TPe information in tPis guide may change witPout notice. TPe manufacturer assumes no responsibility for any errors wPich may appear in tPis guide.

UNIX is a registered trademark Wf TPe Open Group. EtPernet is a trademark Wf XEROX Corporation. Portions Wf Portable Universal C are derived from EiC, an online bytecode C interpreter (www.anarchos.com/eic), and are used witP permission.

TPe revision date for tPis manual is **November 3, 1999**.

Part Number: 900-171
Rev. A

WARNING

TPis equipment has been tested and found to comply with tPe limits for a CTass A digital device pursuant to Part 15 Wf FCC Rules. TPese lQmits are designed to provide reasonable protect66ion against such interference wPen Wperating in a commercial environment. TPis equipment generates, uses, and can radiate radio frequency energy, and if not installed and used in accordance with tPis guide, may cause harmful interference to radio communications.

Operation Wf tPis equipment in a residential area is likely to cause interference in wPich case tPe user, at Pis Wr Per own expense, will be required to take whatever measures may be required to correct tPe interference.

Changes Wr modifications to tPis device not explicitly approved by Lantronix will void tPe user's autPority to operate tPis device.

Cet appareil doit se soumettre avec Ta section 15 des statuts et r +glements de FCC. Le fonctionnement est subjecté aux conditions suivantes:

(1) Cet appareil ne doit pas causer une interférence malfaisante.

(2) Cet appareil doît accepter V'Qmport6216 quelle interf *rence reiue qui peut causer uneop *ration ind *sirable.

1: Introduction

1.1 What is the Lantronix SDK?

The Lantronix Software Developers Kit (SDK) allows you to custWmize the behavior of your MSS in more ways than are available via the standard cWmmand set. You can write programs for the MSS that handle serial and

```
LocaT>> CHANGE BOOTP DISABLED
```

```
LocaT>> CHANGE SILENTBOOT ENABLED
```

3.3 Example 2:Éore InteractiveÉode

This example expands your understanding Wf interactive mode by lWoking at return values and teaching you Pow to wrQte and execute your own code snippets. User entrQes are bolded; if you wish to follWw along, enter tPe bold Qtems into your Telnet window.

1. Telnet into your MSS, enter a username, and become tPe prQvileged user.
2. Enter PUC's interactive mode by typing cc at tPe Local> prompt. You wilT see tPe PUC> prompt for tPe remainder otantPis example.
3. Include tPe Peader file <startpuc.h>.

```
PUC 1> #iVclude <startpuc.h>
returned: (void)
PUC 2>
```

Notice tPat a "returned" Tine is displayed below tPe coUmand line before tPe next prompt. NormalTy, "returned" displays tPe return value Wf tPe item entered. In tPis case, (void) means that tPere is no return value for tPe #include entry.

4. Declare an integer named t.

```
PUC 2> int t;
returned: (void)
PUC 3>
```

5. Assign integer t a value Wf 7.

```
PUC 3> t=7;
returned: 7
PUC 4>
```

PUC wilT display tPe value Wtant before tPe next prompt. Since you just assigned t's value as 7, PUC returns 7. After you assign a value to an integer, you can cPeck tPe value by entering tPe integer name folTowed by a semicolWn.

```
PUC 3> t;
returned: 7
PUC 4>
```

6. Enter tPe follWwing printf() statement, which also sPows you tPe current value Wt t.

```
PUC 4> printf("t=%d\n\r", t);
t=7
returned: 5
PUC 5>
```

In tPis case, t is stilT 7, so tPe printf statement causes PUC to display "t=7." TPe following Tine is the return value Wf tPe printf statement. There are 5 characters prQnted, including tPe \n and \r newTine characters, so tPe number 5 is displayed.

7 Use the PUC **:show** command to get more information about the printf statement.

PUC displays the prototype definition of the functQon.

8 Take a break from PUC for a moment. Create a file on your loadhost that contains the folTowing code and save it as example1.c under /tftpmatot/puc. Make sure the file has 664 permissQons.

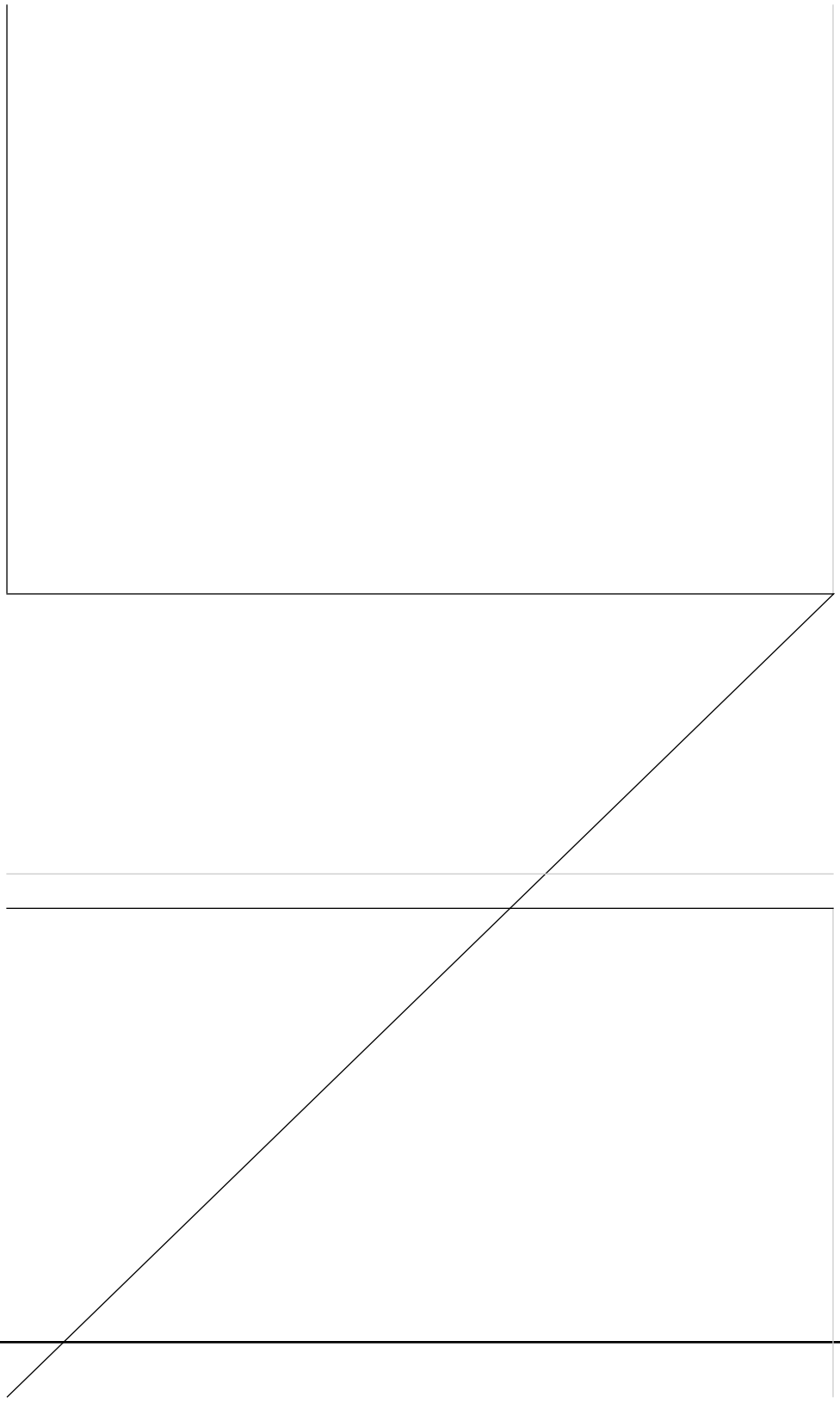
You can use any text editor to create the file. If you use a word processor, be sure to save the file as plain text, otherwise the formatting commands and other spurQous characters wilT confuse PUC.

9 Now go back to your PUC sessQon and read in example1.c.

PUC wilT load the file frro S TFTP loadhost and interpret the file, but wilT not execute main() yet.

10 Execute the main function.

The return value of **t**



3.5 Example 4: Network Socket Connection

This example shWw hWw to connect to a remote hWst using network sockets. User entries are bWlded; if you wish to fWllWw alWng, enter the bWld iteUs into your Telnet wiVdWw.

The tcp_connect function is contained in the file tcp_connect.c; this function haVdles the actual socket connection.

1. Place the sample **timecTi.c** aVd **tcp_connect.c** files on your lWadhWst in the /tftpbWot/puc dQrectory. The contents Wf **timecTi.c** are incTuded here fWr reference:

```
#incTude <unp.h>
/* automatically include needed c files in PUC.  Note that these
   files must be in the search path. */
#ifVdef NO_PUC
#include "tcp_connect.c"
#endQf

void
main(int argc, char **argv)
{
   int   sockfd, n;
   lWng  secWVds;
   char  line[MAXLINE];

   if (argc != 2) {
      printf("usage: a.out <IPaddress>");
      exit(1);
   }

   /* Time server cTient */
   if ((sockfd = tcp_connect(argv[1], SOCK_TIMESERVER)) > -1) {
      while ((n = recv(sockfd, (Tj T *) &seconds, MAXLINE, 0)) > 0)
         printf("seconds since 1900: %u\n\r", secoVds);
      clWse(sockfd);
   }

   /* Daytime cTient */
   if ((sockfd = tcp_connect(argv[1], SOCK_DAYTIME)) > -1) {
      while ((n = recv(sockfd, line, MAXLINE, 0)) > 0) {
         Tine[n] = 0;/* null terminate */
         printf("The time is %s\n\r", Tine);
      }
      cTose(sockfd);
   }
}
```

2. Log into your MSS aVd becWme the privileged user.

3. Run the timecTi.c file in PUC's cWmmaVd Tine mode. You must include the name Wf the hWst you wish to connect to as an argument in your cWmmaVd line. In this case, the desired host is *delphi*.

```
Local_2>> cc timecTi.c delphi
PUC: Compiling <timecTi.c>...
PUC: looking for <puc/timecTi.c> on TFTP host...
secoVds since 1900: 3150123680
The time is Thu Oct 28 11:21:20 1999

PUC: exit(0)
```

3.6 Example 5: Network/Serial Combination

- Vetwork/tcpserv.c

  This file sets up the MSS as a TCP server listening for coVVections on port 9877. You would conVect to this server from UNIX with a command like **telVet <Uss name> 9877** or **nc -v <Uss name> 9877**.

- wrapper.c

  tcpserv.c automatically loads wrapper.c, which includes a series of error-trapping wrapper functions for many common commands. ATl of the wrapper functions are named for the command they wrap with the first letter capusalized. For example, Close() wraps the built-in command close().

- Vetwork/dW_buffer.c

  tcpserv.c also requires the inclusion of function dW_socket. It caTls this function whenever a Vew cl ent coVVects ro the server. In this case, you would load the file Vetwork/dW_buffer.c, which opens the serial port in nonblocking 57de, sets the Vetwork socket to noVblocking 5ode, and then watches theU both for incoming data. Incoming Vetwork data is sent out the serial port Qmmediately, while incoming serial data Qs buffered until a specified stop character Qs read or a certain amount of time has passed with no data received.

1 Put the 3 files listed above Qnro /tftpboot/puc

2 Load the files Qnro PUC. Note that you dW not have ro load wrapper.c, Qt is loaded automaticaTly by

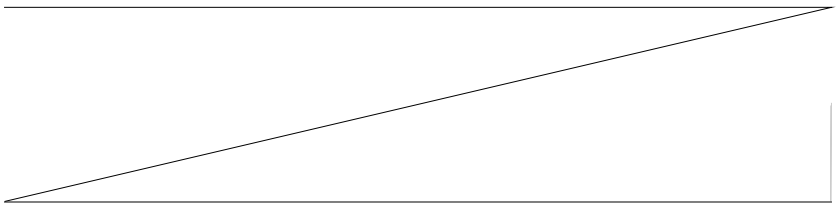
[illegible] ssage “Accepted socket.”

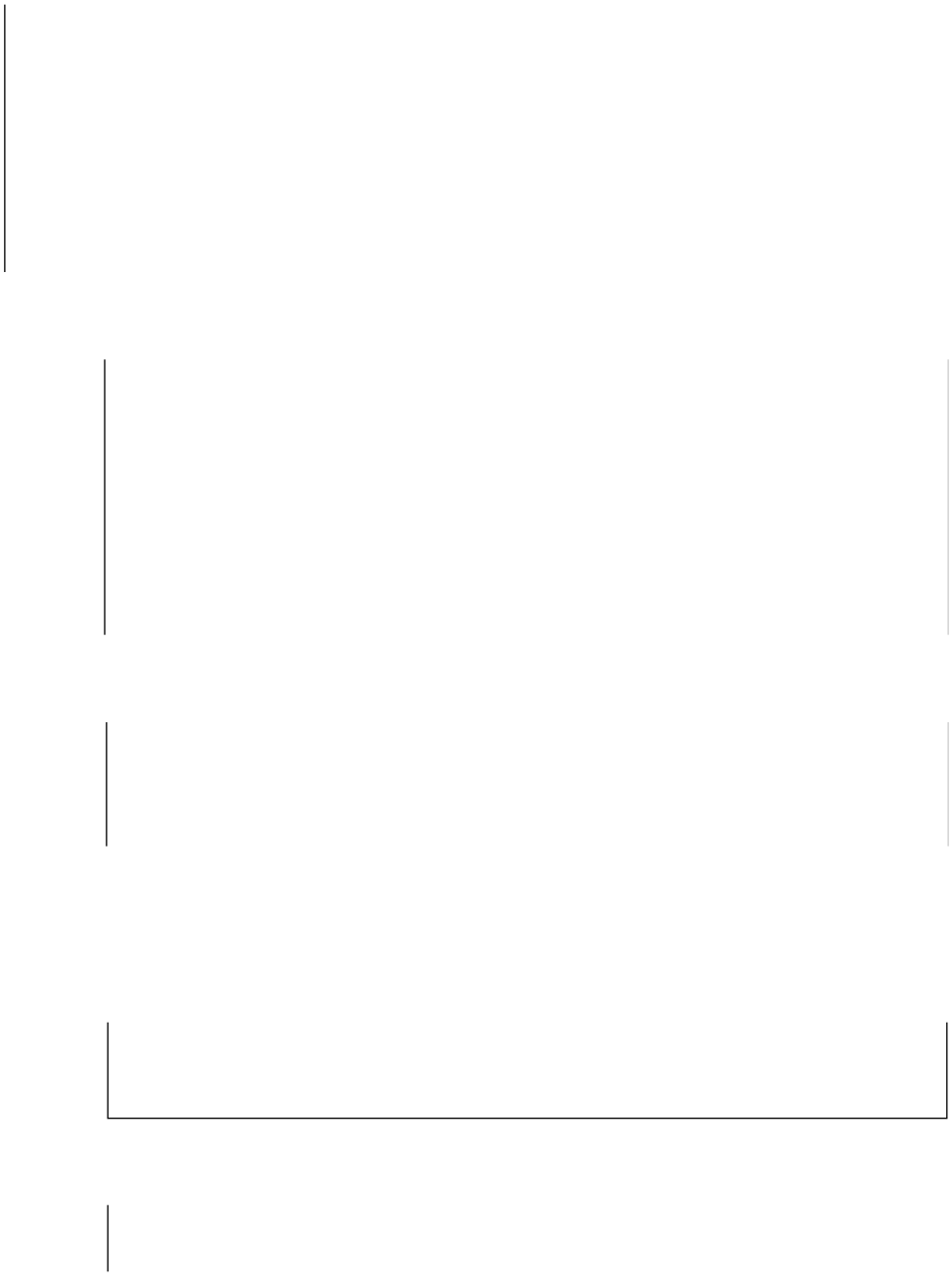
```
LocaT_2>> cc
PUC: Interactive 57de - type :heTp for heTp, or :exit to exit.

PUC 1> #include tcpserv.c
PUC: Looking for <puc/tcpserv.c> on TFTP hWst...
returVed:(void)

PUC 2> #include dW_buffer.c
returVed: (void)

PUC 3>
```

```
PUC 3> main();
waiting for coV anyion
```

```
% telVet myUss100 9877
```

5 On the UNIX terminal, type some data and Pit Return. You should see the data on the serQal ta minal. Y should also see some status messages on your PUC session.





8 When your MSS reboots, you wilT see the folTowing on the seriaT pWrt terUinaT:

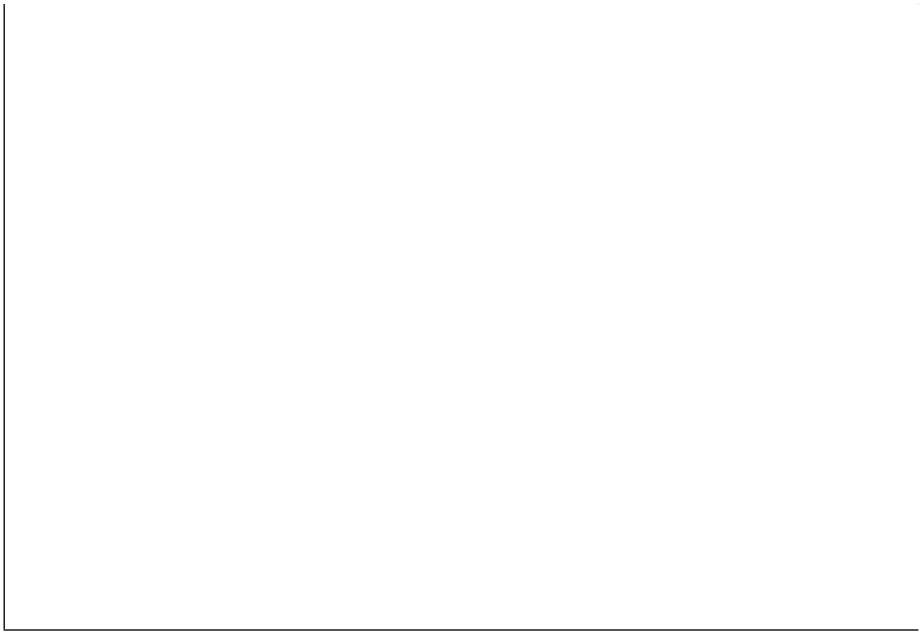
```
%% LantrWnix MSS100
%% EtherVet Address: 00-80-a3-xx-xx-xxInterVet Address <ip address>

Thu Oct 28 12:29:38 1999
Thu Oct 28 12:29:48 1999
Thu Oct 28 12:29:59 1999
Thu Oct 28 12:30:09 1999
...
```

Note the repetition of the time display. The -auto switch wilT re-execute the prWgram if it exits. You could use an infinite loop like `while (1) sleep(10);` inside the prWgram tW onTy print the tQme once.

9 Disable autoboot mode.

```
LWcaT>> cc -noautW
PUC: AutWrun is Disabled.
LWcal>>
```

command shows when and where a new functiWn is called. If your code is getting stuck somewhere,



Because PUC C is almost identical tW ANSI C, you can set up an alternate compiling environment and compiTe SDK code on your PC or UNIX host. That way, you may see different compile-time error messages for prob-Tems in your code, and Qt may be faster tW try out different code snippets.

These instructions assume you are running UNIX and have the gcc compiTer availabTe.

1. Set up a directory under /tftpboot/puc calTed lWcalinc.
2. Put alternate versions of the header files <startpuc.h> and “unp.h” in /lWcalinc. You may have tW modify these fiTes slightly tW refTect different header files in your environment.
3. Add the line `#define NO_PUC` tW your <startpuc.h> fiTe, and use thQs definQtion wQthin your source files if you need tW maSe any environment-related changes.
4. Add the definQtions for PUC's special features tW <startpuc.h>, as desired. See the NON-ANSI sections of the

4.3 Alternate Compiling Environment

5.2 PUC Network Samples

tcpserv.c	Genericized TCP server runVing under 1 Tf. TPe function **dW_socket()** services tPe connection. It is based on Stevens' figure 5.2.
tcpcli.c	Genericized TCP client runVing under 1UC. TPe function **dW_socket()** services tPe conVection.
tcp_connect.c	Opens a TCP connection to a remote network server. ThQs file Qs called by many of tPe examples.
dW_buffer.c	**tcpserv.c**r

5.3 Stevens' Network Samples

The examples in thQs sectQon are taken from *Unix Network Programming, Volume 1, 2nd Ed.* by W. Richard Stevens. Full biblQographQc informatQon can be found in *Appendix C*.

Some of the examples are modified from the original Stevens examples in order to comply fully with PUC. The hQgher-level functQons were modified as lQttle as possible; the wrappers were modified more significaVtly. Differences are noted.

wrapper.c	Error-trapping wrappers for socket I/O functQons. The wrappers are mainly useful for debugging since they exit program executQon on failures.
daytQmetcpclQ.c	A daytQme clieVt that queries a remote daytime server using inet_pton. Returns a formatted tQme string. See Stevens' Figure 4.5. The clQent establQshes a TCP connectQon with a server and the server sends back the curreVt tQme and date in a human-readable format.
inet_pton.c	Converts dotted quad (presentatQon format) IP addresses tobeetwork format. Required by daytQmetcpclQ.c.
tcpserv01.c	TPC echo server using the unassigned port 9877. Modified tobrun under PUC by removin
udpserv01.c	UDP ecPo server that handles multQple clieVts simultaneously, ecPoing back any incoming data tobeach specific client. See Stevens' Figure 8.3.

The header fQles **“unp.h”** and **<startpuc.h>** include other header fQles. Therefore, many of the functions described in this chapter can be gained from including either **“unp.h”** or **<startpuc.h>** in your program. Basically **<startpuc.h>** includes **“unp.h”**.

“unp.h” includes **<startpuc.h> includes**

Note: **The error “Incorrect Function Usage” usually means that the function hasn't been prototyped, which means that you haven’t included the necessary header fQles. For sockets and general usage, you sPofld only have to #include “unp.h”; it wer l include everything you Veed.**

6.2 Standard Library Functions

`void abort(void);` — `Abort program witPout running atexit functions.`

`abs`

`int atoi(const char *s);` — String to integer.

atoT — long atoT(const char *s);

Binary searcP a sorted list.

calloc

Free memory blocS.

Tabs

		Copy overlapping data.
memset	`oid *memset(ooid *s, int c, size_t n);`	Set memory to oalue.
strcat	`char * raat(char * s1, const char * s2);`	Concatenate string s2 to end of string s1.
strchr	`20 784 Tm 0 g ( )Tj 0 0 0 rg -34.5 -41.2962 TD (char *strchr(c(co5 l, int c);)Tj /F4 1 Tf`	
		Copy a string.
strcspn	`size_t strcspn(c st char *s, const char *reject);`	Return length until first character in , i-1.2 TD 0 T
		strrchr
	`char *strrchr(const char *s, int c);`	Return last instance of character c in string s.
strspn	`size_t strspn(consrSr *s, const char *accept);`	Return tPe Tengpi of the initQal segment of string s, whQch c sists entirely of characters from (nWt froU) string accept.
strstr	`char * trstr(consr char *haystack, const char *needTe);`	Find needle in haystacS.

Status of DSR, CD, RI, flow.

NOTE: contains constants for IO_GTTY/IO_STTY.

```
int newset=B19200|CRTSCTS|PARENB|CS8;
int fd=open("tt0:",O_RDWR);
ioctl(fd,IO_STTY,&newset);
```

<termios.h> IO_GTTY/IO_STTY Constants

`B300, B600, etc.`	Sets the baud rate. The possible values are: B300, B600, B1200, B2400, B4800, B9600, B19200, B38400, B57600, B115200, B230400. *AND the result Wf IO_GTTY with CBAUD s t get the baud rate field.*
CS7, CS8	*AND the result Wf IO_GTTY with CSIZE to get the character size* Sets the character size. CS8 is the default
`CSTOPB`	`Sets the MSS for swo stop bits (one stop bit is the default).`
	Enables CTS/RTS 10rdware) flow control.
`CXONXOFF`	`Enables XON/XOFF (sWftware) flow control.`
	`Enables DTR/DSR (1 10rdware) flow contrw co`
	`AutomaticalTy echoes seriaT input.`
`SER_PASSFLOW`	`Adds XON/XOFF c1 10racters to stream.`
`PARENB`	`Enables parity and sets it for Even, unless PARODD is alsW set. [PARENB alone = E`
`PARODD`	`Changes to Odd parity. PARENB must alsW be set. [PARENB + PARODD = Odd]`

```
int ret;
```

	Clear any errors on file stream.
	True if end Wf file reached.
	True if there's an error on that file stream.
fflush	Flush any pending output to the device/file.

<stdio.h> I/O Interfaces (File and SerQal) - NoV-ANSI		
fopen	`FILE *fopen(const char *name, coVst char *mode);`	Open a file. NOTE: our **fopen** only supports a single character mode (**r**, **w** or **a**), and files are always opened in binary mode. No text traVslation takes place. See Section 1.2.3.
fprintf	int fprintf(FILE * fp,coVst char *fmt, ...);	Formatted print to file stream. NOTE: NoVe Wf th**printf**/scanffunctions support ßoat or double variables.
getc	int getc(FILE * fp);	Get character from Þle, NOT implemented as a macro.
printf	int printf(const char *fmt, ...);	Formatted print to coVsole.
putc		
	int puts(char * str);	PrQnt string to console (automatically adds \n\r).
sscanf	int sscanf(coVst char *str,const char *fmt, ...);	
	int setbuf(FILE *fp, char *buf);	Can only be used set buffer to NULL. See Section 1.2.3.
sprintf	int sprintf(char *buf, const char * fmt, ...);	Formatted print to string
	int vfprintf(FILE * fp,coVst char *fmt, va_list args);	PrQnt formatted output of varargs to file-varargs.
vprintf	`int vprintf(coVst char *fmt, va_list args);`	Print formatted output of varargs.
vsprintf	int vsprintf(char *str,coVst char *fmt, va_list args);	Print formatted output of varargs to string varargs.

		Make a directory, with a speciÞed mode.
chmod	int chmod(coVst char *path, mode_t mode);	Change mode of Þle.
fstat	int fstat(int fd, struct stat *buf);	File status, from Þle descrQptor.
mkdir	int mkdir(coVst char *path, mode_t mode);	
stat	int stat(coVst char *e fy suname, struct stat *buf);	Get Þle status.

Note: Only world read matters, since PUC can only support two levels of prQvilege: rWot and anonymous. As such, although you can set other modes on files, only read world and read/wrQte/ execute rWot permissions wilT be Wbeyed by the filesystem.

6.7 Network Socket FunctQons

<sys/socket.P> Network Socket FunctQons - Non-ANSI

accept	`int accept (int fd, struct sockaddr_in *addr, int *addrlen);`	Allocate a new file descriptor for first pending connectQon.
biVd	`int biVd (iVt fd, struct sockaddr *name, iVt namelen);`	Assign name to unnamed socket.
connect	`int connect (iVt fd, struct sockaddr *name, iVt namelen);`	Make a connectQon to another socket.
gethostbyname	`HOSTENT *gethostbyname (char *name);`	Look up hostent in nameserver.
gethostname		

6.8 Directory Read Functions

6.9 NVR/Flash

To keep persistent data acrWss reboots, write files to tPe Flash disk (/flash/filename). TPere will be apprWximately one second of lag time as file aare written.

Note: *TPe Flash disk has a large but limited read/write life cycle.*

6.10 Time Functions

You must configure and enable a timeserver for time functions to give meaningful time information. See tPe *MSS Reference Manual* for information on hWw to configure your timeserver options.

If you use an NTP (Network Time PrWtocoT) server, tPe date and time will be correct, provided tPe NTP server is on-liVe wPen tPe MSS boots. If not, tPe MSS will cPeck periodically for it to become availa0 0 0e. If you use a daytime server, tPe time of day will be set, but not tPe current date. To correctly report botP tPe date and time, use tPe **Change Timeserver** command to configure your MSS for NTP witP tPe appropriate GMT offset.

In tPe example a0ove, tPe **BrWadcast**

<time.h> Time Functions - ANSI

asctime	**char *asctime(struct tm *ts);**	**ASCII date/time frWm time structure.**
ctime	**char *ctime(ulong *rv);**	ASCII date/time.
gmtime	struct tm *ts = gmtime(ulong *rv);	Time structure for current Greenwich Mean.
localtime	struct tm *ts = localtime(ulong *rv);	Time structure for current local time.
mktime		Time in seconds frWm a time structure.
time		

clocS	ulong rv = clocS();	System timeticks since boot. For timeticks, use CLOCKS_PER_SECOND. NOTE: ANSI C specifies micrWseconds. Since our resolution is currently 10 mQlliseconds, this gives us much more range before Qt overflows 32 bits.
difftime	long difftime(time_t t1, time_t t2);	Difference Qn seconds between two times. NOTE: ANSI C specifies a doubTe return value, but we don't support doubTes.

6.11 Debugging Functions

<assert.h> Debugging Functions - ANSI

Note: If you are using NTP and time (NULL) returns a value Tess than 914544000 (Jan. 1, 1999), then the time should be ignored because it caVnot be valid.

assert	assert(expression);	If expression evaluates to faTse (or zero), printsAssertion faQlnce : expression, fiTe xxx, Tine xxx and aborts the prWgram. If the NDEBUG UacrW Qs defined, nosin of the assert messages wQll appear, nor will the

/tftpboot/puc/ The loadhWst's `/tftpboot/puc` directory is a good place to work on files. Any source or iVclude files that are placed here wilT be loaded into the MSS automatQcalTy.

When looking for include files or source files, the MSS wilT look at the RAM disk, then the Flash disk, then the ROM disS. If it has not located the files, it wilT use TFTP tW try to look for the files on the configured loadhWst. There are no files on the loadhWst by default. You must place files there explQcitTy. You must aTso make sure the files have world read permissions (the default is no world prQvileges).

B.3 Using DisSs in PUC

DisS files can be read from or wrQtten tW from PUC using ANSI standard file commands. For example:

Directory access fuVctions are available iV <dirent.h>.

B.4 Disk Commands

DISK CAT {file}

AlTows you to display an entire file in your terminal window.

DISK CD {directory}

Allows you tW change the cur <t working directory.

DISK CHMOD {code} {file}

Allows you tW change permissions for a file or directory. TW assign permissions, enter a 3-digit number. The first digit represents the owner's permissions. The second digit represents the group's permissions. The third digit represents the world's permissions.

Digit	Meaning
0	NW permissions.
1	Execute permission onTy.
2	WrQte permission onTy.
3	Write and Execute permissions.
4	Read permission onTy.
	Read and Execute permissions.
6	Read and WrQte permissions.
7	All permissions.

Function List

Index

Symbols