Quick Start

ICPDAS LinCon-8000 SDK

Implement industry control with Linux Technique
(Version 2.1)

Warranty

All products manufactured by ICPDAS Inc. are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICPDAS Inc. assume no liability for damages consequent to the use of this product. ICPDAS Inc. reserves the right to change this manual at any time without notice. The information furnished by ICPDAS Inc. is believed to be accurate and reliable. However, no responsibility is assumed by ICPDAS Inc. for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Trademark

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

License

The user can use, modify and backup this software **on a single machine.** The user may not reproduce, transfer or distribute this software, or any copy, in whole or in part.

Contents

1. Introduction

Recently, Linux has been adopted widely by many users because of the properties of stability, open source, and free charge. In the mean while, the development of linux is supported by many large international companies and the function in linux is not inferior to Windows so that linux OS is more and more popular and accepted. In the other hand, the hardware requirement that linux OS can works in embedded system smoothly is not high, just only 386 CPU or better and 8 MB RAM. Therefore except Win CE of Microsoft, Linux has been already another good choice in embedded OS.

The Linux OS demands less system resources from the embedded controller and is therefore the best fit for it because of the embedded controller has some limitations in system resources. It is for this reason that the LinCon-8000 embbeded controller has been published to be a new generation product from ICPDAS and the Embedded-Linux OS has been adopted into the LinCon-8000. The LinCon-8000's main purpose is to allow the numerous enthusiastic linux users to control their own embedded systems easily within the Linux Environment.

ICPDAS provides the library file – libi8k.a which includes all the functions from the I-7000/8000/87000 series modules which are used in the LinCon-8000 Embedded Controller. The libi8k.a is desiged specially for the I-7000/8000/87000 series modules on the Linux platform for use in the LinCon-8000. Users can easily develop applications in the LinCon-8000 by using either C or Java Language and the .NET applications will also be supported In the future. The various functions of the libi8k.a are divided into the sub-group functions for ease of use within the different applications. The powerful functions of the LinCon-8000 embedded controller are depicted in figure 1-1, which includes a **VGA**, **USB(Card Reader, Camera …)**, **Mouse**, **Keyboard**, **Compact flash card**, **Series ports(RS-232, RS-485), Ethernet(Hub…) and many I/O slots** in the picture. Presently, HTTP、FTP、Telnet、SSH、SFTP Servers are built in and users can transfer files or use remote control with the LinCon-8000 more conveniently. In network communication, **wireless**、**Bluetooth** transfer and **Modem**、**GPRS**、**ADSL**、**Firewall** are also supported. Fig. 1-2 illustrates the outline of the LinCon-8000 with modules. Fig. 1-3 illustrates hardware architecture of the LinCon-8000.

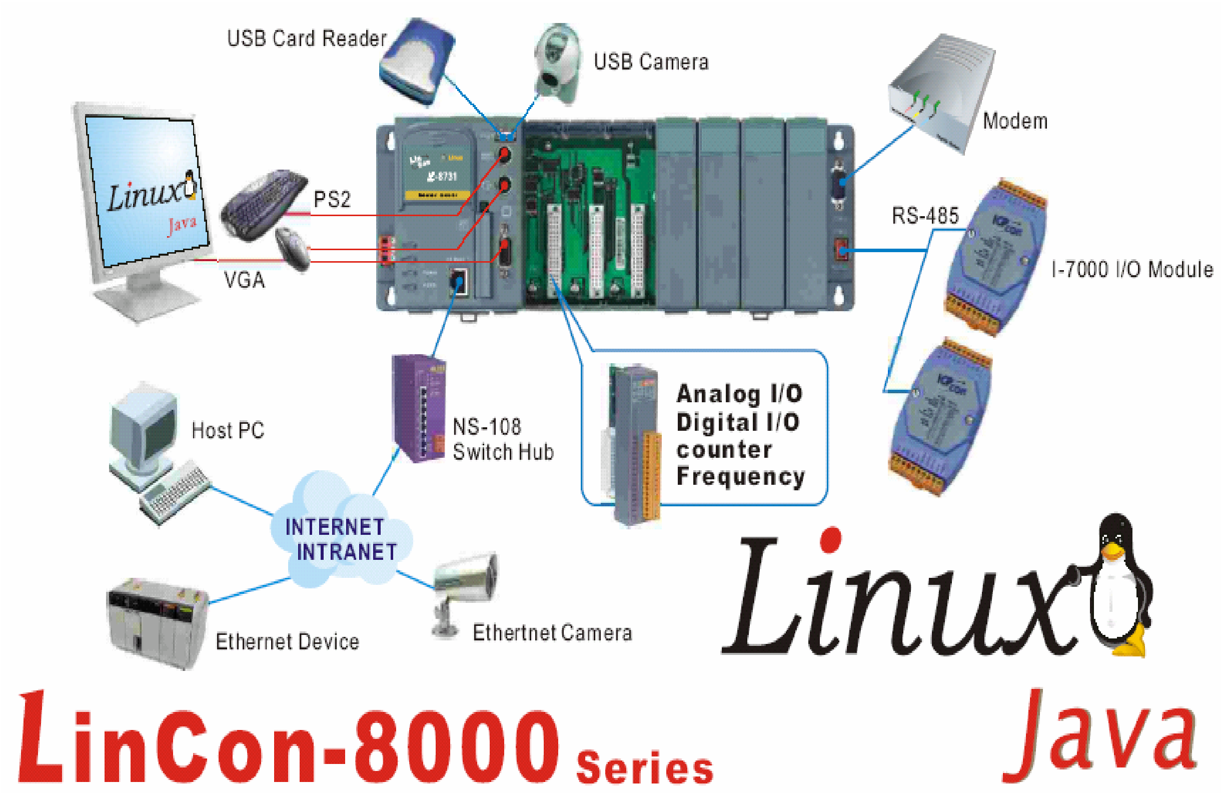

USB Card Reader
USB Camera
Modem
PS2
Linux
Java
VGA
RS-485
I-7000 I/O Module
Host PC
NS-108
Switch Hub
Analog I/O
Digital I/O
counter
Frequency
INTERNET
INTRANET
Ethernet Device
Ethertnet Camera
Linux
LinCon-8000 Series
Java

Fig. 1-1

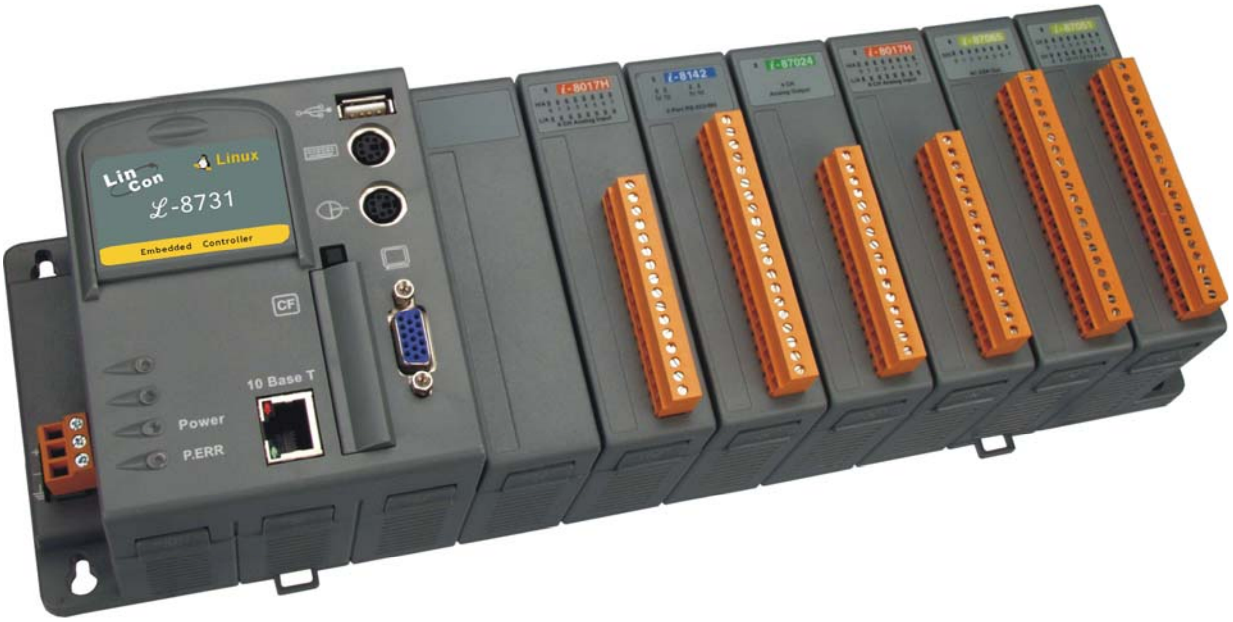

LinCon
Linux
L-8731
Embedded Controller
CF
10 Base T
Power
P.ERR

Fig. 1-2

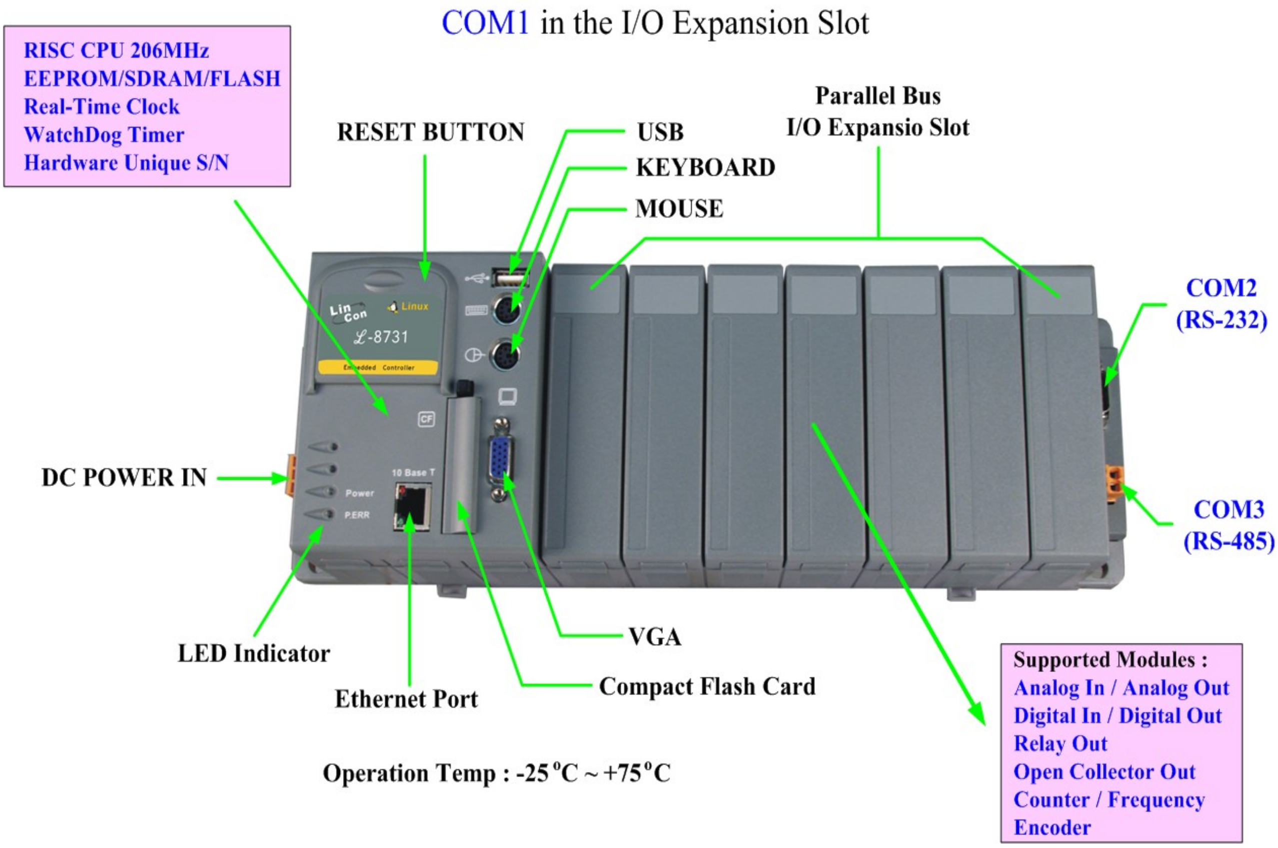

COM1 in the I/O Expansion Slot
RISC CPU 206MHz
EEPROM/SDRAM/FLASH
Real-Time Clock
WatchDog Timer
Hardware Unique S/N
RESET BUTTON
USB
KEYBOARD
MOUSE
Parallel Bus
I/O Expansio Slot
COM2
(RS-232)
DC POWER IN
COM3
(RS-485)
VGA
LED Indicator
Ethernet Port
Compact Flash Card
Operation Temp : -25°C ~ +75°C
Supported Modules :
Analog In / Analog Out
Digital In / Digital Out
Relay Out
Open Collector Out
Counter / Frequency
Encoder
LinCon
Linux
L-8731
Embedded Controller
CF
10 Base T
Power
P.ERR

Fig. 1-3

2. Installation of LinCon-8000 SDK

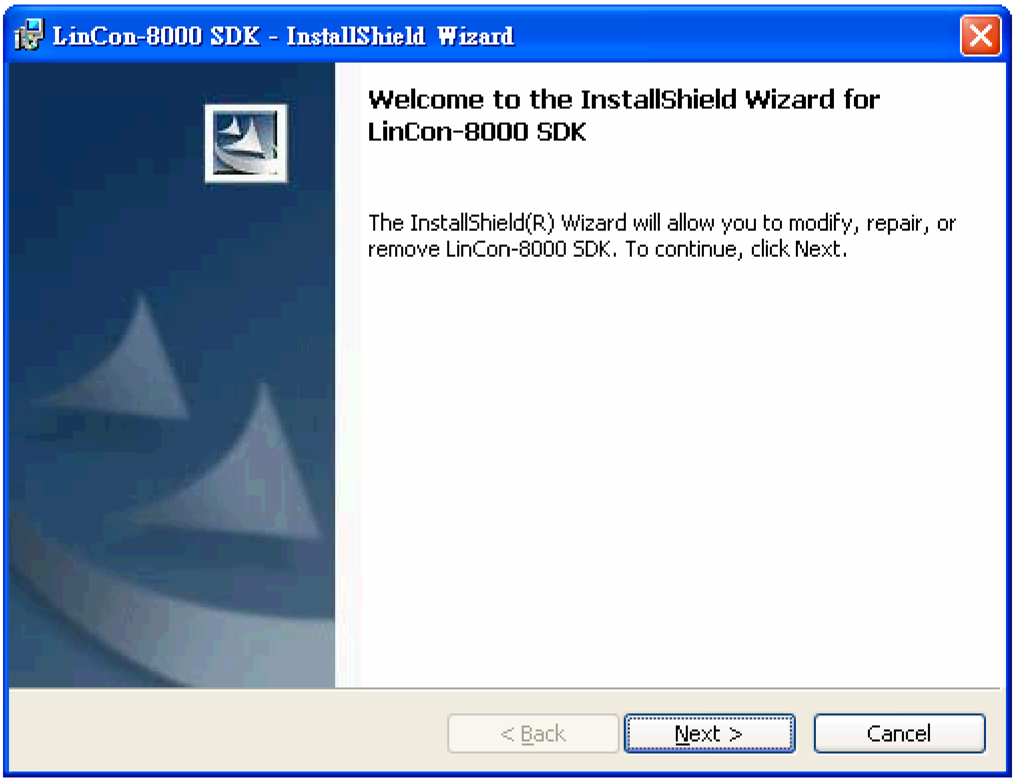
"LinCon-8000 SDK" consists of the following major items.

- LinConSDK library files
- LinConSDK include files
- Demo files
- GNU ToolChain

From ftp://ftp.icpdas.com/pub/cd/linconcd/napdos/linux/sdk/, you can download the latest version of LinCon-8000 SDK and the Manual. And then follows the below steps in order to install the development toolkit which has been provided by ICP DAS for the easy application of the LinCon-8000 embedded controller platform.

2.1 Quick Installation of LinCon-8000 SDK

1. Please insert the installation CD into your CD-ROM driver.
2. Run the "linconsdk_for_windows.exe" file under the folder \napdos\linux\SDK\. Then click on the "Next" button, refer to Fig. 2-1.



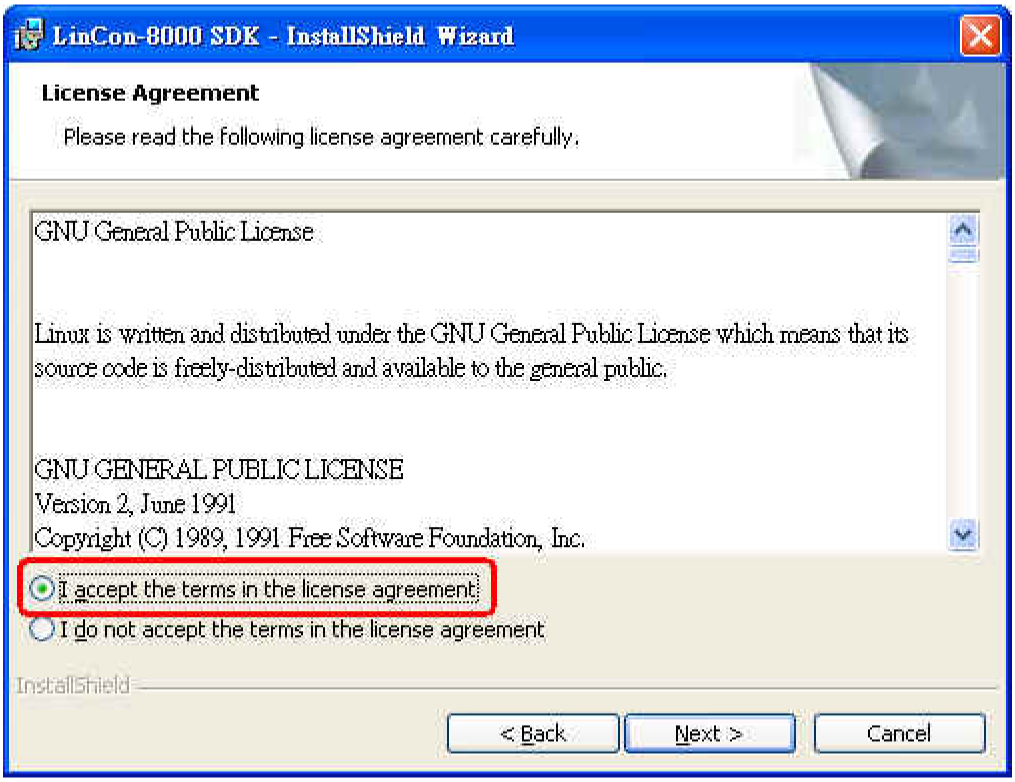
LinCon-8000 SDK - InstallShield Wizard

Welcome to the InstallShield Wizard for LinCon-8000 SDK

The InstallShield(R) Wizard will allow you to modify, repair, or remove LinCon-8000 SDK. To continue, click Next.

Fig. 2 -1

3. Choose the option of "I accept the terms in the license agreement" and click the "next" button, refer to Fig. 2-2 below.



LinCon-8000 SDK - InstallShield Wizard

License Agreement

Please read the following license agreement carefully.

GNU General Public License

Linux is written and distributed under the GNU General Public License which means that its source code is freely-distributed and available to the general public.

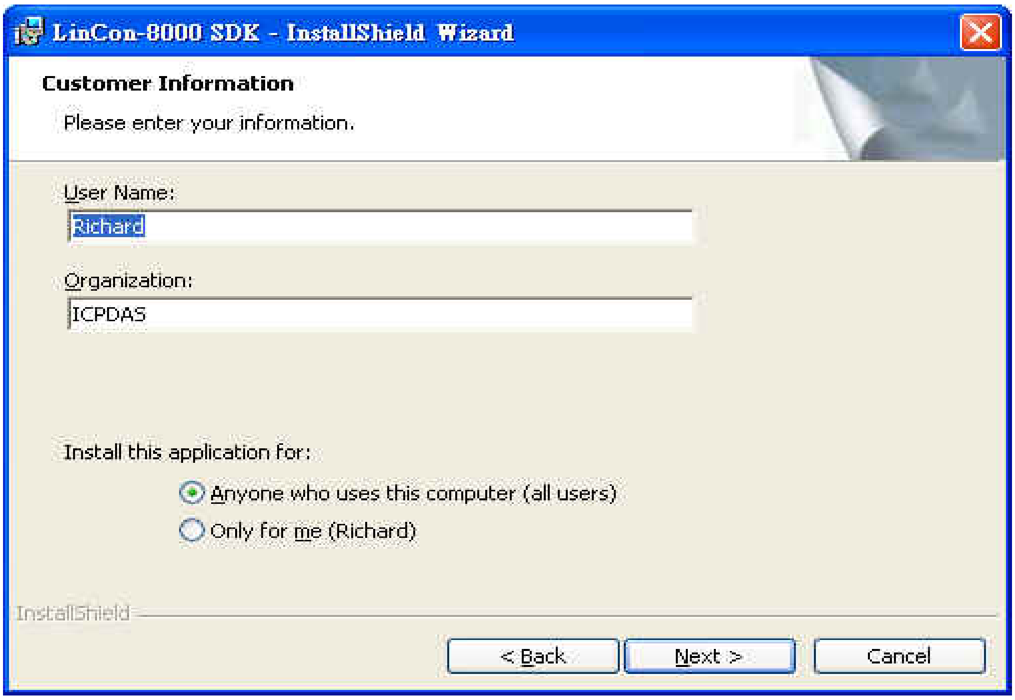
GNU GENERAL PUBLIC LICENSE
Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.

● I accept the terms in the license agreement

○ I do not accept the terms in the license agreement

Fig. 2-2

4. Input your user name and your organization's name, then to click on the "Next" buttion, refer to Fig 2-3.



LinCon-8000 SDK - InstallShield Wizard

Customer Information

Please enter your information.

User Name: Richard

Organization: ICPDAS

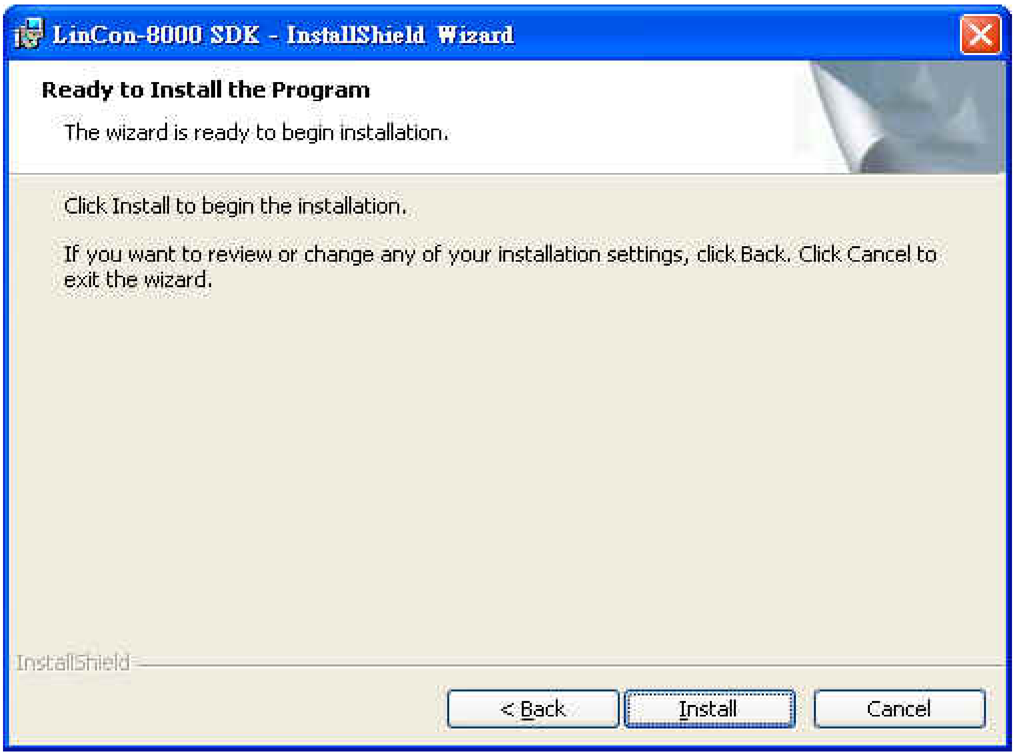
Install this application for:

● Anyone who uses this computer (all users)

○ Only for me (Richard)

Fig. 2-3

5. Then click on the "Install" button to install the LinCon-8000 SDK, refer to Fig 2-4.


LinCon-8000 SDK - InstallShield Wizard

Ready to Install the Program

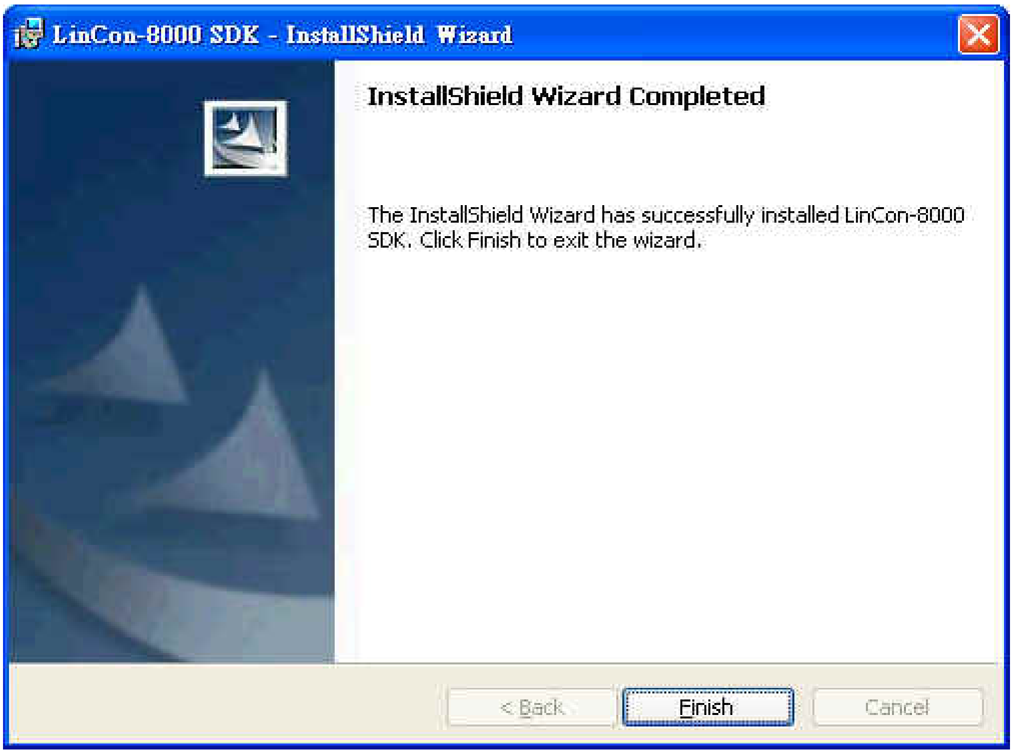
The wizard is ready to begin installation.

Click Install to begin the installation.

If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

Fig. 2-4

6. After successfully installing the software, please click on the "Finish" button to finish the development toolkit installation, refer to Fig. 2-5


LinCon-8000 SDK - InstallShield Wizard

InstallShield Wizard Completed

The InstallShield Wizard has successfully installed LinCon-8000 SDK. Click Finish to exit the wizard.

Fig. 2-5

7. Open the "**C:\cygwin\LinCon8k**" folder and see the content. Refer to Fig 2-6.

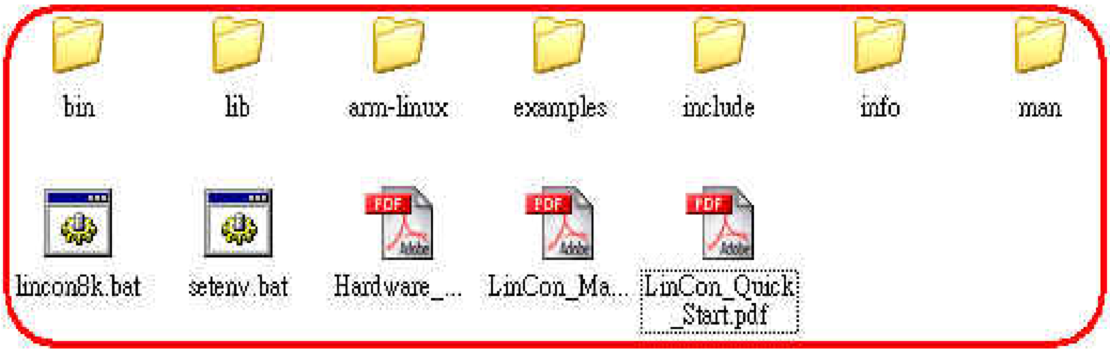
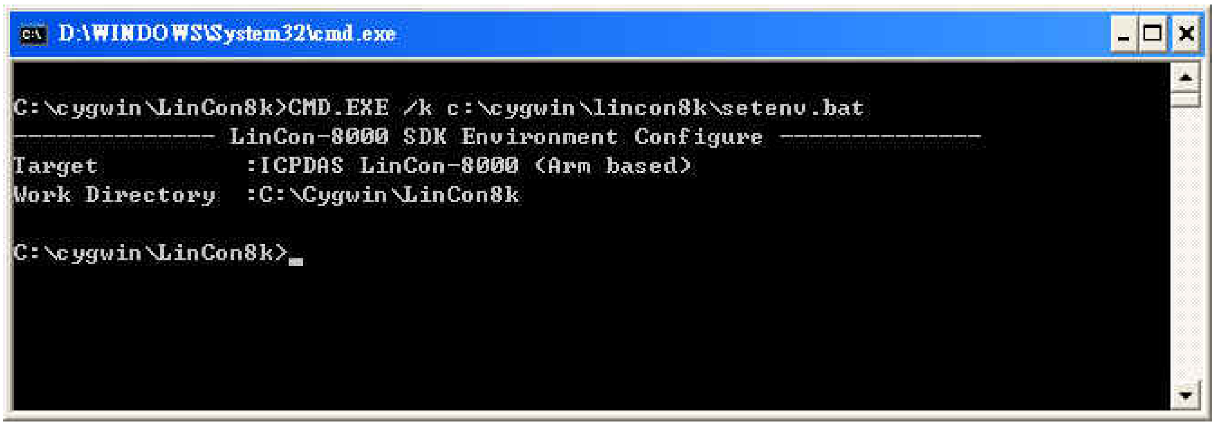


Fig. 2-6

8. Start using the "LinCon-8000 Build Environment" by double clicking the shortcut for the "**LinCon-8000 Build Environment**" on the desktop or by clicking through " Start ">" Programs ">" ICPDAS ">" LinCon-8000 SDK ">" LinCon-8000 Build Environment " icon. Then a special DOSBOX will be displayed in which we can compile applications for the LinCon-8000. refer to Fig. 2-7.



```
C:\cygwin\LinCon8k>CMD.EXE /k c:\cygwin\lincon8k\setenv.bat
--------------- LinCon-8000 SDK Environment Configure ---------------
Target           :ICPDAS LinCon-8000 (Arm based)
Work Directory   :C:\Cygwin\LinCon8k

C:\cygwin\LinCon8k>_
```

Fig. 2-7

Once your Installation is complete, you can find the files for the library and demo in the following paths.

The Libi8k.a path is "**C:\cygwin\LinCon8k\lib**".

The include files path is "**C:\cygwin\LinCon8k\include**"

The demo path is "**C:\cygwin\LinCon8k\examples**".

3. The Architecture of LIBI8K.A in the LinCon-8000

The **libi8k.a** is a library file that is designed for I7000/8000/87000 applications running in the Lincon-8000 Embedded Controller using the Linux OS. Users can apply it to develop their own applications **with C language**. In order to assist users, we provide many demo programs. Based on the demo programs, users can easily understand how to use these functions and develop their own applications within a short period of time.

The relationships among the libi8k.a and user's applications are depicted as Fig. 3-1：

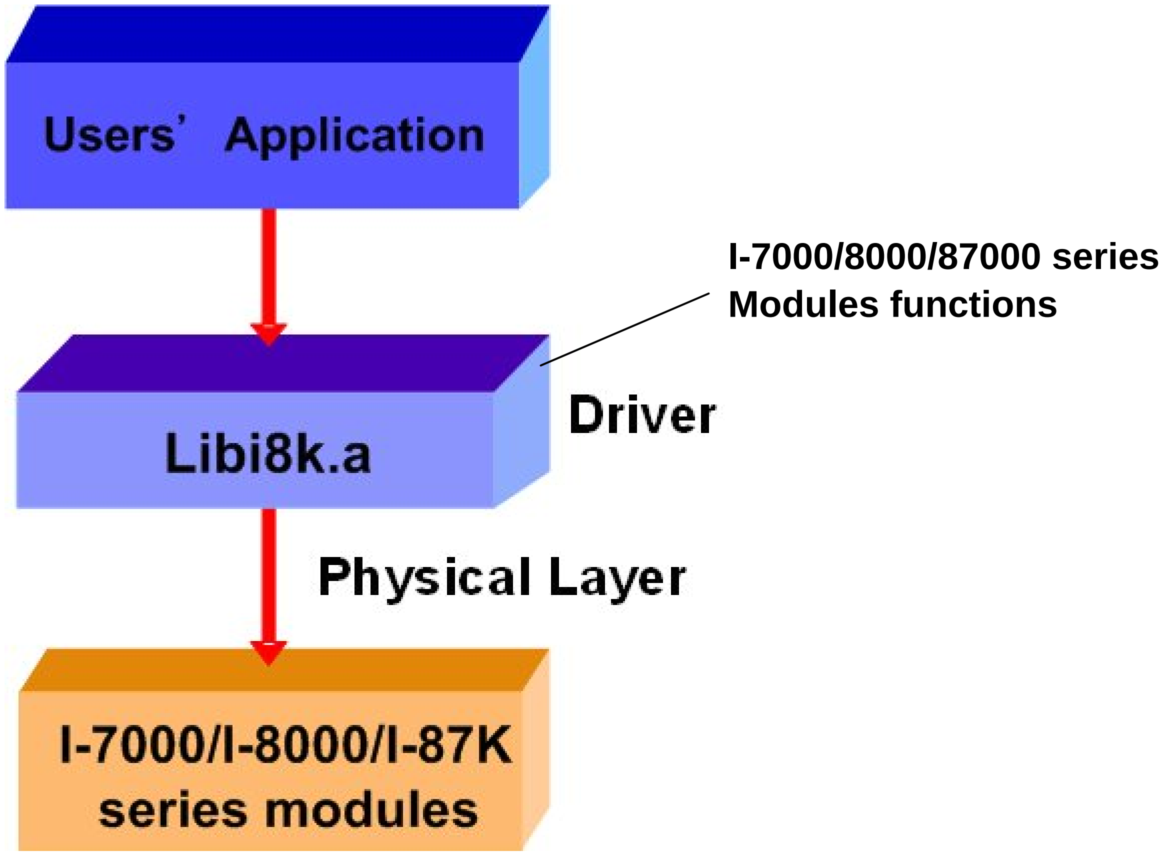


Fig. 3-1

Functions for Lincon-8000 Embedded Controller are divided into sub-groups for ease of use within the different applications：

1. System Information Functions
2. Digital Input/Output Functions
3. Watch Dog Timer Functions
4. EEPROM Read/Write Functions
5. Analog Input Functions
6. Analog Output Functions
7. 3-axis Encoder Functions
8. 2-axis Stepper/Servo Functions

4. LinCon-8000 System Settings

In this section, we will introduce how to setup the LinCon-8000 configuration. Let users can use the LinCon-8000 more easily.

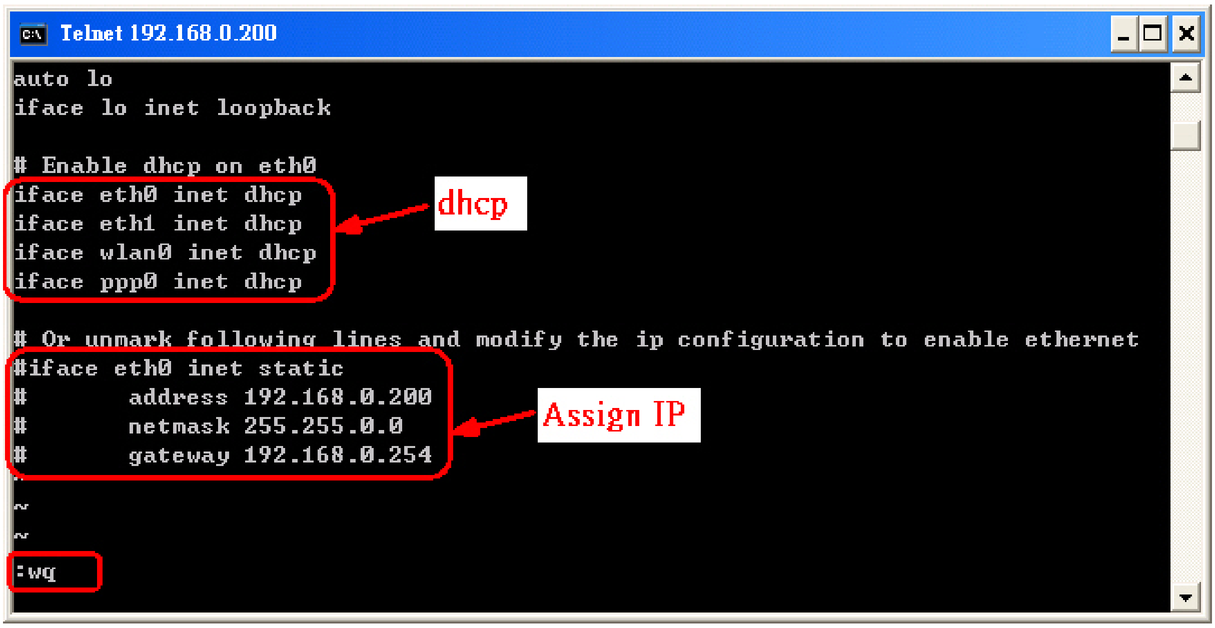
4.1 Settings for the LinCon-8000 Network

The LinCon-8000 network setting includes two ways. One is **DHCP** and the other is “**Assigned IP**”. DHCP is the default setting after the LinCon-8000 is produced and this way is easy for users. However, if your network system is without DHCP server, then users need to configure the network setting by using “Assigned IP”.

4.1.1 Setting the IP、Netmask and Gateway

(1) Using DHCP :

Boot up LinCon-8000 and click the **“ start/xterm ”** to open a **“ command Prompt ”**. Type in **“ vi /etc/network/interfaces ”** to open the network setting file. Remove **“ # ”** in the dhcp block and add **“ # ”** in the Assign IP block. Then type **“ :wq ”** to save the setting. Type **“ ifup eth0 ”** to make the setting work. (Refer to the Fig 4-1)

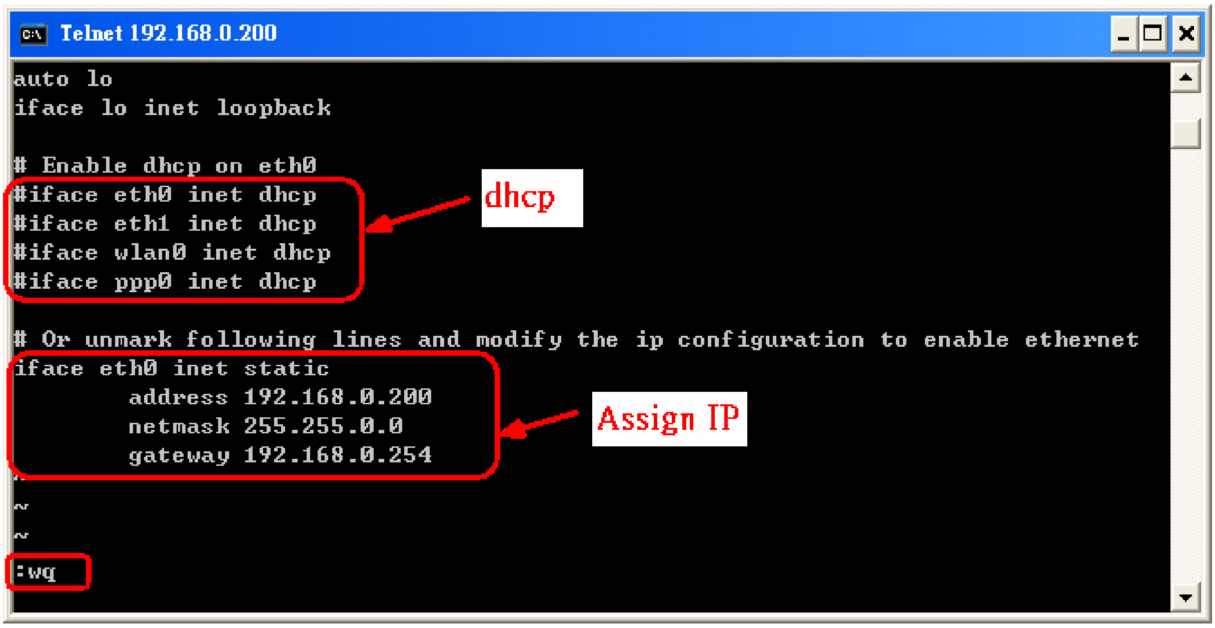


```
auto lo
iface lo inet loopback

# Enable dhcp on eth0
iface eth0 inet dhcp
iface eth1 inet dhcp
iface wlan0 inet dhcp
iface ppp0 inet dhcp

# Or unmark following lines and modify the ip configuration to enable ethernet
#iface eth0 inet static
#       address 192.168.0.200
#       netmask 255.255.0.0
#       gateway 192.168.0.254
~
~
~
:wq
```

Fig 4-1

(2) Using “Assigned IP” :

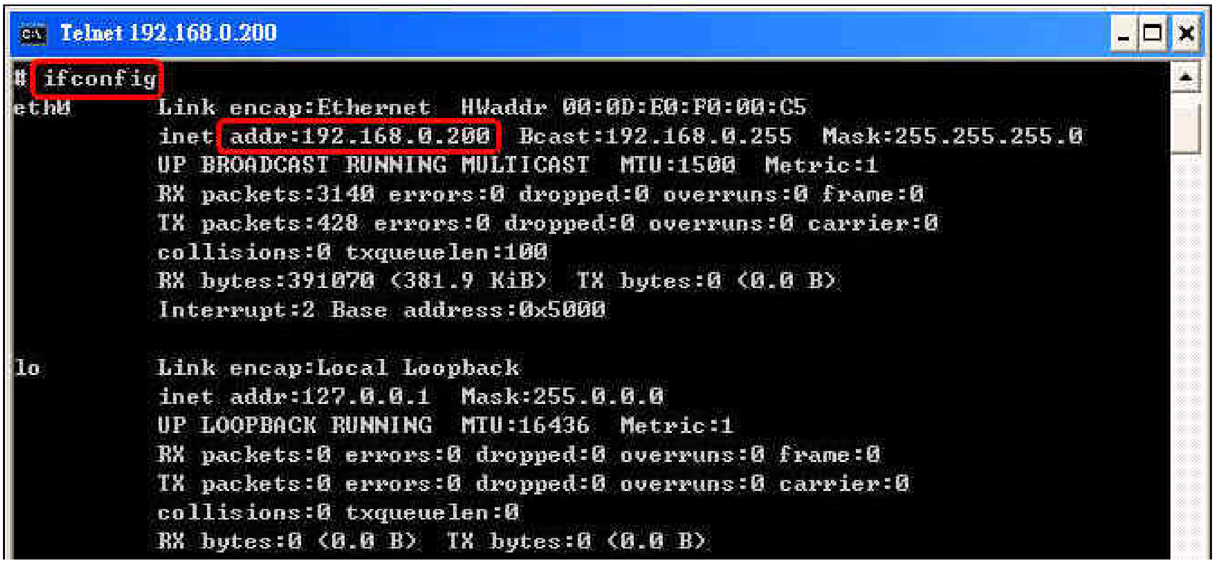
Boot up LinCon-8000 and click the “ **start/xterm** ” to open a “command line”. Type in “ **vi /etc/network/interfaces** ” to open the network setting file. Remove “ # ” in the Assign IP block and add “ # ” in the dhcp block. Type ip、netmask and gateway you want in the Assign IP block. Then type “ **:wq** ” to save the setting. Type “ **ifup eth0** “ to make the setting work. (Refer to the Fig 4-2)



```
auto lo
iface lo inet loopback

# Enable dhcp on eth0
#iface eth0 inet dhcp
#iface eth1 inet dhcp
#iface wlan0 inet dhcp
#iface ppp0 inet dhcp

# Or unmark following lines and modify the ip configuration to enable ethernet
iface eth0 inet static
        address 192.168.0.200
        netmask 255.255.0.0
        gateway 192.168.0.254

~
~
:wq
```

Fig 4-2

After finish the LinCon network setting, users can type “ **ifconfig** “ to see the network setting. (Refer to the Fig 4-3)

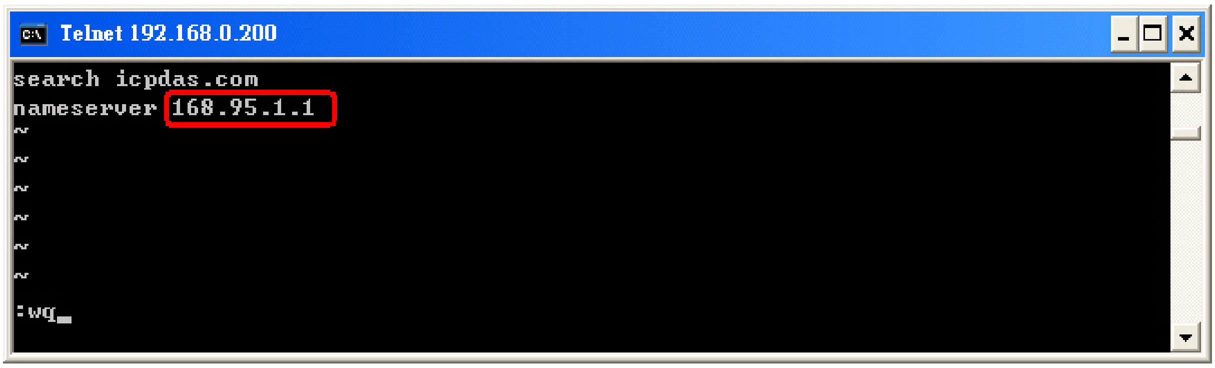


```
# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0D:E0:F0:00:C5
          inet addr:192.168.0.200  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3140 errors:0 dropped:0 overruns:0 frame:0
          TX packets:428 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:391070 (381.9 KiB)  TX bytes:0 (0.0 B)
          Interrupt:2 Base address:0x5000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Fig 4-3

4.1.2 Setting of DNS

Boot up LinCon-8000 and click the **“ start/xterm ”** to open a “command line”. Type in **“ vi /etc/resolv.conf ”** to open the DNS setting file. Type “ DNS server ” in the “ **nameserver** “ field. Then type **“ :wq ”** to save the setting. Type **“ reboot “** to reboot the LinCon-8000 to make the setting work. (Refer to the Fig 4-4)



```
search icpdas.com
nameserver 168.95.1.1
~
~
~
~
~
~
:wq
```

Fig 4-4

4.2 The CF Card Usage

The contents of CF Card in the LinCon-8000 is in the default path of **/mnt/hda**. Therefore, users can access the files of CF Card in the directory.

4.2.1 Mount the CF Card

When you want to use the CF Card, you can insert the CF Card into the Slot of CF Card in the LinCon-8000. (Refer to Fig. 1-3) It will be auto-mounted in the LinCon-8000, and you can access the files of CF Card in the **/mnt/hda** directory.

4.2.2 Umount the CF Card

Before you want to pull out the CF Card from the LinCon-8000, you need to type the “ **umount /mnt/hda** “ command first. Then you can pull out the CF Card safely to prevent the damage to CF Card.

4.3 The USB Device Usage

Before accessing the USB device, users need to mount the USB device to the LinCon-8000. Because it will not auto-mount the USB device in the LinCon-8000.

4.3.1 Mount the USB Device

The steps are as follows :

(1) Change the path to /mnt, and then type " **mkdir usb** " to build a usb directory.

(2) Type " **mount /dev/sda1 /mnt/usb** " to mount the USB device to the usb directory and change the path to /mnt/usb. Then type " **ls** " and you can see the content of USB device.

4.3.2 Umount the USB Device

Before you want to pull out the USB device from the LinCon-8000, you need to type the " **umount /mnt/usb** " command first. Then you can pull out the USB device safely to prevent the damage to USB device.

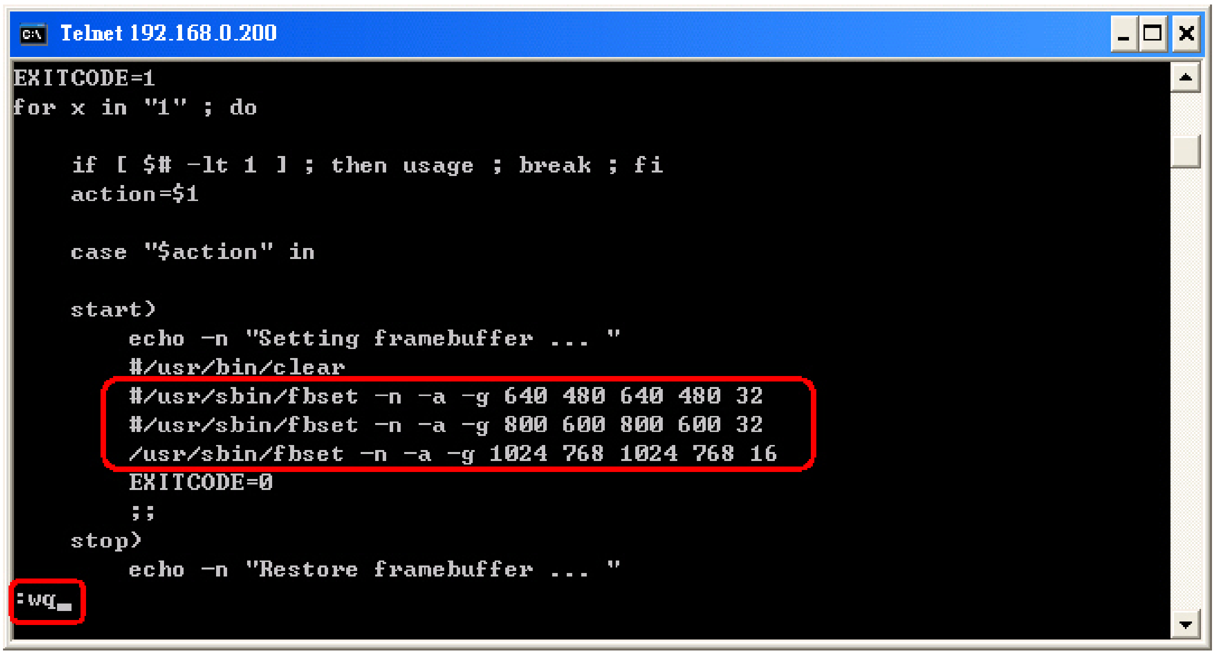
4.4 Adjust VGA Resolution

There are three modes -- **640x480**、**800x600**、**1024x768** supported in the LinCon VGA resolution and the default setting is 800x600. If users want to change the VGA resolution. Please follow below steps :

(1) Type **" vi /etc/init.d/fbman "** to open resolution setting file.

(2) If users want to set the resolution to be 1024x768. First, Add **"#"** in the 800x600 column and then remove **"#"** in the 1024x768 column. Type **" :wq "** to save the setting. (Refer to Fig 4-5)

(3) Type **" Reboot "** to reboot LinCon-8000.



```
EXITCODE=1
for x in "1" ; do

    if [ $# -lt 1 ] ; then usage ; break ; fi
    action=$1

    case "$action" in

    start)
        echo -n "Setting framebuffer ... "
        #/usr/bin/clear
        #/usr/sbin/fbset -n -a -g 640 480 640 480 32
        #/usr/sbin/fbset -n -a -g 800 600 800 600 32
        /usr/sbin/fbset -n -a -g 1024 768 1024 768 16
        EXITCODE=0
        ;;
    stop)
        echo -n "Restore framebuffer ... "
:wq
```

Fig 4-5

4.5 Running applications automatically at boot time

A “run level” determines which programs are executed at system startup. Run level 2 is the default run level of LinCon-8000.

The contents of run level are in the /etc/init.d directory that directory contains the scripts executed at boot time. These scripts are referenced by symbolic links in the /etc/rc2.d.

These links are named S<2-digit-number><original-name>. The numbers determine the order in which the scripts are run, from 00 to 99 — the lower number would earlier executed. Scripts named with an **S** are called with start, and named with a **K or x** are called with stop.

4.5.1 Making program run at boot time

Making program run at boot time, you should create a startup script placed in /etc/init.d directory that runs the required commands for executed automatically at boot time and be symbolically linked to /etc/rc2.d directory.

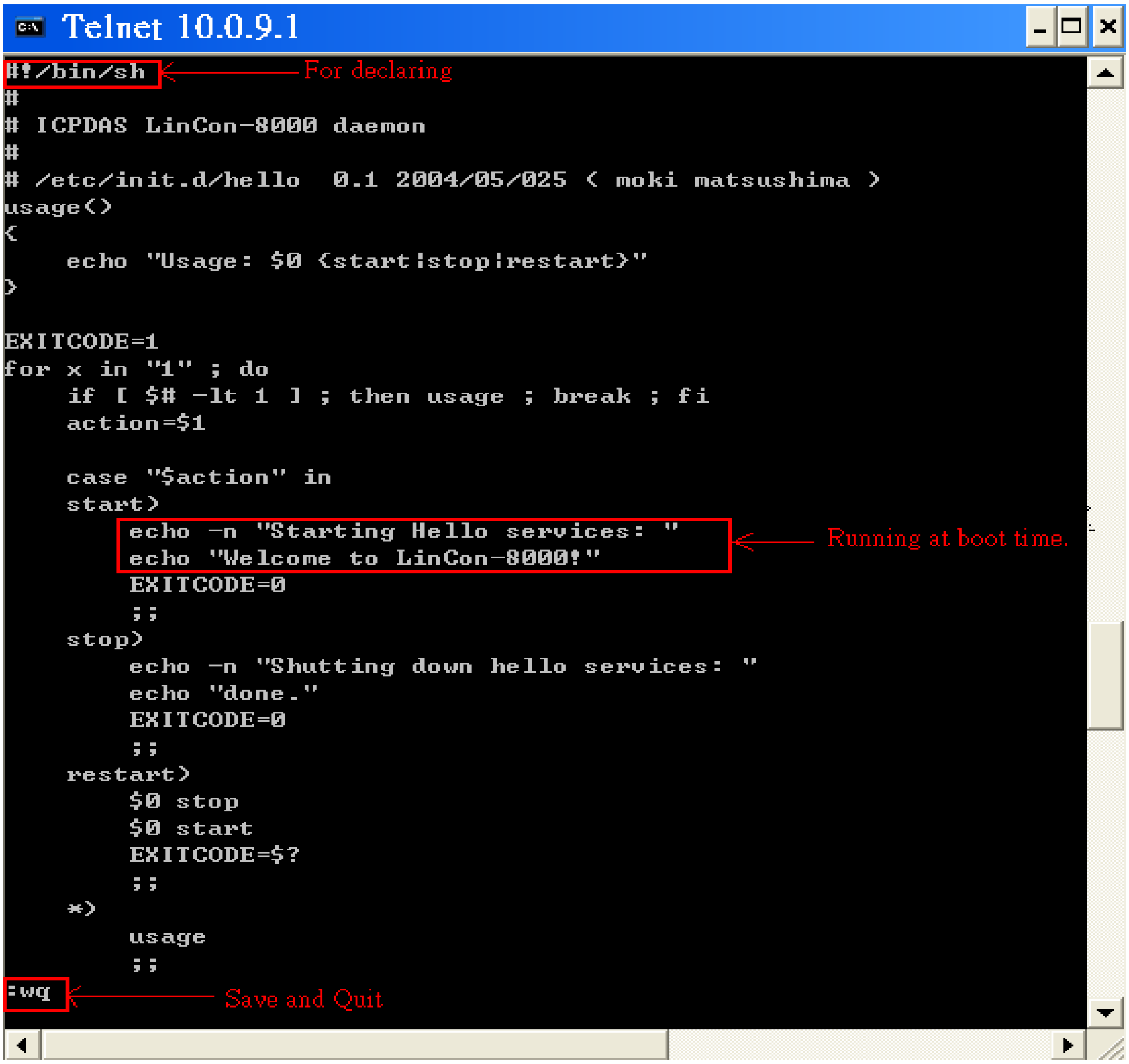
The steps are as follows :

(1) Type “ **vi /etc/init.d/hello** “ to edit a script that would like to executed program, filename is hello. Type “ **:wq** “ to save and quit the script. (Refer to the Fig 4-6)

(2) Type “ **chmod 755 /etc/init.d/hello** “ to change authority.

(3) Type “ **cd /etc/rc2.d** “ to into default run level.

(4) Type ” **ln -s ../init.d/hello /etc/rc2.d/S85hello** “ to make a symbolic link into the script file and it will be executed automatically at boot time. (Refer to the Fig 4-7)



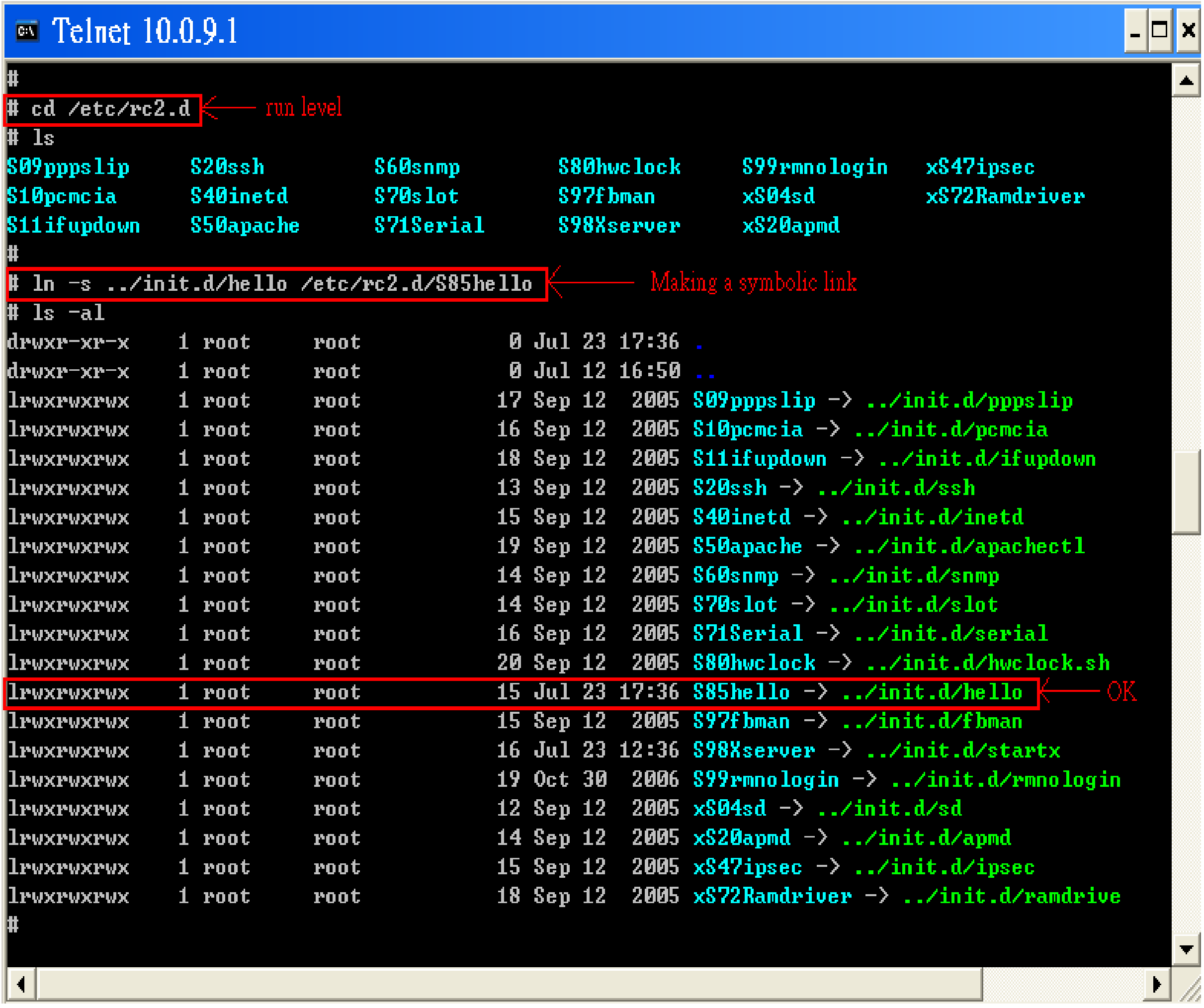
```
#!/bin/sh
#
# ICPDAS LinCon-8000 daemon
#
# /etc/init.d/hello  0.1 2004/05/025 ( moki matsushima )
usage()
{
    echo "Usage: $0 {start|stop|restart}"
}

EXITCODE=1
for x in "1" ; do
    if [ $# -lt 1 ] ; then usage ; break ; fi
    action=$1

    case "$action" in
    start)
        echo -n "Starting Hello services: "
        echo "Welcome to LinCon-8000!"
        EXITCODE=0
        ;;
    stop)
        echo -n "Shutting down hello services: "
        echo "done."
        EXITCODE=0
        ;;
    restart)
        $0 stop
        $0 start
        EXITCODE=$?
        ;;
    *)
        usage
        ;;
:wq
```

For declaring

Running at boot time.

Save and Quit

Fig. 4-6



```
#
# cd /etc/rc2.d
# ls
S09pppslip     S20ssh         S60snmp        S80hwclock     S99rmnologin   xS47ipsec
S10pcmcia      S40inetd       S70slot        S97fbman       xS04sd         xS72Ramdriver
S11ifupdown    S50apache      S71Serial      S98Xserver     xS20apmd
#
# ln -s ../init.d/hello /etc/rc2.d/S85hello
# ls -al
drwxr-xr-x    1 root     root            0 Jul 23 17:36 .
drwxr-xr-x    1 root     root            0 Jul 12 16:50 ..
lrwxrwxrwx    1 root     root           17 Sep 12  2005 S09pppslip -> ../init.d/pppslip
lrwxrwxrwx    1 root     root           16 Sep 12  2005 S10pcmcia -> ../init.d/pcmcia
lrwxrwxrwx    1 root     root           18 Sep 12  2005 S11ifupdown -> ../init.d/ifupdown
lrwxrwxrwx    1 root     root           13 Sep 12  2005 S20ssh -> ../init.d/ssh
lrwxrwxrwx    1 root     root           15 Sep 12  2005 S40inetd -> ../init.d/inetd
lrwxrwxrwx    1 root     root           19 Sep 12  2005 S50apache -> ../init.d/apachectl
lrwxrwxrwx    1 root     root           14 Sep 12  2005 S60snmp -> ../init.d/snmp
lrwxrwxrwx    1 root     root           14 Sep 12  2005 S70slot -> ../init.d/slot
lrwxrwxrwx    1 root     root           16 Sep 12  2005 S71Serial -> ../init.d/serial
lrwxrwxrwx    1 root     root           20 Sep 12  2005 S80hwclock -> ../init.d/hwclock.sh
lrwxrwxrwx    1 root     root           15 Jul 23 17:36 S85hello -> ../init.d/hello
lrwxrwxrwx    1 root     root           15 Sep 12  2005 S97fbman -> ../init.d/fbman
lrwxrwxrwx    1 root     root           16 Jul 23 12:36 S98Xserver -> ../init.d/startx
lrwxrwxrwx    1 root     root           19 Oct 30  2006 S99rmnologin -> ../init.d/rmnologin
lrwxrwxrwx    1 root     root           12 Sep 12  2005 xS04sd -> ../init.d/sd
lrwxrwxrwx    1 root     root           14 Sep 12  2005 xS20apmd -> ../init.d/apmd
lrwxrwxrwx    1 root     root           15 Sep 12  2005 xS47ipsec -> ../init.d/ipsec
lrwxrwxrwx    1 root     root           18 Sep 12  2005 xS72Ramdriver -> ../init.d/ramdrive
#
```

Fig. 4-7

4.5.2 Disabling program run at boot time

The steps are as follows :

(1) Type “ **cd /etc/rc2.d** “ to into default run level.

(2) Type “ **mv S85hello xS85hello** “ to rename the S85hello symbolic link for turn off running program automatically at boot time.

5. Demo of LinCon-8000 Modules With C Language

In this section, we will focus on examples for the description and application of the control functions on the I-7000/I-8000/I-87k series modules for use in the Lincon-8000. After you install the LinCon-8000 SDK, all these demo programs as below are in the path of **"c:/cygwin/lincon8k/examples"**.

5.1 I-7k Modules DIO Control Demo

This demo – **i7kdio.c** will illustrate how to control DI/DO with the I-7050 module (8 DO channels and 7 DI channels). The address and baudrate of the I-7050 module in the RS-485 network are 02 and 9600 separately.

The result of this demo allows the DO channels 0 ~ 7 output and DI channel 2 input. The source code of this demo program is as follows:

```
#include<stdio.h>
#include<stdlib.h>
#include "msw.h"

char szSend[80], szReceive[80], ans;
WORD wBuf[12];
float fBuf[12];

/* ------------------------------------------------------------------- */
int main()
{
      int   wRetVal;

      // Check Open_Com3
      wRetVal = Open_Com(COM3, 9600, Data8Bit, NonParity, OneStopBit);
      if (wRetVal > 0) {
            printf("open port failed!\n");
            return (-1);
      }
```

```
    //   *****   7050 DO && DI Parameter *******
    wBuf[0] = 3;                    // COM Port
    wBuf[1] = 0x02;                 // Address
    wBuf[2] = 0x7050;          // ID
    wBuf[3] = 0;                    // CheckSum disable
    wBuf[4] = 100;                // TimeOut , 100 msecond
    wBuf[5] = 0x0ff;              // 8 DO Channels On
    wBuf[6] = 0;                   // string debug

    // 7050 DO Output
        wRetVal = DigitalOut(wBuf, fBuf, szSend, szReceive);
        if (wRetVal)
            printf("DigitalOut_7050 Error !, Error Code=%d\n", wRetVal);
        printf("The DO of 7050 : %u \n", wBuf[5]);

    // 7050 DI Input
        DigitalIn(wBuf, fBuf, szSend, szReceive);
        printf("The DI of 7050 : %u \n", wBuf[5]);
    Close_Com(COM3);
  return 0;
}
```

Follow the below steps to achieve the desired results：

STEP 1：**(Write i7kdio.c)**

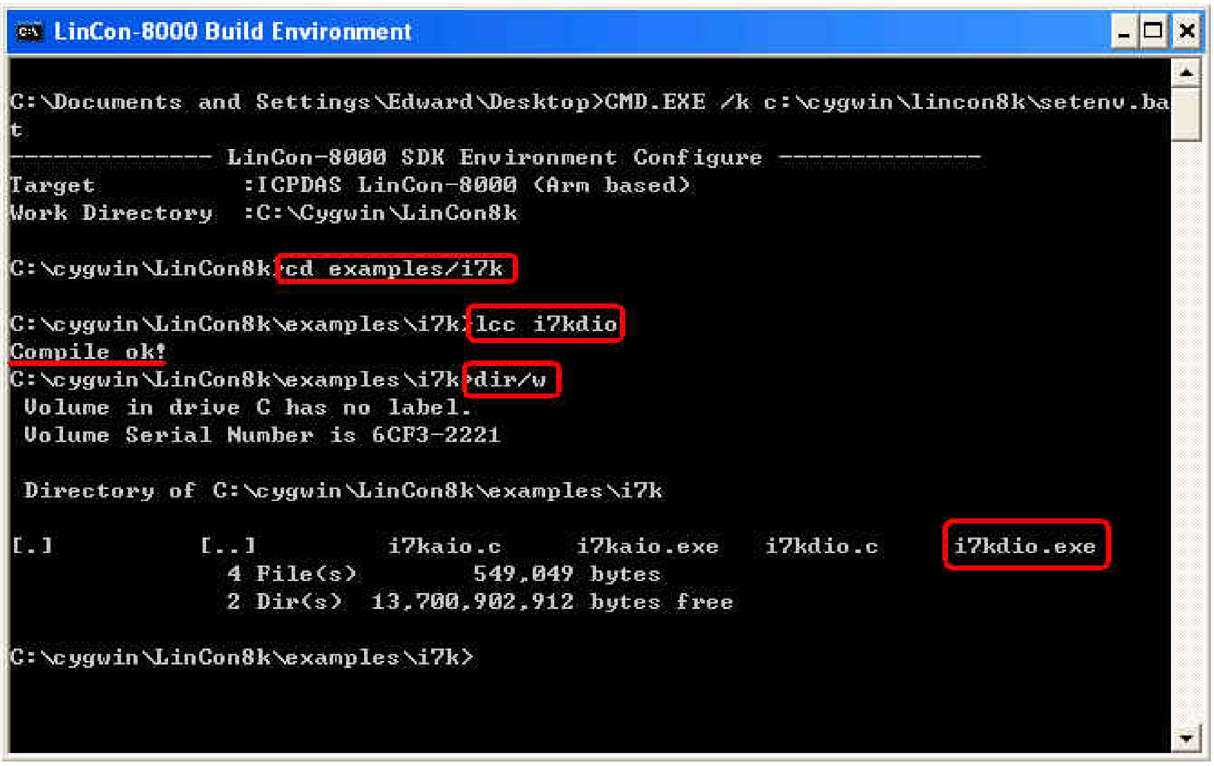
Copy the above source code and save it with the name - i7kdio.c or get the file from C:\cygwin\LinCon8k\examples\i7k.

STEP 2：**(Compile i7kdio.c to i7kdio.exe)**

Here we will introduce two methods to accomplish step 2.

< Method One > Using Batch File (lcc.bat)

Execute Start>Programs>ICPDAS>LinCon-8000 SDK> LinCon-8000 Build Environment to open **LinCon-8000 SDK** and change the path to C:\cygwin\LinCon8k\examples\i7k. Then type **lcc i7kdio** to compile i7kdio.c to i7kdio.exe. (refer to Fig. 5-1)

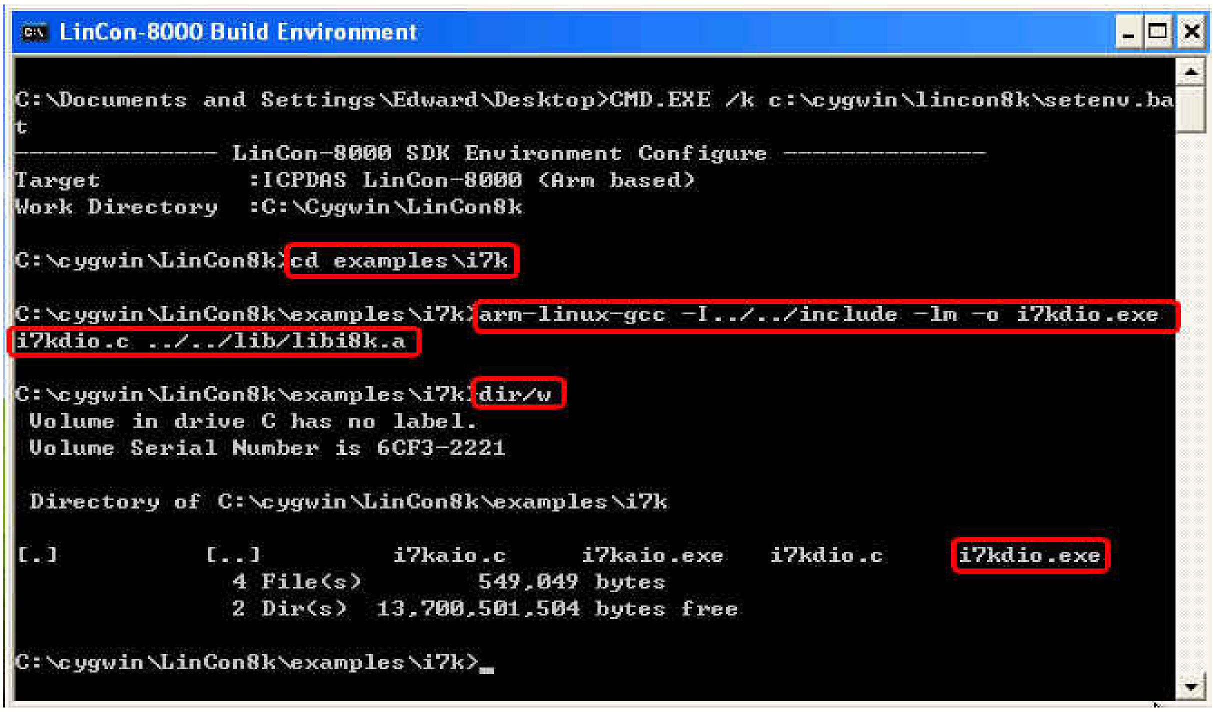


```
C:\Documents and Settings\Edward\Desktop>CMD.EXE /k c:\cygwin\lincon8k\setenv.ba
t
-------------- LinCon-8000 SDK Environment Configure --------------
Target          :ICPDAS LinCon-8000 (Arm based)
Work Directory  :C:\Cygwin\LinCon8k

C:\cygwin\LinCon8k>cd examples/i7k

C:\cygwin\LinCon8k\examples\i7k>lcc i7kdio
Compile ok!
C:\cygwin\LinCon8k\examples\i7k>dir/w
 Volume in drive C has no label.
 Volume Serial Number is 6CF3-2221

 Directory of C:\cygwin\LinCon8k\examples\i7k

[.]           [..]          i7kaio.c      i7kaio.exe    i7kdio.c      i7kdio.exe
               4 File(s)        549,049 bytes
               2 Dir(s)  13,700,902,912 bytes free

C:\cygwin\LinCon8k\examples\i7k>
```

Fig. 5-1

< Method Two > Using Compile Instruction

If you choose this method, change the path to C:\cygwin\LinCon8k\examples\i7k and then type arm-linux-gcc -I../../include –lm –o i7kdio.exe i7kdio.c ../../lib/libi8k.a to compile i7kdio.c to i7kdio.exe. (refer to Fig. 5-2)



```
C:\Documents and Settings\Edward\Desktop>CMD.EXE /k c:\cygwin\lincon8k\setenv.ba
t
-------------- LinCon-8000 SDK Environment Configure --------------
Target          :ICPDAS LinCon-8000 (Arm based)
Work Directory  :C:\Cygwin\LinCon8k

C:\cygwin\LinCon8k>cd examples\i7k

C:\cygwin\LinCon8k\examples\i7k>arm-linux-gcc -I../../include -lm -o i7kdio.exe
i7kdio.c ../../lib/libi8k.a

C:\cygwin\LinCon8k\examples\i7k>dir/w
 Volume in drive C has no label.
 Volume Serial Number is 6CF3-2221

 Directory of C:\cygwin\LinCon8k\examples\i7k

[.]           [..]          i7kaio.c      i7kaio.exe    i7kdio.c      i7kdio.exe
               4 File(s)        549,049 bytes
               2 Dir(s)  13,700,501,504 bytes free

C:\cygwin\LinCon8k\examples\i7k>_
```

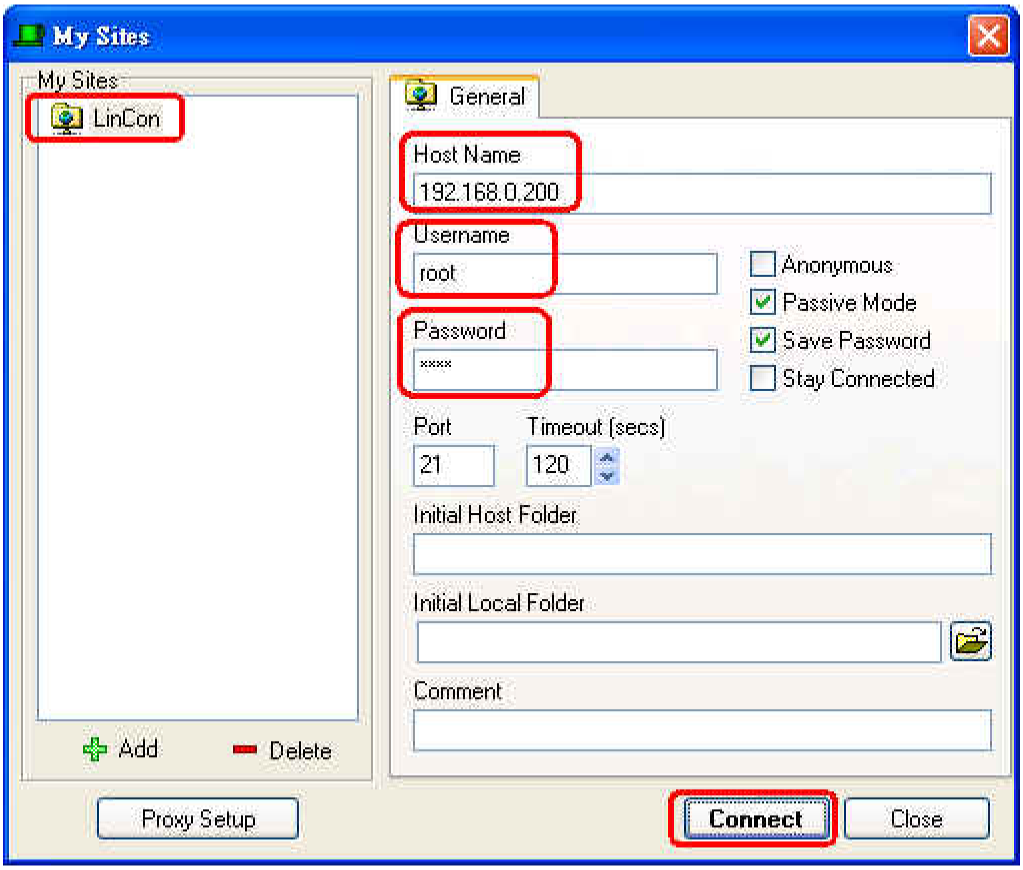
Fig. 5-2

STEP 3：**(Transfer i7kdio.exe to the LinCon-8000)**

Here we introduce two methods for achieving this purpose.

< Method One > Using FTP Software

(1) Open a FTP Software and add a ftp site of the LinCon-8000. The **User_Name** and **Password** default value is **“ root ”**. Then click the **“Connect”** button to connect to the ftp server of the LinCon-8000. (refer to Fig.5-3).



Host Name: 192.168.0.200

Username: root

Password: ****

☐ Anonymous
☑ Passive Mode
☑ Save Password
☐ Stay Connected

Port: 21

Timeout (secs): 120

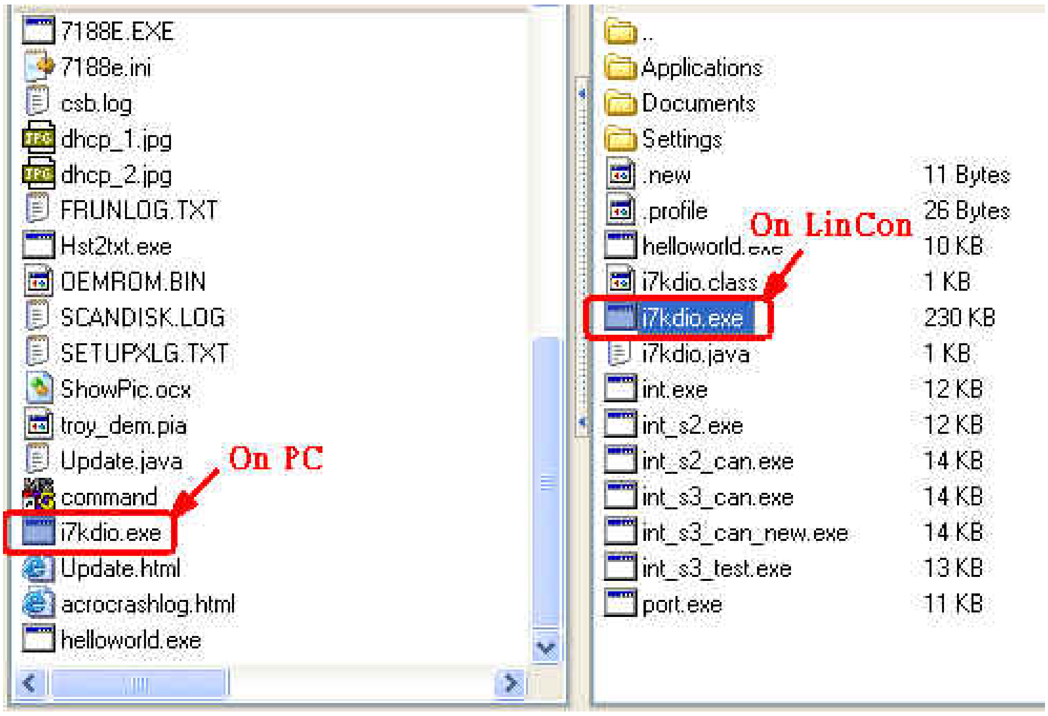
Initial Host Folder

Initial Local Folder

Comment

Fig.5-3

(2) Upload the file – **i7kdio.exe** to the LinCon-8000. (refer to Fig.5-4).

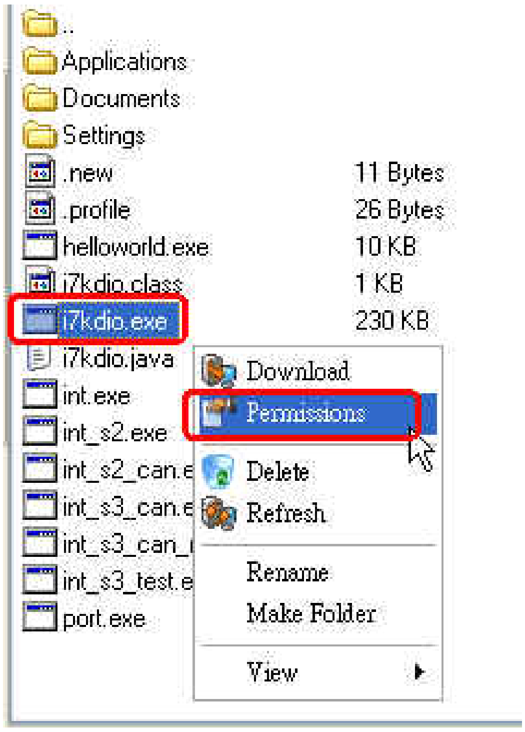


7188E.EXE
7188e.ini
csb.log
dhcp_1.jpg
dhcp_2.jpg
FRUNLOG.TXT
Hst2txt.exe
OEMROM.BIN
SCANDISK.LOG
SETUPXLG.TXT
ShowPic.ocx
troy_dem.pia
Update.java
command
i7kdio.exe
Update.html
acrocrashlog.html
helloworld.exe

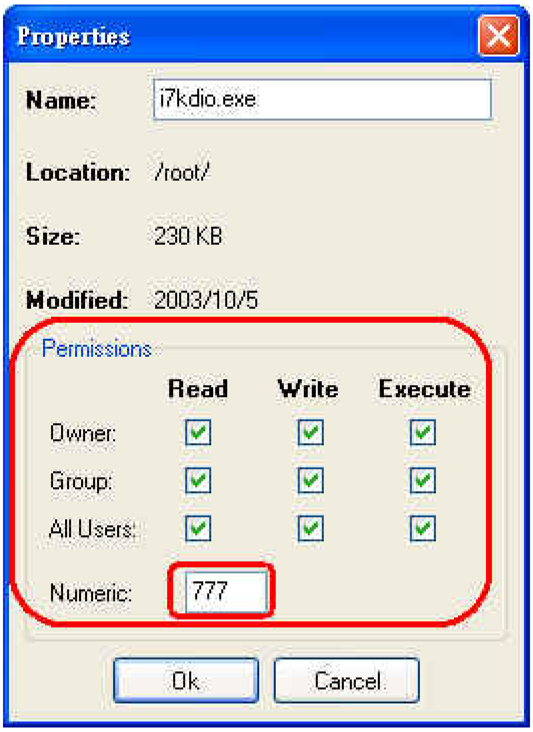
..	
Applications	
Documents	
Settings	
.new	11 Bytes
.profile	26 Bytes
helloworld.exe	10 KB
i7kdio.class	1 KB
i7kdio.exe	230 KB
i7kdio.java	1 KB
int.exe	12 KB
int_s2.exe	12 KB
int_s2_can.exe	14 KB
int_s3_can.exe	14 KB
int_s3_can_new.exe	14 KB
int_s3_test.exe	13 KB
port.exe	11 KB

Fig.5-4

(3) Choose i7kdio.exe in the LinCon-8000 and Click the right mouse button to choose the “ **Permission** ”option. Then type 777 into the Numeric blank textbox. (refer to Fig.5-5 and refer to Fig.5-6).



..	
Applications	
Documents	
Settings	
.new	11 Bytes
.profile	26 Bytes
helloworld.exe	10 KB
i7kdio.class	1 KB
i7kdio.exe	230 KB
i7kdio.java	
int.exe	
int_s2.exe	
int_s2_can.e	
int_s3_can.e	
int_s3_can_	
int_s3_test.e	
port.exe	



Name: i7kdio.exe

Location: /root/

Size: 230 KB

Modified: 2003/10/5

Permissions

	Read	Write	Execute
Owner:	☑	☑	☑
Group:	☑	☑	☑
All Users:	☑	☑	☑

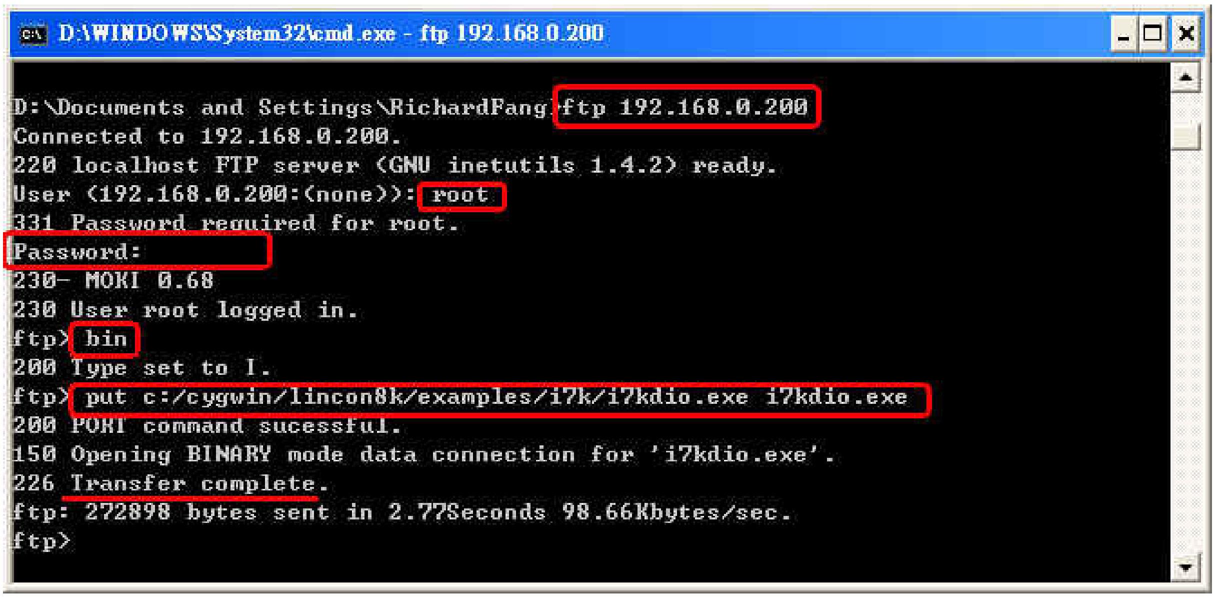
Numeric: 777

Fig.5-5

Fig.5-6

< Method Two > Using DOS Command Prompt

Open DOS Command Prompt and type ftp IP Address of LinCon-8000 in order to connect to the ftp server of the LinCon-8000. Then input **User Name** and **Password** (**root** is the default value) to login to the LinCon-8000. Type **bin** to make the file transference in "binary" mode. Then type put c:/cygwin/lincon8k/examples/i7k/i7kdio.exe i7kdio.exe to transfer the i7kdio.exe to the LinCon-8000. After the "Transfer complete" message appears, the process of transference would have been completed.(refer to Fig. 5-7)

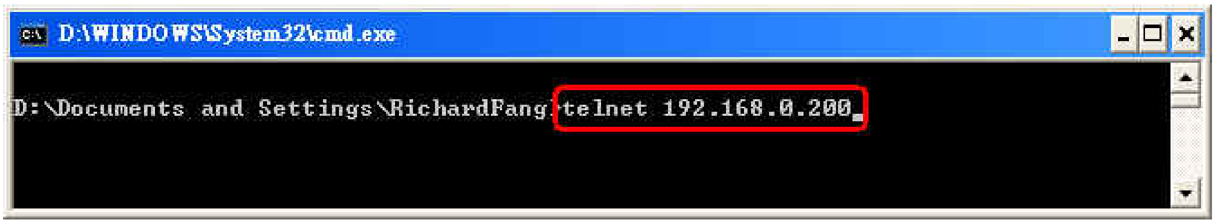


```
D:\Documents and Settings\RichardFang>ftp 192.168.0.200
Connected to 192.168.0.200.
220 localhost FTP server (GNU inetutils 1.4.2) ready.
User (192.168.0.200:(none)): root
331 Password required for root.
Password:
230- MOKI 0.68
230 User root logged in.
ftp> bin
200 Type set to I.
ftp> put c:/cygwin/lincon8k/examples/i7k/i7kdio.exe i7kdio.exe
200 PORT command sucessful.
150 Opening BINARY mode data connection for 'i7kdio.exe'.
226 Transfer complete.
ftp: 272898 bytes sent in 2.77Seconds 98.66Kbytes/sec.
ftp>
```

Fig. 5-7

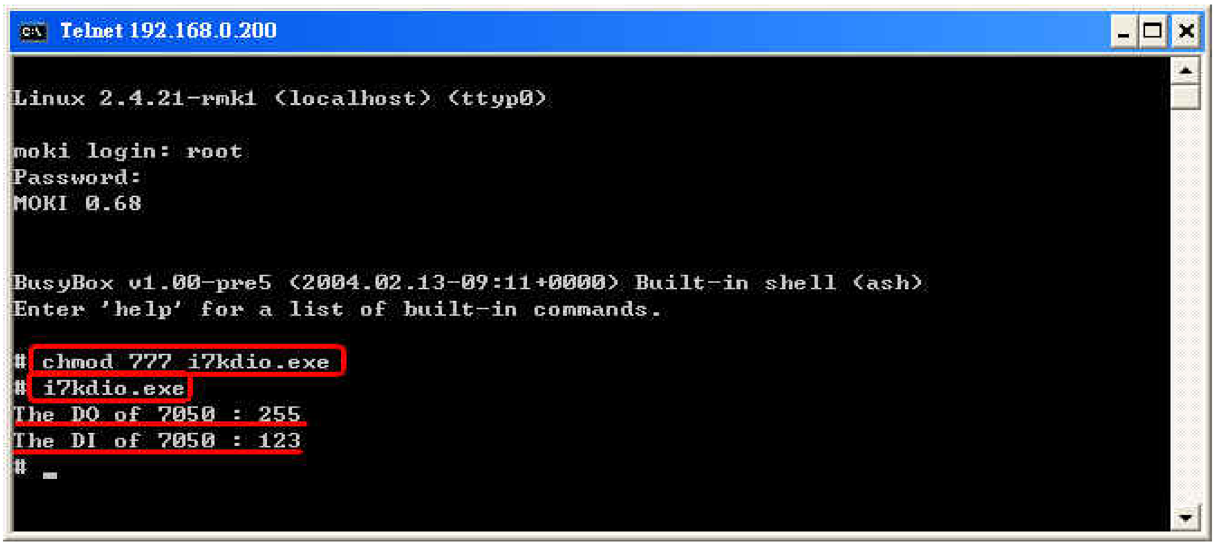
STEP 4：**(Telnet to the LinCon-8000 to execute i7kdio.exe)**

Type telnet IP Address of LinCon-8000 into the remote control the LinCon-8000 and input your **User Name** and **Password** (**root** is the default value) to login to the LinCon-8000. And then type chmod 777 i7kdio.exe to make i7kdio.exe executable. Type i7kdio.exe to execute i7kdio.exe. (refer to Fig. 5-8 and Fig. 5-9)



```
D:\Documents and Settings\RichardFang>telnet 192.168.0.200
```

Fig. 5-8



```
Linux 2.4.21-rmk1 (localhost) (ttyp0)

moki login: root
Password:
MOKI 0.68


BusyBox v1.00-pre5 (2004.02.13-09:11+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

# chmod 777 i7kdio.exe
# i7kdio.exe
The DO of 7050 : 255
The DI of 7050 : 123
# _
```

Fig. 5-9

“ **The DO of I-7050：255** (=2^**8**-1)” means DO channel 0 ~ 7 will output and “ **The DI of I-7050：123** (=127-2^**2**)” means there is input in DI channel 2.

5.2 I-7k Modules AIO Control Demo

This demo – **i7kaio.c** will illustrate how to control the AI/AO with the I-7017 (8 AI channels) and I-7021 modules (1 AO channel). The address for the I-7021 and I-7017 modules are in the RS-485 network where 05 and 03 are separate and the baudrate is 9600.

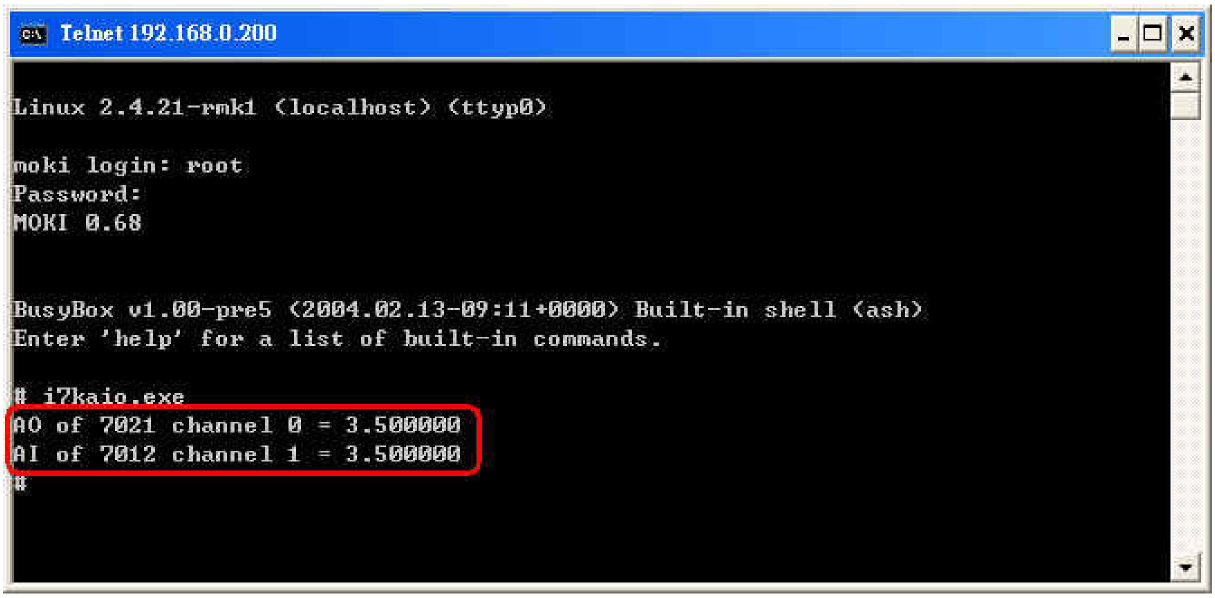
The result of this demo allows the I-7021 module’s AO channel to output 3.5V and the I-7017 ‘s AI channel 2 to input. The source code of this demo program is as follows：

```
#include<stdio.h>
#include<stdlib.h>
#include "msw.h"

char szSend[80], szReceive[80];
WORD wBuf[12];
float fBuf[12];
```

```
/* ---------------------------------------------------------------- */
int main()
{
    int i,j, wRetVal;
    DWORD temp;

    wRetVal = Open_Com(COM3, 9600, Data8Bit, NonParity, OneStopBit);
    if (wRetVal > 0) {
        printf("open port failed!\n");
        return (-1);
    }

    //--- Analog output ----   ****    7021 -- AO   ****
    i = 0;
    wBuf[0] = 3;                // COM Port
    wBuf[1] = 0x05;             // Address
    wBuf[2] = 0x7021;           // ID
    wBuf[3] = 0;                // CheckSum disable
    wBuf[4] = 100;              // TimeOut , 100 msecond
    //wBuf[5] = i;              // Not used if module ID is 7016/7021
                                // Channel No.(0 to 1) if module ID is 7022
                                // Channel No.(0 to 3) if module ID is 7024
    wBuf[6] = 0;                // string debug
    fBuf[0] = 3.5;               // Analog Value

    wRetVal = AnalogOut(wBuf, fBuf, szSend, szReceive);
    if (wRetVal)
        printf("AO of 7021 Error !, Error Code=%d\n", wRetVal);
    else
        printf("AO of 7021 channel %d = %f \n",i,fBuf[0]);

    //--- Analog Input ----   ****    7017 -- AI   ****
    j = 1;
    wBuf[0] = 3;                // COM Port
    wBuf[1] = 0x03;             // Address
    wBuf[2] = 0x7017;           // ID
    wBuf[3] = 0;                // CheckSum disable
    wBuf[4] = 100;              // TimeOut , 100 msecond
```

```
        wBuf[5] = j;                    // Channel of AI
        wBuf[6] = 0;                    // string debug

        wRetVal = AnalogIn(wBuf, fBuf, szSend, szReceive);
        if (wRetVal)
                printf("AI of 7017 Error !, Error Code=%d\n", wRetVal);
        else
                printf("AI of 7017 channel %d = %f \n",j,fBuf[0]);

        Close_Com(COM3);

        return 0;
}
```

All the steps from programming to execution are the same as those in the section 5.1. The result of execution refers to Fig. 5-10.



```
Linux 2.4.21-rmk1 (localhost) (ttyp0)

moki login: root
Password:
MOKI 0.68


BusyBox v1.00-pre5 (2004.02.13-09:11+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

# i7kaio.exe
AO of 7021 channel 0 = 3.500000
AI of 7012 channel 1 = 3.500000
#
```

Fig. 5-10

5.3 I-87k Modules DIO Control Demo

When using I-87k modules for I/O control of the LinCon-8000, the program will be a little different, according to the location of I-87k modules. There are three conditions for the location of the I-87k modules：

(1) When I-87k modules are **in the LinCon-8000 slots**, the two functions “ Open_Slot ” and “ ChangeToSlot ”, must be added before using other functions for the I-87k modules and the function of “Close_Slot() “ also needs to be added to the end of the program. Please refer to demo in section 5.3.1.

(2) When I-87K modules are **in the I-87k I/O expansion unit slots**, then please refer to the demo in section 5.3.2.

(3) When the I-87k modules are **in the I-8000 controller slots**, then the I-87k modules will be regarded as I-8k modules and so please refer to I/O control of I-8k modules in section 5.5.2

5.3.1 I-87k Modules in slots of LinCon-8000

This demo – **i87kdio.c** will illustrate how to control the DI/DO with the I-87054 module (8 DO channels and 8 DI channels). The I-87054 module is in slot 3 of the LinCon-8000. The address and baudrate in the LinCon-8000 are constant and they are 00 and 115200 respectively. The result of this demo lets DO channel 0 ~ 7 of I-87054 output and DI channel 1 of I-87054 input. The source code of this demo program is as follows：

```
#include<stdio.h>
#include<stdlib.h>
#include "msw.h"

char szSend[80], szReceive[80];
DWORD dwBuf[12];
float fBuf[12];

/* ------------------------------------------------------------------ */
int main()
{
    int i, wRetVal;
    DWORD temp;
```

```
  //Check Open_Slot
  wRetVal = Open_Slot(0);
  if (wRetVal > 0) {
        printf("open Slot failed!\n");
        return (-1);
  }

  //Check Open_Com1
  wRetVal = Open_Com(COM1, 115200, Data8Bit, NonParity, OneStopBit);
  if (wRetVal > 0) {
        printf("open port failed!\n");
        return (-1);
  }

  //Choose Slot3
   ChangeToSlot(3);
  //--- digital output ----   **(DigitalOut_87k()**)
  dwBuf[0] = 1;                      // COM Port
  dwBuf[1] = 00;                     // Address
  dwBuf[2] = 0x87054;                // ID
  dwBuf[3] = 0;                      // CheckSum disable
  dwBuf[4] = 100;                    // TimeOut , 100 msecond
  dwBuf[5] = 0xff;                   // digital output
  dwBuf[6] = 0;                      // string debug
wRetVal = DigitalOut_87k(dwBuf, fBuf, szSend, szReceive); // DO Output
printf("DO Value= %u", dwBuf[5]);


//--- digital Input ----     **(DigitalIn_87k()**)
  dwBuf[0] = 1;                      // COM Port
  dwBuf[1] = 00;                     // Address
  dwBuf[2] = 0x87054;                // ID
  dwBuf[3] = 0;                      // CheckSum disable
  dwBuf[4] = 100;                    // TimeOut , 100 msecond

  dwBuf[6] = 0;                      // string debug
  getch();
  DigitalIn_87k(dwBuf, fBuf, szSend, szReceive);     // DI Input
```

```
          printf("DI= %u",dwBuf[5]);

        //--- digital output ----      ** Close DO **
           dwBuf[0] = 1;                         // COM Port
           dwBuf[1] = 00;                        // Address
           dwBuf[2] = 0x87054;                   // ID
           dwBuf[3] = 0;                         // CheckSum disable
           dwBuf[4] = 100;                       // TimeOut , 100 msecond
           dwBuf[5] = 0x00;                      // digital output
           dwBuf[6] = 0;                         // string debug
           getch();                             // push any key to continue
          wRetVal = DigitalOut_87k(dwBuf, fBuf, szSend, szReceive);

           Close_Com(COM1);
           Close_SlotAll();
           return 0;
       }
```

5.3.2 I-87k Modules in slots of I-87k I/O expansion unit

If the I-87k modules are in the slots of the I-87k I/O expansion unit, the above program needs to be modified in three parts：

(4) The functions of **Open_Slot() , ChangeToSlot(), Close_SlotAll()** will be deleted.

(5) The **address** and **baudrate** of I-87k modules in the network of RS-485 need to be set by DCON Utility

(6) **Open com1**(internal serial port of LinCon-8000) will be modified to **open com3** (RS-485 port of LinCon-8000)

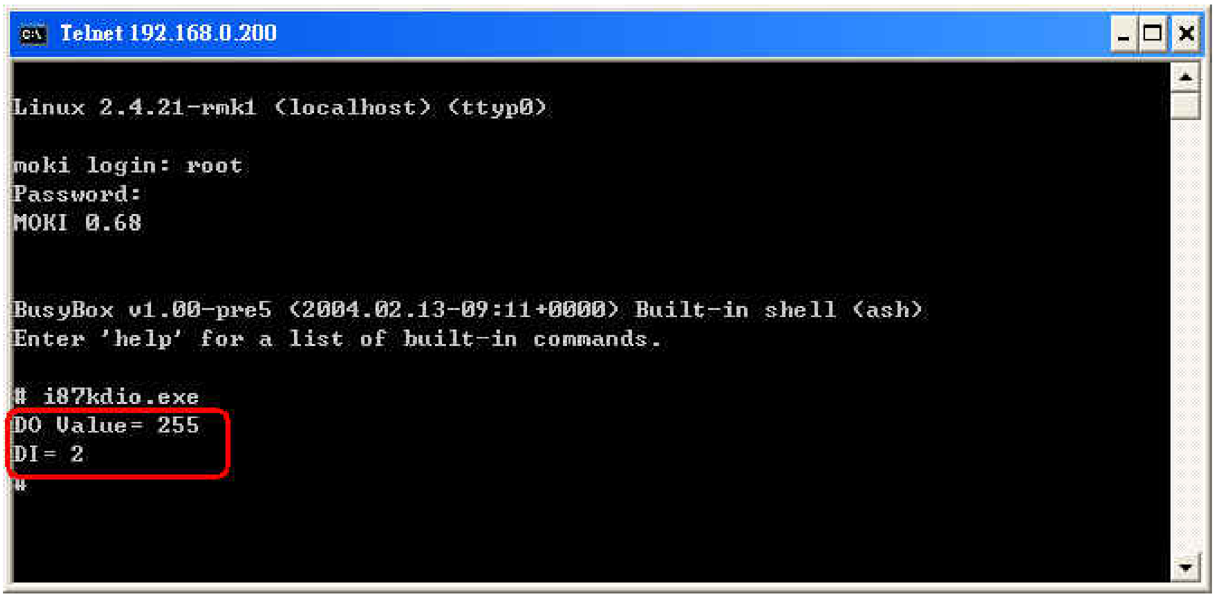
The address and baudrate of the I-87054 in the RS-485 network are set to be 06 and 9600 separately by the DCON Utility. The source code of this demo program – **i87kdio_87k.c** is as follows：

```
#include<stdio.h>
#include<stdlib.h>
```

```
#include "msw.h"

char szSend[80], szReceive[80];
DWORD dwBuf[12];
float fBuf[12];

/* ---------------------------------------------------------------- */
int main()
{
    int i, wRetVal;
    DWORD temp;
    //Check Open_Com3
    wRetVal = Open_Com(COM3, 9600, Data8Bit, NonParity, OneStopBit);
    if (wRetVal > 0) {
        printf("open port failed!\n");
        return (-1);
    }
    //--- digital output ----   **(DigitalOut_87k()**)
    dwBuf[0] = 3;                 // COM Port
    dwBuf[1] = 06;                // Address
    dwBuf[2] = 0x87054;           // ID
    dwBuf[3] = 0;                 // CheckSum disable
    dwBuf[4] = 100;               // TimeOut , 100 msecond
    dwBuf[5] = 0xff;              // digital output
    dwBuf[6] = 0;                 // string debug
  wRetVal = DigitalOut_87k(dwBuf, fBuf, szSend, szReceive);   // DO Output
  printf("DO Value= %u", dwBuf[5]);

  //--- digital Input ----     **(DigitalIn_87k()**)
    dwBuf[0] = 3;                 // COM Port
    dwBuf[1] = 06;                // Address
    dwBuf[2] = 0x87054;           // ID
    dwBuf[3] = 0;                 // CheckSum disable
    dwBuf[4] = 100;               // TimeOut , 100 msecond
    dwBuf[6] = 0;                 // string debug
    getch();
    DigitalIn_87k(dwBuf, fBuf, szSend, szReceive);     // DI Input
  printf("DI= %u",dwBuf[5]);
```

```
        //--- digital output ----      ** Close DO **
        dwBuf[0] = 3;                 // COM Port
        dwBuf[1] = 06;                // Address
        dwBuf[2] = 0x87054;           // ID
        dwBuf[3] = 0;                 // CheckSum disable
        dwBuf[4] = 100;               // TimeOut , 100 msecond
        dwBuf[5] = 0x00;              // digital output
        dwBuf[6] = 0;                 // string debug
        getch();                      // push any key to continue
    wRetVal = DigitalOut_87k(dwBuf, fBuf, szSend, szReceive);

    Close_Com(COM3);
return 0;
}
```

All the steps from programming to execution are the same as those in the section 5.1. The result of execution refers to Fig. 5-11.



```
Linux 2.4.21-rmk1 (localhost) (ttyp0)

moki login: root
Password:
MOKI 0.68


BusyBox v1.00-pre5 (2004.02.13-09:11+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

# i87kdio.exe
DO Value= 255
DI= 2
#
```

Fig. 5-11

5.3.3 I-87k Modules in slots of I-8000 Controller

If the I-87k DI/DO modules are in the I-8000 controller slots, I-87k modules will be regarded as I-8k modules and so please refer to DI/DO control of I-8k modules in the section 5.5.

5.4 I-87k Modules AIO Control Demo

When using I-87k modules for I/O control of the LinCon-8000, according to the location of the I-87k modules, the program will be a little different. There are three conditions for the location of the I-87k modules：

(1) When the I-87k modules are **in the LinCon-8000 slots**, the two functions " Open_Slot " and " ChangeToSlot " must be added before using the other functions of the I-87k modules and the function " Close_Slot() " also needs to be added to the end of the program. Please refer to the demo in section 5.4.1.

(2) When I-87K modules are **in the I-87k I/O expansion unit slots**, please refer to the demo in section 5.4.2.

(3) When the I-87k modules are **in the I-8000 controller slots**, the I-87k modules will be regarded as I-8k modules and so please refer to I/O control of I-8k modules in section 5.6.2

5.4.1 I-87k Modules in slots of LinCon-8000

This demo – **i87kaio.c** will illustrate how to control the AI/AO with the I-87022 module (2 AO channels) and the I-87017 module (8 AI channels).The I-87022 and I-87017 modules are plugged into slot 2 and slot 3 of the LinCon-8000 separately. The address and baudrate in the LinCon-8000 are constant and they are 00 and 115200 separately. The result of this demo lets AO channel 0 of I-87022 output 2.5V and AI channel 1 of I-87017 input. The source code of this demo program is as follows：

```
#include<stdio.h>
#include<stdlib.h>
#include "msw.h"

char szSend[80], szReceive[80];
DWORD wBuf[12];
DWORD   wBuf7[12];
float fBuf[12];
```

```
/* --------------------------------------------------------------- */
int main()
{
      int i,j, wRetVal;
      DWORD temp;

   //Check Open_Slot
   wRetVal = Open_Slot(0);
      if (wRetVal > 0) {
            printf("open Slot failed!\n");
            return (-1);
      }

      //Check Open_Com1
      wRetVal = Open_Com(COM1, 115200, Data8Bit, NonParity, OneStopBit);
      if (wRetVal > 0) {
            printf("open port failed!\n");
            return (-1);
      }

      ChangeToSlot(2);
      //--- Analog output ----   ****     87022 -- AO    ****
      i=0;
      wBuf[0] = 1;                    // COM Port
      wBuf[1] = 0x00;                 // Address
      wBuf[2] = 0x87022;              // ID
      wBuf[3] = 0;                    // CheckSum disable
      wBuf[4] = 100;                  // TimeOut , 100 msecond
      wBuf[5] = i;                    // Channel Number of AO
      wBuf[6] = 0;                    // string debug
      fBuf[0] = 2.5;                  // AO Value

      wRetVal = AnalogOut_87k(wBuf, fBuf, szSend, szReceive);
      if (wRetVal)
            printf("AO of 87022 Error !, Error Code=%d\n", wRetVal);
      else
            printf("AO of 87022 channel %d = %f \n",i,fBuf[0]);
```

```
    ChangeToSlot(3);
    //--- Analog Input ----   ****    87017 -- AI   ****
    j=1;
    wBuf7[0] = 1;             // COM Port
    wBuf7[1] = 0x00;          // Address
    wBuf7[2] = 0x87017;       // ID
    wBuf7[3] = 0;             // CheckSum disable
    wBuf7[4] = 100;           // TimeOut , 100 msecond
    wBuf7[5] = j;             //Channel Number of AI
    wBuf7[6] = 0;             // string debug

    wRetVal = AnalogIn_87k(wBuf7, fBuf, szSend, szReceive);
    if (wRetVal)
        printf("AI of 87017 Error !, Error Code=%d\n", wRetVal);
    else
        printf("AI of 87017 channel %d = %f \n",j,fBuf[0]);

    Close_Com(COM1);
    Close_SlotAll();
    return 0;
}
```

5.4.2 I-87k Modules in slots of I-87k I/O expansion unit

If the I-87k modules are in slots of I-87k I/O expansion unit, the above program needs to be modified in three parts：

(1) The functions of **Open_Slot() , ChangeToSlot(), Close_SlotAll()** will be deleted.

(2) The **addrss** and **baudrate** of I-87k modules in the network of RS-485 need to be set by DCON Utility

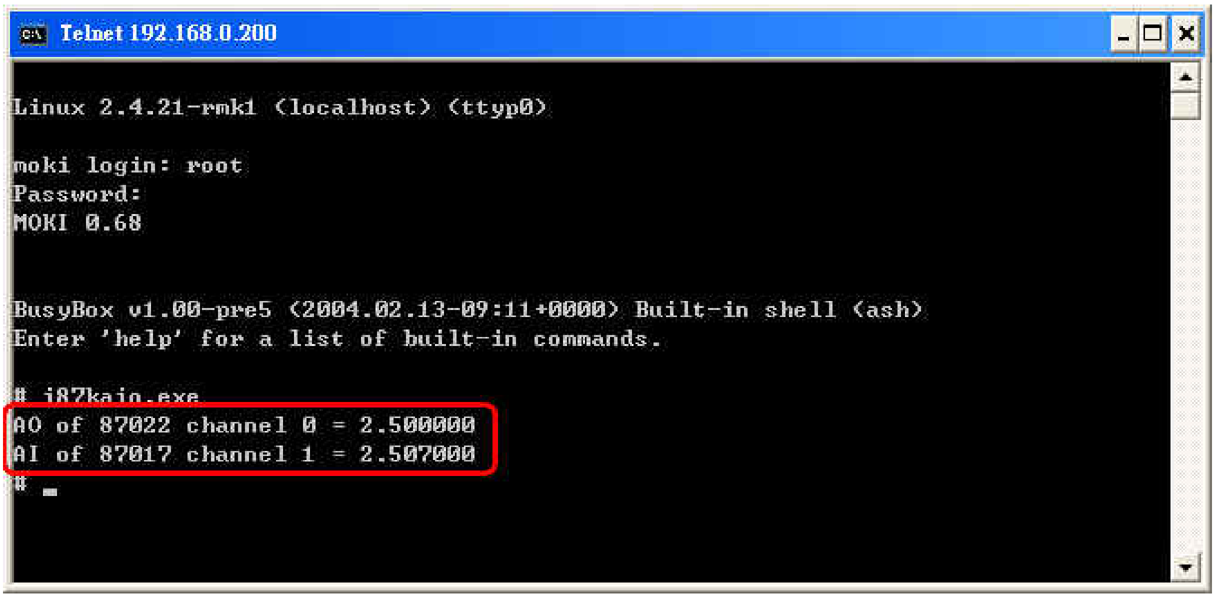
(3) **Open com1**(internal serial port of LinCon-8000) will be modified to **open com3** (RS-485 port of LinCon-8000)

The addresses I-87022 and I-87017 are in the RS-485 network and are set to be 01 and 02 separately and the baudrate is 9600 by DCON Utility. The source code of this demo program – **i87kaio_87k.c** is as follows：

```
#include<stdio.h>
#include<stdlib.h>
#include "msw.h"

char szSend[80], szReceive[80];
DWORD wBuf[12];
DWORD   wBuf7[12];
float fBuf[12];

/* --------------------------------------------------------------- */
int main()
{
     int i,j, wRetVal;
     DWORD temp;

     //Check Open_Com3
     wRetVal = Open_Com(COM3, 9600, Data8Bit, NonParity, OneStopBit);
     if (wRetVal > 0) {
          printf("open port failed!\n");
          return (-1);
     }

     //--- Analog output ----   ****    87022 -- AO   ****
     i=0;
     wBuf[0] = 3;                    // COM Port
     wBuf[1] = 0x01;                 // Address
     wBuf[2] = 0x87022;              // ID
     wBuf[3] = 0;                    // CheckSum disable
     wBuf[4] = 100;                  // TimeOut , 100 msecond
     wBuf[5] = i;                    // Channel Number of AO
     wBuf[6] = 0;                    // string debug
     fBuf[0] = 2.5;                  // AO Value

     wRetVal = AnalogOut_87k(wBuf, fBuf, szSend, szReceive);
     if (wRetVal)
          printf("AO of 87022 Error !, Error Code=%d\n", wRetVal);
     else
          printf("AO of 87022 channel %d = %f \n",i,fBuf[0]);
```

```
    //--- Analog Input ----   ****    87017 -- AI   ****
    j=1;
    wBuf7[0] = 3;               // COM Port
    wBuf7[1] = 0x02;            // Address
    wBuf7[2] = 0x87017;         // ID
    wBuf7[3] = 0;               // CheckSum disable
    wBuf7[4] = 100;             // TimeOut , 100 msecond
    wBuf7[5] = j;               //Channel Number of AI
    wBuf7[6] = 0;               // string debug

    wRetVal = AnalogIn_87k(wBuf7, fBuf, szSend, szReceive);
    if (wRetVal)
        printf("AI of 87017 Error !, Error Code=%d\n", wRetVal);
    else
        printf("AI of 87017 channel %d = %f \n",j,fBuf[0]);

    Close_Com(COM3);

    return 0;
}
```

All the steps from programming to execution are the same as those in the section 5.1. The result of execution refers to Fig. 5-12.



```
Linux 2.4.21-rmk1 (localhost) (ttyp0)

moki login: root
Password:
MOKI 0.68


BusyBox v1.00-pre5 (2004.02.13-09:11+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

# i87kaio.exe
AO of 87022 channel 0 = 2.500000
AI of 87017 channel 1 = 2.507000
# _
```

Fig. 5-12

5.4.3 I-87k Modules in slots of I-8000 Controller

If the I-87k AI/AO modules are in slots of I-8000 controller, I-87k modules will be regarded as I-8k modules and refer to AI/AO control of I-8k modules in the section 5.6.

5.5 I-8k Modules DIO Control Demo

I8000.c of Libi8k.a is the source file for i8k modules in slots of I-8000 controller. **Slot.c** of Libi8k.a is the source file for i8k modules in slots of LinCon-8000. Therefore the functions for i8k modules in slots of LinCon-8000 and in slots of I-8000 controller are different completely.

5.5.1 I-8k Modules in slots of LinCon-8000

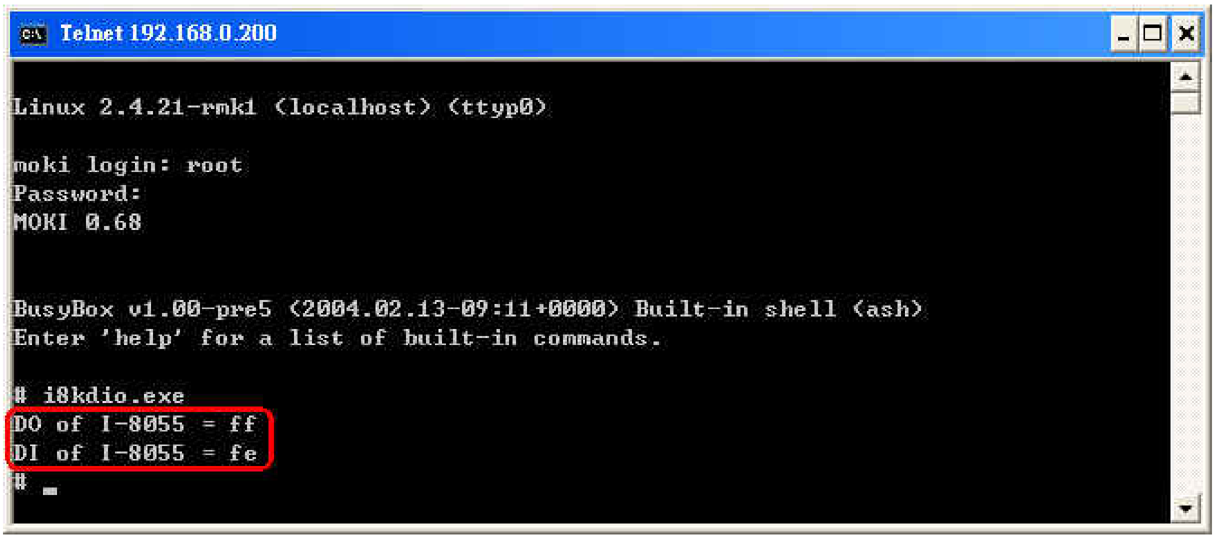
In this section, this demo program – **i8kdio.c** will introduce how to control the DI/DO with the I-8055 (8 DO channels and 8 DI channels) module and it is plugged into slot 3 of the LinCon-8000.

The address and baudrate in the LinCon-8000 are constant and they are 00 and 115200 separately. The result of this demo lets DO channel 0 ~7 of I-8055 output and DI channel 0 of I-8055 input. The source code of this demo program is as follows：

```
#include<stdio.h>
#include<stdlib.h>
#include "msw.h"

char szSend[80], szReceive[80];
DWORD dwBuf[12];
float fBuf[12];

/* ----------------------------------------------------------------- */
int main()
{
      int i,j, wRetVal;
      WORD DOval,temp;
```

```
        wRetVal = Open_Slot(3);

        if (wRetVal > 0) {
              printf("open Slot failed!\n");
              return (-1);
        }

        //I-8055_DO
        DO_8(3,255);
        printf("DO of I-8055 = 0x%x \n", 255);


        //I-8055_DI
        printf("DI of I-8055 = %x",DI_8(3));

        Close_Slot(3);

        return 0;
}
```

All the steps from programming to execution are the same as those in the section 5.1. The result of execution refers to Fig. 5-13.



```
Linux 2.4.21-rmk1 (localhost) (ttyp0)

moki login: root
Password:
MOKI 0.68


BusyBox v1.00-pre5 (2004.02.13-09:11+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

# i8kdio.exe
DO of I-8055 = ff
DI of I-8055 = fe
# _
```

Fig. 5-13

5.5.2 I-8k Modules in slots of I-8000 Controller

In this section, this demo program – **i8kdio_8k.c** will illustrate how to control the DI/DO with the I-8055 (8 DO channels and 8 DI channels) module. Please follow the below steps to configure the hardware：

(1) Put the I-8055 module in slot 0 of I-8000 controller.

(2) Install 8k232.exe or R232_300.exe to flash memory of I-8000 controller as firmware.

(3) Connect the com2 of LinCon-8000 to the com1 of I-8000 controller with the RS-232 cable.

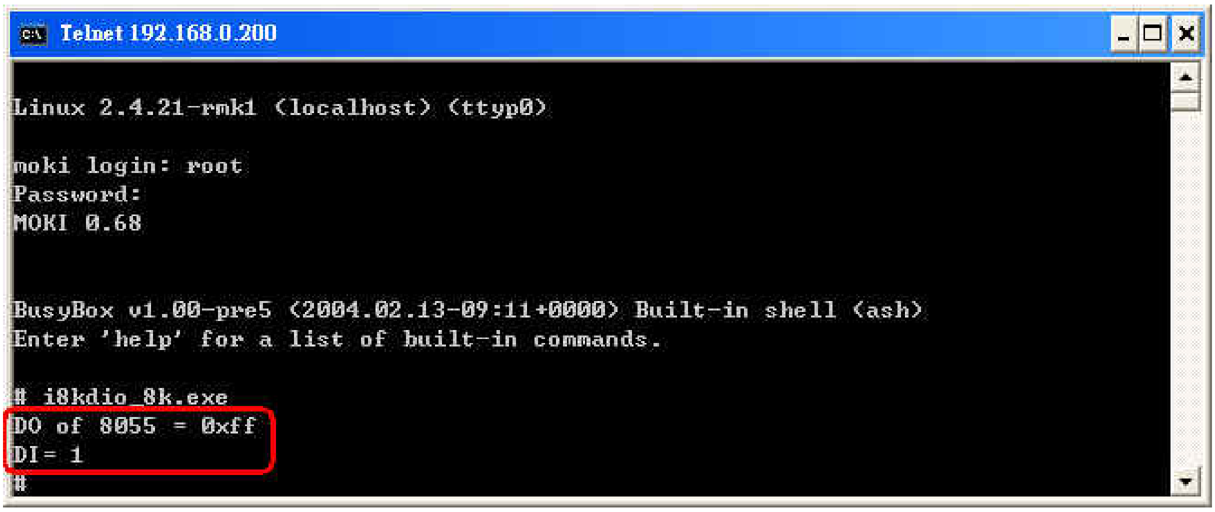
The address of I-8000 controller is 01 and the baudrate is 115200 that can be modified by DCON Utility. The result of this demo lets DO channel 0 ~7 of I-8055 output and DI channel 0 of I-8055 input. The source code of this demo program is as follows：

```
#include<stdio.h>
#include<stdlib.h>
#include "msw.h"

char szSend[80], szReceive[80];
DWORD dwBuf[12];
float fBuf[12];

/* --------------------------------------------------------------- */
int main()
{
      int i, wRetVal;
      DWORD temp;

      //Check Open_Com2
      wRetVal = Open_Com(COM2, 115200, Data8Bit, NonParity, OneStopBit);
      if (wRetVal > 0) {
            printf("open port failed!\n");
            return (-1);
      }
```

```
  //--- digital output ----   **(DigitalOut_8K()**)
  dwBuf[0] = 2;               // COM Port
  dwBuf[1] = 01;              // Address
  dwBuf[2] = 0x8055;          // ID
  dwBuf[3] = 0;               // CheckSum disable
  dwBuf[4] = 100;             // TimeOut , 100 msecond
  dwBuf[5] = 0xff;            // digital output
  dwBuf[6] = 0;               // string debug
  dwBuf[7] = 1;              // slot number

wRetVal = DigitalOut_8K(dwBuf, fBuf, szSend, szReceive);
if (wRetVal)
          printf("DO of 8055 Error !, Error Code=%d\n", wRetVal);
  else
          printf("DO of 8055 = 0x%x" ,dwBuf[5]);

//--- digital Input ----    **(DigitalIn_8K()**)
  dwBuf[0] = 2;                 // COM Port
  dwBuf[1] = 01;                // Address
  dwBuf[2] = 0x8055;            // ID
  dwBuf[3] = 0;                 // CheckSum disable
  dwBuf[4] = 100;               // TimeOut , 100 msecond

  dwBuf[6] = 0;                 // string debug
  dwBuf[7] = 1;                // slot number

  getch();
  DigitalIn_8K(dwBuf, fBuf, szSend, szReceive);
  printf("DI = %u",dwBuf[5]);

//--- digital output ----     ** Close DO **
  dwBuf[0] = 2;                // COM Port
  dwBuf[1] = 01;               // Address
  dwBuf[2] = 0x8055;           // ID
  dwBuf[3] = 0;                // CheckSum disable
  dwBuf[4] = 100;              // TimeOut , 100 msecond
  dwBuf[5] = 0x00;             // digital output
  dwBuf[6] = 0;                // string debug
```

```
        dwBuf[7] = 1;                    // slot number

        getch();                         // push any key to continue
    wRetVal = DigitalOut_8K(dwBuf, fBuf, szSend, szReceive);

        Close_Com(COM2);
        return 0;
}
```

All the steps from programming to execution are the same as those in the section 5.1. The result of execution refers to Fig. 5-14.



```
Linux 2.4.21-rmk1 (localhost) (ttyp0)

moki login: root
Password:
MOKI 0.68


BusyBox v1.00-pre5 (2004.02.13-09:11+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

# i8kdio_8k.exe
DO of 8055 = 0xff
DI= 1
#
```

Fig. 5-14

5.6 I-8k Modules AIO Control Demo

I8000.c of Libi8k.a is the source file for i8k modules in slots of I-8000 controller. **Slot.c** of Libi8k.a is the source file for i8k modules in slots of the LinCon-8000. Therefore the functions for the i8k modules in LinCon-8000 slots and in the I-8000 controller slots are completely different.

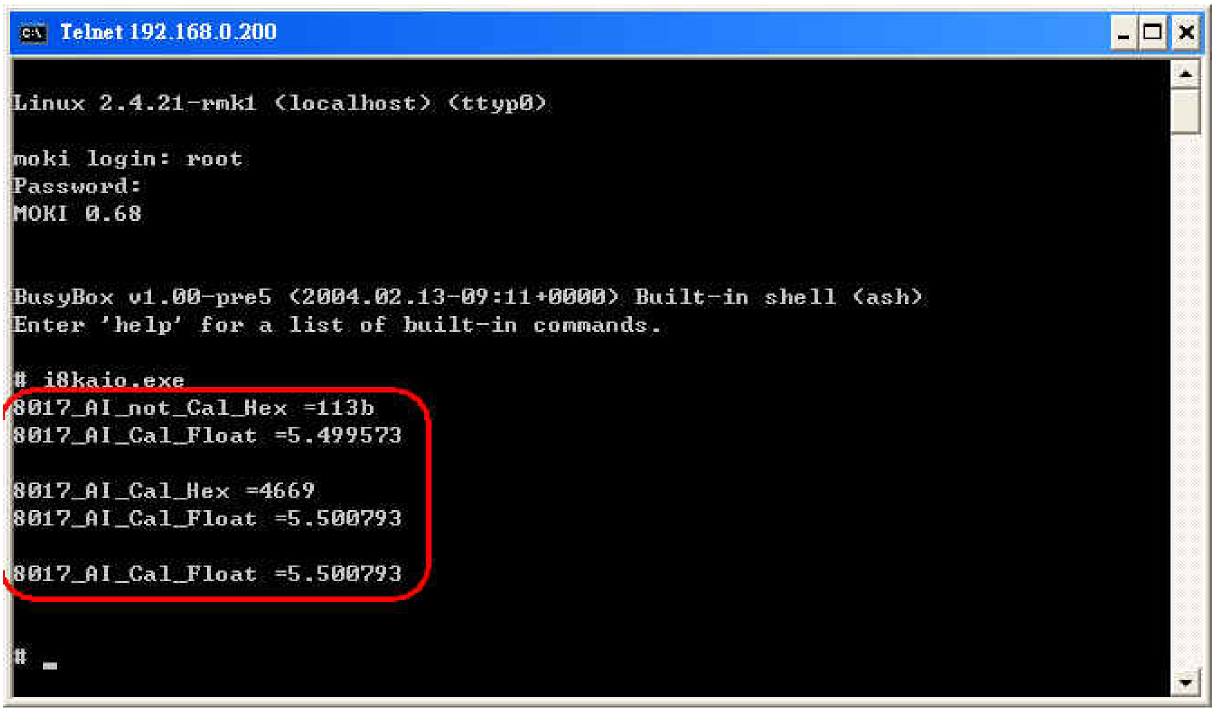
7.6.1 I-8k Modules in slots of LinCon-8000

In this section, this demo program – **i8kaio.c** will illustrate how to control the AI/AO with the I-8024 (4 AO channels) and I-8017 (8 AI channels) module and they are in slot 1 and slot 2 of the LinCon-8000 separately.

The address and baudrate in the LinCon-8000 are constant and they are 00 and 115200 separately. The result of this demo lets AO voltage channel 0 of I-8024 output 5.5V and AI channel 2 of I-8017H input. The source code of this demo program is as follows：

```
#include<stdio.h>
#include<stdlib.h>
#include "msw.h"

char szSend[80], szReceive[80];
DWORD dwBuf[12];
float fBuf[12];

/* --------------------------------------------------------------- */
int main()
{
    int i, wRetVal,j;
    float fAi;
    int hAi, chAi, Succ;
    int Arr_hAi[5];
    float Arr_fAi[5];

    //I-8024
    wRetVal = Open_Slot(1);
    if (wRetVal > 0) {
        printf("open Slot failed!\n");
        return (-1);
    }

    //I8024 Initial
    I8024_Initial(1);

    //I8024_AO Output
    I8024_VoltageOut(1,0,5.5);
    Close_Slot(1);

    /*************************************/
    //I-8017H
    wRetVal = Open_Slot(2);
    if (wRetVal > 0) {
        printf("open Slot failed!\n");
        return (-1);
    }

    //I8017H Initial
    I8017_Init(2);
    //I8017H _Channel Setup
    I8017_SetChannelGainMode(2,2,0,0);

    // First Method：Get AI Value
    hAi = I8017_GetCurAdChannel_Hex(2);     //Get Not-calibrated AI Hex Value
    printf("8017_AI_not_Cal_Hex =%x\n",hAi);
    fAi = HEX_TO_FLOAT_Cal(hAi,2,0);         //Not-calibrated AI Hex Value modify
to calibrated AI Float Value
    printf("8017_AI_Cal_Float =%f\n\n",fAi);

    // Second Method：Get AI Value
    hAi = I8017_GetCurAdChannel_Hex_Cal(2);   //Get Calibrated AI Hex Value
    printf("8017_AI_Cal_Hex =%x\n",hAi);
    fAi = CalHex_TO_FLOAT(hAi,0);                    //Calibrated AI Hex Value modify
to Calibrated AI Float Value
    printf("8017_AI_Cal_Float =%f\n\n",fAi);

    // Third Method：Get AI Value
    fAi = I8017_GetCurAdChannel_Float_Cal(2);   //Get Calibrated AI Float Value
    printf("8017_AI_Cal_Float =%f\n\n\n",fAi);

    Close_Slot(2);
    return 0;
}
```

All the steps from programming to execution are the same as those in the section 5.1. The result of execution refers to Fig. 5-15.



```
Linux 2.4.21-rmk1 (localhost) (ttyp0)

moki login: root
Password:
MOKI 0.68


BusyBox v1.00-pre5 (2004.02.13-09:11+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

# i8kaio.exe
8017_AI_not_Cal_Hex =113b
8017_AI_Cal_Float =5.499573

8017_AI_Cal_Hex =4669
8017_AI_Cal_Float =5.500793

8017_AI_Cal_Float =5.500793


# _
```

Fig. 5-15

5.6.2 I-8k Modules in slots of I-8000 Controller

In this section, this demo program – **i8kaio_8k.c** will introduce how to control the AI/AO with the I-8024 (4 AO channels) and I-8017 (8 AI channels) module and they are plugged into slot 0 and slot 1 of the I-8000 controller separately. Please follow the below steps to configure the hardware：

(1) Put the I-8024 and I-8017 modules in slot 0 and slot 1 of I-8000 controller.

(2) Install 8k232.exe or R232_300.exe to flash memory of I-8000 controller as firmware.

(3) Connect com2 of LinCon-8000 to com1 of I-8000 controller with RS-232 cable.

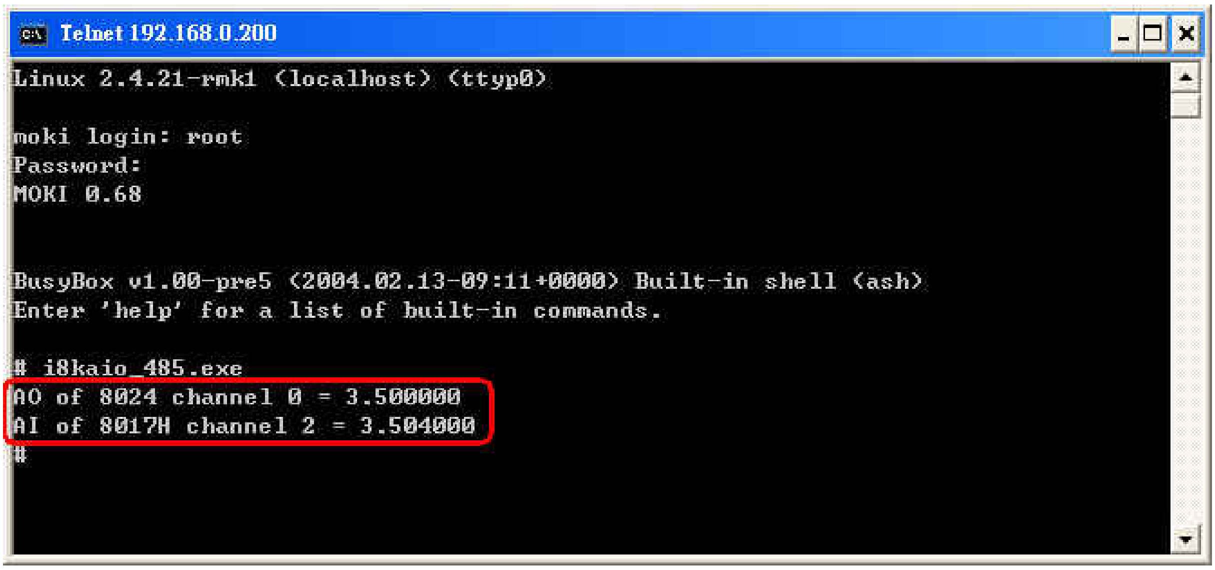
The address and baudrate of I-8000 controller are 01 and 115200 that can be modified by DCON Utility. The result of this demo lets AO voltage channel 0 of 8024 output 3.5V and AI channel 2 of 8017H input. The source code of this demo program is as follows：

```
#include<stdio.h>
#include<stdlib.h>
#include "msw.h"
```

```
char szSend[80], szReceive[80];
DWORD wBuf[12];
float fBuf[12];

/* -------------------------------------------------------------------- */
int main()
{
    int i,j, wRetVal;
    DWORD temp;

    wRetVal = Open_Com(COM2, 115200, Data8Bit, NonParity, OneStopBit);

    if (wRetVal > 0) {
        printf("open port failed!\n");
        return (-1);
    }

    //--- Analog output ----    ****     8024 -- AO    ****
    i = 0;
    wBuf[0] = 2;                // COM Port
    wBuf[1] = 0x01;             // Address
    wBuf[2] = 0x8024;           // ID
    wBuf[3] = 0;                // CheckSum disable
    wBuf[4] = 100;              // TimeOut , 100 msecond
    wBuf[5] = i;                // Channel No. of AO
    wBuf[6] = 0;                // string debug
    wBuf[7] = 0;                // Slot Number
    fBuf[0] = 3.5;

    wRetVal = AnalogOut_8K(wBuf, fBuf, szSend, szReceive);

    if (wRetVal)
        printf("AO of 8024 Error !, Error Code=%d\n", wRetVal);
    else
        printf("AO of 8024 channel %d = %f \n",i,fBuf[0]);

    //--- Analog Input ----    ****     8017H -- AI    ****
    j = 2;
```

```
        wBuf[0] = 2;                    // COM Port
        wBuf[1] = 0x01;                 // Address
        wBuf[2] = 0x8017;               // ID
        wBuf[3] = 0;                    // CheckSum disable
        wBuf[4] = 100;                  // TimeOut , 100 msecond
        wBuf[5] = j;                    // Channel of AI
        wBuf[6] = 0;                    // string debug
        wBuf[7] = 1;                    // Slot Number

        wRetVal = AnalogIn_8K(wBuf, fBuf, szSend, szReceive);
        if (wRetVal)
              printf("AI of 8017H Error !, Error Code=%d\n", wRetVal);
        else
              printf("AI of 8017H channel %d = %f \n",j,fBuf[0]);

        Close_Com(COM2);

        return 0;
}
```

All the steps from programming to execution are the same as those in the section 5.1. The result of execution refers to Fig. 5-16

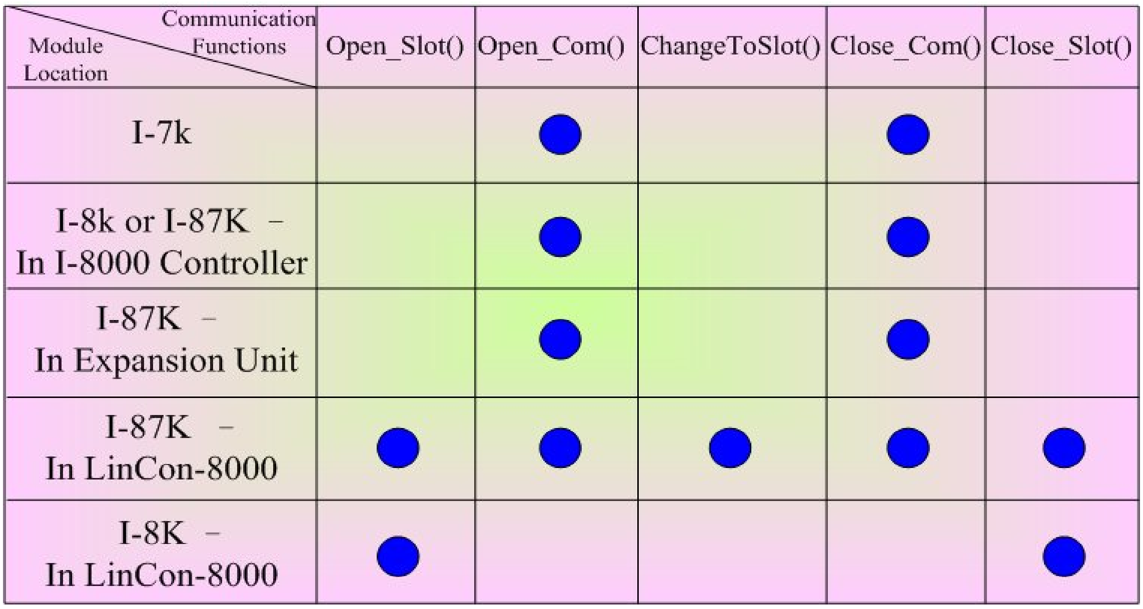


```
Linux 2.4.21-rmk1 (localhost) (ttyp0)

moki login: root
Password:
MOKI 0.68


BusyBox v1.00-pre5 (2004.02.13-09:11+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

# i8kaio_485.exe
AO of 8024 channel 0 = 3.500000
AI of 8017H channel 2 = 3.504000
#
```

Fig. 5-16

5.7 Conclusion of Functions and Source Files of Modules

Fig. 5-17 is the table of communication functions for the I-7000/I-8000/I-87000 modules in different locations. When using the ICPDAS modules in the LinCon-8000, this table will be helpful to let users understand which functions of communication should be used.



Communication Functions Module Location	Open_Slot()	Open_Com()	ChangeToSlot()	Close_Com()	Close_Slot()
I-7k		●		●	
I-8k or I-87K – In I-8000 Controller		●		●	
I-87K – In Expansion Unit		●		●	
I-87K – In LinCon-8000	●	●	●	●	●
I-8K – In LinCon-8000	●				●

Fig. 5-17

6. Additional Support

In this chapter, ICPDAS provides extra module supported and instructions to enhance LinCon-8000 functionality and affinity.

6.1 Support N-Port Module (i-8114, i-8112)

i-8114 and **i-8112** modules provide **four** and **two serial ports** separately. Users can insert them in the slots of the LinCon-8000. In this way, users can use more serial ports in the LinCon-8000 and the expanded maximum number of serial port in the LinCon-8000 is twenty-eight. Because it is multi-tasking in the LinCon, users can control all the serial ports simultaneously. **The serial port number** of i-8114 and i-8112 are figured in the fig.6-1 and fig.6-2 and it is **fixed** according to their slot position in the LinCon-8000.

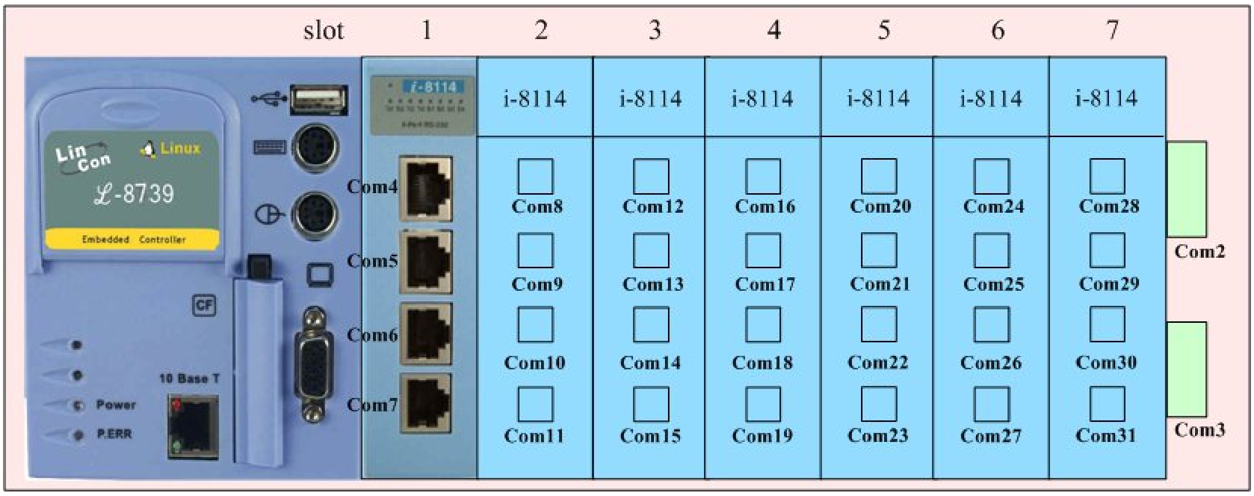


Fig.6-1

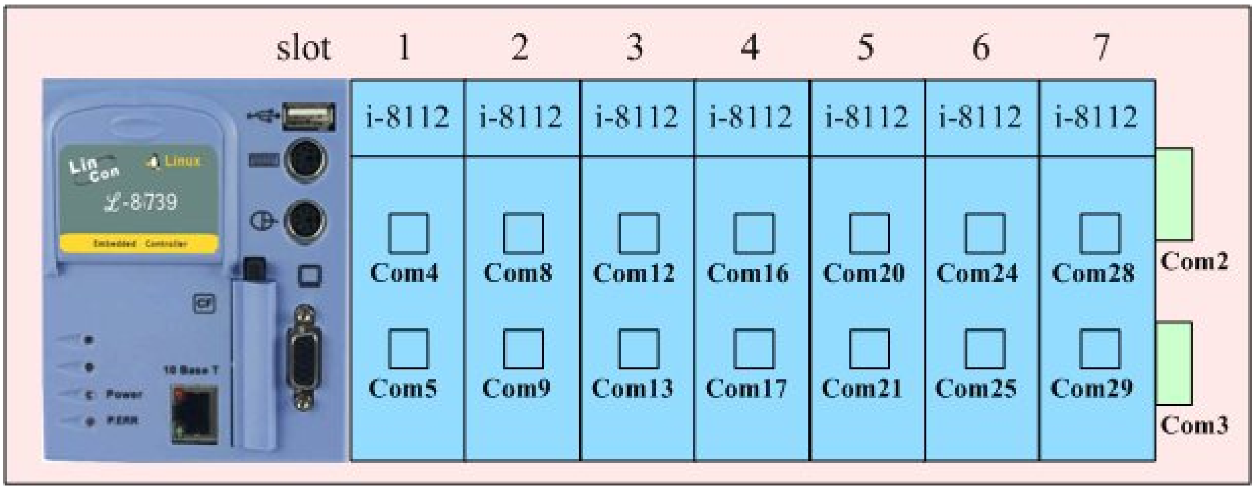


Fig.6-2

Fig.6-3 is the serial port number corresponding to the **device name** in the LinCon-8000.

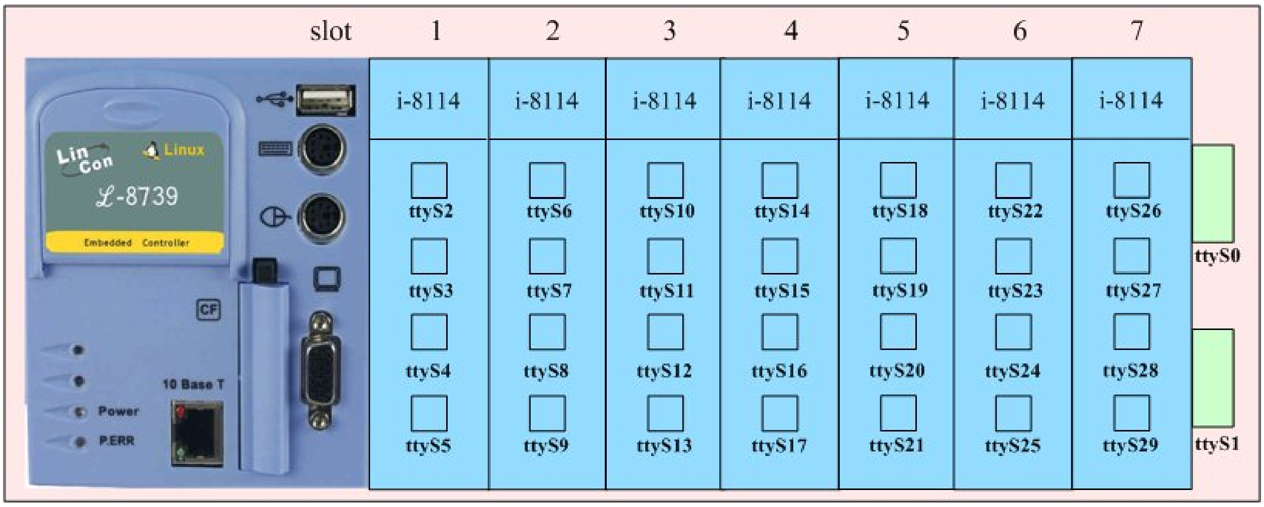


fig.6-3

The demo - **i7kdio_8114.c** will illustrate how to use the i-8114 module in the LinCon-8000. In this demo, we will control the I-7044 (8 DO channels and 4 DI channels) through the second serial port of i-8114 plugged in the slot 2 of the LinCon-8000. The address and baudrate of the I-7044 module in the RS-485 network are 02 and 115200 separately. Fig.6-4 is the control diagram.

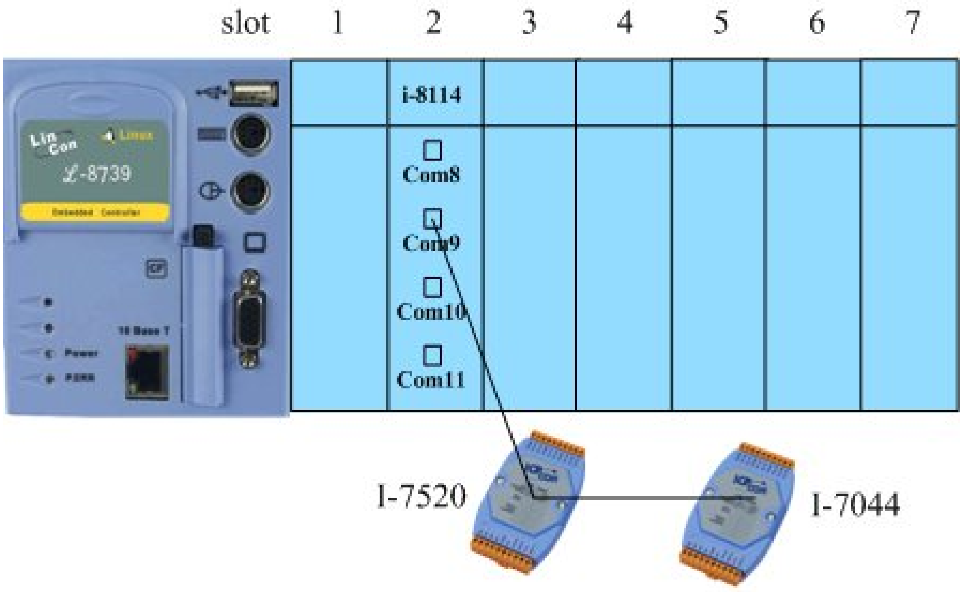


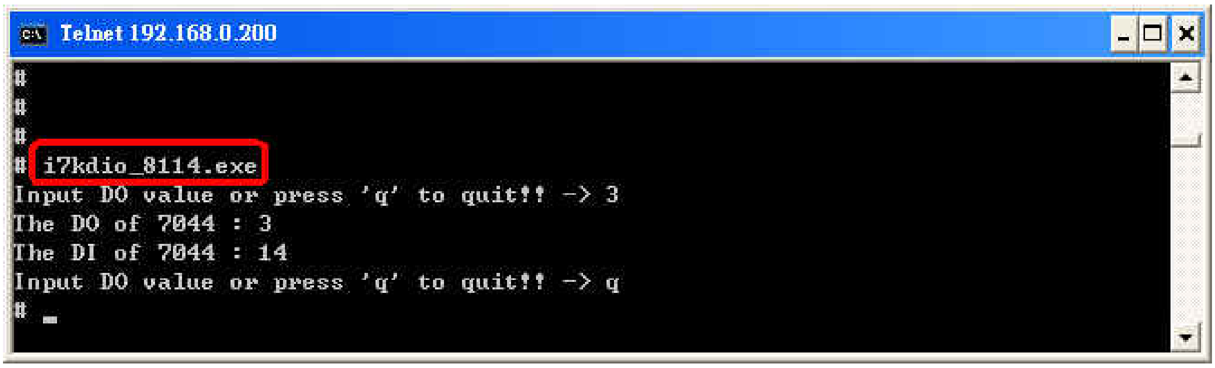
Fig.6-4

The result of this demo allows users to control which DO channels' state on and return the DI channels' state. The source code of this demo program is as follows:

```
#include<stdio.h>
#include<stdlib.h>
#include "msw.h"

char szSend[80], szReceive[80], ans;
WORD wBuf[12];
float fBuf[12];

/* ------------------------------------------------------------------ */
int main()
{
      int   wRetVal,j=0;
      char i[10];

   // Check Open_Com9 in I-8114
      wRetVal = Open_Com(COM9, 115200, Data8Bit, NonParity, OneStopBit);
      if (wRetVal > 0) {
            printf("open port failed!\n");
            return (-1);
      }
```

```
        //    *****    7044 DO & DI Parameter *******
        wBuf[0] = 9;                    // COM Port
        wBuf[1] = 0x02;                 // Address
        wBuf[2] = 0x7044;               // ID
        wBuf[3] = 0;                    // CheckSum disable
        wBuf[4] = 100;                  // TimeOut , 100 msecond
        wBuf[6] = 0;                    // string debug

    // 7044 DO
    while(j!=113) {
        printf("Input DO value or press 'q' to quit!! -> ");
        scanf("%s",i);

        if (i[0]=='q') {
            wBuf[5] = 0;            // All DO Channels Off
                wRetVal = DigitalOut(wBuf, fBuf, szSend, szReceive);
                break;
            }

        j=atoi(i);
        if (j>=0 & j<=255)
            wBuf[5] = j;       // DO Channels On
        else if (j>255)
            wBuf[5] = 255;
            wRetVal = DigitalOut(wBuf, fBuf, szSend, szReceive);
                if (wRetVal)
                    printf("DigitalOut_7044 Error !, Error Code=%d\n", wRetVal);

                printf("The DO of 7044 : %u \n", wBuf[5]);

        // 7044 DI
        DigitalIn(wBuf, fBuf, szSend, szReceive);
        printf("The DI of 7044 : %u \n", wBuf[5]);
        }
        Close_Com(COM9);

    return 0;
}
```

All the steps from programming to execution are the same as those in the section 5.1. The result of execution refers to Fig. 6.5.

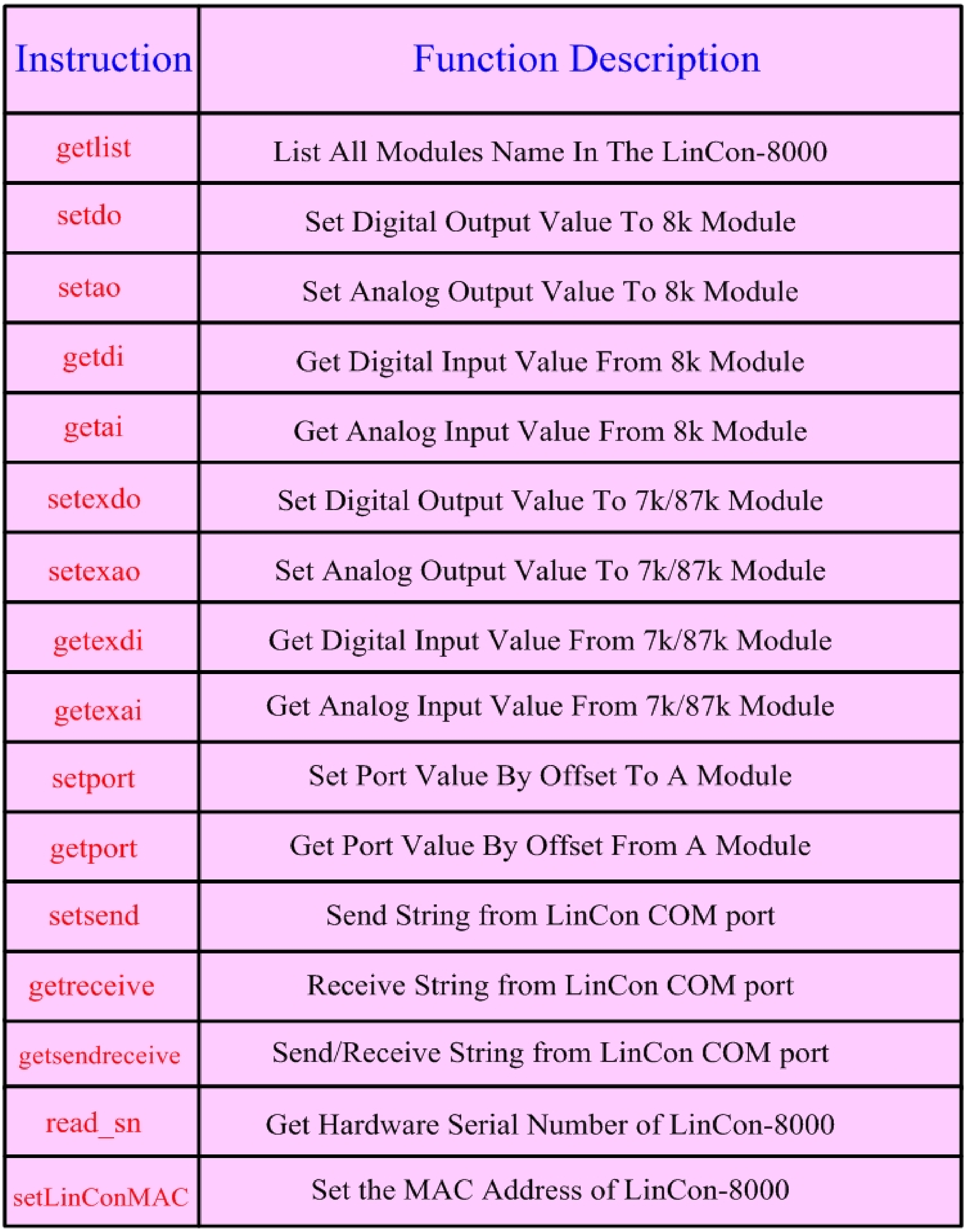


```
#
#
#
# i7kdio_8114.exe
Input DO value or press 'q' to quit!! -> 3
The DO of 7044 : 3
The DI of 7044 : 14
Input DO value or press 'q' to quit!! -> q
# 
```

Fig. 6.5

6.2 I-talk Utility

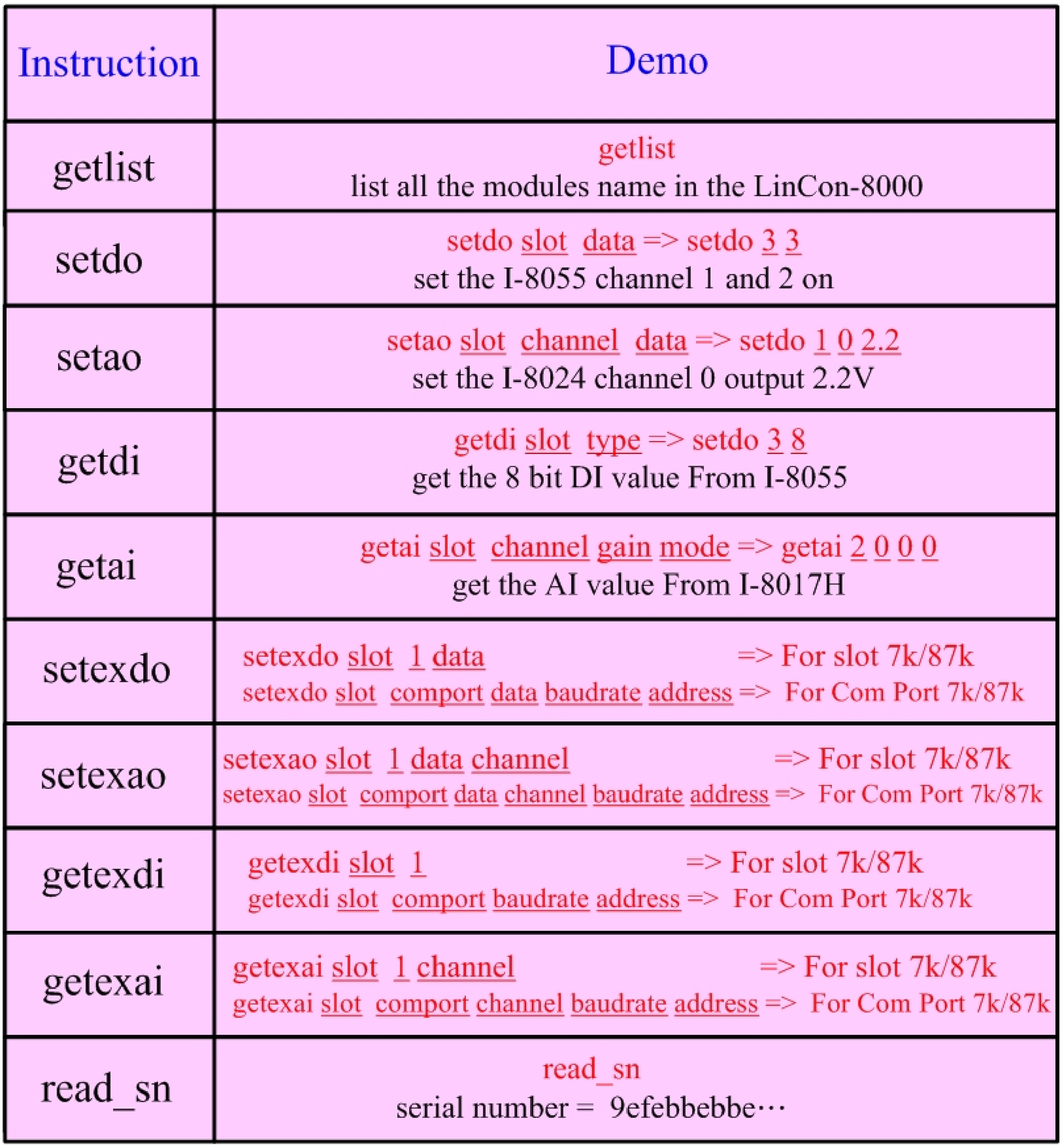
The **i-Talk utility** provides **sixteen instructions** that make it convenient for users to access the modules and hardware in the LinCon-8000 and they are placed in the path — **/usr/local/bin**. Fig. 6-6 describes the functions of i-Talk utility.



Instruction	Function Description
getlist	List All Modules Name In The LinCon-8000
setdo	Set Digital Output Value To 8k Module
setao	Set Analog Output Value To 8k Module
getdi	Get Digital Input Value From 8k Module
getai	Get Analog Input Value From 8k Module
setexdo	Set Digital Output Value To 7k/87k Module
setexao	Set Analog Output Value To 7k/87k Module
getexdi	Get Digital Input Value From 7k/87k Module
getexai	Get Analog Input Value From 7k/87k Module
setport	Set Port Value By Offset To A Module
getport	Get Port Value By Offset From A Module
setsend	Send String from LinCon COM port
getreceive	Receive String from LinCon COM port
getsendreceive	Send/Receive String from LinCon COM port
read_sn	Get Hardware Serial Number of LinCon-8000
setLinConMAC	Set the MAC Address of LinCon-8000

Fig. 6-6

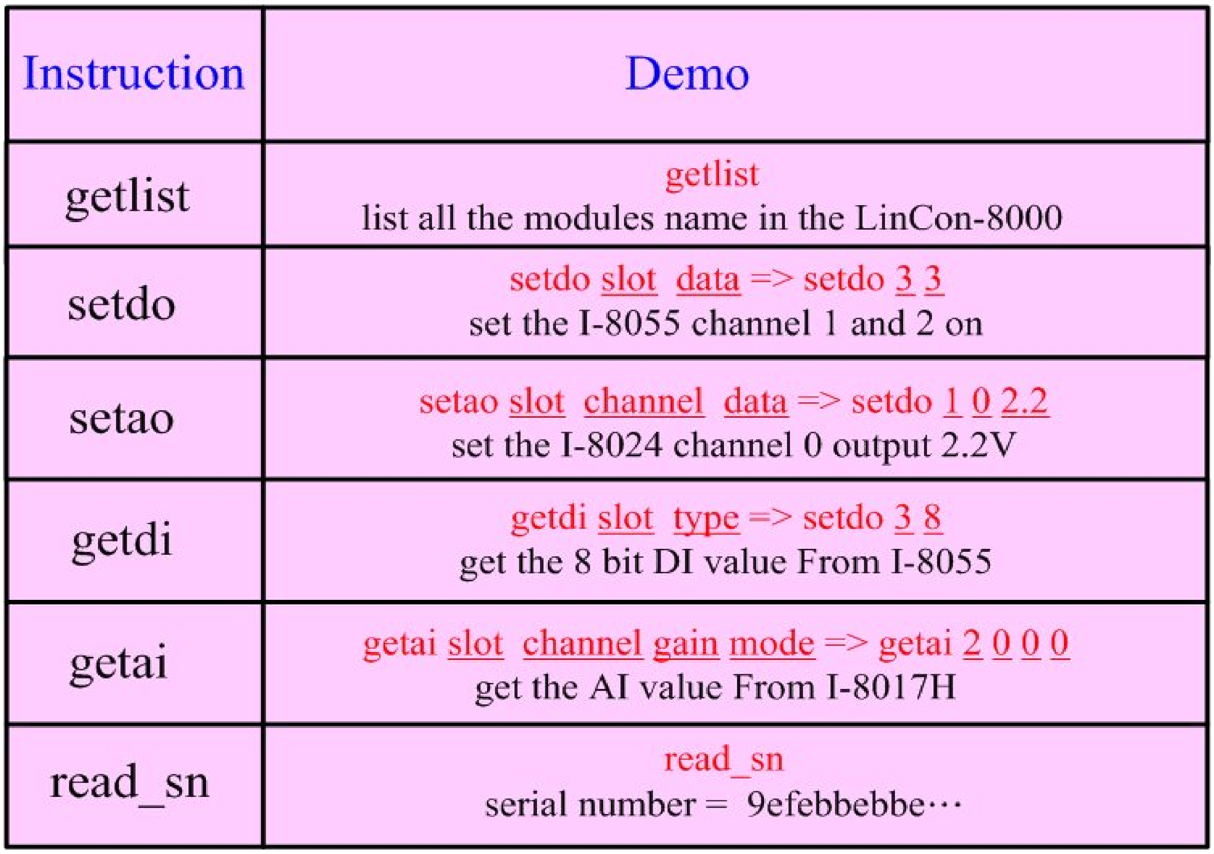
Fig. 6-7 lists the demo that show how to use the I-talk utility. In the demo,the **I-8024** (AO Module) 、**I-8017H** (AI Module) and **I-8055** (DIO Module) are all used and they are plugged into the slots 1、2 and 3 of the LinCon seperately.



Instruction	Demo
getlist	getlist list all the modules name in the LinCon-8000
setdo	setdo slot data => setdo 3 3 set the I-8055 channel 1 and 2 on
setao	setao slot channel data => setdo 1 0 2.2 set the I-8024 channel 0 output 2.2V
getdi	getdi slot type => setdo 3 8 get the 8 bit DI value From I-8055
getai	getai slot channel gain mode => getai 2 0 0 0 get the AI value From I-8017H
setexdo	setexdo slot 1 data => For slot 7k/87k setexdo slot comport data baudrate address => For Com Port 7k/87k
setexao	setexao slot 1 data channel => For slot 7k/87k setexao slot comport data channel baudrate address => For Com Port 7k/87k
getexdi	getexdi slot 1 => For slot 7k/87k getexdi slot comport baudrate address => For Com Port 7k/87k
getexai	getexai slot 1 channel => For slot 7k/87k getexai slot comport channel baudrate address => For Com Port 7k/87k
read_sn	read_sn serial number = 9efebbebbe···

Fig. 6-7

Users can also type in the instructions name and it will show the instructions usage.



Instruction	Demo
getlist	getlist list all the modules name in the LinCon-8000
setdo	setdo slot data => setdo 3 3 set the I-8055 channel 1 and 2 on
setao	setao slot channel data => setdo 1 0 2.2 set the I-8024 channel 0 output 2.2V
getdi	getdi slot type => setdo 3 8 get the 8 bit DI value From I-8055
getai	getai slot channel gain mode => getai 2 0 0 0 get the AI value From I-8017H
read_sn	read_sn serial number = 9efebbebbe···

Fig. 6-7

Users can also type the name of instructions and it will show the usage of instructions.

6.3 Crash Free Support

If it is unfortunate, the LinCon-8000 is crashed and can’t reboot. Please follow the steps to make the LinCon recover the normal state :

(1) Reboot the LinCon-8000.

(2) When the “ **boot up screen** “ shows up, connect the pin 1 and pin 5 of the second row of any slot with a wire immediately. (Refer to the Fig. 6-8)

(3) The LinCon-8000 will boot up with its correct “ **/etc** “ directory built in and user’s “ **/etc** “ directory will be placed to the path – “ **/tmp/etc** “. When the LinCon-8000 boot up successfully, users should copy the correct files in the /etc to the path - /tmp/etc or correct the files in the /tmp/etc.

(4) After the correction, remove the connection wire and reboot LinCon-8000 again.

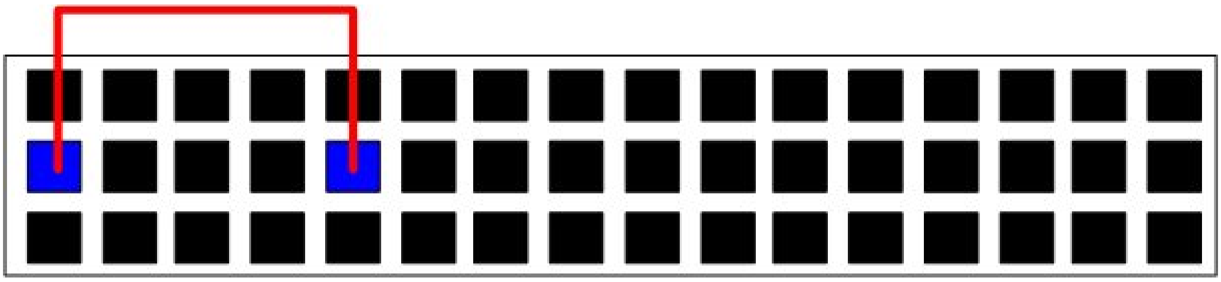


Fig. 6-8

6.4 GUI Funtion Support

Now **“ X-window “** is supported in the LinCon-8000 and when the LinCon-8000 boot up, the GUI like **“ Windows screen “** will show up. The most important thing is that users can write GUI programs and run them in the LinCon-8000. The GUI Library in the LinCon-8000 is provided with **GTK+ v1.2 & v2.0** Library. Therefore users can design their own **“ SCADA “** screen by the GTK+ Library in the LinCon-8000. In the meanwhile, we provide some GUI demo programs to control I/O modules of ICPDAS and assist users to develop own GUI programs quickly. These demo programs are placed in the path – **C:\cygwin\LinCon8k\examples\gui** after users install the LinCon-8000 SDK. (Refer to the Fig. 6-9)

Except GTK+ GUI Function, **“ Java GUI “** is also supported in the LinCon-8000. So if users are familiar with Java, users can also use Java to develop own GUI programs. But just Awe and Swing v1.1 elements below are supported in the LinCon-8000. To execute Java GUI program – Stylepad.jar in the LinCon-8000, users just type in **“ java -jar Stylepad.jar -cp .:Stylepad.jar “.** Then it will take some time to run up the Java GUI program.

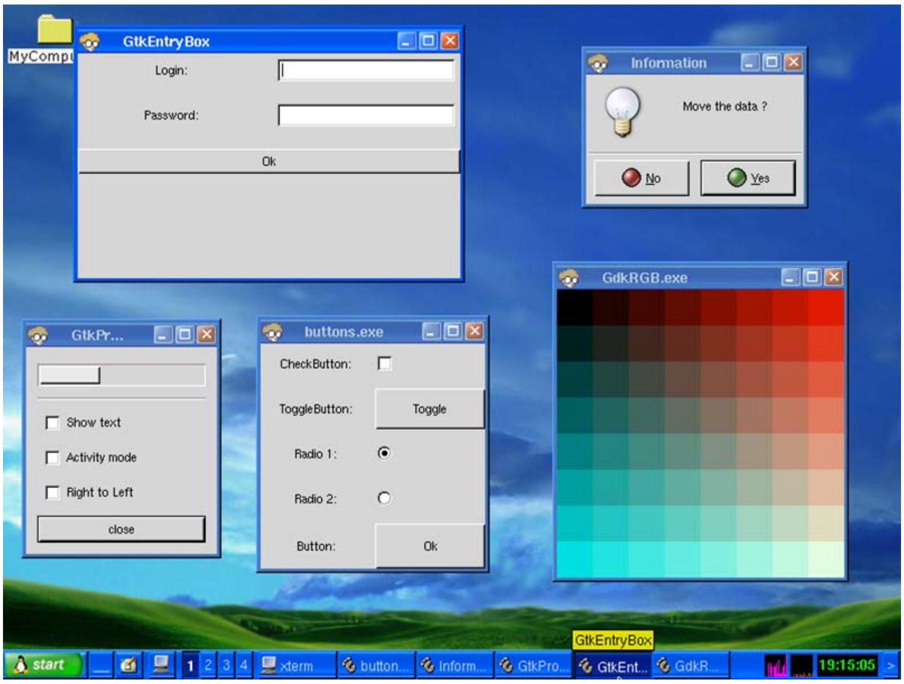


Fig. 6-9

6.4.1 How to boot LinCon-8000 without loading X-window

LinCon-8000 can boot without loading X-window by the steps as follows :

(1) Type “ **cd /etc/rc2.d** “ to into default run level.

(2) Type ” **ls -al** ” to see the S98Xserver link into ../init.d/startx.

(3) Type ” **mv S98Xserver xS98Xserver** “ to rename the S98Xserver for turn off X-window. Then exit and reboot LinCon-8000.

6.4.2 Enabling X-window load at boot time

If you type the “ **ls -al /etc/rc2.d** “ that can fine the link about ../init.d/startx, and then type the “ **mv xS98Xserver S98Xserver** “ to rename the xS98Xserver for turn on X-window or else if you can’t fine any link about ../init.d/startx, and please follow the below steps :

(1) Type “ **cd /etc/rc2.d** “ to into default run level.

(2) Type ” **ln -s ../init.d/startx /etc/rc2.d/S98Xserver** “ to make a symbolic link into the script file of X-window for turn on X-window. Then exit and reboot LinCon-8000.

6.5 ScreenShot Support

There is a screenshot program — **“ fbshot “** built in to let users to catch the LinCon-8000 screen conveniently. Users just type in **“ fbshot -d /dev/fb0 /mnt/hda/catch1.png “** and the screen will be catched and saved to the file — /mnt/hda/catch1.png. If users want to take a look the picture, just type in **“ iv /mnt/hda/catch1.png “.** If users want to know the detailed parameters of fbshot, just type in **“ fbshot --help ”**.

6.6 WebCAM Support

WebCAM is also supported in the LinCon-8000 and Logitech brand works successfully now. Other brands will need to do a test. Please follow the steps to make the Webcam work smoothly :

(1) Connect the webcam to the LinCon-8000 with **“ USB Interface “**.

(2) Reboot the LinCon-8000.

(3) Open a **“ Command Prompt “**. Type in **“ insmod pwcx.o “** to load the gqcam program decompressor and then type in **“ gqcam “** to see the webcam screen. If users want to know the detailed parameters of gqcam, just type in **“ gqcam --help ”**.

If users want to catch the picture through webcam, users can use gqcam program to do that. Please follow the steps as below :

(1) Click **“ File/Save Image… “**

(2) At **“ Gqcam: Save Image “** screen, input the path and file name in the **“ File Field “** and then click “ OK “ button.

6.7 Network Support

There are many network functions already built in the LinCon-8000. Here are the network functions supported in the LinCon-8000 :

(1) Support UPnP :

UPnP is **“ Universal Plug and Play “** and allows automatic discovery and control of services available on the network from other devices without user intervention. Devices that act as servers can advertise their services to clients. Client systems, known as control points, can search for specific services on the network. When they find the devices with the desiredservices, the control points can retrieve detailed descriptions of the devices and services and interact from that point on.

(2) Support VPN

VPN is **“ Virtual Private Network “** and describes a network that includes secure remote access for client computers. It can be explained best by looking at its parts. **“ Virtual “** describes the fact that the network doesn't need to be physically connected directly. The **“ Private “** confirms that the data is encrypted and can only be viewed by a defined group. The last word, **“ Network “**, means that the users configured for VPN can be connected and share files or information. So it's extremely difficult for anyone to snoop on confidential information through VPN. (Refer to the Fig. 6-10)

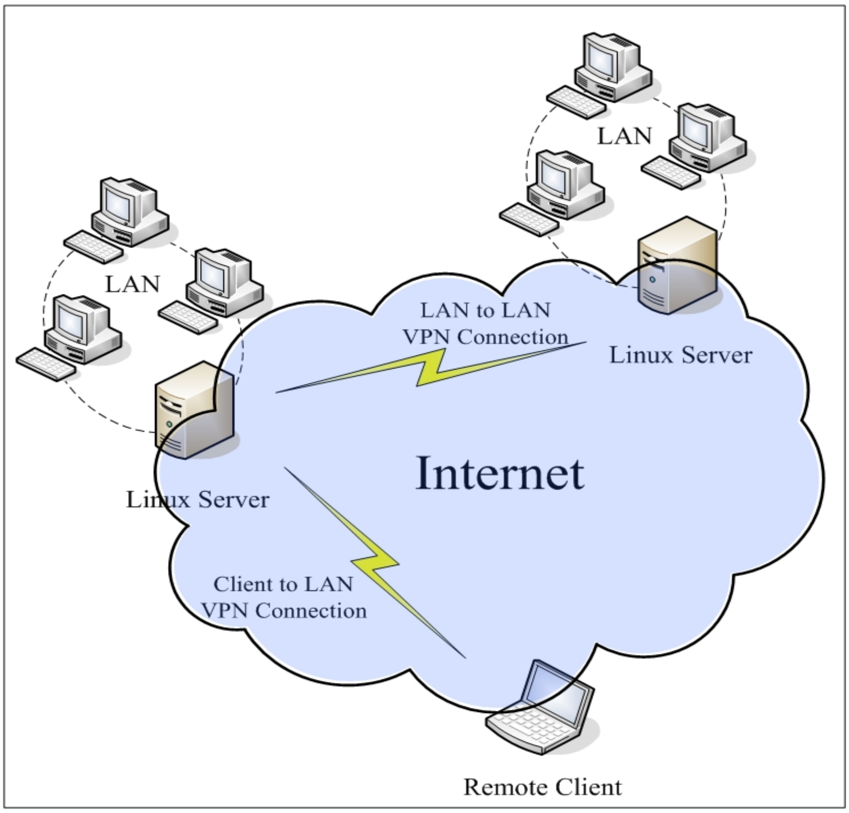


Fig. 6-10

(3) Support QoS

QoS is **“ Quality of Service “**. It means when the kernel has several packets to send out over a network device, it has to decide which ones to send first, which ones to delay, and which ones to drop. With Linux QoS subsystem, it is possible to make very flexible traffic control. Let users be able to control flow rate of assigned port to improve the network quality.

(4) Support Wireless LAN

“ Wireless communication “ is a networking technology allowing the connection of computers without any wires and cables, mostly using **radio** technology (and sometime **infrared**). It's called LAN because the range targeted is small (generally within an office, a building, a store, a small campus, a house...). This technology is slowly growing and Linux is able to take advantage of some of the wireless networks available.

(5) Support Dual LAN

Dual LAN means that users can combine wireless and cable network together through LinCon-8000. Therefore the communication between Cable LAN and Wireless LAN. If one of these LANs can connect to internet, then all the PC can connect to internet. (Refer to Fig. 6-11)

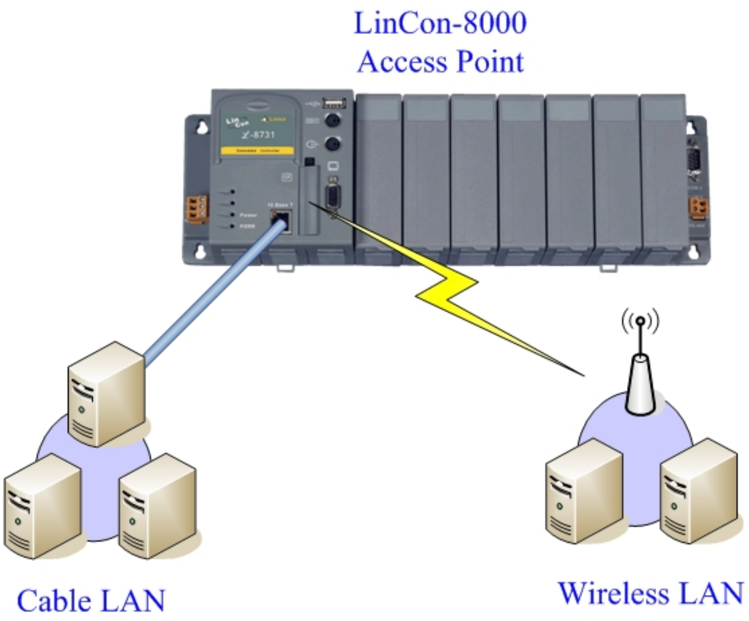


Fig. 6-11

(6)Support BlueTooth

The Bluetooth wireless technology is a worldwide specification for a small-form factor, low-cost radio solution that provides links between mobile computers, mobile phones, other portable handheld devices, and connectivity to the Internet. Now **“ BlueZ “** is built in the LinCon-8000 and provides support for the core Bluetooth layers and protocols. It is flexible, efficient and uses a modular implementation.

(7) Support Modem / GPRS / ADSL

LinCon-8000 can be connected to the Internet with “ Modem “, “ GPRS “ or “ ADSL “ mode. The setup method is described separately as follows :

[Modem]

[GPRS]

The default GPRS baudrate is **“ 115200 “** in the LinCon, so if users finish the setting of gprs modem and connect the gprs modem to the COM 2 of LinCon-8000, just type in **“ pppd call wavecom “** and then LinCon-8000 will be connected to the internet automatically. Remember that the network interface card of LinCon should stop first, just type in **“ ifdown eth0 “** to stop it. If users type in “ ifconfig “ will see the **“ ppp0 “** option.

[ADSL]

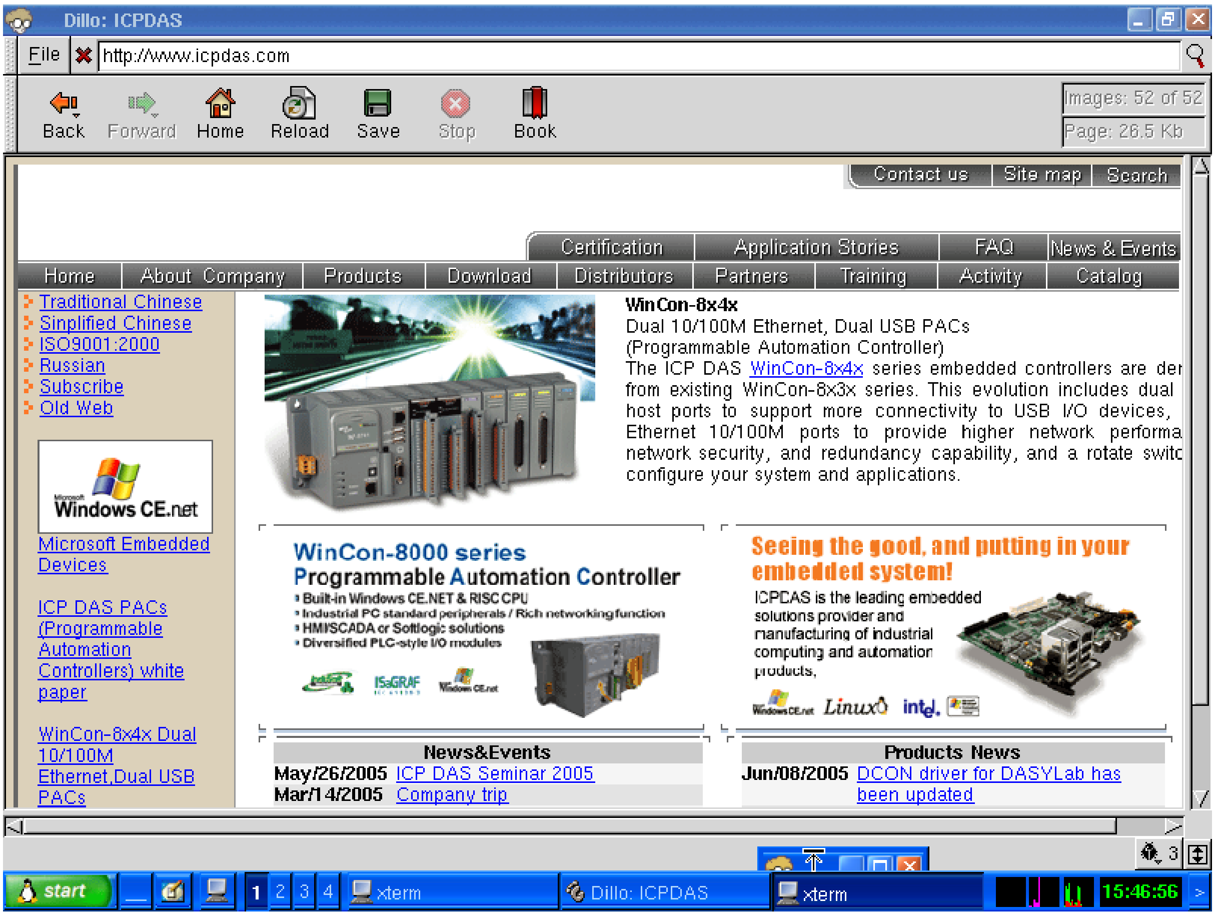
Users need to type in **“ adsl-setup “** first to setup ADSL options. After that, users need to type in **“ adsl-connect “** to make LinCon-8000 connect to the internet. If users want to stop adsl connection, just type in **“ adsl-stop “**.

(8) Support Firewall (iptables function)

A firewall can controls outside access to a local network, locking out intruders to ensure your systems and data safe on the inside, even against an intentional attack from outside network.

(9) Provide Web Browser

Users can see the Web Page by using the Web Browser built in the LinCon-8000. Just type in **“ dillo “** to open the web browser and input the web site address. (Refer to Fig 6-12)



WinCon-8x4x
Dual 10/100M Ethernet, Dual USB PACs
(Programmable Automation Controller)
The ICP DAS WinCon-8x4x series embedded controllers are der from existing WinCon-8x3x series. This evolution includes dual host ports to support more connectivity to USB I/O devices, Ethernet 10/100M ports to provide higher network performa network security, and redundancy capability, and a rotate switc configure your system and applications.

- Built-in Windows CE.NET & RISC CPU
- Industrial PC standard peripherals / Rich networking function
- HMI/SCADA or Softlogic solutions
- Diversified PLC-style I/O modules

ICPDAS is the leading embedded solutions provider and manufacturing of industrial computing and automation products.

News&Events	
May/26/2005	ICP DAS Seminar 2005
Mar/14/2005	Company trip

Products News	
Jun/08/2005	DCON driver for DASYLab has been updated

Fig 6-12

(10) Provide Apache Server

The Web Server — “ **Apache Server** “ has been built in the LinCon-8000. These files are placed in the path — **/mnt/hda/opt/apache2**. Users can type in like “ **http://192.168.0.200** “ in web browser with PC to attempt to connect to the web server in the LinCon-8000. If it returns a successful web page, it means the web server in the LinCon-8000 has been started.

These files are placed in the CF Card. So if users want to use the Web Server in the LinCon-8000, users must plug in CF Card in the LinCon-8000 and the “ **apache2** ” directory must be in the “ **/opt** “ directory of CF Card. Then when users boot up the LinCon-8000, the Web Server will start automatically.

6.8 Other Optional Function

These optional functions are listed below all supported in the LinCon-8000. Users can choose which function to be used in the LinCon-8000 and just copy the corresponding file directory to the “ opt “ directory of CF Card. Then reboot LinCon-8000 and the function users choose will work automatically.

(1) Support MySQL

MySQL is a small database server and it is “ Relational DataBase Management System (RDBMS) “. By using MySQL, users can add or delete data easily and it is open source and supports many platforms, like UNIX、Linux or Windows operating system. If users want to use MySQL in the LinCon-8000, remember to copy the “ mysql “ directory to the “ opt “ directory of CF Card and reboot LinCon-8000.

(2) Support PHP

PHP is a kind of “ open source script language “ and used to design active web page. When PHP combined with MySQL are cross-platform. It means that users can develop in Windows and serve on a Linux platform. (Refer to Fig 11-11)

PHP has been built in the LinCon-8000 Kernel so users just boot up LinCon-8000 and can use PHP directly in the LinCon-8000.

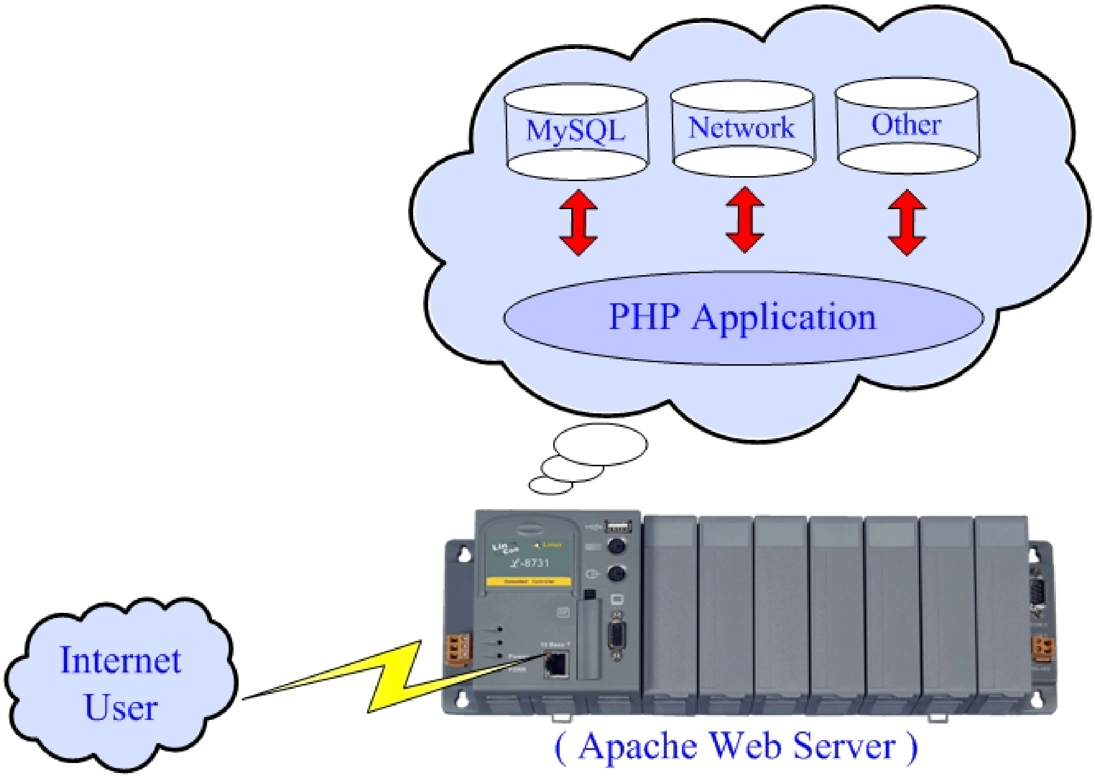


Fig 11-11

(3) Support Perl

Perl（Practical Extraction and Report Language）is also a “ open source script language “ and has been built in the LinCon-8000 Kernel so users just boot up LinCon-8000 and can use Perl directly in the LinCon-8000.