# UC-7112/7110 User's Manual

**Fourth Edition, January 2007**

***www.moxa.com/product***

MOXA Systems Co., Ltd.
Tel: +886-2-8919-1711
Fax: +886-2-8919-1722
Web: www.moxa.com

**MOXA Technical Support**
Worldwide: support@moxa.com
The Americas support@usa.moxa.com

# UC-7112/7110 User's Manual



## Copyright Notice



## Trademarks



## Disclaimer

Information in this document is subject to change without notice and does not represent a commitment on the part of MOXA.

MOXA provides this document "as is," without warranty of any kind, either expressed or implied, including, but not limited to, its particular purpose. MOXA reserves the right to make improvements and/or changes to this manual, or to the products and/or the programs described in this manual, at any time.

Information provided in this manual is intended to be accurate and reliable. However, MOXA assumes no responsibility for its use, or for any infringements on the rights of third parties that may result from its use.

This product might include unintentional technical or typographical errors. Changes are periodically made to the information herein to correct such errors, and these changes are incorporated into new editions of the publication.

# Table of Contents

# 1

# Introduction

The MOXA UC-7112/7110 Series of mini RISC-based ready-to-run embedded computers feature dual 10/100 Mbps Ethernet ports and two RS-232/422/485 serial ports in a built-in µClinux ARM9 box. In addition, the UC-7112 provides an internal SD socket for storage expansion, offers high performance communication and unlimited storage in a super compact, palm-size box. The UC-7112 and UC-7110 are the right solutions for embedded applications that use a lot of memory, but that must be housed in a small physical space without sacrificing performance.

This chapter covers the following topics:

# Overview

The UC-7112/7110 Series of mini RISC-based communication platforms are ideal for your embedded applications. The UC-7112/7110 comes with 2 RS-232/422/485 serial ports and dual 10/100 Mbps Ethernet LAN ports to provide users with a versatile communication platform.

The UC-7112/7110 embedded computers use the ARM9 RISC CPU. Unlike the X86 CPU, which uses a CISC design, the ARM9's RISC design architecture and modern semiconductor technology provide the UC-7112/7110 with a powerful computing engine and communication functions, but without generating too much heat. The built-in 8 MB NOR Flash ROM and 16 MB SDRAM give you enough storage capacity, and an additional SD socket provides you with flexible storage expansion to run a wide range of applications. The dual LAN ports built into the ARM9 make the UC-7112/7110 an ideal communication platform for data acquisition and protocol conversion applications, and the two RS-232/422/485 serial ports allow you to connect a variety of serial devices.

The pre-installed μClinux operating system provides an open software operating system for software program development. Software written for desktop PCs is easily ported to the UC-7112/7110 with a GNU cross complier, so that you will not need to spend time modifying existing software code. The operating system, device drivers, and your own software can all be stored in the UC-7112/7110's Flash memory.

# Package Checklist

The UC-7112/7110 Series models currently available are:

**UC-7112-LX**

Mini RISC-based Ready-to-Run Embedded Computer with 2 Serial Ports, Dual Ethernet, SD, μClinux OS

**UC-7110-LX**

Mini RISC-based Ready-to-Run Embedded Computer with 2 Serial Ports, Dual Ethernet, μClinux OS. UC-7112/7110 Series products are shipped with the following items:

- 1 UC-7112 or UC-7110
- UC-7112/7110 Quick Installation Guide
- Document and Software CD
- Ethernet cross-over cable: RJ45 to RJ45, 100 cm
- Console port cable: CBL-4PINDB9F-100 (4-pin header to female DB9 cable, 100 cm)
- Universal Power Adaptor
- Product Warranty Statement

*Optional Accessories*

- DK-35A DIN-Rail Mounting Kit (35 mm)

NOTE*: Please notify your sales representative if any of the above items are missing or damaged.*

# Product Features

UC-7112/7110 Series products have the following features:

- Mini controller with ready-to-run platform for customized applications
- 32-bit ARM9 RISC microcontroller
- On-board 16 MB RAM, 8 MB Flash ROM
- Two RS-232/422/485 serial ports
- Dual 10/100 Mbps Ethernet
- SD expansion slot for storage expansion (UC-7112 only)
- µClinux-ready communication platform
- DIN-Rail or wall mounting installation
- Robust fanless design

# Product Specifications

## Hardware Specifications

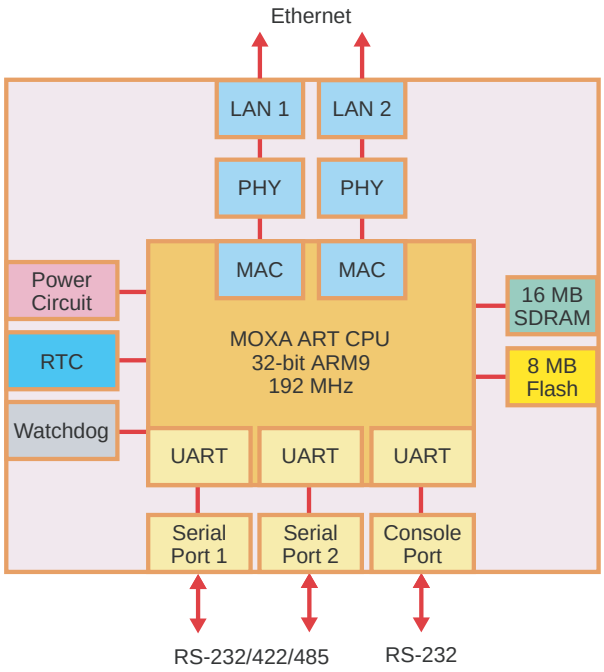
| | |
|---|---|
| **CPU** | MOXA ARM9-based 32-bit RISC CPU, 192 MHz |
| **RAM** | 16 MB (12 MB of user programmable space) |
| **Flash** | 8 MB (4 MB of user programmable space) |
| **Storage Expansion** | Internal SD socket x 1 for SD memory card<br>(only for UC-7112) |
| **LAN** | Auto-sensing 10/100 Mbps x 2 |
| **LAN Protection** | Built-in 1.5 KV magnetic isolation |
| **Serial Ports** | Both RS-232/422/485 ports support:<br>RS-232 signals: TxD, RxD, DTR, DSR, RTS, CTS, DCD, GND<br>RS-422 signals: TxD+, TxD-, RxD+, RxD-, GND<br>4-wire RS-485 signals: TxD+, TxD-, RxD+, RxD-, GND<br>2-wire RS-485 signals: Data+, Data-, GND |
| **Serial Protection** | 15 KV ESD for all signals |
| **Data bits** | 5, 6, 7, 8 |
| **Stop bit(s)** | 1, 1.5, 2 |
| **Parity** | None, Even, Odd, Space, Mark |
| **Flow Control** | RTC/CTS, XON/XOFF, RS-485 ADDCTM |
| **Speed** | 50 bps to 921.6 Kbps; Any Baudrate supported |
| **Watchdog Timer** | Yes |
| **Real Time Clock** | Yes |
| **Buzzer** | Yes |
| **Console Port** | 3-wire RS-232 (Tx, Rx, GND) (19200, n, 8 , 1) |
| **LEDs** | Ready<br>Serial Tx, Rx (2 of each)<br>LAN 10/100 (one on each LAN connector) |
| **Dimensions (WxDxH)** | 77 x 111 x 26 mm (3.03 x 4.37 x 1.02 in) |
| **Gross Weight** | 190g |
| **Power input** | 12-48 VDC |
| **Power Consumption** | 340 mA @ 12 VDC, 4.5W |

| | |
|---|---|
| **Operating temperature** | -10 to 60°C, (14 to 140°F), 5 to 95% RH |
| **Storage temperature** | -20 to 80°C, (-4 to 176°F), 5 to 95% RH |
| **Regulatory Approvals** | EMC: FCC Class A, CE Class A<br>Safety: UL, CUL, TÜV |
| **Warranty** | 5 years |

## Software Specifications

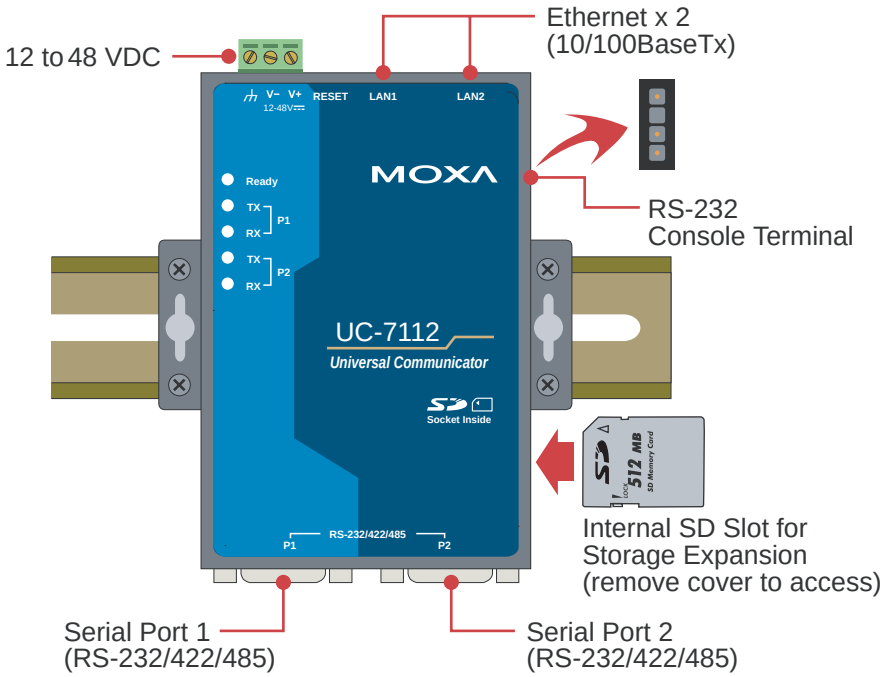
| | |
|---|---|
| **Kernel** | µClinux Kernel 2.6.9<br>Support for dynamic driver module load / unload |
| **Protocol Stack** | ARP, ICMP, IPV4, TCP, UDP, FTP, Telnet, SNMP V1/V2c, HTTP, CHAP, PAP, DHCP, NTP, NFS V2/V3, SMTP, Telnet, FTP, PPP, PPPoE |
| **File System** | JFFS2 for Kernel, Root File System (Read Only) and User Directory (Read / Write) |
| **Msh** | Minix shell command |
| **pppd** | Dial in/out over serial port daemon |
| **PPPoE** | Point-to-Point over Ethernet daemon |
| **snmpd** | SNMP V1/V2c Agent daemon |
| **busybox** | Linux normal command utility |
| **Tinylogin** | login and user manager utility |
| **Telnetd** | Telnet server daemon |
| **telnet** | Telnet client program |
| **inetd** | TCP server manager program |
| **ftpd** | FTP server program |
| **ftp** | FTP client program |
| **boa** | Web server daemon |
| **ntpdate** | Network Time Protocol client utility |
| **Tool Chain** | |
| **Linux Tool Chain** | Arm-elf-gcc (V2.95.3): C/C++ PC Cross Compiler<br>uClibc (V0.9.26): POSIX standard C library |
| **Windows Tool Chain** | Arm-elf-gcc (V2.95.3): C/C++ PC Cross Compiler<br>uClibc (V0.9.26): POSIX standard C library |
| **UC Finder** | UC's LAN IP broadcast searching utility for Windows and Linux |

# Hardware Block Diagram



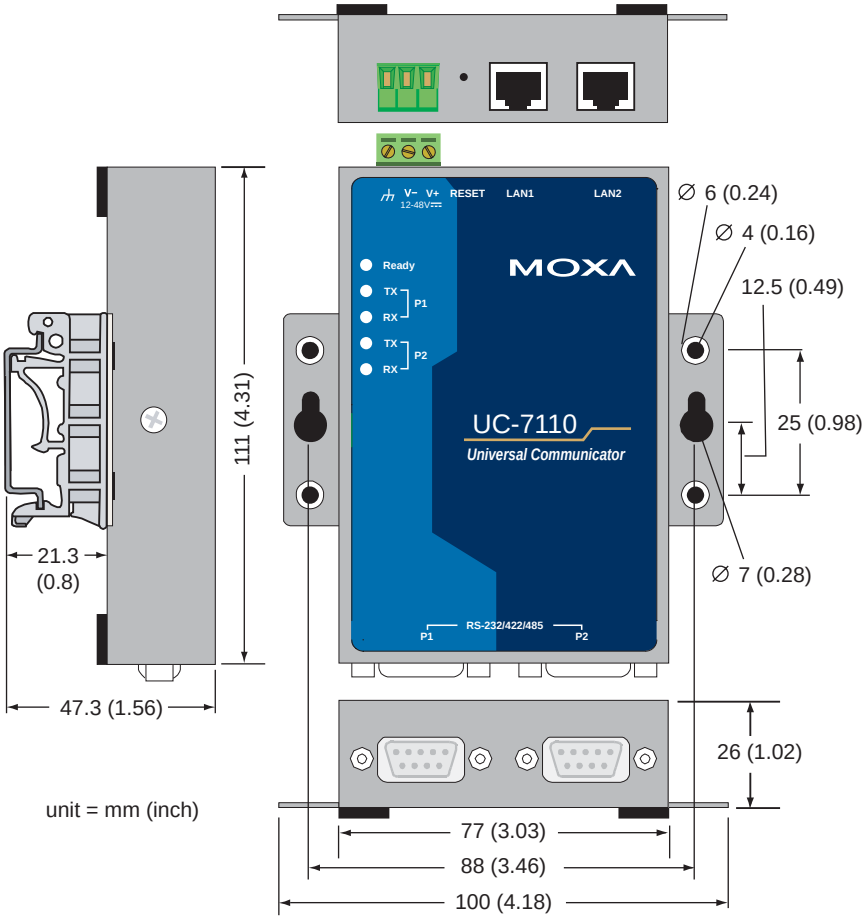

# Appearance

The top view of the UC-7112 is shown in the following figure. The UC-7110 looks similar, except that the UC-7110 does not have an internal SD slot.



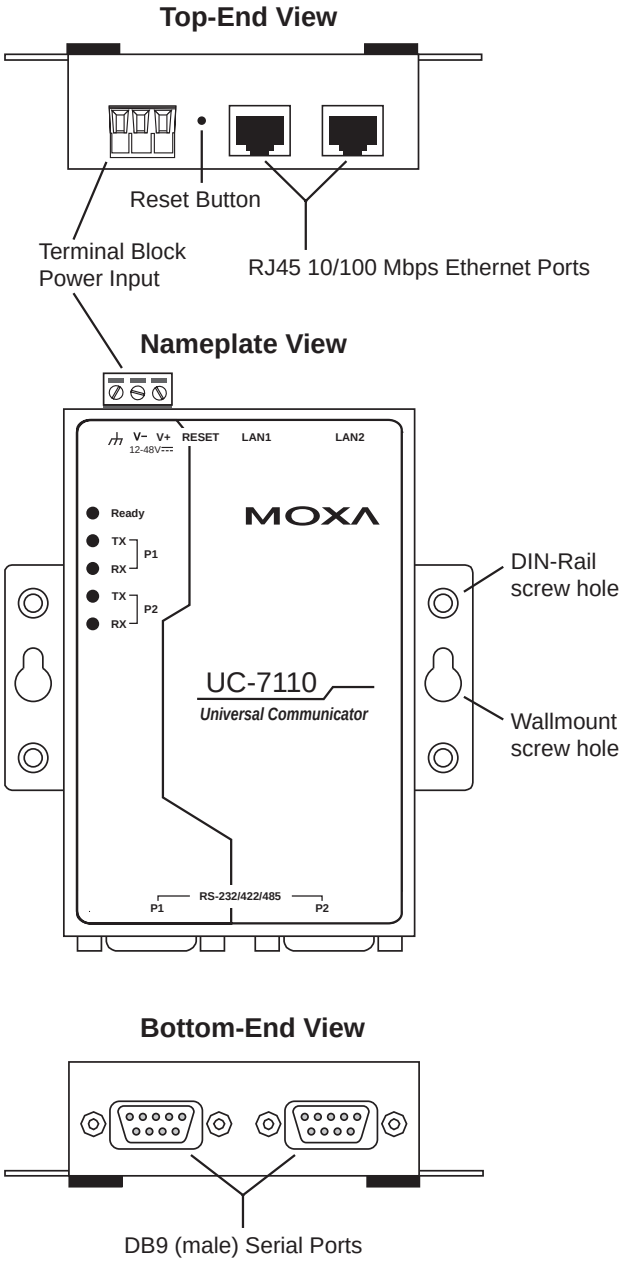
# Dimensions

The dimensions of the UC-7112 and UC-7110 are shown in the following figure.



# UC-7112/7110 Schematic

The UC-7110 schematic is shown in the following figure. The layout of the UC-7112 is identical.



# LED Indicators

The following table explains the function of the five LED indicators located on the UC-7112/7110's top panel.

| LED Name | LED Color | LED Function |
|---|---|---|
| Ready | Green | Power is on and functioning normally. |
| P1/P2 (Tx) | Green | Serial port 1 or 2 is transmitting data. |
| | Off | Serial port 1 or 2 is not transmitting data. |
| P1/P2 (Rx) | Yellow | Serial port 1 or 2 is receiving data. |
| | Off | Serial port 1 or 2 is not receiving data. |

# Wiring Requirements

This section describes how to connect the UC-7112/7110 to serial devices.

You should heed the following common safety precautions before proceeding with the installation of any electronic device:

- Use separate paths to route wiring for power and devices. If power wiring and device wiring paths must cross, make sure the wires are perpendicular at the intersection point.

  **NOTE**: Do not run signal or communication wiring and power wiring in the same wire conduit. To avoid interference, wires with different signal characteristics should be routed separately.

- Use the type of signal transmitted through a wire to determine which wires should be kept separate. The rule of thumb is that wiring that shares similar electrical characteristics can be bundled together.
- Keep input wiring and output wiring separate.
- We advise users to label the wiring to all devices in the system.

ATTENTION

**Safety First!**

Be sure to disconnect the power cord before installing and/or wiring your UC-7112/7110.

**Wiring Caution!**

Calculate the maximum possible current in each power wire and common wire. Observe all electrical codes dictating the maximum current allowable for each wire size.

If the current goes above the maximum ratings, the wiring could overheat, causing serious damage to your equipment.

**Temperature Caution!**

Be careful when handling the UC-7112/7110. When plugged in, the UC-7112/7110's internal components generate heat, and consequently the outer casing may feel hot to the touch.

## Connecting the Power

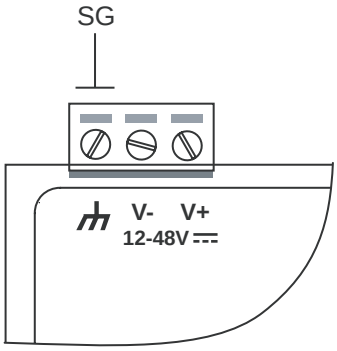
Connect the "live-wire" end of the 12-48 VDC power adaptor to the UC-7112/7110's terminal block. If the power is properly supplied, the "Ready" LED will glow a solid green after a 25 to 30 second delay.

## Grounding the UC-7112/7110

Grounding and wire routing help limit the effects of noise due to electromagnetic interference (EMI). Run the ground wire from the ground screw to the grounding surface prior to connecting devices.

ATTENTION

This product should be mounted to a well-grounded mounting surface such as a metal panel.




**SG**: The ***Shielded Ground*** (sometimes called Protected Ground) contact is the left most contact of the 3-pin power terminal block connector when viewed from the angle shown here. Connect the SG wire to an appropriate grounded metal surface.

# Connecting Data Transmission Cables

This section describes how to connect the UC-7112/7110 to the network, serial devices, and serial COM terminal.

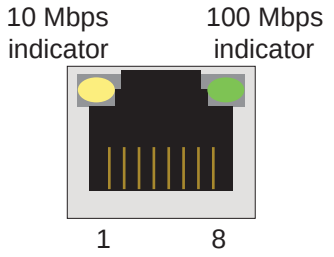
## Connecting to the Network

Connect one end of the Ethernet cable to the UC-7112/7110's 10/100M Ethernet port and the other end of the cable to the Ethernet network. If the cable is properly connected, the UC-7112/7110 will indicate a valid connection to the Ethernet in the following ways:

- The top-right LED on the connector glows a solid green when connected to a 100 Mbps Ethernet network.
- The top-left LED on the connector glows a solid orange when connected to a 10 Mbps Ethernet network.
- The LEDs will flash when Ethernet packets are being transmitted or received.

The 10/100 Mbps Ethernet LAN 1 and LAN 2 ports use 8-pin RJ45 connectors. Pinouts for these ports are given in the following diagram.

**8-pin RJ45**



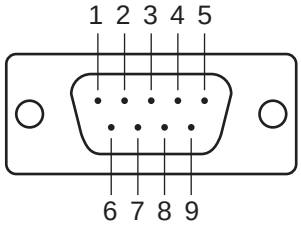

| Pin | Signal |
|---|---|
| 1 | ETx+ |
| 2 | ETx- |
| 3 | ERx+ |
| 4 | --- |
| 5 | --- |
| 6 | ERx- |
| 7 | --- |
| 8 | --- |

## Connecting to a Serial Device

Connect the serial cable between the UC-7112/7110 and the serial device(s).

Serial ports P1 and P2 use male DB9 connectors, and can be configured for RS-232/422/485 by software. The pin assignments are shown in the following table:

**DB9 Male Port**



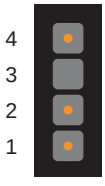

**RS-232/422/485 Pinouts**

| Pin | RS-232 | RS-422 | RS-485 (4-wire) | RS-485 (2-wire) |
|---|---|---|---|---|
| 1 | DCD | TxDA(-) | TxDA(-) | --- |
| 2 | RxD | TxDB(+) | TxDB(+) | --- |
| 3 | TxD | RxDB(+) | RxDB(+) | DataB(+) |
| 4 | DTR | RxDA(-) | RxDA(-) | DataA(-) |
| 5 | GND | GND | GND | GND |
| 6 | DSR | --- | --- | --- |
| 7 | RTS | --- | --- | --- |
| 8 | CTS | --- | --- | --- |

## Serial Console Port

The serial console port is a 4-pin pin-header RS-232 port. It is designed for serial console terminals, which are useful for identifying the UC-7112/7110's boot up message.

**Serial Console Port & Pinouts**




| Pin | Signal |
|---|---|
| 1 | TxD |
| 2 | RxD |
| 3 | NC |
| 4 | GND |

**Serial Console Cable**

# Internal SD Socket

The UC-7112 has an internal SD socket for storage expansion. Users can plug Secure Digital (SD) memory cards compliant with the SD 1.0 standard into the slot for up to 1 GB of additional memory space. To install an additional SD card, you must first remove the UC-7112's outer cover to access the slot. The internal SD socket is located at the backside of the UC-7112's bottom board; the SD plug-in slot is located on the UC-7112's right side, lower than the cover screw. Plug the SD card directly into the socket, and remember to press the SD card first if you want to remove it.

# Additional Functions

## Reset Button

Press the "RESET" button continuously for more than 5 seconds to load the factory default configuration. After loading the factory defaults, the system will reboot automatically. The System Ready LED will blink for the first 5 seconds. We recommend that you only use this function if the software is not working properly. To reset the μClinux system software, always use the software reboot command (reboot) to protect the integrity of data that is in the process of being transmitted. The reset button is NOT designed to hard reboot the UC-7112/7110.

ATTENTION

Resetting to factory defaults will not format the user directory and erase all of the user's data. Loading factory defaults will only load the configuration file. The files in the UC-7112/7110 that will be replaced include:

a. /etc/boa.conf
b. /etc/hosts
c. /etc/inittab
d. /etc/password
e. /etc/ramfs.img
f. /etc/resolv.conf
g. /etc/version
h. /etc/group
i. /etc/inetd.conf
j. /etc/motd
k. /etc/protocols
l. /etc/rc
m. /etc/services
n. /home/httpd/index.html

ATTENTION

This function only takes effect when the user directory is working correctly. If the user directory has crashed, the kernel will automatically load the factory defaults.

## Real Time Clock

UC-7112/7110's real time clock is powered by a lithium battery. We strongly recommend that you do not replace the lithium battery without the help of MOXA's support team. If the battery needs to be changed, contact the MOXA RMA service team for RMA service.

ATTENTION

The battery may explode if replaced by an incorrect type. To avoid this potential danger, always be sure to use the correct type of battery.

# 2

# Getting Started

In this chapter, we explain the basic procedure for getting UC-7112/7110 connected and ready to use.

This chapter covers the following topics:

- **Powering on the UC-7112/7110**
- **Connecting the UC-7112/7110 to a PC**
  - Console Port
  - Telnet
- **Configuring the Ethernet Interface**
- **Installing an SD Card (UC-7112 only)**
- **Developing Your Applications**
  - Installing the UC-7112/7110 Tool Chain
  - Compiling Hello.c
  - Uploading “Hello” to the UC-7112/7110
  - Running “Hello” on the UC-7112/7110
  - Sample Makefile Code

# Powering on the UC-7112/7110

Connect the SG wire to the Shielded Contact located on the upper left corner of the UC-7112/7110, and then power on UC-7112/7110 by connecting the power adaptor. It takes about 16 seconds for the system to boot up. Once the system is ready, the Ready LED will light up.

ATTENTION

After connecting the UC-7112/7110 to the power supply, it will take about 16 seconds for the operating system to boot up. The green Ready LED will not turn on until the operating system is ready.

# Connecting the UC-7112/7110 to a PC

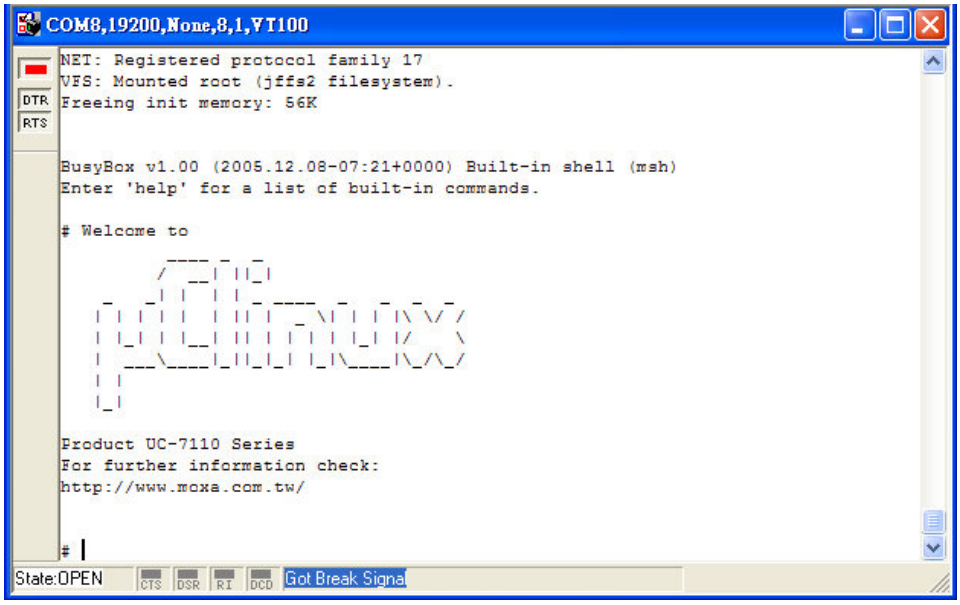
You may connect the UC-7112/7110 to a PC through the UC-7112/7110's console port, or by Telnet over the network.

## Console Port

The serial console port offers users a convenient means of connecting to the UC-7112/7110. This method is particularly useful when using the UC-7112/7110 for the first time. Since the communication is over a direct serial connection, you do not need to know either of the IP addresses in order to make contact.

Use the serial console port settings shown on the right. Once the connection is established, the window below will open.

| Serial Console Port Settings | |
|---|---|
| **Baudrate** | 19200 bps |
| **Parity** | None |
| **Data bits** | 8 |
| **Stop bits** | 1 |
| **Flow Control** | None |
| **Terminal** | VT100 |



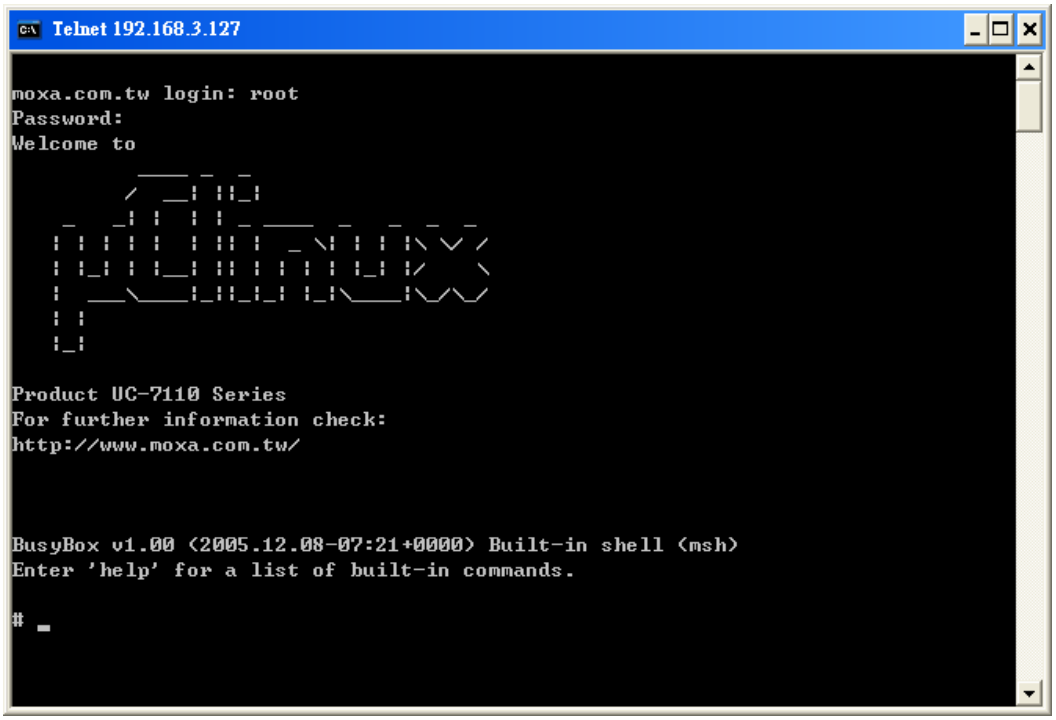
## Telnet

If you know the IP address and netmask of one of the UC-7112/7110's Ethernet ports, , then you can use Telnet to connect to the UC-7112/7110 console.

| | Default IP Address | Default Netmask |
|---|---|---|
| **LAN 1** | 192.168.3.127 | 255.255.255.0 |
| **LAN 2** | 192.168.4.127 | 255.255.255.0 |

Telnet can be used locally by using a crossover Ethernet cable to connect your computer to the UC-7112/7110, or by connecting to a hub or switch that connects to a LAN or the Internet. The default IP addresses and netmasks are shown above. To login, type the Login name and password as requested. The defaults are:

**Login: root**
**Password: root**




Once you open the "msh command shell" you can proceed to configure the UC-7112/7110's network settings, as described in the next section.

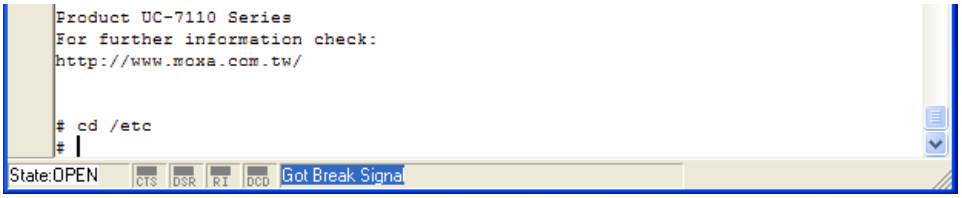
ATTENTION

- **Serial Console Reminder**: Remember to choose VT100 as the terminal type. Use the cable CBL-RJ45F9-150 that comes with the UC-7112/7110 to connect to the serial console port. If you are not able to connect on the first try, unplug and then re-plug the UC-7112/7110's power cord.
- **Telnet Reminder**: When connecting to the UC-7112/7110 over a LAN, configure your PC's Ethernet card to be on the same subnet as the UC-7112/7110 you wish to contact.

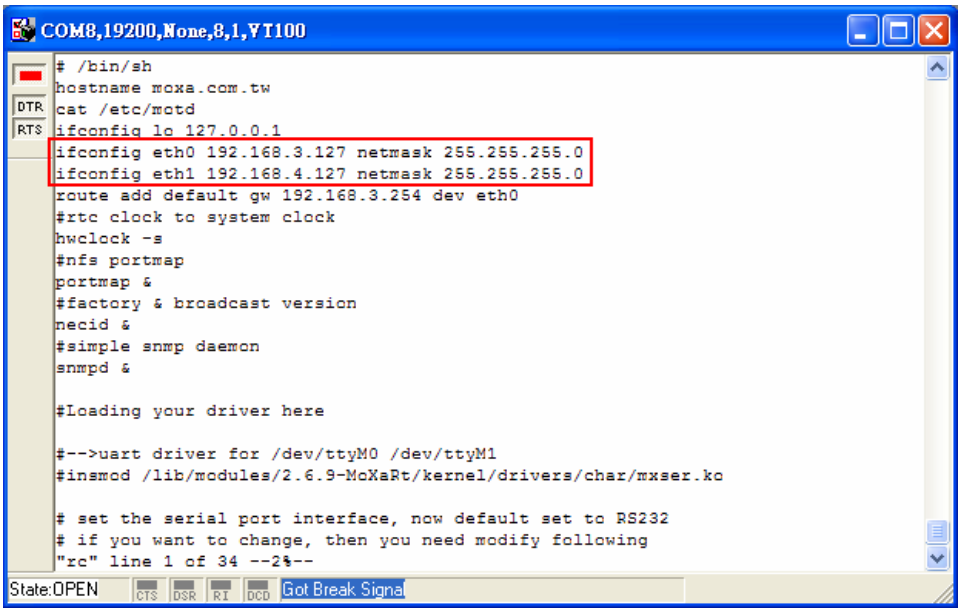
# Configuring the Ethernet Interface

In this section, we use the serial console to explain how to modify the UC-7112/7110's network settings.

1. Change directories by issuing the command *cd /etc*.



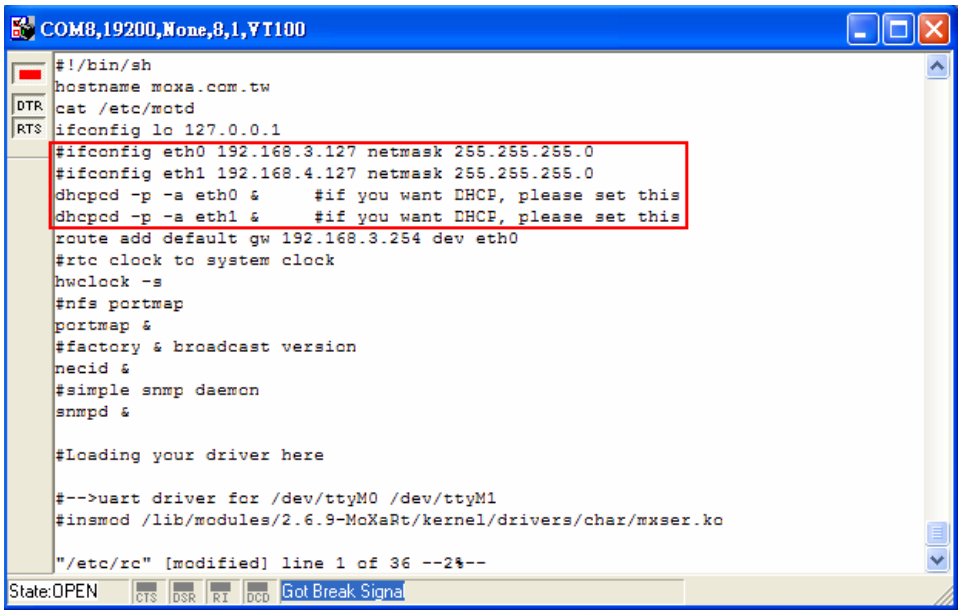

2. Type the command *vi rc* to use VI Editor to edit the configuration file. Use the following commands to modify the IP addresses for the UC-7112/7110's LAN1 and LAN2 ports:
***ifconfig eth0 192.168.3.127***
***ifconfig eth1 192.168.4.127***
Edit these two lines to modify the static IP addresses.



3. You may also configure the UC-7112/7110 to request IP addresses from a DHCP server. In this case, use the sharp sign (#) to comment out one or both "ifconfig" lines, and then use the following commands to add the "dhcpcd" setting to the rc file:

   **dhcpcd -p -a eth0 &**
   **dhcpcd -p -a eth1 &**

   Note that the UC-7112/7110 will send out DHCP broadcast packets, and then get the IP addresses from the first DHCP server that responds.




4. Issue the vi "write" command to save the file, and then reboot. Since the UC-7112/7110 only reads the "rc" file when booting up, you must reboot (e.g., by issuing the vi **`reboot`** command) for the changes to take affect.

ATTENTION

Use the following command to reset the IP address immediately:

**ifconfig eth0 192.168.5.127**

This will change the IP address of LAN1. However, issuing this command will NOT update the "rc" file in the UC-7112/7110's flash memory. For this reason, the next time you reboot, the IP address will revert to its previous value.

# Installing an SD Card (UC-7112 only)

the UC-7112 has an internal SD socket for storage expansion. To access the socket, perform the following steps to install the SD memory card.

**Step 1:** Loosen the screws of the UC-7112's casing.



**Step 2:** Remove the casing.

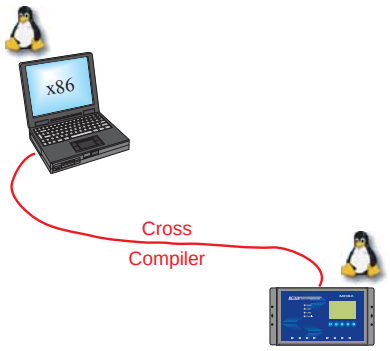

**Step 3:** The SD socket is located on the back side of the bottom board. Insert the SD memory card as shown below.

**Step 4:** Before using SD card, check the **/etc/rc** file to ensure that the driver module for controlling the SD card is loaded. The loading sequence is shown below:

**insmod /lib/modules/2.6.9-MoXaRt/kernel/drivers/mmc/mmc_core.ko**
**insmod /lib/modules/2.6.9-MoXaRt/kernel/drivers/mmc/mmc_block.ko**
**insmod /lib/modules/2.6.9-MoXaRt/kernel/drivers/mmc/moxasd.ko**

**Step 5:** To remove the SD memory card, first press it in and then release. The card will pop out partially. You may now grasp the card and pull it out of the slot.

# Developing Your Applications




**Step 1:**
Connect the UC-7112/7110 to a Linux PC.

**Step 2:**
Install the Tool Chain (GNU Cross Compiler & uClibc).

**Step 3:**
Configure the cross compiler and uClibc environment variables.

**Step 4:**
Code and compile your program.

**Step 5:**
Download the program to the UC-7112/7110 by FTP or NFS.

**Step 6:**
Debug the program. If the program is OK, proceed to Step 7. If the program needs to be modified, go back to Step 4.

**Step 7:**
Back up the user directory, and then if needed, distribute the code to additional UC-7112/7110 units.

## Installing the UC-7112/7110 Tool Chain

### Linux

The PC must have the Linux operating system pre-installed to install the UC-7112/7110 Linux GNU Tool Chain. Debian 3.0R-Woody, Redhat 7.3/8.0, and compatible versions are recommended. The Tool Chain requires about 100 MB of hard disk space on your PC. The UC-7112/7110 Tool Chain can be found on the UC-7112/7110 CD. To install the Tool Chain, insert the CD into your PC and then issue the following command:

**#mount –t iso9660 /dev/cdrom /mnt/cdrom**

Next, run the following script as root to install the compilers, linkers, and libraries in the **/usr/local** directory:

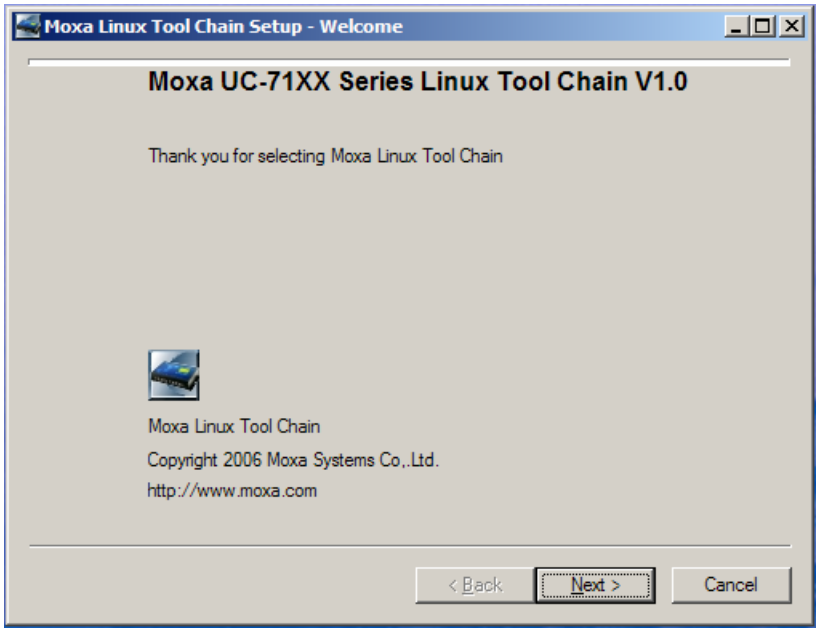
**#sh /mnt/cdrom/tool-chain/linux/installer/arm-elf-moxa-toolchain-1.1.sh**

The Tool Chain installation will take a few minutes to complete.

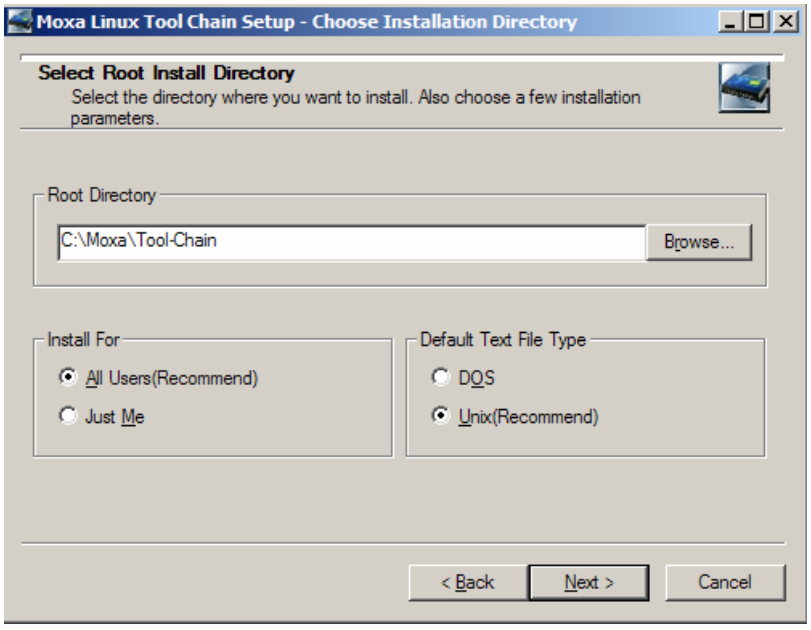
### Windows

In addition to the Linux Tool Chain, the Windows Tool Chain for the UC-7112/7110 is on the UC-7112/7110 CD. Use the installation procedure described below to install the UC-7112/7110 Windows Tool Chain.

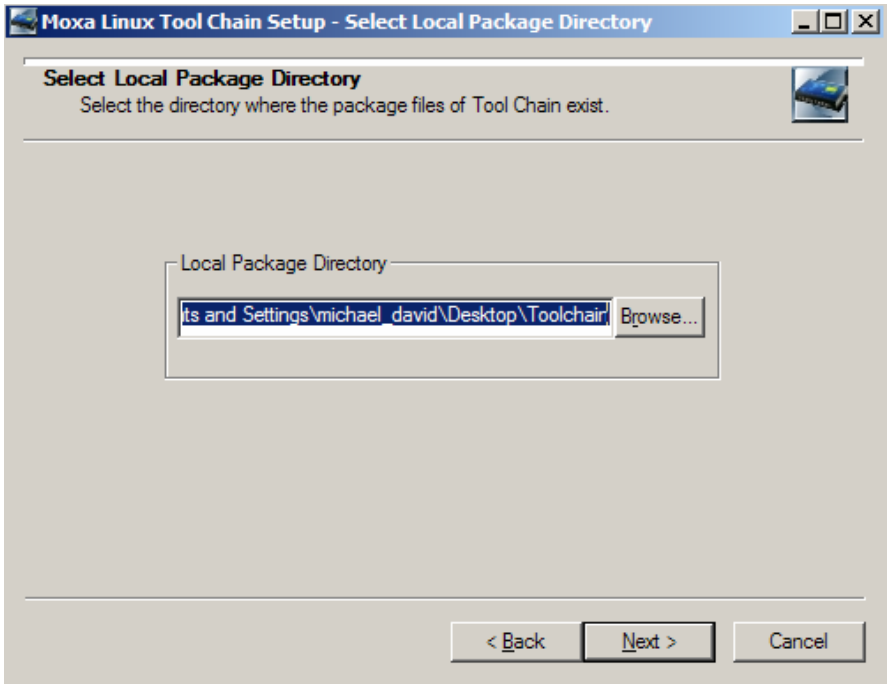
**Step 1:** Double click the "tool-chain\windows\setup.exe" on the UC-7112/7110 CD to begin the installation, and then click **Next**.




**Step 2:** Click **Browse…** to select your installation location. The default location is "C:\Moxa\Tool-Chain".



**Step 3:** Click **Next** to select the local package file directory, and then click **Browse…** to select where your installation source file is located. The default path is to the location of the file **setup.exe**.




**Step 4:** Click **Next** to begin the package installation. A progress bar appears to check the MD5 status of each software package. Click **Next** to finish the installation.

ATTENTION

You can download the Tool Chain software from MOXA's website. Navigate to the UC-7112/7110 product page, click the Documentation & Drivers link, and then click **Go** under Driver & Software Downloads.

## Compiling Hello.c

The Tool Chain path is:

**PATH=/usr/local/arm-elf/bin:$PATH**

The UC-7112/7110 CD includes several example programs. We use **Hello.c** to illustrate how to compile and run applications.

Issue the following commands from your PC to compile **Hello.c**:

**# cd /tmp/**
**# mkdir example**
**# cp –r /mnt/cdrom/example/* /tmp/example**

Go to the Hello subdirectory, and issue the command **#make** to compile Hello.c. Finally, execute the program to generate **hello** and **hello.gdb**.

```
[root@localhost hello]# ls -al
total 20
drwxr-xr-x    2 root     root         4096 Aug 18 10:58 .
drwxr-xr-x    5 root     root         4096 Aug  5 10:34 ..
-rw-rw-rw-    1 root     root         1498 Jan  6  2004 elf2flt.ld
-rw-rw-rw-    1 root     root           74 Jan  6  2004 hello.c
-rw-rw-rw-    1 root     root          875 Jan  6  2004 Makefile
[root@localhost hello]# make
/usr/local/bin/arm-elf-gcc  -g -O2 -pipe -Wall -I.   -c -o hello.o hello.c
/usr/local/bin/arm-elf-gcc  -o hello hello.o -g,-W1,-T,/usr/local/arm-elf/lib/el
f2flt.ld -elf2flt
[root@localhost hello]# ls -al
total 116
drwxr-xr-x    2 root     root         4096 Aug 18 10:59 .
drwxr-xr-x    5 root     root         4096 Aug  5 10:34 ..
-rw-rw-rw-    1 root     root         1498 Jan  6  2004 elf2flt.ld
-rwxr--r--    1 root     root        28624 Aug 18 10:59 hello
-rw-rw-rw-    1 root     root           74 Jan  6  2004 hello.c
-rwxr-xr-x    1 root     root        84543 Aug 18 10:59 hello.gdb
-rw-r--r--    1 root     root         7608 Aug 18 10:59 hello.o
-rw-rw-rw-    1 root     root          875 Jan  6  2004 Makefile
[root@localhost hello]#
```

## Uploading "Hello" to the UC-7112/7110

Issue the following commands from the PC to use FTP to upload **hello** to UC-7112/7110:

**#ftp 192.168.3.127**
**ftp> cd /home**
**ftp> bin**
**ftp> put ./hello**
**ftp> quit**
**#telnet 192.168.3.127**

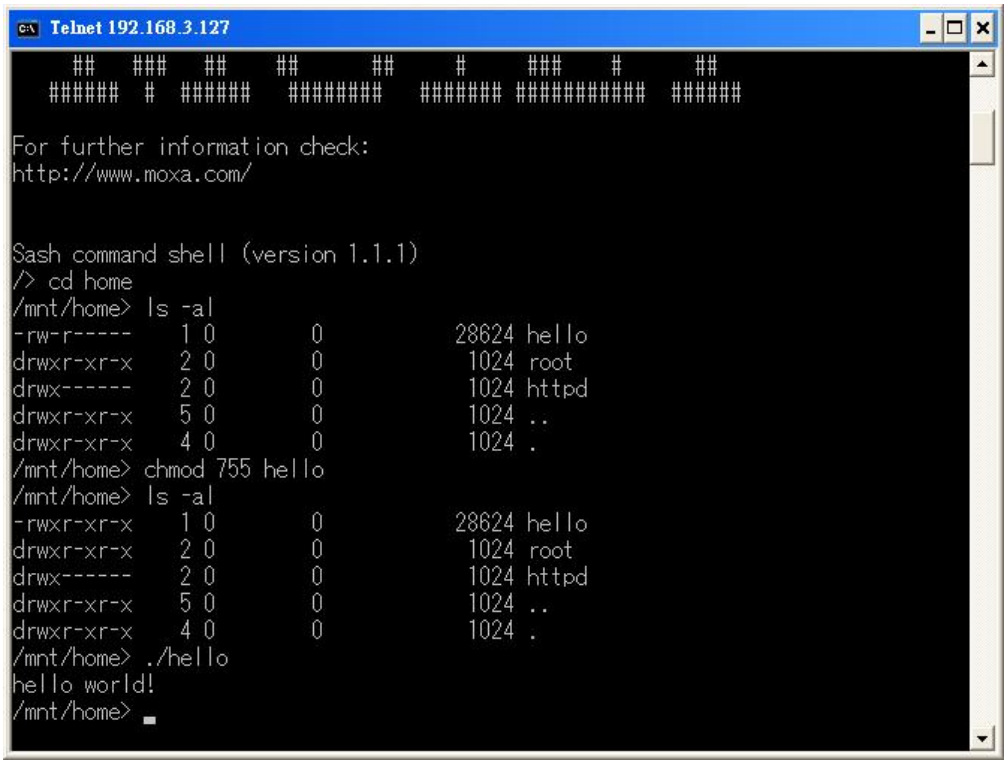
```
230 User root logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
227 Entering Passive Mode (192,168,3,127,8,0)
150 Opening ASCII mode data connection for '/bin/ls'.
lrwxrwxrwx    1 0        0               9 home -> /mnt/home
lrwxrwxrwx    1 0        0               8 etc -> /mnt/etc
lrwxrwxrwx    1 0        0               8 tmp -> /var/tmp
drwxr-xr-x    1 0        0              32 ramdisk
drwxr-xr-x    1 0        0              32 _home
drwxr-xr-x    1 0        0              32 _etc
drwxr-xr-x    1 0        0               0 var
dr-xr-xr-x    2 0        0               0 proc
drwxr-xr-x    5 0        0            1024 mnt
drwxr-xr-x    1 0        0              32 dev
drwxr-xr-x    1 0        0              32 bin
drwxr-xr-x    1 0        0              32 lib
226 Transfer complete.
ftp> cd /home
250 CWD command successful.
ftp> pwd
257 "/mnt/home" is current directory.
ftp>
```

## Running "Hello" on the UC-7112/7110

Issue the following commands on UC-7112/7110 to run the "Hello" program:

**# chmod 755 hello**
**#./hello**

The words "hello world" will be printed on the screen.




ATTENTION

Be sure to calculate the amount of Flash Memory used by the User File System in the Flash ROM. Use one of the following two commands to determine the amount of memory being used:

**# df -k** or **# df**

```
# df
Filesystem           1k-blocks      Used Available Use% Mounted on
rootfs                    1525      1525         0 100% /
/dev/rom0                 1525      1525         0 100% /
/dev/mtdblock2            4096       688      3408  17% /mnt
```

If the flash memory is full, you will no longer be able to save data in Flash ROM. To free up some memory, use the console cable to connect to UC-7112/7110's serial console terminal, and then delete files from the Flash ROM.

## Sample Makefile Code

The following Makefile example codes are copied from the Hello example on the UC-7112/7110's CD-ROM.

```
srcdir = .
LDFLAGS =  -Wl,-elf2flt
LIBS =
CFLAGS =

# Change these if necessary

CC = arm-elf-gcc
CPP = arm-elf-gcc -E

all:  hello

hello:
  $(CC) -o $@ $(CFLAGS) $(LDFLAGS) $(LIBS) $@.c

clean:
  rm -f $(OBJS) hello core *.gdb
```

# 3

# Software Package

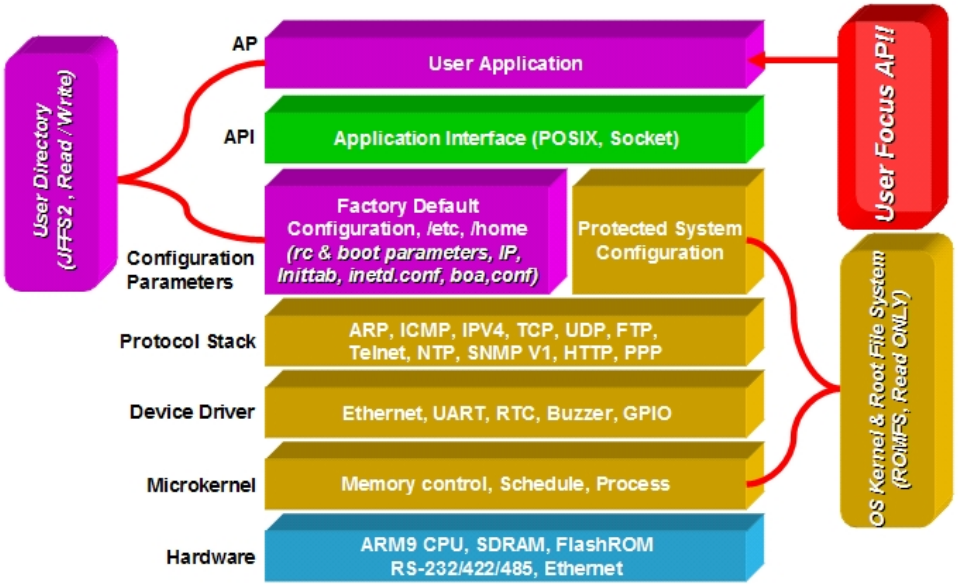
This chapter includes information about the software that is used with UC-7112/7110 Series products.

This chapter covers the following topics:

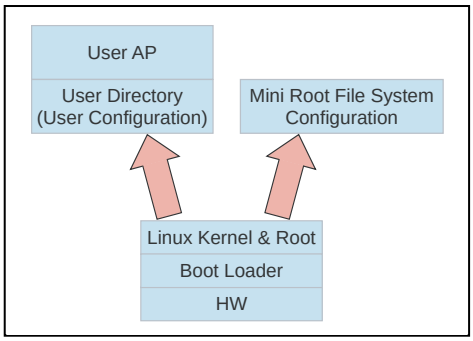
- **UC-7112/7110 Software Architecture**
  - Journaling Flash File System (JFFS2)
- **UC-7112/7110 Software Package**

# UC-7112/7110 Software Architecture

The UC-7112/7110 embedded computers come with the µClinux operating system pre-installed. The operating system follows the standard µClinux architecture. The GNU Tool Chain provided by www.uClinux.org can be used to port programs that follow the POSIX standard to the UC-7112/7110. In addition to the Standard POSIX API, device drivers for the serial ports' buzzers and UARTs are also included.




The UC-7112/7110's Flash ROM has multiple smaller partitions for the Boot Loader, Linux Kernel & Root (/) File System Image, and User Directory.




For most applications, users need to spend a lot time maintaining the operating system and modifying the system configuration. In order to save on the total cost of development and maintenance, the UC-7112/7110 is specially design to partition a "User Directory" for storing the user's system configuration parameters.

The UC-7112/7110 has a built-in mechanism that prevents system crashes and improves system reliability. The procedure is described below.

When the Linux kernel boots up, the kernel mounts the root file system and then enables services and daemons. The kernel also looks for the system configuration parameters using rc or inittab.

Normally, the kernel uses the User Directory to boot up the system. The kernel will only use the default configuration _etc & _home when the User Directory crashes.

The UC-7112/7110 uses ROMFS for the Linux kernel image, Root File System, and Protected configuration, and uses JFFS2 for the User Directory.

The partition sizes are hard coded into the kernel binary. You must rebuild the kernel to change the partition sizes. The flash memory map is shown in the following table.

| Flash Context | Flash Address | Size | Access control |
|---|---|---|---|
| Boot loader | 0 – 0x3ffff | 256 K | Read ONLY |
| Kernet & Root File System | 0x40000– 0x3fffff | 4 M | Read ONLY **JFFS2** |
| User Directory | 0x400000 – 0x7fffff | 4 M – 256 K | Read / Write **JFFS2** |

Developers should only save their own programs in partitions **/etc**, **/home**, **/tmp**, and **/usr/bin**.We also advise users to store executable files in **/usr/bin**, since doing so will allow developers to use hotkeys.

In addition to the flash file systems, a RAM based file system is mounted in **/var/**.

There are fundamental differences between programming an embedded computer and programming a PC. When programming your embedded computer, you should follow two important programming guidelines to ensure that your applications run smoothly:

1. Install your executable programs in the on-board flash.
2. Use the external SD card for data storage.

Following these guidelines will help to ensure that your applications run smoothly and trouble free.

## Journaling Flash File System (JFFS2)

The flash user directory is formatted by the Journaling Flash File System (JFFS2), which places a compressed file system on the flash (transparent to the user).

Axis Communications in Sweden developed the Journaling Flash File System (JFFS2).

JFFS2 provides a file system directly on flash, rather than emulating a block device designed for use on flash-ROM chips. It recognizes flash-ROM chips' special write requirements, does wear-leveling to extend flash life, keeps the flash directory structure in the RAM at all times, and implements a log-structured file system that is always consistent—even if the system crashes or unexpectedly powers down. It does not require fsck on boot up.

JFFS2, a newer version of JFFS, provides improved wear-leveling and garbage-collection performance, an improved RAM footprint and response to system-memory pressure, improved concurrency and support for suspending flash erases, marking of bad sectors with continued use of the remaining good sectors (to enhance the write-life of the devices), native data compression inside the file system design, and support for hard links.

Key features of JFFS2 are:

- Directly targeted to Flash ROM
- Robust
- Consistent across power failure
- No integrity scan (fsck) is required at boot time after normal or abnormal shutdown
- Explicit wear leveling
- Transparent compression

Although JFFS2 is a journaling file system, this does not ensure that data will not be lost. The file system will remain in a consistent state across power failures, and will always be mountable. However, if the board is powered down during a write, then the incomplete write will be rolled back on the next boot. Any writes that were already completed will not be affected.

**Additional information about JFFS2 is available on the following websites:**
http://sources.redhat.com/jffs2/jffs2.pdf

http://developer.axis.com/software/jffs/
http://www.linux-mtd.infradead.org/

# UC-7112/7110 Software Package

| bin | dev | |
|---|---|---|
| upkernel | mtdblock1 | ptype |
| passwd -> tinylogin | mtdr1 | ptypd |
| login -> tinylogin | mtd1 | ptypc |
| tinylogin | mtdblock0 | ptypb |
| telnetd | mtdr0 | ptypa |
| snmpd | mtd0 | ptyp9 |
| mail | cum1 | ptyp8 |
| sh | cum0 | ptyp7 |
| routed | ttyM1 | ptyp6 |
| netstat | ttyM0 | ptyp5 |
| arp | urandom | ptyp4 |
| chat | random | ptyp3 |
| pppd | zero | ptyp2 |
| portmap | ttypf | ptyp1 |
| ntpdate | ttype | ptyp0 |
| necid | ttypd | ppp |
| eraseall | ttypc | pio |
| kversion | ttypb | rtc |
| init | ttypa | ram1 |
| expand | ttyp9 | ram0 |
| inetd | ttyp8 | null |
| hwclock | ttyp7 | kmem |
| ftpd | ttyp6 | mem |
| ftp | ttyp5 | cua0 |
| mke2fs | ttyp4 | console |
| e2fsck | ttyp3 | tty |
| discard | ttyp2 | |
| dhcpcd | ttyp1 | |
| cpu | ttyp0 | |
| busybox | ttyS0 | |
| boa | tty3 | |
| bf | tty2 | |
| backupfs | tty1 | |
| downramdisk | tty0 | |
| upramdisk | rom1 | |
| | rom0 | |
| | ptypf | |

# 4

# Configuring the UC-7112/7110

In this chapter, we describe how to configure the UC-7112/7110 embedded computers.

The following topics are covered in this chapter:

# Enabling and Disabling Daemons

The following daemons are enabled when the UC-7112/7110 boots up for the first time.

- SNMP Agent daemon: **snmpd**
- Telnet Server / Client daemon: **telnetd**
- Internet Daemons: **inetd**
- FTP Server / Client daemon: **ftpd**
- WWW Server daemon: **boa**

ATTENTION

## How to enable/disable telnet/ftp server

a. Edit the file '/etc/inetd.conf'
**Example (default enable):**
discard dgram udp wait root /bin/discard
discard stream tcp nowait root /bin/discard
telnet stream tcp nowait root /bin/telnetd
ftp stream tcp nowait root /bin/ftpd -l
b. Disable the daemon by typing '#' in front of the first character of the row.

## How to enable/disable /etc/inittab www server

a. Edit the file '/etc/inittab'
b. Disable the www service by typing "#" in front of the first character of the row.

## How to enable Network Time Protocol

**ntpdate** is a time adjusting client utility. The UC-7112/7110 plays the role of Time client, and sends requests to the Network Time Server to request the correct time.

Set the time server address for adjusting the system time with the command:
```
/>ntpdate ntp_server_ip
```

Save the system time to the hardware's real time clock with the command:
```
/>hwclock --systohc
```

Visit http://www.ntp/org for a list of recommended public NTP servers.

## How to update the system time periodically with Network Time Protocol

1. Create a shell script file that includes the following description.
```
#!/bin/sh
ntpdate ntp_server_ip
hwclock –systohc
sleep 100        ← The min time is 100ms.
```
2. Save and make this shell script executable by typing
```
chmod 755 <shell-script_name>
```

Edit the file '/etc/inittab' by adding the following line:
```
ntp: unknown: /directory/<shell_script_name>
```

# Adding a Web Page

**Default Home Page address:**

***/home/httpd/index.html***

You may change the default home page directory by editing the web server's configuration file, located at: **/etc/boa.conf**.

Type the following command to edit the boa.conf file:

*/etc>vi boa.conf*


To add your web page, place your home page in the following directory:

***/home/httpd/***

# IPTABLES

IPTABLES is an administrative tool for setting up, maintaining, and inspecting the Linux kernel's IP packet filter rule tables. Several different tables are defined, with each table containing built-in chains and user-defined chains.

Each chain is a list of rules that apply to a certain type of packet. Each rule specifies the action to be taken with a matching packet. A rule (such as a jump to a user-defined chain in the same table) is called a "target."

The UC-7112/7110 supports three types of IPTABLES tables: Filter tables, NAT tables, and Mangle tables:

A. **Filter Table**—includes three chains:

   INPUT chain
   OUTPUT chain
   FORWARD chain

B. **NAT Table**—includes three chains:

   PREROUTING chain—transfers the destination IP address (DNAT)
   POSTROUTING chain—works after the routing process and before the Ethernet device process to transfer the source IP address (SNAT)
   OUTPUT chain—produces local packets

   *sub-tables*

Source NAT (SNAT)—changes the first source packet IP address

Destination NAT (DNAT)—changes the first destination packet IP address

MASQUERADE—a special form for SNAT. If one host can connect to the Internet, then other computers that connect to this host can connect to the Internet when the computer does not have an actual IP address.

REDIRECT—a special form of DNAT that re-sends packets to a local host independent of the destination IP address.

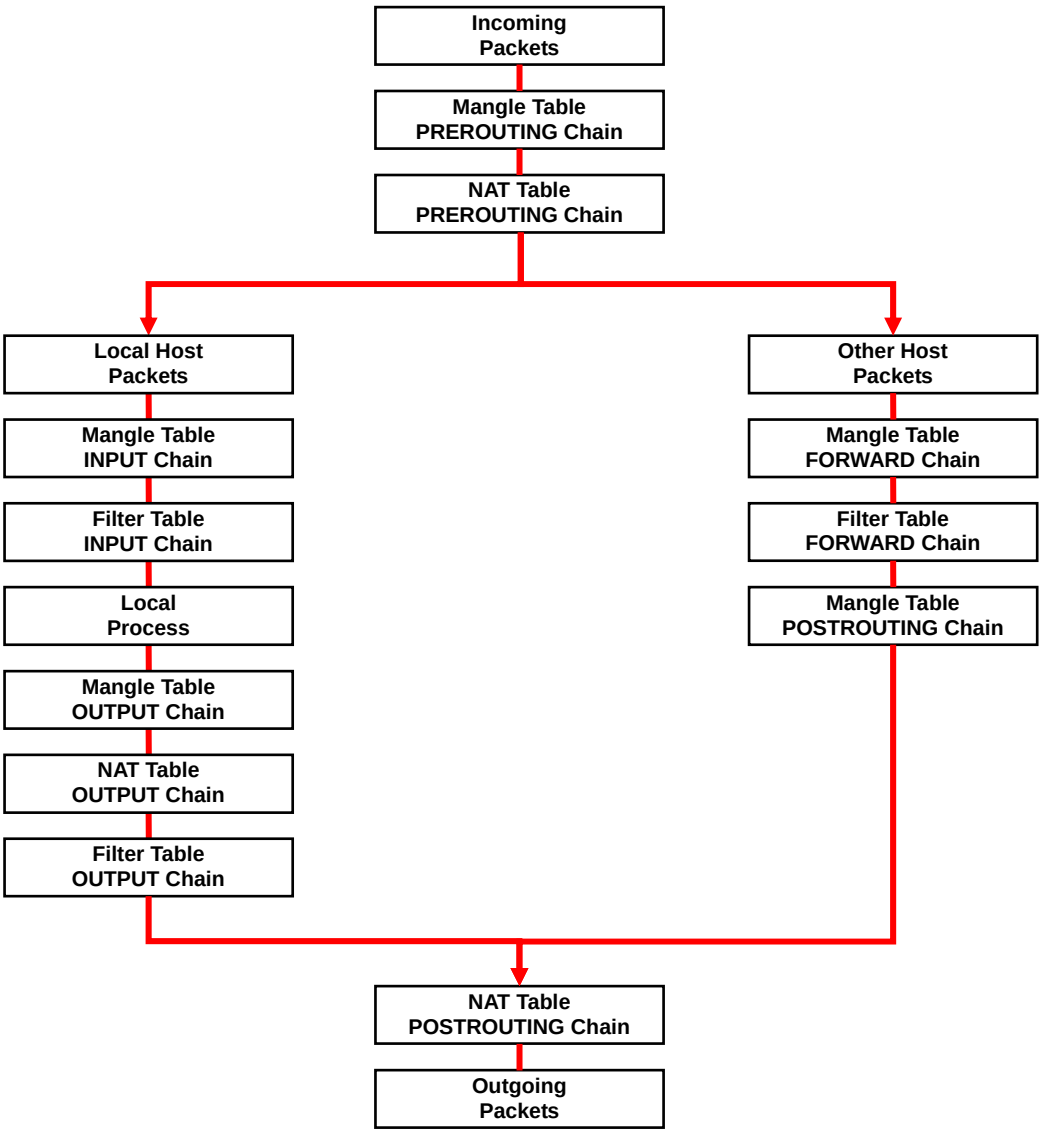
C. **Mangle Table**—includes two chains

PREROUTING chain—pre-processes packets before the routing process.

OUTPUT chain—processes packets after the routing process.

It has three extensions—TTL, MARK, TOS.

The following figure shows the IPTABLES hierarchy.



The UC-7112/7110 supports the following sub-modules. Be sure to use the module that matches your application.

| | | | |
|---|---|---|---|
| ip_conntrack | ipt_MARK | ipt_ah | ipt_state |
| ip_conntrack_ftp | ipt_MASQUERADE | ipt_esp | ipt_tcpmss |
| ipt_conntrack_irc | ipt_MIRROT | ipt_length | ipt_tos |
| ip_nat_ftp | ipt_REDIRECT | ipt_limit | ipt_ttl |
| ip_nat_irc | ipt_REJECT | ipt_mac | ipt_unclean |
| ip_nat_snmp_basic | ipt_TCPMSS | ipt_mark | |
| ip_queue | ipt_TOS | ipt_multiport | |
| | | ipt_owner | |

NOTE The UC-7112/7110 does NOT support IPV6 and ipchains.

Use **`iptables, iptables-restore, iptables-save`** to maintain the database.

NOTE IPTABLES supports packet filtering or NAT. Take care when setting up the IPTABLES rules. If the rules are not correct, remote hosts that connect via a LAN or PPP may be denied access. We recommend using the Serial Console to set up IPTABLES.

Click on the following links for more information about iptables.

http://www.linuxguruz.com/iptables/
http://www.netfilter.org/documentation/HOWTO//packet-filtering-HOWTO.html

Since the IPTABLES command is very complex, to illustrate the IPTABLES syntax we have divided our discussion of the various rules into three categories: **Observe and erase chain rules**, **Define policy rules**, **and Append or delete rules.**

## Observe and erase chain rules

**Usage:**

**`# iptables [-t tables] [-L] [-n]`**

-t tables: Table to manipulate (default: 'filter'); example: nat or filter.
-L [chain]: List List all rules in selected chains. If no chain is selected, all chains are listed.
-n: Numeric output of addresses and ports.

**`# iptables [-t tables] [-FXZ]`**

-F: Flush the selected chain (all the chains in the table if none is listed).
-X: Delete the specified user-defined chain.
-Z: Set the packet and byte counters in all chains to zero.

**Examples:**

**`# iptables -L -n`**

In this example, since we do not use the -t parameter, the system uses the default 'filter' table. Three chains are included: INPUT, OUTPUT, and FORWARD. INPUT chains are accepted automatically, and all connections are accepted without being filtered.

```
#iptables –F
#iptables –X
#iptables –Z
```

## Define policy for chain rules

**Usage:**

```
# iptables [-t tables] [-P] [INPUT, OUTPUT, FORWARD, PREROUTING, OUTPUT, POSTROUTING]
[ACCEPT, DROP]
```

| | |
|---|---|
| -P: | Set the policy for the chain to the given target. |
| INPUT: | For packets coming into the UC-7112/7110. |
| OUTPUT: | For locally-generated packets. |
| FORWARD: | For packets routed out through the UC-7112/7110. |
| PREROUTING: | To alter packets as soon as they come in. |
| POSTROUTING: | To alter packets as they are about to be sent out. |

**Examples:**

```
#iptables –P INPUT DROP
#iptables –P OUTPUT ACCEPT
#iptables –P FORWARD ACCEPT
#iptables –t nat –P PREROUTING ACCEPT
#iptables –t nat –P OUTPUT ACCEPT
#iptables -t nat –P POSTROUTING ACCEPT
```

In this example, the policy accepts outgoing packets and denies incoming packets.

## Append or delete rules:

**Usage:**

```
# iptables [-t table] [-AI] [INPUT, OUTPUT, FORWARD] [-io interface] [-p tcp, udp, icmp,
all] [-s IP/network] [--sport ports] [-d IP/network] [--dport ports] –j [ACCEPT. DROP]
```

| | |
|---|---|
| -A: | Append one or more rules to the end of the selected chain. |
| -I: | Insert one or more rules in the selected chain as the given rule number. |
| -i: | Name of an interface through which a packet will be received. |
| -o: | Name of an interface through which a packet will be sent. |
| -p: | The protocol of the rule or of the packet to check. |
| -s: | Source address (network name, host name, network IP address, or plain IP address). |
| --sport: | Source port number. |
| -d: | Destination address. |
| --dport: | Destination port number. |
| -j: | Jump target. Specifies the target of the rules; i.e., how to handle matched packets. For example, ACCEPT the packet, DROP the packet, or LOG the packet. |

**Examples:**

Example 1: Accept all packets from lo interface.

```
# iptables –A INPUT –i lo –j ACCEPT
```

Example 2: Accept TCP packets from 192.168.0.1.

```
# iptables –A INPUT –i eth0 –p tcp –s 192.168.0.1 –j ACCEPT
```

Example 3: Accept TCP packets from Class C network 192.168.1.0/24.

```
# iptables –A INPUT –i eth0 –p tcp –s 192.168.1.0/24 –j ACCEPT
```

Example 4: Drop TCP packets from 192.168.1.25.

```
# iptables –A INPUT –i eth0 –p tcp –s 192.168.1.25 –j DROP
```

Example 5: Drop TCP packets addressed for port 21.

```
# iptables –A INPUT –i eth0 –p tcp --dport 21 –j DROP
```

Example 6: Accept TCP packets from 192.168.0.24 to UC-7112/7110's port 137, 138, 139

```
# iptables –A INPUT –i eth0 –p tcp –s 192.168.0.24 --dport 137:139 –j ACCEPT
```

Example 7: Log TCP packets that visit UC-7112/7110's port 25.

```
# iptables –A INPUT –i eth0 –p tcp --dport 25 –j LOG
```

Example 8: Drop all packets from MAC address 01:02:03:04:05:06.

```
# iptables –A INPUT –i eth0 –p all –m mac –mac-source 01:02:03:04:05:06 –j DROP
```

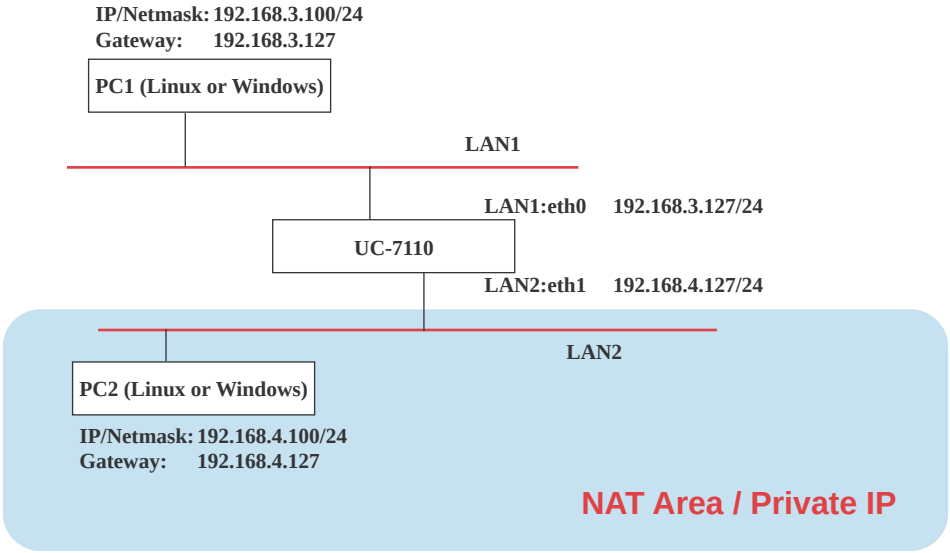
# NAT

NAT (Network Address Translation) protocol translates IP addresses used on one network into different IP addresses used on another network. One network is designated the inside network and the other is the outside network. Typically, the UC-7112/7110 connects several devices on a network and maps local inside network addresses to one or more global outside IP addresses, and remaps the global IP addresses on incoming packets back into local IP addresses.

NOTE Click the following link for more information about iptables and NAT:
http://www.netfilter.org/documentation/HOWTO/NAT-HOWTO.html

## NAT Example

The IP addresses of all packets leaving LAN1 are changed to 192.168.3.127 (you will need to load the module ipt_MASQUERADE):




1. `#echo 1 > /proc/sys/net/ipv4/ip_forward`
2. `#iptables -t nat –A POSTROUTING –o eth0 –j SNAT --to-source 192.168.3.127`
   or
3. `#iptables –t nat –A POSTROUTING –o eth0 –j MASQUERADE`

## Enabling NAT at Bootup

In most real world situations, you should use a simple shell script to enable NAT when the UC-7112/7110 boots up, as indicated by the following:

1. setting iptables
2. iptables-save > /home/xxx.file (xxx.file is the user defined file name)
3. vi /etc/rc
4. Append echo 1 > /proc/sys/net/ipv4/ip_forward
5. Append iptables-restore /home/xxx.file (xxx.file is the user defined file name)

# Configuring Dial-in/Dial-out Service

## Dial-out Service

### Direct cable connection:

- *Without* username and password, use:
  **/>pppd connect 'chat –v' /dev/ttyM0 38400 crtscts&**
- *With* username and password, use:
  **/>pppd connect 'chat –v' user xxxxx password xxxxx /dev/ttyM0 38400 crtscts&**

### Connect Using a Modem:

- Use:
  **/>pppd connect 'chat –v ATDT<phone_number> CONNECT' user xxxxx password xxxxx /dev/ttyM0 38400 crtscts&**

ATTENTION

If dial out fails, the pppd connection will be blocked, and the users will need to shut down pppd, and re-dial. Since the return value is always OK (regardless of whether or not the connection is blocked), the API must be set up to check the network status to determine if the connection is complete.

## Dial-in Service

### Direct cable connection:

- Use either of the following:
  **/>pppd <Local_IP_Address>:<Remote_IP_Address> /dev/ttyM1 38400 local crtscts**
  or
  **/>pppd <Local_IP_Address>:<Remote_IP_Address> /dev/ttyM0 38400 local crtscts login auth**

### Connect Using a Modem:

- Use:
  **/>pppd connect 'chat –v AT CONNECT' <local_IP_Address>:<Remote_IP_Address> /dev/ttyM0 38400 crtscts login auth**

# Configuring PPPoE

PPPoE relies on two widely accepted standards: PPP and Ethernet, which permit the use of PPPoE(Point-to-Point Over Ethernet).

PPPoE is a specification for connecting users on an Ethernet to the Internet through a common broadband medium, such as a single DSL line, wireless device or cable modem, used by many ADSL service providers. All users on the Ethernet share a common connection, so the Ethernet principles that support multiple users on a LAN combine with the PPP principles, which apply to serial connections.

- Create the Connection:
  **/>pppd pty "pppoe -I <ETHERNET_INTERFACE> -m 1412" user <USER_NAME> password <USER_PASSWORD>&**

  <ETHERNET_INTERFACE>: Ethernet card connected to ADSL modem, for example, eth0
  <USER_NAME>: User account, for example, moxa@adsl.net

<USER_PASSWORD>: Password for user account

To check if PPPOE is successfully connected, use the command:

- **/>ifconfig ppp0**

# How to Mount a Remote NFS Server

Currently, the UC-7112/7110 only supports NFS (Network File System) clients. Users can open NFS service on a Linux PC to enable the UC-7112/7110 to push data to it. The UC-7112/7110 can use NFS to mount a remote disk as a local disk for data or log purposes.

1. First, the NFS server must open an export directory and allow access to the IP address. Edit the file "/etc/exports" on your Linux PC, and then run the NFS daemon. The following example gives one possibility (refer to the NFS-HOWTO document at http://nfs.sourceforge.net/nfs-howto/server.html):

   **/home/usr 192.168.3.1 (rw,no_root_squash,no_all_squash)**

2. The UC-7112/7110 must run the "portmap" utility. This program is enabled by default in the "/etc/rc" file. Use the following command to mount the remote NFS server:

   **/>mount –t nfs <remote-ip>:<remote-export-directory> <local-directory>**

# Dynamic Driver Module Load / Unload

In addition to supporting traditional static drivers, the UC-7110/7112 also supports the dynamic driver module load / unload mechanism. It allows user to load a special driver into the kernel to enable hardware features for specific applications. To load / unload a dynamic driver module, use the following commands.

Load module:
**/>insmod <module-directory>/<module file name>**

For example, to load the UART driver, type the following command:
**/>insmod /lib/modules/2.6.9-MoXaRt/kernel/drivers/char/mxser.ko**

Show module list:
**/>lsmod**

Unload module:
**/>rmmod <module-name listed by lsmod command>**

For example, to unload the UART driver, type the following command:
**/>rmmod mxser**

For the UC-7110, the factory default is to load the UART driver "mxser.ko". The additional driver module to control the SD/MMC memory card is loaded for the UC-7112. Please see the information below for the locations and file names of these driver modules.

UART:

**/lib/modules/2.6.9-MoXaRt/kernel/drivers/char/mxser.ko**

SD/MMC:

**/lib/modules/2.6.9-MoXaRt/kernel/drivers/mmc/mmc_core.ko**

**/lib/modules/2.6.9-MoXaRt/kernel/drivers/mmc/mmc_block.ko**

**/lib/modules/2.6.9-MoXaRt/kernel/drivers/mmc/moxasd.ko**

# Upgrading the Kernel

The UC-7112/7110 kernel is ***uc7110-3.x..bin (uc7112-1.x.bin for UC-7112)***, which can be downloaded from www.moxa.com. You must first download this file to your PC, and then use the Console Terminal or Telnet Console to copy the file to the UC-7112/7110.

You can save this file to the UC-7112/7110's RAM disk, and then upgrade the kernel. The following is a step-by-step example.

To enable the RAM disk, use the following command:

**/>upramdisk**

After executing "upramdisk", you may use "mount" to find out if the new ramdisk has been created successfully:

```
# upramdisk
# mount
/dev/mtdblock2 on / type jffs2 (ro,noatime)
/proc on /proc type proc (rw,nodiratime)
/dev/ram0 on /var type ext2 (rw)
/dev/mtdblock3 on /var/tmp type jffs2 (rw,noatime)
/dev/mtdblock3 on /home type jffs2 (rw,noatime)
/dev/mtdblock3 on /etc type jffs2 (rw,noatime)
/dev/mtdblock3 on /usr/bin type jffs2 (rw,noatime)
/dev/ram0 on /ramdisk type ramfs (rw)
#
```

To navigate to the device node, use the following command:

**/>cd ramdisk**

Use the built-in FTP client to download the uc7110-3.x.bin file from the PC.

**/ramdisk>ftp <destination PC's IP>**

**Login Name: xxxx**

**Login Password: xxxx**

**ftp> bin**

**ftp> get uc7110-3.x.bin**

Use the **upkernel** command to upgrade the kernel and root file system.

**/ramdisk>upkernel uc7110-3.x.bin**

**/ramdisk>reboot**

```
# upramdisk
# cd /ramdisk
# upkernel uc7112-1.0.bin
To check the source file context.
The kernel source file is OK.
The version is 1.0.
This step will destory your old kernel.
Do you want to continue it ? (Y/N) : Y
Formating disk !!!
Erased 2048 Kibyte @ 0 -- 100% complete.
Format OK. Now update the kernel.
Please wait ...
Update the kernel OK. Please restart system.
#
```

# Upgrading the Root File System & User Directory

The UC-7112/7110 uses JFFS2 for the root file system and user directory. By default, the root file system is pre-set to READ only. The UC-7112/7110 provides a read/write user's directory in the JFFS2 file system. Use this user's directory to store the system configuration file and user's programs on the disk.

You can search the UC-7112/7110's CD-ROM for the latest user directory file, or download the file from www.moxa.com. The format of the file is uc7110-3.x.dsk (uc7112-1.x.dsk for UC-7112). You must download this file to a PC first, and then use the console terminal or Telnet console to copy the file to the UC-7112/7110.

You can save this file to the UC-7112/7110's RAM disk, and then upgrade the user directory. A step-by-step example is shown below.

Use the following commands to enable the RAM disk:

**/>upramdisk**
**/>cd ramdisk**

Use the built-in FTP client to download the uc7110-3.x.dsk file from the PC:

**/ramdisk>ftp <destination PC's IP>**
**Login Name: xxxx**
**Login Password: xxxx**
**ftp> bin**
**ftp> get uc7110-3.x.dsk**
**ftp>quit**
**/ramdisk>upkernel /ramdisk/uc7110-3.x.dsk**
**/reboot**

You will also need to restore factory defaults to load the new settings. To do this, either press the RESET button for more than 5 seconds, or input the command "stdef" from the Telnet console.

```
# upkernel uc7112-1.0.dsk
To check the source file context.
The firmware source file is OK.
The version is 1.0.
This step will destory your old kernel.
Do you want to continue it ? (Y/N) : Y
Formating disk !!!
Erased 2560 Kibyte @ 0 -- 100% complete.
Format OK. Now update the root filesystem.
Please wait ...
Update the root file system OK. Please push the reset button.
# _
```

# User Directory Backup—UC-7112/7110 to PC

Use the following commands to enable the RAM disk:

**/>upramdisk**
**/>cd ramdisk**

Use the **backupfs** command to backup the file system:

**/ramdisk>backupfs /ramdisk/usrdisk-backup**

```
/> backupfs /ramdisk/usrdisk-backup
Sync the file system.
Now backup the user directory. Please wait ...
Backup the user directory OK.
/>
```

The file system will be backed up. Use FTP commands to transfer "usrdisk-backup" to the FTP server on the PC.

```
/> cd /ramdisk
/ramdisk> ls -al
----------    1 0        0         4194304 usrdisk-backup
drwxr-xr-x    1 0        0              32 ..
drwxr-xr-x    2 0        0            1024 .
/ramdisk> ftp 192.168.3.11
Connected to 192.168.3.11.
220 TYPSoft FTP Server 1.10 ready...
Name (192.168.3.11:root): root
331 Password required for root.
Password:
230 User root logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> put usrdisk-backup
local: usrdisk-backup remote: usrdisk-backup
200 Port command successful.
150 Opening data connection for usrdisk-backup.
226 File received complete
4195224 bytes sent in 37 secs (113 Kbytes/sec)
ftp>
```

# Loading Factory Defaults

The easiest way to "Load Factory Defaults" is with the "Upgrade User directory" operation.

Refer to the previous section **Upgrading the Root File Sysem & User Directory** for an introduction.

You may also press the RESET button for more than 5 seconds to load the factory default configuration or input the command "stdef" from the Telnet console to restore the factory defaults.

# Mirroring the Application Program and Configuration

For some applications, you may need to "Mirror" (or sometimes "Ghost") one UC-7112/7110's user directory, and duplicate it to other UC-7112/7110 embedded computers. We recommend using the following procedure to do this:

1. Backup the user directory to a PC:

   **/ramdisk>backupfs /ramdisk/<user defined file name>**

   (Refer to the previous topic **User Directory Backup—UC-7112/7110 to PC**.)

2. Download the backed up user directory to the other UC-7112/7110.

**/ramdisk>bf /ramdisk/<User directory file name>**

(Refer to the previous topic **Upgrading the Root File System & User Directory**.)

# Autostarting User Applications on Bootup

To autostart user applications on bootup, edit the **/etc/rc** file by adding your application program. For example, you might add the following line to the file:

**/ap-directory/ap-program &**

# Checking the Kernel and Root File System Versions

Use the following commands to check the version of the kernel and root file system:

Use the following command to check the kernel version:

**/>kversion**

Use the following command to check the root file system (firmware) version of the UC-7112/7110:

**/>fsversion**

Use the following command to check the user directory version of the UC-7112/7110:

**/>cat /etc/version**

# 5

# UC-7112/7110 Device API

In this chapter, we discuss the Device API for the UC-7112/7110 Series. We introduce the APIs for the following functions:

- **RTC (Real Time Clock)**
- **Buzzer**
- **UART Interface**

# RTC (Real Time Clock)

The device node is located at **/dev/rtc**. The UC-7112/7110 supports µClinux standard simple RTC control. You must include **<linux/rtc.h>** to use these functions.

1. Function: RTC_RD_TIME

   ```
   int ioctl(fd, RTC_RD_TIME, struct rtc_time *time);
   ```

   Description: Reads time information from RTC.

2. Function: RTC_SET_TIME

   ```
   int ioctl(fd, RTC_SET_TIME, struct rtc_time *time);
   ```

   Description: Sets RTC time.

# Buzzer

The device node is located at **/dev/console**. The UC-7112/7110 supports µClinux standard buzzer control. The UC-7112/7110's buzzer runs at a fixed frequency of 100 Hz. You must include **<sys/kd.h>** to use these functions.

1. Function: KDMKTONE

   ```
   ioctl(fd, KDMKTONE, unsigned int arg);
   ```

   Description: Buzzer will beep, as stipulated by the function arguments.

# UART Interface

The normal tty device node is located at **/dev/ttyM0…ttyM1**, and modem tty device node is located at **/dev/com0 … com1**. The UC-7112/7110 Series supports µClinux standard termios control. The MOXA UART Device API supports the configuration of ttyM0 to ttyM1 as RS-232/422/485. To use these functions, after the Tool Chain package is installed, copy the file "CDROM/libuc7110/uc7110.h" to the directory "/usr/local/arm-elf/include/" on your Linux PC, and then include <uc7110.h> in your application.

```
#define RS232_MODE          0
#define RS485_2WIRE_MODE        1
#define RS422_MODE          2
#define RS485_4WIRE_MODE        3
```

1. Function: MOXA_SET_OP_MODE

   ```
   int mode;
   mode=which mode you want to set;
   int ioctl(fd, MOXA_SET_OP_MODE, &mode)
   ```

   Description: Sets the interface mode.

2. Function: MOXA_GET_OP_MODE

   ```
   int mode;
   int ioctl(fd, MOXA_GET_OP_MODE, &mode)
   ```

   Description: Gets the interface mode.

# 6

# UC Finder

The UC-7112/7110 comes with a UC Finder utility, which is used to search the LAN or intranet for UC-7112/7110 embedded computers.

For most applications, it is not easy to remember the IP addresses of embedded computers connected to the LAN. This is especially true for troubleshooting and/or testing in the field. The UC Finder utility broadcasts messages over the LAN to determine the IP addresses of MOXA embedded computers connected to the LAN. UC Finder does this by searching for the class of MAC addresses assigned to Universal Communicators. The UC-7112/7110 supports the GUI-style Windows UC Finder, and a command line utility for Linux environments.

In this chapter, we discuss the following UC Finder topics:

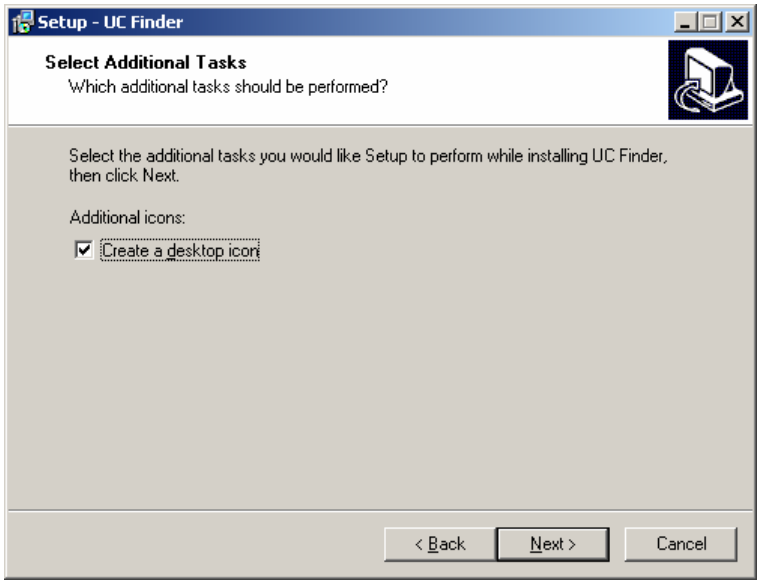
- **UC Finder for Windows**
- **UC Finder for Linux**

# UC Finder for Windows

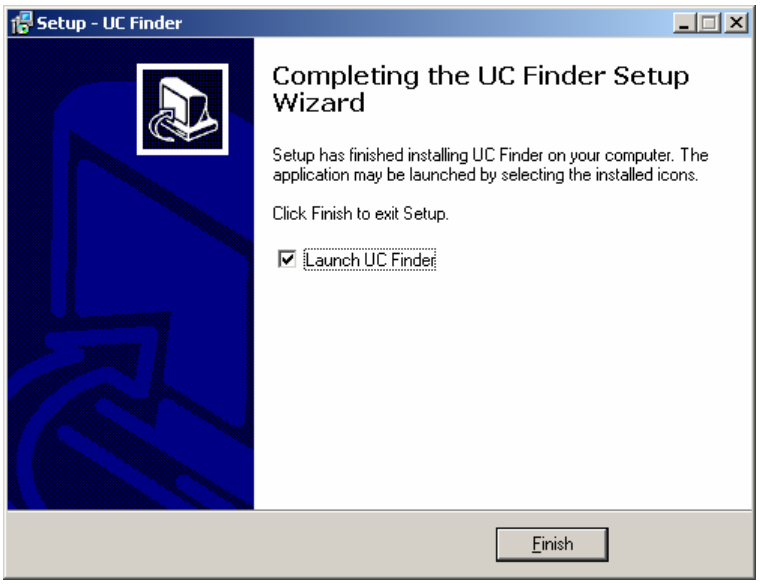
In this section, we describe how to install UC Finder on a Windows PC.

1. Double click the UC Finder installation program, **Setup.exe**, to start the installation.
2. When the **Welcome to the UC Finder Setup Wizard** window opens, click **Next** to continue.

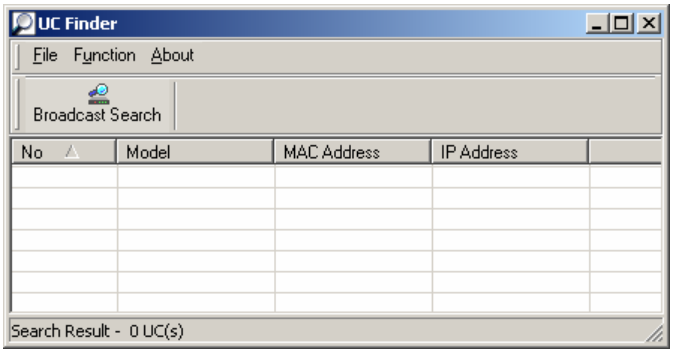

3. Select the **Create a desktop icon** option, and then click **Next** to continue.



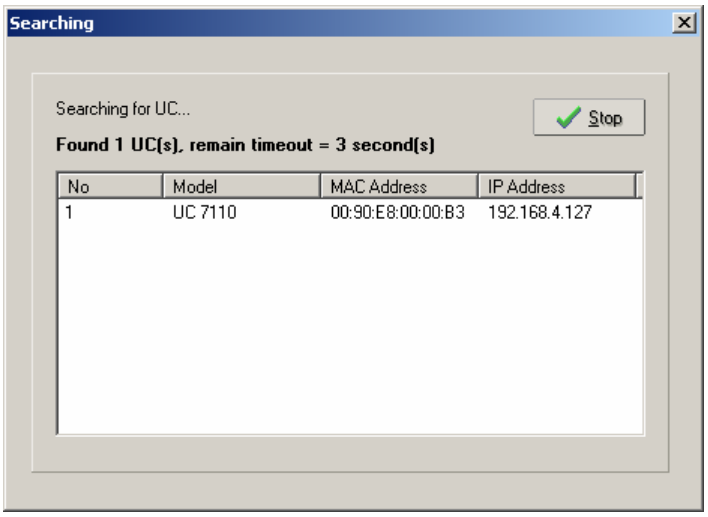
4. Check the **Launch UC Finder** checkbox to use UC Finder immediately after the installation has finished, and then click **Finish** to complete the installation.



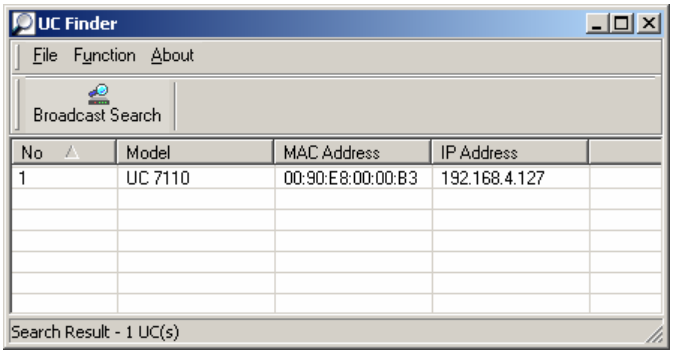

5. When the UC Finder window opens, click **Broadcast Search** to search for all MOXA embedded computers connected to the LAN.



6. The **Searching** window will show the embedded computers that have been located. You may click **Stop** as soon as the computer you are searching for is listed.




7. When the search is complete, the Broadcast Search window closes, and the **Model**, **MAC Address**, and **IP Address** of all embedded computers that were located will be listed in the UC Finder window.




ATTENTION

UC Finder is designed solely to locate the IP addresses of MOXA embedded computers connected to the network. UC Finder cannot be used to configure your embedded computers over the network. If you need to configure IP address or other parameters of an embedded computer, connect to the embedded computer's console utility by Telnet over the network, or through the serial console using the serial console cable that came with the product.

Note that you can launch UC finder by double clicking the UC Finder icon located on your PC's desktop.

# UC Finder for Linux

To use the Linux **ucfinder** utility, copy ucfinder from the CD-ROM to your Linux PC, and then use the following command to start ucfinder. The ucfinder utility will automatically broadcast a message over your LAN network to find the IP addresses of all UC's connected to the LAN.

***#./ucfinder***

# A

# System Commands

## busybox: µClinux normal command utility collection

### File manager

| | |
|---|---|
| **cp** | copy file |
| **ls** | list file |
| **ln** | make symbolic link file |
| **mount** | mount and check file system |
| **rm** | delete file |
| **chmod** | change file owner & group & user |
| **chown** | change file owner |
| **chgrp** | change file group |
| **sync** | sync file system; save system file buffer to hardware |
| **mv** | move file |
| **pwd** | display active file directly |
| **df** | list active file system space |
| **du** | estimate file space usage |
| **mkdir** | make new directory |
| **rmdir** | delete directory |
| **head** | print the first 10 lines of each file to standard output |
| **tail** | print the last 10 lines of each file to standard output |
| **touch** | update the access and modification times of each file to the current time |

### Editor

| | |
|---|---|
| **vi** | text editor |
| **cat** | dump file context |
| **grep** | print lines matching a pattern |
| **cut** | remove sections from each line of files |
| **find** | search for files in a directory hierarchy |
| **more** | dump file by one page |
| **test** | test if file exists or not |
| **echo** | echo string |

## Network

| | |
|---|---|
| **ping** | ping to test network |
| **route** | routing table manager |
| **netstat** | display network status |
| **ifconfig** | set network IP address |
| **tracerout** | trace route |
| **tftp** | tftp protocol |
| **telnet** | user interface to TELNET protocol |
| **ftp** | file transfer protocol |
| **iptables-restore** | restore iptables configuration file to network |
| **iptables** | iptables command |
| **iptables-save** | save recent iptables configuration to file |

## Process

| | |
|---|---|
| **kill** | kill process |
| **killall** | kill process by name |
| **ps** | report process status |
| **sleep** | suspend command on time |

## Other

| | |
|---|---|
| **dmesg** | dump kernel log message |
| **stty** | set serial port |
| **mknod** | make device node |
| **free** | display system memory usage |
| **date** | print or set the system date and time |
| **env** | run a program in a modified environment |
| **clear** | clear the terminal screen |
| **reboot** | reboot / power off/on the server |
| **halt** | halt the server |
| **gzip, gunzip, zcat** | compress or expand files |
| **hostname** | show system's host name |
| **tar** | tar archiving utility |

## MOXA Special Utilities

| | |
|---|---|
| **backupfs** | backup file system (user directory) |
| **bf** | build file system (user directory) |
| **cat /etc/version** | show user directory version |
| **upramdisk** | mount ramdisk |
| **downramdisk** | unmount ramdisk |
| **kversion** | show kernel version |
| **setinterface** | set UART interfaces program |

# B

# SNMP Agent with MIB II & RS-232 Like Group

The UC-7112/7110 has a built-in SNMP (Simple Network Management Protocol) agent that supports RFC1317 RS-232 like group and RFC 1213 MIB-II. The following table lists the variable implementation for the UC-7112/7110.

The full SNMP object ID of the UC-7112/7110 is **.iso.3.6.1.4.1.8691.12.7112** and **.iso.3.6.1.4.1.8691.12.7110**.

Note that the UC-7112/7110 does not support SNMP trap.

**RFC1213 MIB-II supported SNMP variables:**

| system MIB | interface MIB | at MIB | icmp MIB |
|---|---|---|---|
| sysDescr | ifNumber | atTable | icmpInMsgs |
| sysObjectID | ifTable | atIfIndex | icmpInErrors |
| sysUpTime | ifIndex | | icmpInDestUnreachs |
| sysContact | ifDescr | atPhysAddress | icmpInTimeExcds |
| sysName | ifType | atNetAddress | icmpInParmProbs |
| sysLocation | ifMtu | | icmpInSrcQuenchs |
| sysServices | ifSpeed | | icmpInRedirects |
| | ifPhysAddress | | icmpInEchos |
| | ifAdminStatus | | icmpInEchoReps |
| | ifOperStatus | | icmpInTimestamps |
| | ifLastChange | | icmpInAddrMasks |
| | ifInOctets | | icmpInAddrMaskReps |
| | ifInUcastPkts | | icmpOutMsgs |
| | ifInNUcastPkts | | icmpOutErrors |
| | ifInDiscards | | icmpOutDestUnreachs |
| | ifInErrors | | icmpOutTimeExcds |
| | ifInUnknownProtos | | icmpOutParmProbs |
| | ifOutOctets | | icmpOutSrcQuenchs |
| | ifOutUcastPkts | | icmpOutRedirects |
| | ifOutNUcastPkts | | icmpOutEchos |
| | ifOutDiscards | | icmpOutEchoReps |
| | ifOutErrors | | icmpOutTimestamps |
| | ifOutQLen | | icmpOutAddrMasks |
| | ifSpecific | | icmpOutAddrmaskReps |

| ip MIB | tcp MIB | udp MIB |
|---|---|---|
| ipForwarding | tcpRtoAlgorithm | udpInDatagrams |
| ipDefaultTTL | tcpRtoMin | udpNoPorts |
| ipInReceives | tcpRtoMax | udpInErrors |
| ipInHdrErrors | tcpMaxConn | udpOutDatagrams |
| ipInAddrErrors | tcpActiveOpens | udpTable |
| ipForwDatagrams | tcpPassiveOpens | udpLocalAddress |
| ipInUnknownProtos | tcpAttemptFails | udpLocalPort |
| ipInDiscards | tcpEstabResets | |
| ipInDelivers | tcpCurrEstab | |
| ipOutRequests | tcpInSegs | |
| ipOutDiscards | tcpOutSegs | |
| ipOutNoRoutes | tcpRetransSegs | |
| ipReasmTimeout | tcpConnTable | |
| ipReasmReqds | tcpConnState | |
| ipReasmFails | tcpConnLocalAddress | |
| ipFragOKs | tcpConnLocalPort | |
| ipFragFails | tcpConnRemAddress | |
| ipFragCreates | tcpConnRemPort | |
| ipAddrTable | tcpInErrs | |
| ipAdEntAddr | tcpOutRsts | |
| ipAdEntIfIndex | | |
| ipAdEntNetMask | | |
| ipAdEntBcastAddr | | |
| ipAdEntReasmMaxSize | | |
| ipRouteTable | | |
| ipRouteDest | | |
| ipRouteIfIndex | | |
| ipRouteMetric1 | | |
| ipRouteMetric2 | | |
| ipRouteMetric3 | | |
| ipRouteMetric4 | | |
| ipRouteNextHop | | |
| ipRouteType | | |
| ipRouteProto | | |
| ipRouteAge | | |
| ipRouteMask | | |
| ipRouteMetric5 | | |
| ipRouteInfo | | |
| ipNetToMediaTable | | |
| ipNetToMediaIfIndex | | |
| ipNetToMediaPhysAddress | | |
| ipNetToMediaNetAddress | | |
| ipNetToMediaType | | |
| ipRoutingDiscards | | |

| snmp MIB |
|---|
| snmpInPkts |
| snmpOutPkts |
| snmpInBadVersions |
| snmpInBadCommunityNames |
| snmpInBadCommunityUses |
| snmpInASNParseErrs |
| snmpInTooBigs |
| snmpInNoSuchNames |
| snmpInBadValues |
| snmpInReadOnlys |
| snmpInGenErrs |
| snmpInTotalReqVars |
| snmpInTotalSetVars |
| snmpInGetRequests |
| snmpInGetNexts |
| snmpInSetRequests |
| snmpInGetResponses |
| snmpInTraps |
| snmpOutTooBigs |
| snmpOutNoSuchNames |
| snmpOutBadValues |
| snmpOutGenErrs |
| snmpOutGetRequests |
| snmpOutGetNexts |
| snmpOutSetRequests |
| snmpOutTraps |
| snmpEnableAuthenTraps |

**RFC1317 RS-232 like group supported variables**

| rs232 MIB |
|---|
| rs232Number |
| rs232PortTable |
| rs232PortIndex |
| rs232PortType |
| rs232PortInSigNumber |
| rs232PortOutSigNumber |
| rs232PortInSpeed |
| rs232PortOutSpeed |
| rs232AsyncPortTable |
| rs232AsyncPortIndex |
| rs232AsyncPortBits |
| rs232AsyncPortStopBits |
| rs232AsyncPortParity |
| rs232InSigTable |
| rs232InSigPortIndex |
| rs232InSigName |
| rs232InSigState |
| rs232OutSigTable |
| rs232OutSigPortIndex |
| rs232OutSigName |
| rs232OutSigState |

# C
# FAQ

**FAQ 1** Why am I only able to use vfork( ),and cannot use fork( )?

**Answer 1** µClinux only supports vfork( ). It does not support fork( ). Note that when using vfork( ), the parent process will hang until the child process calls an exec group API, or exits.

**FAQ 2** When using a pthread group API, why can I not use SIGUSR1 and SIGUSR2?

**Answer 2** We cannot use the SIGUSR1 and SIGUSR2 signals since a pthread group API uses SIGUSR1 and SIGUSR2 to do a pthread control suspend, restart exit function. You will get the same result if you link the pthread. This means that you cannot use **`-1pthread`** to add an option to the linker.

**FAQ 3** What is the correct format for linking to an API?

**Answer 3** **`arm-elf-gcc –W1, -elf2flt`**
(In this example, the API converts elf format to flat format.)

# D

# Service Information

This appendix shows you how to contact MOXA for information about this and other products, and how to report problems.

In this appendix, we cover the following topics.

- **MOXA Internet Services**
- **Problem Report Form**
- **Product Return Procedure**

# MOXA Internet Services

Customer satisfaction is our primary concern. To ensure that customers receive the full benefit of our products, MOXA Internet Services has been set up to provide technical support, driver updates, product information, and user's manual updates.

## The following services are provided

E-mail for technical support.....support@moxa.com

Website for product information:.....http://www.moxa.com

# Problem Report Form

*MOXA UC-7112/7110*

| | |
|---|---|
| **Customer name:** | |
| **Company:** | |
| **Tel:** | **Fax:** |
| **Email:** | **Date:** |

1. **MOXA Product:** ☐ UC-7112 ☐ UC-7110
2. **Serial Number:** _____

**Problem Description:** Please describe the symptoms of the problem as clearly as possible, including any error messages you see. A clearly written description of the problem will allow us to reproduce the symptoms, and expedite the repair of your product.

# Product Return Procedure

For product repair, exchange, or refund, the customer must:

- Provide evidence of original purchase.
- Obtain a Product Return Agreement (PRA) from the sales representative or dealer.
- Fill out the Problem Report Form (PRF). Include as much detail as possible for a shorter product repair time.
- Carefully pack the product in an anti-static package, and send it, pre-paid, to the dealer. The PRA should be visible on the outside of the package, and include a description of the problem, along with the return address and telephone number of a technical contact.