Xinu Programmer's Manual

Version 7.9 (PC)

(XINU IS NOT UNIX)

The Xinu Programmer's Manual contains a description of the Xinu software. It has been divided into sections, following the style of the *UNIX Programmer's Manual*. This introduction outlines how to compile a C program, load it along with Xinu, and execute the resulting image.

The body of the manual gives a terse description of Xinu procedures and the details of their arguments - it is intended as a quick reference for programmers, not as a way to learn Xinu. Section 1 describes commands that run on the PC under the Xinu shell. These commands allow the user to peer into the system by formatting information from internal data stuctures like the process table. Some commands allow the user to exercise data communications functions like sending datagrams. Section 2 describes Xinu system calls that programs call to invoke operating system services. Section 3 describes procedures available from the Xinu libraries. (From the programmer's point of view, there is little distinction between library routines and system calls). Section 4 describes the Xinu device drivers.

As in the UNIX manual, each page describes one command, system call, or library routine; section numbers appear in the page header as "(digit)" following the name of the program. Within a section, all pages are arranged in alphabetical order. References have the same form as headers (e.g., "getc(2)" refers to the page for "getc" in section 2). Related commands are sometimes mentioned on one page (which may make it difficult for beginners to find them).

A Short Introduction to Using Xinu

Overview. To run a program under Xinu, you create the source file, invoke conventional development software to compile and link the program along with the Xinu system. Once a complete memory image has been produced, it can be executed as an application from MSDOS. During execution, the Xinu Operating System takes over the PC and handles the keyboard, display and network devices.

Development commands. The development environment consists of a conventional C compiler, library manager and linker as well as a few miscellaneous commands like the UNIX "make" utility. The details of these commands can be found in your vendor's manuals; examples shown below are for tutorial purposes only.

Compiling programs. The batch command *run filename* invokes the C compiler to translate a source program into an object program. If the compilation was successful, the batch command links the object program with the Xinu library to produce a memory image. *Filename.exe* is then executed.

An Example

DOS environment setup. Always run *c:\xinu79\pcxnet.bat* before doing any Xinu development. This batch file sets up the MSDOS environment.

Create a C program. Shown below is a C program that prints the string "Hello world." on the console and exits. Create the file in c:\xinu79\pcxnet\main using a programmer's editor.

```
/* example C program in file example.c */

xmain()
{
     printf("Hello world.\n");
     xdone();
}
```

Compile. Compile and run the program with the command *run example*. The compiler will compile the C program, link it with the Xinu library and produce an executable called *example.exe.* When the PC then executes *example* you will first see a few startup messages:

```
C:\XINU79\PCXNET\MAIN>example
Initializing . . .
Disk read error 80H reading drive 0

XINU Version 7.9 PC (1-Jun-93)
651376 real mem
340480 base addr
310896 avail mem

Hit any key to continue . . .
```

The screen will now clear, a network initialization message will appear, and the output from *example* will be displayed. The disk read error is of no concern, it simply indicates that there is no disk in drive A:.

System Termination. Whenever an *xdone()* is executed or <ctrl-break> is pressed, the Xinu system terminates and control is passed back to MSDOS. The termination message appears:

```
-- system halt --

XINU 7.9 terminated with 19 processes active
Returning . . .
```

Xinu started *xmain* as a process, which was still running when *xdone()* was executed. The other processes were created when Xinu initialized. They handle network services and devices.

Xinu Directory Organization

The Xinu software is distributed as a self-extracting archive called xinu79.exe. It is available from csc.canberra.edu.au in /pub/ise/xinu. The file should be copied to a hard disk and then executed. The following directory tree will be generated:

```
C:\XINU79
|---BIN                    editor, compiler, make etc
|---BAT                    make make batch files
|---LIB                    library dependency files
|---PCXNET
|   |---MAIN               xinu.c
|   |---LIB                xinu libraries
|   |---BAT                batch files
|   |---SRC                all source files
|       |---ARP
|       |---DGRAM
|       |---H
|       |---ICMP
|       |---IP
|       |---NET
|       |---NETAPP
|       |---SHELL
|       |---SNMP
|       |---KERNEL
|       |---TCP
|       |---TCPD
|       |---CHARDEVS
|       |---UDP
|       |---UTILS
|       |---CONFIG
|       |---MISCDEVS
|       |---DISKDEVS
|       |---SERIAL
|       |---TCL
```

Altering the Xinu System

Xinu can be expanded and customized because it is distributed as source code. In fact, before running Xinu for the very first time, a working system must be generated. To do so, perform the following steps:

```
cd c:\xinu79
C:\XINU79\pcxnet                setup the MSDOS environment
C:\XINU79\PCXNET\MAIN>src .
C:\XINU79\PCXNET\SRC>make lib
```

The entire Xinu system will now be compiled. This takes about 20 minutes on a 16MHz 386SX system with a 1 Meg SMARTDRIVE. Note that an environment space of 4096 bytes is required.

When compilation is complete perform the following steps:

```
C:\XINU79\PCXNET\SRC>main
C:\XINU79\PCXNET\MAIN>make
C:\XINU79\PCXNET\MAIN>xinu
```

Xinu will now execute and present a login prompt. After login, a shell will be run, which interprets commands contained in a file called *startup*. This file currently contains commands which create two shells and a status window. Keyboard input goes to the window in which the cursor blinks. The PC function keys can be used to switch between windows.

When customizing Xinu, always do a *make* in any directory containing C or ASM files which have been altered. If you have added source files to the system, do a *makemake* in the directory containing those files, then do a *make*. This will generate new dependency files. *Make* will compile (or assemble) altered source files and add the objects to the relevant library file. Then switch to the *main* directory and do a *make*. This will generate a new *xinu.exe* using the updated libraries. Changes to the contents of the *h* directory will require a *make lib* in the *src* directory.

Writing Xinu Shell Commands

All commands which the shell recognizes are contained in *src\shell\cmd.c*. This file must be edited and recompiled when you add a new command to the shell.

The shell executes all commands (except *builtins*) in a separate process. Before the process executing a command is resumed, the standard I/O devices are redirected if the user requests this on the shell command line. A shell command otherwise inherits the standard I/O devices of its parent (the shell which started it).

The shell passes exactly two arguments to the command: *nargs* and *args*. You should read Section 11 of "An Introduction to C" if you are not familiar with the UNIX string argument passing convention.

Shell commands are contained in *src\shell* and are named *x_name.c* (where *name* represents the name of your new command). After you have coded a new command you must do a *makemake* in the *src\shell* directory followed by a *make*. You then switch back to the *main* directory and do a *make*. Then you start Xinu and your new command should appear on the help screen and should be executable.

Hardware Requirements

Xinu 7.9 has extensive TCP/IP support and thus requires a network adapter in order to run. ETHERNET device drivers for the WD8003E and 3C503 adapters are provided. The distribution software has been configured for 3C503 using Int 5. Memory mapping of on-board RAM has been disabled. Xinu 7.9 currently supports a serial device on COM2:. The device can be reconfigured to use COM1: if required.

A cut-down version of Xinu with no networking support is available from csc.canberra.edu.au in /pub/ise/xinu/cs2. This version will not be updated in the future.

References

Comer & Fossum: Operating System Design: The Xinu Approach, PC-Edition
Prentice-Hall 1988 ISBN 0-13-638180-4

Douglas E. Comer: Operating System Design: Vol. 2 Internetworking with Xinu
Prentice-Hall 1987 ISBN 0-13-637414-X

Comer & Stevens: Internetworking with TCP/IP Vol. 2
Prentice-Hall 1991 ISBN 0-13-465378-5

Bug Reports

Please send all bug reports to chrisc@ise.canberra.edu.au.

This page is empty.