# DA-681 Series Linux Manual

**Fourth Edition, August 2012**

www.moxa.com/product

# DA-681 Series Linux 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.

## Technical Support Contact Information

**www.moxa.com/support**

**Moxa Americas**
Toll-free: 1-888-669-2872
Tel: +1-714-528-6777
Fax: +1-714-528-6778

**Moxa China (Shanghai office)**
Toll-free: 800-820-5036
Tel: +86-21-5258-9955
Fax: +86-21-5258-5505

**Moxa Europe**
Tel: +49-89-3 70 03 99-0
Fax: +49-89-3 70 03 99-99

**Moxa Asia-Pacific**
Tel: +886-2-8919-1230
Fax: +886-2-8919-1231

# Table of Contents

# 1

# Introduction

Thank you for purchasing the Moxa DA-681 Series of x86 ready-to-run embedded computers. This manual introduces the software configuration and management of the DA-681-LX, which runs the Linux operating system. For hardware installation, connector interfaces, setup, and upgrading the BIOS, please refer to the "DA-681 Series Hardware User's Manual."

Linux is an open, scalable operating system that allows you to build a wide range of innovative, small footprint devices. Software written for desktop PCs can be easily ported to the embedded computer with a GNU cross compiler and a minimum of source code modifications. A typical Linux-based device is designed for a specific use, and is often not connected to other computers, or a number of such devices connect to a centralized, front-end host. Examples include enterprise tools such as industrial controllers, communications hubs, point-of-sale terminals, and display devices, which include HMIs, advertisement appliances, and interactive panels.

The following topics are covered in this chapter:

- **Overview**
- **Product Features**
- **Software Specifications**
- **Read-only Root File System**
- **Software Components**

# Overview

The DA-681 embedded computer is based on the Intel Celeron M processor and 910GMLE chipset, which supports standard X86 VGA, USB, PS/2 keyboard/mouse, 6 10/100 Mbps LAN ports, and SATA disk interface. In addition, the DA-681 supports a CompactFlash Socket and pre-installed embedded ready-to-run operating system. Programmers will find the full-function development kit a great benefit for developing software and building reliable communication applications.

The housing is a standard 1U, 19-inch wide rack-mounted rugged enclosure. This robust, rack-mountable design provides the hardened protection needed for industrial environment applications.

Moreover, the DPP-T models are in full compliance with IEC 61850-3 standards to meet the demands of power substation automation.

# Product Features

The DA-681 Series Basic System has the following features:

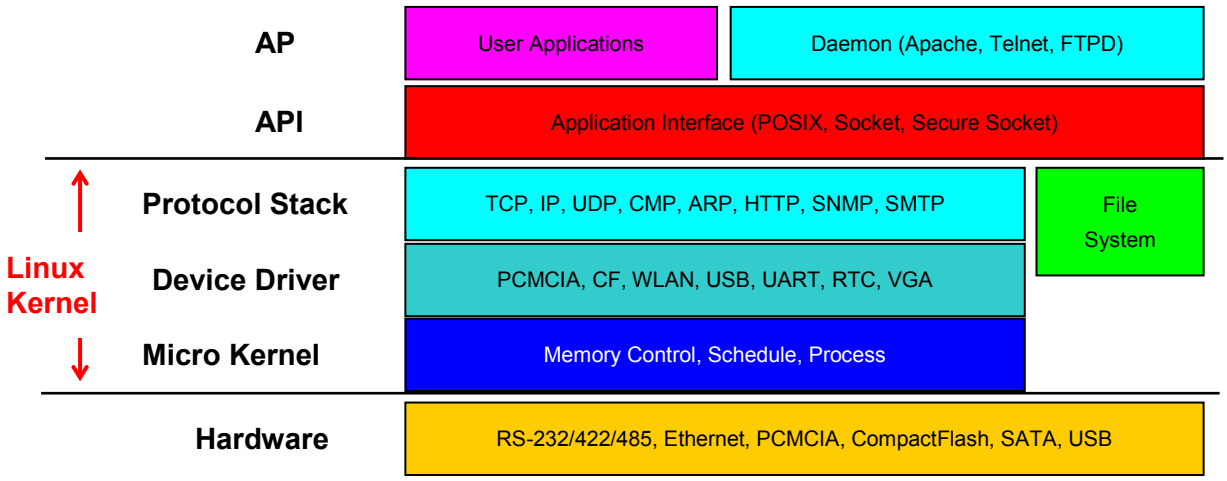
- IEC 61850-3 certified for power substation automation systems (DPP-T models only)
- Intel Celeron M 1 GHz processor with 400 MHz FSB
- Intel 910GMLE + ICH6M chipset
- 200-pin DDR2 SODIMM socket supporting DDR2 400 up to 1 GB (built-in 512 MB)
- 6 Ethernet ports for network redundancy
- 1 CompactFlash socket
- 1 SATA connector for storage expansion
- 4 RS-232 and 8 RS-485 serial ports (supports most nonstandard baudrates in this range)
- 2 USB 2.0 ports for high speed peripherals
- 19-inch rackmount, 1U high form factor
- Fanless Design
- 100/240 VAC/VDC power input (single power and dual power models available)

**ATTENTION**

Refer to section "Baud Rate Speed" for calculation of baud rate speed supported.

# Software Specifications

The Linux operating system pre-installed on the DA-681 embedded computer is the **Debian Etch 4.0r2** distribution. The Debian project is a worldwide group of volunteers who endeavor to produce an operating system distribution that composed entirely of free software. The Debian GNU/Linux follows the standard Linux architecture, making it easy to use programs that meet the POSIX standard. In addition to Standard POSIX APIs, device drivers for Moxa UART and other special peripherals are also included. An example software architecture is shown below:




**ATTENTION**

Refer to http://www.debian.org/ and http://www.gnu.org/ for information and documentation of the Debian GNU/Linux and free software concept.

**ATTENTION**

The above software architecture is only an example. Different models or different build revisions of the Linux operating system may include components not shown in the above graphic.

# Read-only Root File System

The pre-installed root file system is protected in a read-only partition to prevent file system crash problems normally caused by power loss. But some directories or files, such as **/home, /root**, **/var**, **/etc/network/**, **/etc/ppp/**, **/ect/openvpn/**, and **/etc/resolv.conf**, which need write permission, are located in another writable partition and formatted with the EXT2 file system. You can read/write above files or directories directly without re-mounting it.

# Software Components

The DA-681-LX pre-installed Debian Etch 4.0r2 Linux distribution has the following software components:

| Component | Version | Description |
|---|---|---|
| acpid | 1.0.4-5 | Utilities for using ACPI power management |
| adduser | 3.102 | Add and remove users and groups |
| apache2 | 2.2.3-4+etch6 | Next generation, scalable, extendable web server |
| apache2-mpm-prefork | 2.2.3-4+etch6 | Traditional model for Apache HTTPD 2.1 |
| apache2-utils | 2.2.3-4+etch6 | Utility programs for webservers |
| apache2.2-common | 2.2.3-4+etch6 | Next generation, scalable, extendable web server |
| apt | 0.6.46.4-0.1 | Advanced front-end for dpkg |
| apt-utils | 0.6.46.4-0.1 | APT utility programs |
| aptitude | 0.4.4-4 | Terminal-based apt frontend |
| base-files | 4 | Debian base system miscellaneous files |
| base-passwd | 3.5.11 | Debian base system master password and group |
| bash | 3.1dfsg-8 | The GNU Bourne Again SHell |
| binutils | 2.17-3 | The GNU assembler, linker and binary utilities |
| bridge-utils | 1.2-1 | Utilities for configuring the Linux ethernet bridge |

| Component | Version | Description |
|---|---|---|
| bsdmainutils | 6.1.6 | Collection of more utilities from FreeBSD |
| bsdutils | 2.12r-19etch1 | Basic utilities from 4.4BSD-Lite |
| busybox | 1.1.3-4 | Tiny utilities for small and embedded system |
| console-common | 0.7.69 | Basic infrastructure for text console configuration |
| console-data | 1.01-7 | Keymaps, fonts, charset maps, fallback table for console-tools |
| console-tools | 0.2.3dbs-65 | Linux console and font utilities |
| coreutils | 5.97-5.3 | The GNU core utilities |
| cpio | 2.6-18.1+etch1 | GNU cpio -- a program to manage archives of files |
| cpp | 4.1.1-15 | The GNU C preprocessor (cpp) |
| cpp-4.1 | 4.1.1-21 | The GNU C preprocessor |
| cron | 3.0pl1-100 | Management of regular background processing |
| debconf | 1.5.11etch1 | Debian configuration management system |
| debconf-i18n | 1.5.11etch1 | Full internationalization support for debcon |
| debian-archive-keyring | 2007.07.31~etc | GnuPG archive keys of the Debian archive |
| debianutils | 2.17 | Miscellaneous utilities specific to Debian |
| dhcp3-client | 3.0.4-13 | DHCP Client |
| dhcp3-common | 3.0.4-13 | Common files used by all the dhcp3* packages |
| dictionaries-common | 0.70.10 | Common utilities for spelling dictionary tools |
| diff | 2.8.1-11 | File comparison utilities |
| dmidecode | 2.8-4 | Dump Desktop Management Interface data |
| dnsutils | 9.3.4-2etch1 | Clients provided with BIND |
| dpkg | 1.13.25 | Package maintenance system for Debian |
| dselect | 1.13.25 | User tool to manage Debian packages |
| e2fslibs | 1.39+1.40-WIP-2006. | 11.14+dfsg-2etch1 ext2 filesystem libraries |
| e2fsprogs | 1.39+1.40-WIP-2006. | 11.14+dfsg-2etch1 ext2 file system utilities and libraries |
| findutils | 4.2.28-1etch1 | Utilities for finding files--find, xargs, an |
| ftp | 0.17-16 | The FTP client |
| g++ | 4.1.1-15 | The GNU C++ compiler |
| g++-4.1 | 4.1.1-21 | The GNU C++ compiler |
| gcc | 4.1.1-15 | The GNU C compiler |
| gcc-4.1 | 4.1.1-21 | The GNU C compiler |
| gcc-4.1-base | 4.1.1-21 | The GNU Compiler Collection (base package) |
| gnupg | 1.4.6-2 | GNU privacy guard - a free PGP replacement |
| gpgv | 1.4.6-2 | GNU privacy guard - signature verification tool |
| grep | 2.5.1.ds2-6 | GNU grep, egrep and fgrep |
| grub | 0.97-27etch1 | GRand Unified Bootloader |
| gzip | 1.3.5-15 | The GNU compression utility |
| hostname | 2.93 | Utility to set/show the host name or domain |
| ifrename | 28-1+etchnhalf | Rename network interfaces based on various static criteria |
| ifupdown | 0.6.8 | High level tools to configure network interfaces |
| initramfs-tools | 0.85i | Tools for generating an initramfs |
| initscripts | 2.86.ds1-38+et | Scripts for initializing and shutting down the system |
| iproute | 20061002-3 | Professional tools to control the networking in Linux kernels |
| iptables | 1.3.6.0debian1 | Administration tools for packet filtering and NAT netfilter and iptables provide a Linux kernel framework for stateful and stateless packet filtering, network and port address translation, and other IP packet manipulation. The framework is the successor to ipchains. |
| iputils-ping | 20020927-6 | Tools to test the reachability of network hosts |

| Component | Version | Description |
|---|---|---|
| klibc-utils | 1.4.34-2 | Small statically-linked utilities built with klibc |
| klogd | 1.4.1-18 | Kernel Logging Daemon |
| libacl1 | 2.2.41-1 | Access control list shared library |
| libattr1 | 2.4.32-1 | Extended attribute shared library |
| libbind9-0 | 9.3.4-2etch1 | BIND9 Shared Library used by BIND |
| libblkid1 | 1.39+1.40-WIP-2006.<br>. | 11.14+dfsg-2etch1 block device id library |
| klibc | 1.0.3-6 | Small statically-linked utilities built with klibc |
| libc6 | 2.3.6.ds1-13etch5 | GNU C Library: Shared libraries |
| libc6-dev | 2.3.6.ds1-13etch5 | GNU C Library: Development Libraries and Header Files |
| libc6-i686 | 2.3.6.ds1-13etch5 | GNU C Library: Shared libraries [i686 optimized] |
| libcap1 | 1.10-14 | Support for getting/setting POSIX.1e capabilities |
| libcomerr2 | 1.39+1.40-WIP-2006.<br>. | 11.14+dfsg-2etch1 common error description library |
| libconsole | 0.2.3dbs-65 | Shared libraries for Linux console and font |
| libcupsys2 | 1.2.7-4etch2 | Common UNIX Printing System(tm) - libs |
| libdb4.2 | 4.2.52+dfsg-2 | Berkeley v4.2 Database Libraries [runtime] |
| libdb4.3 | 4.3.29-8 | Berkeley v4.3 Database Libraries [runtime] |
| libdb4.4 | 4.4.20-8 | Berkeley v4.4 Database Libraries [runtime] |
| libdevmapper1.02 | 1.02.08-1 | The Linux Kernel Device Mapper userspace library |
| libdns22 | 9.3.4-2etch1 | DNS Shared Library used by BIND |
| libedit2 | 2.9.cvs.20050518-2.<br>2 | BSD editline and history libraries |
| libevent1 | 1.1a-1 | An asynchronous event notification library |
| libgc1c2 | 6.8-1 | Conservative garbage collector for C and C++ |
| libgcc1 | 4.1.1-21 | GCC support library |
| libgcrypt11 | 1.2.3-2 | LGPL Crypto library - runtime library |
| libgdbm3 | 1.8.3-3 | GNU dbm database routines (runtime version) |
| libgnutls13 | 1.4.4-3 | The GNU TLS library - runtime library |
| libgpg-error0 | 1.4-1 | Library for common error values and messages |
| libgpmg1 | 1.19.6-25 | General Purpose Mouse - shared library |
| libgssapi2 | 0.10-4 | A mechanism-switch gssapi library |
| libidn11 | 0.6.5-1 | GNU libidn library, implementation of IETF I |
| libisc11 | 9.3.4-2etch1 | ISC Shared Library used by BIND |
| libisccc0 | 9.3.4-2etch1 | Command Channel Library used by BIND |
| libisccfg1 | 9.3.4-2etch1 | Config File Handling Library used by BIND |
| libiw28 | 28-1 | Wireless tools - library |
| libklibc | 1.4.34-2 | Minimal libc subset for use with initramfs |
| libkrb53 | 1.4.4-7etch4 | MIT Kerberos runtime libraries |
| libldap2 | 2.1.30-13.3 | OpenLDAP libraries |
| liblocale-gettext-perl | 1.05-1 | Using libc functions for internationalization in Perl |
| liblockfile1 | 1.06.1 | NFS-safe locking library, includes dotlockfile program |
| liblwres9 | 9.3.4-2etch1 | Lightweight Resolver Library used by BIND |
| liblzo1 | 1.08-3 | Data compression library (old version) |
| liblzo2-2 | 2.02-2 | Data compression library |
| libmagic1 | 4.17-5etch3 | File type determination library using “magic” numbers |
| libmysqlclient15off | 5.0.32-7etch5 | mysql database client library |
| libncurses5 | 5.5-5 | Shared libraries for terminal handling |
| libncursesw5 | 5.5-5 | S Shared libraries for terminal handling<br>(wide character support) |
| libnet-lite-ftp-perl | 0.47-2 | Perl FTP client with support for TLS |
| libnet-ssleay-perl | 1.30-1 | Perl module for Secure Sockets Layer (SSL) |

| Component | Version | Description |
|---|---|---|
| libnet-telnet-perl | 3.03-1 | Script telnetable connections |
| libnewt0.52 | 0.52.2-10 | Not Erik's Windowing Toolkit - text mode windowing with slang |
| libnfsidmap2 | 0.18-0 | An nfs idmapping library |
| libopencdk8 | 0.5.9-2 | Open Crypto Development Kit (OpenCDK) (runtime) |
| libpam-modules | 0.79-5 | Pluggable Authentication Modules for PAM |
| libpam-runtime | 0.79-5 | Runtime support for the PAM library |
| libpam0g | 0.79-5 | Pluggable Authentication Modules library |
| libpcap0.8 | 0.9.5-1 | System interface for user-level packet capture |
| libpci2 | 2.1.11-3 | Obsolete shared library for accessing pci devices |
| libpcre3 | 6.7+7.4-3 | Perl 5 Compatible Regular Expression Library |
| libpopt0 | 1.10-3 | lib for parsing cmdline parameters |
| libpq4 | 8.1.11-0etch1 | PostgreSQL C client library |
| libreadline5 | 5.2-2 | GNU readline and history libraries, run-time |
| libroken16-heimdal | 0.7.2.dfsg.1-10 | Libraries for Heimdal Kerberos |
| librpcsecgss3 | 0.14-2etch3 | Allows secure rpc communication using the rpcsec_gss protocol |
| libsasl2 | 2.1.22.dfsg1-8 | Authentication abstraction library |
| libsasl2-2 | 2.1.22.dfsg1-8 | Authentication abstraction library |
| libselinux1 | 1.32-3 | SELinux shared libraries |
| libsemanage1 | 1.8-1 | Shared libraries used by SELinux policy manipulation tools |
| libsensors3 | 2.10.1-3 | Library to read temperature/voltage/fan sensors |
| libsepol1 | 1.14-2 | Security Enhanced Linux policy library for changing policy binaries |
| libsigc++-2.0-0c2a | 2.0.17-2 | Type-safe Signal Framework for C++ - runtime |
| libslang2 | 2.0.6-4 | The S-Lang programming library - runtime version |
| libslp1 | 1.2.1-6.2 | OpenSLP libraries |
| libsnmp-base | 5.2.3-7etch2 | NET SNMP (Simple Network Management Protocol) MIBs and Docs |
| libsnmp9 | 5.2.3-7etch2 | NET SNMP (Simple Network Management Protocol ) MIBs and Docs |
| libss2 | 1.39+1.40-WIP-2006. | 11.14+dfsg-2etch1 command-line interface parsing library |
| libssl0.9.8 | 0.9.8c-4etch1 | SSL shared libraries |
| libssp0 | 4.1.1-21 | GCC stack smashing protection library |
| libstdc++6 | 4.1.1-21 | The GNU Standard C++ Library v3 |
| libstdc++6-4.1-dev | 4.1.1-21 | The GNU Standard C++ Library v3 (development files) |
| libsysfs2 | 2.1.0-1 | Interface library to sysfs |
| libtasn1-3 | 0.3.6-2 | Manage ASN.1 structures (runtime) |
| libtasn1-3-bin | 0.3.6-2 | Manage ASN.1 structures (binaries) |
| libtext-charwidth-perl | 0.04-4 | Get display widths of characters on the term |
| libtext-iconv-perl | 1.4-3 | Converts between character sets in Perl |
| libtext-wrapi18n-perl | 0.06-5 | Internationalized substitute of Text:Wrap |
| libusb-0.1-4 | 0.1.12-5 | userspace USB programming library |
| libuuid1 | 1.39+1.40-WIP-2006. | 11.14+dfsg-2etch1 universally unique id library |
| libvolume-id0 | 0.105-4 | libvolume_id shared library |
| libwrap0 | 7.6.dbs-13 | Wietse Venema's TCP wrappers library |
| linux-image-2.6-686 | 2.6.18+6etch2 | Linux kernel 2.6 image on PPro/Celeron/PII/PIII/P4 |
| linux-image-2.6.18-5-686 | 2.6.18.dfsg.1-17 | Linux 2.6.18 image on PPro/Celeron/PII/PIII/P4 |

| Component | Version | Description |
|---|---|---|
| linux-kernel-headers | 2.6.18-7 | Linux Kernel Headers for development |
| locales | 2.3.6.ds1-13etch5 | GNU C Library: National Language (locale) data [support] |
| lockfile-progs | 0.1.10 | Programs for locking and unlocking files and mailboxes |
| login | 4.0.18.1-7 | System login tools |
| logrotate | 3.7.1-3 | Log rotation utility |
| lsb-base | 3.1-23.2etch1 | Linux Standard Base 3.1 init script function |
| make | 3.81-2 | The GNU version of the "make" utility. |
| makedev | 2.3.1-83 | Creates device files in /dev |
| manpages | 2.39-1 | Manual pages about using a GNU/Linux system |
| mawk | 1.3.3-11 | A pattern scanning and text processing language |
| mime-support | 3.39-1 | MIME files "mime.types" & "mailcap", and support programs |
| minicom | 2.2-5 | Friendly menu driven serial communication program |
| mktemp | 1.5-2 | Makes unique filenames for temporary files |
| modconf | 0.3.1 | Device Driver Configuration |
| module-init-tools | 3.3-pre4-2 | Tools for managing Linux kernel modules |
| mount | 2.12r-19etch1 | Tools for mounting and manipulating filesystems |
| mysql-common | 5.0.32-7etch8 | Mysql database common files<br>(e.g. /etc/mysql my.cnf) |
| ncurses-base | 5.5-5 | Descriptions of common terminal types |
| ncurses-bin | 5.5-5 | Terminal-related programs and man pages |
| net-tools | 1.60-17 | The NET-3 networking toolkit |
| netbase | 4.29 | Basic TCP/IP networking system |
| nfs-common | 1.0.10-6+etch.1 | NFS support files common to client and server |
| openbsd-inetd | 0.20050402-6 | The OpenBSD Internet Superserver |
| openssh-client | 4.3p2-9 | Secure shell client, an rlogin/rsh/rcp replacement |
| openssh-server | 4.3p2-9 | Secure shell server, an rshd replacement |
| openssl | 0.9.8c-4etch3 | Secure Socket Layer (SSL) binary and related cryptographic tools |
| openvpn | 2.0.9-4etch1 | Virtual Private Network daemon |
| passwd | 4.0.18.1-7 | Change and administer password and group data |
| pciutils | 2.2.4~pre4-1 | Linux PCI Utilities |
| perl | 5.8.8-7etch3 | Larry Wall's Practical Extraction and Report |
| perl-base | 5.8.8-7etch3 | The Pathologically Eclectic Rubbish Lister |
| perl-modules | 5.8.8-7etch3 | Core Perl modules |
| portmap | 5-26 | The RPC portmapper |
| ppp | 2.4.4rel-8 | Point-to-Point Protocol (PPP) daemon |
| pppconfig | 2.3.15.etch1 | A text menu based utility for configuring ppp |
| pppoe | 3.8-1.1 | PPP over Ethernet driver |
| procps | 3.2.7-3 | /proc file system utilities |
| proftpd | 1.3.0-19etch1 | Versatile, virtual-hosting FTP daemon |
| readline-common | 5.2-2 | GNU readline and history libraries, common files |
| sed | 4.1.5-1 | The GNU sed stream editor |
| snmp | 5.2.3-7etch4 | NET SNMP (Simple Network Management Protocol) Apps |
| snmpd | 5.2.3-7etch4 | NET SNMP (Simple Network Management Protocol) Agents |
| ssh | 4.3p2-9etch3 | Secure shell client and server (transitional package) |
| ssl-cert | 1.0.14 | Simple debconf wrapper for openssl |
| strace | 4.5.14-2 | A system call tracer |
| sysklogd | 1.4.1-18 | System Logging Daemon |
| sysv-rc | 2.86.ds1-38+et | System-V-like runlevel change mechanism |
| sysvinit | 2.86.ds1-38+et | System-V-like init utilities |
| sysvinit-utils | 2.86.ds1-38+et | System-V-like utilities |
| tar | 1.16-2etch1 | GNU tar |

| Component | Version | Description |
|---|---|---|
| tcpd | 7.6.dbs-13 | Wietse Venema's TCP wrapper utilities |
| tcpdump | 3.9.5-2etch1 | A powerful tool for network monitoring and data acquisition |
| telnet | 0.17-34 | The telnet client |
| telnetd | 0.17-34 | The telnet server |
| tftpd | 0.17-15 | Trivial file transfer protocol server |
| time | 1.7-21 | The GNU time program for measuring cpu resource usage |
| traceroute | 1.4a12-21 | Traces the route taken by packets over a TCP |
| tzdata | 2008e-1etch3 | Time Zone and Daylight Saving Time Data |
| ucf | 2.0020 | Update Configuration File: preserves user changes to config files. |
| udev | 0.105-4 | /dev/ and hotplug management daemon |
| update-inetd | 4.27-0.5 | inetd.conf updater |
| usbmount | 0.0.14.1 | Automatically mount and unmount USB mass storage devices |
| usbutils | 0.72-7 | USB console utilities |
| util-linux | 2.12r-19etch1 | Miscellaneous system utilities |
| vim | 7.0-122+1etch3 | Vi IMproved - enhanced vi editor |
| vim-common | 7.0-122+1etch3 | Vi IMproved - Common files |
| vim-runtime | 7.0-122+1etch3 | Vi IMproved - Runtime files |
| vim-tiny | 7.0-122+1etch3 | Vi IMproved - enhanced vi editor - compact version |
| w3m | 0.5.1-5.1 | WWW browsable pager with excellent tables/frames support |
| wget | 1.10.2-2 | Retrieves files from the web |
| whiptail | 0.52.2-10 | Displays user-friendly dialog boxes from shell scripts |
| whois | 4.7.20 | The GNU whois client |
| zlib1g | 1.2.3-13 | Compression library - runtime |

# 2

# Software Configuration

In this chapter, we explain how to operate a DA-681-LX computer directly or from a PC near you. There are three ways to connect to the DA-681-LX computer: through VGA monitor, by using Telnet over the network, or by using an SSH console from a Windows or Linux machine. This chapter describes basic Linux operating system configurations. The advanced network management and configuration will be described in the next chapter "Managing Communications."

The following topics are covered in this chapter:

- **Starting from a VGA Console**
- **Connecting from a Telnet Console**
- **Connecting from an SSH Console**
  - Windows Users
  - Linux Users
- **Adjusting the System Time**
  - Setting the Time Manually
  - NTP Client
  - Updating the Time Automatically
- **Enabling and Disabling Daemons**
- **Setting the Run-Level**
- **Cron—Daemon for Executing Scheduled Commands**
- **Inserting a SATA Hard Drive into the Computer**
- **Inserting a USB Storage Device into the Computer**
- **Inserting a CompactFlash Card into the Computer**
- **Checking the Linux Version**
- **APT—Installing and Removing Packages**
- **WDT (Watchdog Timer)**

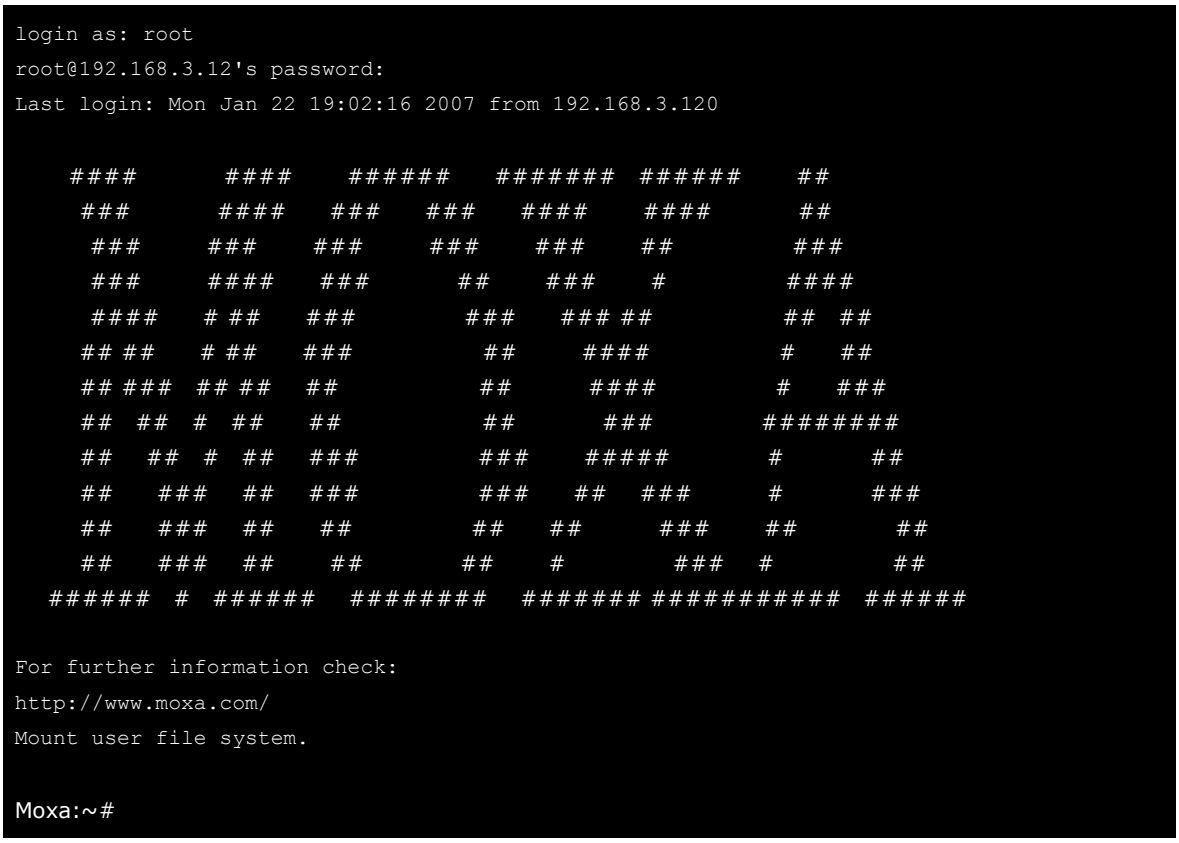
# Starting from a VGA Console

Connect the display monitor to the DA-681-LX VGA connector, and then power it up by connecting it to the power adaptor. It takes about 30 to 60 seconds for the system to boot up. Once the system is ready, a login screen will appear on your monitor.

To log in, type the login name and password as requested. The default values are both **root**.

**Login: root**

**Password: root**




# Connecting from a Telnet Console

The DA-681-LX computer comes with 6 10/100 Mbps Ethernet ports named LAN1 to LAN6. The default IP addresses and netmasks of the network interfaces are as follows:

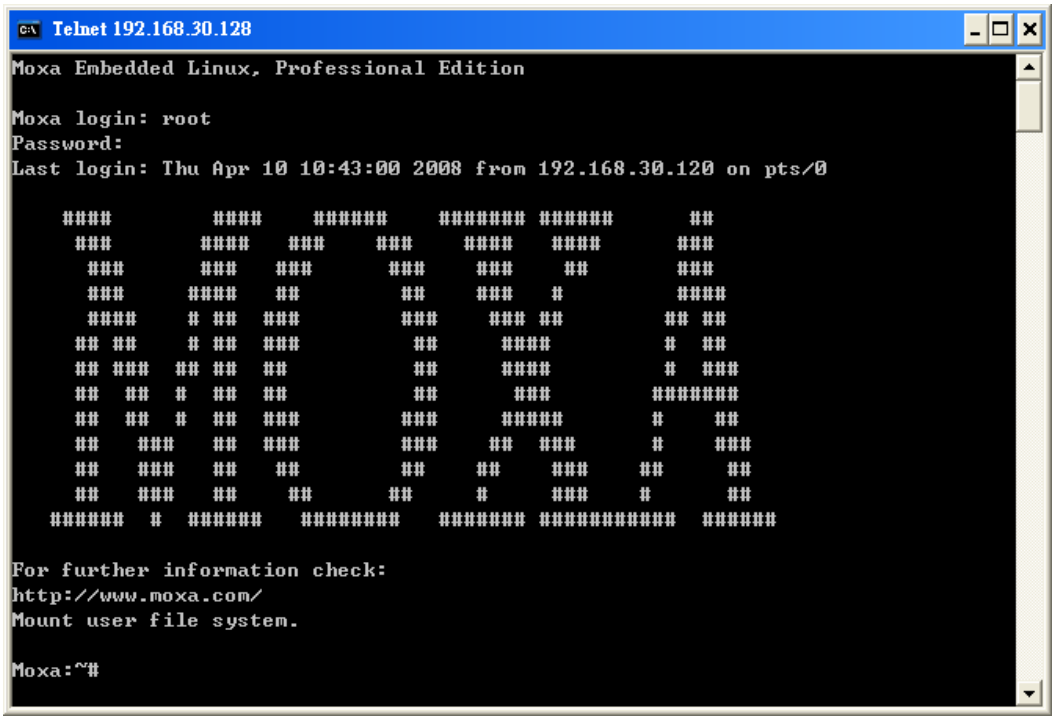
| | Default IP Address | Netmask |
|---|---|---|
| LAN 1 | 192.168.3.127 | 255.255.255.0 |
| LAN 2 | 192.168.4.127 | 255.255.255.0 |
| LAN 3 | 192.168.5.127 | 255.255.255.0 |
| LAN 4 | 192.168.6.127 | 255.255.255.0 |
| LAN 5 | 192.168.5.127 | 255.255.255.0 |
| LAN 6 | 192.168.6.127 | 255.255.255.0 |

Before using the Telnet client, you should change the IP address of your development workstation so that the network ports are on the same subnet as the IP address for the LAN port that you connect to. For example, if you connect to LAN 1, you could set your PC's IP address to 192.168.3.126, and the netmask to 255.255.255.0. If you connect to LAN 2, you can set your PC's IP address to 192.168.4.126, and the netmask to 255.255.255.0.

Use a cross-over Ethernet cable to connect your development workstation directly to the target computer, or use a straight-through Ethernet cable to connect the computer to a LAN hub or switch. Next, use a Telnet client on your development workstation to connect to the target computer. After a connection has been established, type the login name and password as requested to log on to the computer. The default values are both **root**.

**Login: root**

**Password: root**




**ATTENTION**

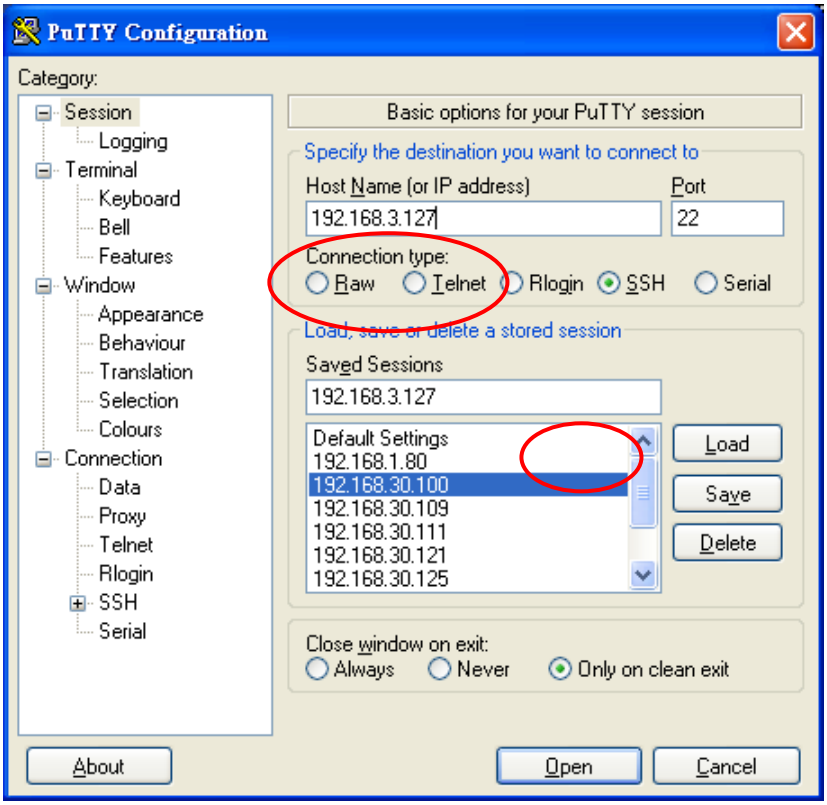
If you cannot get connected on the first try, re-check the IP address and netmask settings, and then unplug and re-plug the DA-681-LX's power cord.

# Connecting from an SSH Console

The DA-681-LX computer supports an SSH Console to offer users with better security over the network compared to Telnet.

## Windows Users

Click on the link http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html to download **PuTTY** (free software) to set up an SSH console for the DA-681-LX in a Windows environment. The following screen shows an example of the configuration that is required.




## Linux Users

From a Linux machine, use the **ssh** command to access the DA-681-I-LX's console utility via SSH.

**#ssh 192.168.3.127**

Select **yes** to open the connection.

```
[root@bee_notebook root]# ssh 192.168.3.127
The authenticity of host '192.168.3.127 (192.168.3.127)' can't be established.
RSA key fingerprint is 8b:ee:ff:84:41:25:fc:cd:2a:f2:92:8f:cb:1f:6b:2f.
Are you sure you want to continue connection (yes/no)? yes_
```

# Adjusting the System Time

The DA-681-LX has two time settings. One is the system time, and the other is provided by an RTC (Real Time Clock) built into the DA-681- LX's hardware.

## Setting the Time Manually

Use the **date** command to query the current system time or set a new system time. Use **hwclock** to query the current RTC time or set a new RTC time.

Use the following command to set the system time.

**# date MMDDhhmmYYYY**

MM: Month
DD: Date
hhmm: Hour and Minute
YYYY: Year

Use the following command to write the current system time to the RTC.

**# hwclock –w**

```
Moxa:~# date
            Fri Jun 23 23:30:31 CST 2000

Moxa:~# hwclock
Fri Jun 23 23:30:35 2000  -0.557748 seconds
Moxa:~# date 120910002004
Thu Dec  9 10:00:00 CST 2004
Moxa:~# hwclock -w
Moxa:~# date ; hwclock
Thu Dec  9 10:01:07 CST 2004
Thu Dec  9 10:01:08 2004  -0.933547 seconds
Moxa:~#
```

# NTP Client

The DA-681-LX has a built-in NTP (Network Time Protocol) client that is used to initialize a time request to a remote NTP server. Use **ntpdate** to update the system time.

**#ntpdate time.stdtime.gov.tw**
**#hwclock –w**

Visit http://www.ntp.org for more information about NTP and NTP server addresses.

```
Moxa:~# date ; hwclock
Sat Jan  1 00:00:36 CST 2000
Sat Jan  1 00:00:37 2000  -0.772941 seconds
Moxa:~#
Moxa:~# ntpdate time.stdtime.gov.tw
 9 Dec 10:58:53 ntpdate[207]: step time server 220.130.158.52 offset 155905087.9
84256 sec
Moxa:~#
Moxa:~# hwclock -w
Moxa:~# date ; hwclock
Thu Dec  9 10:59:11 CST 2004
Thu Dec  9 10:59:12 2004  -0.844076 seconds
Moxa:~#
```

**ATTENTION**

Before using the NTP client utility, check your IP address and network settings to make sure an Internet connection is available.

# Updating the Time Automatically

This section describes how to use a shell script to update the time automatically.

**Example shell script for updating the system time periodically**

```
#!/bin/sh
ntpdate time.nist.gov
# You can use the time server's ip address or domain
# name directly. If you use domain name, you must
# enable the domain client on the system by updating
# /etc/resolv.conf file.
hwclock –w
sleep 100
# Updates every 100 seconds. The min. time is 100 seconds.
# Change 100 to a larger number to update RTC less often.
```

Save the shell script using any file name. For example, **fixtime**.

**How to run the shell script automatically when the kernel boots up**

Because the root file system is mounted in Read-only mode, we need to re-mount it using writable permission.

```
# mount -o remount,rw /dev/hda1 /
```

Copy the example shell script **fixtime** to directory **/etc/init.d**, and then use **chmod 755 fixtime** to change the shell script mode.

```
# chmod 755 fixtime
```

Next, use **vi** editor to edit the file **/etc/inittab**.

```
# vi /etc/inittab
```

Add the following line to the bottom of the file:

```
ntp : 2345 : respawn : /etc/init.d/fixtime
```

After you finish writing or modifying the code, remember to execute "umount /" to change the root directory back to Read-only mode.

```
# umount /
```

Use the command **#init q** to re-initialize the kernel.

```
# init q
```

# Enabling and Disabling Daemons

The following daemons are enabled when the DA-681-LX boots up for the first time.

- **snmpd** SNMP Agent Daemon
- **telnetd** Telnet Server/Client Daemon
- **inetd** Internet Daemons
- **ftpd** FTP Server/Client Daemon
- **sshd** Secure Shell Server Daemon
- **httpd** Apache WWW Server Daemon

Type the command **ps –ef** to list all processes currently running.

```
Moxa:~# ps -ef
UID          PID  PPID    C STIME  TTY          TIME CMD
root         1     0 0 Feb18  ?        00:00:01 init [2]
root         2     1 0 Feb18  ?        00:00:00 [migration/0]
root         3     1 0 Feb18  ?        00:00:00 [ksoftirqd_0]
root         4     1 0 Feb18  ?        00:00:00 [events/0]
root         5     1 0 Feb18  ?        00:00:00 [khelper]
root         6     1 0 Feb18  ?        00:00:00 [kthread]
root         9     6 0 Feb18  ?        00:00:00 [kblockd/0]
root        10     6 0 Feb18  ?        00:00:00 [kacpid]
root       107     6     0 Feb18  ?        00:00:00 [kseriod]
root       143     6     0 Feb18  ?        00:00:00 [pdflush]
root       144     6     0 Feb18  ?        00:00:00 [pdflush]
root       145     6     0 Feb18  ?        00:00:00 [kswapd0]
root       146     6     0 Feb18  ?        00:00:00 [aio/01]
root       622     6     0 Feb18  ?        00:00:00 [khubd]
root       763     6     0 Feb18  ?        00:00:00 [scsi_eh_0]
root       765     6     0 Feb18  ?        00:00:00 [usb-storage]
root      1119     1     0 Feb18  ?        00:00:00 udevd --daemon
root      1527     6     0 Feb18  ?        00:00:00 [kpsmoused]
root      1754     6     0 Feb18  ?        00:00:00 [kmirrord]
daemon        2094     1     0 Feb18  ?        00:00:00 /sbin/portmap
root      2311     1     0 Feb18  ?        00:00:00 /usr/sbin/acpid
-c /etc/acpi/evemts –s /var/run/acpid.socket
root      2318     1     0 Feb18  ?        00:00:00 /usr/sbin/inetd
```

To run a private daemon, you can edit the file **rc.local** as follows:

1. Because the root file system is mounted in Read-only mode, you need to re-mount it with write permission.

```
Moxa:~# mount -o remount,rw /dev/hda1 /
```

2. Type **cd /etc/** to change directories.

```
Moxa:~# cd /etc/
```

3. Type **vi rc.local** to edit the configuration file with vi editor.

```
Moxa:/etc/# vi rc.local
```

4. Next, add the application daemon that you want to run. We use the example program **tcps2-release** to illustrate, and configure it to run in the background.

```
# !/bin/sh
# Add the daemonyou want to run daemon
/root/tcps2-release &
```

5. After you finish writing or modifying the code, remember to execute "umount /" to change the root directory back to Read-only mode.

```
Moxa:~# umount /
```

6. You should be able to find the enabled daemon after you reboot the system.

```
Moxa:~# ps -ef
UID      PID  PPID     C STIME  TTY      TIME CMD
root       1     0     0 Feb18  ?        00:00:01 init [2]
root       2     1     0 Feb18  ?        00:00:00 [migration/0]
root       3     1     0 Feb18  ?        00:00:00 [ksoftirqd_0]
root       4     1     0 Feb18  ?        00:00:00 [events/0]
root       5     1     0 Feb18  ?        00:00:00 [khelper]
root       6     1     0 Feb18  ?        00:00:00 [kthread]
root       9     6     0 Feb18  ?        00:00:00 [kblockd/0]
root      10     6     0 Feb18  ?        00:00:00 [kacpid]
root     107     6     0 Feb18  ?        00:00:00 [kseriod]
root     143     6     0 Feb18  ?        00:00:00 [pdflush]
root     144     6     0 Feb18  ?        00:00:00 [pdflush]
root     145     6     0 Feb18  ?        00:00:00 [kswapd0]
root     146     6     0 Feb18  ?        00:00:00 [aio/01]
root     622     6     0 Feb18  ?        00:00:00 [khubd]
root     763     6     0 Feb18  ?        00:00:00 [scsi_eh_0]
root     765     6     0 Feb18  ?        00:00:00 [usb-storage]
root    1119     1     0 Feb18  ?        00:00:00 udevd --daemon
root    1527     6     0 Feb18  ?        00:00:00 [kpsmoused]
root    1754     6     0 Feb18  ?        00:00:00 [kmirrord]
daemon  2094     1     0 Feb18  ?        00:00:00 /sbin/portmap
root    2311     1     0 Feb18  ?        00:00:00 /usr/sbin/acpid
-c /etc/acpi/evemts –s /var/run/acpid.socket
root    2318     1     0 Feb18  ?        00:00:00 /usr/sbin/inetd
root    2452     1     0 Feb18  ?        00:00:00 /root/tcps2-release
```

# Setting the Run-Level

To set the Linux run-level and execution priority of a program, use the following command (because the root file system is mounted in Read-only mode, we need to re-mount it with write permission).

```
Moxa:~# mount -o remount,rw /dev/hda1 /
```

Edit a shell script to execute **/root/tcps2-release** and save to **tcps2** as an example. This program can be found in Example Directory in CD-ROM.

**#cd /etc/rc2.d**
**#ln –s /etc/root/tcps2 S60tcps2**

or

**#ln –s /etc/root/tcps2 k30tcps2**

```
MOXA:~# cd /etc/rc2.d
MOXA:/etc/rc2.d#
MOXA:/etc/rc2.d# ls
S19nfs-common     S25nfs-user-server    S99showreadyled
S20snmpd          S55ssh
S24pcmcia              S99rmnologin
MOXA:/etc/rc2.d#
MOXA:/etc/rc2.d# ln –s /root/tcps2-release S60tcps2
MOXA:/etc/rc2.d# ls
S19nfs-common     S25nfs-user-server    S99rmnologin
S20snmpd          S55ssh                S99showreadyled
S24pcmcia              S60tcps2
MOXA:/etc/rc2.d#
```

The command **SxxRUNFILE** has the following meaning:

**S:** **Start the run file while Linux boots up.**
**xx:** **A number between 00-99. The smaller number has a higher priority.**
**RUNFILE:** **The script file name**

The command **KxxRUNFILE** has the following meaning:

**K:** **Start the run file while Linux shuts down or halts.**
**xx:** **A number between 00-99. The smaller number has a higher priority.**
**RUNFILE:** **The script file name**

To remove the daemon, remove the run file from /etc/rc2.d by using the following command:

```
# rm –f /etc/rc2.d/S60tcps2
```

After you finish writing or modifying the code, remember to execute "umount /" to change the root directory back to Read-only mode.

```
MOXA:~# umount /
```

# Cron—Daemon for Executing Scheduled Commands

The Cron daemon will search **/etc/crontab** for crontab files, which are named after accounts in **/etc/passwd**.

Cron wakes up every minute and checks each command to see if it should be run in that minute. When executing commands, output is mailed to the owner of the **crontab** (or to the user named in the MAILTO environment variable in the **crontab**, if such a user exists).

Modify the file **/etc/crontab** to set up your scheduled applications. **Crontab** files have the following format:

| mm | h | dom | mon | dow | user | command |
|---|---|---|---|---|---|---|
| minute | hour | date | month | week | user | command |
| 0-59 | 0-23 | 1-31 | 1-12 | 0-6 (0 is Sunday) | | |

For example, if you want to launch a program at 8:00 every day

```
#minute hour  date  month  week  user   command
*       8     *     *      *     root   /path/to/your/program
```

The following example demonstrates how to use **Cron** to update the system time and RTC time every day at 8:00.

1. Write a shell script named fixtime.sh and save it to **/home/**.

   **#!/bin/sh**
   **ntpdate time.nist.gov**
   **hwclock –w**
   **exit 0**

2. Change mode of **fixtime.sh**

   **# chmod 755 fixtime.sh**

3. Modify /etc/crontab file to run fixtime.sh at 8:00 every day.

   Add the following line to the end of crontab:
   *** 8 * * * root /home/fixtime.sh**

# Inserting a SATA Hard Drive into the Computer

The DA-681 offers one hard drive connector supporting a SATA-based disk that can be added to the computer. Follow the next step.

1. Change the access right of the file system **mount –o remount,rw /dev/hda1 /**

   Then use mount command to check if the status has been changed from **read only** to **read and write**.

```
Moxa:~# mount
rootfs on / type rootfs (rw)
none on /sys type sysfs (rw)
none on /proc type proc (rw)
udev on /dev type tmpfs (rw)
/dev/hdal on / type ext2 (ro)
/dev hdal on /dev/.static/dev type ext2 (ro)
tmpfs on /dev shm type tmpfs (rw, nosuid, nodev)
devpts on /dev/pts type devpts (rw, nosuid, noexec)
none on /temp type tmpfs (rw)
/dev/mtdblock0 on /home type jffs2 (rw)
Moxa:~# mount -0 remount, rw /dev/hda1 /
Moxa:~# mount
rootfs on / type rootfs (rw)
none on / sys type sysfs (rw)
none on / proc type proc (rw)
udev on /dev type tmpfs (rw)
/dev/hda1 on / type ext2 (rw)
/dev/hda1 on /dev/.static/dev type ext2 (rw)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
devpts on /dev/pts type devpts (rw,nosuid,noexec)
none on /tmp type tmpfs (rw)
/dev/mtdblock0 on /home type jffs2 (rw)
Moxa:~#
```

2. Edit **/boot/grub/menu.lst**.

   Change the device name of DOM, from root=/dev/hda => root=/dev/sdb, and then save the file.

```
title                Debian GNU/Linux, kernel 2.6.18-5-686
root                 (hd0,0)
kernel               /boot/vmlinuz-2.6.18-5-686 root*/dev/sdb1 ro
initrd               /boot/initrd.img-2.6.18-5-686
savedefault
```

```
title                Debian GNU/Linux, kernel 2.6.18-5-686 (single-user mode)
root                 (hd0,0)
kernel               /boot/vmlinuz-2.6.18-5-686 root*/dev/sdb1 ro single
initrd               /boot/initrd.img-2.6.18-5-686
savedefault
```

3. Edit the **/etc/fstab** file and change the selected hard disk for system bootup.

```
# /etc/fstab: static file system information.
#
# <file system>   <mount> point>    <type>       <options>    <dump>   <pass>
Proc              /proc             proc         defaults     0        0
/dev/sdb1         /                 ext2         ro,defaults,errors=remount-ro 0   1
# Mount the CF,   /dev/hdb1, on     /mnt. You should edit this line
#/dev/hdb1        /mnt              ext3         defaults,errors=remount-ro 0    1
none              /tmp              tmpfs        defaults     0        1
/dev/hda2         /home             ext2         defaults     0        2
/dev/hdc          /media/cdrom0     udf,iso9660  user,noauto  0        0
#/dev/fd0         /media/floppy0    auto         rw,user,nauto0        0
```

4. Shut down the computer.
5. Remove the top cover of the DA-681, and then add the hard drive into the computer.
6. Reboot the computer to finish.
7. If you would like to uninstall the SATA hard drive, simply reverse the above procedures.

# Inserting a USB Storage Device into the Computer

Since mounting USB storage devices manually can be difficult, a program named **usbmount** to mount the USB drivers automatically. **usbmount** is a small application that relies on **udev** to mount USB storage devices automatically at certain mount points. The USB storage devices will be mounted on **/media/usb0, /media/usb1**, etc.

```
MOXA:~# mount
/dev/hda1 on / type ext2 (rw,errors=remount-ro)
tmpfs on /lib/init/rw type tmpfs (rw,nosuid,mode=0755)
proc on /proc type proc (rw,noexec,nosuid,nodev)
sysfs on /sys type sysfs (rw,noexec,nosuid,nodev)
procbususb on /proc/bus/usb type usbfs (rw)
udev on /dev type tmpfs (rw,mode=0755)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
devpts on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=620)
/dev/hdb2 on /home type ext2 (rw)
nfsd on /proc/fs/nfsd type nfsd (rw)
rpc_pipefs on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
/dev/sda1 on /media/usb0 type vfat
(rw,noexec,nodev,sync,noatime,gid=25,dmask=0007,fmask=0117)
/dev/sdb1 on /media/usb1 type vfat
(rw,noexec,nodev,sync,noatime,gid=25,dmask=0007,fmask=0117)
MOXA:~#
```

**ATTENTION**

Remember to type the command # sync before you disconnect the USB storage device. If you do not issue the command, you may lose data.

**ATTENTION**

Remember to exit the /media/usb0 or /media/usb1 directory when you disconnect the USB storage device. If you stay in /media/usb0 or /media/usb1, the automatic un-mount process will fail. If that happens, type # umount /media/usb0 to un-mount the USB device manually.

# Inserting a CompactFlash Card into the Computer

The CompactFlash card is treated as a local disk drive in the DA-681-LX computer. It is identified as a block device at **/dev/hdb**. You can add one line to **/etc/fstab** to force the CompactFlash card to mount automatically at boot time.

**ATTENTION**

The DA-681 Series Embedded Computer does not support the CompactFlash hot swap function. You must remove the power source first before inserting or removing the CompactFlash card. If you do not shut down the power source, you could damage your CompactFlash card.

```
Moxa:~# mount -o remount,rw /dev/hda1 /
Moxa:~# vi /etc/fstab
# /etc/fstab: static file system information.
#
# <file system>   <mount point>     <type>   <options>         <dump>   <pass>
Proc              /proc             proc     defaults          0        0
/dev/hda1         /                 ext2     ro,defaults,errors=remount-ro 0    1
none              /tmp              tmpfs    defaults          0        1
/dev/hda2         /home             ext2     defaults          0        2
/dev/hdc          /media/cdrom0 udf,iso9660 user,noauto        0        0
#/dev/fd0         /media/floppy0    auto     rw,user,noauto    0        0
Moxa:~#
Moxa:~# umount /
Moxa:~#
```

# Checking the Linux Version

The program **uname**, which stands for "Unix Name" and is part of the Unix operating system, prints the name, version, and other details about the operating system running on the computer. Use the -**a** option to generate a response similar to the one shown below:

```
MOXA:~# uname -a
Linux DA680 2.6.18-5-686 #1 SMP Mon Dec 24 16:41:07 UTC 2007 i686 GNU/Linux
MOXA:~#
```

# APT—Installing and Removing Packages

APT is the Debian tool used to install and remove packages. Before installing a package, you need to configure the apt source file, **/etc/apt/sources.list**, which is located in the read-only partition.

1. Mount the root file system with write permission.

```
MOXA:~# mount -o remount,rw /dev/hda1 /
```

2. Next, configure the **/etc/apt/sources.list** using **vi** editor.

```
MOXA:~# vi /etc/apt/sources.list

deb http://archive.debian.org/debian etch main
deb-src http://archive.debian.org/debian etch main

deb http://archive.debian.org/debian/ etch main
deb-src http://archive.debian.org/debian/ etch main

deb http://archive.debian.org/debian-security/ etch/updates main contrib
deb-src http://archive.debian.org/debian-security/ etch/updates main contrib
```

3. Update the source list after you configure it.

```
MOXA:~# apt-get update
MOXA:~#
```

4. Once you indicate which package you want to install (**openswan**, for example), type:

```
MOXA:~# apt-get install openswan
MOXA:~#
```

5. Use one of the following commands to remove a package:
   a. For a simple package removal:

```
MOXA:~# apt-get remove openswan
MOXA:~#
```

   b. For a complete package removal:

```
MOXA:~# apt-get remove openswan --purge
MOXA:~#
```

6. If the installation is complete, remember to umount the root directory back to read-only mode.

```
MOXA:~# umount /
MOXA:~#
```

**ATTENTION**

The APT cache space /etc/cache/apt is located in tmpfs. If you need to install a huge package, link /etc/cache/apt to USB mass storage or mount it to an NFS space to generate more free space. Use df –h to check how much free space is available on tmpfs.

```
MOXA:~# df -h
Filesystem        Size  Used Avail Use% Mounted on
rootfs            790M  219M  531M  30% /
udev                 10M   44K   10M   1% /dev
/dev/hdb1            790M  219M  531M  30% /
/dev/hdb1            790M  219M  531M  30% /dev/.static/dev
tmpfs                248M     0  248M   0% /dev/shm
none                 248M   13M  236M   6% /tmp
/dev/mtdblock0       161M   25M  136M  16% /home
MOXA:~#
```

**ATTENTION**

You can free up the cache space with the command # apt-get clean

```
MOXA:~# apt-get clean
MOXA:~#
```

# WDT (Watchdog Timer)

1. **Introduction**

   The WDT works like a hardware timing device that triggers a system reboot whenever necessary. You can enable it or disable it. When users enable WDT but the application does not acknowledge it, the system will reboot. You can set the ack time from a minimum of 1 sec to a maximum of 60 seconds.

2. **How the WDT works**

   The watchdog is enabled when the system boots up. The kernel will auto ack it. The user application can also enable ack. When users do not ack, it will let the system reboot.

3. **The user IO control commands**

   The user application can program the watchdog via below IO control commands.

```
// Enable the user mode watchdog driver
#define IOCTL_SWATCHDOG_ENABLE     1
// Disable the user mode watchdog driver
#define IOCTL_SWATCHDOG_DISABLE    2
// Get the driver mode and timeout information
#define IOCTL_SWATCHDOG_GET    3
// Acknowledge the user mode watchdog driver
#define IOCTL_SWATCHDOG_ACK    4
```

   User application can programming the watchdog like this.

```
main() {
    …
  swtd_fd=open("/dev/swtd", O_RDWR)
    …
     time=4000; //4000 ms
  ioctl(swtd_fd, IOCTL_SWATCHDOG_ENABLE, &time);
    …
  while(1) {
       …
```

```
  ioctl(swtd_fd, IOCTL_SWATCHDOG_ACK, NULL);
      …
  }
  close(swtd_fd);
}
```

For convenient programming, we re-package these IO control commands in a convenient API in libswtd.c like this.

`int swtd_open(void)`

**Description**

Open the file handle to control the sWatchDog. If you want to do something you must first to this. And keep the file handle to do other.

**Input**

None

**Output**

The return value is file handle. If has some error, it will return < 0 value.

You can get error from errno().

`int swtd_enable(int fd, unsigned long time)`

**Description**

Enable application sWatchDog. And you must do ack after this process.

**Input**

int fd - the file handle, from the swtd_open() return value.

unsigned long time - The time you wish to ack sWatchDog periodically. You must ack the sWatchDog before timeout. If you do not ack, the system will be reboot automatically. The minimal time is 50 msec, the maximum time is 60 seconds. The time unit is msec.

**Output**

OK will be zero. The other has some error, to get the error code from errno().

**`int swtd_disable(int fd)`**

**Description:**

Disable the application to ack sWatchDog. And the kernel will be auto ack it. User does not to do it at periodic.

**Input:**

int fd - the file handle from swtd_open() return value.

**Output:**

OK will be zero. The other has some error, to get error code from errno.

`int swtd_get(int fd, int *mode, unsigned long *time)`

**Description:**

Get current setting values.

**mode –**

1 for user application enable sWatchDog: need to do ack.
0 for user application disable sWatchdog: does not need to do ack.
time – The time period to ack sWatchDog.

**Input:**

int fd - the file handle from swtd_open() return value.

int *mode - the function will be return the status enable or disable user application need to
do ack.

unsigned long *time – the function will return the current time period.

**Output:**
OK will be zero.

The other has some error, to get error code from errno().

`int swtd_ack(int fd)`

**Description:**
Acknowledge sWatchDog. When the user application enable sWatchDog. It need to call this function periodically with user predefined time in the application program.

**Input:**
int fd - the file handle from swtd_open() return value.

**Output:**
OK will be zero.

The other has some error, to get error code from errno().

**`int swtd_close(int fd)`**

**Description:**
Close the file handle.

**Input:**
int fd - the file handle from swtd_open() return value.

**Output:**
OK will be zero.

The other has some error, to get error code from errno().

**Special Note.**
When you "kill the application with -9" or "kill without option" or "Ctrl+c" the kernel will change to auto ack the sWatchDog.

When your application enables the sWatchDog and does not ack, your application may have a logical error, or your application has made a core dump. The kernel will not change to auto ack. This can cause a serious problem, causing your system to reboot again and again.

4. **User application example.**

**Example 1:**

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


int main(int argc, char *argv[])
{
      int fd;

      fd = swtd_open();
      if ( fd < 0 ) {
          printf("Open sWatchDog device fail !\n");
          exit(1);
      }
      swtd_enable(fd, 5000);  // enable it and set it 5 seconds
      while ( 1 ) {
          // do user application want to do
          …..
          ….
          swtd_ack(fd);
          …..
          ….
      }
      swtd_close(fd);
```

```
    exit(0);
}

/*
* The convenient watchdog API --- libswtd.c
*/
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>

// following for sWatchDog implement
#define IOCTL_SWATCHDOG_ENABLE   1
#define IOCTL_SWATCHDOG_DISABLE  2
#define IOCTL_SWATCHDOG_GET  3
#define IOCTL_SWATCHDOG_ACK  4
int  swtd_open(void)
{
 return open("/dev/swtd", O_RDWR);
}

int  swtd_enable(int fd, unsigned long time)
{
 return ioctl(fd, IOCTL_SWATCHDOG_ENABLE, &time);
}

int  swtd_disable(int fd)
{
 return ioctl(fd, IOCTL_SWATCHDOG_DISABLE, NULL);
}

int  swtd_get(int fd, int *mode, unsigned long *time)
{
 struct {
     int     mode;
     unsigned long   time;
 } nowset;
 int         ret;

 ret = ioctl(fd, IOCTL_SWATCHDOG_GET, &nowset);
 *mode = nowset.mode;
 *time = nowset.time;
 return ret;
}

int  swtd_ack(int fd)
{
 return ioctl(fd, IOCTL_SWATCHDOG_ACK, NULL);
}

int  swtd_close(int fd)
{
 return close(fd);
}

The makefile is shown below:
all:
     gcc xxxx.c libswtd.c -o xxxx

Example 2:
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/ioctl.h>
#include <sys/select.h>
#include <sys/time.h>
#include <moxadevice.h>

static void mydelay(unsigned long msec)
{
```

```
      struct timeval  time;

      time.tv_sec = msec / 1000;
      time.tv_usec = (msec % 1000) * 1000;
      select(1, NULL, NULL, NULL, &time);
}

static int   swtdfd;
static int   stopflag=0;

static void stop_swatchdog()
{
      stopflag = 1;
}

static void do_swatchdog(void)
{
      swtd_enable(swtdfd, 500);
      while ( stopflag == 0 ) {
          mydelay(250);
          swtd_ack(swtdfd);
      }
      swtd_disable(swtdfd);
 }

int  main(int argc, char *argv[])
{
      pid_t       sonpid;

      signal(SIGUSR1, stop_swatchdog);
      swtdfd = swtd_open();
      if ( swtdfd < 0 ) {
          printf("Open sWatchDog device fail !\n");
          exit(1);
      }
      if ( (sonpid=fork()) == 0 )
          do_swatchdog();
      // do user application main function
      …..
      …..
      …..
      // end user application
      kill(sonpid, SIGUSR1);
      swtd_close(swtdfd);
      exit(1);
}

/*
* The convenient watchdog API --- libswtd.c
*/
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>

// following for sWatchDog implement
#define IOCTL_SWATCHDOG_ENABLE   1
#define IOCTL_SWATCHDOG_DISABLE  2
#define IOCTL_SWATCHDOG_GET  3
#define IOCTL_SWATCHDOG_ACK  4
int  swtd_open(void)
{
 return open("/dev/swtd", O_RDWR);
}

int  swtd_enable(int fd, unsigned long time)
{
 return ioctl(fd, IOCTL_SWATCHDOG_ENABLE, &time);
}

int  swtd_disable(int fd)
{
```

```
 return ioctl(fd, IOCTL_SWATCHDOG_DISABLE, NULL);
}

int  swtd_get(int fd, int *mode, unsigned long *time)
{
 struct {
     int     mode;
     unsigned long   time;
 } nowset;
 int         ret;

 ret = ioctl(fd, IOCTL_SWATCHDOG_GET, &nowset);
 *mode = nowset.mode;
 *time = nowset.time;
 return ret;
}

int  swtd_ack(int fd)
{
 return ioctl(fd, IOCTL_SWATCHDOG_ACK, NULL);
}

int  swtd_close(int fd)
{
 return close(fd);
}

The makefile is shown below:
all:
     gcc xxxx.c libswtd.c -o xxxx
```

# 3

# Managing Communications

The DA-681-LX ready-to-run embedded computer is a network-centric platform designed to serve as a front-end for data acquisition and industrial control applications. This chapter describes how to configure the various communication functions supported by the Linux operating system.

The following topics are covered in this chapter:

# Changing the Network Settings

The DA-681-LX computer has6 Ethernet ports named LAN1 to LAN6. The LAN Port Expansion Module supports an additional four 10/100 Mbps Ethernet ports named LAN5 to LAN8. The default IP addresses and netmasks of the network interfaces are as follows:

| | Default IP Address | Netmask |
|---|---|---|
| LAN 1 | 192.168.3.127 | 255.255.255.0 |
| LAN 2 | 192.168.4.127 | 255.255.255.0 |
| LAN 3 | 192.168.5.127 | 255.255.255.0 |
| LAN 4 | 192.168.6.127 | 255.255.255.0 |
| LAN 5 | 192.168.5.127 | 255.255.255.0 |
| LAN 6 | 192.168.6.127 | 255.255.255.0 |

These network settings can be modified by changing the **interfaces** configuration file, or they can be adjusted temporarily with the **ifconfig** command.

# Changing the "interfaces" Configuration File

1. Type **cd /etc/network** to change directory.

```
MOXA:~# cd /etc/network
```

2. Type **vi interfaces** to edit the network configuration file with **vi** editor. You can configure the DA-681-LX's Ethernet ports for static or dynamic (DHCP) IP addresses.

```
MOXA:/etc/network# vi interfaces
```

## Static IP Address

As shown in the example shown below, the default static IP addresses can be modified.

```
# The loopback network interface
auto lo eth0 eth1 eth2 eth3 eth4 eth5
iface lo inet loopback

# The primary network interface
allow-hotplug eth0
iface eth0 inet static
       address 192.168.3.127
       netmask 255.255.255.0
       broadcast 192.168.3.255

allow-hotplug eth1
iface eth1 inet static
       address 192.168.4.127
       netmask 255.255.255.0
       broadcast 192.168.4.255

allow-hotplug eth2
iface eth2 inet static
       address 192.168.5.127
       netmask 255.255.255.0
       broadcast 192.168.5.255
```

### Dynamic IP Address using DHCP

To configure one or both LAN ports to request an IP address dynamically, replace **static** with **dhcp** and then delete the rest of the lines.

```
# The primary network interface
allow-hotplug eth0
iface eth0 inet dhcp
```

After modifying the boot settings of the LAN interface, issue the following command to activate the LAN settings immediately.

**# /etc/init.d/networking restart**

```
MOXA:~# /etc/init.d/networking restart
```

## Adjusting IP Addresses with "ifconfig"

IP settings can be adjusted during run-time, but the new settings will not be saved to the flash ROM without modifying the file **/etc/network/interfaces**. For example, type the command **# ifconfig eth1 192.168.1.1** to change the IP address of LAN1 to 192.168.1.1.

```
MOXA:~# ifconfig eth1 192.168.1.1
MOXA:~#
```

# Telnet Server

In addition to supporting Telnet client/server, the DA-681-LX also supports SSH and sftp client/server. To enable or disable the Telnet server, you need to edit the file **/etc/inetd.conf**.

1. Mount the root file system with write permission.

```
MOXA:~# mount -o remount,rw /dev/hda1 /
```

2. Type **# cd /etc** to change the directory.

```
MOXA:~# cd /etc
```

3. Type **# vi inetd.conf** to edit the configuration file.

```
MOXA:/etc# vi inetd.conf
```

## Enabling the Telnet Server

The following example shows the default content of the file **/etc/inetd.conf**. The default is to "enable the Telnet/ftp server:"

```
discard dgram udp wait root /bin/discard
discard stream tcp nowait root /bin/discard
telnet stream tcp nowait root /bin/telnetd
```

## Disabling the Telnet Server

Disable the daemon by typing "**#**" in front of the first character of the row to comment out the line. For example, to disable the Telnet server, use the following commands:

```
discard dgram udp wait root /bin/discard
discard stream tcp nowait root /bin/discard
telnet stream tcp nowait root /bin/telnetd
```

After you finish writing or modifying the code, remember to execute "umount /" to change the root directory back to Read-only mode.

```
MOXA:~# umount /
```

# FTP Server

Refer to the following commands to enable or disable the FTP Server service.

## Enabling the FTP Server

Use the following command to enable the FTP server.

```
Moxa:~# /etc/init.d/proftpd start
```

```
Starting ftp server: proftpd.
```

Use the following command to confirm if the FTP has been started.

```
Moxa:~#ps aux|grep proftp
```

If profptd string has appeared, the FTP server has been started.

## Disabling the FTP Server

Use the following command to disable the FTP server.

```
Moxa:~# /etc/init.d/proftpd stop
```

```
Stopping ftp server: proftpd.
```

To confirm if FTP has been disabled, use the following command.

```
Moxa:~# ps auxgrep proftp
```

If profptd string has not appeared, the FTP server has been disabled.

# DNS Client

The DA-681-LX supports DNS client (but not DNS server). To set up DNS client, you need to edit three configuration files: **/etc/hostname, /etc/resolv.conf,** and **/etc/nsswitch.conf**.

# /etc/hostname

1. Mount the root file system with write permission.

```
MOXA:~# mount -o remount,rw /dev/hda1 /
```

2. Edit **/etc/hostname**:

```
MOXA:~# vi /etc/hostname
MOXA
```

3. After you finish writing or modifying the code, remember to execute "umount /" to change the root directory back to Read-only mode.

```
MOXA:~# umount /
```

4. Re-configure the hostname.

```
MOXA:~# /etc/init.d/hostname.sh start
```

5. Check the new hostname.

```
MOXA:~# hostname
```

# /etc/resolv.conf

This is the most important file that you need to edit when using DNS. For example, before you using **# ntpdate time.nist.goc** to update the system time, you will need to add the DNS server address to the file. Ask your network administrator which DNS server address you should use. The DNS server's IP address is specified with the **nameserver** command. For example, add the following line to /etc/resolv.conf (assuming the DNS server's IP address is 168.95.1.1):

**nameserver 168.95.1.1**

```
MOXA:/etc# cat resolv.conf
#
# resolv.conf  This file is the resolver configuration file
# See resolver(5).
#
#nameserver 192.168.1.16
nameserver 168.95.1.1
nameserver 140.115.1.31
nameserver 140.115.236.10
MOXA:/etc#
```

# /etc/nsswitch.conf

This file defines the sequence of files, **/etc/hosts** or **/etc/resolv.conf**, to be read to resolve the IP address.

The **hosts** line in **/etc/nsswitch.conf** means use **/etc/host** first and DNS service to resolve the address.

```
# /etc/nsswitch.conf
#
# Example configuration of GNU Name Service Switch functionality.
# If you have the `glibc-doc-reference' and `info' packages installed, try:
```

```
# `info libc "Name Service Switch"' for information about this file.

passwd:         compat
group:          compat
shadow:         compat

hosts:          files dns
networks:       files

protocols:      db files
services:       db files
ethers:         db files
rpc:            db files

netgroup:       nis
```

# Apache Web Server

## Default Homepage

The Apache web server's main configuration file is **/etc/apache2/sites-available/default**, with the default homepage located at **/var/www/apache2-default/index.html**.

Save your own homepage to the following directory:

**/var/www/apache2-default**

Save your CGI page to the following directory:

**/var/www/apache2-default/cgi-bin/**

Add a main page file under cgi-bin directory. For example, you may add a file called index.cgi at /var/www/apache2-default/cgi-bin/.

```
#!/bin/sh
# /var/www/apache2-default/cgi-bin/index.cgi
# disable filename globbing
set -f
echo "Content-type: text/plain; charset=iso-8859-1"
echo
echo CGI/1.0 test script report:
echo
echo argc is $#. argv is "$*".
echo
echo SERVER_SOFTWARE = $SERVER_SOFTWARE
echo SERVER_NAME = $SERVER_NAME
echo GATEWAY_INTERFACE = $GATEWAY_INTERFACE
echo SERVER_PROTOCOL = $SERVER_PROTOCOL
echo SERVER_PORT = $SERVER_PORT
echo REQUEST_METHOD = $REQUEST_METHOD
echo HTTP_ACCEPT = "$HTTP_ACCEPT"
echo PATH_INFO = "$PATH_INFO"
echo PATH_TRANSLATED = "$PATH_TRANSLATED"
echo SCRIPT_NAME = "$SCRIPT_NAME"
echo QUERY_STRING = "$QUERY_STRING"
echo REMOTE_HOST = $REMOTE_HOST
```

```
echo REMOTE_ADDR = $REMOTE_ADDR
echo REMOTE_USER = $REMOTE_USER
echo AUTH_TYPE = $AUTH_TYPE
echo CONTENT_TYPE = $CONTENT_TYPE
echo CONTENT_LENGTH = $CONTENT_LENGTH
```

Before you modify the homepage, use a browser (such as Microsoft Internet Explore or Mozilla Firefox) from your PC to test if the Apache web server is working. Type the LAN1 IP address in the browser's address box to open the homepage. For example, if the default IP address 192.168.3.127 is still active, type:

**http://192.168.3.127/**

To test the default CGI page, type:

**http://192.168.3.127/cgi-bin/index.cgi**

**ATTENTION**

When you develop your own CGI application, make sure your CGI file is executable.

# Saving Web Pages to a USB Storage Device

Some applications may have web pages that take up a lot of memory space. This section describes how to save web pages to the USB mass storage device, and then configure the Apache web server's DocumentRoot to open these pages. The files used in this example can be downloaded from Moxa's website.

1. Prepare the web pages and then save the pages to the USB storage device. Click on the following link to download the web page test suite: **http://www.w3.org/MarkUp/Test/HTML401.zip**.
2. Uncompress the zip file to your desktop PC, and then use FTP to transfer it to the DA-681-LX's **/media/usb0** directory.
3. Mount the root file system with write permission.

```
MOXA:~# mount -o remount,rw /dev/hda1 /
```

4. Type **# vi /etc/apache2/sites-available/default** to edit the configuration file.

```
MOXA:/etc# vi /etc/apache2/sites-available/default
```

5. Change the DocumentRoot directory to the USB storage directory **/media/usb0/www**.

```
...
<VirtualHost *:80>
...
...
       DocumentRoot /media/usb0/www
       <Directory />
              Options FollowSymLinks
              AllowOverride None
       </Directory>
...
...
       ScriptAlias /cgi-bin/ /media/usb0/www/cgi-bin/
       <Directory "/media/usb0/www/cgi-bin/">
              AllowOverride None
              Options ExecCGI -MultiViews +SymLinksIfOwnerMatch
              Order allow,deny
```

```
            Allow from all
      </Directory>
...
</VirtualHost>
...
<VirtualHost *:443>
...
...
      DocumentRoot /media/usb0/www
      <Directory />
            Options FollowSymLinks
            AllowOverride None
      </Directory>
...
...
      ScriptAlias /cgi-bin/ /media/usb0/www/cgi-bin/
      <Directory "/media/usb0/wwwz/cgi-bin/">
            AllowOverride None
            Options ExecCGI -MultiViews +SymLinksIfOwnerMatch
            Order allow,deny
            Allow from all
      </Directory>
...
</VirtualHost>
```

6. Use the following commands to restart the Apache web server:

   **#cd /etc/init.d**
   **#./apache2 restart**

7. Open your browser and connect to the DA-681-LX by typing the current LAN1 IP address in the browser's address box.
8. After finishing modification or writing, remember to execute "umount /" to change the root directory back to Read-only mode.

```
MOXA:~# umount /
```

9. Re-start the apache server.

```
MOXA:~# /etc/init.d/apache2 restart
```

**ATTENTION**

Visit the Apache website at http://httpd.apache.org/docs/ for more information about setting up Apache servers.

# 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 what to do with a matching packet. A rule (such as a jump to a user-defined chain in the same table) is called a **target**.

The DA-681-LX supports three types of IPTABLES: Filter tables, NAT tables, and Mangle tables.

## Filter Table—includes three chains:

INPUT chain
OUTPUT chain
FORWARD chain

## 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 the 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.

## Mangle Table—includes two chains

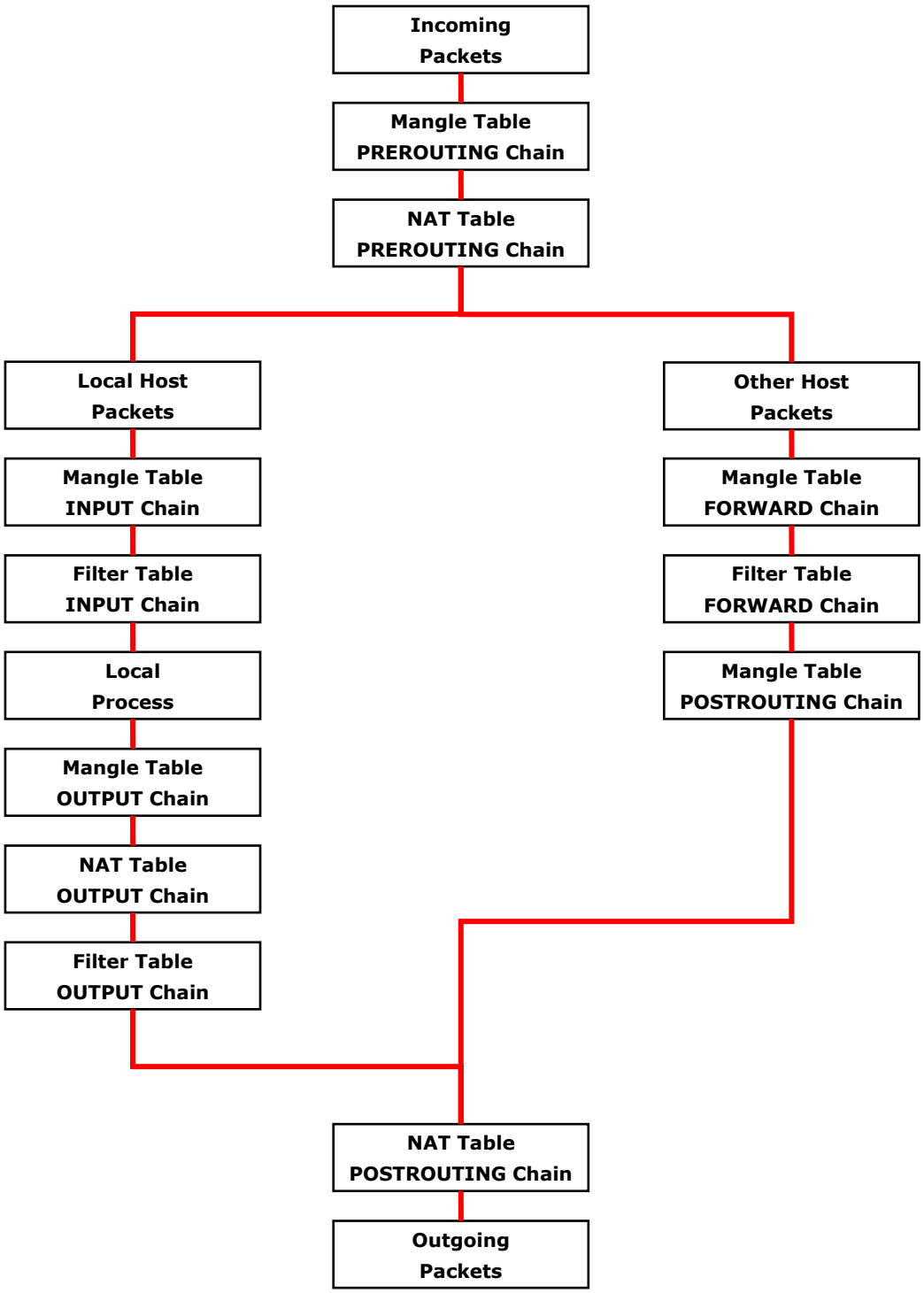
PREROUTING chain—pre-processes packets before the routing process.

OUTPUT chain—processes packets after the routing process.

Mangle tables can have one of three extensions—TTL, MARK, TOS.

# IPTABLES Hierarchy

The following figure shows the IPTABLES hierarchy.



# IPTABLES Modules

The DA-681-LX supports the following sub-modules. Be sure to use the module that matches your application.

| | | | |
|---|---|---|---|
| arptable_filter.ko | arp_tables.ko | arpt_mangle.ko | ip_conntrack_amanda.ko |
| ip_conntrack_ftp.ko | ip_conntrack_h323.ko | ip_conntrack_irc.ko | ip_conntrack.ko |
| ip_conntrack_netbios_ns.ko | ip_conntrack_netlink.ko | ip_conntrack_pptp.ko | ip_conntrack_proto_sctp.ko |
| ip_conntrack_sip.ko | ip_conntrack_tftp.ko | ip_nat_amanda.ko | ip_nat_ftp.ko |
| ip_nat_h323.ko | ip_nat_irc.ko | ip_nat.ko | ip_nat_pptp.ko |
| ip_nat_sip.ko | ip_nat_snmp_basic.ko | ip_nat_tftp.ko | ip_queue.ko |
| iptable_filter.ko | iptable_mangle.ko | iptable_nat.ko | iptable_raw.ko |
| ip_tables.ko | ipt_addrtype.ko | ipt_ah.ko | ipt_CLUSTERIP.ko |
| ipt_dscp.ko | ipt_DSCP.ko | ipt_ecn.ko | ipt_ECN.ko |
| ipt_hashlimit.ko | ipt_iprange.ko | ipt_LOG.ko | ipt_MASQUERADE.ko |
| ipt_NETMAP.ko | ipt_owner.ko | ipt_recent.ko | ipt_REDIRECT.ko |
| ipt_REJECT.ko | ipt_SAME.ko | ipt_TCPMSS.ko | ipt_tos.ko |
| ipt_TOS.ko | ipt_ttl.ko | ipt_TTL.ko | ipt_ULOG.ko |

The basic syntax to enable and load an IPTABLES module is as follows:

**# lsmod**
**# modprobe ip_tables**
**# modprobe iptable_filter**

Use **lsmod** to check if the **ip_tables** module has already been loaded in the DA-681-LX. Use **modprobe** to insert and enable the module.

Use the following command to load the modules (**iptable_filter, iptable_mangle, iptable_nat**):

**# modprobe iptable_filter**

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

ATTENTION
IPTABLES plays the role of packet filtering or NAT. Be careful when setting up the IPTABLES rules. If the rules are not correct, remote hosts that connect via a LAN or PPP may be denied. We recommend using the VGA console to set up the 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 DA-681-I-LX.
OUTPUT: For locally-generated packets.
FORWARD: For packets routed out through the DA-681-I-LX.
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 via which a packet is going to be received.
-o: Name of an interface via which a packet is going to 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 the 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 DA-681-I-LX'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 DA-681-I-LX'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**

**ATTENTION**

In Example 8, remember to issue the command # modprobe ipt_mac first to load the module ipt_mac.

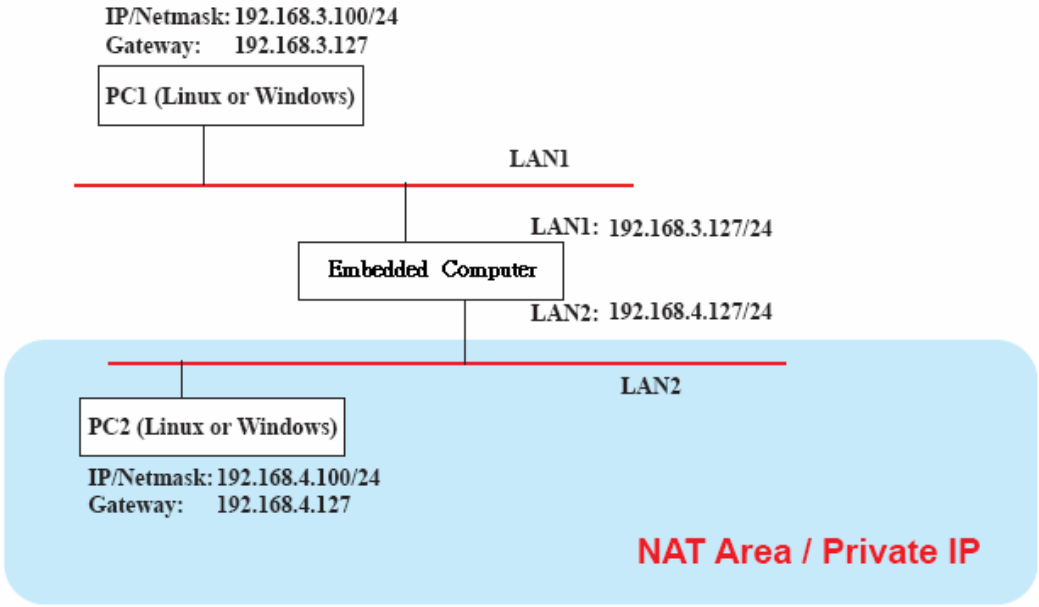
# NAT (Network Address Translation)

The NAT (Network Address Translation) protocol translates IP addresses used on one network into IP addresses used on a connecting network. One network is designated the inside network and the other is the outside network. Typically, the DA-681-LX connects several devices on a network and maps local inside network addresses to one or more global outside IP addresses, and un-maps the global IP addresses on incoming packets back into local IP addresses.

ATTENTION
Click on the following links for more information about NAT.
- http://www.netfilter.org/documentation/HOWTO//packet-filtering-HOWTO.html

# NAT Example

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




**#ehco 1 > /proc/sys/net/ipv4/ip_forward**
**#modprobe ipt_MASQUERADE**
**#iptables –t nat –A POSTROUTING –o eth0 –j MASQUERADE**

# Enabling NAT at Bootup

In most real world situations, you will want to use a simple shell script to enable NAT when the DA-681-LX boots up. The following script is an example.

```
#!/bin/bash
# If you put this shell script in the /home/nat.sh
# Remember to chmod 744 /home/nat.sh
# Edit the rc.local file to make this shell startup automatically.
# vi /etc/rc.local
# Add a line in the end of rc.local /home/nat.sh

EXIF= "eth0"  #This is an external interface for setting up a valid IP address.
EXNET= "192.168.4.0/24"  #This is an internal network address.

# Step 1. Insert modules.

# Here 2> /dev/null means the standard error messages will be dump to null device.

modprobe ip_tables  2> /dev/null
modprobe ip_nat_ftp  2> /dev/null
modprobe ip_nat_irc  2> /dev/null
modprobe ip_conntrack  2> /dev/null
modprobe ip_conntrack_ftp  2> /dev/null
modprobe ip_conntrack_irc  2> /dev/null

# Step 2. Define variables, enable routing and erase default rules.

PATH=/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin
export PATH
echo "1" > /proc/sys/net/ipv4/ip_forward
```

**/sbin/iptables -F**
**/sbin/iptables -X**
**/sbin/iptables -Z**
**/sbin/iptables -F -t nat**
**/sbin/iptables -X -t nat**
**/sbin/iptables -Z -t nat**
**/sbin/iptables -P INPUT ACCEPT**
**/sbin/iptables -P OUTPUT ACCEPT**
**/sbin/iptables -P FORWARD ACCEPT**
**/sbin/iptables -t nat -P PREROUTING ACCEPT**
**/sbin/iptables -t nat -P POSTROUTING ACCEPT**
**/sbin/iptables -t nat -P OUTPUT ACCEPT**

**# Step 3. Enable IP masquerade.**

# PPP (Point to Point Protocol)

PPP (Point to Point Protocol) is used to run IP (Internet Protocol) and other network protocols over a serial link. PPP can be used for direct serial connections (using a null-modem cable) over a Telnet link, and links established using a modem over a telephone line.

Modem/PPP access is almost identical to connecting directly to a network through the DA-681-LX's Ethernet port. Since PPP is a peer-to-peer system, the DA-681-LX can also use PPP to link two networks (or a local network to the Internet) to create a Wide Area Network (WAN).

**ATTENTION**

Click on the following links for more information about PPP.

http://tldp.org/HOWTO/PPP-HOWTO/index.html
http://axion.physics.ubc.ca/ppp-linux.html

## Connecting to a PPP Server over a Simple Dial-up Connection

The following command is used to connect to a PPP server by modem. Use this command for old ppp servers that prompt for a login name (replace "username" with the correct name) and password (replace "password" with the correct password). Note that "debug crtscts" and "defaultroute 192.1.1.17" are optional.

**#pppd connect 'chat -v "" ATDT5551212 CONNECT ""' ogin: username word: password' /dev/ttyM0 115200 debug crtscts modem defaultroute 192.1.1.17**

If the PPP server does not prompt for the username and password, the command should be entered as follows. Replace "username" with the correct username and replace "password" with the correct password.

**#pppd connect 'chat -v "" ATDT5551212 CONNECT ""' user username password password /dev/ttyM0 115200 crtscts modem**

The pppd options are described below:

| | |
|---|---|
| **connect 'chat etc...'** | This option gives the command to contact the PPP server. The **chat** program is used to dial a remote computer. The entire command is enclosed in single quotes because pppd expects a one-word argument for the **connect** option. The options for **chat** are given below: |
| **-v** | verbose mode; log what we do to syslog |
| " " | Double quotes—don't wait for a prompt, but instead do ... (note that you must include a space after the second quotation mark) |
| **ATDT5551212** | Dial the modem, and then ... |
| **CONNECT** | Wait for an answer. |

" " Send a return (null text followed by the usual return)

**ogin: username word: password**
Log in with username and password.

Refer to the chat man page, chat.8, for more information about the **chat** utility.

| | |
|---|---|
| **/dev/** | Specify the callout serial port. |
| **115200** | The baud rate. |
| **debug** | Log status in syslog. |
| **crtscts** | Use hardware flow control between computer and modem (at 115200 this is a must). |
| **modem** | Indicates that this is a modem device; pppd will hang up the phone before and after making the call. |
| **defaultroute** | Once the PPP link is established, make it the default route; if you have a PPP link to the Internet, this is probably what you want. |
| **192.1.1.17** | This is a degenerate case of a general option of the form x.x.x.x:y.y.y.y. Here x.x.x.x is the local IP address and y.y.y.y is the IP address of the remote end of the PPP connection. If this option is not specified, or if just one side is specified, then x.x.x.x defaults to the IP address associated with the local machine's hostname (located in /**etc/hosts**), and y.y.y.y is determined by the remote machine. |

# Connecting to a PPP Server over a Hard-wired Link

If a username and password are not required, use the following command (note that **noipdefault** is optional):

**#pppd connect 'chat –v" " " " ' noipdefault /dev/ttyM0 19200 crtscts**

If a username and password is required, use the following command (note that **noipdefault** is optional, and root is both the username and password):

**#pppd connect 'chat –v" " " " ' user root password root noipdefault /dev/ttyM0 19200 crtscts**

# Checking the Connection

Once you have set up a PPP connection, there are some steps you can take to test the connection. First, type:

**#/sbin/ifconfig**

Depending on your distribution, the command might be located elsewhere. After executing the command, you should be able to see all of the network interfaces that are UP.

**ppp0** should be one of them, and you should recognize the first IP address as your own and the **P-t-P address** (point-to-point address, the address of your server). The output is similar to the following:

```
lo        Link encap Local Loopback
          inet addr 127.0.0.1   Bcast 127.255.255.255 Mask 255.0.0.0
          UP LOOPBACK RUNNING   MTU 2000   Metric 1
          RX packets 0 errors 0 dropped 0 overrun 0

ppp0 Link encap Point-to-Point Protocol
          inet addr 192.76.32.3   P-t-P 129.67.1.165 Mask 255.255.255.0
          UP POINTOPOINT RUNNING   MTU 1500   Metric 1
          RX packets 33 errors 0 dropped 0 overrun 0
          TX packets 42 errors 0 dropped 0 overrun 0
```

Now, type:

**#ping z.z.z.z**

where z.z.z.z is the address of your name server. The output is similar to the following:

```
MOXA:~# ping 129.67.1.165
PING 129.67.1.165 (129.67.1.165): 56 data bytes
64 bytes from 129.67.1.165: icmp_seq=0 ttl=225 time=268 ms
64 bytes from 129.67.1.165: icmp_seq=1 ttl=225 time=247 ms
64 bytes from 129.67.1.165: icmp_seq=2 ttl=225 time=266 ms
^C
--- 129.67.1.165 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 247/260/268 ms
MOXA:~#
```

Try typing:

**#netstat -nr**

This should show three routes similar to the following:

```
Kernel routing table
Destination    Gateway     Genmask          Flags    Metric   Ref  Use
iface
129.67.1.165  0.0.0.0      255.255.255.255   UH       0        0    6
ppp0
127.0.0.0      0.0.0.0     255.0.0.0         U        0        0    0 lo
0.0.0.0        129.67.1.165    0.0.0.0           UG       0        0    6298
ppp0
```

If your output looks similar but does not have the "destination 0.0.0.0" line (which refers to the default route used for connections), you may have run pppd without the **defaultroute** option. At this point, you can try using Telnet, ftp, or finger, bearing in mind that you will have to use numeric IP addresses unless you have configured **/etc/resolv.conf** correctly.

# Setting up a Machine for Incoming PPP Connections

**Method 1: pppd dial-in with pppd commands**

This first example applies to using a modem, and requiring authorization with a username and password.

**#pppd /dev/ttyM0 115200 crtscts modem 192.168.16.1:192.168.16.2 login auth**

You should also add the following line to the file **/etc/ppp/pap-secrets**:

***  *  ""  ***

The first star (*) lets everyone login. The second star (*) lets every host connect. The pair of double quotation marks ("") indicates that the file **/etc/passwd** can be used to check the password. The last star (*) is to let any IP connect.

The following example does not check the username and password:

**# pppd/dev/ttyM0 115200 crtscts modem 192.168.16.1:192.168.16.2**

**Method 2: pppd dial-in with pppd script**

Configure a dial-in script **/etc/ppp/peer/dialin**

```
# You usually need this if there is no PAP authentication
noauth
#auth
#login

# The chat script (be sure to edit that file, too!)
```

```
init "/usr/sbin/chat -v -f /etc/ppp/ppp-ttyM0.chat"

# Set up routing to go through this PPP link
defaultroute

# Default modem (you better replace this with /dev/ttySx!)
/dev/ttyM0

# Speed
115200

# Keep modem up even if connection fails
persist
crtscts
modem
192.168.16.1:192.168.16.2
debug
-detach
```

Configure the chat script **/etc/ppp/ppp-ttyM0.chat**

```
SAY    'Auto Answer ON\n'
''     ATS0=1
```

Start the **pppd** dial-in service.

```
# pppd call dialin
```

**ATTENTION**

If you hope to have auto dial-in service, you can respawn the dial-in service in /etc/inittab.

```
MOXA:~# mount –o remount,rw /dev/hda1 /
MOXA:~# echo “p0:2345:respawn:pppd call dialin” >> /etc/inittab
MOXA:~# umount /
```

# PPPoE

The following procedure is for setting up PPPoE:

1. Connect the DA-681-LX’s LAN port to an ADSL modem with a cross-over cable, HUB, or switch.
2. Log in to the DA-681-LX as the root user.
3. Edit the file **/etc/ppp/chap-secrets** and add the following:
   **“username@hinet.net” * “password” ***

```
# Secrets for authentication using CHAP
# client       server  secret              IP addresses

# PPPOE example, if you want to use it, you need to unmark it and modify it
"username@hinet.net"   *      "password"     *
```

**username@hinet.net** is the username obtained from the ISP to log in to the ISP account. **password** is the corresponding password for the account.

4. Edit the file **/etc/ppp/pap-secrets** and add the following:
   **"username@hinet.net" * "password" ***

```
# ATTENTION: The definitions here can allow users to login without a
# password if you don't use the login option of pppd! The mgetty Debian
# package already provides this option; make sure you don't change that.

# INBOUND connections

# Every regular user can use PPP and has to use passwords from /etc/passwd
*      hostname       ""     *
"username@hinet.net"    *      "password"     *

# UserIDs that cannot use PPP at all. Check your /etc/passwd and add any
# other accounts that should not be able to use pppd!
guest   hostname       "*"    -
master  hostname       "*"    -
root    hostname       "*"    -
support hostname       "*"    -
stats   hostname       "*"    -

# OUTBOUND connections
```

**username@hinet.net** is the username obtained from the ISP to log in to the ISP account. **password** is the corresponding password for the account.

5. Edit the file **/etc/ppp/options** and add the following line:
   **plugin rp-pppoe**

```
# received.  Note: it is not advisable to use this option with the persist
# option without the demand option.  If the active-filter option is given,
# data packets which are rejected by the specified activity filter also
# count as the link being idle.
#idle <n>

# Specifies how many seconds to wait before re-initiating the link after
# it terminates.  This option only has any effect if the persist or demand
# option is used.  The holdoff period is not applied if the link was
# terminated because it was idle.
#holdoff <n>

# Wait for up n milliseconds after the connect script finishes for a valid
# PPP packet from the peer.  At the end of this time, or when a valid PPP
# packet is received from the peer, pppd will commence negotiation by
# sending its first LCP packet.  The default value is 1000 (1 second).
# This wait period only applies if the connect or pty option is used.
#connect-delay <n>

# Load the pppoe plugin
plugin rp-pppoe.so

# ---<End of File>---
```

6. If you use LAN1 to connect to the ADSL modem, add file **/etc/ppp/options.eth0**. If you use LAN2 to connect to the ADSL modem, then add **/etc/ppp/options.eth1**, etc.

```
name username@hinet.net
mtu 1492
mru 1492
defaultroute
noipdefault
~
~
"/etc/ppp/options.eth0" 5 lines, 67 characters
```

Type your username (the one you set in the **/etc/ppp/pap-secrets and /etc/ppp/chap-secrets** files) after the **name** option. You may add other options as desired.

7. Set up DNS.

If you are using DNS servers supplied by your ISP, edit the file **/etc/resolv.conf** by adding the following lines of code:

**nameserver ip_addr_of_first_dns_server**
**nameserver ip_addr_of_second_dns_server**

For example:

**nameserver 168.95.1.1**
**nameserver 139.175.10.20**

```
MOXA:/etc# cat resolv.conf
#
# resolv.conf  This file is the resolver configuration file
# See resolver(5).
#
#nameserver 192.168.1.16
nameserver 168.95.1.1
nameserver 139.175.10.20
nameserver 140.115.1.31
nameserver 140.115.236.10
MOXA:/etc#
```

8. Use the following command to create a **pppoe** connection:
   **#pppd eth0**

The ADSL modem is connected to the **LAN1** port, which is named **eth0**. If the ADSL modem is connected to **LAN2**, use **eth1**, etc.

9. Type **#ifconfig ppp0** to check if the connection is OK. If the connection is OK, you should see the IP address of ppp0. Use **#ping** to test the IP address.

```
ppp0 Link encap Point-to-Point Protocol
         inet addr 192.76.32.3   P-t-P 129.67.1.165 Mask 255.255.255.0
         UP POINTOPOINT RUNNING   MTU 1500   Metric 1
         RX packets 33 errors 0 dropped 0 overrun 0
         TX packets 42 errors 0 dropped 0 overrun 0
```

10. If you want to disconnect it, use the kill command to kill the **pppd** process.

# NFS (Network File System) Client

The Network File System (NFS) is used to mount a disk partition on a remote machine (as if it were on a local hard drive), allowing fast, seamless sharing of files across a network. NFS allows users to develop applications

for the DA-681-LX without worrying about the amount of disk space that will be available. The DA-681-LX supports only NFS client protocol.

ATTENTION
Click on the following links for more information about NFS.

- http://www.tldp.org/HOWTO/NFS-HOWTO/index.html
- http://nfs.sourceforge.net/nfs-howto/client.html

The following procedures illustrate how to mount a remote NFS Server.

1. Scan the NFS Server's shared directory.
   **#showmount –e HOST**
   showmount: Show the mount information of an NFS Server
   -e: Show the NFS Server's export list.
   HOST: IP address or DNS address
2. Establish a mount point on the NFS Client site.
   **#mkdir –p /home/nfs/public**
3. Mount the remote directory to a local directory.
   **#mount –t nfs 192.168.3.100/home/public /home/nfs/public**
   This is where 192.168.3.100 is the example IP address of the NFS server.

# SNMP (Simple Network Management Protocol)

The DA-681-LX comes with the SNMP V1 (Simple Network Management Protocol) agent software pre-installed. It supports RFC1317 **RS-232 like group** and **RFC 1213 MIB-II**. The following example shows an SNMP agent responding to a query from the SNMP browser on the host site:

```
***** SNMP QUERY STARTED *****
[root@jaredRH90 root]# snmpwalk -v 1 -c public 192.168.30.128|more
SNMPv2-MIB::sysDescr.0 = STRING: Linux Moxa 2.6.18-5-686 #1 SMP Mon Dec 24 16:41
:07 UTC 2007 i686
SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-SMI::enterprises.8691.12.680
SNMPv2-MIB::sysUpTime.0 = Timeticks: (134544) 0:22:25.44
SNMPv2-MIB::sysContact.0 = STRING: "Moxa Inc."
SNMPv2-MIB::sysName.0 = STRING: Moxa
SNMPv2-MIB::sysLocation.0 = STRING: "Fl.8, No.6, Alley 6, Lane 235, Pao-Chiao Rd
. Shing Tien City, Taipei, Taiwan, R.O.C."
SNMPv2-MIB::sysORLastChange.0 = Timeticks: (12) 0:00:00.12
SNMPv2-MIB::sysORID.1 = OID: IF-MIB::ifMIB
SNMPv2-MIB::sysORID.2 = OID: SNMPv2-MIB::snmpMIB
SNMPv2-MIB::sysORID.3 = OID: TCP-MIB::tcpMIB
SNMPv2-MIB::sysORID.4 = OID: IP-MIB::ip
SNMPv2-MIB::sysORID.5 = OID: UDP-MIB::udpMIB
SNMPv2-MIB::sysORID.6 = OID: SNMP-VIEW-BASED-ACM-MIB::vacmBasicGroup
SNMPv2-MIB::sysORID.7 = OID: SNMP-FRAMEWORK-MIB::snmpFrameworkMIBCompliance
SNMPv2-MIB::sysORID.8 = OID: SNMP-MPD-MIB::snmpMPDCompliance
SNMPv2-MIB::sysORID.9 = OID: SNMP-USER-BASED-SM-MIB::usmMIBCompliance
SNMPv2-MIB::sysORDescr.1 = STRING: The MIB module to describe generic objects fo
r network interface sub-layers
SNMPv2-MIB::sysORDescr.2 = STRING: The MIB module for SNMPv2 entities
SNMPv2-MIB::sysORDescr.3 = STRING: The MIB module for managing TCP implementatio
...
SNMPv2-MIB::snmpOutBadValues.0 = Counter32: 0
SNMPv2-MIB::snmpOutGenErrs.0 = Counter32: 0
```

```
SNMPv2-MIB::snmpOutGetRequests.0 = Counter32: 0
SNMPv2-MIB::snmpOutGetNexts.0 = Counter32: 0
SNMPv2-MIB::snmpOutSetRequests.0 = Counter32: 0
SNMPv2-MIB::snmpOutGetResponses.0 = Counter32: 540
SNMPv2-MIB::snmpOutTraps.0 = Counter32: 0
SNMPv2-MIB::snmpEnableAuthenTraps.0 = INTEGER: disabled(2)
SNMPv2-MIB::snmpSilentDrops.0 = Counter32: 0
SNMPv2-MIB::snmpProxyDrops.0 = Counter32: 0
[root@jaredRH90 root]#
***** SNMP QUERY FINISHED *****
```

ATTENTION
Click on the following links for more information about RFC1317 RS-232 like group and RFC 1213 MIB-II.

- http://www.tldp.org/HOWTO/NFS-HOWTO/index.html
- http://nfs.sourceforge.net/nfs-howto/client.html

# OpenVPN

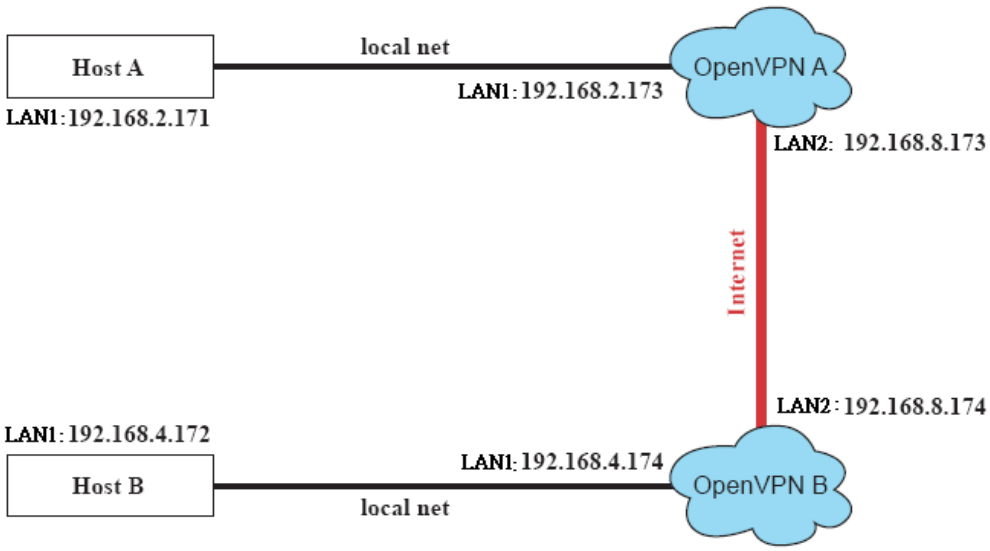
OpenVPN provides two types of tunnels for users to implement VPNS: **Routed IP Tunnels** and **Bridged Ethernet Tunnels**.

An Ethernet bridge is used to connect different Ethernet networks together. The Ethernets are bundled into one bigger, "logical" Ethernet. Each Ethernet corresponds to one physical interface (or port) that is connected to the bridge.

On each OpenVPN machine, you should carry out configurations in the **/etc/openvpn** directory, where script files and key files reside. Once established, all operations will be performed in that directory.

## Ethernet Bridging for Private Networks on Different Subnets

1. Set up four machines, as shown in the following diagram.




Host A represents the machine that belongs to OpenVPN A, and Host B represents the machine that belongs to OpenVPN B. The two remote subnets are configured for a different range of IP addresses. When this configuration is moved to a public network, the external interfaces of the OpenVPN machines should be configured for static IPs, or connected to another device (such as a firewall or DSL box) first.

2. Generate a preset shared key by typing the command:
   **# openvpn --genkey --secret secrouter.key**
3. Copy the file that is generated to the OpenVPN machine:
   **# scp /etc/openvpn/secrouter.key 192.168.8.174:/etc/openvpn**

**ATTENTION**

A preshared key is located at /etc/openvpn/secrouter.key. You can use it for testing purposes. We suggest creating a new key for non-testing purpose.

4. On machine OpenVPN A, modify the remote address in the configuration file **/etc/openvpn/tap0-br.conf**.

```
# point to the peer
remote 192.168.8.174
dev tap0
secret /etc/openvpn/secrouter.key
cipher DES-EDE3-CBC
auth MD5
tun-mtu 1500
tun-mtu-extra 64
ping 40
up /etc/openvpn/tap0-br.sh
#comp-lzo
```

5. Next, modify the routing table in the **/etc/openvpn/tap0-br.sh script** file.

```
#--------------------------Start--------------------------
#!/bin/sh
# value after "-net" is the subnet behind the remote peer
route add -net 192.168.4.0 netmask 255.255.255.0 dev br0
#---------------------------end--------------------------
```

And then configure the bridge interface in **/etc/openvpn/bridge**.

```
#!/bin/bash
# Create global variables
# Define Bridge Interface
br="br0"
# Define list of TAP interfaces to be bridged,
# for example tap="tap0 tap1 tap2".
tap="tap0"
# Define physical ethernet interface to be bridged
# with TAP interface(s) above.
eth="eth1"
eth_ip="192.168.8.173"
eth_netmask="255.255.255.0"
eth_broadcast="192.168.8.255"
#gw="192.168.8.174"
...
```

Start the bridge script file to configure the bridge interface.

**# /etc/openvpn/bridge restart**

6. On machine OpenVPN B, modify the remote address in configuration file **/etc/openvpn/tap0-br.conf**.

```
# point to the peer
remote 192.168.8.173
dev tap0
secret /etc/openvpn/secrouter.key
cipher DES-EDE3-CBC
auth MD5
tun-mtu 1500
tun-mtu-extra 64
ping 40
up /etc/openvpn/tap0-br.sh
#comp-lzo
```

7. Next modify the routing table in **/etc/openvpn/tap0-br.sh script** file.

```
#----------------------------------Start-----------------------------
#!/bin/sh
# value after "-net" is the subnet behind the remote peer
route add -net 192.168.2.0 netmask 255.255.255.0 dev br0
#---------------------------------- end -----------------------------
```

And then configure the bridge interface in **/etc/openvpn/bridge**.

```
#!/bin/bash
# Create global variables
# Define Bridge Interface
br="br0"
# Define list of TAP interfaces to be bridged,
# for example tap="tap0 tap1 tap2".
tap="tap0"
# Define physical ethernet interface to be bridged
# with TAP interface(s) above.
eth="eth1"
eth_ip="192.168.8.174"
eth_netmask="255.255.255.0"
eth_broadcast="192.168.8.255"
#gw="192.168.8.173"
...
```

Start the bridge script file to configure the bridge interface.

**# /etc/openvpn/bridge restart**

ATTENTION
Select cipher and authentication algorithms by specifying cipher and auth. To see which algorithms are available, type:

# openvpn --show-ciphers
# openvpn --show-auths

8. Start both OpenVPN peers on machine OpenVPN A and OpenVPN B.
   **# openvpn --config /etc/openvpn/tap0-br.conf&**
   If you see the line **Peer Connection Initiated with 192.168.8.173:5000**on each machine, the connection between OpenVPN machines has been established successfully on UDP port 5000.

**ATTENTION**

You can create link symbols to start the OpenVPN service at boot time:
# ln -sf /etc/init.d/openvpn /etc/rc2.d/S16openvpn

To stop the service, you should create these links:
# ln -sf /etc/init.d/openvpn /etc/rc0.d/K80openvpn
# ln -sf /etc/init.d/openvpn /etc/rc6.d/K80openvpn

9. On each OpenVPN machine, check the routing table by typing the command **# route**

```
Destination     Gateway  Genmsk           Flags    Metric   Ref  Use  Iface
192.168.5.0     0.0.0.0 255.255.255.0   U       0        0    0    eth2
192.168.4.0     0.0.0.0 255.255.255.0   U       0        0    0    br0
192.168.3.0     0.0.0.0 255.255.255.0   U       0        0    0    eth0
192.168.30.0    0.0.0.0 255.255.255.0   U       0        0    0    eth3
192.168.8.0     0.0.0.0 255.255.255.0   U       0        0    0    br0
```

Interface **eth1** and device **tap0** both connect to the bridging interface, and the virtual device **tun** sits on top of **tap0**. This ensures that all traffic coming to this bridge from internal networks connected to interface eth1 write to the TAP/TUN device that the OpenVPN program monitors. Once the OpenVPN program detects traffic on the virtual device, it sends the traffic to its peer.

10. To create an indirect connection to Host B from Host A, you need to add the following routing item:
**# route add –net 192.168.4.0 netmask 255.255.255.0 dev eth0**

To create an indirect connection to Host A from Host B, you need to add the following routing item:

**# route add –net 192.168.2.0 netmask 255.255.255.0 dev eth0**

Now ping Host B from Host A by typing:

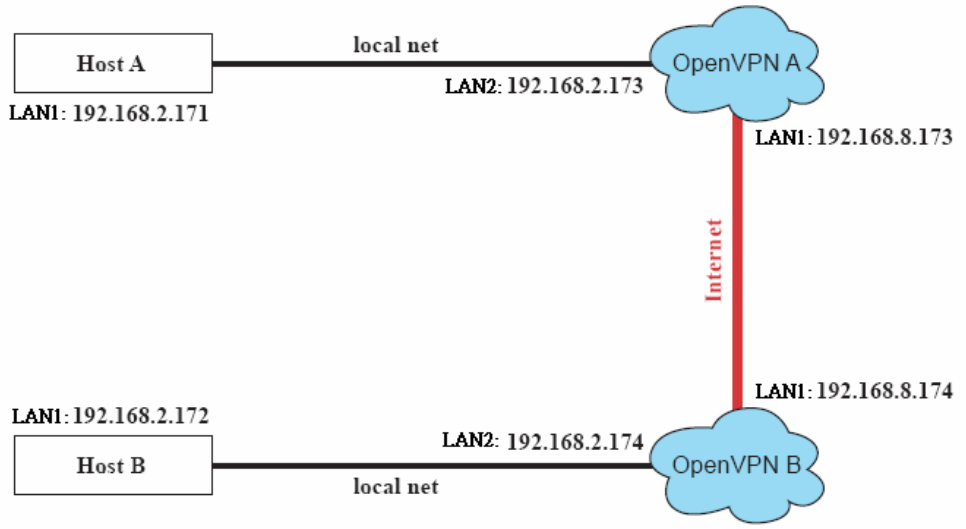
**# ping 192.168.4.174**

A successful ping indicates that you have created a VPN system that only allows authorized users from one internal network to access users at the remote site. For this system, all data is transmitted by UDP packets on port 5000 between OpenVPN peers.

11. To shut down OpenVPN programs, type the command:
**# killall -TERM openvpn**

# Ethernet Bridging for Private Networks on the Same Subnet

1. Set up four machines, as shown in the following diagram.

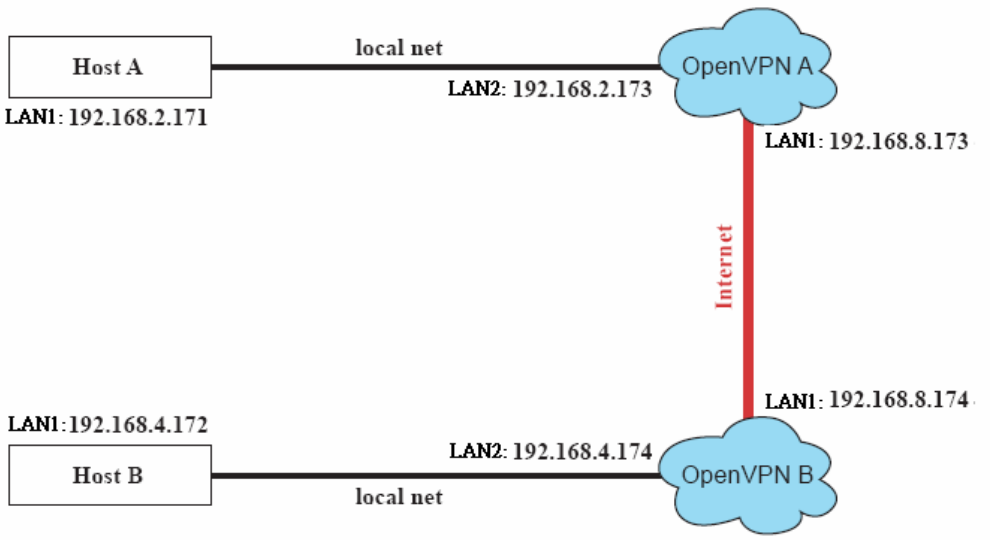



2. The configuration procedure is almost the same as for the previous example. The only difference is that you will need to comment out the parameter **up** in **/etc/openvpn/tap0-br.conf** of OpenVPN A and **/etc/openvpn/tap0-br.conf** of OpenVPN B.

```
# point to the peer
remote 192.168.8.174
dev tap0
secret /etc/openvpn/secrouter.key
cipher DES-EDE3-CBC
auth MD5
tun-mtu 1500
tun-mtu-extra 64
ping 40
#up /etc/openvpn/tap0-br.sh
#comp-lzo
```

# Routed IP

1. Set up four machines, as shown in the following diagram.



2. On machine OpenVPN A, modify the remote address in configuration file **/etc/openvpn/tun.conf**.

```
# point to the peer
remote 192.168.8.174
dev tun
secret /etc/openvpn/secrouter.key
cipher DES-EDE3-CBC
auth MD5
tun-mtu 1500
tun-mtu-extra 64
ping 40
ifconfig 192.168.2.173 192.168.4.174
up /etc/openvpn/tun.sh
#----------------------------end-----------------------------
```

3. Next, modify the routing table in script file **/etc/openvpn/tun.sh**.

```
#---------------------------Start----------------------------
#!/bin/sh
# value after "-net" is the subnet behind the remote peer
route add -net 192.168.2.0 netmask 255.255.255.0 gw $5
#----------------------------end-----------------------------
```

4. On machine OpenVPN B, modify the remote address in configuration file **/etc/openvpn/tun.conf**.

```
# point to the peer
remote 192.168.8.173
dev tun
secret /etc/openvpn/secrouter.key
cipher DES-EDE3-CBC
auth MD5
tun-mtu 1500
tun-mtu-extra 64
ping 40
ifconfig 192.168.4.174 192.168.2.173
up /etc/openvpn/tun.sh
```

And then modify the routing table in script file **/etc/openvpn/tun.sh**.

```
#---------------------------Start----------------------------
#!/bin/sh
# value after "-net" is the subnet behind the remote peer
route add -net 192.168.2.0 netmask 255.255.255.0 gw $5
#----------------------------end-----------------------------
```

The first argument of parameter **ifconfig** is the local internal interface and the second argument is the internal interface at the remote peer.

**$5** is the argument that the OpenVPN program passes to the script file. Its value is the second argument of **ifconfig** in the configuration file.

5. Check the routing table after you run the OpenVPN programs, by typing the command **# route**.

```
Destination     Gateway         Genmsk          Flags   Metric  Ref  Use  Iface
192.168.4.174*                  255.255.255.255 UH      0       0    0    tun0
192.168.4.0     192.168.4.174 255.255.255.0 UG      0       0    0    tun0
192.168.2.0     *               255.255.255.0 U       0       0    0    eth1
192.168.8.0     *               255.255.255.0 U       0       0    0    eth0
```

# 4

# System Recovery

The DA-681-LX is installed with the Embedded Linux operating system, which is located in the Flash DOM (CompactFlash card) shipped with the DA-681-LX computer. Although it happens rarely, you may find on occasion that operating system files and/or the disk file system are damaged. This chapter describes how to recover the Linux operating system.

The following topics are covered in this chapter:

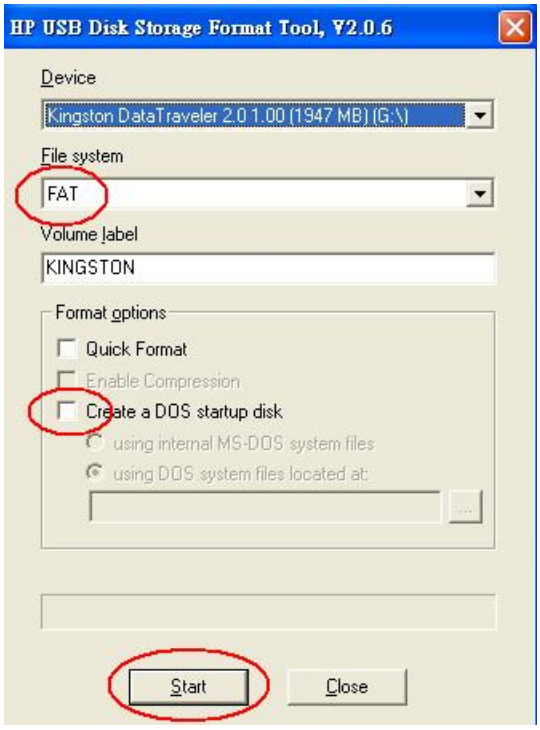
- **Recovery Environment**
- **Recovery Procedure**

# Recovery Environment

The recovery environment includes the DA-681-LX embedded computer and a bootable USB disk with the recovery programs and system image file.


# Recovery Procedure

1. **Format an Empty USB Disk.**
   a. Prepare a USB disk that has at least a 256 MB capacity.
   b. Format your USB disk with the **HP USB Disk Format Tool**. Open the utility and select the device and FAT file system. You need empty disk only. DO NOT check the option **Create a DOS startup disk**.
   c. Click **Start**.



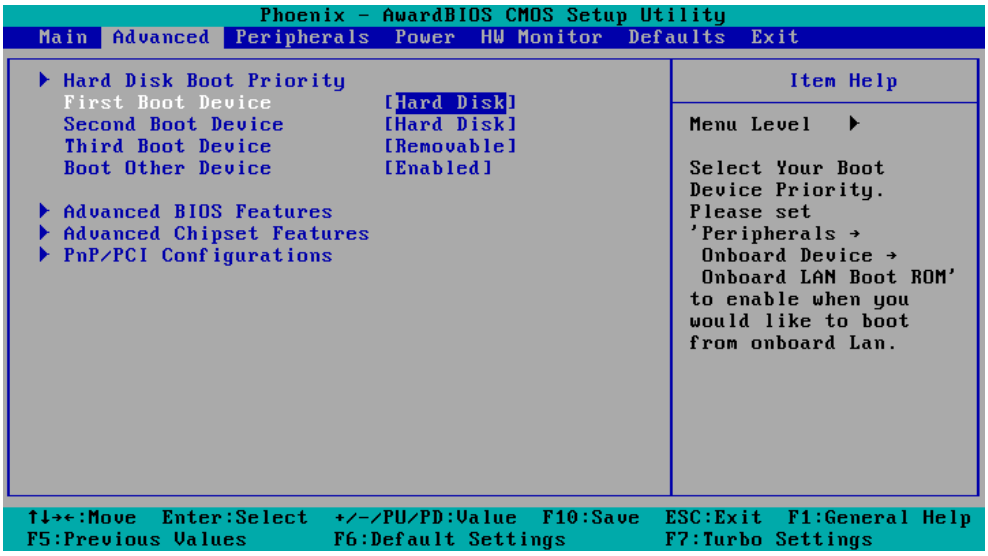

**ATTENTION**

The HP USB Disk Storage Format Tool can be downloaded from many web sites. Do a search on HP USB Disk Storage Format Tool from any search engine to locate the tool.

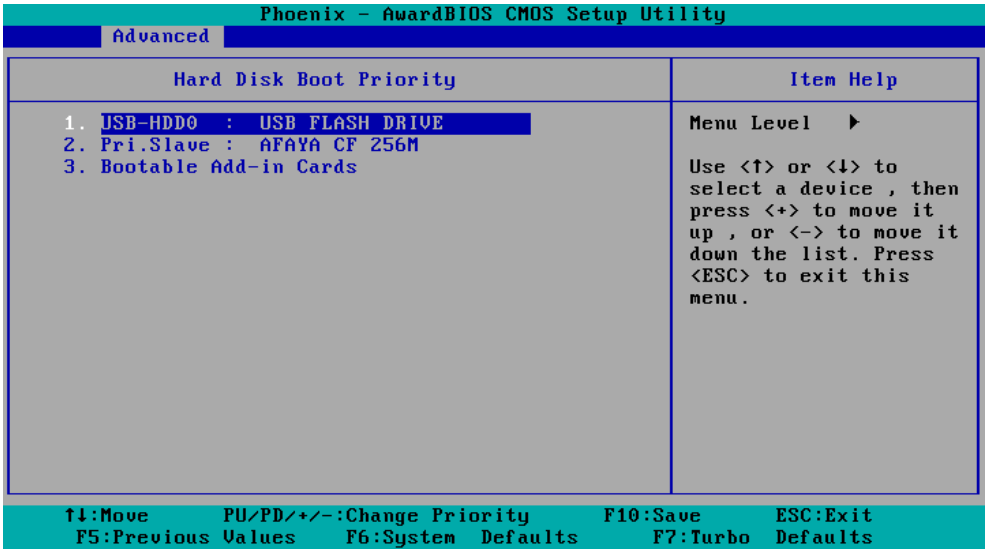
2. **Create a Linux Bootable USB Disk.**
   a. You can find the **firmware** directory in the Recovery CD shipped with the DA-681-LX computer.
   b. Configure Windows Explorer to show hidden files (including protected operating system files).
   c. Copy all files in the **firmware** directory to the root directory of your USB disk.
   d. Open a DOS prompt and type **M:\syslinux.exe M:** to create a bootable Linux disk. In this example, M: is the USB Disk drive number.
3. **Set up the BIOS to Boot from a USB Disk.**
   a. Insert the USB disk.

b. Power on and press **DEL** to enter the bios setup menu.

c. Select **Advanced** → **Hard Disk Boot Priority** and then press **Enter**.




d. From the setup menu, use "↑" or "↓" to select the USB device



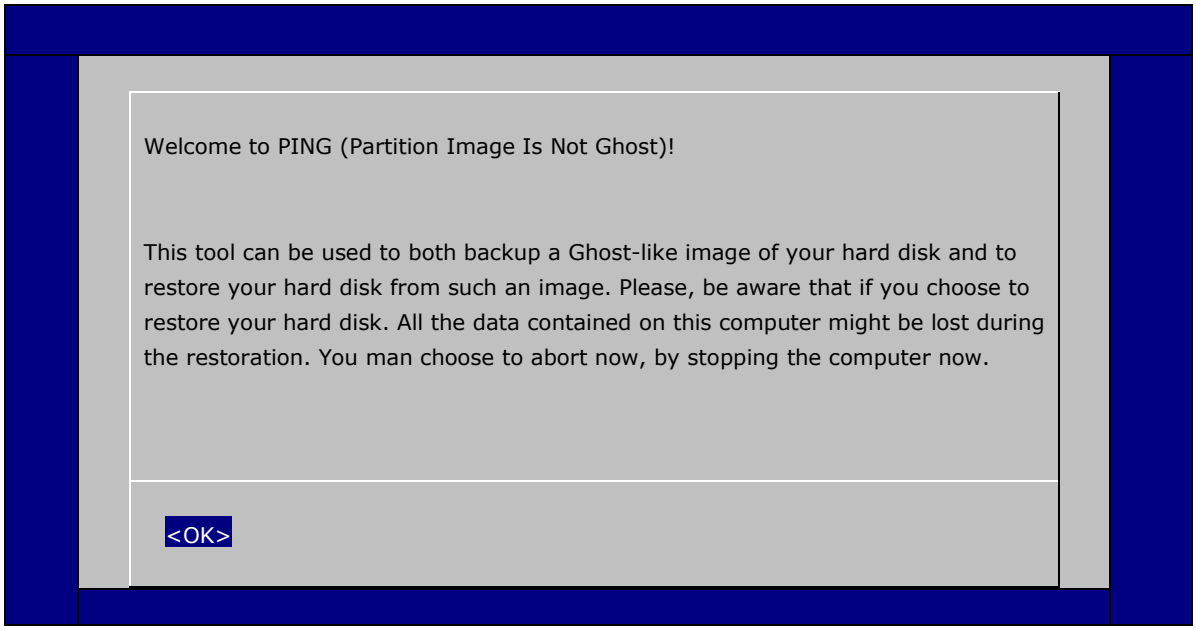

e. Press "+" to move the selection up to the first priority, and press **Esc** to exit the setup menu.

f. Make sure the first boot device is **Hard Disk**. If not, press **Enter** to change it.

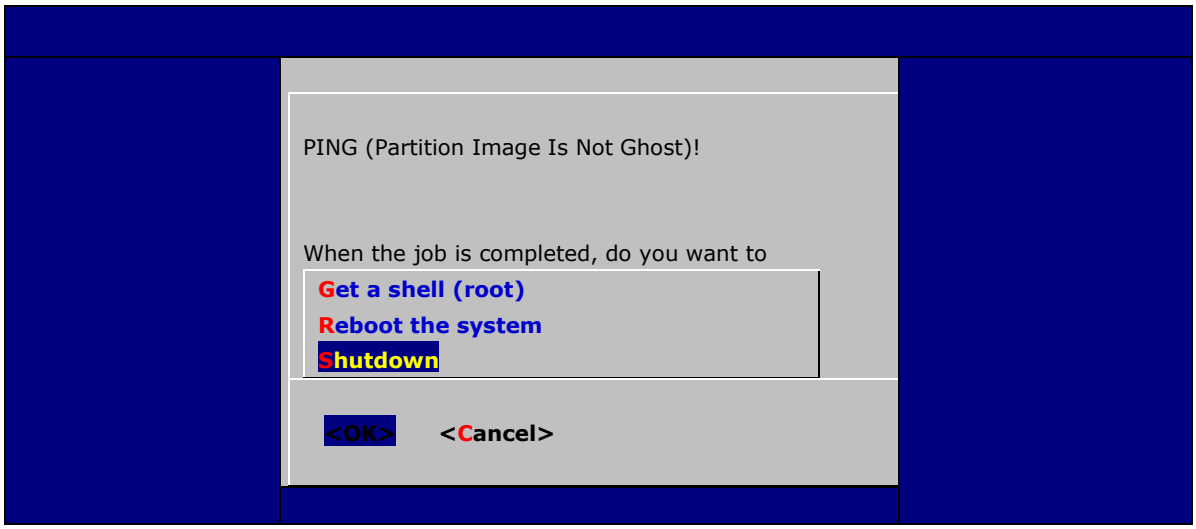
g. Select **Exit** → **Save & Exit Setup** and then press **Enter**.

h. Choose **Y** to save to the CMOS and then exit.

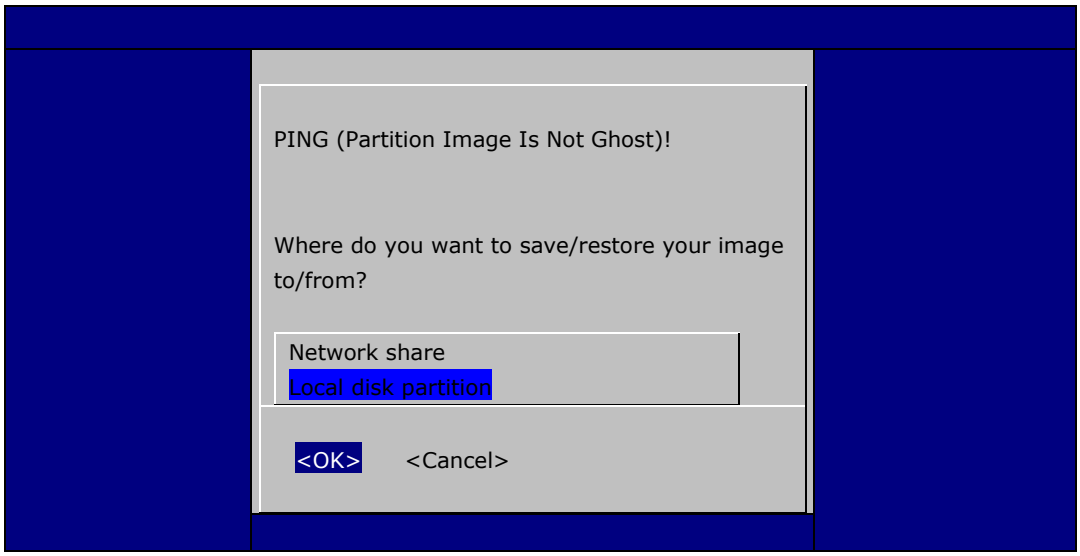
4. **Recover the Linux system from a USB Disk.**
   a. If the BIOS setup is correct, it will boot from the USB disk. Follow the steps below to set up recovery parameters.



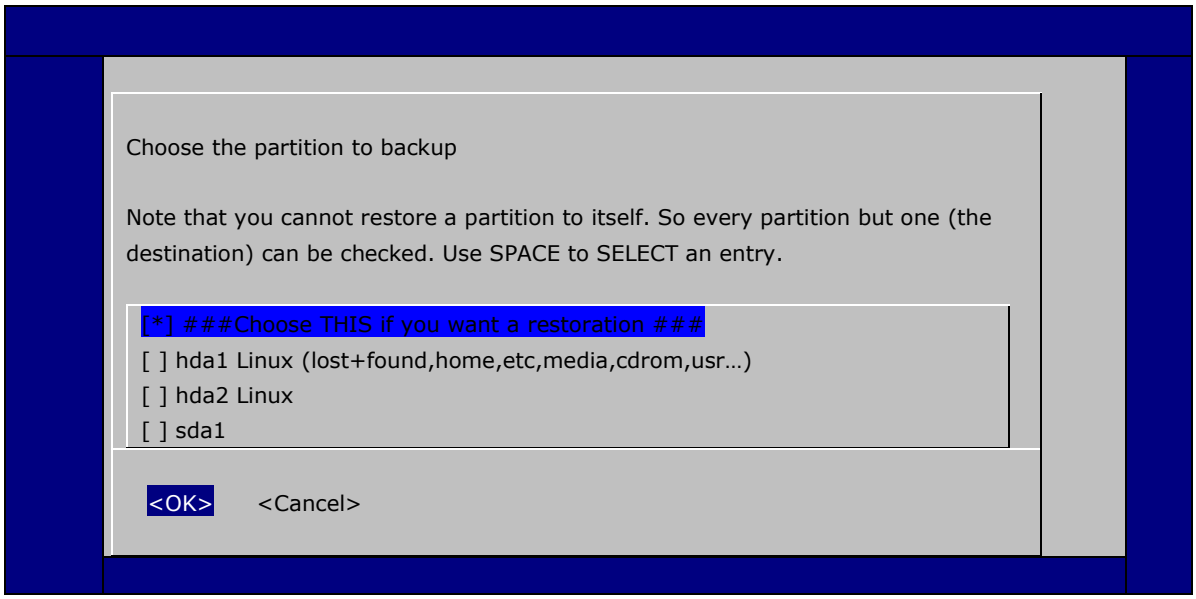

   b. Choose **OK** to go to the next step.
   c. Choose shut down the DA-680-LX when the restoration is finished.



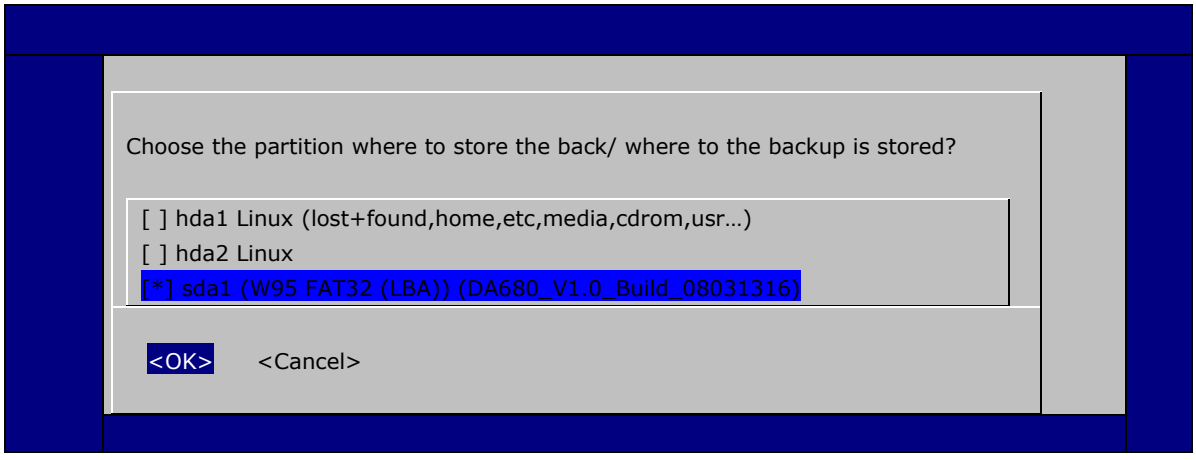
d. Choose restore image from **Local disk partition**.



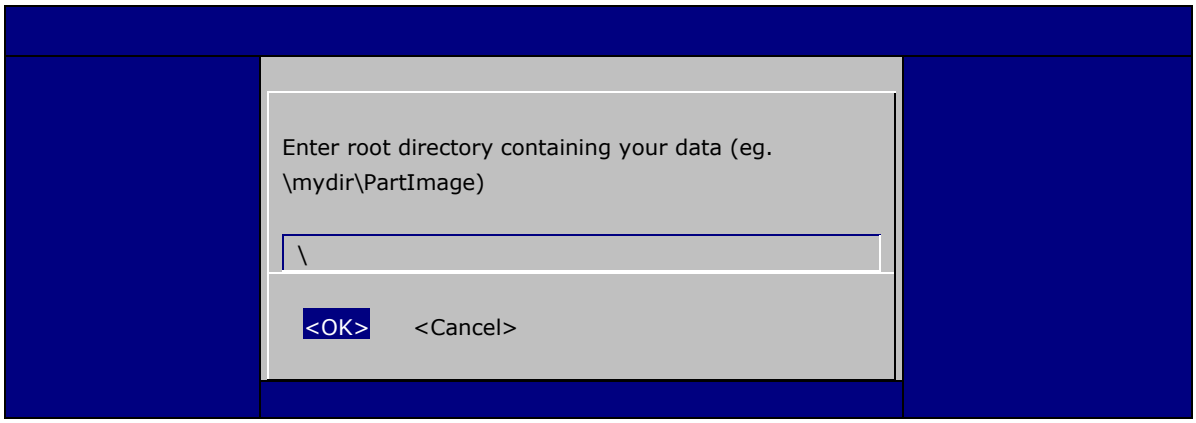

e. Choose **### Choose THIS if you want a restoration ###**



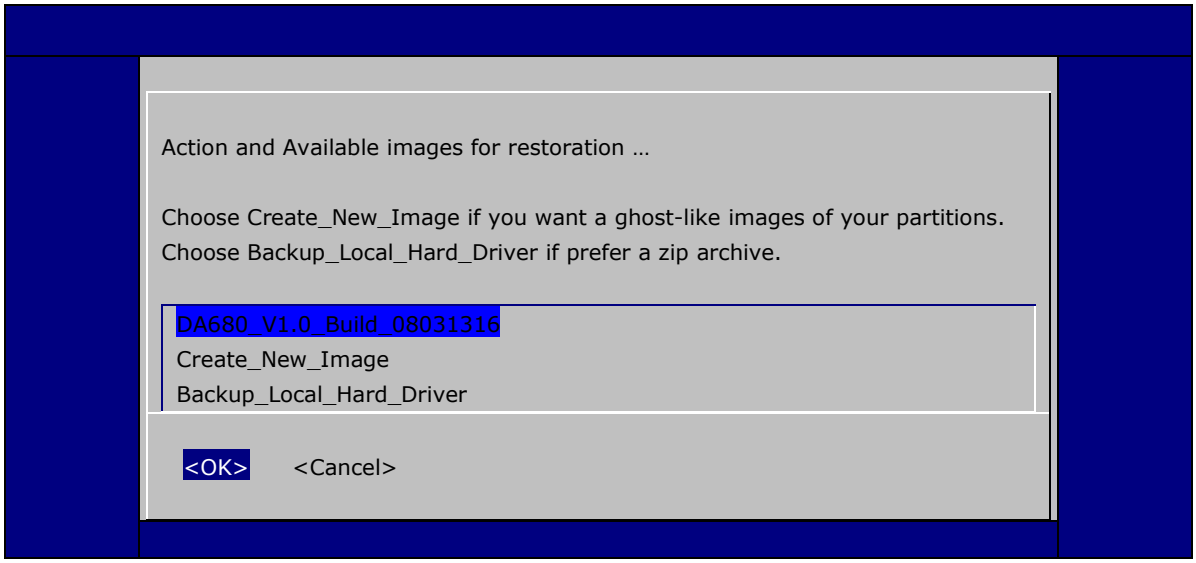

f. Choose the restoration source device **sda1**.



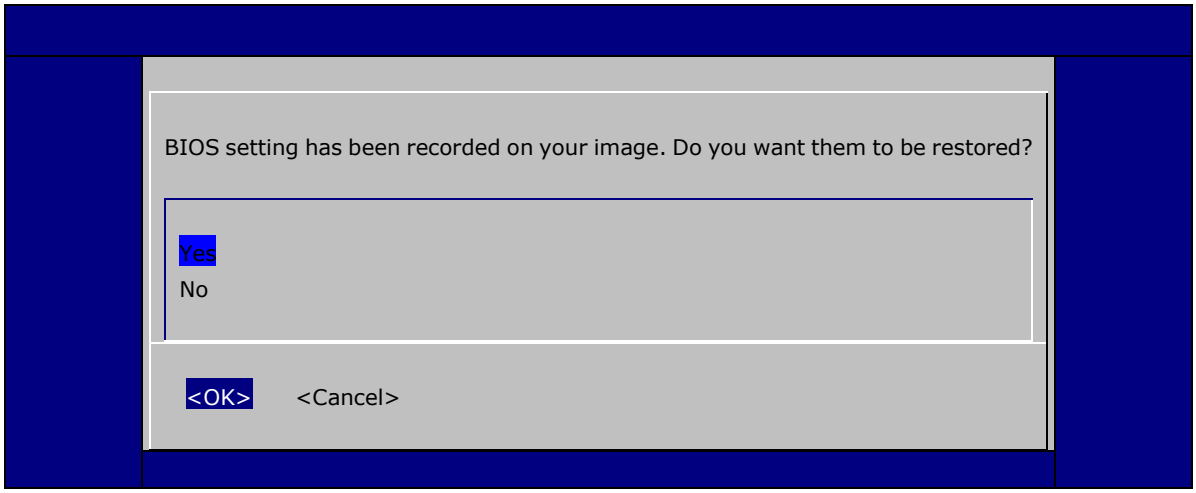
g. Enter **"\"** to choose the root directory of the restoration image.




h. Choose **DA680_V1.0_Build_08031316** for the restoration image.




i. Choose **Yes** to start the restoration. After the restoration is finished, the system will halt and you will need to reboot to restart the restored system.

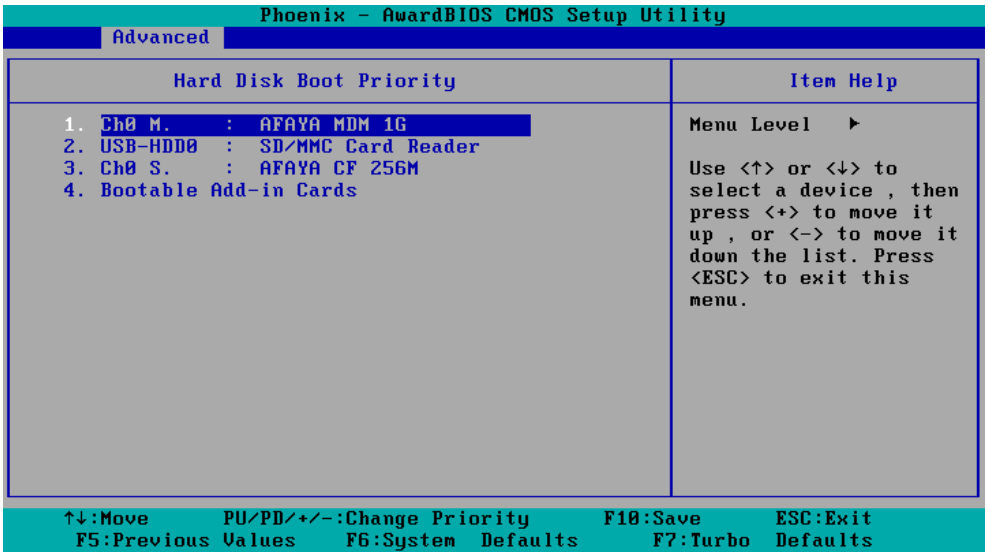



When operation is finished, turn off the computer and remove the USB disk.

**ATTENTION**

DO NOT turn off the power during system recovery, as the system may crash.

5. **Set up the BIOS back to boot from DOM or CompactFlash Disk.**
   a. Power on and press **DEL** to enter the bios setup menu.
   b. Select **Advanced** → **Hard Disk Boot Priority** and then press **Enter**.




   c. From the setup menu, use "↑" or "↓" to select the DOM or CompactFlash device.
   d. Press "**+**" to move the selection up to the first priority, and press **Esc** to exit the setup menu.
   e. Select **Exit** → **Save & Exit Setup** and then press **Enter**.
   f. Choose **Y** to save to the CMOS and then exit.
   g. Wait a few minutes for the system to boot. When the recovery process is finished, you will again be able to see the Linux desktop.