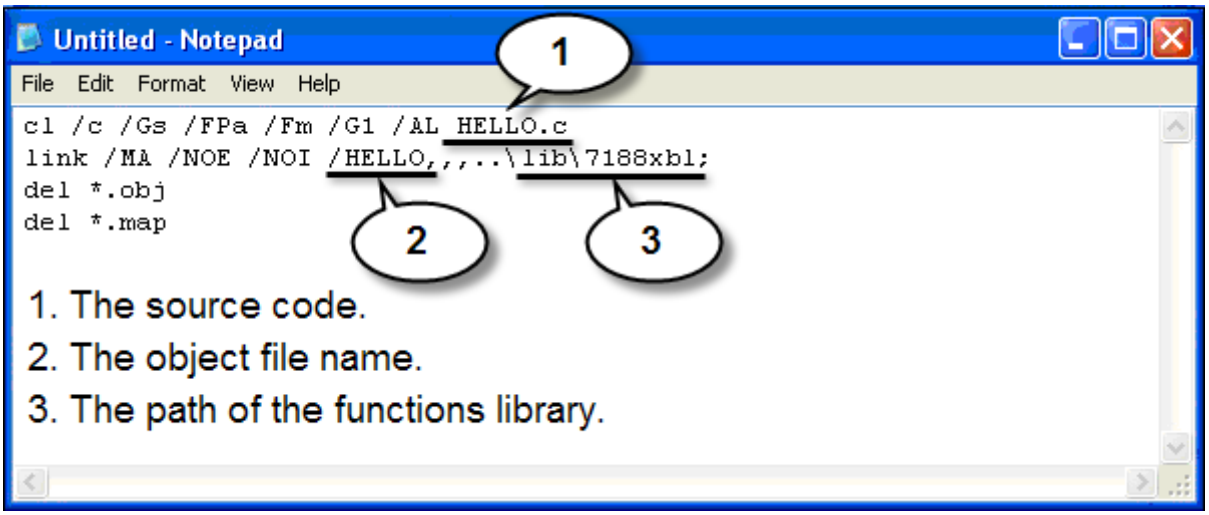


Step 8: Make the project



G.3. MSC 6.00

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

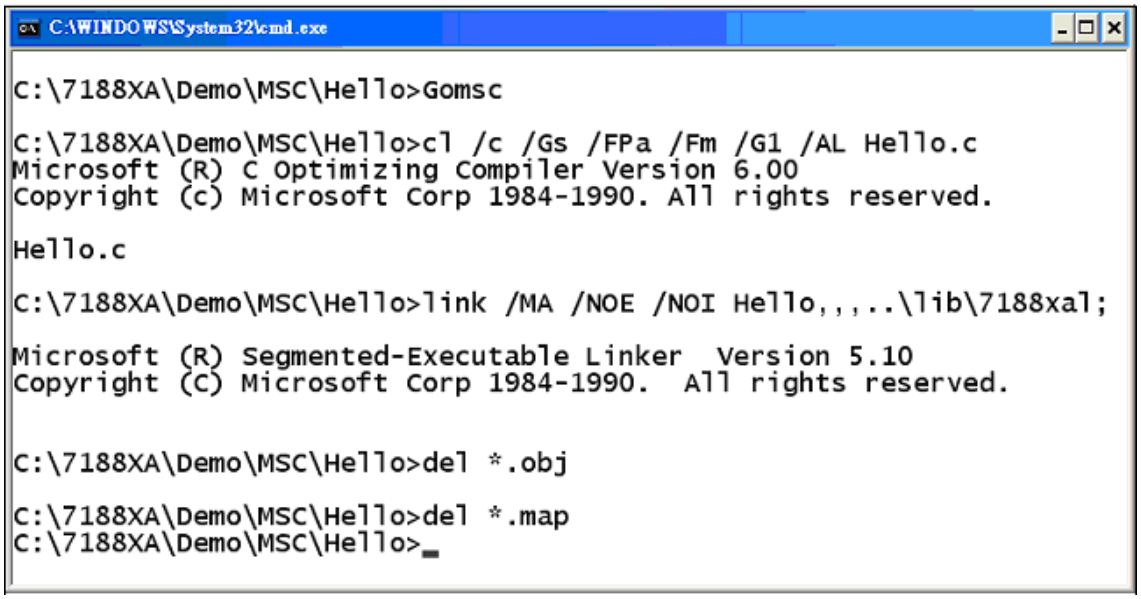


Note: ：/C Don't strip comments /GS No stack checking

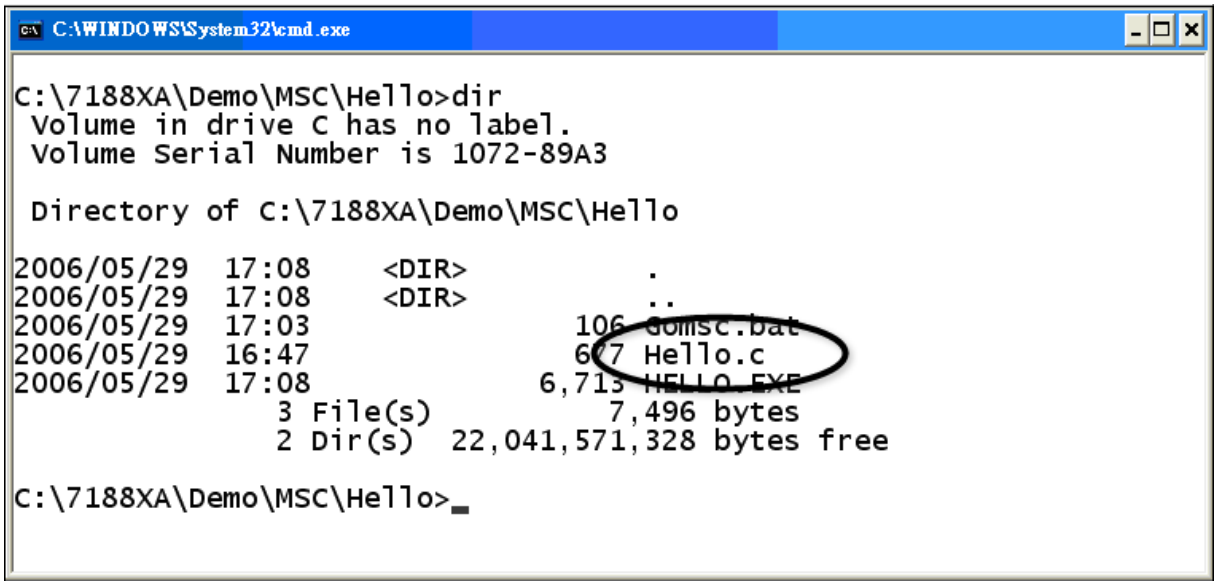
/Fpa：Calls with altmath /Fm [map file]

/G1：186 instructions /AL Large model

Step 2: Run the Gomsc.bat file

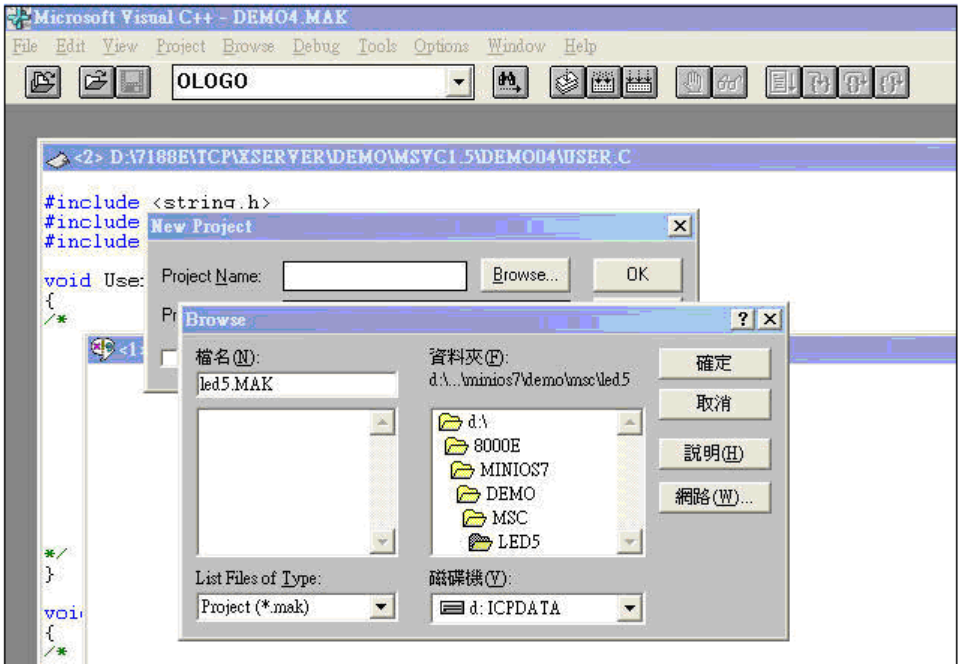


Step 3: A new executable file will be created if it is successfully compiled

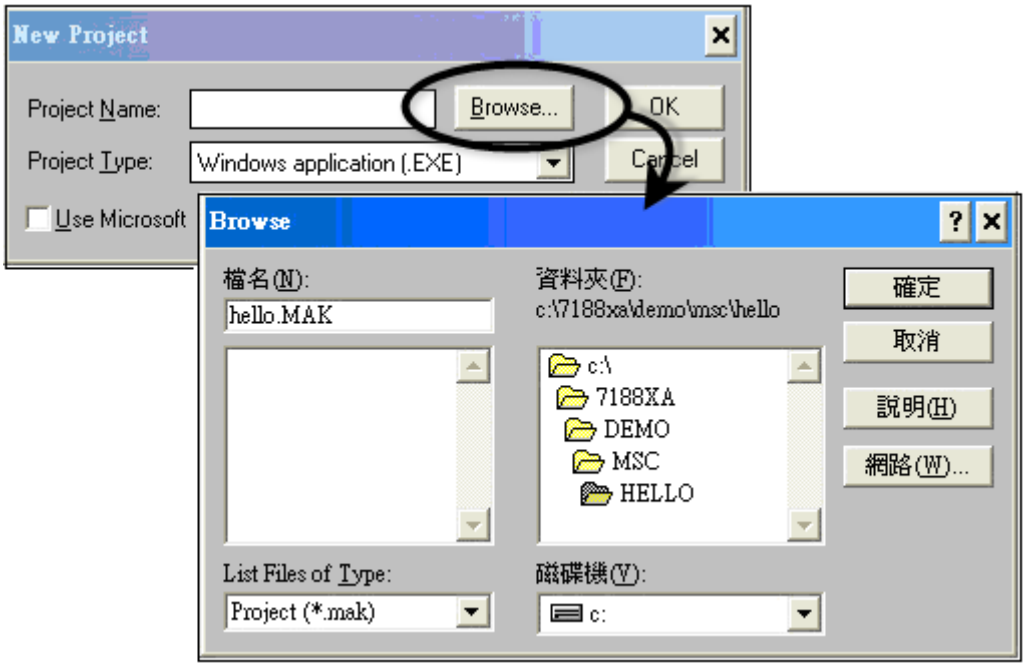


G.4. MSVC 1.50

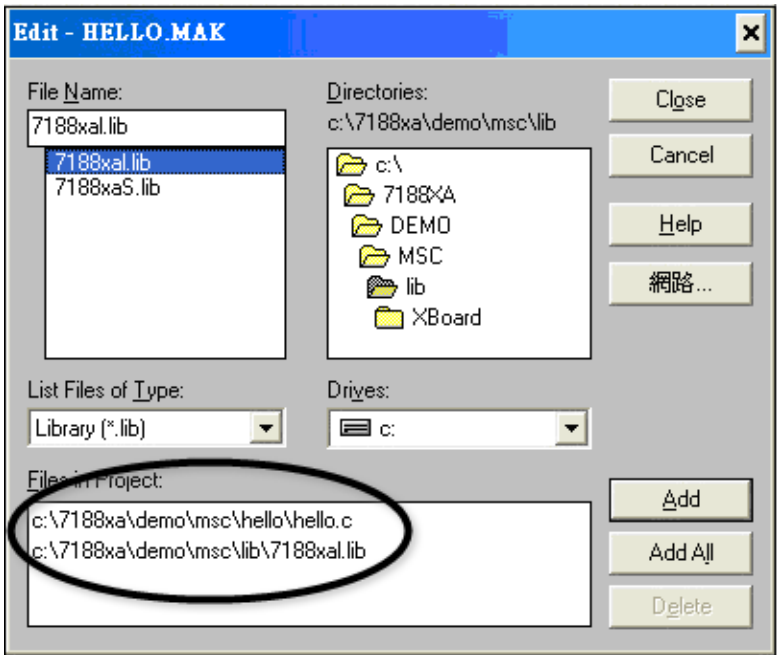
Step 1: Run MSVC.exe



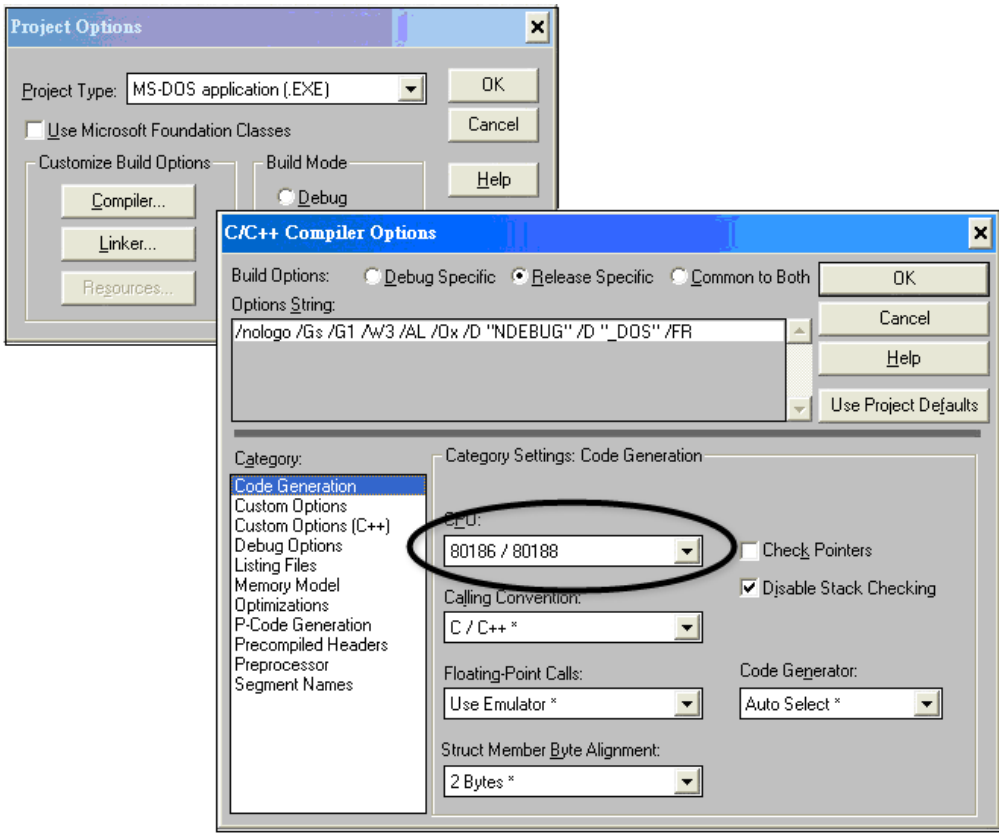
Step 2: Create a new project (*.mak) by entering the name of the project in the Project Name field and then select MS-DOS application (EXE) as the Project type



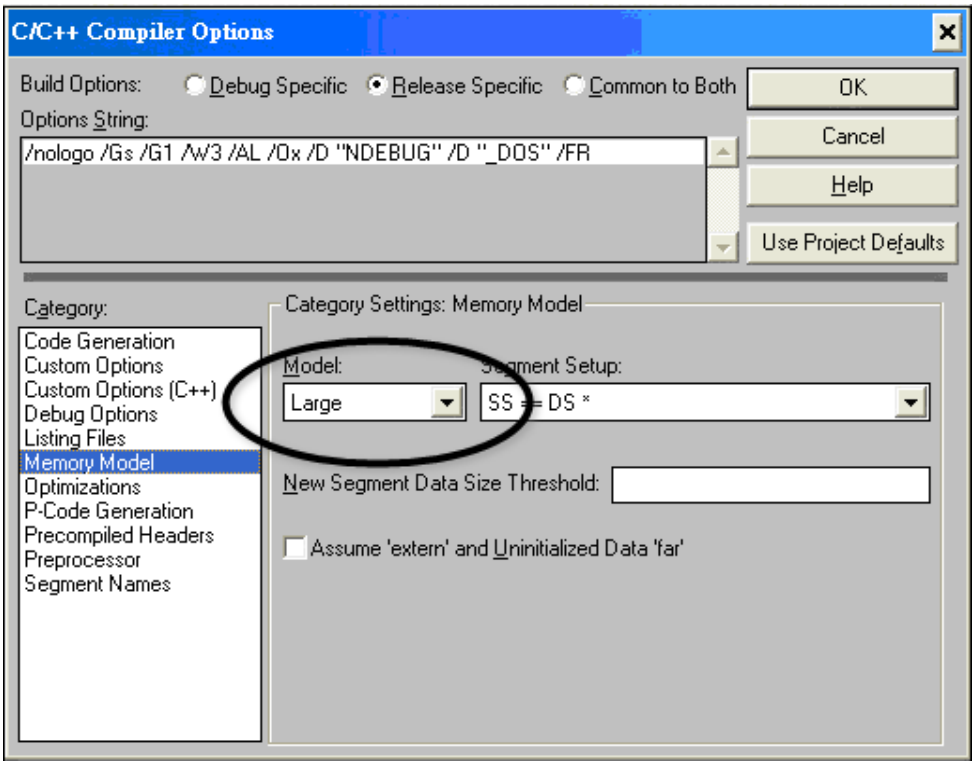
Step 3: Add the user's program and the necessary library files to the project



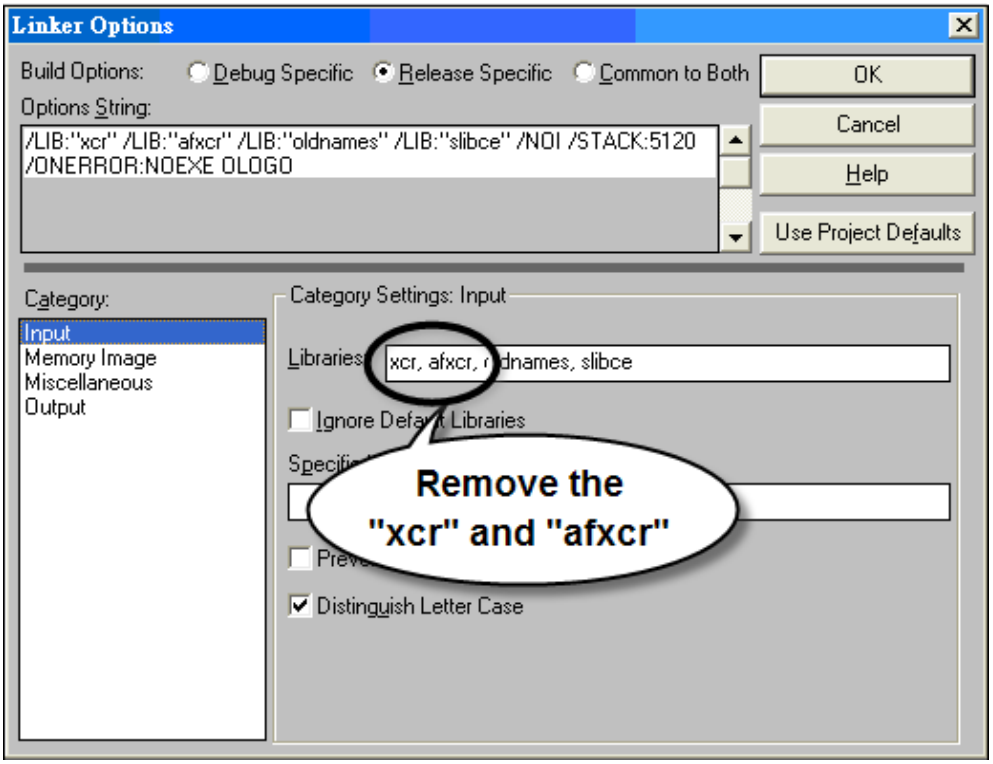
Step 4: Set the Code Generation on the Compiler.



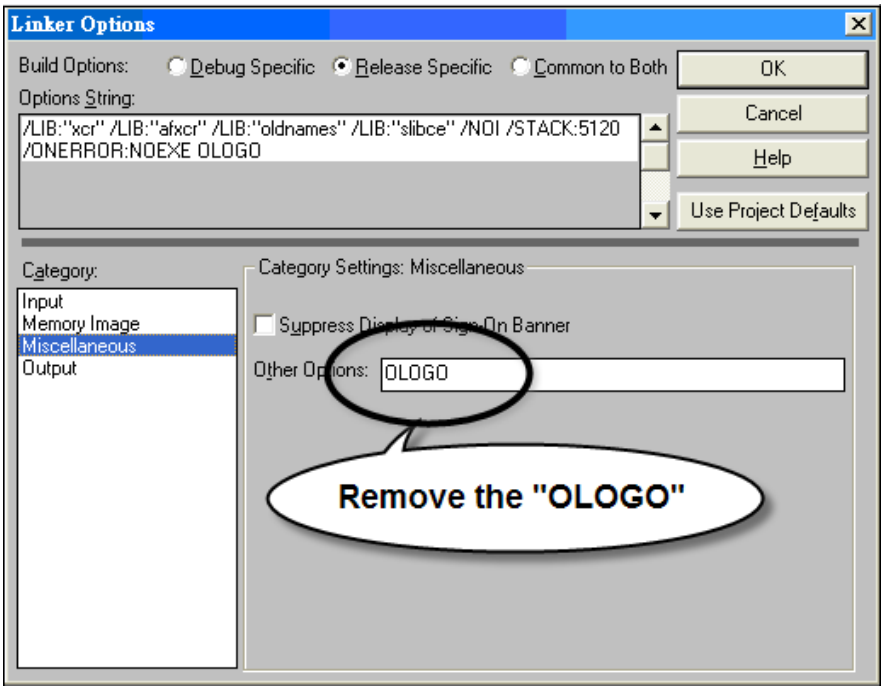
Step 5: Change the Memory model (large for 8000e.lib)



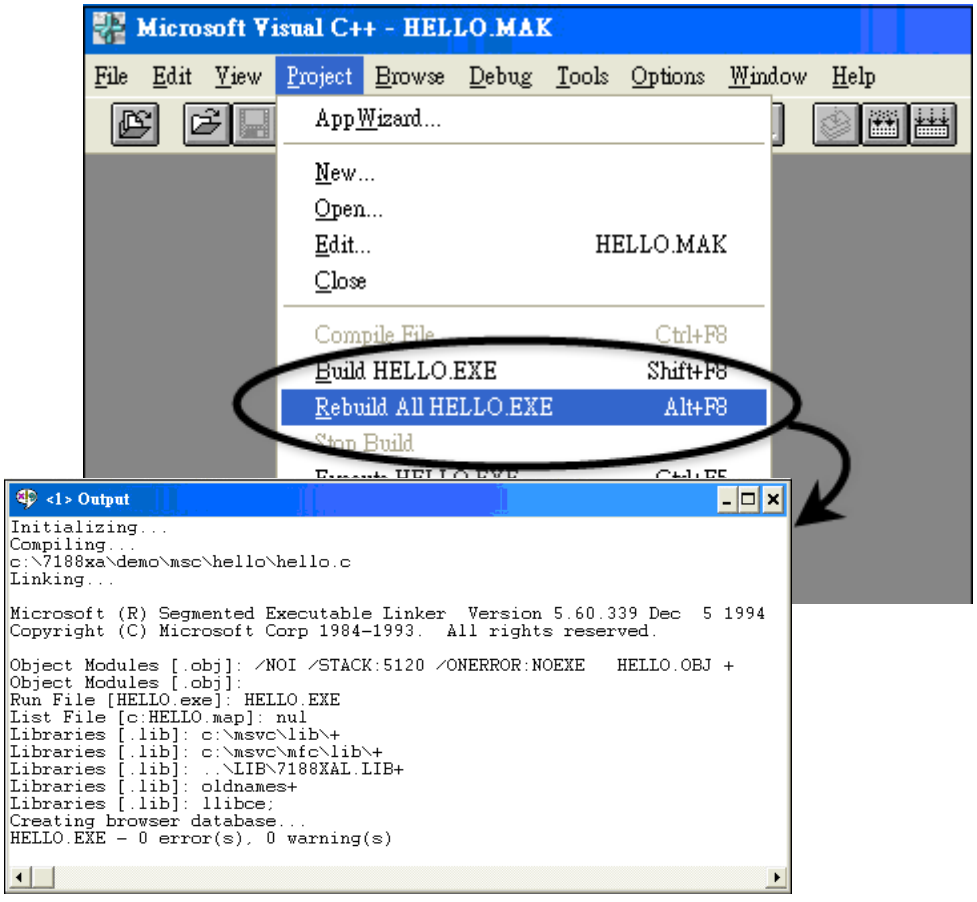
Step 6: Remove the xcr, afxcr library from the Input Category



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



Step 8: Rebuild the project



Appendix H. Application of RS-485 Network

The RS-485 length can be up to 4000 ft or 1.2 km over a single set of twisted–pair cables, if the RS-485 network is over 4000 ft or 1.2Km, the RS-485 repeater must be added to extend the RS-485 network.

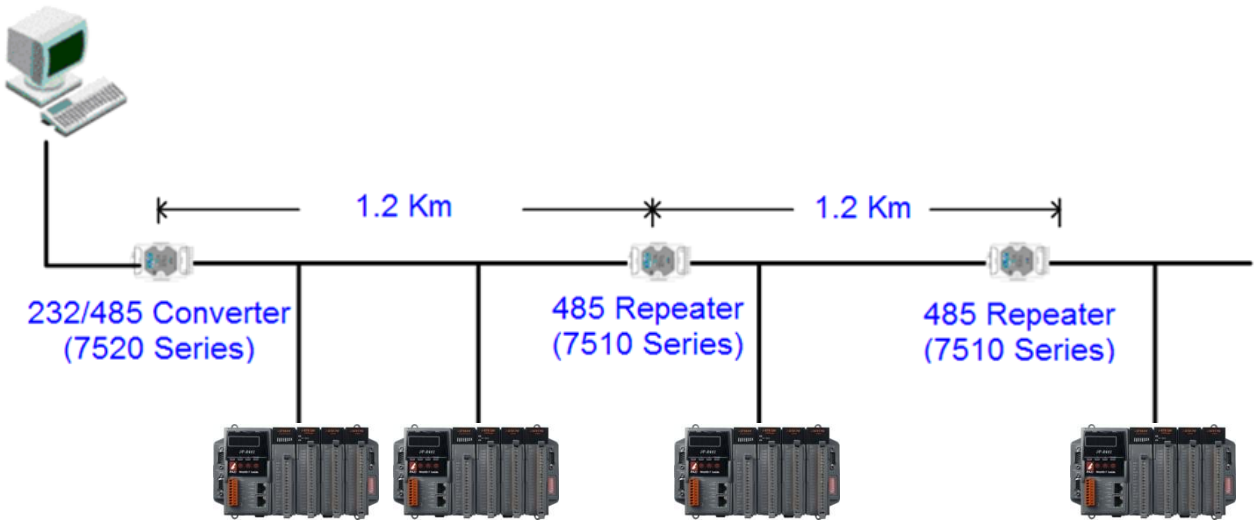
H.1. Basic RS-485 network

The basic component of the RS-485 network consist of a Master Controller (or using a PC as a host controller), and some RS-485 devices.

H.2. Daisy chain RS-485 network

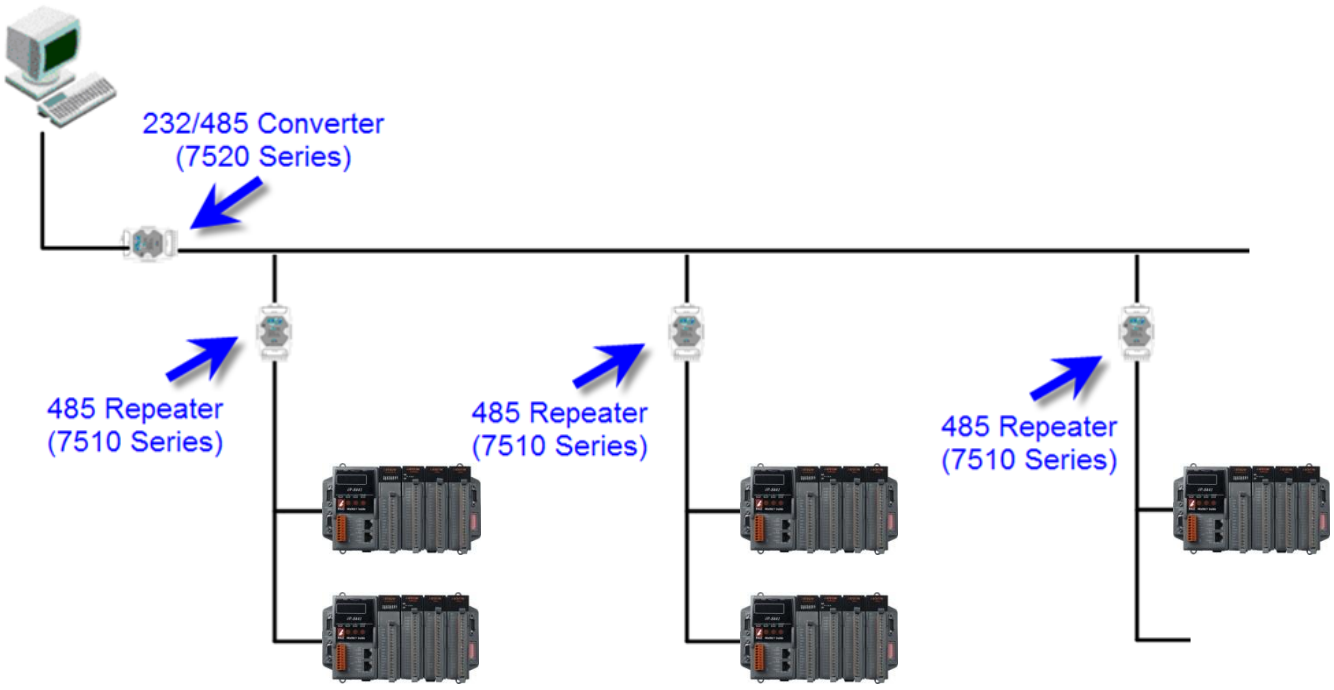
There are branches along the main network. In this case, it is better to have a repeater to isolate or filter the noise that is made by devices.

There is a better choice to use 7513 as a RS-485 hub on start type network.

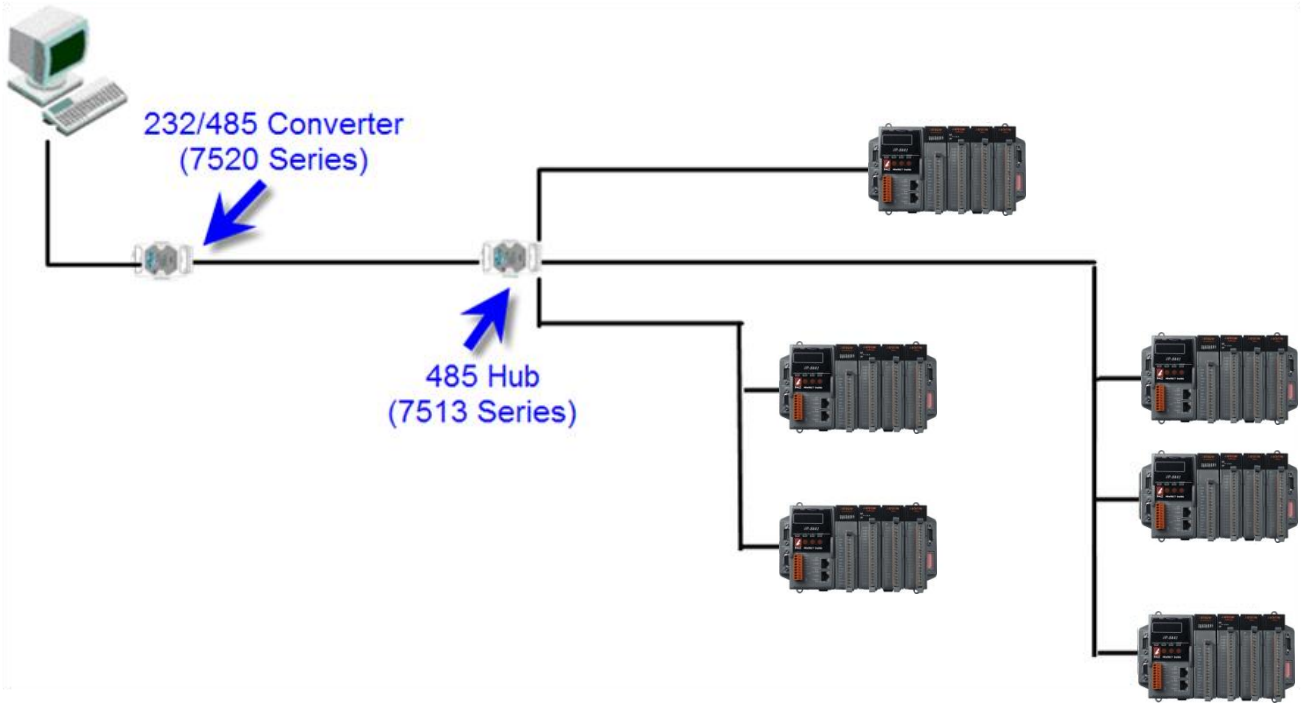


H.3. Star type RS-485 network

All RS-485 devices are wired directly to the main network, If the network is up to 1.2 Km, it will need a repeater (7510 series) to extend the network length.

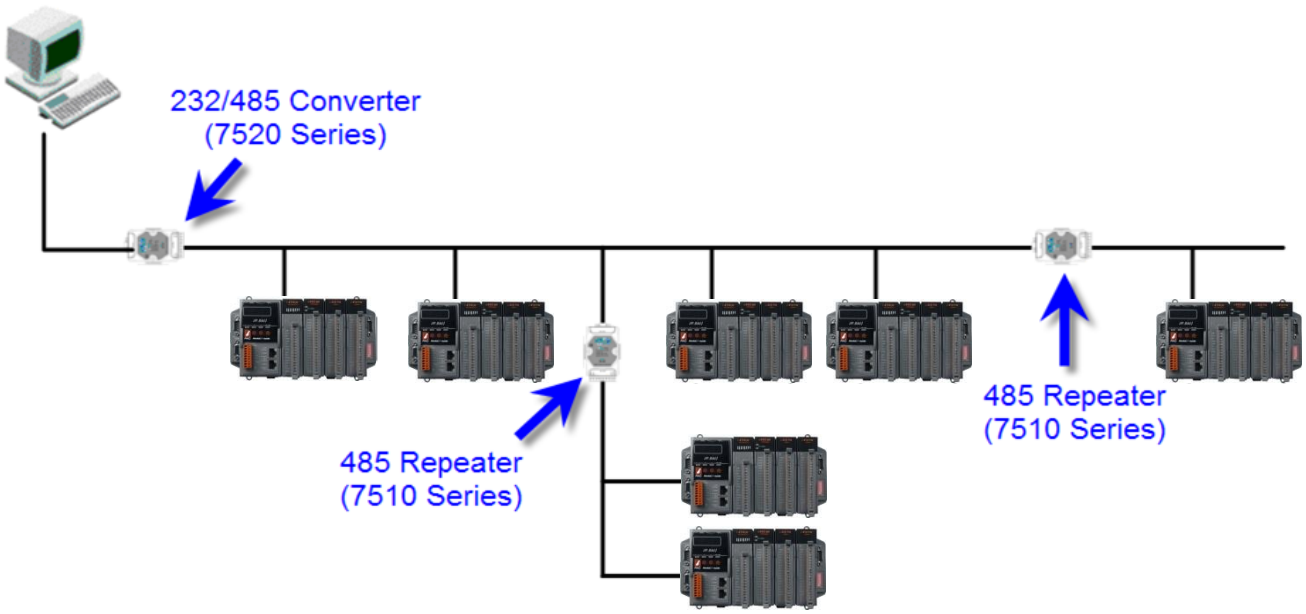


There is a better choice to use 7513 as a RS-485 hub on start type network.



H.4. Random RS-485 network

There are branches along the main wire. In this case, it is better to have a repeater to isolate or filter the noise that is made by devices.

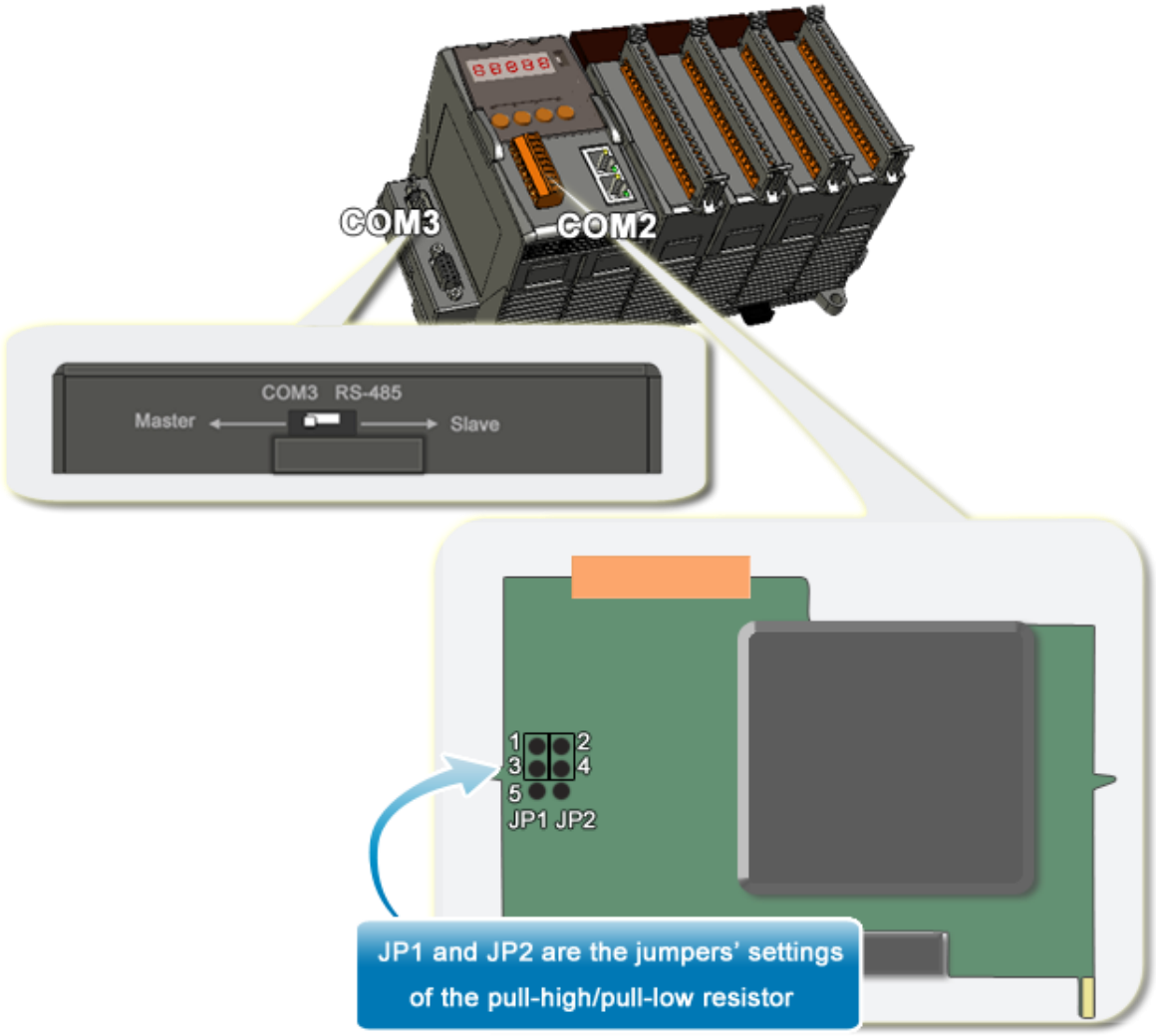


H.5. Pull-High/Pull-Low Resistors

The RS-485 network based on master-slave architecture consist of a single master device and one or more slave devices.

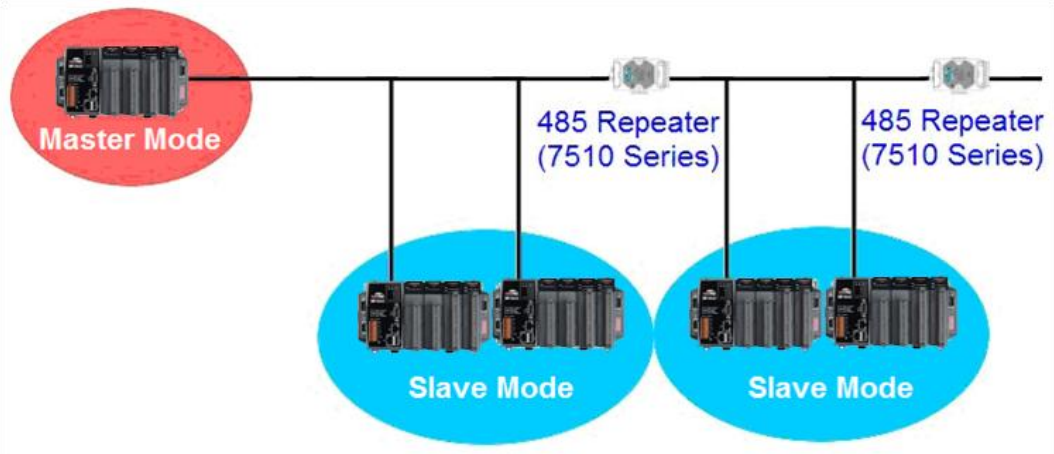
The iPAC-8000 provides two RS-485 communication interface based on the master-slave system architecture, all of which have a pull-high/pull-low resistor, user can set it to master or slave for implementing an RS-485 multi-drop network.

One of the RS-485 communication, COM2, its pull-high/pull-low resistor located on power board, the other, COM3, located on the right side and its pull-high/pull-low resistor located on the bottom of the right side, as shown below.

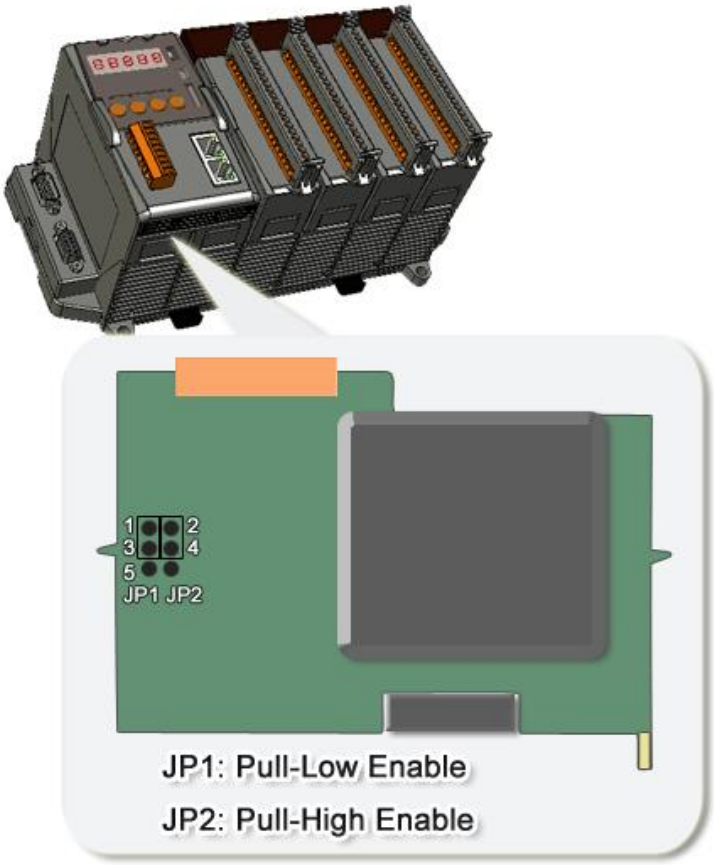


H.5.1. iPAC-8000 as a Master

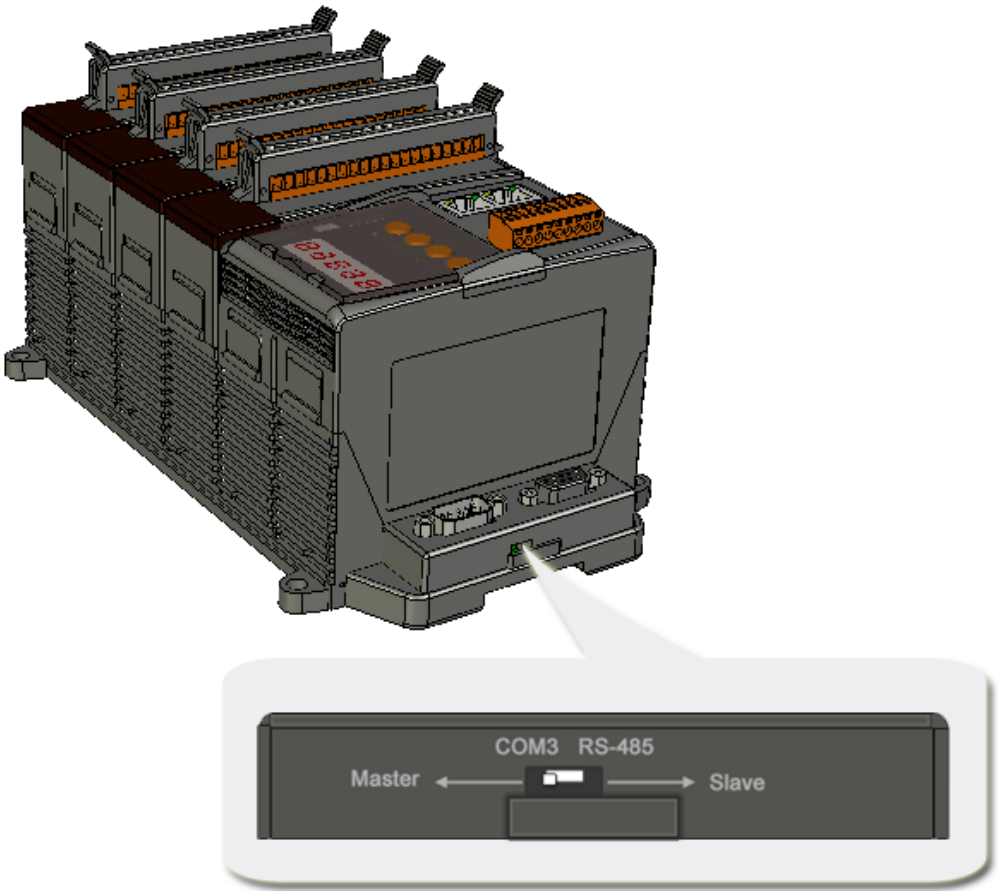
When one of iPAC-8000 is set to master, then all the other devices on the same network must be slave mode. If the network is up to 1.2 KM, it will need a repeater (7510 series) to extend the network length.



When iPAC-8000 as a master using COM2 communication interface, the pull-high /pull-low resistor located on the power board must adjust to enabled as shown below.



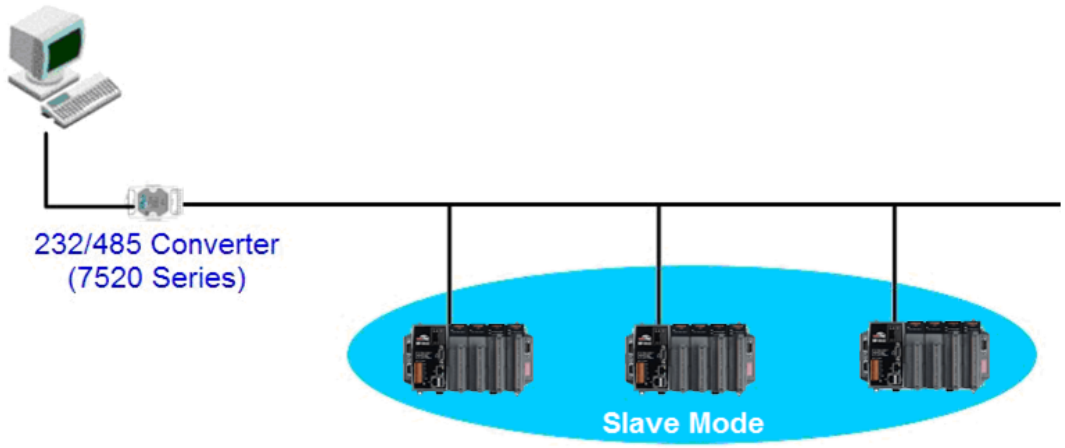
When iPAC-8000 as a master using COM3 communication interface, the pull-high /pull-low resistor located on the power board must set to “Master” as shown below.



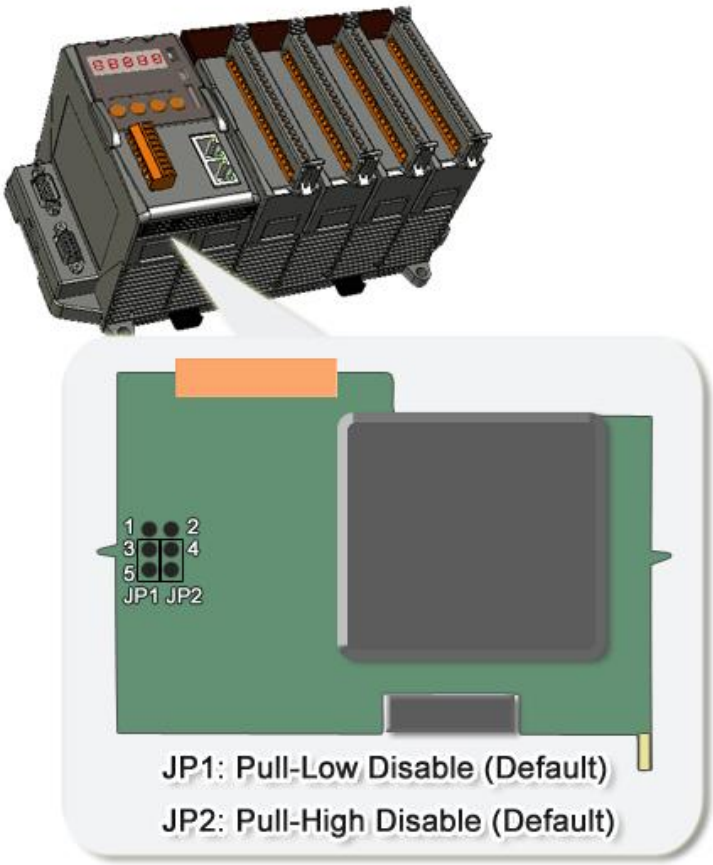
H.5.2. iPAC-8000 as a slave

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

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



When iPAC-8000 as a master using COM2 communication interface, the pull-high /pull-low resistor located on the power board must adjust to disabled as shown below.



When iPAC-8000 as a master using COM3 communication interface, the pull-high /pull-low resistor located on the power board must set to “Slave” as shown below.