CO-IDRIS

INSTALLATION GUIDE

EDITION 2.37

AUGUST 1985

Copyright (c) 1984, 1985

by Whitesmiths, Ltd.

All rights reserved.

TABLE OF CONTENTS

I. INTRODUCTION

CO-IDRIS (CI) is a Unix-like operating system developed by Whitesmiths, Ltd. for personal computers (PCs) utilizing the Intel 8086 processor family. It is based on the portable PDP-11 and MC68000 versions of IDRIS currently used in a variety of software development and embedded application systems. The CI operating system is implemented as an MS/PC-DOS application program. This implementation allows the CI user to alternate between the DOS environment and the IDRIS environment without rebooting, and to communicate extensively between the two systems.

CI consists of the IDRIS resident program and a collection of IDRIS application programs, called utilities, that execute under control of the resident. Whitesmiths' C and Pascal compilers are also included in this package, along with libraries which allow for the creation of programs that can execute under CI, CP/M-86 or MS/PC-DOS.

In order to install and operate this CI package, you must provide the following:

1) a PC of a type described in the Installation Supplements (Section X) with DOS version 2.0 or equivalent

2) a fixed disk (sometimes called a Winchester or hard disk)

3) at least one 5 1/4" diskette drive (or a 3 1/2" drive for HP)

Your CO-IDRIS program package is contained on six double-sided, double density, 5 1/4" diskettes; or on eight single sided 3 1/2" diskettes for the HP-150. The diskettes are labelled Disk 1 through Disk 6 (or Disk 8 for HP). There are write protect tabs on the diskettes which you should not remove. The following manuals are included with the CI package:

1) CO-IDRIS Installation Guide (the manual you are reading now)

2) How to Use IDRIS

3) IDRIS Users' Manual

4) IDRIS Programmers' Manual

5) C Programmers' Manual

6) Pascal Programmers' Manual

7) C Interface Manual for 8086

8) Ctext Users' Manual

You should check your program package to ensure that all diskettes and manuals have been included. Before you proceed with the installation, you should read Sections II, III and X of this manual. There are different versions of Section X for different PCs, so be careful to identify and use the version appropriate for your PC. The remaining sections of this manual and all the other manuals can be safely ignored until the installation is complete.

Whitesmiths has also packaged an optional "sysgen" (system generation) kit for CO-IDRIS. The sysgen kit is available as an add-on product to the basic CO-IDRIS package. It includes the source code for most of the CI device drivers, as well as sources for the tailorable components of the IDRIS resident. An _IDRIS Interface Manual for 8086_ is included with the sysgen kit to allow users to build and/or reconfigure their CI resident for special hardware or application needs. Please contact Whitesmiths, Ltd. for further information on the sysgen kit.

II. PREPARING FOR INSTALLATION

After verifying the contents of your package, you should make a backup copy of your CI diskettes. The procedure for doing this is described in the subsection titled Backing up your CI diskettes in the Installation Supplement (Section X) that pertains to your specific PC.

During the installation procedure (Section III), you will be copying CI from the diskettes to the fixed disk. Before that can be done, you must consider your requirements for fixed disk space. To install and operate CI you will need a minimum of two distinct disk areas, called filesystems:

1) a DOS filesystem to contain the CI resident (idris.com) executable file, temporary files which will be used during the verification procedure (Section IV), and any other files you wish to retain for use under DOS.

2) a CI root filesystem to contain CI utility programs, temporary files used during their execution, and user files which you create.

If you optionally want CI to swap programs between main memory and disk (useful when running more programs than can fit in main memory at one time), you will need a third disk area for swap space. The swap area is not part of the previously mentioned filesystems; it is a separate contiguous area on disk used exclusively by the CI resident. For this implementation of CI, the swap area is normally appended to the CI root filesystem on the same physical disk.

The fixed disk storage requirements for CI are summarized as follows:

Disk Area: Component	Description	Size in blocks (512 bytes/block)
DOS filesystem:		
idris.com file	CI resident program	100
temporary files	work space used during verification procedure	10
user files	DOS user space	optional
CI filesystem:		
utilities	CI utility programs	3819
temporary files	work space used during normal CI operation	400
user files	CI user space	variable depending on user needs and disk size
Swap area:		
temporary space	storage area for programs which CI swaps to disk	optional (must be allocated if swapping is desired)

You can use the following guidelines to determine your needs for a swap area:

1) If you allocate a swap area, swapping can be enabled and disabled at your option whenever you load and start CI. If you do not allocate a swap area, you cannot enable swapping.

2) If you only have 128KB of main memory in your PC, you must allocate a swap area and enable swapping during the installation procedure. The swap area should contain at least 512 blocks. Due to performance limitations with that amount of main memory, you should plan to use CI as a single user system.

3) If you have 256KB of main memory, you do not have to allocate a swap area if you are using CI as a single user system. If you are using CI as a multi-user system, you should allocate a swap area of at least 1024 blocks.

4) If you have 512KB or more of main memory, you do not have to allocate a swap area if you are using CI as a single user system. You may not need a swap area even if you plan to use CI as a multi-user system. However, if you ever intend to activate more CI processes than can fit in memory you should allocate a swap area. The area should contain at least 2048 blocks.

DOS and CI can share the fixed disk if it is configured into distinct physical regions, referred to as partitions. There are three partitioning methods, described as follows:

1) The IBM PC and PC's which have an IBM-compatible ROM BIOS allow the fixed disk to be divided into physical partitions. In this case, DOS will have its own partition and the CI root filesystem will have its own partition.

2) PCs such as the DEC Rainbow allow the fixed disk to be divided into multiple logical drives. In this case, DOS will have an entire logical drive, and the CI root filesystem will have another entire logical drive.

3) Other PC's, including the WANG PC and the Olympia, do not support either of the above partitioning methods. On these machines, the CI root filesystem is implemented inside a large file in the DOS filesystem. In this case, care must be exercised so that programs running under DOS do not accidentally damage the CI root filesystem. This method of partitioning may be used for any PC in lieu of methods 1 or 2, and must be used for PC's which do not support methods 1 or 2.

Prior to installing CI, you must create the two partitions, one for DOS and one for CI. As part of the partitioning process you will be determining several important parameters which you will use during the actual installation procedure. These parameters are listed in Table 1. The values of these parameters determine the layout of the CI partition as shown in Figure 1. Since the partitioning process differs among various manufacturers, it is described in the Installation Supplement (Section X). The Installation Supplement will guide you in determining parameter values appropriate for your needs and will direct you to fill in the values in Table 1. You should now read the <u>Disk Partitioning</u> paragraph in the Installation Supplement appropriate for your PC and perform the instructions described there.

Note: After you have installed CI and have become familiar with its operation, you may wish to customize your installation. By customizing, it is possible to set up your CI package to run floppy only, use a separate device for swapping, or use a separate device for pipes, etc. You may also acquire the CI sysgen kit as an add-on to the CI package. The sysgen kit allows you to write your own device drivers, make a standalone IDRIS which doesn't require DOS, and otherwise reconfigure and tailor the CI resident.

Parameter	Description	Value
(1)	root device name	________
(2)	size of root filesystem	________
(3)	size of swap area	________

Table 1 - Installation Parameters

```
        Parameters        Disk layout

           (1)           -------------
                         | RESERVED  |    blocks 0 and 1
                         -------------
                         |           |
                         |           |
                         |           |
           (2)           |   ROOT    |
                         |           |
                         |           |          Partition size
                         |           |
                         -------------
                         |           |
           (3)           |   SWAP    |
                         |           |
                         -------------
```

Definitions:

Parameter (1) is a logical device name (a character string) used by CI to access the root filesystem.

Parameter (2) is the size of the root filesystem in blocks.

Parameter (3) is the size of the swap area in blocks.

For proper operation, the following conditions must be true:

```
    (2) + (3) <= partition size - 2
    size of main memory < (3) if swapping is to be used, or
    (3) = zero to disable swapping
```

NOTES:

The first two blocks of the partition (blocks 0 and 1) are reserved for DOS.
One block = 512 bytes.

Figure 1 - Arrangement of CI partition

III. INSTALLATION

This section describes the steps you should follow to install CI on your system. Some of the steps differ depending on the amount of main memory in your PC. If you only have 128KB of main memory the procedure will direct you to invoke CI with swapping enabled. You must therefore have allocated a swap area on the fixed disk if you only have 128KB. This procedure assumes that a floppy disk drive can be accessed as the **a:** device from DOS. The Installation Supplement will discuss any machine-specific changes that need to be made to this procedure. The amount of time required for various installation steps is given in the text. Your times may vary from the given times depending on your PC's hardware performance.

Lines that you type are highlighted in **boldface.** In these lines, character strings containing angle brackets (like **<root>** or **<fsize>**) should be replaced with values from the Installation Parameters Table. The text will tell you how to make the replacement. The character string **(C/R)** appears at the end of the lines you will be typing and denotes the **C/R** or **ENTER** key. As you are performing the installation, a **#** will appear each time CI is ready to accept a command. If you make a typing mistake on a line beginning with **#**, type **ctl-U** to erase the line, then retype it.

Before beginning the procedure you should ensure that no diskettes are inserted in any of the diskette drives. Then power down or reset your PC as you normally do when it is not in use. You should also have your CI diskettes handy. The procedure will direct you when to insert each diskette. The diskettes will be referred to by their number (e.g., Disk 1, Disk 2, etc.).

1) Start your PC as you normally do and boot your DOS system from the fixed disk.

2) Insert Disk 1 in diskette drive **A.** (Disk 1 is a DOS diskette containing the **idris.com** file.)

3) If you have a dual diskette drive system, insert Disk 2 in drive **B.** (Disk 2 is a CI diskette containing a small root filesystem.) If you do not have a second diskette drive, you will have to wait to insert Disk 2.

4) If you only have 128KB of main memory, wait for the DOS prompt, and then type:

```
a:idris -swap <root> -swapoff <fsize> -swapsize 512 -nbufs 3    (C/R)
```

where **<root>** is the value of parameter 1 and **<fsize>** is the value of parameter 2 that you wrote down in the Installation Parameter Table in Section II. There must be a space between **a:idris, -swap <root>, -swapoff <fsize>,** and **-swapsize 512.** This will load **idris.com** into memory and start its execution. CI will then display its startup banner.

If you have 256KB or more of main memory, wait for the DOS prompt and then type:

a:idris (C/R)

This will load **idris.com** into memory and start its execution. CI will then display its startup banner.

5) If you have a dual diskette system, go to Step 6. If you have only one diskette drive, DOS will issue a message requesting a diskette for drive **B**, so remove Disk 1 and insert Disk 2. Then tell DOS to proceed by striking a key.

6) If you have only 128KB of main memory, wait for the **#** prompt, and then type:

   exec sh -e Install <root> <fsize> (C/R)

   where **<root>** is the value of parameter 1 and **<fsize>** is the value of parameter 2 that you wrote in the Installation Parameter Table in Section II. There must be spaces between **exec, sh, -e, Install, <root>,** and **<fsize>.**

   If you have 256KB or more of main memory then, following the **#** prompt, type:

   sh -e Install <root> <fsize> (C/R)

   where **<root>** is the value of parameter 1 and **<fsize>** is the value of parameter 2 that you wrote in the Installation Parameter Table in Section II. There must be spaces between **sh, -e, Install, <root>,** and **<fsize>.**

   After you type **C/R,** a series of command lines will automatically be displayed on your console followed by the Whitesmiths' copyright notice.

On a dual diskette drive system, the copyright notice will be followed by more command lines until the DOS prompt appears all alone on a line. The total elapsed time should be about seventeen minutes for a 128KB machine and nine minutes for a 256KB or larger machine.

On a single diskette drive system, DOS will issue a series of messages requesting diskettes for drive **A,** then drive **B,** then drive **A,** and finally drive **B** again. Each time a diskette is requested for drive **A,** insert Disk 1 and strike any key to continue. Each time a diskette is requested for drive **B,** insert Disk 2 and strike any key to continue. Command lines should continue to appear until the DOS prompt appears all alone on a line. The total elapsed time should be about the same as for the dual diskette machine plus the additional time to change diskettes.

7) When the DOS prompt appears, remove diskette(s) from drive(s).

8) If you only have 128KB of main memory, you must invoke CI with swapping enabled. Type in the following command to start CI from the fixed disk:

idris -root<root> -nbufs 3 -swapoff<rootsize> -swapsize<swapsize> -swap<root> (C/R)

where **<root>** is the value of parameter 1, **<rootsize>** is the value of parameter 2, and **swapsize>** is the value of parameter 3 from the Installation Parameters Table in Section II. There must be a space preceding each '-' character in the line. This will load **idris.com** from the fixed disk into memory and begin its execution with swapping enabled.

If you have 256KB or more of main memory, invoke CI from the fixed disk by typing:

idris -root <root> (C/R)

where **<root>** is the value of parameter 1 from the Installation Parameters Table in Section II. There must be a space preceding the '-' character in the line. This will load **idris.com** from the fixed disk into memory and begin its execution with swapping disabled.

9) CI will display its startup banner followed by a **#** prompt.

10) If you have a dual diskette system, insert Disk 3 in drive **A**.
If you have only one diskette drive, insert Disk 3 in the drive.

11) Following the **#** prompt, type:

Copy (C/R)

If you only have one diskette drive, DOS will now request a diskette for drive **A**; simply type a C/R in response. A series of file names will automatically appear on the console as this part of the installation is performed. If you only have 128KB of main memory, this will take about nine minutes. If you have 256KB or more of main memory, this will take about five minutes.

12) When the **#** prompt appears, remove Disk 3 and insert Disk 4.
Then type:

Copy (C/R)

A series of file names will automatically appear on the console as this part of the installation is performed. If you only have 128KB of main memory, this will take about ten minutes. If you have 256KB or more of main memory, this will take about five minutes.

13) When the **#** prompt appears, remove Disk 4 and insert Disk 5. Then type:

Copy (C/R)

A series of file names will automatically appear on the console as this part of the installation is performed. If you only have 128KB of main memory, this will take about thirteen minutes. If you have 256KB or more of main memory, this will take about five minutes.

14) When the # prompt appears, remove Disk 5 and insert Disk 6. Then type:

Copy (C/R)

A series of file names will automatically appear on the console as this part of the installation is performed. If you only have 128KB of main memory, this will take about twelve minutes. If you have 256KB or more of main memory, this will take about three minutes.

Note for HP-150 users: You will need to invoke the **Copy** command two more times to complete the installation, since there are eight diskettes in the HP-150 package.

When the # prompt appears, the installation of CI is complete. You should now proceed to Section IV and perform the steps outlined there to verify the installation.

IV. VERIFYING THE INSTALLATION

Immediately following installation and before doing anything else, you should verify that your system operates properly. You are not required to understand CI before doing this. A test suite was included in the package you installed. By using the following procedure, you can invoke the test suite and perform a confidence check on your system.

1) The last line displayed on your terminal should be the # prompt indicating that CI is ready to accept another command. Type in the following command after the prompt:

   ```
   quit   (C/R)
   ```

   This command will terminate CI execution and return you to DOS. The DOS prompt should now be displayed on your terminal.

2) After the DOS prompt, type in the following:

   ```
   dir   (C/R)
   ```

   You should now see a directory listing on your console. One of the files displayed should be **idris.com.** When the directory listing completes, the DOS prompt should reappear.

3) Invoke CI from the fixed disk by typing in a command line identical to the one you used in Step 8 of the Installation Procedure. The CI startup banner should be displayed, followed by a line containing the CI prompt.

4) If you only have 128KB of main memory, the following command may not work on your system due to memory limitations. Go to Step 5 if your PC has only 128KB. Otherwise, type in the following command:

   ```
   dir   (C/R)
   ```

   You should now see the same directory listing that you saw when you ran the **dir** command under DOS in step 2. The CI prompt should reappear on the next line following the directory listing.

5) Type in the following commands:

   ```
   cd /etc/suite   (C/R)
   Isuite   (C/R)
   ```

   The first command changes your working directory so that CI can locate the file named **Isuite. Isuite** is the name of the IDRIS shell script which will exercise most of the utilities included in your program package. As **Isuite** runs, it will display lines on your terminal indicating that "test xx passed". The very last line it displays will indicate whether the overall testing is successful or failed. The length of time required to execute **Isuite** varies among different PCs. When it completes, the CI prompt should reappear on the next line.

6) Type the following command after the prompt:

   Csuite (C/R)

   Csuite is another IDRIS shell script. It will exercise the compilers, assemblers, linkers and object utilities in the CI program package. As **Csuite** runs, it should display lines that appear as "test xx passed". The length of time required to execute **Csuite** varies among different PCs. When it completes, the CI prompt should reappear.

This completes the confidence check on your CI system. Following this verification of your installed system, you should proceed to Section V.

V. USING THE DOCUMENTATION

In the preceding sections of this document, you were given the minimum amount of information and explicit instructions required to install CI. From this point on, you will be deciding how to best use CI to meet your specific needs. To assist you in that effort, this section describes the manuals which you received in your package and provides some pointers on how to use them. The manual you are reading now, the CO-IDRIS Installation Guide, leads you through the installation process and contains information specific to the CI implementation of IDRIS. There are five sections in this document which you have not yet used. Section VI describes the basic CI operating procedures. Section VII describes the files included in the CI program package. Section VIII describes the devices supported by CI. Section IX describes the special commands provided with CI to interact with DOS. Section X is a series of "installation supplements" provided to give you additional information that is specific to your particular hardware.

Unless you are an experienced IDRIS user, the first manual you should read is How to Use IDRIS. It has been specially designed and written to introduce new users to the IDRIS environment. It will take you on a guided tour through the IDRIS system. If you follow the sample exercises described in that manual, you can immediately begin using CI and learning about it at the same time.

The IDRIS Users' Manual contains information of interest to everyone who uses the system. Section I of this manual contains a series of essays on the following topics: logging in, using commands, editing, using files, and advanced command usage. There is also a useful glossary of IDRIS terminology at the end of Section I. Section II begins with a description of conventions governing the design, operation, and documentation of IDRIS utilities. The remainder of Section II contains technical descriptions for each standard utility. These are in alphabetical order by utility name. Section III describes the format of standard files used by the IDRIS resident and utility programs. Section IV contains tutorials on how to administer and maintain your system; technical descriptions for maintenance utilities are included at the end of this section.

The IDRIS Programmers' Manual is intended for the more technically sophisticated users who will be developing programs in the IDRIS environment. Section I contains tutorial descriptions introducing the IDRIS development environment and the tools used to build new programs. The IDRIS system interface is described in Section II. Section III contains descriptions of programming file formats. Section IV contains technical descriptions of library functions used internally by IDRIS utility programs. These functions can be used in new utilities which you develop.

The C Programmers' Manual describes the C programming language and the various library routines that make up the machine-independent C environment. Programming utilities and machine-dependent features of the C support software are described in the C Interface Manual for the 8086. The Pascal Programmers' Manual describes the Pascal to C translator and its associated runtime system.

The ctext Users' Manual describes the ctext formatting system, which is of interest to anyone who needs to process text to produce formatted documents. ctext can assist you in preparing program documentation, letters, memos, or general manuscripts.

VI. OPERATING CO-IDRIS

This section describes several important procedures which you can use in operating CI. These include starting CI, shutting down CI, saving and restoring CI files, using a printer, using the ctext text formatter, and developing your own programs. There is much more that you can do with your CI system than we are able to describe here. The intent of these procedures is to help you get started using CI and to clarify operations unique to this implementation of IDRIS.

Starting CI

Since CI runs as an application under DOS, you must always start DOS first, and then start CI. When you type **idris** following the DOS prompt, DOS reads the **idris.com** file from a DOS filesystem into memory and passes execution control to it. That is the basic procedure for starting CI.

There are several options and parameters you can select when you start CI. These are specified by a list of arguments appearing on the command line. In the IDRIS environment these arguments are called flags. The syntax of the command line used to invoke CI is:

```
idris <flags>
```

Each flag begins with a leading '+' or '-' character. The name of the flag immediately follows the '+' or '-'. Some flags are simply YES/NO indicators; but others must be followed by a value of some kind. Any of the flags can be omitted if their default values are appropriate for your needs.

The following flags are defined for the **idris** command:

-clkint# This flag specifies the interrupt vector address which CI will use for the system clock. You should replace **#** with a decimal value appropriate for your PC. If not specified, the default condition is no clock.

+fpp This flag indicates that an 8087 floating point processor is present in your system and you want application programs to be able to use it. You should not use this flag unless you are sure your system has an 8087. If **+fpp** is not specified, the default is to assume no 8087 is present.

-k# This flag specifies the total amount of memory space available to CI. If none is specified, CI will ask DOS how much memory is available.

-nbufs# This flag specifies the number of blocks (512 bytes each) which CI uses for its disk buffer cache. You should replace **#** with a decimal value appropriate to your needs. The default value is 10. You should never use a value less than 2.

-pipe* This flag specifies the device name which CI uses to locate the pipe device. If not specified, it defaults to the root filesystem device name.

+readonly This flag tells CI to disallow writing to the root filesystem. You should not use this flag unless you are performing maintenance actions on filesystems and want to ensure the integrity of the root filesystem.

-root* This flag specifies the device name which CI uses to locate its root filesystem. You should replace * with the character string you wrote in the Installation Parameter Table for parameter 1. The default value for this flag is **b**.

-swapoff# This flag specifies the start of the swap area relative to the start of the swap device. You should replace # with the decimal value you wrote in the Installation Parameters Table for parameter 2. The default value is 9000. Both **-swapoff** and **-swapsize** must have non-zero values for swapping to occur. If you specify a device other than **<root>** for **-swap**, then replace # with the value 1 to enable swapping.

-swapsize# This flag specifies the size in blocks (512 bytes each) of the swap area. You should replace # with the decimal value you wrote in the Installation Parameters Table for parameter 3. The default value for this flag is 0, which disables swapping.

-swap* This flag specifies the device name which CI uses to locate the device it will use for swapping. This parameter must be specified if you want swapping enabled.

-s This flag tells CI to start silently, i.e. not to display the startup banner or anything else when invoked. This can be useful for embedded systems or other cases when you want CI to be quiet about its operation.

-tinc# This flag specifies the clock frequency. If your clock frequency is between 10 and 300 Hz, you should replace # with a decimal value calculated by the following formula: **# = 614400 / clock frequency in Hz.** If your clock frequency is less than 10 Hz, you should replace # with a negative decimal value corresponding to the actual clock frequency, e.g. -1 for a 1 Hz clock. The default value for **-tinc** is 33746, which is appropriate for an IBM PC.

To illustrate the use of these flags when invoking CI, suppose you have an IBM PC with an 8087, a clock at address **0x8**, a root filesystem at **hdc2**, a swap offset of 9000 blocks, a swap area of 1538 blocks, and you want to use 6 buffers. To start CI you would type the following:

```
idris +fpp -clkint 8 -root hdc2 -swap hdc2 swapsize 1538 -nbufs 6
```

You would not need to use the **tinc** or **swapoff** flags because the default values are correct. To simplify starting CI this way each time, you can place this command line (or any other) in a DOS .BAT file and just enter the name of the .BAT file following the DOS prompt. However, you must be careful to use a file name which doesn't match any .COM or .EXE files in the current DOS directory.

After **idris.com** has been loaded and receives control from DOS, it performs some internal initialization and then locates the file **/odd/init** in its root filesystem. The contents of **/odd/init** determine the remainder of the CI initialization sequence. As described in the "Startup" manual page in Section IV of the IDRIS Users' Manual, **/odd/init** can be linked to one of four IDRIS utility processes:

sh -the standard shell for single user operation;

hsh -the "half shell" for stripped-down operation;

log -the login utility for single user restricted operation; or

multi -the general multi-user startup utility.

The **multi** utility offers the greatest flexibility for system startup since it reads a file, called **/adm/init**, for specific startup commands. As first installed, **/odd/init** is linked to **multi** and the **/adm/init** file sets up a single user system. If you wish to setup a multi-user system, you must create an appropriate **/adm/init** file. The following example shows the contents of **/adm/init** for a multi-user IBM PC:

```
W /usr/bin/dosdate /bin/date
W /etc/mount -it
W /odd/log -boot
m /odd/log console -unum0
m /odd/log com1    -unum1 -s1200 -s300
m /odd/log com2    -unum2 -s1200 -s300
k /bin/sh -i -P"# " -X"¦/etc/¦/bin/¦/etc/bin/¦/usr/bin/" -H"/"
```

Shutting down CI

If you are using CI as a single user system and you have no CI processes active at the time, then shutting down CI is very simple. Just type:

```
# quit
```

If you aren't sure about process states, use the **ps -al** command to check for active processes. If any processes have "run" status, use the **kill -2 1** command to terminate all of them, and then use the **quit** command.

If you are using CI as a multi-user system you should first check with the other users before terminating CI. After coordinating logouts with them you can follow the previous procedure for shutting down a single user system.

Using a printer

NOTE: CI does not support the DOS print spooler. Therefore, you should not use the DOS **print** command when using CI.

There are two ways to use a printer with CI. One works best if you have a serial printer, the other if you have a parallel printer.

If you have a serial printer you do not need to make any modifications to the distribution. Connect the serial printer to the **com1** serial port. The serial port is accessed via the CI device names **/dev/com1.lnk** and **dev/lp.** Since the **lpr** program sends its output to **/dev/lp,** the sequence:

```
        # lpr <myfile
```

will print the document **myfile** on the printer.

If you have a parallel printer you can use the DOS device **PRN** as a destination (sink) for files sent to DOS via the **export** command. For example:

```
        # export -size 1 PRN <myfile
```

You may create a shell script to replace the serial-type driver as follows:

```
        # echo "export -size 1 PRN" >/bin/lpr
```

This creates a file in **/bin** called **lpr** which you can invoke as follows to create a printed listing of **myfile:**

```
        # lpr < myfile
```

lpr will not queue multiple files for printing, so you should not have more than one **lpr** running at any given time.

Saving and Restoring Files

There are several different means by which you can save and restore CI files to and from diskettes. The **dump** and **restor** utilities can be used to save and restore complete filesystems. They are described in the "dump" and "restore" manual pages in Section IV of the IDRIS Users' Manual. The **tp** utility, described on the "tp" manual page in Section II of that manual, can be used to save/restore individual files and directories. There is also a **tar** utility described on the "tar" manual page in Section II, which provides greater flexibility than **tp** and which is the preferred choice for most user level needs.

Document Formatting

Included with CI is a family of programs collectively called ctext (pronounced See-Text) that are used for document formatting. These programs are located in the **/usr/bin** and **/usr/ctext** directories. If you have ANSI.SYS installed into your DOS you can use:

```
    # cth19 <documentname>
```

to get output on your console. If you want to send the output to a parallel printer type:

```
    # ctdbl <documentname> ¦ lpr
```

The processing performed by ctext can be quite complicated and therefore requires a fair amount of CPU time. For more information on the usage of ctext refer to the ctext Users Manual.

Developing Programs

The CI program package includes a Pascal to C translator, a C compiler, an 8086 assembler, a linker, and several object utilities. There are libraries included in object form which can be used to build programs to run under CI, DOS-86, and CP/M-86. Program compilation and linking is performed using a multi-pass compiler driver, called **c**. The compiler driver is described in the "Compile" manual page in Section I of the IDRIS Programmers' Manual. The following commands use the driver to generate programs for various target execution environments:

Command	Description
c	C for CI, no 8087
pc	Pascal for CI, no 8087
cf	C for CI, assumes 8087
pcf	Pascal for CI, assumes 8087
cdos86	C for DOS, no 8087
pdos86	Pascal for DOS, no 8087
cdos86f	C for DOS, assumes 8087
pdos86f	Pascal for DOS, assumes 8087
ccpm86	C for CP/M, no 8087
pcpm86	Pascal for CP/M, no 8087
ccpm86f	C for CP/M, assumes 8087
pcpm86f	Pascal for CP/M, assumes 8087

With the exception of **c**, all of the above commands are implemented as shell scripts which call **c**. They are located in the **/etc/bin** directory. These commands are set up to automatically scan the appropriate libraries for the target execution environments.

The libraries scanned by the compiler driver/scripts are identified as follows:

For CI targeted programs

/lib/libu.86	Portable V7 subset
/lib/libc.86	Portable C, CI system interface, and 8086 machine support (FP emulation, no 8087)
/lib/libp.86	Portable Pascal and Pascal runtime
/lib/libf.86	Portable C, CI system interface, and 8086 machine support (assuming 8087 present)

For DOS targeted programs

/lib/libudos.86	Portable V7 subset
/lib/libcdos.86	Portable C, DOS system interface, and 8086 machine support (FP emulation, no 8087)
/lib/libpdos.86	Portable Pascal and Pascal runtime
/lib/libfdos.86	Portable C, DOS system interface, and 8086 machine support (assuming 8087 present)

For CP/M targeted programs

/lib/libucpm.86	Portable V7 subset
/lib/libccpm.86	Portable C, CP/M system interface, and 8086 machine support (FP emulation, no 8087)
/lib/libpcpm.86	Portable Pascal and Pascal runtime
/lib/libfcpm.86	Portable C, CP/M system interface, and 8086 machine support (assuming 8087 present)

In addition to these libraries, there is an IDRIS support library, called **/lib/libi.86**, which contains functions useful in developing new IDRIS utility programs. The previously mentioned compiler driver/scripts are not pre-built to scan this library. If you wish to use it you can make your own driver scripts by using ours as models. Similarly, if you do not wish to use our portable V7 subset library, you can modify driver scripts to use only the standard WSL libraries.

Documentation on the library functions may be found as follows:

Portable V7 subset	C Programmers' Manual "V7lib" manual page
Portable C	C Programmers' Manual, all of Section II
Portable Pascal	Pascal Programmers' Manual, all of Section II, III
Pascal runtime	Pascal Programmers' Manual, all of Section IV
CI system interface	C Interface Manual for 8086, all of Section III.a
8086 machine support	C Interface Manual for 8086, all of Section IV
DOS and CP/M system interface	C Interface Manual for 8086, all of Section III.b.
CI support library	IDRIS Programmers' Manual, all of Section IV

Development Examples

The following examples describe how to compile and execute programs for a variety of environments. The source programs **hello.c** and **world.p** used in these examples are not included in the CI program package. (The **hello.c** program is described on page 1-8 and the **world.p** program on page 8-6 of the How to Use IDRIS manual. They are used here for illustration purposes only.)

Developing Programs For IDRIS

To compile a C program in the current directory for use under IDRIS type:

```
    # c hello.c
```

and observe the response:

```
    hello.c:
    link:
```

The file named **hello.c** will be compiled into an object module named **hello.o.** **hello.o** will then be linked with the C library for IDRIS and the resulting executable file will be named **xeq** in the current directory. To run the program just type:

```
    # xeq
```

and observe the response:

```
    Hello world!
```

Developing Programs For DOS-86

To compile the same program (**hello.c**) to run under version 2.0 of DOS type:

```
    # cdos86 hello.c
```

and observe the response:

```
    hello.c:
    link:
```

This will build an object module as before and link it with a DOS-specific library. The resulting DOS executable file will be left under the name **xeq** in the current IDRIS directory. To run it type:

```
    # export -b xeq.com <xeq
    # dos xeq
```

and observe the response:

```
    Hello World!
```

The **export** command line will send the binary file out to the current DOS drive under the name **xeq.com.** The **dos** command is then used to run the file in the DOS environment.

Developing Programs For CP/M-86

To compile the same program for running under CP/M-86 type:

```
    # ccpm86 hello.c
```

and observe the response:

```
    hello.c
    link:
```

To convert to CP/M object form type:

```
    # to86 -b1
```

This will leave you with a file called **xeq.cmd** in the current directory that will run under CP/M-86 once moved there. If it is not possible for you to move a binary file to CP/M-86 you can always take **xeq** from the current directory and translate it into Digital Research Hex records. Just type:

```
    # hex -dr >xeq.hex
```

This will create an ASCII file of hex records that can be sent through a serial link to a CP/M system. To convert it back to binary once on CP/M type: (**NOTE**: the 8080 argument to **gencmd** is not an error)

```
    A> gencmd xeq 8080 code[xfff]
    A> xeq
```

and observe the response:

```
    Hello World!
```

Compiling Pascal Programs

To compile a Pascal program in the current directory for use under IDRIS type:

```
    # pc world.p
```

and observe the response:

```
    world.p:
    link:
```

This will compile the Pascal file **world.p** into a object file named **world.o.** It will then link this file with a Pascal library for use under IDRIS leaving the result in the file called **xeq** in the current directory.

Similarly, to compile a Pascal program for use under DOS type:

```
    # pdos86 world.p
```

and observe the response:

```
    world.p:
    link:
```

To run it you will need to "export" it first to DOS, as in the previous examples. Developing Programs To Use 8087 Instructions

To compile a program and embed inline 8087 floating point instructions in it, invoke the compiler driver by the same name as before, only add an **f** on the end of the name. Thus **c** becomes **cf**, **cdos86** becomes **cdos86f**, etc. Type:

```
    # pcpm86f world.p
```

and observe the response:

```
    world.p
    link:
```

This will compile the **world.p** file and insert floating point instructions where needed in the object code. The object will then be linked with a special 8087 startup routine and libraries that also have floating point instructions in them.

Special IDRIS Features

IDRIS on the 8086 supports shared text (instructions) in its executable format. This offers the following advantages:

lower memory requirements - since only one copy of the text exists for all invocations.

faster fork operations - since only the data section need be copied during a fork.

larger program space - since text and data don't have to live in the same segment, each can be up to 64KB in length.

Programs linked with 'text fill' to the nearest block boundary and a data bias of 0 will automatically have their text segment shared among all invocations of the same program. To make **newxeq** a shareable image version of **xeq:**

```
    # link -o newxeq -tf9 -db0 xeq
```

The compiler driver scripts for IDRIS automatically produce shareable images.

It is also possible to make a text portion stay in memory even after all invocations have finished. Programs that are made "sticky" like this offer faster load times on a system with swapping because the program need only be brought in from the swap device. A program to be made "sticky" must be shareable (as above) and have the 01000 bit in the file mode turned on by chmod (i.e. **chmod -01555 newxeq**). Once stuck, a sticky program may be removed from swap memory via a kill signal directed to its process id. The **ps -alt** command may be used to identify shared processes.

VII. CO-IDRIS FILESYSTEM

As a part of the installation process, you created a CI filesystem on your fixed disk. All of the files from your CI package are stored in that filesystem. The CI filesystem has a hierarchical tree structure. The root of the tree is denoted by the symbol /. The branching points of the tree are called directories and the ends of the branches themselves are the files.

The following pages contain a listing of all the files installed in your CI filesystem. For more information on IDRIS filesystems, you should refer to the IDRIS Users' Manual, to the "Filesystem" manual page in Section IV and the "Files" manual page in Section I.

CO-IDRIS Files

```
  1 drwxr-xr-x 10 root         240 Aug 09 13:57 /

/
  1 drwxr-xr-x 10 root         240 Aug 09 13:57 .
  1 drwxr-xr-x 10 root         240 Aug 09 13:57 ..
  1 -r--r--r--  1 root          49 Aug 09 13:57 .login
  1 -rwxr--r--  1 root          37 Aug 09 13:57 Copy
  1 drwxr-xr-x  4 root         208 Aug 10 11:59 adm
  3 drwxr-xr-x  2 root        1088 Aug 10 11:59 bin
  3 drwxr-xr-x  2 root        1088 Aug 09 13:57 dev
  1 drwxr-xr-x  4 root         256 Aug 10 11:59 etc
  1 drwxr-xr-x  3 root         384 Aug 10 11:59 lib
  1 drwxr-xr-x  2 root         192 Aug 10 11:59 odd
  1 drwxrwxrwx  2 root          32 Aug 09 13:57 tmp
  1 drwxrwxrwx  6 root          96 Aug 10 11:59 usr
  0 -rw-r--r--  1 root           0 Aug 09 13:57 x
  0 -rw-r--r--  1 root           0 Aug 09 13:57 y
  0 -rw-r--r--  1 root           0 Aug 09 13:57 z

/adm
  1 drwxr-xr-x  4 root         208 Aug 10 11:59 .
  1 drwxr-xr-x 10 root         240 Aug 09 13:57 ..
  1 -r--r--r--  1 root          78 Aug 09 13:57 .login
  1 -rw-rw-rw-  1 root          29 Aug 10 11:59 date
  1 -rw-r--r--  1 root         109 Aug 09 13:57 init
  1 -rw-r--r--  1 root          52 Aug 10 11:59 log
  1 drwxr-xr-x  2 root          32 Aug 09 13:57 lpd
  1 drwxr-xr-x  2 root          32 Aug 09 13:57 mail
  1 -rw-r--r--  1 root         132 Aug 10 11:59 mount
  1 -rw-r--r--  1 root         121 Aug 09 13:57 passwd
  1 ----------  1 root           9 Aug 09 13:57 salt
  0 -rw-r--r--  1 root           0 Aug 09 13:57 who
  1 -rw-r--r--  1 root          11 Aug 09 13:57 zone

/adm/lpd
  1 drwxr-xr-x  2 root          32 Aug 09 13:57 .
  1 drwxr-xr-x  4 root         208 Aug 10 11:59 ..
/adm/lpd total 1


/adm/mail
  1 drwxr-xr-x  2 root          32 Aug 09 13:57 .
  1 drwxr-xr-x  4 root         208 Aug 10 11:59 ..
/adm/mail total 1

/adm total 13


/bin
  3 drwxr-xr-x  2 root        1088 Aug 10 11:59 .
  1 drwxr-xr-x 10 root         240 Aug 09 13:57 ..
```

```
 8 -r-xr-xr-x  1 root       3834 Aug 09 13:57 cat
14 -r-xr-xr-x  1 root       6576 Aug 09 13:57 chmod
17 -r-xr-xr-x  1 root       8094 Aug 10 11:59 cmds
17 -r-xr-xr-x  1 root       7764 Aug 09 13:57 cmp
22 -r-xr-xr-x  1 root      10606 Aug 10 11:59 comm
19 -r-sr-xr-x  1 root       8920 Aug 09 13:57 cp
14 -r-xr-xr-x  1 root       6518 Aug 10 11:59 crypt
17 -r-xr-xr-x  1 root       7784 Aug 10 11:59 cu
18 -r-xr-xr-x  1 root       8576 Aug 09 13:57 date
16 -r-xr-xr-x  1 root       7378 Aug 09 13:57 dd
18 -r-xr-xr-x  1 root       8618 Aug 10 11:59 deque
14 -r-xr-xr-x  1 root       6508 Aug 10 11:59 detab
18 -r-sr-xr-x  1 root       8666 Aug 10 11:59 df
24 -r-xr-xr-x  1 root      11672 Aug 10 11:59 diff
13 -r-xr-xr-x  1 root       5696 Aug 10 11:59 dn
39 -r-xr-xr-x  1 root      19340 Aug 10 11:59 e
 8 -r-xr-xr-x  1 root       3824 Aug 09 13:57 echo
12 -r-xr-xr-x  1 root       5478 Aug 10 11:59 enque
14 -r-xr-xr-x  1 root       6508 Aug 10 11:59 entab
13 -r-xr-xr-x  1 root       5940 Aug 10 11:59 error
 8 -r-xr-xr-x  1 root       3848 Aug 10 11:59 first
20 -r-xr-xr-x  1 root       9308 Aug 10 11:59 grep
18 -r-xr-xr-x  1 root       8572 Aug 10 11:59 head
10 -r-xr-xr-x  1 root       4488 Aug 10 11:59 kill
14 -r-xr-xr-x  1 root       6596 Aug 10 11:59 last
 8 -r-xr-xr-x  1 root       3982 Aug 09 13:57 ln
 5 -r-sr-xr-x  1 root       2354 Aug 10 11:59 lpr
28 -r-xr-xr-x  1 root      13804 Aug 09 13:57 ls
49 -r-sr-xr-x  1 root      24568 Aug 10 11:59 mail
22 -r-xr-xr-x  1 root      10644 Aug 10 11:59 mc
27 -r-xr-xr-x  1 root      13140 Aug 10 11:59 md
11 -r-xr-xr-x  1 root       4978 Aug 10 11:59 mesg
12 -r-sr-xr-x  1 root       5450 Aug 09 13:57 mkdir
13 -r-sr-xr-x  1 root       5958 Aug 09 13:57 mv
12 -r-xr-xr-x  1 root       5360 Aug 10 11:59 nice
12 -r-xr-xr-x  1 root       5328 Aug 10 11:59 nohup
17 -r-xr-xr-x  1 root       7806 Aug 09 13:57 od
20 -r-xr-xr-x  1 root       9616 Aug 10 11:59 page
23 -r-sr-xr-x  1 root      11110 Aug 10 11:59 passwd
28 -r-xr-xr-x  2 root      13590 Aug 10 11:59 pk
28 -r-xr-xr-x  1 root      13574 Aug 10 11:59 pr
18 -r-xr-xr-x  1 root       8474 Aug 10 11:59 ps
 8 -r-xr-xr-x  1 root       3936 Aug 09 13:57 pwd
28 -r-sr-xr-x  1 root      13414 Aug 09 13:57 rm
32 -r-xr-xr-x  1 root      15776 Aug 10 11:59 roff
13 -r-xr-xr-x  1 root       6090 Aug 09 13:57 setb
36 -r-xr-xr-x  2 root      17544 Aug 09 13:57 sh
10 -r-xr-xr-x  1 root       4338 Aug 10 11:59 sleep
23 -r-xr-xr-x  1 root      10824 Aug 10 11:59 sort
50 -r-xr-xr-x  1 root      24652 Aug 10 11:59 spell
20 -r-xr-xr-x  1 root       9422 Aug 10 11:59 stty
21 -r-sr-xr-x  1 root       9864 Aug 10 11:59 su
 8 -r-xr-xr-x  1 root       3816 Aug 09 13:57 sync
59 -r-sr-xr-x  1 root      29398 Aug 10 11:59 tar
```

```
 13 -r-xr-xr-x  1 root        6004 Aug 10 11:59 tee
  4 -r-xr-xr-x  1 root        1774 Aug 09 13:57 test
 15 -r-xr-xr-x  1 root        6978 Aug 10 11:59 time
 32 -r-sr-xr-x  1 root       15408 Aug 10 11:59 tp
 14 -r-xr-xr-x  1 root        6526 Aug 10 11:59 tr
 22 -r-xr-xr-x  1 root       10556 Aug 10 11:59 uniq
  6 -r-xr-xr-x  1 root        2748 Aug 10 11:59 up
 28 -r-xr-xr-x  2 root       13590 Aug 10 11:59 upk
 16 -r-xr-xr-x  1 root        7516 Aug 10 11:59 wc
 22 -r-xr-xr-x  1 root       10256 Aug 10 11:59 who
 23 -r-xr-xr-x  1 root       10982 Aug 10 11:59 write
 36 -r-xr-xr-x  2 root       17544 Aug 09 13:57 [
/bin total 1280


/dev
  3 drwxr-xr-x  2 root        1088 Aug 09 13:57 .
  1 drwxr-xr-x 10 root         240 Aug 09 13:57 ..
  0 cr--r--r--  1 root     0,    0 Aug 09 13:57 NODEV
  0 brw-r--r--  4 root     1,    0 Aug 09 13:57 a
  0 brw-r--r--  2 root     1,    1 Aug 09 13:57 b
  0 cr--r--r--  1 root     0,    8 Aug 09 13:57 bnames
  0 brw-r--r--  1 root     1,    2 Aug 09 13:57 c
  0 cr--r--r--  1 root     0,    9 Aug 09 13:57 cnames
  0 crw--w--w-  1 root     2,    0 Aug 09 13:57 com1
  0 crw-r--r--  3 root     2,    2 Aug 09 13:57 com1.lnk
  0 crw--w--w-  1 root     2,    1 Aug 09 13:57 com2
  0 crw-r--r--  1 root     2,    3 Aug 09 13:57 com2.lnk
  0 crw--w--w-  1 root     1,    0 Aug 09 13:57 console
  0 brw-r--r--  1 root     1,    3 Aug 09 13:57 d
  0 brw-r--r--  1 root     1,    4 Aug 09 13:57 e
  0 brw-r--r--  1 root     1,    5 Aug 09 13:57 f
  0 brw-r--r--  4 root     1,    0 Aug 09 13:57 floppy0
  0 brw-r--r--  2 root     1,    1 Aug 09 13:57 floppy1
  0 brw-r--r--  1 root     1,    6 Aug 09 13:57 g
  0 brw-r--r--  1 root     1,    7 Aug 09 13:57 h
  0 brw-r--r--  1 root     2,    0 Aug 09 13:57 hdc0
  0 brw-r--r--  1 root     2,    1 Aug 09 13:57 hdc1
  0 brw-r--r--  1 root     2,    2 Aug 09 13:57 hdc2
  0 brw-r--r--  1 root     2,    3 Aug 09 13:57 hdc3
  0 brw-r--r--  1 root     2,    4 Aug 09 13:57 hdc4
  0 brw-r--r--  1 root     2,    8 Aug 09 13:57 hdd0
  0 brw-r--r--  1 root     2,    9 Aug 09 13:57 hdd1
  0 brw-r--r--  1 root     2,   10 Aug 09 13:57 hdd2
  0 brw-r--r--  1 root     2,   11 Aug 09 13:57 hdd3
  0 brw-r--r--  1 root     2,   12 Aug 09 13:57 hdd4
  0 brw-r--r--  1 root     3,    0 Aug 09 13:57 idris.0
  0 brw-r--r--  1 root     3,    1 Aug 09 13:57 idris.1
  0 cr--r--r--  1 root     0,    6 Aug 09 13:57 inode
  0 cr--r--r--  1 root     0,    1 Aug 09 13:57 kmem
  0 crw-r--r--  3 root     2,    2 Aug 09 13:57 lnk0
  0 crw-r--r--  3 root     2,    2 Aug 09 13:57 lp
  0 brw-r--r--  1 root     4,    0 Aug 09 13:57 md0
  0 brw-r--r--  1 root     4,    1 Aug 09 13:57 md1
```

```
   0 cr--r--r--  1 root      0,   5 Aug 09 13:57 mount
   0 brw-r--r--  4 root      1,   0 Aug 09 13:57 mt0
   0 cr--r--r--  1 root      0,   4 Aug 09 13:57 myps
   0 crw-rw-rw-  1 root      0,   2 Aug 09 13:57 null
   0 cr--r--r--  1 root      0,   3 Aug 09 13:57 ps
   0 crw-r--r--  1 root      3,   0 Aug 09 13:57 ra
   0 crw-r--r--  1 root      3,   1 Aug 09 13:57 rb
   0 crw-r--r--  1 root      3,   2 Aug 09 13:57 rc
   0 crw-r--r--  1 root      3,   3 Aug 09 13:57 rd
   0 crw-r--r--  1 root      3,   4 Aug 09 13:57 re
   0 crw-r--r--  1 root      3,   5 Aug 09 13:57 rf
   0 crw-r--r--  1 root      3,   6 Aug 09 13:57 rg
   0 crw-r--r--  1 root      3,   7 Aug 09 13:57 rh
   0 crw-r--r--  1 root      4,   0 Aug 09 13:57 rhdc0
   0 crw-r--r--  1 root      4,   1 Aug 09 13:57 rhdc1
   0 crw-r--r--  1 root      4,   2 Aug 09 13:57 rhdc2
   0 crw-r--r--  1 root      4,   3 Aug 09 13:57 rhdc3
   0 crw-r--r--  1 root      4,   4 Aug 09 13:57 rhdc4
   0 crw-r--r--  1 root      4,   8 Aug 09 13:57 rhdd0
   0 crw-r--r--  1 root      4,   9 Aug 09 13:57 rhdd1
   0 crw-r--r--  1 root      4,  10 Aug 09 13:57 rhdd2
   0 crw-r--r--  1 root      4,  11 Aug 09 13:57 rhdd3
   0 crw-r--r--  1 root      4,  12 Aug 09 13:57 rhdd4
   0 crw-r--r--  1 root      6,   0 Aug 09 13:57 ridris.0
   0 crw-r--r--  1 root      6,   1 Aug 09 13:57 ridris.1
   0 crw-r--r--  1 root      5,   0 Aug 09 13:57 rmd0
   0 crw-r--r--  1 root      5,   1 Aug 09 13:57 rmd1
   0 brw-r--r--  4 root      1,   0 Aug 09 13:57 rmt0
   0 cr--r--r--  1 root      0,   7 Aug 09 13:57 where
   0 crw-r--r--  1 root      0,  10 Aug 09 13:57 zero
/dev total 3


/etc
   1 drwxr-xr-x  4 root        256 Aug 10 11:59 .
   1 drwxr-xr-x 10 root        240 Aug 09 13:57 ..
   1 -r-xr-xr-x  1 root      11024 Aug 09 13:57 badblk
   1 drwxr-xr-x  2 root        432 Aug 10 11:59 bin
  14 -r-xr-xr-x  1 root       6196 Aug 09 13:57 chown
  24 -r-xr-xr-x  1 root      11642 Aug 09 13:57 dcheck
  15 -r-xr-xr-x  1 root       7006 Aug 10 11:59 devs
  31 -r-xr-xr-x  1 root      15126 Aug 10 11:59 dump
  30 -r-xr-xr-x  1 root      14518 Aug 09 13:57 fcheck
  40 -r-xr-xr-x  1 root      19610 Aug 09 13:57 icheck
   8 -r-xr-xr-x  1 root       3966 Aug 09 13:57 mkdev
  15 -r-xr-xr-x  1 root       6790 Aug 09 13:57 mkfs
  24 -r-xr-xr-x  1 root      11302 Aug 09 13:57 mount
  36 -r-xr-xr-x  1 root      17894 Aug 09 13:57 ncheck
   5 -r-xr-xr-x  1 root       2314 Aug 09 13:57 quit
  36 -r-xr-xr-x  1 root      17766 Aug 10 11:59 restor
   1 drwxr-xr-x  2 root        208 Aug 10 11:59 suite

/etc/bin
   1 drwxr-xr-x  2 root        432 Aug 10 11:59 .
  1 drwxr-xr-x  4 root         256 Aug 10 11:59 ..
 44 -r-xr-xr-x  1 root       21702 Aug 10 11:59 as.86
 26 -r-xr-xr-x  1 root       12568 Aug 10 11:59 c
  1 -r-xr-xr-x  1 root         190 Aug 10 11:59 c.proto
  1 -r-xr-xr-x  1 root         209 Aug 10 11:59 ccpm86
  1 -r-xr-xr-x  1 root         228 Aug 10 11:59 ccpm86f
  1 -r-xr-xr-x  1 root         227 Aug 10 11:59 cdos86
  1 -r-xr-xr-x  1 root         246 Aug 10 11:59 cdos86f
  1 -r-xr-xr-x  1 root         231 Aug 10 11:59 cf
 78 -r-xr-xr-x  1 root       39024 Aug 09 13:57 db
 18 -r-xr-xr-x  1 root        8700 Aug 10 11:59 hex
 23 -r-xr-xr-x  1 root       11158 Aug 10 11:59 lib
 30 -r-xr-xr-x  1 root       14640 Aug 09 13:57 link
 14 -r-xr-xr-x  1 root        6204 Aug 10 11:59 lord
  1 -r-xr-xr-x  1 root         236 Aug 10 11:59 pc
  1 -r-xr-xr-x  1 root         256 Aug 10 11:59 pcf
  1 -r-xr-xr-x  1 root         240 Aug 10 11:59 pcpm86
  1 -r-xr-xr-x  1 root         255 Aug 10 11:59 pcpm86f
  1 -r-xr-xr-x  1 root         258 Aug 10 11:59 pdos86
  1 -r-xr-xr-x  1 root         277 Aug 10 11:59 pdos86f
 41 -r-xr-xr-x  1 root       20190 Aug 10 11:59 pp
 31 -r-xr-xr-x  1 root       15358 Aug 10 11:59 prof
 65 -r-xr-xr-x  1 root       32626 Aug 10 11:59 ptc
 29 -r-xr-xr-x  1 root       13848 Aug 10 11:59 rel
 15 -r-xr-xr-x  1 root        6764 Aug 10 11:59 to86
 26 -r-xr-xr-x  1 root       12402 Aug 10 11:59 unhex
/etc/bin total 453


/etc/suite
  1 drwxr-xr-x  2 root         208 Aug 10 11:59 .
  1 drwxr-xr-x  4 root         256 Aug 10 11:59 ..
  6 -rwxr--r--  1 root        2886 Aug 10 11:59 Chkroot
  2 -rwxr--r--  1 root         698 Aug 10 11:59 Csuite
  2 -rwxr--r--  1 root         686 Aug 10 11:59 Isuite
  4 -rw-r--r--  1 root        1546 Aug 10 11:59 cfiles
  5 -rwxr--r--  1 root        2427 Aug 10 11:59 cpgm
  3 -rwxr--r--  1 root        1064 Aug 10 11:59 crosspgm
  2 -rwxr--r--  1 root         841 Aug 10 11:59 minimal
  4 -rwxr--r--  1 root        1827 Aug 10 11:59 paspgm
  3 -rwxr--r--  1 root        1048 Aug 10 11:59 scmd
  6 -rwxr--r--  1 root        2867 Aug 10 11:59 tools
  3 -rwxr--r--  1 root        1294 Aug 10 11:59 ucmd
/etc/suite total 41

/etc total 775


/lib
  1 drwxr-xr-x  3 root         384 Aug 10 11:59 .
  1 drwxr-xr-x 10 root         240 Aug 09 13:57 ..
  1 -r--r--r--  1 root         400 Aug 10 11:59 Crtcpm.86
  2 -r--r--r--  1 root         521 Aug 10 11:59 Crtdos.86
  1 -r--r--r--  1 root          63 Aug 10 11:59 Crtf.86
  1 -r--r--r--  2 root       30 Aug 10 11:59 Crtfcpm.86
  1 -r--r--r--  2 root       30 Aug 10 11:59 Crtfdos.86
  1 -r--r--r--  1 root      274 Aug 10 11:59 Crtp.86
  1 -r--r--r--  1 root      395 Aug 10 11:59 Crts.86
  4 -r--r--r--  1 root     1564 Aug 10 11:59 dmain.o
  1 drwxr-xr-x  2 root      336 Aug 10 11:59 hdrs
 94 -r--r--r--  1 root    47294 Aug 10 11:59 libc.86
 94 -r--r--r--  2 root    47165 Aug 10 11:59 libccpm.86
 94 -r--r--r--  2 root    47165 Aug 10 11:59 libcdos.86
 85 -r--r--r--  1 root    42983 Aug 10 11:59 libf.86
 85 -r--r--r--  2 root    42854 Aug 10 11:59 libfcpm.86
 85 -r--r--r--  2 root    42854 Aug 10 11:59 libfdos.86
 38 -r--r--r--  1 root    18817 Aug 10 11:59 libi.86
 26 -r--r--r--  3 root    12366 Aug 10 11:59 libp.86
 26 -r--r--r--  3 root    12366 Aug 10 11:59 libpcpm.86
 26 -r--r--r--  3 root    12366 Aug 10 11:59 libpdos.86
 28 -r--r--r--  3 root    13566 Aug 10 11:59 libu.86
 28 -r--r--r--  3 root    13566 Aug 10 11:59 libucpm.86
 28 -r--r--r--  3 root    13566 Aug 10 11:59 libudos.86

/lib/hdrs
  1 drwxr-xr-x  2 root      336 Aug 10 11:59 .
  1 drwxr-xr-x  3 root      384 Aug 10 11:59 ..
  2 -r--r--r--  1 root      874 Aug 10 11:59 ctype.h
  3 -r--r--r--  1 root     1186 Aug 10 11:59 db.h
  4 -r--r--r--  1 root     1650 Aug 10 11:59 dos.h
  2 -r--r--r--  1 root      562 Aug 10 11:59 dump.h
  2 -r--r--r--  1 root      685 Aug 10 11:59 ino.h
  1 -r--r--r--  1 root      220 Aug 10 11:59 math.h
  1 -r--r--r--  1 root      335 Aug 10 11:59 mount.h
  1 -r--r--r--  1 root      350 Aug 10 11:59 pan86.h
  1 -r--r--r--  1 root      323 Aug 10 11:59 pascal.h
  2 -r--r--r--  1 root      663 Aug 10 11:59 pat.h
  3 -r--r--r--  1 root     1043 Aug 10 11:59 sh.h
  1 -r--r--r--  1 root      301 Aug 10 11:59 sort.h
  4 -r--r--r--  1 root     1666 Aug 10 11:59 std.h
  3 -r--r--r--  1 root     1218 Aug 10 11:59 stdio.h
  3 -r--r--r--  1 root     1336 Aug 10 11:59 stdtyp.h
  1 -r--r--r--  1 root      495 Aug 10 11:59 sup.h
  6 -r--r--r--  1 root     2985 Aug 10 11:59 sys.h
  2 -r--r--r--  1 root      688 Aug 10 11:59 time.h
  1 -r--r--r--  1 root      229 Aug 10 11:59 who.h
/lib/hdrs total 44

/lib total 795


/odd
  1 drwxr-xr-x  2 root      192 Aug 10 11:59 .
  1 drwxr-xr-x 10 root      240 Aug 09 13:57 ..
  7 -r-xr-xr-x  1 root     3298 Aug 10 11:59 alarm
 16 -r-xr-xr-x  1 root     7422 Aug 09 13:57 glob
 18 -r-xr-xr-x  1 root     8238 Aug 10 11:59 hsh
  8 -r-xr-xr-x  2 root     3934 Aug 09 13:57 init
 28 -r-xr-xr-x  1 root       13634 Aug 10 11:59 log
  8 -r-xr-xr-x  2 root        3934 Aug 09 13:57 multi
 55 -r-xr-xr-x  1 root       27416 Aug 10 11:59 p1
 59 -r-xr-xr-x  1 root       29630 Aug 10 11:59 p2.86
  8 -r-xr-xr-x  1 root        3840 Aug 10 11:59 recv
  2 -r-xr-xr-x  1 root         640 Aug 10 11:59 throttle
/odd total 210


/tmp
  1 drwxrwxrwx  2 root          32 Aug 09 13:57 .
  1 drwxr-xr-x 10 root         240 Aug 09 13:57 ..
/tmp total 1


/usr
  1 drwxrwxrwx  6 root          96 Aug 10 11:59 .
  1 drwxr-xr-x 10 root         240 Aug 09 13:57 ..
  2 drwxr-xr-x  2 root         608 Aug 10 11:59 bin
  1 drwxrwxrwx  4 root         448 Aug 10 11:59 ctext
  1 drwxr-xr-x  2 root          64 Aug 10 11:59 lib
  1 drwxrwxrwx  2 root          48 Aug 09 13:57 pat

/usr/bin
  2 drwxr-xr-x  2 root         608 Aug 10 11:59 .
  1 drwxrwxrwx  6 root          96 Aug 10 11:59 ..
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 backup
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 break
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 chdir
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 chkdsk
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 cls
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 command
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 copy
  2 -r-xr-xr-x  1 root         845 Aug 10 11:59 ctdbl
118 -rwxrwxrwx 1 root 59779 Aug 10 11:59 ctext
  2 -r-xr-xr-x  1 root         593 Aug 10 11:59 cth19
  5 -r--r--r--  1 root        2071 Aug 10 11:59 ctmac.h
  8 -r--r--r--  1 root        3953 Aug 10 11:59 ctmac.s
  3 -r--r--r--  1 root        1143 Aug 10 11:59 ctmac.w
  2 -r-xr-xr-x  1 root         875 Aug 10 11:59 ctvt
 27 -rwxrwxrwx  1 root       12831 Aug 10 11:59 dbl
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 dir
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 diskcopy
 12 -r-xr-xr-x  1 root        5216 Aug 09 13:57 dos
  8 -r-xr-xr-x  1 root        3802 Aug 09 13:57 dosdate
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 erase
 12 -r-xr-xr-x  1 root        5608 Aug 09 13:57 export
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 format
 12 -r-xr-xr-x  1 root        5622 Aug 09 13:57 import
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 path
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 recover
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 rename
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 restore
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 rmdir
 30 -rwxrwxrwx  1 root       14515 Aug 10 11:59 toctext
 27 -rwxrwxrwx  1 root       12831 Aug 10 11:59 tty
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 type
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 ver
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 verify
  1 -rwxrwxrwx 20 root          29 Aug 09 13:57 vol
 17 -rwxrwxrwx  1 root        7682 Aug 10 11:59 vt52
 35 -rwxrwxrwx  1 root       17102 Aug 10 11:59 vtyper
/usr/bin total 342


/usr/ctext
  1 drwxrwxrwx  4 root         448 Aug 10 11:59 .
  1 drwxrwxrwx  6 root          96 Aug 10 11:59 ..
  1 -r--r--r--  1 root         101 Aug 10 11:59 ctext.ci
  1 -r--r--r--  1 root         201 Aug 10 11:59 dbl.ds
  1 drwxrwxrwx  2 root         176 Aug 10 11:59 dev
  1 -r--r--r--  1 root         102 Aug 10 11:59 goudy.vf
  1 -r--r--r--  1 root         204 Aug 10 11:59 goudyb.vc
  6 -r--r--r--  1 root        2767 Aug 10 11:59 goudyb.vd
  1 -r--r--r--  1 root         104 Aug 10 11:59 goudyo.vc
  6 -r--r--r--  1 root        2763 Aug 10 11:59 goudyo.vd
  1 -r--r--r--  1 root          84 Aug 10 11:59 misc.df
  1 -r--r--r--  1 root          84 Aug 10 11:59 misc.tf
  1 -r--r--r--  1 root         188 Aug 10 11:59 misccs.dc
  1 -r--r--r--  1 root         188 Aug 10 11:59 misccs.tc
  1 -r--r--r--  1 root         273 Aug 10 11:59 miscft.dd
  1 -r--r--r--  1 root         272 Aug 10 11:59 miscft.td
  1 drwxrwxrwx  2 root          96 Aug 10 11:59 sys
  1 -r--r--r--  1 root         146 Aug 10 11:59 times.vf
  1 -r--r--r--  1 root         169 Aug 10 11:59 timesb.vc
  1 -r--r--r--  1 root         138 Aug 10 11:59 timesb.vd
  1 -r--r--r--  1 root          86 Aug 10 11:59 timesi.vc
  1 -r--r--r--  1 root          69 Aug 10 11:59 timesi.vd
  1 -r--r--r--  1 root         116 Aug 10 11:59 timesr.vc
  4 -r--r--r--  1 root        1599 Aug 10 11:59 timesr.vd
  1 -r--r--r--  1 root         206 Aug 10 11:59 tty.ds
  1 -r--r--r--  1 root          56 Aug 10 11:59 vt52.ds
  1 -r--r--r--  1 root         204 Aug 10 11:59 vtyper.ds
  1 -r--r--r--  1 root         108 Aug 10 11:59 vtyper.in

/usr/ctext/dev
  1 drwxrwxrwx  2 root         176 Aug 10 11:59 .
  1 drwxrwxrwx  4 root         448 Aug 10 11:59 ..
  2 -r-xr-xr-x  1 root         591 Aug 10 11:59 Make
  1 -r--r--r--  1 root         425 Aug 10 11:59 ctint.h
 20 -r--r--r--  1 root        9401 Aug 10 11:59 dbl.c
  4 -r--r--r--  1 root        1666 Aug 10 11:59 std.h
 18 -r--r--r--  1 root        8289 Aug 10 11:59 sts.c
  3 -r--r--r--  1 root        1463 Aug 10 11:59 sts.h
 25 -r--r--r--  1 root       11791 Aug 10 11:59 toctext.c
  7 -r--r--r--  1 root        3451 Aug 10 11:59 vt52.c
 15 -r--r--r--  1 root        6684 Aug 10 11:59 vtyper.c
/usr/ctext/dev total 96
```

```
/usr/ctext/sys
  1 drwxrwxrwx  2 root         96 Aug 10 11:59 .
  1 drwxrwxrwx  4 root        448 Aug 10 11:59 ..
  1 -r--r--r--  1 root        380 Aug 10 11:59 alarm.c
  3 -r--r--r--  1 root       1072 Aug 10 11:59 mktran.c
  5 -r--r--r--  1 root       2319 Aug 10 11:59 sys.h
  4 -r--r--r--  1 root       1578 Aug 10 11:59 timer.c
/usr/ctext/sys total 14

/usr/ctext total 150


/usr/lib
  1 drwxr-xr-x  2 root         64 Aug 10 11:59 .
  1 drwxrwxrwx  6 root         96 Aug 10 11:59 ..
213 -rw-rw-rw- 1 root 108133 Aug 10 11:59 dict
 17 -rw-rw-rw-  1 root       8100 Aug 10 11:59 dict.ix
/usr/lib total 231


/usr/pat
  1 drwxrwxrwx  2 root         48 Aug 09 13:57 .
  1 drwxrwxrwx  6 root         96 Aug 10 11:59 ..
  0 -rw-rw-rw-  1 pat           0 Aug 09 13:57 .login
/usr/pat total 1

/usr total 730

/ total 3822
```

VIII. CO-IDRIS DEVICES

This section describes the devices supported in all versions of CI. Note that not all of them will apply to your particular configuration.

The Device Interface

The files in **/dev** are "special". They are the hooks into the device drivers of IDRIS. Some of these drivers in turn call upon the DOS device drivers to actually get work done. Here is a description of the files and what they do:

/dev/console
This is the file that, when accessed via **read, write,** or **stty** system calls, will cause activity on the current DOS console. IDRIS accesses the DOS console one character at a time and checks for typed-ahead input periodically. Because IDRIS does not request line-at-a-time input from DOS the DOS "function keys" (used to replay and edit the previous line of input) are not available to IDRIS programs. The function keys instead generate an escape sequence that is dependent on the DOS BIOS used. Function keys are re-enabled for programs run under the **dos** command.

/dev/a
/dev/b
/dev/c
/dev/d
/dev/e
/dev/f
/dev/g
/dev/h
These are files that directly access DOS disk devices by the same name. (i.e. **a:**, **b:**, **c:**, etc.) The interface to these devices uses INT 25 and 26 (DOS absolute disk read/write) interrupts into DOS with a mandatory two-sector offset in order to preserve the boot sector and DOS media descriptor byte. Standard double-sided double-density IBM-style 9-sector diskettes will support a filesystem of 718 blocks. Standard double-sided double density DEC Rainbow-style 9-sector diskettes will support a filesystem of 798 blocks. Raw versions (supporting arbitrary blocking factors) of these devices are available as **/dev/ra** through **/dev/rf.** To make a diskette filesystem for a standard 360KB IBM diskette in drive **a:**, type:

```
    # dos format a:
    # mkfs -s718 /dev/a
    # mount a /x
```

You can make directories and files on the diskette by working in the directory **/x.** When done, unmount the diskette by typing:

```
    # mount -u a
```

/dev/com1
/dev/com2
/dev/com1.lnk
/dev/com2.lnk
These are the files that, when accessed via **read, write,** or **stty** system calls, will cause activity on IBM style "Asynchronous Communications Adapters" (serial ports). **com1** reads and writes an INS8250 USART at I/O address 0x3f8. Another USART at address 0x2f8 is also accommodated. This is known as **com2.** These ACAs are presumed to generate interrupts 12 and 11 respectively via an 8259 interrupt controller at I/O address 0x20. (All these parameters are IBM standard. If the corresponding hardware is not present on your system, you should remove these entries by using the **rm** command.)

These ports (**com1** and **com2**) may be used as the tty argument to **/odd/log** for multiuser operation. One or more instances of **/odd/log** may be invoked via entries in the **/adm/init** control file. For more information see the <u>IDRIS Users' Manual</u> tutorial on "Startup" in Section IV. **com1.lnk** and **com2.lnk** are for use with the callup utility, **cu.** They can also be used for printer ports if you have your printer connected to either **com1** or **com2.**

/dev/hdc0
/dev/hdc1
/dev/hdc2
/dev/hdc3
/dev/hdc4
/dev/hdd0
/dev/hdd1
/dev/hdd2
/dev/hdd3
/dev/hdd4
These are files that use the IBM PC ROM BIOS to access the hard disk through its partition table. With an IBM PC it is possible to have multiple hard disk partitions - one for DOS and the remainder for "other" systems like IDRIS. These files are not useable if your PC does not support partitioning of the hard disk, or if your PC ROM BIOS differs from the IBM version. **hdc2** is the first IDRIS partition on the first drive (usually **c:**). **hdc1** is the DOS partition on the first drive. **hdc0** is the entire first drive (no partitions recognized) - including the partition table itself. If you have two hard disk drives (**c:** and **d:**), then **/dev/hdd[01234]** refers to the second drive. Raw versions of these devices come under the names **/dev/rhd[cd]0** through **/dev/rhd[cd]4.** For more information see appendix G of the IBM DOS 2.0 manual.

The IDRIS device handler for **hd[cd][0-4]** will map bad blocks if the bad block table on the disk is set up to do so. See the installation supplement for the IBM PC and look-alikes in Section X for more information. **Note:** other drivers can be made to support bad block handling as well. Contact Whitesmiths, Ltd. for details.

/dev/idris.0
/dev/idris.1
These files access a DOS file of the same name (**idris.0, idris.1**) on the current DOS disk and directory. It is possible, therefore, to contain an IDRIS filesystem within the DOS file. You can use **mkfs** to create the file before mounting it as a filesystem. The I/O rate is about 35% slower than the BIOS

driver. If **idris.0** is not the root filesystem for CI, you may change the current DOS disk or DOS directory and make **idris.0** point at another file of the same name in the DOS filesystem. Once **idris.0** is mounted, the DOS file it refers to stays open until you quit CI. It is then okay to change the current DOS disk or directory at that point.

/dev/md0
/dev/md1
/dev/rmd0
/dev/rmd1
These files interface to the memory-disk device driver. **md1** and **rmd1** are reserved for use by IDRIS and the **dos** program. **md0** can be used as a very fast "disk" file. **rmd0** is the character special version of **md0** and is used to allocate memory via **stty** calls.

From a C application program the following code would make **/dev/md0** a file of the size specified:

```
#include <std.h>
#include <sys.h>

GLOBAL FILE mdfd {-1};  /* declare file descriptor */
GLOBAL ULONG mdmem {0};

BOOL getmem(size)
    BLOCK size;
    {
    struct
        {
        BLOCK sz;
        ULONG phys;
        } tbuf;

    if ((mdfd = open("/dev/rmd0", UPDATE, 1)) < 0)
        return (NO);            /* fails if busy already */
    tbuf.sz = size;
    stty(mdfd, &tbuf);          /* ask for memory */
    gtty(mdfd, &tbuf);          /* check how much really got */
    if (tbuf.sz != size)
        {
        close(mdfd);            /* not enough memory, put it back */
        return (NO);
        }
    mdmem = tbuf.phys;
    return (YES);
    }
```

The **getmem** routine opens the file and succeeds if the requested number of blocks (**size**) could be allocated. **read(mdfd, <buffername>, <bytecount>)** and **write(mdfd, <buffername>, <bytecount>)** can then be used to access this memory. Alternatively, the memory area can be accessed by the 32 bit physical memory address returned by **gtty** (when 0 < **size**). The memory is released when the file is closed or the program terminates.

Another way to use this device (from the shell command interpreter) is via the following sequence:

```
# mount md0 /x
(attach filesystem at /x [arbitrary location] and hold open)

# stty -i100 /dev/rmd0
(ask for the memory)

# stty -c /dev/rmd0
(observe -i# printout and if -i == 100 ok)

# mkfs -s100 /dev/md0
(make initial directory structure)
```

This will allocate up to 100 blocks of memory to a memory disk and mount it at **/x** as a small filesystem. All files on this filesystem will actually live in memory. When **md0** is last closed (by unmounting in this case) the memory will be given back to IDRIS.

/dev/NODEV
This is a place-holder for an illegal device (i.e. one that always generates an error).

/dev/inode
/dev/kmem
/dev/mount
/dev/myps
/dev/null
/dev/ps
/dev/where
/dev/cnames
/dev/bnames
These are all pseudo-files, documented in Section III of the IDRIS Programmers Manual.

/dev/floppy0
/dev/floppy1
/dev/mt0
/dev/rmt0
/dev/lnk0
/dev/lp
These are "links" to other **/dev** entries. Utilities like **dump, restor, cu,** and **lpr** use these names by default. They are maintained by the system administrator to point to convenient devices.

mt0 and **rmt0** normally point to a "tape" device. The "tape" is likely to be a floppy disk on systems that have no other form of backup. Initially they point to **/dev/a** and **/dev/ra** (**a:** and "raw" **a:**).

lnk0 is set to point at the device used for calling up another computer. Initially this is **/dev/com1.lnk.**

lp is set to point at **com1.lnk** which can have a printer attached to it. **com1.lnk** allows simultaneous writes from different users so it is best on a multi-user system to control access to **/dev/lp** through a script using the enque and deque utility programs. If **com1.lnk** cannot be a printer you may opt to replace **/bin/lpr** with a script that uses **export** (to PRN) to take advantage of whatever printer DOS thinks it has.

Clock and Other Devices

CI also supports a clock device at an interrupt address and frequency which may be specified via flags when **idris.com** is invoked. The paragraph titled Unique Device Support in the Installation Supplement appropriate for your hardware should be referenced for further details on your PC's clock and other devices.

IX. SPECIAL UTILITIES

CO-IDRIS includes several utility programs not found in other IDRIS implementations. These programs allow CI to interact with DOS. Like the other IDRIS utilities, these programs can be invoked via command lines typed in at your terminal, or from within shell scripts. The special utilities included with CI allow you to get files from the DOS system, send files to the DOS system, execute commands in the DOS environment, get the time and date from DOS, and terminate CI execution to return to DOS.

These programs conform to the IDRIS utility conventions as described in the IDRIS Users' Manual, Section II. An executable copy of the **quit** program is stored in your CI filesystem in directory **/etc.** Executable copies of the other programs are stored in your CI filesystem in the directory **/usr/bin.** The technical descriptions of each utility are provided on the following pages. These descriptions are in "manual page" format as described in the "Conventions" manual page in Section II of the IDRIS Users' Manual.

In addition to the above mentioned CI utilities, there is a partitioning utility for creating CI disk partitions on IBM style fixed disks. This program is called IDISK and is contained on Disk 1 of the CI installation package.

NAME

badblk - create and manage bad block maps on the IBM PC/AT and PC/XT

SYNOPSIS

```
# badblk -[b#^ b#-#^ +copy fill g init off# s# t u] <device>
```

FUNCTION

badblk will create a bad block map, add (bad out) blocks to the map, delete (good in) blocks from the map, tabulate a map, and inform the resident that there is a new map. Once a map has been set up, the resident will replace all bad blocks in the map with new blocks allocated from the bad block replacement area.

The flags are:

-b#^ use **#** as a block to bad out or good in. At most ten block numbers, or a total of ten individual block numbers and block number ranges in combination (see below) may be specified.

-b#-#^ use **#-#** as a range of blocks to bad out or good in. At most ten ranges of blocks, or a total of ten block number ranges and individual block number designations in combination, may be specified. Ranges may not be larger than 65534 blocks (i.e., 0-65534 or 1-65535), and the second number cannot be less than the first. The number of blocks in the range is also limited by the size of the replacement device.

+copy try to copy the data from the bad blocks to the new area when adding blocks to the map.

-fill don't fill the replacement blocks with zeros when badding out blocks. The replacement blocks will then have in them whatever was left there originally.

-g delete the listed blocks from the map instead of adding them to it. The blocks must be listed the same way they were when they were added, since **badblk** cannot delete part of a range.

-init create a new bad block map. Unless **-off#** or **-s#** are given, the map will have the offset and size that the device driver returns.

-off# use **#** as the starting block number of the bad block area instead of the value provided by the device driver.

-s# use **#** as the size of the bad block area instead of the value provided by the device driver.

-t tabulate the bad block map for this device.

-u force the resident to update the version of the bad block map in memory. This is done automatically if the map is changed.

If neither **+copy** or **-fill** is specified, the new blocks will be filled with zeros when they are badded out. If **-g** is given, **+copy** and **-fill** are ignored, and the blocks are not written to.

If the map is changed, either by specifying one or more of the flags **-[b#^ b#-#^ g init]** or by specifing the **-u** flag, **badblk** will prompt for confirmation before actually writing the new map out. When it writes the new map out, it also informs the resident that a new map exists and the resident will start using it immediately.

NOTE: the **<device>** specified is the one to be mapped, not the one that the bad block map is on, although they may both be the same device.

The output from the **-t** flag consists of 1) the resident's idea of the bad block offset; 2) a list of blocks and devices that are mapped; 3) the number of free map entries and total number of map entries; 4) the number of free replacement blocks and total number of blocks in the map; and 5) the size of the largest contiguous area in the free map (if it is less that the number of free blocks).

RETURNS

badblk returns success if all flags were correct, it was able to read the device, the device had a bad block device associated with it, and it was able to read [and write] the bad block map.

EXAMPLE

To enable bad block handling on a device:

```
# icheck -patch -fsize <new size of filesystem> hdc2
# badblk -init -t hdc2
```

If the console has been getting lots of error messages saying error reading (or writing) block 23490 on **hdc2**:

```
# badblk -b 23490 hdc2
/dev/hdc2:
map device: hdc2
commit to the new map? y
#
```

Note that at this point, some filesystem damage may exist. If so, consult the essay entitled "Patch" in Section IV of the IDRIS Users' Manual.

If a **badblk -t** shows that blocks 273 to 283 on **wd1** are mapped to blocks 3180 to 3190 on **wd3** and you don't want them to be:

```
# badblk -g -b273-283 wd1
```

SEE ALSO

The manual page on patching IDRIS filesystems, entitled "Patch", in Section IV of the IDRIS Users' Manual.

NOTES

badblk will allow bad blocks to be worked around, but it cannot restore lost data. The best solution is to periodically scan the disks for bad blocks and map them before they become a problem.

The CO-IDRIS error message "bad block ***#####*** on device **<device name>**" may be confusing to users trying to map bad blocks. This message indicates that the block number is bad, not the block itself. It refers to a block number that doesn't make sense because it is off the end of the filesystem. Do not try to map such a block, as doing so will only waste space in the bad block map. Future residents will generate a more appropriate error message.

NOTE: This utility is currently available only on the IBM PC/AT and PC/XT.

NAME

dos - run a DOS command

SYNOPSIS

```
dos -[k# l r] <dos command line>
```

FUNCTION

The command line is passed to DOS, and executed by DOS exactly as if it were passed to DOS directly.

The flags are:

-k# use # KB of memory. The default is to use as much memory as possible.

-l use the largest memory size possible, if # KB is not available.

-r raw command line. The first argument of the command line is taken as the program name. The rest of the command line is passed as is.

The **dos** command looks up the DOS environment variable **$COMSPEC** (which has the name of the DOS command processor, typically **C:COMMAND.COM**) and invokes that program with **/C <dos command line>** as its argument. The effect is as though **<dos command line>** were typed directly to DOS.

The amount of memory given to the program being run is determined by the **-k#** and **-l** flags. If neither flag is given, the largest contiguous block of memory under control of IDRIS, but not allocated to an IDRIS process, is given back to DOS in order to execute the command. If you specify **-k#** (e.g. **-k20**) without the **-l** flag, **dos** attempts to get a contiguous block of memory of that size. If the memory was available, **dos** returns it to DOS and and invokes the command. If the memory was not available, **dos** fails and writes the message "try again" to STDERR. If you specify **-k#** and **-l**, dos attempts to get the requested amount of memory, but if it is not available it uses the largest amount it can get.

RETURNS

dos returns failure if any operations fails. Otherwise, it returns the success or failure status of the invoked program. Note that **command.com** always returns success. Use the **-r** flag to get the status of the invoked program.

EXAMPLE

To get a DOS directory listing of the **a:** device, type the following:

```
# dos dir a:
```

Assuming that **$COMSPEC** is set to **C:COMMAND.COM**, the following command is identical:

```
# dos -r c:command.com /c dir a:
```

The **dos** command can be used within CI shell scripts to allow you to directly invoke DOS applications. An example of this usage is contained in the file **dir** in the directory **/usr/bin.** That file contains the following lines:

```
set 0 -s/ /$0
exec dos $0 $@
```

By using the **chmod** command under CI, this file was made into an executable shell script. By using the **ln** command under CI, the following basic DOS commands were "linked" to the script file:

backup, break, chdir, chkdsk, cls, command, copy, diskcopy, edlin erase, format, path, recover, rename, restore, rmdir, type, ver, verify, vol

Thus you can type:

```
    # dir "*.com"
or,
    # vol
or,
    # discopy a: b:
and so on.
```

Notes:

1) You put **"*.com"** in quotes so that the wild-card expansion is done by DOS and not by CI.

2) **$0** in the shell script refers to the name under which the script was invoked. **$@** refers to the rest of the command line.

3) If you invoke **"command"** from CI, you should remember to use the special DOS command **exit** to return to CI.

SEE ALSO

sh, chmod, ln, export, import

NOTES

There are several important considerations when using the **dos** command, which are summarized as follows:

1) DOS system calls cannot be reentered.

2) CI uses DOS system calls (INT 21) for keyboard input, screen output and portable disk I/O. None of these calls will "hang" in DOS; they will always complete. Multi-tasking of normal CI processes, with no DOS applications programs active, is therefore guaranteed.

3) Any DOS application program which uses DOS system calls that "hang", e.g. waiting for a line of keyboard input, can cause CI to roadblock other processes. This can happen when a process requires a DOS-provided service and the DOS application has not completed its DOS system call. CI multi-tasking can thus be stopped while the DOS application is waiting for keyboard input. There is no way of knowing which DOS programs may do this, but the problem should seldom appear on IBM-style PCs since CI can use the IBM ROM BIOS disk driver, which bypasses DOS for disk I/O.

NAME

dosdate - get current time and date from DOS

SYNOPSIS

dosdate <command>

FUNCTION

dosdate gets the current time and date stored in DOS and converts it to a string of digits of the form **YYMMDDHHMM** (year, month, day, hour, minute). This form is acceptable to the IDRIS **date** command.

<command> is invoked with the converted date as its argument. If no **<command>** is specified, the date is written to STDOUT.

RETURNS

dosdate always succeeds.

EXAMPLE

```
    # dosdate date
    Mon Apr 02 13:03:00 1984 EST
```

This runs the IDRIS date command to set the time and date to the values currently used by DOS.

```
    # dosdate | date -it
    Tue Jan 01 00:11:00 1980 EST
```

In this case the **dosdate** output is piped to the IDRIS **date** command and displayed, but the IDRIS date is not changed.

SEE ALSO

date

NAME

export - send a file to a DOS filesystem

SYNOPSIS

```
export -[b i*] <dosfile>
```

FUNCTION

export sends a copy of an IDRIS file to the DOS file **<dosfile>.**

The flags are:

-b open the output file as a binary file. The default is to open it as a text file.

-i* read input from the file * instead of STDIN.

-size# transfer # bytes at a time whenever possible. The default is 2048 bytes.

<dosfile> has the format **d:filename,** where **d:** is a DOS drive name. The default drive name is that of the current drive. A filename, which must be specified, is an arbitrary DOS pathname, e.g. **abc, test.com, c:\\abc,** or **/echo.com.** Wildcard characters are not interpreted.

RETURNS

export returns success if all **opens, creates, reads,** and **writes** succeed.

EXAMPLE

To export the IDRIS file **autofile** to drive **c:**

```
# export -i autofile c:autoexec.bat
```

This will make a copy of **autofile,** renamed to **autoexec.bat,** in the current DOS directory. If **autoexec.bat** exists in the current directory, its contents are replaced by the contents of **autofile.**

To print **listfile** on the printer:

```
# export PRN < listfile
```

This will copy the contents of the IDRIS file **listfile** to the DOS printer.

To use the DOS printer with IDRIS tools:

```
# pr *.c | export PRN
```

This will run the IDRIS utility **pr,** printing all files in the current directory with names ending in **.c** on the DOS printer.

SEE ALSO

import

NOTES

If you use the \ (backslash) character in DOS filenames, you must "escape" it so that it will not be interpreted as a command to IDRIS by preceding it with a second \, as shown in the list of sample DOS filenames above.

NAME

idisk - make a CI partition on an IBM PC hard disk

SYNOPSIS

idisk -[cyl# c d id# r t size# unit#]

FUNCTION

idisk makes an IDRIS partition on the winchester disk (**C:** or **D:**). **idisk** should be run after running the DOS command FDISK.

The flags are:

-cyl# start at cylinder **#** when replacing or creating a partition.

-c create a new partition.

-d delete a partition.

-id# Select partition by id number (not partition number). Default id number is **-i0x69** (in ASCII, this is a lowercase **i**), which is the IDRIS partition id number. Values must be in the range [0, 255].

-r replace a partition.

-t tabulate (print out) the partition table. This is the default mode.

-size# allocate # cylinders when replacing or creating a partition.

-unit# select disk unit. -unit 0 is the **C:** disk (this is the default). -unit 1 is the D: disk.

Only one of the flags **-[c d r]** may be selected. **-cyl#** and **-size#** must be selected for **-c** and **-r**. There is no check to see if the partition being defined overlaps an existing partition. The partition must, however, fit on the disk.

idisk seeks reassurance before making any change to the partition table. A question of the form "Are you sure ?" is asked before any action is completed. If you type **y** or **Y** when prompted, **idisk** continues, otherwise, **idisk** will make no changes. **idisk** will automatically print out the new partition table and seek verification before writing the changes to disk.

The flag **-id1** selects the DOS partition, and **-id0x69** selects the IDRIS partition.

NOTES

idisk cannot make a bootable partition. Also, **idisk** should be able to calculate defaults for **-size#** and **-cyl#**. It currently cannot.

NAME

import - get a file from a DOS filesystem

SYNOPSIS

```
import -[b o*] <dosfile>
```

FUNCTION

import gets the DOS input file **<dosfile>** and copies it to the specified IDRIS output file.

The flags are:

-b open the input file as a binary file. The default is to open it as a text file.

-o* write output to file * instead of to STDOUT.

-size# transfer # bytes at a time whenever possible. The default is 2048 bytes.

<dosfile> has the format **d:filename**, where **d:** is a DOS drive name. The default is the current drive name. A filename, which must be specified, is an arbitrary DOS pathname, e.g. **abc**, **test.com**, **\\abc**, or **echo.com**. Wildcard characters are not interpreted.

RETURNS

import returns success if all **opens**, **creates**, **reads**, and **writes** succeed.

EXAMPLE

To import the binary DOS file **idris.com** from drive **c:**

```
# import -bo test.com c:idris.com
```

This will make a copy of **idris.com**, renamed to **test.com**, in the current IDRIS directory. If **test.com** already exists in the current directory, its contents are replaced by the contents of **idris.com**.

To use IDRIS tools with a DOS file:

```
# import c:unsorted ¦ sort ¦ export c:sorted
```

This command will read the DOS file **c:unsorted** (presumably lines of text), sort it, and write the sorted version back to the DOS file **c:sorted**.

SEE ALSO

export, sort

NOTES

If you use the \ (backslash) character in DOS filenames, you must "escape" it so that it will not be interpreted as a command to IDRIS by preceding it with a second \, as shown in the list of sample DOS filenames above.

NAME

quit - return to DOS

SYNOPSIS

quit

FUNCTION

quit causes IDRIS to exit to DOS. Control returns to the DOS command processor that originally invoked **idris.com.** A **sync** operation is automatically performed so that disk buffers are written to disk, if necessary. The user should ensure that no processes are left running prior to invoking the **quit** command. (Process status can be determined by using the **ps** command; processes can be terminated by using the **kill** command.) Prior to using the **quit** command, a **date +0** command can be used to timestamp the root filesystem. If you don't set the system time or date when you restart CI, that timestamp is used by default.

RETURNS

quit does not return.

EXAMPLE

```
# date +0
# quit
```

SEE ALSO

sync, date, ps, kill

NOTES

If you reset your machine (via power off, ctrl-alt-del, etc) without quitting CI first, you may corrupt the filesystem data. The degree of corruption is variable, depending upon what processes and I/O were active at the time.

ESTORE COM `ASIC EXE 1&ESTORE HLP `ESTORE MSG `IDRIS 0 *!DRIS COM m! DISK COM , !8DRX COM *cRONFIG BAK bL iEBUG COM `hj.ONFIG BAK tmX BAT }&DDRY COM 1'BMX COM 2^DRZ COM df=BM COM 1H is:

```
    >DISKCOPY A: B: (return)
```

When asked "copy another?" respond with a **y** until all the diskettes have been copied. If you only have a single floppy drive the system will request that you change diskettes a few times for each copy operation.

NOTE: The DOS COPY command will not copy diskettes made with IDRIS filesystems on them. This is because IDRIS does not maintain the same kind of directory structure on its filesystems that DOS does. Use DISKCOPY as described above instead of COPY and the diskettes will be copied properly.

Disk Partitioning

Disk partitioning is performed using the DOS commands FDISK and FORMAT and a special utility called IDISK which is included on Disk 1 of the CI program package. You may also need to use BACKUP and RESTORE if the fixed disk is already in use. Refer to the IBM Disk Operating System manual for instructions on the operation of the DOS programs. BACKUP, FORMAT and RESTORE are described in the chapter named "DOS commands". FDISK is described in the chapter named "Preparing your fixed disk".

Before performing the partitioning process, you must decide how large your CI partition will be. If you wish to make extensive use of both CI and DOS, we recommend the following partitioning scheme:

10 Mbyte (IBM PC/XT)

Partition	Owner	Starting Cylinder	Size
1	DOS	0	150 cylinders
2	CI	150	155 cylinders

30 Mbyte (IBM PC/AT)

Partition	Owner	Starting Cylinder	Size
1	DOS	0	300 cylinders
2	CI	300	314 cylinders

On the 10 Mbyte fixed disk there are 68 blocks (of 512 bytes each) per cylinder. With this scheme there are a total of 10538 blocks available for CI (155 cylinders X 68 blocks/cylinder - 2 blocks reserved for DOS format information). These 10538 blocks can be allocated entirely to the root filesystem if you do not require swapping.

On the 30 Mbyte PC/AT disk there are 102 blocks per cylinder. The recommended configuration results in 32026 blocks for CI (314 cylinders X 102 blocks/cylinder - 2 blocks reserved for DOS format information).

If swapping is desired you must reduce the size of the root filesystem (installation parameter 2) by the number of blocks you allocate to the swap area (installation parameter 3). If you require bad block handling, you will need to subtract 64 blocks from the size of the root filesystem and move the swap area up accordingly, or just subtract 64 blocks from the size of the swap area and leave the swap offset alone. For example, on an IBM PC/AT with 314 cylinders for CI and 2962 blocks of swap space, the size of the root filesystem would be changed to 29000 blocks, the swap offset would be 29000, the swap size would be 2962 blocks, and the bad block map would take up 64 blocks starting at 31962. Alternatively, you could change the size of the swap area to 1982 and leave its offset at 29980, with the bad block map still starting at 31962.

The values for starting cylinder and size of the DOS partition will be used as input to the FDISK utility. The values for starting cylinder and size of the CI partition will be used as input to the IDISK utility. The number of blocks allocated to the CI root filesystem is the value for parameter 2 in the Installation Parameter Table. The number of blocks allocated for the swap area is the value for parameter 3 in the Installation Parameter Table.

You should maintain the partition ordering as described above, i.e., partition 1 should be allocated to DOS and partition 2 for CI. With that ordering, the value for parameter 1 is always **hdc2.** You can select your own values for parameters 2 and 3 or use our recommended values. Note that our recommended values allow for 64 blocks (the default) of bad block map. The recommended values are:

PC/XT	PC/AT
(1) = hdc2	(1) = hdc2
(2) = 9000	(2) = 29000
(3) = 1474	(3) = 2962

You should now write values for all three parameters in the Installation Parameters Table in Section II and proceed with the following steps.

1) If the fixed disk already contains usable data, use the BACKUP command to save the fixed disk files on diskettes. A backup of the disk is required since FDISK, the disk partitioning utility, destroys all data on the fixed disk as part of the partitioning process. The following command will save all fixed disk files on diskettes:

```
BACKUP C:\ A:/S
```

2) Use FDISK to create a DOS partition for the entire fixed disk as follows:

   a) Insert a DOS system diskette in drive **A** and type ctl-alt-del to initiate a system reset. After the **A>** prompt appears, type:

```
FDISK
```

b) Select option 4 from the options menu to determine if a DOS partition already exists. If one does, return to the options menu and select option 3 so that it can be deleted. Return to the options menu.

c) Select option 1 from the options menu to create a new DOS partition. FDISK will ask you if you wish to use the entire fixed disk for DOS. Answer "yes" to that question by typing in a **y**. When FDISK completes its partitioning process, it will return to DOS.

3) Format the newly created DOS partition by typing:

```
FORMAT C: /S
```

4) Use FDISK to delete the DOS partition and recreate one of the desired size by following these steps:

a) Following the **A>** prompt, type:

```
FDISK
```

b) Select option 3 from the options menu to delete the DOS partition

c) Select option 1 from the options menu to create a new DOS partition. FDISK will ask you if you wish to use the entire fixed disk for DOS. Answer "no" to that question by typing in an **n**. Following the "Enter the partition size" prompt, type in the number 150 if you are using the default partitioning scheme; otherwise type in your value for DOS partition size. Following the "Enter starting cylinder number" prompt, type in 0 (zero) if you are using the default partitioning scheme; otherwise type in your value. When FDISK completes its partitioning process, select option 2 from the menu. FDISK will ask what partition you want to make active. Respond by selecting partition number 1 (the one you just created). Exit FDISK to DOS. (The default values create a DOS partition occupying cylinders 0-149 on the fixed disk.)

5) Format the newly created DOS partition and copy the DOS operating system files to it so that DOS can be started from the fixed disk. Type:

```
FORMAT C: /S
```

6) Use the IDISK utility to create the CI partition as follows:

   a) Remove the DOS system diskette from drive **A.** Insert Disk 1 of the CI program package in drive **A.**

   b) If you are using our recommended default values for partition sizes, type the following line just as it appears. Otherwise, use your value for the starting cylinder for the CI partition in place of the number 150; and use your value for the number of cylinders for the CI partition in place of the number 155; then type the line.

   ```
   IDISK -c -cyl150 -size155
   ```

   IDISK will display the new partition table and ask you to confirm that you want it written to disk. Type **y** in response.

7) If you used the BACKUP command in Step 1, you should now use RESTORE to copy files from the backup diskettes to the new DOS partition on the fixed disk. Replace the DOS system disk and type:

```
RESTORE A: C:\ /S
```

This completes the steps for disk partitioning.

Unique Device Support

The Clock:

On the IBM PC or XT a clock is maintained that interrupts through vector 8. To turn on the clock in IDRIS, add **-clkint 8** to the command line. This will allow files to be accurately time-stamped when they are created or modified. It will also help the process scheduler distribute the processing load better among multiple compute-only programs.

Keycodes:

The layout of the IBM PC keyboard makes it hard to find some of the ASCII key sequences used under IDRIS. Here are their mappings:

ASCII	IBM keys	Function
BS	**Backspace**	Erase Last Character
NAK	**Ctrl+U**	Erase Line
DEL	**Ctrl+Backspace**	Program Interrupt
FS	**Ctrl+**	Program Terminate (with memory dump)

Bad Disk Block Handling

Bad block handling is currently supported only for the "pchd" driver (**/dev/hdc2** and so forth). The bad block map is maintained by the bad block utility **badblk.** A description of **badblk** can be found on the **badblk** manual page in Section IX, which deals with special CO-IDRIS utilities.

X. INSTALLATION SUPPLEMENT - DEC RAINBOW

Minimum Memory Requirements

To install and operate CI on the DEC Rainbow you will need a minimum of 128KB of main memory.

Backing up your CI diskettes

Backing up your diskettes can be done with the DOS DISKCOPY utility. The command used to copy CI diskettes is:

```
>DISKCOPY A: B: (return)
```

When asked "copy another" respond with a **y** until all the diskettes have been copied. If you only have a single floppy drive the system will request that you change diskettes a few times for each copy operation.

NOTE: The DOS COPY command will not work to copy diskettes made with IDRIS filesystems on them. This is because IDRIS does not maintain the same kind of directory structure on its filesystems that DOS does. Use DISKCOPY as described above instead of COPY and the diskettes will be copied properly.

Disk Partitioning

The Rainbow Hard Disk Utility Program is used to perform disk partitioning. You should use this utility program to create two partitions on the fixed disk. After the disk has been partitioned, the FORMAT command is used to prepare the disk to contain files. FORMAT may also be used to copy DOS operating system files to a fixed disk partition, so that DOS can be started from the fixed disk. You can then use an option of the Rainbow Hard Disk Utility Program to select the partition containing those files as a bootable DOS partition. You may also need to use BACKUP and RESTORE if the fixed disk is already in use. Refer to the Rainbow MS-DOS Users' Guide, chapters 5 and 6, for complete instructions on how to partition and format the fixed disk, and how to select a bootable partition.

Before performing the partitioning process, you must decide how large your CI partition will be. We recommend the following partitioning scheme:

Partition Name	Drive	Owner	Size
MSDOS-1	E	MSDOS	5 MB
MSDOS-2	F	CI	5 MB

With this scheme there are a total of 10238 blocks available for CI. (5MB/partition divided by 512 bytes/block - 2 blocks reserved for DOS.) These

10238 blocks can be allocated entirely to the root filesystem if you do not require swapping. If swapping is desired you must reduce the size of the root filesystem by the number of blocks you allocate to the swap area.

The number of blocks allocated to the CI root filesystem is the value for parameter 2 in the Installation Parameter Table. The number of blocks allocated for the swap area is the value for parameter 3 in the Installation Parameter Table.

With this partitioning scheme, drive **E** must be allocated to DOS and drive **F** for CI. Drive **E** will also be used as the bootable DOS partition. The value for parameter 1 is therefore **F.** You can select your own values for parameters 2 and 3 or use our recommended values as follows:

(1) = F
(2) = 8700
(3) = 1538

You should now write values for all three parameters in the Installation Parameters Table in Section II and proceed with the disk partitioning procedures described in the MS-DOS Users' Guide.

This completes the steps for disk partitioning.

Unique Device Support

The Clock:

On the DEC Rainbow a clock is maintained that interrupts through vector 160 (decimal) at a 60 HZ rate. To turn on the clock in IDRIS, add **-clkint 160 -tinc 10240** to the command line. This will allow files to be accurately time-stamped when they are created or modified. It will also help the process scheduler distribute the processing load better among multiple compute-only programs.

Keycodes:

The layout of the DEC Rainbow keyboard makes it hard to find some of the ASCII key sequences used under IDRIS. Here are their mappings:

ASCII	**Rainbow keys**	**Function**
BS	**F12(BS)**	Erase Last Character
NAK	**Ctrl+U**	Erase Line
DEL	**<--**	Program Interrupt
FS	**Ctrl+**	Program Terminate (with memory dump)

X. INSTALLATION SUPPLEMENT - DATA GENERAL DESKTOP

Minimum Memory Requirements

To install and operate CI on the DG Desktop you will need a minimum of 256KB of main memory.

Backing up your CI diskettes

Before you use the diskette drives to back up your CI diskettes, you must use the Data General DUTIL utility to indicate what type of diskettes are being used in the diskette drives. Your CI package is distributed on MS-DOS double sided, 9 sector diskettes.

Insert CI Disk 1 in drive **a** and CI Disk 2 in drive **b**. Type the following MS-DOS command to start DUTIL:

```
    >DUTIL  (return)
```

The DUTIL menu will now be displayed on your screen. Select the "INSERT" option from the main menu. The INSERT function menu should now appear. Follow the instructions listed in the menu and "insert" diskettes for drives **a** and **b**. Do not initialize the diskettes. When both "inserts" have been completed, return to MS-DOS and remove the diskettes from the drives.

Backing up your diskettes can now be done with the MS-DOS DISKCOPY utility. The command used to copy CI diskettes is:

```
    >DISKCOPY A: B: (return)
```

When asked "copy another?" respond with a **y** until all the diskettes have been copied. If you only have a single floppy drive the system will request that you change diskettes a few times for each copy operation.

NOTE: The DOS COPY command will not copy diskettes made with IDRIS filesystems on them. This is because IDRIS does not maintain the same kind of directory structure on its filesystems that DOS does. Use DISKCOPY as described above instead of COPY and the diskettes will be copied properly.

Disk Partitioning

The DUTIL program is also used to perform disk partitioning. You should use this utility to create a diskette image on the hard disk by selecting the "CREATE, DELETE, or RENAME diskette image" option from the DUTIL main menu. Refer to the Data General RDOS manual for complete instructions on how to create diskette images on the hard disk using DUTIL.

Before performing the partitioning process, you must decide how large your CI partition will be. We recommend the following partitioning scheme:

Diskette Image Name	Drive	Owner	Size
MC$$.AP	C	MSDOS	5 MB
MD$$.AP	D	CI	5 MB

With this scheme there are a total of 10238 blocks available for CI. (5MB/partition divided by 512 bytes/block - 2 blocks reserved for DOS.) These 10238 blocks can be allocated entirely to the root filesystem if you do not require swapping. If swapping is desired you must reduce the size of the root filesystem by the number of blocks you allocate to the swap area.

The number of blocks allocated to the CI root filesystem is the value for parameter 2 in the Installation Parameter Table, as pictured in the section "Preparing for Installation". The number of blocks allocated for the swap area is the value for parameter 3 in the Installation Parameter Table.

With this partitioning scheme, drive **C** must be allocated to DOS and drive **D** for CI. Drive **C** will also be used as the bootable DOS partition. The value for parameter 1 is therefore **d.** You can select your own values for parameters 2 and 3 or use our recommended values as follows:

```
(1) = d
(2) = 8700
(3) = 1538
```

You should now write values for all three parameters in the Installation Parameters Table in Section II.

This completes the steps for disk partitioning.

<u>Unique Device Support</u>

The Clock:

On the DG Desktop a clock is maintained that interrupts through vector 28 (decimal) at a 1 HZ rate. To turn on the clock in IDRIS, add **-clkint 28 -tinc** -1 to the command line. This will allow files to be accurately time-stamped when they are created or modified. It will also help the process scheduler distribute the processing load better among multiple compute-only programs.

Keycodes:

ASCII	**Desktop keys**	**Function**
BS	**DEL**	Erase Last Character
NAK	**Ctrl+U**	Erase Line
DEL	**Ctrl+DEL**	Program Interrupt
FS	**Ctrl+**	Program Terminate (with memory dump)

X. INSTALLATION SUPPLEMENT - OLYMPIA

Minimum Memory Requirements

To install and operate CI on the Olympia you will need a minimum of 128KB of main memory.

Backing up your CI diskettes

Backing up your diskettes can now be done with the MS-DOS DISKCOPY utility. The command used to copy CI diskettes is:

```
    >DISKCOPY C: C: (return)
```

When asked "copy another?" respond with a **y** until all the diskettes have been copied. The system may request that you change diskettes a few times for each copy operation depending on how much memory is in your system.

NOTE: The DOS COPY command will not copy diskettes made with IDRIS filesystems on them. This is because IDRIS does not maintain the same kind of directory structure on its filesystems that DOS does. Use DISKCOPY as described above instead of COPY and the diskettes will be copied properly.

Disk Partitioning

The Olympia does not offer partitioning capability. Under these circumstances the IDRIS filesystem is actually maintained as a single MS-DOS file on the fixed disk. The file will be named **idris.0** on the **A:** disk. The size of the filesystem and swap area is limited only by the amount of space available for the file **idris.0** in the MS-DOS filesystem. The number of blocks allocated to the filesystem is the value for parameter 2 in the Installation Parameter Table. The number of blocks allocated for the swap area is the value for parameter 3 in the Installation Parameter Table.

With this partitioning scheme, drive **A** must be allocated to DOS and formatted with the fixed disk **format** utility. The value for parameter 1 is therefore **idris.0.** You can select your own values for parameters 2 and 3 or use our recommended values as follows:

```
(1) = idris.0
(2) = 5000
(3) = 2000
```

You should now write values for all three parameters in the Installation Parameters Table in Section II.

This completes the steps for disk partitioning.

NOTE: The following changes to Section III (Installation) need to be made to install CI on the Olympia:

1) Step 2 should read "Insert Disk 1 in drive **C.**"

2) Step 3 should read "Copy the file **idris.com** from the floppy disk to the fixed disk by typing:"

   ```
   copy c:idris.com a:     (C/R)
   ```

   After copying **idris.com** onto the fixed disk, insert disk 2 into drive **C.**

3) In step 4, the command to invoke IDRIS is:

   ```
   idris -root c -swap idris.0 -swapsize 512 -swapoff 5000     (C/R)
   ```

   if you have 128KB of main memory, and

   ```
   idris -root c       (C/R)
   ```

   if you have 256KB or more of main memory.

4) Skip step 5.

5) In step 6, the command to start the installation should read:

   ```
   exec sh -e Install.oly idris.0 5000     (C/R)
   ```

   if you have 128KB of main memory, or

   ```
   sh -e Install.oly idris.0 5000      (C/R)
   ```

   if you have 256KB or more of main memory.

6) In steps 11 through 14, use the command

   ```
   Copy c      (C/R)
   ```

   to copy disks 3 through 6 onto the fixed disk.

Unique Device Support

The Clock:

IDRIS does not currently support the clock on the Olympia.

Keycodes:

ASCII	**Olympia keys**	**Function**
BS	**Backspace**	Erase Last Character
NAK	**Ctrl+U**	Erase Line
DEL	**DEL**	Program Interrupt
FS	**Ctrl+**	Program Terminate (with memory dump)

X. INSTALLATION SUPPLEMENT - WANG PROFESSIONAL WITHOUT MENU DRIVER

Minimum Memory Requirements

To install and operate CI on the Wang you will need a minimum of 256KB of main memory. None of this memory should be allocated to the Wang ramdisk device.

Backing up your CI diskettes

Backing up your diskettes can now be done with the MS-DOS DISKCOPY utility. The command used to copy CI diskettes is:

```
    >DISKCOPY A: B: (return)
```

When asked "copy another?" respond with a **y** until all the diskettes have been copied. If you only have a single floppy drive the system will request that you change diskettes a few times for each copy operation.

NOTE: The DOS COPY command will not copy diskettes made with IDRIS filesystems on them. This is because IDRIS does not maintain the same kind of directory structure on its filesystems that DOS does. Use DISKCOPY as described above instead of COPY and the diskettes will be copied properly.

Disk Partitioning

The Wang Professional does not offer partitioning capability. Under these circumstances the IDRIS filesystem is actually maintained as a single MS-DOS file on the fixed disk. The file will be named **idris.0** on the **C:** disk. The size of the filesystem and swap area is limited only by the amount of space available for the file **idris.0** in the MS-DOS filesystem. The number of blocks allocated to the filesystem is the value for parameter 2 in the Installation Parameter Table. The number of blocks allocated for the swap area is the value for parameter 3 in the Installation Parameter Table.

With this partitioning scheme, drive **C** must be allocated to DOS and **format**ted with the format utility. Drive **C** will also be used as the bootable DOS partition. The value for parameter 1 is therefore **idris.0**. You can select your own values for parameters 2 and 3 or use our recommended values as follows:

```
(1) = idris.0
(2) = 5000
(3) = 2000
```

You should now write values for all three parameters in the Installation Parameters Table in Section II.

This completes the steps for disk partitioning.

Unique Device Support

The serial port driver:

After you have installed CI, you must copy the Wang version of the resident onto the fixed disk. You may do this from DOS as follows:

1) Insert disk 1 into drive **A**.

2) Type the following command:

```
        copy a:wang.com c:idris.com     (C/R)
```

The file **wang.com** contains a CI resident with the serial communications port device driver for the Wang built into it. The **idris.com** file that was used for the installation process contains the appropriate device drivers for the IBM PC.

NOTE: CI does not support bi-synchronous mode on the Wang serial ports. Therefore, the file **/dev/com2** is not used by CI.

The Clock:

On the Wang Professional a clock is maintained that interrupts through vector 0x80 every 20ms. To turn on the clock in IDRIS, add **-clkint0x80 -tinc6144** to the command line. This will allow files to be accurately time-stamped when they are created or modified. It will also help the process scheduler distribute the processing load better among multiple compute-only programs.

Keycodes:

The layout of the Wang Professional keyboard makes it hard to find some of the ASCII key sequences used under IDRIS. Here are their mappings:

ASCII	Wang keys	Function
BS	**Backspace**	Erase Last Character
NAK	**Ctrl+U**	Erase Line
DEL	**Ctrl+Shift+6**	Program Interrupt
FS	**(See below)**	Program Terminate (with memory dump)
¦	**2nd+Shift+1**	Pipe or C program bitwise OR operator
{	**2nd+Shift+[**	Begin C program block
}	**2nd+]**	End C Program block
~	**2nd+-**	C program complement symbol
\e	**2nd+/**	Escape character or Backslash

The FS character (used to cause a program to terminate with memory dump) cannot be generated without reprogramming the tables in the Wang BIOS.

X. INSTALLATION SUPPLEMENT - WANG PROFESSIONAL WITH MENU DRIVER

Minimum Memory Requirements

To install and operate CI on the Wang you will need a minimum of 256KB of main memory. None of this memory should be allocated to the Wang ramdisk device.

Backing up your CI diskettes

Backing up your diskettes can now be done with the DISKCOPY utility. To use it, select "disk copy" from the utilities menu. Select drive **A** as the input drive and drive **B** as the output drive. If you only have a single floppy drive the system will request that you change diskettes a few times for each copy operation. Re-select this menu entry for each remaining diskette not yet backed up.

NOTE: The DOS COPY command will not copy diskettes made with IDRIS filesystems on them. The "file copy" option from the utilities menu will also not work to back up diskettes made with IDRIS filesystems on them. This is because IDRIS does not maintain the same kind of directory structure on its filesystems that DOS does. Use DISKCOPY as described above instead of COPY and the diskettes will be copied properly.

Disk Partitioning

The Wang Professional does not offer partitioning capability. Under these circumstances the IDRIS filesystem is actually maintained as a single MS-DOS file on the fixed disk. The file will be named **idris.0** on the **C:** disk. The size of the filesystem and swap area is limited only by the amount of space available for the file **idris.0** in the MS-DOS filesystem. The number of blocks allocated to the filesystem is the value for parameter 2 in the Installation Parameter Table. The number of blocks allocated for the swap area is the value for parameter 3 in the Installation Parameter Table.

With this partitioning scheme, drive **C** must be allocated to DOS and formatted with the **format** utility. Drive **C** will also be used as the bootable DOS partition. The value for parameter 1 is therefore **idris.0.** You can select your own values for parameters 2 and 3 or use our recommended values as follows:

```
(1) = idris.0
(2) = 5000
(3) = 2000
```

You should now write values for all three parameters in the Installation Parameters Table in Section II.

This completes the steps for disk partitioning.

Unique Device Support

The serial port driver:

After you have installed CI, you must copy the Wang version of the CI resident onto the fixed disk. You may do this from DOS as follows:

1) Insert disk 1 into drive **A.**

2) Select the DOS command processor from the main system menu.

3) Type the following command to the DOS command processor:

```
copy a:wang.com c:idris.com     (C/R)
```

The file **wang.com** contains a CI resident with the serial communications port device driver for the Wang built into it. The **idris.com** file that was used for the installation process contains the appropriate device drivers for the IBM PC.

NOTE: CI does not support bi-synchronous mode on the Wang serial ports. Therefore, the file **/dev/com2** is not used by CI.

The Clock:

On the Wang Professional a clock is maintained that interrupts through vector 0x80 every 20ms. To turn on the clock in IDRIS, add **-clkint0x80 -tinc6144** to the command line. This will allow files to be accurately time-stamped when they are created or modified. It will also help the process scheduler distribute the processing load better among multiple compute-only programs.

Keycodes:

The layout of the Wang Professional keyboard makes it hard to find some of the ASCII key sequences used under IDRIS. Here are their mappings:

ASCII	**Wang keys**	**Function**
BS	**Backspace**	Erase Last Character
NAK	**Ctrl+U**	Erase Line
DEL	**Ctrl+Shift+6**	Program Interrupt
FS	**(See below)**	Program Terminate (with memory dump)
|	**2nd+Shift+1**	Pipe or C program bitwise OR operator
{	**2nd+Shift+[**	Begin C program block
}	**2nd+]**	End C Program block
~	**2nd+-**	C program complement symbol
\e	**2nd+/**	Escape character or Backslash

The FS character (used to cause a program to terminate with memory dump) cannot be generated without reprogramming the tables in the Wang BIOS.

X. INSTALLATION SUPPLEMENT - HEWLETT PACKARD 150

Minimum Memory Requirements

To install and operate CI on the HP-150 you will need a minimum of 128KB of main memory.

NOTE: All the procedures described here assume that you are in the DOS command processor.

Backing up your CI diskettes

Backing up your diskettes can now be done with the MS-DOS DISKCOPY utility. The command used to copy CI diskettes is:

```
    >DISKCOPY C: C: (return)
```

When asked "copy another?" respond with a **y** until all the diskettes have been copied. The system may request that you change diskettes a few times for each copy operation depending on how much memory is in your system.

NOTE: The DOS COPY command will not copy diskettes made with IDRIS filesystems on them. This is because IDRIS does not maintain the same kind of directory structure on its filesystems that DOS does. Use DISKCOPY as described above instead of COPY and the diskettes will be copied properly.

Disk Partitioning

The HP-150 does not offer partitioning capability. Under these circumstances the IDRIS filesystem is actually maintained as a single MS-DOS file on the fixed disk. The file will be named **idris.0** on the **A:** disk. The size of the filesystem and swap area is limited only by the amount of space available for the file **idris.0** in the MS-DOS filesystem. The number of blocks allocated to the filesystem is the value for parameter 2 in the Installation Parameter Table. The number of blocks allocated for the swap area is the value for parameter 3 in the Installation Parameter Table.

With this partitioning scheme, drive **A** must be allocated to DOS and formatted with the fixed disk **format** utility. The value for parameter 1 is therefore **idris.0.** You can select your own values for parameters 2 and 3 or use our recommended values as follows:

```
                (1) = idris.0
                (2) = 5000
                (3) = 2000
```

5000 blocks is the recommended minimum for the size of the root filesystem. This is appropriate for a fixed disk with a capacity of five megabytes. You may want to increase this value if you have a larger capacity fixed disk.

You should now write values for all three parameters in the Installation Parameters Table in Section II.

This completes the steps for disk partitioning.

NOTE: The following changes to Section III (Installation) need to be made to install CI on the HP-150. Also note that **<fsize>** refers to the number which you wrote down for parameter 2 in the Installation Parameters Table in Section II.

1) Step 2 should read "Insert Disk 1 in drive **C.**"

2) Step 3 should read "Copy the file **idris.com** from the floppy disk to the fixed disk by typing:"

   ```
   copy c:idris.com a:     (C/R)
   ```

   After copying **idris.com** onto the fixed disk, insert disk 2 into drive **C.**

3) In step 4, the command to invoke IDRIS is:

   ```
   idris -root c -swap idris.0 -swapsize 512 -swapoff <fsize>  (C/R)
   ```

   if you have 128KB of main memory, and

   ```
   idris -root c       (C/R)
   ```

   if you have 256KB or more of main memory.

4) Skip step 5.

5) In step 6, the command to start the installation should read:

   ```
   exec sh -e Install.oly idris.0 <fsize>      (C/R)
   ```

   if you have 128KB of main memory, or

   ```
   sh -e Install.oly idris.0 <fsize>      (C/R)
   ```

   if you have 256KB or more of main memory.

6) In steps 11 through 14, use the command

   ```
   Copy c      (C/R)
   ```

   to copy disks 3 through 8 onto the fixed disk. Note that you will have to perform this operation six times instead of four.

Unique Device Support

The Clock:

IDRIS does not currently support the clock on the HP-150. If you have the HP-2674 internal printer, you will want to substitute **INT** for **PRN** in Section VI under the heading **Using a Printer.** There is currently no support for the RS-232 or RS-422 serial interfaces.

Keycodes:

ASCII	HP-150 keys	Function
BS	**Backspace**	Erase Last Character
NAK	**Ctrl+U**	Erase Line
DEL	**DEL**	Program Interrupt
FS	**Ctrl+**	Program Terminate (with memory dump)

CO-IDRIS

EDITION 2.37

RELEASE NOTES
AND
PRODUCT UPDATE INSTALLATION PROCEDURE

AUGUST 1985

Copyright (c) 1985

by Whitesmiths, Ltd.

All rights reserved.

INTRODUCTION

The following is a list of differences between this release and release 2.32. It is divided into two parts. The first part (**New Release Features**) is of interest to all customers receiving this update. The second (**Bad Block Handling**) applies only to customers running CO-IDRIS on the IBM PC/XT and PC/AT.

Note: bad block handling is currently supported only for the "pchd" driver on the IBM PC/AT and PC/XT. Other drivers and configurations can be made to support bad blocks as well. Contact Whitesmiths, Ltd. for details.

New Release Features Affecting All PCs

Since the release of 2.32 CI there have been some improvements which we have incorporated into the 2.37 package. These are:

1) Changes have been made to the resident, resulting in improved system performance when running background ("niced") processes (the scheduler time slice table has been eliminated). Multi-tasking should proceed more smoothly, and there should be less competition between background and foreground processes. The segment register corruption problem has also been fixed.

2) Character processing has been modified so that the system no longer crashes when **clist** (a queue of small buffers for characters) becomes full.

3) The **-k#** flag to the **idris** startup command now works like the CO-IDRIS Installation Guide says it should. (See Section VI, "Operating CO-IDRIS", for details.)

4) Several bugs in **libu.86** (the UNIX/V7 libraries) have been fixed, most notably those in **fread.c** and **stdio.h.** Bugs in the **ls** and **grep** utilities and in the IDRIS shell command **test** have also been fixed.

5) Improvements have been made to the code generator (**p2.86**), the preprocessor (**pp**), and the assembler (**as.86**).

6) A bug has been fixed in the **time.s** routine in **libc.86**.

Bad Block Handling on the IBM PC/AT and PC/XT

Bad block handling has been added to the "pchd" device handler (devices **/dev/hdc0 ... /dev/hdc4**) on the IBM PC/AT, PC/XT, and lookalikes. Installations that have older versions of the CO-IDRIS distribution will have to modify their hard disk configuration to make use of bad blocking capabilities.

On the IBM PC/XT and other 10 Mbyte hard disk systems, all that must be changed is the size of the swap area. To do this, the flags passed to the **idris** startup command (which loads the CI resident into memory) must now include **-swapoff 9000** and **-swapsize 1474.** If you have simplified starting CI by placing this command line in a DOS .BAT file, all you need to do is edit this file.

On the IBM PC/AT and other 30 Mbyte hard disk systems, the size of the root filesystem should be reduced, and the swap space moved up so that the bad block map can reside in the last 64 blocks of the partition.

To do this, first type:

```
# icheck hdc2
```

icheck will print out several lines of information about the filesystem, among which will be the value of **bhigh,** which is the highest-numbered block in use at that time. If this number is less than 29000, type:

```
# icheck -patch -fsize29000 hdc2
```

If the value of **bhigh** is greater than 29000, perform the following steps:

1) Save your files on some kind of backup media. (**idris.0** or **idris.1** may be used if you have lots of free space on your DOS partition.)

2) Remove your files from the IDRIS root partition (hdc2).

3) Modify the size of your root partition with the command:

   ```
   # icheck -patch -fsize29000 hdc2
   ```

4) Now you can restore your files.

Regardless of which set of commands you used above, the flags passed to the **idris** startup command (which loads the CI resident into memory) will have to be changed to include **-swapoff 29000** and **-swapsize 2962.** If you have simplified starting CI by placing this command line in a DOS .BAT file, all you need to do is edit this file.

Now that the disk layout has been updated, the CO-IDRIS update package may be installed according to the procedures in the CO-IDRIS Installation Update to follow. When the installation is complete and the new resident has been booted, the bad block map is ready to use. To add new blocks to the map, use the **badblk** utility as per the attached manual page.

CO-IDRIS UPDATE INSTALLATION PROCEDURE

To install the 2.37 version of CO-IDRIS on your existing CO-IDRIS system, perform the sequence of steps appropriate for your hardware. # represents the CI prompt. The commands you are to type appear in **boldface.** Type them exactly as shown.

IBM PC/AT, PC/XT, and lookalikes

1) Insert the floppy disk containing the update files into drive **A:**.

2) Mount the update diskette by typing:

```
# mount -r a /x
```

3) Change your working directory to where the floppy is mounted by typing:

```
# cd /x
```

4) Invoke the script to install the new files and build the new resident by typing:

```
# Updatei
```

The update is now installed. NOW REBOOT TO RUN 2.37 CO-IDRIS.

5) Unmount the floppy by typing:

```
# cd /
# mount -u a
```

Wang PC

1) Insert the floppy disk containing the update files into drive **a:**.

2) Mount the update diskette by typing:

```
# mount -r a /x
```

3) Change your working directory to where the floppy is mounted by typing:

```
# cd /x
```

4) Invoke the script to install the new files and build the new resident by typing:

```
     # Updatew
```

The update is now installed. NOW REBOOT TO RUN 2.37 CO-IDRIS.

5) Unmount the floppy by typing:

```
     # cd /
     # mount -u a
```

DEC Rainbow

1) Insert the floppy disk containing the update files into drive **A:**.

2) Mount the update diskette by typing:

```
     # mount -r a /x
```

3) Change your working directory to where the floppy is mounted by typing:

```
     # cd /x
```

4) Invoke the script to install the new files and build the new resident by typing:

```
     # Updater
```

The update is now installed. NOW REBOOT TO RUN 2.37 CO-IDRIS.

5) Unmount the floppy by typing:

```
     # cd /
     # mount -u a
```

NAME

badblk - create and manage bad block maps on the IBM PC/AT and PC/XT

SYNOPSIS

```
# badblk -[b#^ b#-#^ +copy fill g init off# s# t u] <device>
```

FUNCTION

badblk will create a bad block map, add (bad out) blocks to the map, delete (good in) blocks from the map, tabulate a map, and inform the resident that there is a new map. Once a map has been set up, the resident will replace all bad blocks in the map with new blocks allocated from the bad block replacement area.

The flags are:

-b#^ use **#** as a block to bad out or good in. At most ten block numbers, or a total of ten individual block numbers and block number ranges in combination (see below) may be specified.

-b#-#^ use **#-#** as a range of blocks to bad out or good in. At most ten ranges of blocks, or a total of ten block number ranges and individual block number designations in combination, may be specified. Ranges may not be larger than 65534 blocks (i.e., 0-65534 or 1-65535), and the second number cannot be less than the first. The number of blocks in the range is also limited by the size of the replacement device.

+copy try to copy the data from the bad blocks to the new area when adding blocks to the map.

-fill don't fill the replacement blocks with zeros when badding out blocks. The replacement blocks will then have in them whatever was left there originally.

-g delete the listed blocks from the map instead of adding them to it. The blocks must be listed the same way they were when they were added, since **badblk** cannot delete part of a range.

-init create a new bad block map. Unless **-off#** or **-s#** are given, the map will have the offset and size that the device driver returns.

-off# use **#** as the starting block number of the bad block area instead of the value provided by the device driver.

-s# use **#** as the size of the bad block area instead of the value provided by the device driver.

-t tabulate the bad block map for this device.

-u force the resident to update the version of the bad block map in memory. This is done automatically if the map is changed.

If neither **+copy** or **-fill** is specified, the new blocks will be filled with zeros when they are badded out. If **-g** is given, **+copy** and **-fill** are ignored, and the blocks are not written to.

If the map is changed, either by specifying one or more of the flags **-[b#^ b#-#^ g init]** or by specifing the **-u** flag, **badblk** will prompt for confirmation before actually writing the new map out. When it writes the new map out, it also informs the resident that a new map exists and the resident will start using it immediately.

NOTE: the **<device>** specified is the one to be mapped, not the one that the bad block map is on, although they may both be the same device.

The output from the **-t** flag consists of 1) the resident's idea of the bad block offset; 2) a list of blocks and devices that are mapped; 3) the number of free map entries and total number of map entries; 4) the number of free replacement blocks and total number of blocks in the map; and 5) the size of the largest contiguous area in the free map (if it is less that the number of free blocks).

RETURNS

badblk returns success if all flags were correct, it was able to read the device, the device had a bad block device associated with it, and it was able to read [and write] the bad block map.

EXAMPLE

To enable bad block handling on a device:

```
# icheck -patch -fsize <new size of filesystem> hdc2
# badblk -init -t hdc2
```

If the console has been getting lots of error messages saying error reading (or writing) block 23490 on **hdc2:**

```
# badblk -b 23490 hdc2
/dev/hdc2:
map device: hdc2
commit to the new map? y
#
```

Note that at this point, some filesystem damage may exist. If so, consult the essay entitled "Patch" in Section IV of the *IDRIS Users' Manual*.

If a **badblk -t** shows that blocks 273 to 283 on **wd1** are mapped to blocks 3180 to 3190 on **wd3** and you don't want them to be:

```
# badblk -g -b273-283 wd1
```

SEE ALSO

The manual page on patching IDRIS filesystems, entitled "Patch", in Section IV of the *IDRIS Users' Manual*.

NOTES

badblk will allow bad blocks to be worked around, but it cannot restore lost data. The best solution is to periodically scan the disks for bad blocks and map them before they become a problem.

The CO-IDRIS error message "bad block ##### on device **<device name>**" may be confusing to users trying to map bad blocks. This message indicates that the block number is bad, not the block itself. It refers to a block number that doesn't make sense because it is off the end of the filesystem. Do not try to map such a block, as doing so will only waste space in the bad block map. Future residents will generate a more appropriate error message.

NOTE: This utility is currently available only on the IBM PC/AT and PC/XT.