Quick Start Guide

LynxOS Version 2.5

DOC-0167

Product names mentioned in the *Quick Start Guide* are trademarks of their respective manufacturers and are used here only for identification purposes.

U.S. Patents 5,469,571; 5,594,903

Printed in the United States of America.

Contents

QUICK START GUIDE

Welcome to LynxOS!

Whether you are an experienced UNIX programmer who wants to quickly learn specifics of LynxOS, or you are new to UNIX-like operating systems and need a basic introduction to get you started, this guide is for you. As you will see in the following pages, LynxOS is a powerful, UNIX-like development environment providing the utilities and tools you need to easily and quickly build real-time applications. Read this guide for a quick overview of the steps needed to develop, compile, link, debug, and execute your applications using LynxOS.

This guide assumes that LynxOS has been installed and configured on your system. See the LynxOS *LynxOS Installation Guide* for more information.

All references to file names and commands are case sensitive and should be typed exactly as shown. Different typefaces are also used to emphasize important concepts. These are described below.

Description	Example
`Courier` font denotes commands, programs, file names, and words that would appear on your computer screen.	`myprog.c` Enter `ls` on the command line
`Bold Courier` font indicates words that you have typed on your computer.	`login:` **`myname`** **`cd /usr/home`**
Words that you must replace with a real name or value appear in *italics*.	`cat` *filename* `mv` *file1 file2*
New terms, book titles, and emphasized words appear in *italics*.	Refer to the *LynxOS Reference Manuals.*

For More Information

Browse our Web site for additional information and answers to Frequently Asked Questions (FAQs).

Lynx Home Page

http://www.lynx.com

Also, Lynx Technical Support is available to answer your questions Monday through Friday between 8:00 AM and 5:00 PM Pacific Time (Lynx U.S. Headquarters) or between 9:00 AM and 6:00 PM Central European Time (Lynx Europe).

Lynx U.S. Headquarters

Internet: support@Lynx.COM
Phone: (408) 879-3940
Fax: (408) 879-3945

Lynx Europe

Internet: tech_europe@Lynx.COM
Phone: (+33) 1 30 85 06 00
Fax: (+33) 1 30 85 06 06

What is LynxOS?

LynxOS is a UNIX-compatible, POSIX-compliant, real-time operating system. It is designed to be used in time-critical applications where predictable real-time response is necessary. At its fullest, LynxOS is a self-hosted workstation development environment that is available in both native and cross development versions.

LynxOS supports IBM-compatible X86 (386, 486, Pentium, and Pentium Pro); PowerPC (603, 603e, and 604); 68K (68030, 68040, 68LC040, and 68060); and SPARC (microSPARC and microSPARC II) processor families. Please see the *LynxOS Hardware Support* Guide for more details.

LynxOS Features

- Multi-processes and a multi-threaded environment
- Sophisticated memory management through the hardware Memory Management Unit (MMU)
- Configurable demand-paged virtual memory
- Hierarchical, UNIX-like file system
- Modular scalable architecture
- Kernel threads
- Network File System (NFS)
- Industry Standard Networking (TCP/IP)
- STREAMS Facilities
- Remote Procedure Calls (RPC)
- Support for diskless clients
- X11R6 and OSF/Motif graphical user interface
- Industry standard GNU tools, UNIX-like utilities, and UNIX-like shell scripts
- ROM-able kernel

LynxOS Documentation

LynxOS comes with a complete suite of programmer and reference documentation that detail the utilities and tools introduced in this guide. LynxOS also provides on-line reference documentation through UNIX-like manual pages.

LynxOS Specific Documentation

- *Release Notes*
- *LynxOS Hardware Support*
- *LynxOS Installation Guide*
- *LynxOS User's Guide*
- *Writing Device Drivers for LynxOS*
- *LynxOS Reference Manuals* (seven volumes)
- *GNU C/C++ Tools* (five volumes)

Other Product Documentation

- *X Window System User's Guide*
- *X Window System Programmer's Supplement for Release 6*
- *POSIX Programmer's Guide*
- *POSIX.4: Programming for the Real World*

What's all the Excitement about POSIX?

POSIX, the Portable Operating System Interface for Computer Environments, is both an IEEE Standard (IEEE 1003.1) and an ISO Standard (ISO 9945-1). POSIX defines the interface of the operating system at a high level.

"Software portability yields significant cost and time savings, resulting in shorter time-to-market and lower cost-to-market for your products."

The primary benefit of developing POSIX-conformant software is that it promotes platform (hardware) independent code and thus aids in source code portability.

LynxOS and POSIX Standards

LynxOS conforms to the following POSIX standards:

- POSIX 1003.1—the operating system interface standard to ensure application portability
- POSIX 1003.1b—the base real-time extensions to POSIX 1003.1
- POSIX 1003.1c (Draft 14)—the threads extensions to POSIX 1003.1

POSIX Benefits

Lynx views POSIX as a significant benefit to customers allowing application developers and system integrators the ability to:

- Get products to market faster
- Lower the cost of getting products to market
- Preserve software development investment
- Reuse software
- Gain a higher level of software verification
- Integrate software more easily
- Extend POSIX conformant software with less error

For More Information

POSIX Programmer's Guide
POSIX.4: Programming for the Real World

Getting Started

Now you're ready to start using LynxOS. The following sections describe how to:

- Start and shutdown the LynxOS Development Environment
- Start and exit from the X Windows
- Use PosixWorks Desk® (x86 systems only)
- Use very basic UNIX commands

Starting and Stopping LynxOS

You will notice that LynxOS boots from power-up to a command line prompt. A banner identifying LynxOS is also displayed. If your system administrator has set up password access, you will be prompted to enter your login name and password. Depending on how your system administrator has configured your access and privileges, you will be in either the root directory or your user directory. To view your directory, enter `pwd` at the command prompt.

LynxOS should not be powered-off without going through a normal shutdown sequence. Enter the following to shutdown LynxOS†:

```
reboot -h
```

You can power off after disk activity has stopped and you see the following message:

```
**** LynxOS is down ****
```

†. You must have root privileges to shutdown. Or, your system administrator can use chmod to assign less restrictive user privileges to reboot.

Starting and Stopping X Windows

Most configurations of LynxOS come with X11R6 and OSF/Motif (referred to as the “X Window System” or just “X Windows” throughout this document). X Windows provides an industry standard graphical user interface (GUI) based on X11R6. To start X Windows, enter the following at the command prompt:

```
startx
```

To exit from the X Windows System, press **Ctrl-Alt-Backspace**.

Using PosixWorks Desk

PosixWorks Desk provides quick access to LynxOS tools and utilities. It has toolbars with buttons and icons for a visual picture of your development environment. The folder-style file manager allows you to drag and drop files. You can choose the option to install PosixWorks Desk when you install the X Windows. Press the Help button on the main toolbar to get detailed information on the features supported by PosixWorks Desk.

Learning Basic LynxOS Commands

Since LynxOS is UNIX-like, commands that you may know from experience with other UNIX-like operating systems work similarly under LynxOS. This section briefly describes some commands that are necessary for general work with LynxOS. To find more detailed information about these commands, access the on-line reference manuals (UNIX-like “man pages”) by typing `man` *command* at the prompt.

Following are some basic LynxOS commands that you should know:

`cd` *directory_name*	Change current working directory to *directory_name*. To go back up one directory level, enter: **`cd ..`**
`cp`	Copy files. **`cp`** *current_file another_file*
`find` *exp*	Find files that match expression. `find` searches the subtree beneath each directory for files that match the given search expression (*exp*). **`find / -name`** *file* **`-print`**
`gcc`	GNU C compiler. **`gcc`** *filename*
`less` *filename*	Display the contents of file *filename* one screen page at a time. Press the space bar to display the next page. **`less`** *filename* The display is treated as though viewing a file via an editor; therefore, `less` must also be exited with the `q` (quit) command. However, `less` is more like a read-only editor in that you can scroll up and down while viewing the file. Also, `less` has a help facility that describes additional options, such as string search.
`ls`	Show directory contents and file information. **`ls`**
`man`	On-line help utility that displays manual pages. When you enter `man` and the name of a command, utility,

	or tool, you will be presented with information about that resource. **`man`** *command*
`mkdir`	Create a directory. **`mkdir`** *directory_name*
`mv`	Move or rename files. **`mv`** *current_file new_file*
`pwd`	Show the current working directory. You may see something like: `/` The slash indicates that you are in the root directory.
`rm`	Remove (delete) files. **`rm`** *filename*
`rmdir`	Remove (delete) directories. **`rmdir`** *directory_name*
`vi`	Visual text editor. **`vi`** *filename*
`reboot`	Reboot the system—shutdown LynxOS. **`reboot -h`**

Accessing On-line Reference Pages

LynxOS provides a UNIX-like on-line reference utility called `man`. This utility displays reference pages, called "man pages", that describe the commands available in LynxOS. The man pages are actually text files that are located in the `/usr/man` directory. However, you can access the `man` command from anywhere on your system.

Since the LynxOS man pages are just ASCII text files, the `man` command actually uses the LynxOS utility `less` to display the information (see the description of `less` in the previous section). `less` is very similar to a read-only editor in that you can view a text file and perform some operations on the file, such as scrolling and string search. To view an on-line reference manual page enter:

`man` *reference_command_of_interest*

If you are interested in a topic but aren't sure what commands are available, you can have `man` search for a keyword using the **-k** option:

`man -k` *keyword*

This is the least restrictive search—it is not case-sensitive, the keyword can be within the description of one or many commands, and the keyword can be a command name itself, (as shown in the example below).

```
man -k less
less (1)        - interactive paginator files, viewing
lesskey (1)     - specify key bindings for less
slaveboot (1)  - transfers a kernel image to a diskless
SCMP client
```

You can also use `man` to display the description line of a command using the **-f** option.

man -f *command*

This search is more restrictive: it is case sensitive and *command* must be a valid command. For example:

```
man -f less
less (1)        - interactive paginator files, viewing
```

If neither of these options help you find the information that you are looking for, you can view an alphabetical index of all available on-line manual pages by entering:

less /usr/man/windex

While you are viewing the manual page with the `man` command, you can get help on available viewing options by pressing the **h** key. To exit a session, press the **q** key to quit.

Copying Files to Floppy Disks

LynxOS supports floppy disks on X86, PowerPC, and microSPARC systems. This section gives an example of using a high density floppy disk on an X86 system.

1. Insert a floppy.
2. Format the floppy with `fmtflop`.
   `fmtflop -v /dev/rfd1440.0`
3. Create a LynxOS file system on the floppy.
   `mkfs -v /dev/fd1440.0`
4. Mount the floppy on `/mnt`. (This directory must be empty and not currently mounted by any other file system.)
   `mount /dev/fd1440.0 /mnt`
5. Copy files to the floppy.
   `cp /stuff/file.xyz /mnt`
6. To see what's on the floppy, use the `ls` command.
   `ls /mnt`
7. Unmount the floppy when you're done.
   `umount /mnt`

Making Backups

You can use the archive utility, `tar`, to back up important data files. The media that you use will depend on your system and the amount of data you plan to back up. For example, if your "home" directory is `/usr/home/mystuff` and you want to back up its contents to a tape device with SCSI ID 4, use the following command:

tar -cvf /dev/*rawdevice* **/usr/home/mystuff**

The `cvf` arguments direct `tar` to (`c`) create a new tar archive, (`v`) be verbose, and (`f`) direct the output to a file (in this case, the tape device *rawdevice*). For this example, if you have an X86 system with an Adaptec1542 SCSI controller, *rawdevice* is `rst1542.4`. On PowerPC, 68k, and SPARC systems, *rawdevice* is `rtape4`.

To display the contents of an archive:

tar -tvf /dev/*rawdevice*

To extract the archive from the tape device and put it back on the disk:

tar -xvf /dev/*rawdevice* **/usr/home/mystuff**

If the directory `/usr/home/mystuff` already exists on the disk, the `tar` command shown above will overwrite it with the data from *rawdevice*.

For More Information

LynxOS Reference Manuals, “Utility Programs”
LynxOS User’s Guide

The Application Development Process

Figure 1 shows the basic steps in the application development process. The following sections briefly describe these components with references to more detailed information in the *LynxOS User's Guide*.

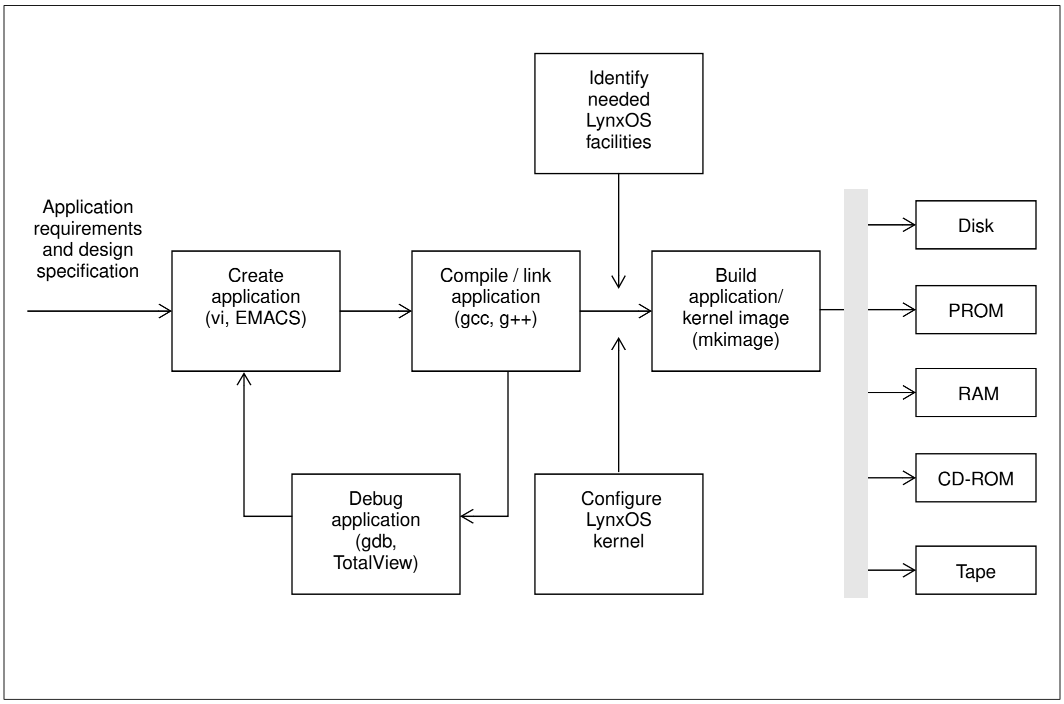

Figure 1: Application Development Process Overview

Creating Source Code, the vi Way

Those of you already familiar with UNIX-like operating systems probably have used the `vi` text editor. LynxOS provides `vi` with the features and capabilities that you are familiar with. For those that aren't familiar with `vi`, this section provides a quick list of important functions.

First of all, `vi` is a full screen text editor used to create and modify ASCII files. There are two main modes in `vi`—*command mode* and *input mode*. Command mode is used for issuing short commands. Input mode is used to add text to the file.

The following example shows how to create a text file, in this case a simple C language program, using `vi`. Other C program source files can be found in `/src/examples`.

Create and open the file by typing:

```
vi HelloWorld.c
```

Put the `vi` editor into input mode (by pressing the **i** key) and type some characters. To quit `vi` and save the changes that you made, first return to command mode by pressing the **Esc** key, then enter:

```
:wq
```

To quit `vi` without saving anything you entered or changed, enter:

```
:q!
```

To quit `vi` when nothing has been changed, enter:

```
:q
```

Basic vi Commands

Move Cursor

`h`	move cursor left one character
`l`	move cursor right one character
`k`	move cursor up one character
`j`	move cursor down one character

Delete/Undo

`x`	delete character at cursor
`dd`	delete row at cursor
`u`	undo previous action

Input Mode Options

`i`	enters input mode and starts insertion before the character at the cursor
`a`	enters input mode and starts insertion after the character at the cursor

Try It!

Begin `vi` as previously shown and enter the following text:

```
#include <stdio.h>
main ()
{
    int index = 10;
    printf("\n\n*****************\n");
    printf("Welcome to LynxOS\n");
    printf("*****************\n\n");
}
```

Compiling and Linking Source Code

After entering and saving the source code file from the previous example, you're ready to create the executable application by compiling and linking it.

Since the previous example was a C language program, we will use the LynxOS GNU C compiler. To compile, enter:

```
gcc HelloWorld.c
```

If the program is error free, the compiler will invoke the linker and generate a default executable file called `a.out`. You can execute the file by typing `a.out` on the command line.

You can also make the compiler rename the executable something different than `a.out` by using the **–o** *name* option as shown in the next example:

```
gcc –o HelloWorld HelloWorld.c
```

The resulting executable will be a file named `HelloWorld`. To execute the file, enter:

```
HelloWorld
```

You will see displayed:

```
****************
Welcome to LynxOS
****************
```

For More Information

Using GNU CC
GNU Make
Development Support Tools

Debugging Source Code

Lynx provides several debuggers. `gdb` is provided as a normal part of the LynxOS development environment while TotalView is offered as a separate product. For this brief introduction and because you may not have purchased TotalView, we will use `gdb` for basic debugging services.

First, don't forget that you must compile your source using the **-g** debug option as part of the compile command line.

```
gcc -g -o HelloWorld HelloWorld.c
```

To start `gdb`, enter:

```
gdb HelloWorld
```

To quit `gdb`, enter:

```
quit
```

`gdb` does not initialize and set a breakpoint† at a source line. You must provide that initial breakpoint. It is usually convenient to set a breakpoint at main.

```
break main
```

You only have to type as much of the command to make the command unique. Some basic `gdb` commands are listed below.

`help`	List the `gdb` commands and options. Many useful commands are found under `Help` and `Data`, `Breakpoints`, or `Files`.
`list`	Display the source.
`step`	Step to the next source line.
`break` *option*	Set a breakpoint at a specified line number or function.

†. A breakpoint makes your program stop whenever a certain point in the program is reached during program execution.

`clear` *option*	Clear all breakpoints or clear the specified breakpoint *option*.
`inspect` *expression*	Show the value of a data structure.
`run`	Run program to completion or to the next breakpoint.
`quit`	Quit `gdb`.

For More Information

Debugging with GDB
DejaGNU Testing Framework

Configuring LynxOS for Your Application

LynxOS allows you to remove those components and features of the LynxOS Kernel that your application does not require. This allows you to configure LynxOS and your application to the smallest possible memory footprint, an advantage when memory may be constrained or where the cost of memory is a key component of your product definition.

See the chapter “Customizing LynxOS” in the *LynxOS User’s Guide* for more information and examples of this process.

For More Information

LynxOS User’s Guide

Identifying LynxOS Facilities for Run-Time

You can choose to distribute your final software product as a ROM- or RAM-based file system containing a LynxOS kernel and your application. Or, you can choose to distribute your software product as a LynxOS kernel, your application, and some subset of development facilities on more conventional storage media such as a disk or a CD-ROM.

To distribute LynxOS in any of these configurations, you must purchase appropriate run-time royalty licenses from Lynx.

NOTE: *The number of run-time images that you can create depends on your license agreement with Lynx. Refer to your agreement for specific details, feature capabilities, allowed time frame, and reporting procedure.*

Making Final Images

Use the LynxOS `mkimage` utility to create a final image of your application, LynxOS kernel, and LynxOS facilities for run-time. The *LynxOS User's Guide* and the `mkimage` reference page describe this process in detail.

For More Information

LynxOS User's Guide

LynxOS Cross Development

The LynxOS Cross Development System is a self-contained environment that allows you to develop both LynxOS bootable kernels and LynxOS user application code on a non-LynxOS machine. It contains cross development compilers, linkers, libraries and other development tools.

See the *LynxOS Hardware Support* list for more information about the host operating system environments that can function as LynxOS Cross Development Systems.

Why Use Cross Development?

By developing your LynxOS real-time applications on a cross development workstation, you can access workstation-based utilities such as CASE and configuration management tools while developing your real-time applications for LynxOS.

The LynxOS Cross Development System has been designed to provide a seamless native and cross development environment. In other words, the steps taken to develop code on a native LynxOS target can be duplicated on the cross development host. Makefiles and source code are identical in both environments, and in fact, can be shared.

For More Information

LynxOS User's Guide

Other Documentation

Lynx documentation provides the information you need to use LynxOS and its development environment for creating your real-time applications. However, we do recommend that you peruse some of the many commercially available references for more information on advanced real-time programming, UNIX and POSIX, networking, industry standard tools, and software engineering processes. We offer the following titles as a starting point—this list is by no means comprehensive.

General UNIX Titles

- *Learning the UNIX Operating System*, G. Todino, J. Strang & J. Peek, O'Reilly & Associates, Inc.
- *Essential System Administration*, A. Frisch, O'Reilly & Associates, Inc.
- *Managing NFS and NIS,* H. Stern, O'Reilly & Associates, Inc.
- *TCP/IP Network Administration*, C. Hunt, O'Reilly & Associates, Inc.

Programming Titles

- *The C Programming Language,* B. Kernighan & D. Ritchie, Prentice Hall.
- *C: A Reference Manual*, S. Harbison & G. Steele, Prentice Hall.
- *Learning the BASH Shell*, C. Newham & B. Rosenblatt, O'Reilly & Associates, Inc.
- *Exploring Expect: A Tcl-based Toolkit for Automating Interactive Programs,* D. Libes, O'Reilly & Associates, Inc.
- *Advanced Programming in the UNIX Environment*, Stevens, Addison-Wesley Publishing Company, Inc.

- *Programming with GNU Software*, M. Loukides, O'Reilly & Associates, Inc.
- *Practical C Programming*, S. Oualline, O'Reilly & Associates, Inc.
- *Practical C++ Programming*, S. Oualline, O'Reilly & Associates, Inc.
- *Managing Projects with Make*, A. Oram & S. Talbott, O'Reilly & Associates, Inc.

POSIX Titles

- *POSIX Programmer's Guide*, Donald Lewine, O'Reilly & Associates, Inc.
- *POSIX.4: Programming for the Real World*, Bill Gallmeister, O'Reilly & Associates, Inc.

General Titles

- *Managing the Software Process*, Watts S. Humphrey, Addison-Wesley Publishing Company, Inc.
- *The Mythical Man-Month: Essays on Software Engineering,* F. Brooks, Addison-Wesley Publishing Company, Inc.
- *Zen and the Art of Motorcycle Maintenance*, Robert M. Pirsig, Bantam Books

Lynx Real-Time Systems, Inc. does not endorse any of the publishers, titles, or authors listed throughout this guide but only provides them as a convenience for you. There are many alternate books that may be equally good or better reference material. *Zen and the Art of Motorcycle Maintenance* is listed because this author thinks it is one of the best books ever written on the subject of the definition of quality, especially as it relates to technology and human understanding.