CROMEMCO

DISK OPERATING SYSTEM

(CDOS)

User's Manual

CROMEMCO, INC.
280 Bernardo Avenue
Mountain View, CA 94040

Part No. 023-0036 November 1978

This manual was produced on a Cromemco model 3355 printer using the Cromemco Word Processing System.

Table of Contents

Introduction

CDOS is the Cromemco Disk Operating System.

The primary use of CDOS is to perform Input/Output with floppy diskettes (disks). It is designed to allow users of Cromemco microcomputer systems to create and manipulate both random and sequential disk files using symbolic names.

CDOSGEN is the Cromemco Disk Operating System GENerator.

It is designed to allow users of CDOS to tailor the Operating System to the needs and hardware configuration at hand. It allows standard or custom Functions to be programmed into the Function Keys of the Cromemco 3101 Terminal.

Most Cromemco software packages are provided with a 32K version of CDOS which may be directly booted up as shipped. CDOSGEN is also provided with most Cromemco software packages.

Additional features including the ability to implement custom device drivers are available in the Advanced Utilities Package.

This is designed as both a reference and an instructional manual. Section One describes the structure of CDOS, its memory allocation, disk layout, and file structure. Section Two covers CDOSGEN including the various parameters necessary to use this program. CDOS operation, start up, and command structure are described in Section Three. Intrinsic commands and Utility programs are covered in Section Four. Section Five is the CDOS Programmer's Manual. This section is designed for the advanced user who wants to gain an in depth

understanding of CDOS and its file structure. Section Six contains a list and explanation of the CDOS error messages. Finally, Section Seven contains a glossary of terms and symbols as they are used throughout this manual.

System Structure

1.1 Memory Allocation

Under CDOS, memory is divided into two major parts. (Please refer to the CDOS Memory Use Map while reading this section.)

The first part is that area which is reserved for CDOS itself. CDOS occupies memory from location Ø through 1ØØH (Low Memory) as well as approximately 8K of memory above the user area (High Memory).

The second part is the User Area. The User Area occupies memory from 1ØØH up to the bottom of CDOS. The size of the user area is determined when CDOSGEN is run and is limited by the amount of memory in the system.

The system is described by the total number of bytes it occupies. Most Cromemco software packages are supplied with a 32K system.

CDOS is loaded from the System Area of the disk into memory by a bootstrap routine. The CONPROC portion of CDOS is the last portion to be loaded and is read from the File Area of the disk.

By special use of Low Memory (Ø-1ØØH), all user programs call CDOS through a standard sequence which is transparent to the size of CDOS.

Referring to the CDOS Memory Use Map, we see that Random Access Memory (RAM) is divided into the following areas:

IOS
The Input/Output System contains the basic input/output functions for the console, printer,

punch, and reader as well as the disk I/O drivers.

DOS
The Disk Operating System is responsible for managing, creating, opening, reading, and writing disk files. It also is in charge of calling user programs and editing console input.

CONPROC
The CONsole PROCessor acts as the normal user interface. It calls a user program based on a simple command line typed on the console and passes parameters from the command to the program. It has some internal functions called intrinsic commands (see section 4.1).

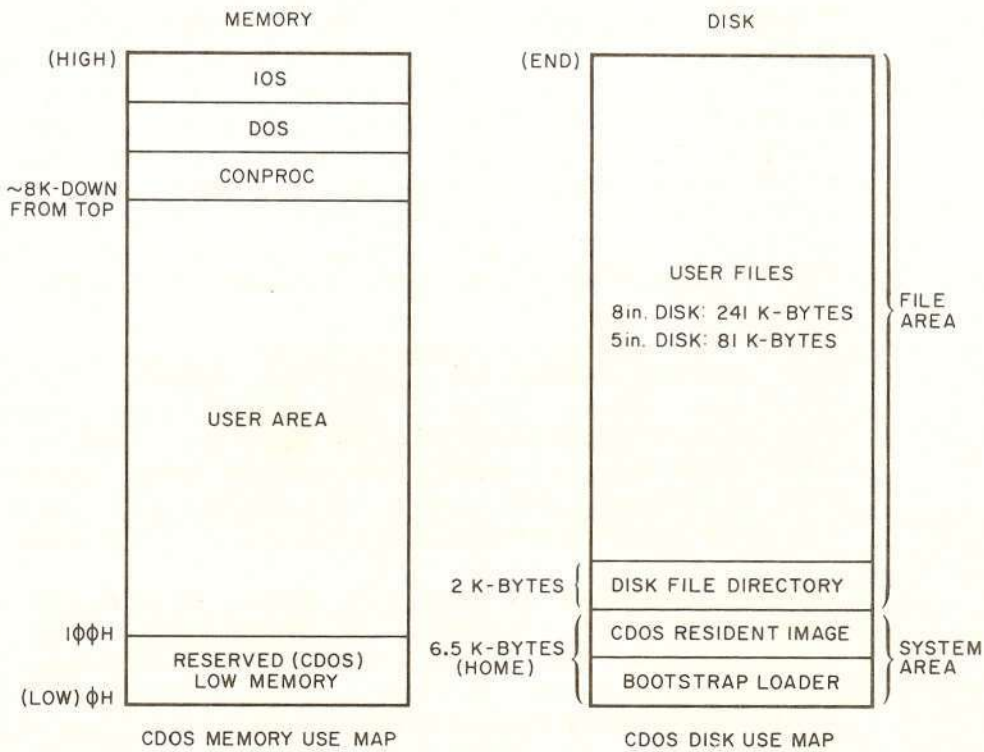
The external functions are the utility (see section 4.2) and user COMmand files which are located on the disk. These files can be identified by the COM file name extension. They are executed by typing the file name without the file name extension (COM is assumed) in response to the CDOS prompt.

User Area
This is where programs are actually run. The User Area begins at 100H (256 decimal) and extends to the system stack below the Console Processor. All programs which are not intrinsic to CDOS are run in this area. Intrinsic programs do not run in this area and therefore do not alter it.

Low Memory
Memory below the User Area is reserved for CDOS for the following special purposes:

0H	System warm start vector
5H	System call vector for user requests
8-40H	Reserved for interrupt vectors
40-5BH	Reserved for system
5C-7FH	Standard user file control blocks
80-FFH	Standard user I/O buffer

The reader is referred to section 5, CDOS Programmer's Guide, for a more detailed discussion of the use of Low Memory.



1.2 Floppy Diskettes

A floppy diskette (or disk) is a flat, round magnetic storage medium. It is protected by a square envelope which exposes only a small portion of the surface of the disk in addition to the hole in the center. When used in conjunction with a disk drive and disk controller, the disk becomes a data file storage device. Data can be added to the data already on the disk or data on the disk can be changed or deleted.

1.2.1 Disk Organization

Each disk used under CDOS is divided into two general areas.

The first area is the System Area. It is accessible to the user only through the WRTSYS utility program. The contents of this area is not listed by the DIRectory intrinsic command. The system files occupy the outer three (for 5 inch disks) or two (for 8 inch disks) tracks of the disk. As can be seen from the CDOS Disk Use Map, starting from track zero, sector one, the system files are the Bootstrap Loader and CDOS Resident Image.

The second area is the File Area. This is the section where user files (programs, data, etc.) and the disk directory are stored. The size of the user file area (exclusive of the File Directory) is 241 kilobytes for large (8 inch) and 81 kilobytes for small (5 inch) disks.

NOTE: The use of the two areas described above is not related. Even if the DIRectory command indicates a full disk, a copy of CDOS may still be written to the System Area using WRTSYS. The DIRectory Intrinsic only indicates the User File portion of the File Area which is occupied on the disk. This has no bearing on the System Area.

1.2.2 Write Protecting Diskettes

8" Diskettes
The large 8" diskettes are write-protected by a notch on the bottom right side (as you face the label) of the plastic disk cover. To write on the disk, cover the notch with a de-write-protect sticker or a piece of masking tape.

5.25" Diskettes
The small 5.25" diskettes are write-protected by the presence of the silver write-protect sticker covering the notch. Remove this sticker to allow you to write on the disk.

Important Distinction
It is important to note that large disks are write-protected by removing the silver sticker, and small

disks are write-protected by placing the silver sticker over the notch.

1.2.3 Precautions Concerning Diskettes

The following precautions are suggested. They are designed to minimize the chance of damage to files stored on floppy diskettes.

1) Whenever changing diskettes, log in the newly inserted disk by initiating a warm start. This is done by sending a CTRL-C character to the computer (refer to section 3.1.4) This procedure is not necessary immediately after booting up CDOS or Resetting the computer.

2) Execute the STATus Utility program occasionally in order to verify the directory (refer to section 4.2.4).

3) Diskettes are magnetic media. The following care and attention should be given to floppy disks:

   a) Keep them away from all sources of magnetic fields such as power transformers and solenoids.

   b) Store them in their dust covers and NEVER lay the bare disk down on a dusty surface.

   c) Keep them out of direct sunlight as the black plastic heats up rapidly. Normal storage temperature is 50 to 125 degrees Fahrenheit (10 to 52 degrees Celsius).

   d) Do not write on the plastic disk jacket with anything but a soft felt tip pen.

   e) Do not touch or try to clean the disk surface. Abrasions may cause loss of data.

   f) Never bend or fold the disk.

   g) It is suggested that the disk not be loaded (i.e., inserted in the drive (with the door closed)) while powering up or

down. Under these conditions random data may be written to the disk. In case of power failure it is wise to check the disk for errors following the return of power.

4) As an additional safety precaution, maintain adequate archives of back up disks. Data may occasionally be lost and the additional cost of back up disks is well worth the valuable programs, data, and time which may be saved.

1.3 Data Files

Data is information. Some examples of data are: a list of names and addresses, a Fortran program, the text of a letter or a manual, etc.

A file is a place and method for storing many individual items of information. Some examples of files are: a telephone or address book, a filing cabinet, the paper on which a grocery list is written, etc.

A computer data file (or simply file) is described by:

1. the storage medium (paper tape, floppy disk, etc.),
2. the method of accessing the data (sequential or random), and
3. the code by which the data is translated for storage (ASCII or internal machine representation).

When a file is created it is given an identifier so that it may be referenced at a later time. This identifier is the file name and optionally the file name extension.

Files may be stored in the same format as data is stored inside the computer. This is referred to as Internal Machine Representation. Files also may be coded (or formatted) according to the American Standard Code for Information Interchange (ASCII). An ASCII file contains only numbers from the ASCII table. On output, each of these numbers is

translated into the character it represents. An ASCII file may be TYPEd while a file stored in internal machine representation must be DUMPed.

Files may be read from or written to a number of devices. The standard devices available under CDOS are:

Device	Data Transfer
Console	Input & Output
Printer	Output
Reader	Input
Punch	Output
Floppy Disk	Input & Output

As normally delivered, only the console, printer, and disk are active. The reader and punch drivers are implemented using the same port assignments as the console. These must typically be changed for a particular user's application.

The primary use of CDOS is to perform I/O with the disk. Any combination of up to four large (8 inch) and small (5 inch) floppy disk drives may be connected to a Cromemco 4FDC Controller. Unlike some large computer systems, all disk files under CDOS may be accessed in either random or sequential order.

Devices are pre-defined by CDOS, but disk files are dynamically created, extended or deleted as required. As will be described in the next section, symbolic names are often used to describe files.

1.3.1 Device Names

The following symbolic names may be used when referring to devices accessible by CDOS.

Format: xxx:[#]

where:

xxx represents a three character name and # is an optional number (Ø thru 7) from the following table:

Device	Name	Number Range
Console	CON:	Ø...7
Card Reader	RDR:	Ø...3
Paper-tape Punch	PUN:	Ø...3
Line Printer	LST:	Ø,1 (either term
	PRT:	Ø,1 may be used)
Dummy Device	DUM:	---(bit bucket/EOF)

1.3.2 Disk File References

The following convention is followed throughout this manual:

the term:

file-ref or file reference

is used to describe:

a) a single file reference (see section 1.3.2.1) including a file name and optionally a disk drive specifier and file name extension,

or

b) an ambiguous file reference if it is specifically stated that the file-ref may include the * and ? replacement characters (see section 1.3.2.2).

1.3.2.1 Single File Reference

A Single File Reference is a unique reference to a unique file stored on a floppy disk and accessible by CDOS. By default or by specification this type of reference addresses a particular file (file name plus an optional file name extension) on a particular disk drive.

Format: [X:]filename[.ext]

where:

X is an optional disk drive specifier indicating the location of the file being referenced. Legal values are

A, B, C, and D.

filename	is a file name composed of up to eight printable ASCII characters except as specified in Note 1.
ext	is an optional 1 to 3 character extension to the file name. See Notes 1 and 3.

Notes:

1) A file name or extension may include any printable ASCII character except the following:

$ * ? = / . , : "space"

2) Although lower case characters are accepted without modification by some user programs, all system functions convert lower case input of file names to upper case.

3) There are several standard types of filename extensions expected by some Cromemco system programs. These are listed below:

BAK Editor back-up file
BAS BASIC LISTed source file (optional)
CMD Batch command file
COB COBOL source file
COM Executable command program
FOR FORTRAN source file
HEX Intel Hex format object file
LIB Source Library
LIS BASIC LISTed source file (optional)
PRN Printer or listing file
REL Relocatable module (object file)
SAV BASIC SAVEd source file (optional)
SYS System image file
TXT Text Formatter input file (optional)
Z8Ø Assembler source file

4) When an executable COMmand file is referred to without the optional disk drive specifier, the system will search the current drive for the file. If this search fails (and the current drive is not the A drive) the A drive is then searched for the file. This procedure is followed only for COM files.

Examples:

A:PROGRAM1.FOR refers to a (FORTRAN source) file on the disk in drive A named PROGRAM1 with a file name extension of FOR.

C:BASIC.COM refers to an (executable COMmand) file on the disk in drive C. The file name is BASIC and the file name extension is COM.

PROG.REL refers to a (relocatable object) file on the disk in the current drive named PROG with a file name extension of REL.

1.3.2.2 Ambiguous File Reference Using Replacement Characters

The asterisk (*) and the question mark (?) may be used as replacement characters in a file name or file name extension to create an ambiguous file reference. The format of the ambiguous file reference is the same as that of the single file reference (see section 1.3.2.1).

The asterisk replaces any character(s) from the position it occupies, to the right, up to the next delimiter (i.e., period (.), question mark (?), or carriage RETURN).

The question mark replaces any single character in the exact position it occupies.

Notes:

1) These replacement characters in no way alter the original file reference. They do not become part of the file name or file name extension. The asterisk and question mark serve only to refer to several files at once by creating an ambiguous file reference.

2) These replacement characters may be used only in commands and programs as specified in this manual.

Examples:

For the following examples, these single file references will be used:

```
PROG.FOR
PROGRAM1.FOR
PROGRAM2.FOR
PROG.REL
PROGRAM1.REL
PROGRAM2.REL
```

Ambiguous File Reference	Files Referred To
PROGRAM?.REL	PROGRAM1.REL PROGRAM2.REL
PROG*.FOR	PROG.FOR PROGRAM1.FOR PROGRAM2.FOR
PROGRAM1.*	PROGRAM1.FOR PROGRAM1.REL
.	All Files
*.FOR	PROG.FOR PROGRAM1.FOR PROGRAM2.FOR

CDOSGEN

2.1 Introduction and Features

CDOSGEN is a very powerful feature of the Cromemco Disk Operating System. It allows CDOS to be built around the user's particular hardware configuration and software needs. As needs and equipment change, CDOS can be reconfigured in a matter of minutes to conform to a new hardware environment.

The ability to program sixteen individual console Function Keys gives CDOS, and all programs run under CDOS, a new flexibility. These programmable keys can be used to facilitate user interaction with the Word Processing System, Basic, Fortran, and CDOS itself.

CDOS now supports up to 64 kilobytes of memory in 1K blocks. CDOSGEN will design an operating system around any combination of up to 4 large and small disk drives.

2.2 Generating a New CDOS

CDOSGEN is run by responding to the CDOS prompt with "CDOSGEN". The file CDOSGEN.COM must be located on the current drive or on drive A if a disk drive specifier is not used.

The program will prompt the user with questions concerning the desired system. Each of these questions is covered in the balance of section 2.2.

2.2.1 Memory Size

After the header, the first prompt CDOSGEN will display is:

```
Memory Size (3FFF - FFFF) ?
```

There are two types of responses to this.

The user may respond with a carriage return. This will cause the program to determine the amount of contiguous memory from 4000H and generate an operating system based on this determination. The system thus generated will be the largest possible for the current hardware configuration.

The user may alternately respond with a hexadecimal number in the range 3FFF to FFFF (inclusive) followed by a carriage return. The number entered specifies the highest address available to CDOS. For example, 7FFF would be entered for a 32K system (because this is the highest address of the top RAM card), BFFF for a 48K system, and FFFF for a 64K system.

NOTE: The device I/O jump table and the bottom of CONPROC (i.e., the bottom address of CDOS) will always be loaded on an even 100H byte boundary.

2.2.2 Disk Drive Configuration

After establishing the system size, CDOSGEN will query the user about the disk drive configuration with the prompt:

```
How Many Disk Drives (1-4) ?
```

Valid responses are any of the numbers 1, 2, 3, or 4 followed by a carriage return. This number represents the total number of drives, both large and small, which are to be used with the generated system.

Next the user will be asked to specify the type of drive in each position. For this series of questions, it is assumed that Large (8 inch - model PFD) drives will always appear in the system in adjacent pairs (as A & B, B & C, or C & D), and that small (5 inch - model WFD or those delivered with the System 2) drives may stand alone (as A, B,

C, or D).

The prompt for the drive type is:

Drive X Type (S - Small, L - Large) -

The X above stands for A, B, C, or D. The user may respond with either an "S" or "L" followed by a carriage RETURN. These stand for Small (5") and Large (8") drives respectively.

2.2.3 Function Key Decoding

The user is then asked to specify the type of Function Key decoding desired:

```
Enter  "S" For Standard Function Key Decoding
       "N" For No Function Key Decoding
       "U" For User Defined Function Key Decoding
       "F" For File Defined Function Key Decoding
       ?
```

These various options are covered in the next sections.

NOTE: The Function Key Decoding options are supported only by the Cromemco model 3101 terminal. Users who have not incorporated this terminal into their system should respond to this prompt with an "N".

2.2.3.1 Standard Function Key Decoding

Responding to the Function Key decoding prompt with an "S" will cause each of the Function Keys to issue a predefined standard command. These standard commands are:

F1	A:<CR>	F9	STAT<CR>
F2	B:<CR>	F10	EDITb
F3	C:<CR>	F11	-BN^Z0P<CR>
F4	D:<CR>	F12	-BI<CR>
F5	DIRb	F13	XFER/Vb
F6	TYPEb	F14	XFER/Cb
F7	*	F15	DEBUG<CR>
F8	.*<CR>	F16	BASIC<CR>

All Function Keys (except F11 and F12) are designed to be used in response to the CDOS prompt. Some (those terminated with a carriage RETURN <CR>) are stand alone Functions and will cause CDOS to respond. Others (those terminated with a blank (b)) will wait for the user to input a file reference followed by a carriage RETURN. Functions 11 and 12 are designed to be used with the Text EDITor program.

2.2.3.2 No Function Key Decoding

Responding to the Function Key decoding prompt with an "N" will disable the Function Keys. This will also free space in CDOS (for drivers) and allow CDOS to occupy less memory after booting.

2.2.3.3 User Defined Function Key Decoding

Responding to the Function Key decoding prompt with a "U" will cause CDOSGEN to prompt the user for the desired decoding of each Function Key. In response to each prompt (F1:, F2:, etc.) the user may enter any series of characters not including the ESCape character. In most applications, CTRL-Z may be substituted for the ESCape character. The ESCape character terminates the current Function Key definition.

Any command, response, or instruction may be entered as a Function. Then, when the Function Key is depressed, it will repeat the characters which were entered during the definition of the Function. Functions may be used while in CDOS, the EDITor, or any program using CDOS (i.e., System Calls) for console I/O.

Referring to section 2.2.3.1 (Standard Function Key

Decoding) note that Function sequences may contain or be terminated with a carriage RETURN character which, in CDOS, will cause execution of the command. Function sequences may also be terminated with a blank, allowing the user to supply additional information as well as a terminating carriage RETURN.

Function Keys may be programmed with a command line which includes carriage RETURNs. Thus F1 may be programmed with the sequence:

```
DIR A:<CR>
DIR B:<CR>
<ESC>
```

When the F1 key is then depressed, the directory of the disk in drive A will be listed followed by the directory of the disk in drive B.

2.2.3.4 File Defined Function Key Decoding

The file referred to in response to this query must be an assembled file which defines each of 16 Functions.

Each Function definition contains the ASCII equivalent to the (command) line to be displayed when the Function Key is depressed and must be terminated by a -1 (FFH). There must be 16 terminators in the file or the message:

```
Function Key Error
```

will be displayed by CDOSGEN.

Example:

The following file was assembled (using the Cromemco Macro Assembler) and SAVEd on the disk as a COM file to give the standard CDOS function key decoding described in section 2.2.3.1:

```
;
;STANDARD FUNCTION KEY DECODING FOR CDOS
;
;NOTE THAT THIS FILE MUST CONTAIN 16 EOM'S
;REGARDLESS OF ANY OTHER CHARACTERS IT USES.
;
;
F1:     DB      'A:',CR,EOM
F2:     DB      'B:',CR,EOM
F3:     DB      'C:',CR,EOM
F4:     DB      'D:',CR,EOM
F5:     DB      'DIR ',EOM
F6:     DB      'TYPE ',EOM
F7:     DB      '*',EOM
F8:     DB      '.*',CR,EOM
F9:     DB      'STAT',CR,EOM
F10:    DB      'EDIT ',EOM
F11:    DB      '-BN',CTRLZ,'0P',CR,EOM
F12:    DB      '-BI',CR,CTRLA,EOM
F13:    DB      'XFER/V ',EOM
F14:    DB      'XFER/C ',EOM
F15:    DB      'DEBUG ',EOM
F16:    DB      'BASIC',CR,EOM
;
CTRLA:  EQU     1       ;CONTROL-A (AN EDIT COMMAND)
CTRLZ:  EQU     1AH     ;CONTROL-Z (DELIMITER IN EDIT)
CR:     EQU     13      ;CARRIAGE RETURN
EOM:    EQU     -1      ;END OF MESSAGE
        END
```

2.2.4 Output File

The Output File may either be a file reference using the file name extension SYS or a disk specifier (A:, B:, C:, or D:) with no file reference.

If a file reference is used, a standard CDOS file with the extension SYS will be created as specified by the user. In order to bring up the operating system which is contained in this file, the WRTSYS utility must be used to move the file to the System Area of the disk and then the system must be booted up.

If a disk specifier alone is used in response to the prompt for an Output File, the file will be written to the System Area of the disk and will NOT

appear in the directory. In order to bring up the system which was just created, the disk upon which the system was written must be placed in the A drive and then booted up. The user will not be running under the new CDOS until it is brought into memory and this is not done until CDOS is reloaded (booted up).

Note: In order to load (boot up) CDOS, the file CONPROC.COM must be present on the disk containing CDOS (the disk in drive A when the system is booted). An error message will be generated if it is not present.

Reference is made to sections 4.2.5 (WRTSYS), 1.2.1 (Disk Organization), and 3.1.4 (System Start-up).

2.2.5 Addresses

At the conclusion of the CDOSGEN procedure, several important addresses are displayed.

Starting address of CDOS -
This is the bottom of CDOS before CONPROC is loaded. After CONPROC is loaded, the bottom of CDOS will always fall on an even 256 (100H) byte boundary.

Starting address of I/O drivers -
This is the first location of the CDOS I/O drivers.

Last address of CDOS -
This is the highest address used by CDOS. Memory between this address and the highest address in the system may be allocated by the user.

CDOS size (max = 1980H) -
This is the Last address minus the Starting address. The maximum size of this area is fixed by the amount of disk space which is allocated to the System Area.

2.3 WRTSYS Utility

The WRTSYS Utility is the only program which will transfer files to and from the System Area of the disk.

NOTE: The WRTSYS Utility does not transfer the file CONPROC.COM. It is necessary for this file to be present on the A drive whenever the system is booted up. CONPROC.COM must be transferred by means of the XFER Utility.

Please refer to section 4.2.5 for a complete description of the WRTSYS Utility program.

CDOS Operation

3.1 System Start-up

3.1.1 4FDC Switch Settings

A brief description of the function of each of the 4FDC switches and their recommended setting follows. For further information on the 4FDC switch settings please refer to the Cromemco 4FDC Disk Controller Manual (part number 023-0005).

Switch 1 is the RDOS (PROM Resident Disk Operating System) DISABLE switch. When ON, the PROM containing RDOS cannot be accessed. When OFF, the PROM resides from C000H to C3FFH in memory. This switch should be OFF for initial system operation.

Switch 2 is the RDOS DISABLE AFTER BOOT switch. When ON, RDOS will automatically be disabled from address space following CDOS boot. When OFF, RDOS remains in memory at C000H following CDOS boot. This switch should be OFF for initial system operation.

Switch 3 is the BOOT ENABLE switch. When ON, CDOS boot-strap is executed from power-on or a computer reset. When OFF, RDOS comes up when power is applied to the system or when the computer is reset. This switch should be ON for initial system operation.

Switch 4 is the INITIALIZATION INHIBIT switch. When ON, diskettes cannot be initialized under software control. When OFF, disks

may be initialized. This switch should be OFF for initial system operation.

NOTE: When configuring a system with 64 kilobytes of memory it is important that switch 2 be ON. This will disable RDOS after CDOS is booted up so that RDOS and system memory do not overlap at locations CØØØH to C3FFH.

With switch 2 ON the only way RDOS can be re-entered after booting CDOS is by resetting the machine. If switch 3 is also ON the user will never be able to access RDOS because CDOS will automatically be booted up any time RDOS is called.

3.1.2 ZPU Switch Settings

The power-on jump should initially be set to CØØØH, the location of RDOS. To do this, the DIP switch should be set as follows:

#15 = 1
#14 = 1
#13 = Ø
#12 = Ø

The clock switch should be set to 4MHz.

3.1.3 16KZ Modification -64K System

In a 64K system using 4 - 16KZ RAM cards, the high card must be modified. In order to avoid a conflict with RDOS (which is resident at COOO-C3FFH on power up or reset) the 16KZ card addressed at CØØØH must be disabled on power up and reset.

This modification can be accomplished by attaching a jumper wire between IC3 pin 3 and IC4 pin 4.

For operation, the modified 16KZ card must have its Bank Ø switch OFF and be addressed at CØØØH (A15 and A14 switches ON).

Switch 2 on the 4FDC must be ON in order to disable RDOS after CDOS has been booted up and the modified 16KZ card has been enabled.

3.1.4 System Start-up

With all the circuit boards installed, the terminal connected, and the switches set as in the previous sections, the following procedure will load CDOS:

1. Turn on power to the computer, terminal, and disk (if the system is using a model PFD).

2. Place the CDOS system disk supplied with this manual in disk drive A.

3. Press the carriage RETURN key up to four times to set the console baud rate. If switch 3 is set to the ON position, CDOS will automatically boot up at this point. If switch 3 is set OFF RDOS will respond with a prompt (;) to which the user must respond with "B" and a carriage RETURN to boot up CDOS.

The system is now up and running.

Either of the above procedures is known as a Cold Bootstrap which includes reading CDOS and the I/O routines from disk.

NOTE: It is advisable to insert the disks after powering-up and remove them before powering-down the machine. The disks may be left in the drives when resetting the machine.

3.1.5 Warm Start and Drive Selection

When a command is issued, the current disk drive is always referred to unless another drive is specified in the command. The current or default drive can be changed by issuing the disk specifier followed by a colon (and a carriage return to terminate).

If drive A is the current drive and it is desired to make drive B the current drive, the user should type:

B:<CR>

and the console will display: "B." indicating

that drive B is now the current drive.

When a drive is FIRST selected in this way, the disk is logged in so the system knows what free space is on the disk. CDOS maintains an internal bit map of the unused (available) sectors on each disk. Whenever a disk is CHANGED, the new disk MUST be logged in by typing CTRL-C. THIS IS ESSENTIAL. If the new disk is not logged in, the old disk will still be logged in and the system will improperly write to the new disk. Note that CTRL-C is needed only when changing disks and not when specifying another drive.

When a CTRL-C is issued, the system performs a WARM START which means it logs off all drives (resets the bit maps) and logs on drive A plus the current drive (if it is other than drive A). This means that the next time a particular drive is accessed a new bit map will be obtained. CTRL-C does NOT re-boot CDOS from disk. See the STAT utility program for a method of determining whether or not a disk has been written to improperly.

NOTE: If a file with an extension of COM resides on drive A and that filename is given as a command, the system will find the file even if drive A is not the current drive. For example, if the Editor (EDIT.COM) resides on drive A, the file which is to be edited (PROGRAM1.FOR) resides on drive C, and the current drive is drive C, the following command line will call the Editor from drive A and edit the user file on drive C:

```
C.EDIT PROGRAM1.FOR
```

3.2 Control Functions

Certain non-printing (control) characters serve to control specific console and printer operations. These characters are described and summarized in the following sections.

Note: A control character is entered on the console by holding down the CTRL key (as you would the shift key on a typewriter) and depressing the appropriate additional key (i.e., A for CTRL-A, C for CTRL-C, etc.). Because control characters are non-printing characters, they are displayed

symbolically on the console by the up arrow (^) followed by the appropriate character.

3.2.1 Console Control Characters

Once the system is running, basic user interaction is with the CONsole PROCessor (CONPROC). This program takes care of file maintenance by means of intrinsic commands and utility programs (described elsewhere in this manual). The prompt given by the system is the disk identifier for the current drive (e.g., "A" for drive A). A command may be entered anytime the prompt is displayed.

While typing a command, the standard buffer input mode is active and certain control characters are usable. To type a control character, press the CTRL key first and hold it in a depressed position (similar to a shift key) while typing the letter. Since a control character is non-printing, it is displayed on the console as the character preceded by an up-arrow (e.g. "^I"). Following is a list of control chracters and their functions:

^E — Physical carriage return and line feed, go to the next line without terminating.

Backspace
Underscore
RUBout
DELete — any of these will delete the last character entered without echo. These will backspace the cursor on a CRT terminal.

RETURN
^M — Either of these will terminate a command line.

^R — Retype current line (after many corrections).

^S — Pause during device I/O. This is primarily used to stop and re-start a listing on the console. Any key may be typed to resume processing, but only ^S can be used to pause.

^U — Delete the current line.

^X Delete the last character with echo. This deletes and echoes the character following three backslashes; three forward slashes are generated by resuming typing. This is primarily used with hard-copy terminals.

3.2.2 Printer Control Characters

There are three control characters which are used to control output to the printer. They are:

^N This character is only for use with Cromemco Printer model 37Ø3. When this character is included in a line which is sent to the printer, it will cause the entire line to be printed in double width characters. A line printed in double width characters may contain only half as many characters as a normal line. This is because each double width character takes up twice as much room as a normal character.

^P Send all console output to the printer as well as to the terminal. This is a toggle action switch. If it is off, the printer is turned on by CTRL-P and vice versa. If a CTRL-P is inadvertently sent while a printer is either not connected to the system or not enabled, another CTRL-P will cancel the first one.

^T Turn off all output to the printer. This control character can be output by a user program but will have no effect if issued from the console.

^W Send all output to the printer as well as to the console. This control character can be output by a user program but will have no effect if issued from the console.

3.3 Automatic Startup and Program Execution

A very powerful feature of the Cromemco Disk Operating System (CDOS) is the ability to enter directly into an application program when powering up the computer. This is especially useful for the inexperienced user as there is no need to deal with any of the commands which are used to load and execute a program.

The following procedure will cause the BASIC user program START.SAV to automatically begin execution when CDOS is entered.

1. Make sure that there is a copy of the batch command file '@.COM' on disk A.

2. Save the BASIC program you want RUN in a file (in this example we are using START.SAV). The program must be SAVEd (not LISTed) in order for this to work.

   Our program for this example is:

```
100 REM THIS IS MY APPLICATION PROGRAM
110 A = 5
120 B = 10
130 PRINT "THE ANSWER IS:  "; A*B
140 END
```

3. Using the Cromemco Editor, create a file named STARTUP.CMD on disk A. Note that this must be named STARTUP.CMD as this is the file name that CDOS looks for.

   In this example, the command file should contain the line:

```
BASIC START.SAV
```

   When CDOS is entered the batch command will call BASIC which will RUN the saved program 'START.SAV'.

4. When the computer is turned on and CDOS is entered (you have to depress the carriage return several times), our example will output the following:

```
A.@ STARTUP
BATCH VERSION ØØ.Ø3

A.BASIC START.SAV

CDOS 16K BASIC, VERSION 5.Ø

THE ANSWER IS:  5Ø

***14Ø END***
```

NOTE: While the STARTUP.CMD file is controlling the operation of the system the CTRL-C (warm start) function as well as RETURN (method of terminating a batch command) are both disabled. After the STARTUP.CMD file has finished, these two functions will be returned to their normal mode of operation. The disabling of these functions during the startup procedure can be useful in preventing the novice or unskilled user from inadvertently gaining control of the machine.

See the batch (@) command (section 4.2.1) for further information.

3.4 Command Structure and Syntax

A request to CDOS is initiated by the user entering a command on the console. CONPROC processes the command to determine if it is one of the eight intrinsic commands (those commands which are internal to CDOS and are not saved as disk files - see section 4.1). If the command is intrinsic it is executed. If the command is not recognized as intrinsic, it is assumed to be a COMmand file on the disk and CONPROC attempts to locate the file with the COM extension. The current disk is searched first, and if the file is not located, the disk on drive A is searched as a Master Library. If the program is found it is loaded into memory starting at 1ØØH, the remainder of the command line is passed to it as control information (see sec. 5), and execution is started at 1ØØH. If it is not found, a message to that effect is displayed on the console.

The command line starts with an optional disk drive specifier. If this is omitted, the default or current disk drive is assumed (except as noted

above). This is followed by the command with no extension (COM is assumed). The rest of the line is determined by the command. The following conventions are observed:

1. All options are preceded by a slash (/).

2. An assignment command follows this format:

   Destination-file-ref=Source-file-ref

3. A comma, blank, or equal sign acts as a delimiter to separate filenames.

4. All letters in command lines are translated into upper case upon entry. All filenames appear in upper case only, but may be referenced by any combination of upper and lower case characters.

5. A blank will be ignored except as a delimiter separating filenames.

3.5 Reset Switch

Turning the key on a System Three computer to the RESET position or depressing the RESET switch on the back panel of a System Two computer causes a hardware reset. This causes control to be transferred to the power on jump address selected on the ZPU card. With the switches on the ZPU and 4FDC cards set as suggested in sections 3.1.1 and 3.1.2, resetting the computer will cause control to be transferred to RDOS and (if switch 3 on the 4FDC is ON) cause CDOS to automatically be re-loaded into memory (cold bootstrap).

Note: After resetting the computer the carriage RETURN key must be depressed several times to re-establish the terminal baud rate.

3.6 Cromemco 3355 Daisywheel Printer Driver

CDOS is supplied with a printer driver designed for use with Cromemco dot matrix printers (model numbers 3703 and 3779). If a dot matrix printer is part of the system, it will operate as specified in section 3.2.2 without any modification to CDOS.

There are two ways to activate the 3355 Daisywheel printer driver.

If the daisywheel printer is to be used with the Cromemco Text Formatter, the option @output=1 must be specified at the beginning of the file which is to be Formatted. This will cause the Formatter program to use an internal daisywheel driver which incorporates variable spacing to achieve margin justification. Refer to the Cromemco Text Formatter Instruction Manual, part number 023-0046 for further information on the output (@o) command.

If the daisywheel printer is to be used as the system printer, as specified in section 3.2.2, than the special driver which is supplied with the Cromemco model 3355 printer must be used. This driver is loaded by executing the program 3355.COM. The driver will remain loaded as long as the system is not rebooted. Once this driver is loaded, the dot matrix type of printer will not function with the system until CDOS has been reloaded.

A procedure for loading the 3355 driver follows.

After CDOS has been loaded, place the disk containing the file 3355.COM in the current drive or on drive A. Then type CTRL-C. Finally type 3355 followed by a carriage RETURN. A message will be displayed when the driver has been properly loaded.

CDOS Commands

4.1 Intrinsic Commands

The Intrinsic Commands are a part of the CONsole PROCessor. They reside in part of high memory occupied by CDOS after the system has been loaded. Because these commands are intrinsic to CDOS, their execution does NOT alter the user area. All files referred to by Intrinsic Commmands are disk files.

4.1.1 ATRIButes

ATRIB establishes or changes allowable file access modes.

Format: ATRIB file-ref [+][p...]

where:

file-ref — is a file reference which may include the * and ? replacement characters (see section 1.3.2.2).

\+ — is an optional parameter which indicates that the following ATRIButes are to be added to those already describing the file.

p... — are optional ATRIBute parameters. They are abbreviated by one or more of the following letters:

E — Erase protect. This file cannot be erased or renamed.

R — Read protect. The system cannot

read from this file. The file may be erased or executed.

W Write protect. The system cannot write to this file. The file may be erased or executed.

ATRIButes may be deleted by assigning a new set of ATRIButes or by giving the ATRIB command with only a file reference and no optional parameters. This will cause all user assignable (erase, read, and write protect) ATRIButes to be deleted. Atributes may be added to those already existing by use of the '+' symbol.

NOTE: ATRIB is a software protection against writing, reading, or erasing disk files. As such, it is possible to override this protection by using an older version of CDOS, using RDOS, etc. If more positive write protection is desired, the use of a write protect sticker is recommended.

Examples:

These examples assume that the following directory is on the current disk:

```
PROGRAM1  FOR    7K   1
PROGRAM2  FOR   18K   2
PROG             2K   1
PROGRAM1  REL    2K   1
PROGRAM2  REL    5K   1
  5 Files ****  34K   6 *****
```

This directory indicates that none of the files have limited access modes (i.e., none of the allowable access modes have been altered by ATRIB). If the command:

```
ATRIB *.FOR R
```

is given, then the directory will appear as follows:

```
PROGRAM1  FOR    7K   1 R
PROGRAM2  FOR   18K   2 R
PROG             2K   1
PROGRAM1  REL    2K   1
PROGRAM2  REL    5K   1
  5 Files ****  34K   6 *****
```

The command used an ambiguous file reference to refer to all files on the current disk with the extension FOR (*.FOR). The command instructed the ATRIB utility to make all the referenced files Read protected (by means of the R parameter). The R following each of two directory entries indicates that PROGRAM1.FOR and PROGRAM2.FOR have been given a Read protect status. If, following this, the command:

```
ATRIB PROGRAM1.FOR +EW
```

is given, then the directory will appear as:

```
PROGRAM1  FOR     7K   1 EWR
PROGRAM2  FOR    18K   2 R
PROG              2K   1
PROGRAM1  REL     2K   1
PROGRAM2  REL     5K   1
  5 Files ****  34K   6 *****
```

This time ATRIB used a single file reference (PROGRAM1.FOR). The command added (by means of the plus sign) categories of protection to the already existing category. The EWR following the file entry in the resulting directory indicates that the file PROGRAM1.FOR is now Write and Erase protected in addition to its previous status of being Read protected. If the plus sign had been omitted from the parameters specified for this command, the file would no longer be Read protected as the Write and Erase protect would have replaced, not have been added to, this status.

4.1.2 BYE

The BYE command forces a return to RDOS. Once in RDOS, control may be returned to CDOS by typing B (for Boot) or GØ (for GO starting at location Ø). In addition, any of the other RDOS commands may be used at this time (i.e., Display Memory, Substitute Memory, etc.). Note that the BYE command will not work if switch 1 or 2 of the 4FDC is ON.

Format: BYE

4.1.3 DIRectory

DIR lists disk file names and sizes followed by a summary of the total disk space used by the files which were listed.

Format: DIR [{ y: / file-ref }]

where:

y	is an optional disk drive specifier. When included in the command line this parameter will specify the drive whose disk directory is to be examined. When omitted, the DIR command will default to the disk in the current drive. Legal values are A, B, C, and D.
file-ref	is an optional file reference which may include the * and ? replacement characters (see section 1.3.2.2). When this parameter is included, only file name(s) which match the file reference will be listed.

The directory is listed in the following format:

```
PROGRAM1  FOR     7K   1
PROGRAM2  FOR    18K   2
PROG              2K   1
PROGRAM1  REL     2K   1
PROGRAM2  REL     5K   1
  5 Files ****  34K   6 *****
```

Each line of the directory listing (except for the last line) includes:

1. the filename,
2. the filename extension (if one exists),
3. the length of the file in kilobytes, and
4. the number of extents (16K blocks) occupied by the file.

The last line of the directory is a summary of the listing. This is not always the same as a summary of all of the files on the disk. The summary line includes the total number of files, kilobytes, and extents which were listed.

Examples:

Assmume that the DIR command, given without any of the optional parameters, will yield the following Directory:

```
PROGRAM1  FOR    7K   1
PROGRAM2  FOR   18K   2
PROG             2K   1
PROGRAM1  REL    2K   1
PROGRAM2  REL    5K   1
  5 Files ****  34K   6 *****
```

This is a listing of the names of all of the files on the current disk. If the current drive is not drive C, the command:

```
DIR C:
```

might yield the following directory:

```
FILENAME  BAS    5K   1
BASIC     COM   19K   2
  2Files ****  24K   3 *****
```

This is a listing of the names of all of the files on the disk in drive C.

The following command would give the user the names of all of the REL files on the current disk:

```
DIR *.REL
```

The directory would appear as:

```
PROGRAM1  REL    2K   1
PROGRAM2  REL    5K   1
  2 Files ****   7K   2 *****
```

4.1.4 ERAse

ERA deletes file(s) from a disk directory.

Format: ERA file-ref

where:

file-ref — is a file reference which may include the * and ? replacement characters (see section 1.3.2.2). All file(s) which match the file reference will be deleted from the disk directory. The space on the disk which the erased files had occupied will then be available for other use.

Note: It is possible to delete a great many files at one time using an ambiguous file reference. Caution is recommended when using replacement characters in the ERAse command file reference. Prior to issuing the ERA command, the DIR command may be given with the same file reference in order to obtain a list of the files which will be deleted by the ERA command.

Example:

If the current disk drive directory is:

```
PROGRAM1  FOR    7K   1
PROGRAM2  FOR   18K   2
PROG             2K   1
PROGRAM1  REL    2K   1
PROGRAM2  REL    5K   1
  5 Files ****  34K   6 *****
```

then the command:

```
ERA PROGRAM1.*
```

would erase the two files referred to by the ambiguous file reference. The resulting directory would appear as:

```
PROGRAM2  FOR   18K   2
PROG             2K   1
PROGRAM2  REL    5K   1
  3 Files ****  25K   4 *****
```

4.1.5 RENname

REN changes the file name and/or file name extension of an existing file.

Format: REN new file-ref=old file-ref

where:

new file-ref — is a file reference which may include the * and ? replacement characters (see section 1.3.2.2). This is the file reference which will exist in the disk directory after the execution of the command. NOTE: If replacement characters are used in the new file-ref, they will be replaced by characters from the file name and file name extension referred to by the old file-ref. Replacement characters never appear in an actual file name or file name extension.

old file-ref — is a file reference which may include the * and ? replacement characters (see section 1.3.2.2). This is the file reference which existed in the disk directory before the execution of the command.

Initially, this command verifies that no file exists on the disk which satisfies the new file-ref. If the new file-ref includes a replacement character, any existing file which satisfies the ambiguous file reference will cause the message 'File already exists' to appear and command execution will be aborted. After this initial check, no further file reference checking takes place. It is possible, in a multiple RENname command, to create more than one file with the same file reference. It is up to the user to ensure that this does not happen.

Examples:

Assume the directory on the current disk drive appears as follows:

```
PROGRAM1  FOR     7K   1
PROGRAM2  FOR    18K   2
PROG              2K   1
PROGRAM1  REL     2K   1
PROGRAM2  REL     5K   1
  5 Files ****  34K   6 *****
```

If the files PROGRAM1.FOR and PROGRAM2.FOR are to be used as text files and the user wants to have their extensions reflect this, the following command will change each file name extension of FOR to TXT on the current disk.

```
REN *.TXT=*.FOR
```

If, in addition, the user desired to change the name of the file PROG to PROGRAM.FOR the following command line would be entered:

```
REN PROGRAM.FOR=PROG
```

After giving these two commands, the directory would appear as:

```
PROGRAM1  TXT     7K   1
PROGRAM2  TXT    18K   2
PROGRAM   FOR     2K   1
PROGRAM1  REL     2K   1
PROGRAM2  REL     5K   1
  5 Files ****  34K   6 *****
```

4.1.6 SAVE

SAVE causes part of the user area to be saved on disk.

Format: SAVE file-ref n

where:

file-ref	will become the name of the SAVEd disk file.
n	is the decimal number of 256 byte pages to be saved.

After linking a FORTRAN, COBOL, or Assembler program and before beginning execution the SAVE command may be issued to create a COMmand file. A

COMmand file may have any file name and must have the file name extension 'COM'.

The number of pages to be saved is displayed by the linker as the last of a series of three exit parameters enclosed in a set of brackets.

It may also be computed by converting the two high digits of the highest address to be saved to decimal (e.g., if the user area is to be saved through address ØBFFH, convert ØB to decimal (11) and save 11 pages).

Remember that the user area starts at 1ØØH and that the SAVE command saves from this address on.

4.1.7 TYPE

TYPE causes an ASCII file to be output to the console (and optionally to the printer).

Format: TYPE file-ref

where:

file-ref is the file to be TYPEd.

Note that only ASCII files may be TYPEd and that an attempt to TYPE a binary (i.e., relocatable or REL) file will yield unpredictable results.

During the execution of this command all of the applicable Console Control Functions (see section 3.2.1) will be in effect. CTRL-S will cause the listing to pause, CTRL-P will cause the listing to go to the printer, and any other character will abort an active listing. Note that any character will restart a listing which has paused in response to a CTRL-S.

If a CTRL-W is included in the file to be TYPEd, all output following this character will be sent to the printer as well as the console. Output to the printer may be stopped by using the CTRL-T character.

4.2 Utility Programs

Utility Programs are NOT part of CDOS. They reside on the disk as COMmand files which can be called into the user area as desired. As opposed to Intrinsic Commands, execution of Utility Programs DOES alter the user area.

4.2.1 @ (Batch)

The Batch (@) command allows the user to execute a file of commands from CDOS. In addition, in the immediate mode, it allows the user to create a file of commands for one time execution.

Format (one time mode):
```
[x:]@[/R][/y] <CR>
```

Format (file mode):
```
[x:]@[/R][/y] [file-ref] [p1 p2...p9]
```

where:

x — is an optional disk drive specifier indicating the location of the batch COM file (@.COM). This parameter is required only if the COM file is NOT located on either drive A or the current drive. Legal values are A, B, C, and D.

R — is an optional recursive Batch function switch. If this option is invoked it allows Batch command file 1 to call Batch file 2 and to return to Batch file 1 when Batch file 2 is completed. If this switch is not set processing will stop when execution of the second file has been completed. The nesting level is only limited to a maximum of 128 commands pending at one time.

y — is an optional disk drive specifier indicating the location of the Batch work file.

p — is an (are) optional parameter(s) to be passed to the CMD file.

Batch takes its commands sequentially from a file containing all of the commands which are to be executed. The Batch command will create its own temporary command file (filename.$$$) when used in the one time mode and will prompt the user with an exclamation mark (!). Valid responses include all legal responses to the CDOS prompt. Execution of the batch command file will commence when a carriage RETURN is entered in response to the prompt.

When used in the file mode, the Batch command references an ASCII file containing a list of CDOS commands. This file must have a file name extension of CMD.

The parameters PØ through P9 are inserted wherever ^Ø, ^1,...^9 appear(s) in the CMD file.

Note: These are not control characters, but rather are the two separate characters, up arrow (^) followed by a number.

Parameter Ø stands for the command file reference and with it you may refer to the CMD file reference itself. Parameters 1 through 9 are those in the command line. These parameter numbers may be repeated in a file. The up arrow itself is represented in the command line by two successive up arrow characters, only one of which is transmitted.

When the Batch command line is given, each word after the filename is treated as a parameter. More complex parameters may be enclosed in single quotation marks. If too many or too few parameters are given, Batch ignores either the extra parameters or the extra commands, respectively.

Examples:

The one time mode can be used to issue a long string of commands which are to be executed without user intervention. The user might issue the following sequence at the console (the A. is the CDOS prompt while the ! is the Batch one time mode prompt):

```
A.@<CR>                          (Batch - one time mode)
!DIR<CR>                         (types the DIRectory)
!TYPE PROGRAM1.FOR<CR>           (types the file)
!REN TEMP=PROGRAM1.FOR<CR>       (renames the file)
!<CR>                            (begins execution)
```

Following the null line, Batch immediately begins execution of the three commands issued, giving the command line for each one just prior to execution.

In the file mode, Batch allows the user to create a file containing the desired command stream and to execute this file as often as desired. As the following example demonstrates, this can be useful for making a backup CDOS disk. The file used by Batch may be created using the Text EDITor and must have an extension of CMD to be found by Batch. In this example, the file used by Batch is called COPY.CMD and contains:

```
XFER/V B:*.COM=A:*.COM
DIR B:
```

The user inserts a blank diskette containing only the CDOS resident image (refer to sections 1.2.1 and 4.2.5) into drive B while the master copy of the CDOS COM files is in drive A, types CTRL-C, and then the Batch command:

```
@ COPY
```

The system then copies all files with the file name extension COM from the disk in drive A to the disk in drive B. The copy routines are followed by a directory so the user may verify that all the desired files have been copied.

Suppose the user creates (using EDIT) a file called EXAMPL.CMD containing the following:

```
DIR ^1
REN OLDFILE^2
```

The user then types:

```
@ EXAMPL NEWFILE '=NEWFILE'
```

to which the system then types the directory listing NEWFILE (along with its size) followed by:

```
REN OLDFILE=NEWFILE
```

and continues by renaming NEWFILE. The equal sign (=) was included in the single quotation marks so that it could be passed as part of the second parameter.

4.2.2 DUMP

DUMP is used to display the contents of a file by 128 byte records.

Format: [x:]DUMP file-ref

where:

x	is an optional disk drive specifier indicating the location of the DUMP COM file. This parameter is required only if the COM file is NOT located on either drive A or the current drive. Legal values are A, B, C, and D.
file-ref	is the file to be DUMPed.

The file is DUMPed in hexadecimal with the first address of a line displayed along the left margin and the ASCII characters corresponding to the hex displayed as characters on the right margin. When a binary file is DUMPed, the ASCII display is meaningless.

Unlike the TYPE intrinsic, both ASCII and binary files may be DUMPed. The records are numbered starting with Ø.

4.2.3 INITialize

INIT is used to INITialize or format (i.e., number the tracks and sectors) large and small floppy diskettes (disks).

Format: [x:]INIT

where:

x	is an optional disk drive specifier indicating the location of the INIT

COM file. This parameter is required only if the COM file is NOT located on either drive A or the current drive. Legal values are A, B, C, and D.

Switch 4 on the 4FDC card must be OFF for this program to execute to completion (see section 3.1.1).

This program will prompt the user to obtain the information it requires. In response to a request for a drive letter, A, B, C, or D must be entered. This is the drive in which the disk to be initialized must be placed (a carriage RETURN typed in response to this question will abort INIT and return control to CDOS). Next the program will request information on the size of the disk. Before responding to this question, make certain that the disk which requires formatting is in place in the drive. Answer 'Y' if the disk is a 5 inch mini floppy disk. Answer 'N' if the disk is an 8 inch large floppy disk. A carriage RETURN typed in response to the prompt for a drive number will terminate execution of the INIT program and return the user to CDOS. See section 7 for an explanation of any error messages which may be displayed.

Cromemco 8 inch floppy disks are supplied already formatted according to the IBM 3740 Data Entry System Format. It is recommended that the user NOT re-initialize these disks when new. Blank 5 inch floppy disks require initialization before use. Occasionally any disk may require re-initialization due to magnetic damage.

Note: Formatting a disk will destroy any information which may have been present on the disk.

4.2.4 STATus

STAT displays any errors in the disk directory. In addition, it summarizes the use of disk and RAM space.

Format: [x:]STAT [y:]

where:

x — is an optional disk drive specifier indicating the location of the STAT COM file. This parameter is required only if the COM file is NOT located on either drive A or the current drive. Legal values are A, B, C, and D.

y — is an optional disk drive specifier indicating the disk whose directory is to be examined. Legal values are A, B, C, and D.

STAT displays information for the default disk drive unless the optional disk drive specifier is used. The program informs the user of:

1) the total amount of disk space available for user files (81 K bytes for small disks and 241 K bytes for large disks),

2) the size of the User Area,

3) the amount of unused disk space (space available for user files),

4) the number of directory entries,

5) the names of any null files (these files contain no information and may be deleted), and

6) any disk errors.

STAT runs a validation of the disk directory to see if any cross-linked files have been created or any clusters have not been allocated. These errors are caused by failing to depress CTRL-C after changing disks.

4.2.5 WRTSYS

WRTSYS is used to write to or read from the CDOS resident image at the front of a system disk (see section 1.2).

Format: [x:]WRTSYS[/S] $\left\{\begin{matrix} \text{d:} \\ \\ \text{file-ref-1} \end{matrix}\right\} = \left\{\begin{matrix} \text{f:} \\ \\ \text{file-ref-2} \end{matrix}\right\}$

where:

x — is an optional disk drive specifier indicating the location of the WRTSYS COM file. This parameter is required only if the COM file is NOT located on either drive A or the current drive. Legal values are A, B, C, and D.

S — is an optional switch indicating that the system is to be written from one disk to another disk, but that only one disk drive is to be used. The program will prompt the user for insertion of the second disk.

d — is a disk drive specifier indicating the disk upon which the CDOS resident image is to be written. Using this specifier alone indicates that CDOS is to be written to the System Area of the disk.

f — is a disk drive specifier indicating the disk from which the CDOS resident image is to be copied. Using this specifier alone indicates that CDOS is to be copied from the System Area of the disk.

file-ref-1 & file-ref-2 — are each file references indicating the source and destination files respectively. Using a file reference indicates that CDOS is to be copied to or from the File Area of the Disk.

The following conventions apply to both the left (destination) and right (source) sides of the equal sign. If only a disk drive specifier is used, the CDOS resident image is copied to or from the System Area of the disk. Refer to section 1.2.1 for a discussion of the System Area vs. the User Area of the disk. If a file reference is used it must have a file name extension of SYS. In this case, the system will be written to or from a User File on the disk.

Note: Using the WRTSYS program to copy any system files does not change the CDOS which is resident in the computer. To change the operating system in use, CDOS must be rebooted (see section 3.1.3).

Examples:

The command:

```
WRTSYS B:=A:
```

will copy CDOS from the System Area of the disk in drive A to the System Area of the disk in drive B. The WRTSYS program will be read from the current disk or, if there is no WRTSYS program on the current disk, from the disk in drive A.

The command:

```
D:WRTSYS A:=B:DOSMAX48.SYS
```

will copy DOSMAX48.SYS from the File Area of the disk in drive B to the System Area of the disk in drive A. The WRTSYS program will be read from the disk in drive D.

The command:

```
WRTSYS A:SPECIAL.SYS=A:
```

will copy CDOS from the System Area of the disk in drive A to a file called SPECIAL.SYS in the File Area of the same disk. The WRTSYS program will be read from the current disk or, if there is no WRTSYS program on the current disk, from the disk in drive A.

4.2.6 XFER

The XFER program transfers files from a disk or other device to another disk or other device. It can be used in one of two modes. The repeat mode:

Format: [x:]XFER<CR>

will repeatedly prompt the user with an exclamation mark (!). Valid responses to this prompt are the same as the portion of the command line following the switches when XFER is used in the one-time mode.

The one time mode will complete one (set of) transfer(s) per command and can be used with the optional switch(es).

Format:

$$[x:]XFER[/s1/s2...] \left\{ \begin{matrix} d: \\ \\ file\text{-}ref\text{-}1 \end{matrix} \right\} = file\text{-}ref\text{-}2[,file\text{-}ref\text{-}3...]$$

where:

x — is an optional disk drive specifier indicating the location of the XFER COM file. This parameter is required only if the COM file is NOT located on either drive A or the current drive. Legal values are A, B, C, and D.

s1,s2... — are any number of the following optional switches (each must be preceded by a slash):

- A — ASCII file transfer.
- C — Compare files without transfer. This operation is driven by the source (file-ref-2) file. If file-ref-2 is shorter than file-ref-1, and the two files are identical for the length of file-ref-2, then the two files will compare as the same.
- F — Filter out illegal ASCII characters (ASCII files only).
- I — Ignore ASCII end of file (CTRL-Z).

R Read protected file is to be transferred.

S Strip all rubouts and nulls from file (ASCII files only).

T Tabs are expanded (ASCII files only).

V Verify files after transfer.

Z Do NOT print size statistics at completion of XFER.

d is the destination disk drive specifier. If used alone, the original names and extensions of any files transferred will be preserved.

file-ref-1 is the destination file reference which may include the * and ? replacement characters (see section 1.3.2.2). If replacement characters are used the portion of the destination file reference which is ambiguous will match the source file.

file-ref-2,3... is (are) the source file reference(s). If only one file reference is used, it may include the * and ? replacement characters (see section 1.3.2.2).

If more than one single file reference is given as the source, the files will be concatenated. If ASCII files are concatenated, the /A switch must be used to remove the end of file markers from between the files.

An ambiguous transfer with verification will be terminated by a verification error.

NOTE: The XFER utility will only transfer files to and from the File Area of the disk. The WRTSYS utility must be used to write system files to and from the System Area of the disk (see section 1.2.1).

Examples:

The command:

```
XFER/V B:=PROGRAM1.FOR
```

will copy and verify PROGRAM1.FOR from the current disk to disk B. The copied file will have the same file name and file name extension as the source file. The XFER program will be read from the current drive or drive A.

The command:

```
XFER B:=A:*.FOR
```

will copy all files with the file name extension FOR from drive A to drive B. Each of the copied files will have the same file name and file name extension as each of the source files. The XFER program will be read from the current drive or drive A.

The command:

```
XFER D:*.TXT=A:*.TYP
```

will copy all files with the file name extension TYP from drive A to drive D. Each of the copied files will have the same file name as each of the source files, but will have the file name extension TXT. The XFER program will be read from the current drive or drive A.

The following command will copy all files from drive A to drive B and then verify these copies:

```
XFER/V B:=A:*.*
```

The XFER program will be read from the current drive or drive A.

4.2.7 MEMTEST

MEMTEST is a diagnostic program designed for testing memory. The following sections describe the use of MEMTEST as well as output generated by the program.

4.2.7.1 Using MEMTEST

The Memory Test performs various checks on 16K blocks of memory addressed at 4000H, 8000H, or C000H. In order to run MEMTEST, there must be RAM addressed at 0 in all banks. Load and begin execution of MEMTEST by typing MEMTEST in response to the CDOS prompt. The disk file MEMTEST.COM must be located on the current drive or on drive A. The test program will prompt with:

```
BANK:
```

In response, the user should enter the bank number in which the blocks are to be tested followed by a space. MEMTEST will then prompt with:

```
16K BLOCK:..
```

note commas not needed just put space

The user should enter 4, 8, or C (or any combination of these numbers with spaces in between) followed by a carriage RETURN. The 4 indicates that the 16K block of memory starting at 4000H in the specified bank is to be tested. The 8 or C indicates the block starts at 8000H or C000H.

After this carriage RETURN, MEMTEST will prompt the user for additional 16K blocks in other banks. If no further blocks are to be tested, the user should enter a carriage RETURN in response to the BANK prompt. If an entry error is made, a question mark (?) will be displayed and the user will be re-prompted.

MEMTEST continues testing until the user terminates the test (see Control Functions Section - 4.2.7.3)

Example:

The following responses to the MEMTEST prompts will check 16K blocks of memory starting at 8000H and C000H in bank 0 and at 4000H, 8000H, and C000H in

bank 7.

```
BANK:Ø 16K BLOCK:8 C<CR>
BANK:7 16K BLOCK:4 8 C<CR>
BANK:<CR>
```

4.2.7.2 Tests

MEMTEST performs five different tests on the memory being checked.

The Peak Test shifts a one in a field of zeroes through the block. It loads a byte with one bit high into the top location of the block. The next location gets the byte shifted one position. After the high bit is shifted into the carry flag, the shift direction is reversed. Each byte is checked after it is written and again after the entire block has been loaded. The test is repeated for each of the 18 possible ways of loading this pattern.

The Valley Test is similar to the Peak Test except that the test byte is a zero in a field of ones.

The Delay Test checks long term memory retention. The block is filled with a pattern and then tested to see if it can retain the pattern for at least six seconds (at 4 MHz). The test is repeated for a number of different patterns.

The M1 Test checks the capability of the block to be read during M1 machine cycles.

The Bank Select Test determines in which bank the block of memory appears and tests the Bank Select circuitry of the card.

4.2.7.3 Control Functions

During execution, MEMTEST accepts the following input characters:

CTRL-P	causes MEMTEST to display (Print out) an image of the 16K block currently being checked. After the display, the test continues from where it was interrupted.
ESCape	causes a display of the current 16K block image and the termination of the current test with a prompt for a new test.
CTRL-S	causes the remainder of a print out to be Skipped. The test continues as if the print out had finished.
CTRL-Q C	causes MEMTEST to abort (Quit) and return control to CDOS.
CTRL-E	prints out an Error history showing which cards have had failures since the test began.

4.2.7.4 Error Print Out

If errors occur in the Peak, Valley, or Delay Test, an image of the card will be displayed indicating the physical locations of the ICs in which the errors occurred. The letter displayed as an error message indicates the test in which the error occurred. The M1 and Bank Select Tests are made only if there are no errors in the first three tests.

Example:

The following display indicates that errors occurred in bits 1 and 3 of rows 1 and 2, respectively, of the block of memory starting at CØØØ in bank 7:

```
(16K BLOCK C)  -0- -1- -2- -3- -4- -5- -6- -7-
     (BANK 7)0:... ... ... ... ... ... ... ...
             1:... PVD ... ... ... ... ... ...
             2:... ... ... PVD ... ... ... ...
             3:... ... ... ... ... ... ... ...
```

This display indicates a possible problem in the RAM chips located in the same physical positions on the board as the PVD messages on the display. For proper reference, the user must face the component side of the card.

4.2.8 Cromemco Text EDITor

The Cromemco CDOS Text EDITor enables the user to create, edit, and save ASCII text or program files. The Text Editor is very versatile in that it can be used to manipulate and edit text on a line, word, or character basis. Characters and words can be inserted in, deleted from, or changed within a line of text. The point of change can be chosen to be between any two characters. Insertions and deletions can be made that cover more than one line of text. The Text Editor is not encumbered by line numbers or other extraneous information, and operates using only the text itself as a guideline to changes.

The user who is not familiar with the CDOS Text Editor is referred to the Cromemco Text Editor Instruction Manual (part number 023-0040). In particular, Chapter II will aid the novice user by means of an example of an actual EDIT session.

Programmer's Guide

5.1 Introduction to CDOS System Calls

To a programmer, system calls are the single most important feature of CDOS. The user writing assembly language programs to run under CDOS should become familiar with the use of system calls.

A system call is a call to the operating system which initiates a function, usually involving one of the I/O devices. The most important system calls perform I/O with, and manipulate, the floppy disk drives. CDOS also has system calls to perform device I/O with CRTs, printers, punches, and readers. System calls are available to perform such special purpose functions as storing and reading the date or time of day and multiplying and dividing integers.

Because CDOS is designed to handle all I/O functions in a manner which is independent of system size and configuration, programmers should ordinarily do all I/O through the operating system by means of system calls.

A system call is executed by loading the C register with the number of the call (refer to Section 5.1.5) and loading any entry parameters into the specified registers. Upon execution of a CALL 5 instruction, CDOS will perform the desired function. When CDOS has finished, it will return to the user program with a RET instruction.

All Z80 registers will be preserved by system calls except the F (Flag) register and those containing Return Parameters. The Z80 set of Primed Registers are not used by system calls and hence programs may safely use these registers for

temporary storage. Entry Parameters are preserved by system calls unless otherwise noted in sections 5.1.2 through 5.1.4.

For more detailed information on how system calls index into the operating system, refer to the CDOS Memory Allocation section (5.3). For a summary of the system calls available with CDOS (listed in numerical order), refer to Section 5.1.5. The system calls are grouped under three general headings and described individually in detail in this chapter. They are also cross-referenced in the Index.

5.1.1 Detailed List of CDOS System Calls

The following is a detailed description of the CDOS system calls. They are sub-divided into three sections.

Section 5.1.2 covers Input/Output from/to all devices except disk drives. This includes the system console(s), printer, punch, and reader.

The next section (5.3.3) covers the system calls used to access disk files. This includes functions to search for, create, rename, delete, open and close disk files. Also included are routines to read or write data, and a number of functions designed to manipulate the disk drives.

Section 5.1.4 covers a number of additional calls which fit no specific category and include such functions as integer multiply and divide; set/read the date/time of day; and system abort and program link functions.

The system calls given below are in numerical order in each of the three sections.

All device and disk input and output should be done through the CDOS system calls. This allows user programs to be independent of physical devices or port assignments and assures that the program will be able to run on other Cromemco machines regardless of how I/O devices are connected to those machines. If a change needs to be made in a device driver, it has only to be done once in the system drivers and this change becomes effective in all programs which access that driver through the

system calls.

To use one of these routines the C register must be set to the function number given with the title of each instruction. The other registers are set up as that system call requires (for example, the E or DE registers usually contain the entry parameter passed). A CALL 5 instruction is then executed to carry out the function. Remember that CDOS initializes location 5 with a jump instruction. This is done so that the location of CDOS in memory is transparent to a user program. A program using the CDOS system functions does not therefore need to do a CALL to a particular address in CDOS.

5.1.2 CDOS Device System Calls

The system calls of this section involve device I/O with all devices except disk drives. The number given preceding each CDOS function is the number which should be loaded into the C register prior to the CALL 5 instruction. This number is given first in decimal and then in hexadecimal in parentheses.

1 - READ CONSOLE (with echo)

This call is used to retrieve a single character (one byte) from the console. The byte will be returned in the A register with the parity bit (Bit 7) reset. CDOS does not return control to the user program until a character has been read and echoed back to the CRT. No entry parameters are required other than the value in the C register.

Note that a CTRL-Z (^Z) character is usually to be considered by a user program as an end of file mark. Also, most other control characters will not be echoed back to the CRT and some have special meanings for the operating system. For example, CTRL-J (LF), CTRL-M (CR), and CTRL-G (BEL) are echoed directly, CTRL-I (TAB) is echoed as expanded spaces (see WRITE CONSOLE below), and CTRL-P will toggle on/off a line printer but will not be echoed.

2 - WRITE CONSOLE

This call is used to write a single ASCII character (one byte) to the CRT. The character is placed in the E register before the call. CDOS will wait until the console is ready to receive the character and then print it. No parameters are returned by the call.

After CTRL-P (^P) is typed while CDOS is outputting characters with this system call, all subsequent characters are sent to both the console and the printer until CTRL-P is depressed a second time (thus CTRL-P acts as a toggle switch). CTRL-W (^W) also causes subsequent characters to be sent to both the console and the printer but must be encountered in a file to do so. CTRL-T (^T) in a file cancels the effect of either the CTRL-W or the CTRL-P and causes characters to be sent only to the console. CTRL-W and CTRL-T may be edited into a file so when that file is being typed out on the console, it can stop and start the printer at the appropriate places.

CTRL-I is the tab character and is converted to spaces as it is typed out so that the cursor is positioned at one of the standard tab stops: column 1, 9, 17, 25, 33, 41, 49, 57, 65, or 73. However, the tab is still stored internally in a file as a single ASCII character (9).

3 - READ READER

This call will read one character from a paper tape or card reader. All 8 bits are read and returned in the A register (i.e., the parity bit is not masked). No entry parameters are required other than the value in the C register. Since no card or paper tape reader is connected to a standard Cromemco computer system, the port assignments and method of interface (default is serial) for this system call are set up initially with the console as a reader.

Also note that console status is checked during the read for the CTRL-S (^S) toggle, enabling the user to stop/start the reading process at will. This is useful for pausing during a paper tape jam, for example.

4 - WRITE PUNCH

This call will punch one character on a paper tape punch. All 8 bits are punched (i.e., including parity). The character to be punched is placed in the E register before the call. CDOS will then wait until the punch is ready to receive the character. No parameters are returned by this call.

Also note that console status is checked during the read for CTRL-S (^S), enabling the user to stop/start the punching process. This is useful for pausing during a paper tape jam, for example.

5 - WRITE LIST

This call will print a single character (one byte) on the printer. The character to be printed is placed in the E-register before the call, and CDOS will wait for the printer to be ready before returning to a user program. No parameters are returned by this call.

Tabs are not expanded, and control characters which do not have meaning to the printer will be transmitted anyway. Cromemco printers will ignore such control characters. A useful control character for the Cromemco Model 3703 Printer is CTRL-N (^N), which, when present in a line of printer output, will cause that line to be printed in double width characters.

Also note that console status is checked during the print out for the CTRL-S (^S) character, enabling the user to stop/start the listing. This is useful for pausing to start a new box of line printer paper.

7 - GET I/O BYTE

For extra I/O devices, an IOBYTE has been provided. This byte is not currently used by CDOS, but it is provided for the user's programs. This system call returns the IOBYTE in the A register. The format of the byte is:

Bit:	7	6	5	4	3	2	1	Ø
Device:	PRN	PUNCH		READER		CONSOLE		

Thus up to eight consoles can be designated, four each of paper-tape punch and reader, and one printer.

8 - SET I/O BYTE

This call allows the user program to set the IOBYTE. The E register contains the byte prior to the call. See GET I/O BYTE for the format of the byte.

9 - PRINT BUFFERED LINE

This call will print a string of ASCII characters which has been terminated with the dollar sign ($) character. The DE register pair is set up with the address of the beginning of the string before the call is made to CDOS. If the printer toggle is on, the line will also be sent to the printer.

1Ø (ØAH) - INPUT BUFFERED LINE

This call will read an input line from the console. The DE register must be pointing to an available buffer before the call is made to CDOS. The first byte of the buffer must contain the maximum length of the buffer. On return from this call the second byte of the buffer will contain the actual length entered. The line that is input will be stored beginning at the third byte. If the buffer is not full, the byte at the end of the line will contain a zero.

When the line is being entered, the following

characters will have special meaning:

CTRL-C (^C)	Abort. Warm boot back to CDOS.
CTRL-E (^E)	Physical CR-LF. The line is not terminated and nothing is entered into the buffer. This character is used to enter a line longer than can be entered on the console.
CTRL-P (^P)	Toggle printer/console link. When this character is first typed, the link is toggled ON. All characters will then be sent to the console and the printer. The next time the character is typed, the toggle will be turned off. All characters will then be sent only to the console.
CTRL-R (^R)	Repeat what has been typed so far on the line.
CTRL-U (^U)	Delete the entered line and go back to beginning of buffer for new line.
CTRL-X (^X)	Delete the previous character and echo the deleted character (used for hard-copy terminals).
RUBout	Delete the previous character and back up the cursor (used for CRT terminals).
DEL	Same as RUBout.
Underscore	Same as RUBout.
Backspace (^H)	Same as RUBout.

11 (ØBH) - TEST CONSOLE READY

The console is tested to see if a character has been typed. If a character has been typed, ØFFH is returned in the A register. If no character has been typed, Ø is returned in the A register.

128 (8ØH) - READ CONSOLE (without echo)

This call is the same as READ CONSOLE (with echo) except that it does not echo the character after it is read. The byte is returned in the A register.

142 (8EH) - SET SPECIAL CRT FUNCTION

This call is used to perform special functions on the system console terminal. The call is designed to be very broad and include as many of the special features available in present-day intelligent terminals as possible. In particular it allows the programmer to take full advantage of the features available in Cromemco Model 31ØØ and Model 31Ø1 CRT terminals. The call is executed by first loading the correct entry parameters into the DE register-pair. These values are summarized in the following table. No parameters are returned by this call.

Mnemonic		Function	D	E
Address	*	address cursor on screen	1-8Ø	1-24
Clear	*	clear CRT screen	Ø	Ø
Home	*	home cursor without clearing	1	Ø
Back	*	move cursor to left one character position	2	Ø
Forward	*	move cursor to right one character position	3	Ø
Up	*	move cursor up one line	4	Ø
Down	*	move cursor down one line	5	Ø
Clear-EOL	*	clear from cursor position to end of line	6	Ø
Clear-EOS	*	clear from cursor position to end of screen	7	Ø
Highlight		set intensity to high-light	8	Ø
Lowlight		set intensity to low-light	9	Ø
Normlight		set intensity to normal-light	1Ø	Ø
Keybd-on	*	enable keyboard	11	Ø
Keybd-off	*	disable keyboard	12	Ø
Curpd-on		enable cursor pad	13	Ø
Curpd-off		disable cursor pad	14	Ø
Prtec-on	*	begin protected field	15	Ø
Prtec-off	*	end protected field	16	Ø
Blink-on	*	begin blinking characters	17	Ø

Blink-off	*	end blinking characters	18	Ø
Line-send	*	send from cursor position to end of line	19	Ø
Page-send	*	send from cursor position to end of screen	2Ø	Ø
Aux-send	*	transmit screen out auxiliary port	21	Ø
Del-char	*	delete character at present cursor position	22	Ø
Insr-char		insert character at present cursor position	23	Ø
Del-line		delete line at present cursor position	24	Ø
Insr-line		insert line at present cursor position	25	Ø
Formt-on	*	turn on formatted screen	26	Ø
Formt-off	*	turn off formatted screen	27	Ø
Rever-on		begin reverse background field	28	Ø
Rever-off		end reverse background field	29	Ø

Those features marked with an asterisk (*) above are all standard features of a Cromemco Model 31Ø1 System Terminal. Also note from the above chart that the E register is always loaded with Ø to select any special CRT function except cursor addressing. For cursor addressing the D register should contain the column address (1 through 8Ø for Cromemco CRTs) and the E register should contain the row address (1 through 24 for Cromemco CRTs) of the desired cursor position. The system call will generate no error if these values are exceeded. Note: On some CRTs addressing the cursor at a non-existent location may cause it to disappear from the screen.

For reference, the location (1,1) is considered to be the upper left-hand corner and the location (8Ø,24) the lower right-hand corner of the screen.

5.1.3 CDOS Disk System Calls

CDOS divides the disk into regions called files. Files are referenced through file control blocks (FCBs). FCBs are 33 bytes long and have the following format, where each of the numbers below stands for one byte:

FCBDK	Disk descriptor	Ø	(Ø=current disk, 1=drive-A, 2=B, 3=C, 4=D)
FCBFN	File name	1...8	(right-filled with blanks)
FCBFT	File type (extension)	9...11	(right-filled with blanks)
FCBEX	File entry or extent	12	(initially Ø; is incremented by one in every new entry of 16 Kbytes)
	Reserved	13...14	
FCBRC	Record count	15	(total number of 128-byte sectors or records)
FCBMP	Cluster allocation map	16...31	(allocated clusters 2 through 240)
FCBNR	Next record	32	(next record to be read or written; has the value Ø through 127)

Note: The directory entries on the disk consist of 32 byte FCBs. The last byte, FCBNR, which points to the next record, is omitted.

12 (ØCH) - DESELECT CURRENT DISK

The current disk is deselected. The CDOS disk driver can be changed to perform any desired function at this time to deselect the disk. Currently the driver outputs a Ø to port 34H when this function is selected.

13 (ØDH) - RESET CDOS AND SELECT DRIVE A

CDOS is initialized, all disks are logged-off, and drive A is selected as the current drive. The other disks will be logged-on again as soon as they are accessed.

14 (ØEH) - SELECT DISK DRIVE

The disk drive number in the E register is selected as the current disk. The drive number in the E register is Ø for drive A, 1 for drive B, 2 for drive C, or 3 for drive D.

15 (ØFH) - OPEN DISK FILE

The FCB pointed to by the DE register pair is opened to allow reading or writing to the file whose name is specified in the FCB. The A register returns with -1 (ØFFH or 255D) if the file is not found, or the directory block number if the file is found. Block numbers start at Ø and there is one block number for every four directory entries. The DE register pair returns pointing to the directory entry in memory.

16 (1ØH) - CLOSE DISK FILE

The file described by the File Control Block pointed to by the DE register pair is closed, and the disk directory is updated (i.e., the FCB containing updated cluster information is written to the disk). The A register is returned with -1 (FFH) if the file is <u>not</u> found on the drive, or the directory block number if the file <u>is</u> found on the stated drive. The file described by the FCB should have been previously opened or created. A file to which bytes have just been written <u>must</u> be closed using this function or the entire <u>last</u> entry (or extent) will be unable to be read (i.e., no cluster information will be present for this entry in the directory).

17 (11H) - SEARCH DIRECTORY FOR FILENAME

The directory is searched for the first occurrence of the file specified in the FCB pointed to by the DE register pair. ASCII question mark (? - 3FH) in the FCB matches any character. The block number (see description of directory block numbers in ØFH - Open Disk File, above) is returned in the A register if found, if the file is not found -1 (ØFFH or 255D) is returned in A. HL is returned pointing to the directory entry in memory. An important point to note about this call and the one following (12H) is that they will get the directory entry whether it has been erased or not; i.e., these calls do not check to see if a file has been erased. Files are erased by placing a ØE5H in the first byte (FCBDK); the rest of the FCB is left unchanged.

18 (12H) - FIND NEXT DIRECTORY ENTRY

This call is the same as 11H (17) above except that it finds the NEXT occurrence of the filename in the directory. This may be either the next entry of a file occupying several entries (extents), or another filename if the question mark match character (?) is used in the FCB. This call is made after system call 17 and no other disk system function can be executed between these calls.

19 (13H) - DELETE FILE

The file specified by the FCB pointed to by the DE register pair is deleted from the disk directory. ASCII question mark (?) in the FCB matches any character. The number of directory entries deleted is returned in the A register.

20 (14H) - READ NEXT RECORD

The DE register pair points to a successfully OPENED FCB. The next record (128 bytes) is read into the current disk buffer. The FCBNR in the FCB is incremented to read the next record. One of the following codes is returned in the A register:

0 - read completed

1 - end of file

2 - read attempted on unwritten cluster (random access files only)

21 (15H) - WRITE NEXT RECORD

The DE register pair points to a successfully OPENED FCB. The next record (128 bytes) is written into the file from the current disk buffer. The FCBNR in the FCB is incremented to be ready to write the next record. One of the following codes is returned in the A register:

0 - write completed

1 - entry error (attempted to close an unopened entry)

2 - out of disk space (limited to 81K for small; 241K for large)

-1 - (or FFH) out of directory space (limited to 64 entries)

22 (16H) - CREATE FILE

The file specified in the FCB pointed to by the DE register pair is created on the disk. The A register is returned containing the block number of the directory entry (see 0FH - Open Disk File), or -1 (0FFH or 255) if no more directory space is available.

23 (17H) - RENAME FILE

This call will rename a disk file. The DE register pair points to the FCB to be renamed. The old file name and file type are in the first 16 bytes and the new file name and file type are in the second 16 bytes of the FCB. ASCII question mark (?) in the FCB will match with any character. The A register returns containing the number of directory entries renamed.

24 (18H) - GET DISK LOG-IN VECTOR

The A register is returned specifying the disks that are logged in. Each bit represents one disk drive logged in. If the bit is a one, then it is logged in; else it is off-line. The least significant bit is the A drive, next most significant (Bit 1) is drive B, etc. Since there would be no more than four drives, the upper four bits are Ø's.

25 (19H) - GET CURRENT DISK

The number of the current disk drive is returned in the A register. Ø = drive A, 1 = drive B, 2 = drive C, 3 = drive D.

26 (1AH) - SET DISK BUFFER

The buffer pointed to by the DE register pair is used for disk I/O. When a program is loaded, the disk buffer is initially located at 8ØH.

27 (1BH) - GET DISK CLUSTER ALLOCATION MAP

The BC register pair returns pointing to a bit map that corresponds to the allocated clusters on the disk. The DE register pair returns containing the capacity of the current disk in number of clusters. The A register returns containing the number of records or sectors per cluster (8). This system call is used by the STATus utility program.

131 (83H) - READ LOGICAL BLOCK

This system call will read a logical block from the disk without any attention to the files it may contain (i.e., no FCB is specified). A block is defined to be one sector or record of 128 bytes. When this function is called, the DE register pair should contain the block number and the B register should contain the disk number (Ø for current drive, 1-4 for A-D). The high bit of the B register contains a 1 for an interleaved and a Ø for a non-interleaved read. Interleaved means the block which is read is found in the order CDOS stores it (every fifth sector for small disks and every sixth sector for large disks). Non-interleaved means the block which is read is found in sequential order, the order it is physically stored on the disk. The A register is returned with the status of the read according to the following:

Ø - OK
1 - I/O error
2 - illegal request
3 - illegal block

An example will help to illustrate the use of these parameters. CDOS makes use of 716 sectors on the small floppy disks. The block numbers which can legally be loaded into the DE register are Ø through 715 decimal, or Ø through 2CBH. Suppose that DE is loaded with the value 2 and the B register with Ø (current disk, non-interleaved read). Thus, since the sectors are numbered beginning with 1, sector 3 would be read into memory in the disk buffer (located at 8ØH if it has not been changed). The same read with the B register loaded with 8ØH (current disk, interleaved read) would read sector ØBH (the third sector when

they are read every fifth one).

132 (84H) - WRITE LOGICAL BLOCK

This system call will write a logical block or sector to the disk without any attention to the file there (no FCB is specified). The registers are set up and returned in the same way as they are for the Read Logical Block system call.

134 (86H) - FORMAT NAME TO FILE CONTROL BLOCK

This system call will build a File Control Block. The HL register pair points to the start of the input line. The DE register points to the place in memory where the FCB is to be built. The input line is of the format:

```
d:filename.ext
```

where d stands for one of A-D, the filename is up to 8 letters with a 3 letter extension. The FCB is then built from this input line, converting lower case to upper case. The input line is terminated by an ASCII slash (/) or any character with an ASCII value less than 21H (such as a space or carriage return).

On return the HL register pair points to the terminator that ended the build operation. The DE register pair points to the start of the new FCB.

135 (87H) - UPDATE DIRECTORY ENTRY

The last disk I/O function called must have been system call 17 or 18, Search Directory or Find Next Entry. The DE register pair points to the FCB used in the system call 17 or 18. The directory entry is then updated on the disk; this means that the entry is written back to the disk without the user having to specify a block. The user merely specifies a filename when calling 17 or 18. This is useful if it is desired to change a directory entry and write it back to the disk.

139 (8BH) - HOME DRIVE HEAD

The disk drive specified in the B register (Ø for current drive and 1-4 for drives A-D) is sent a command to HOME the head. The disk drive head will return to track Ø.

14Ø (8CH) - EJECT DISKETTE

This call will eject the disk whose number is given in the E register (Ø for current drive and 1-4 for drives A-D, respectively), only if the disk drive is a CROMEMCO Dual Disk Drive System, Model PFD with the eject option. Otherwise, the call will have no effect.

5.1.4 Miscellaneous System Calls

A number of miscellaneous CDOS system calls have been added for the programmer's convenience. These calls are explained in this section.

Ø - PROGRAM ABORT

This call will abort the current program and return control to CDOS. This call has the same effect as jumping to location Ø.

129 (81H) - GET USER-REGISTER POINTER

This call is provided for expansion of CDOS to a multiprogramming system. The BC register pair returns pointing to the user register pointers.

13Ø (82H) - SET USER CONTROL-C ABORT

When CTRL-C (^C) is typed, the system normally aborts and returns control to CDOS. This call allows the programmer to change the address to which control is transferred when CTRL-C is typed (i.e., a user may assign a new function to CTRL-C). The address is given in the DE register pair. Note that if DE contains a zero, the system abort is reset. Jumping to location Ø at any time still causes a return to CDOS, also with the CTRL-C being restored to its original function.

136 (88H) - LINK TO NEW PROGRAM

This enables one command program to call another. The default command-line buffer and default FCBs for the new program must be set up prior to this call if that program expects to be able to use them. The DE register pair should contain the address of the FCB of the new program (which must have an extension of COM). If the new program is NOT found, the A register returns containing -1 (ØFFH or 255). In this case the first 8ØH bytes (from 1ØØH to 17FH) will be destroyed because this is used in reading the directory. If the program is found execution begins at 1ØØH and no return is made to the original program.

137 (89H) - MULTIPLY INTEGERS

This system call provides a 16 bit multiply. The HL and DE register pairs contain the two 16-bit factors, and the answer is returned in register DE (i.e., DE = DE*HL).

138 (8AH) - DIVIDE INTEGERS

This system call provides a 16-bit divide. The HL register pair should contain the dividend, and the DE register pair, the divisor. The quotient is returned in HL, and the remainder in DE (i.e., HL = HL/DE with DE = remainder).

141 (8DH) - GET CDOS VERSION AND RELEASE NUMBERS

This call will return the version number of CDOS in the B register and the release number in the C register.

143 (8FH) - SET CALENDAR DATE

This call is used to store the date (day/mon/yr) in CDOS. Upon entry to this call, the B register contains the day, the D register the month, and the E register the year minus 1900. These values will be stored in locations in CDOS where they may be accessed by user programs (through system call 144) and thus added to listings or other output. No parameters are returned by this call.

The operating system makes no check for the correctness or plausibility of the incoming values; thus, it is up to the user to supply this error-checking. Also, the date is not stored on the disk and is thus volatile (will be lost if the user re-boots or turns off the power).

144 (90H) - READ CALENDAR DATE

This call is used to retrieve the date (day/mon/yr) stored in CDOS by system call 143. The day is returned in the A register, the month in the B register, and the year minus 1900 in the C register. No entry parameters are required other than the value in the C register. Note that the C register is changed by this call unlike most other system calls which preserve C.

This is the function which should be used by a program to recover the last previously stored date from the operating system. Note that if Set Date has not yet been used, Read Date will return the values 00/00/00.

145 (91H) - SET TIME OF DAY

This call is used to store the time of day (sec/min/hr) in CDOS for use by a hardware clock or user program. Upon entry to the call, the B register contains the seconds, the D register the minutes, and the E register the hours in 24-hour time. These values will be stored in locations in CDOS where they may either be accessed and updated by user programs or may in turn be stored in registers of an electronic clock. No parameters are returned by this call.

The operating system makes no check for the correctness or plausibility of the incoming values. It is up to the user to supply this error checking. Note in the I/O Device Drivers that a dummy routine is supplied to Start Clock. This dummy routine is called by the operating system during the Set Time function; thus, users may substitute their own routine in the drivers to initialize a hardware clock.

146 (92H) - READ TIME OF DAY

This call is used to retrieve the time of day (sec/min/hr) stored in CDOS by system call 145. The seconds are returned in the A register, the minutes in the B register, and hours (24 hour time) in the C register. Note that the C register is changed by this call unlike most other system calls which preserve C.

This is the function which should be used by a program to recover the last previously stored time from the operating system. Note that if Set Time has not yet been used, Read Time will return the values 00/00/00.

The I/O Device Drivers contain a dummy routine to Read Clock. This dummy routine is called by CDOS during the Read Time system call. Thus, users may substitute their own routine in the drivers to read the time from a hardware clock and store it in the time registers also supplied in the drivers.

150 (96H) - TURN DRIVE MOTORS OFF

This call is used to turn off the disk drive motors. It may be used by any program which will perform its primary function in memory over a long period of time during which there will be few disk accesses (e.g., an editor or interpreter). No parameters are required on entry or given on return from this call other than the value in the C register.

Note that there is no corollary call to turn the motors on. This will be performed automatically by the operating system the next time any disk

operation is attempted. CDOS will also pause for approximately 1 second after turning on the motors and before accessing the disk only if the Motor Off call has been issued. This is to allow the motors to come up to speed before the disk is accessed.

151 (97H) - SET BOTTOM OF CDOS IN RAM

This call is used to set the bottom address of CDOS to a lower value than the one at which CDOS was originally loaded when it was booted up. The high byte of the address of the new bottom is placed into the E register prior to executing the call. The low byte is assumed Ø; thus, the bottom of CDOS can never be located on any address other than a 256 byte boundary. If the value is -1 (FFH) or any other value greater than the high byte of the original bottom when booting up, CDOS will restore this original bottom address. No parameters are returned by the call.

This function will change the system call jump at locations 5, 6, and 7. Programs using the address at locations 6 and 7 to determine the size of the present User Area will find this area to be reduced in size. A second set of jumps (9 bytes) will be loaded at the new bottom of CDOS which points to the old bottom so that system calls will still execute correctly. Note that CDOS is in no way relocated by this function and will reside in the same memory space as it did previously. The purpose of the call is to make it possible to attach a permanent patch space to CDOS for programs which are to become a permanent part of the operating system for as long as it resides in memory. The only way the patch space may be removed is by a second Set Bottom call.

5.1.5 Summary of CDOS System Calls

Following is a summary table listing all the system calls described in Chapter 5 along with their entry and return parameters. The system calls are listed in numerical order, i.e., by order of the number which is loaded into the C register to achieve the desired function.

NUMBER	FUNCTION	ENTRY PARAMETERS	RETURN PARAMETERS
Ø	PROGRAM ABORT	none	none
1	READ CONSOLE (with echo)	none	A = character (parity bit reset)
2	WRITE CONSOLE	E = character	none
3	READ READER	none	A = character
4	WRITE PUNCH	E = character	none
5	WRITE LIST	E = character	none
6	(not used presently - reserved for expansion)		
7	GET I/O BYTE	none	A = I/O byte
8	SET I/O BYTE	E = I/O byte	none
9	PRINT BUFFERED LINE	DE = buffer address	none
1Ø (ØAH)	INPUT BUFFERED LINE	DE = buffer address	none
11 (ØBH)	TEST CONSOLE READY	none	A = -1 (FFH) if ready A = Ø if not ready
12 (ØCH)	DESELECT CURRENT DISK	none	none
13 (ØDH)	RESET CDOS AND SELECT DRIVE A	none	none
14 (ØEH)	SELECT CURRENT DISK	E = disk drive no.	none
15 (ØFH)	OPEN DISK FILE	DE = FCB address	A = directory block A = -1 (FFH) if not found
16 (1ØH)	CLOSE DISK FILE	DE = FCB address	A = directory block A = -1 (FFH) if not found
17 (11H)	SEARCH DIRECTORY FOR FILENAME	DE = FCB address	A = directory block A = -1 (FFH) if not found
18 (12H)	FIND NEXT ENTRY IN DIRECTORY	DE = FCB address	A = directory block A = -1 (FFH) if not found
19 (13H)	DELETE FILE	DE = FCB address	A = number of entries deleted
2Ø (14H)	READ NEXT RECORD	DE = FCB address	A = Ø if OK A = 1 if end of file A = 2 if tried to read unwritten records

21 (15H)	WRITE NEXT RECORD	DE = FCB address	A = Ø if OK A = 1 if entry error A = 2 if out of disk space A = -1 (FFH) if out of directory space
22 (16H)	CREATE FILE	DE = FCB address	A = directory block A = -1 (FFH) if out of directory space
23 (17H)	RENAME FILE	DE = FCB address	A = number of entries renamed
24 (18H)	GET DISK LOG-IN VECTOR	none	A = those disks currently logged-in
25 (19H)	CURRENT DISK	none	A = disk drive number
26 (1AH)	SET DISK BUFFER	DE = buffer address	none
27 (1BH)	DISK CLUSTER ALLOCATION MAP	none	BC = address of bitmap DE = number of clusters A = sectors/cluster
128 (8ØH)	READ CONSOLE (with no echo)	none	A = character
129 (81H)	GET USER REGISTER POINTER	none	BC = pointer to user register pointers
13Ø (82H)	SET USER CTRL-C ABORT	DE = address of ^C handler (Ø to reset; -1 to disable)	none
131 (83H)	READ LOGICAL BLOCK	DE = block number B = drive number B top bit = 1 if interleaved	A = Ø if OK A = 1 if I/O error A = 2 if illegal request A = 3 if illegal block
132 (84H)	WRITE LOGICAL BLOCK	DE = block number B = drive number B top bit = 1 if interleaved	A = Ø if OK A = 1 if I/O error A = 2 if illegal request A = 3 if illegal block
133 (85H)	(not used presently - reserved for expansion)		
134 (86H)	FORMAT NAME TO FILE CONTROL BLOCK	HL = address of string DE = FCB address	HL = address of terminator DE = FCB address
135 (87H)	UPDATE DIRECTORY ENTRY	DE = FCB address	none
136 (88H)	LINK TO PROGRAM	DE = FCB address	A = -1 (FFH) if error; else execute at 1ØØH
137 (89H)	MULTIPLY INTEGERS	DE = factor 1 HL = factor 2	DE = product
138 (8AH)	DIVIDE INTEGERS	HL = dividend DE = divisor	HL = quotient DE = remainder

139	(8BH)	HOME DRIVE	B = drive number	none
140	(8CH)	EJECT DISKETTE	E = drive number	none
141	(8DH)	GET VERSION OF OPERATING SYSTEM	none	B = version-number C = release-number
142	(8EH)	SET SPECIAL CRT FUNCTION	D = column address/ special function E = row address/0	none
143	(8FH)	SET DATE	B = day D = month E = year-1900	none
144	(90H)	READ DATE	none	A = day B = month C = year-1900
145	(91H)	SET TIME OF DAY	B = seconds D = minutes E = hours (24 hr. time)	none
146	(92H)	READ TIME OF DAY	none	A = seconds B = minutes C = hours (24 hr. time)
147	(93H)	SET PROGRAM RETURN CODE	E = return code for next program	A = previously set return code
148	(94H)	SET FILE ATTRIBUTES	DE = FCB address B = new attributes	none
149	(95H)	READ DISK LABEL	none	DE = FCB address
150	(96H)	TURN MOTORS OFF	none	none
151	(97H)	SET BOTTOM OF CDOS IN RAM	E = high byte of address of bottom of CDOS	none

5.2 CDOS-CP/M Compatibility

The Cromemco Disk Operating System (CDOS)* is an original product designed and written in Z8Ø machine code by Cromemco, Inc. for its own line of microcomputers. However, due to the large number of programs currently available to run under the CP/M** operating system, CDOS was designed to be upwards CP/M-compatible. Cromemco is licensed by Digital Research, the originator of CP/M, for use of the CP/M data structures and user interface. This means that most programs written for CP/M (versions up to and including 1.3) will run without modification under CDOS. This also means that programs written for CDOS will not generally run under CP/M.

There are several advantages to end-users which result from this compatibility. First, users of Cromemco machines are able to draw on the large library of existing CP/M and CP/M compatible programs available on the market. Second, users familiar with CP/M can easily move up to CDOS taking advantage of the many additional features available with CDOS.

The enhancements contained in CDOS but not CP/M are primarily visible in the system calls. CDOS has added a number of new system calls to allow the user even more flexible means of device and disk I/O. System calls are in all cases executed by first loading the C register with the number of the call. In the case of the non-CP/M calls, the parity bit of the 8-bit quantity in C is set to allow for a new range of 128. This is apparent in the Summary of CDOS System Calls, Section 5.1.5.

* CDOS is a Trademark of Cromemco, Inc.
Mountain View, California

**CP/M is a Trademark of Digital Research, Inc.
Pacific Grove, California

5.3 CDOS Memory Allocation

CDOS resides in high memory. It reserves memory below 100H for its own use. The user is left all memory from 100H to the beginning of CDOS (see below).

A program with the three-letter extension COM can be loaded and executed by typing the program name. The program must have its origin at 100H because that is where CDOS loads and executes it. (Note that when saving files that have been linked using the CROMEMCO Linker, they can be LINKed anywhere using the /P option. This is because LINK automatically puts the correct jump instruction at 100H.) After it is loaded, the program can use any memory at all. Note however that if it alters the CDOS areas, it will have no way of communicating with the disk or returning to CDOS. (CDOS would have to be reloaded by resetting the computer.)

When loaded, CDOS places a jump instruction at bytes 0, 1 and 2. If a jump is made to location 0, the CDOS warm start, control will be returned with the prompt for the current drive (e.g., A). Command lines may then be entered from the console keyboard. CDOS places another jump instruction at locations 5, 6 and 7. The normal way to make system requests of CDOS is to call location 5 (refer to Section 5.1). The address stored at locations 6 and 7 is the address of the beginning of CDOS and thus marks the upper limit of user memory.

The following address map describes the memory area from 0 to 0FFH. All addresses are in hex.

0...2	CDOS re-entry
3	I/O byte
4	reserved
5...7	system request call
8...40	interrupt vectors
40...5B	reserved
5C...6B	default File Control Block 1 (FCB-1)
6C...7B	default File Control Block 2 (FCB-2)
7C...7F	reserved
80...FF	default command-line buffer

When a COM program is run by typing the program

name on the console, the default command-line buffer and default file control blocks are used as follows. FCB-1 will contain the first filename, if any, typed after the program name. FCB-2 will contain the second filename, if any. The default buffer will contain the entire command line following the program name. For example, if this command line is typed:

PROG FILE1.Z80 FILE2.COM

CDOS will place "FILE1Z80"in FCB-1, "FILE2COM" in FCB-2, "FILE1.Z80 FILE2.COM" in the command-line buffer, and load and execute PROG.COM at 100H. Note that the second FCB starts before the end of the first FCB. Before using FCB-1, FCB-2 should be moved. If it is not moved, part of FCB-2 will be destroyed.

The command line which is placed in the default buffer can be used to send more than two filenames to a program, or to start execution of a program with various options specified. For the following command line:

PROG FILE1.Z80 FILE2.COM OPTION1 OPTION2

the string of ASCII characters "FILE1.Z80 FILE2.COM OPTION1 OPTION2" will be stored beginning at location 81H. The byte at location 80H will contain the length of the string. The byte following the string will contain a null (00). PROG.COM can then look at the command line stored in the default buffer to determine which options were specified.

When a program is loaded, the disk buffer is set to 80H, which is the default command buffer. If the disk is then read to or written from, this buffer will be altered. The program must either reset the disk buffer to another area or move the command line before accessing the disk, if it is desired to save the command line.

5.4 Cluster, Track, and Sector Numbers

For system housekeeping, CDOS divides diskettes into Clusters. A cluster stores 1Ø24 (1K or 4ØØH) bytes. A large disk contains 243 clusters in the file area. A small disk contains 83 clusters in the file area. On both large and small disks, the first two clusters are reserved for the directory. Clusters are numbered starting with zero.

CDOS further subdivides each cluster into 8 records, each containing 128 (8ØH) bytes. Under CDOS, the disk is written to and read from in one record units.

Although files are written and read by 128 byte (one record) units, one cluster can not be divided between files. Even if a file is only 17 bytes long, it will still be viewed by CDOS as occupying 1K bytes. Unless the file is erased, or combined with another file, the unoccupied balance of the cluster will not be able to be accessed by CDOS for the use of another file. This is why the DIRectory command lists file sizes in 1K increments.

A Track is a physically defined circular path which is concentric with the hole in the center of the disk. It is defined by its distance from the center of the disk. With the read/write head of the disk drive located on a given track, data may be read from or written to that track. A large disk uses 77 tracks, the first two (numbered zero and one) being dedicated to the system (System Area). A small disk uses 4Ø tracks, the first three (numbered zero, one, and two) being dedicated to the system (System Area).

A Sector is a subdivision of a track. A track on a large disk is divided into 26 sectors while a small disk track is divided into 18 sectors. Sectors are numbered starting from number one and each sector holds one record or 128 (8ØH) bytes.

The 4FDC Floppy Disk Controller accepts instructions using track and sector numbers. CDOS keeps track of files on the disk by means of cluster numbers. When it is necessary to access the disk, CDOS converts its cluster numbers into the 4FDC's track and sector numbers.

The following tables will allow the user to perform

these conversions:

1) small disk - convert track and sector numbers to cluster number

2) large disk - convert track and sector numbers to cluster number

3) small disk - convert cluster number to track and sector numbers

4) large disk - convert cluster number to track and sector numbers

MAPPING OF TRACKS/SECTORS TO CLUSTERS - LARGE DISKS

Tracks	Sectors 01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A
02	00	01	01	02	00	02	00	01	01	02	00	02	00	01	01	03	00	02	00	02	01	03	01	02	00	02
03	03	04	04	06	03	05	03	05	04	06	04	05	03	05	04	06	04	05	03	05	04	06	04	05	03	05
04	06	08	07	09	07	08	06	08	07	09	07	08	06	08	07	09	07	09	06	08	08	09	07	09	07	08
05	09	0B	0A	0C	0A	0C	09	0B	0B	0C	0A	0C	0A	0B	0B	0C	0A	0C	0A	0B	0B	0C	0A	0C	0A	0B
06	0D	0E	0E	0F	0D	0F	0D	0E	0E	0F	0D	0F	0D	0E	0E	10	0D	0F	0D	0F	0E	10	0E	0F	0D	0F
07	10	11	11	13	10	12	10	12	11	13	11	12	10	12	11	13	11	12	10	12	11	13	11	12	10	12
08	13	15	14	16	14	15	13	15	14	16	14	15	13	15	14	16	14	16	13	15	15	16	14	16	14	15
09	16	18	17	19	17	19	16	18	18	19	17	19	17	18	18	19	17	19	17	18	18	19	17	19	17	18
0A	1A	1B	1B	1C	1A	1C	1A	1B	1B	1C	1A	1C	1A	1B	1B	1D	1A	1C	1A	1C	1B	1D	1B	1C	1A	1C
0B	1D	1E	1E	20	1D	1F	1D	1F	1E	20	1E	1F	1D	1F	1E	20	1E	1F	1D	1F	1E	20	1E	1F	1D	1F
0C	20	22	21	23	21	22	20	22	21	23	21	22	20	22	21	23	21	23	20	22	22	23	21	23	21	22
0D	23	25	24	26	24	26	23	25	25	26	24	26	24	25	25	26	24	26	24	25	25	26	24	26	24	25
0E	27	28	28	29	27	29	27	28	28	29	27	29	27	28	28	2A	27	29	27	29	28	2A	28	29	27	29
0F	2A	2B	2B	2D	2A	2C	2A	2C	2B	2D	2B	2C	2A	2C	2B	2D	2B	2C	2A	2C	2B	2D	2B	2C	2A	2C
10	2D	2F	2E	30	2E	2F	2D	2F	2E	30	2E	2F	2D	2F	2E	30	2E	30	2D	2F	2F	30	2E	30	2E	2F
11	30	32	31	33	31	33	30	32	32	33	31	33	31	32	32	33	31	33	31	32	32	33	31	33	31	32
12	34	35	35	36	34	36	34	35	35	36	34	36	34	35	35	37	34	36	34	36	35	37	35	36	34	36
13	37	38	38	3A	37	39	37	39	38	3A	38	39	37	39	38	3A	38	39	37	39	38	3A	38	39	37	39
14	3A	3C	3B	3D	3B	3C	3A	3C	3B	3D	3B	3C	3A	3C	3B	3D	3B	3D	3A	3C	3C	3D	3B	3D	3B	3C
15	3D	3F	3E	40	3E	40	3D	3F	3F	40	3E	40	3E	3F	3F	40	3E	40	3E	3F	3F	40	3E	40	3E	3F
16	41	42	42	43	41	43	41	42	42	43	41	43	41	42	42	44	41	43	41	43	42	44	42	43	41	43
17	44	45	45	47	44	46	44	46	45	47	45	46	44	46	45	47	45	46	44	46	45	47	45	46	44	46
18	47	49	48	4A	48	49	47	49	48	4A	48	49	47	49	48	4A	48	4A	47	49	49	4A	48	4A	48	49
19	4A	4C	4B	4D	4B	4D	4A	4C	4C	4D	4B	4D	4B	4C	4C	4D	4B	4D	4B	4C	4C	4D	4B	4D	4B	4C
1A	4E	4F	4F	50	4E	50	4E	4F	4F	50	4E	50	4E	4F	4F	51	4E	50	4E	50	4F	51	4F	50	4E	50
1B	51	52	52	54	51	53	51	53	52	54	52	53	51	53	52	54	52	53	51	53	52	54	52	53	51	53
1C	54	56	55	57	55	56	54	56	55	57	55	56	54	56	55	57	55	57	54	56	56	57	55	57	55	56
1D	57	59	58	5A	58	5A	57	59	59	5A	58	5A	58	59	59	5A	58	5A	58	59	59	5A	58	5A	58	59
1E	5B	5C	5C	5D	5B	5D	5B	5C	5C	5D	5B	5D	5B	5C	5C	5E	5B	5D	5B	5D	5C	5E	5C	5D	5B	5D
1F	5E	5F	5F	61	5E	60	5E	60	5F	61	5F	60	5E	60	5F	61	5F	60	5E	60	5F	61	5F	60	5E	60
20	61	63	62	64	62	63	61	63	62	64	62	63	61	63	62	64	62	64	61	63	63	64	62	64	62	63
21	64	66	65	67	65	67	64	66	66	67	65	67	65	66	66	67	65	67	65	66	66	67	65	67	65	66
22	68	69	69	6A	68	6A	68	69	69	6A	68	6A	68	69	69	6B	68	6A	68	6A	69	6B	69	6A	68	6A
23	6B	6C	6C	6E	6B	6D	6B	6D	6C	6E	6C	6D	6B	6D	6C	6E	6C	6D	6B	6D	6C	6E	6C	6D	6B	6D
24	6E	70	6F	71	6F	70	6E	70	6F	71	6F	70	6E	70	6F	71	6F	71	6E	70	70	71	6F	71	6F	70
25	71	73	72	74	72	74	71	73	73	74	72	74	72	73	73	74	72	74	72	73	73	74	72	74	72	73
26	75	76	76	77	75	77	75	76	76	77	75	77	75	76	76	78	75	77	75	77	76	78	76	77	75	77
27	78	79	79	7B	78	7A	78	7A	79	7B	79	7A	78	7A	79	7B	79	7A	78	7A	79	7B	79	7A	78	7A
28	7B	7D	7C	7E	7C	7D	7B	7D	7C	7E	7C	7D	7B	7D	7C	7E	7C	7E	7B	7D	7D	7E	7C	7E	7C	7D
29	7E	80	7F	81	7F	81	7E	80	80	81	7F	81	7F	80	80	81	7F	81	7F	80	80	81	7F	81	7F	80
2A	82	83	83	84	82	84	82	83	83	84	82	84	82	83	83	85	82	84	82	84	83	85	83	84	82	84
2B	85	86	86	88	85	87	85	87	86	88	86	87	85	87	86	88	86	87	85	87	86	88	86	87	85	87
2C	88	8A	89	8B	89	8A	88	8A	89	8B	89	8A	88	8A	89	8B	89	8B	88	8A	8A	8B	89	8B	89	8A
2D	8B	8D	8C	8E	8C	8E	8B	8D	8D	8E	8C	8E	8C	8D	8D	8E	8C	8E	8C	8D	8D	8E	8C	8E	8C	8D
2E	8F	90	90	91	8F	91	8F	90	90	91	8F	91	8F	90	90	92	8F	91	8F	91	90	92	90	91	8F	91
2F	92	93	93	95	92	94	92	94	93	95	93	94	92	94	93	95	93	94	92	94	93	95	93	94	92	94
30	95	97	96	98	96	97	95	97	96	98	96	97	95	97	96	98	96	98	95	97	97	98	96	98	96	97
31	98	9A	99	9B	99	9B	98	9A	9A	9B	99	9B	99	9A	9A	9B	99	9B	99	9A	9A	9B	99	9B	99	9A
32	9C	9D	9D	9E	9C	9E	9C	9D	9D	9E	9C	9E	9C	9D	9D	9F	9C	9E	9C	9E	9D	9F	9D	9E	9C	9E
33	9F	A0	A0	A2	9F	A1	9F	A1	A0	A2	A0	A1	9F	A1	A0	A2	A0	A1	9F	A1	A0	A2	A0	A1	9F	A1
34	A2	A4	A3	A5	A3	A4	A2	A4	A3	A5	A3	A4	A2	A4	A3	A5	A3	A5	A2	A4	A4	A5	A3	A5	A3	A4
35	A5	A7	A6	A8	A6	A8	A5	A7	A7	A8	A6	A8	A6	A7	A7	A8	A6	A8	A6	A7	A7	A8	A6	A8	A6	A7
36	A9	AA	AA	AB	A9	AB	A9	AA	AA	AB	A9	AB	A9	AA	AA	AC	A9	AB	A9	AB	AA	AC	AA	AB	A9	AB
37	AC	AD	AD	AF	AC	AE	AC	AE	AD	AF	AD	AE	AC	AE	AD	AF	AD	AE	AC	AE	AD	AF	AD	AE	AC	AE
38	AF	B1	B0	B2	B0	B1	AF	B1	B0	B2	B0	B1	AF	B1	B0	B2	B0	B2	AF	B1	B1	B2	B0	B2	B0	B1
39	B2	B4	B3	B5	B3	B5	B2	B4	B4	B5	B3	B5	B3	B4	B4	B5	B3	B5	B3	B4	B4	B5	B3	B5	B3	B4
3A	B6	B7	B7	B8	B6	B8	B6	B7	B7	B8	B6	B8	B6	B7	B7	B9	B6	B8	B6	B8	B7	B9	B7	B8	B6	B8
3B	B9	BA	BA	BC	B9	BB	B9	BB	BA	BC	BA	BB	B9	BB	BA	BC	BA	BB	B9	BB	BA	BC	BA	BB	B9	BB
3C	BC	BE	BD	BF	BD	BE	BC	BE	BD	BF	BD	BE	BC	BE	BD	BF	BD	BF	BC	BE	BE	BF	BD	BF	BD	BE
3D	BF	C1	C0	C2	C0	C2	BF	C1	C1	C2	C0	C2	C0	C1	C1	C2	C0	C2	C0	C1	C1	C2	C0	C2	C0	C1
3E	C3	C4	C4	C5	C3	C5	C3	C4	C4	C5	C3	C5	C3	C4	C4	C6	C3	C5	C3	C5	C4	C6	C4	C5	C3	C5
3F	C6	C7	C7	C9	C6	C8	C6	C8	C7	C9	C7	C8	C6	C8	C7	C9	C7	C8	C6	C8	C7	C9	C7	C8	C6	C8
40	C9	CB	CA	CC	CA	CB	C9	CB	CA	CC	CA	CB	C9	CB	CA	CC	CA	CC	C9	CB	CB	CC	CA	CC	CA	CB
41	CC	CE	CD	CF	CD	CF	CC	CE	CE	CF	CD	CF	CD	CE	CE	CF	CD	CF	CD	CE	CE	CF	CD	CF	CD	CE
42	D0	D1	D1	D2	D0	D2	D0	D1	D1	D2	D0	D2	D0	D1	D1	D3	D0	D2	D0	D2	D1	D3	D1	D2	D0	D2
43	D3	D4	D4	D6	D3	D5	D3	D5	D4	D6	D4	D5	D3	D5	D4	D6	D4	D5	D3	D5	D4	D6	D4	D5	D3	D5
44	D6	D8	D7	D9	D7	D8	D6	D8	D7	D9	D7	D8	D6	D8	D7	D9	D7	D9	D6	D8	D8	D9	D7	D9	D7	D8
45	D9	DB	DA	DC	DA	DC	D9	DB	DB	DC	DA	DC	DA	DB	DB	DC	DA	DC	DA	DB	DB	DC	DA	DC	DA	DB
46	DD	DE	DE	DF	DD	DF	DD	DE	DE	DF	DD	DF	DD	DE	DE	E0	DD	DF	DD	DF	DE	E0	DE	DF	DD	DF
47	E0	E1	E1	E3	E0	E2	E0	E2	E1	E3	E1	E2	E0	E2	E1	E3	E1	E2	E0	E2	E1	E3	E1	E2	E0	E2
48	E3	E5	E4	E6	E4	E5	E3	E5	E4	E6	E4	E5	E3	E5	E4	E6	E4	E6	E3	E5	E5	E6	E4	E6	E4	E5
49	E6	E8	E7	E9	E7	E9	E6	E8	E8	E9	E7	E9	E7	E8	E8	E9	E7	E9	E7	E8	E8	E9	E7	E9	E7	E8
4A	EA	EB	EB	EC	EA	EC	EA	EB	EB	EC	EA	EC	EA	EB	EB	ED	EA	EC	EA	EC	EB	ED	EB	EC	EA	EC
4B	ED	EE	EE	F0	ED	EF	ED	EF	EE	F0	EE	EF	ED	EF	EE	F0	EE	EF	ED	EF	EE	F0	EE	EF	ED	EF
4C	F0	F2	F1	--	F1	F2	F0	F2	F1	--	F1	F2	F0	F2	F1	--	F1	--	F0	F2	F2	--	F1	--	F1	F2

MAPPING OF TRACKS/SECTORS TO CLUSTERS - SMALL DISKS

Tracks \ Sectors	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12
03	00	01	00	01	01	00	01	00	02	01	00	01	00	02	01	00	01	00
04	02	03	02	04	03	02	03	02	04	03	02	03	03	04	03	02	04	03
05	04	05	05	06	05	04	06	05	06	05	04	06	05	06	05	04	06	05
06	06	08	07	08	07	06	08	07	08	07	07	08	07	08	08	07	08	07
07	09	0A	09	0A	0A	09	0A	09	0B	0A	09	0A	09	0B	0A	09	0A	09
08	0B	0C	0B	0D	0C	0B	0C	0B	0D	0C	0B	0C	0C	0D	0C	0B	0D	0C
09	0D	0E	0E	0F	0E	0D	0F	0E	0F	0E	0D	0F	0E	0F	0E	0D	0F	0E
0A	0F	11	10	11	10	0F	11	10	11	10	10	11	10	11	11	10	11	10
0B	12	13	12	13	13	12	13	12	14	13	12	13	12	14	13	12	13	12
0C	14	15	14	16	15	14	15	14	16	15	14	15	15	16	15	14	16	15
0D	16	17	17	18	17	16	18	17	18	17	16	18	17	18	17	16	18	17
0E	18	1A	19	1A	19	18	1A	19	1A	19	19	1A	19	1A	1A	19	1A	19
0F	1B	1C	1B	1C	1C	1B	1C	1B	1D	1C	1B	1C	1B	1D	1C	1B	1C	1B
10	1D	1E	1D	1F	1E	1D	1E	1D	1F	1E	1D	1E	1E	1F	1E	1D	1F	1E
11	1F	20	20	21	20	1F	21	20	21	20	1F	21	20	21	20	1F	21	20
12	21	23	22	23	22	21	23	22	23	22	22	23	22	23	23	22	23	22
13	24	25	24	25	25	24	25	24	26	25	24	25	24	26	25	24	25	24
14	26	27	26	28	27	26	27	26	28	27	26	27	27	28	27	26	28	27
15	28	29	29	2A	29	28	2A	29	2A	29	28	2A	29	2A	29	28	2A	29
16	2A	2C	2B	2C	2B	2A	2C	2B	2C	2B	2B	2C	2B	2C	2C	2B	2C	2B
17	2D	2E	2D	2E	2E	2D	2E	2D	2F	2E	2D	2E	2D	2F	2E	2D	2E	2D
18	2F	30	2F	31	30	2F	30	2F	31	30	2F	30	30	31	30	2F	31	30
19	31	32	32	33	32	31	33	32	33	32	31	33	32	33	32	31	33	32
1A	33	35	34	35	34	33	35	34	35	34	34	35	34	35	35	34	35	34
1B	36	37	36	37	37	36	37	36	38	37	36	37	36	38	37	36	37	36
1C	38	39	38	3A	39	38	39	38	3A	39	38	39	39	3A	39	38	3A	39
1D	3A	3B	3B	3C	3B	3A	3C	3B	3C	3B	3A	3C	3B	3C	3B	3A	3C	3B
1E	3C	3E	3D	3E	3D	3C	3E	3D	3E	3D	3D	3E	3D	3E	3E	3D	3E	3D
1F	3F	40	3F	40	40	3F	40	3F	41	40	3F	40	3F	41	40	3F	40	3F
20	41	42	41	43	42	41	42	41	43	42	41	42	42	43	42	41	43	42
21	43	44	44	45	44	43	45	44	45	44	43	45	44	45	44	43	45	44
22	45	47	46	47	46	45	47	46	47	46	46	47	46	47	47	46	47	46
23	48	49	48	49	49	48	49	48	4A	49	48	49	48	4A	49	48	49	48
24	4A	4B	4A	4C	4B	4A	4B	4A	4C	4B	4A	4B	4B	4C	4B	4A	4C	4B
25	4C	4D	4D	4E	4D	4C	4E	4D	4E	4D	4C	4E	4D	4E	4D	4C	4E	4D
26	4E	50	4F	50	4F	4E	50	4F	50	4F	4F	50	4F	50	50	4F	50	4F
27	51	52	51	52	52	51	52	51	--	52	51	52	51	--	52	51	52	51

MAPPING OF CLUSTERS TO TRACKS/SECTORS - LARGE DISKS

Cluster	Sectors Within Each Cluster							
	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc
ØØ	Ø2,Ø1	Ø2,Ø7	Ø2,ØD	Ø2,13	Ø2,19	Ø2,Ø5	Ø2,ØB	Ø2,11
Ø1	Ø2,17	Ø2,Ø3	Ø2,Ø9	Ø2,ØF	Ø2,15	Ø2,Ø2	Ø2,Ø8	Ø2,ØE
Ø2	Ø2,14	Ø2,1A	Ø2,Ø6	Ø2,ØC	Ø2,12	Ø2,18	Ø2,Ø4	Ø2,ØA
Ø3	Ø2,1Ø	Ø2,16	Ø3,Ø1	Ø3,Ø7	Ø3,ØD	Ø3,13	Ø3,19	Ø3,Ø5
Ø4	Ø3,ØB	Ø3,11	Ø3,17	Ø3,Ø3	Ø3,Ø9	Ø3,ØF	Ø3,15	Ø3,Ø2
Ø5	Ø3,Ø8	Ø3,ØE	Ø3,14	Ø3,1A	Ø3,Ø6	Ø3,ØC	Ø3,12	Ø3,18
Ø6	Ø3,Ø4	Ø3,ØA	Ø3,1Ø	Ø3,16	Ø4,Ø1	Ø4,Ø7	Ø4,ØD	Ø4,13
Ø7	Ø4,19	Ø4,Ø5	Ø4,ØB	Ø4,11	Ø4,17	Ø4,Ø3	Ø4,Ø9	Ø4,ØF
Ø8	Ø4,15	Ø4,Ø2	Ø4,Ø8	Ø4,ØE	Ø4,14	Ø4,1A	Ø4,Ø6	Ø4,ØC
Ø9	Ø4,12	Ø4,18	Ø4,Ø4	Ø4,ØA	Ø4,1Ø	Ø4,16	Ø5,Ø1	Ø5,Ø7
ØA	Ø5,ØD	Ø5,13	Ø5,19	Ø5,Ø5	Ø5,ØB	Ø5,11	Ø5,17	Ø5,Ø3
ØB	Ø5,Ø9	Ø5,ØF	Ø5,15	Ø5,Ø2	Ø5,Ø8	Ø5,ØE	Ø5,14	Ø5,1A
ØC	Ø5,Ø6	Ø5,ØC	Ø5,12	Ø5,18	Ø5,Ø4	Ø5,ØA	Ø5,1Ø	Ø5,16
ØD	Ø6,Ø1	Ø6,Ø7	Ø6,ØD	Ø6,13	Ø6,19	Ø6,Ø5	Ø6,ØB	Ø6,11
ØE	Ø6,17	Ø6,Ø3	Ø6,Ø9	Ø6,ØF	Ø6,15	Ø6,Ø2	Ø6,Ø8	Ø6,ØE
ØF	Ø6,14	Ø6,1A	Ø6,Ø6	Ø6,ØC	Ø6,12	Ø6,18	Ø6,Ø4	Ø6,ØA
1Ø	Ø6,1Ø	Ø6,16	Ø7,Ø1	Ø7,Ø7	Ø7,ØD	Ø7,13	Ø7,19	Ø7,Ø5
11	Ø7,ØB	Ø7,11	Ø7,17	Ø7,Ø3	Ø7,Ø9	Ø7,ØF	Ø7,15	Ø7,Ø2
12	Ø7,Ø8	Ø7,ØE	Ø7,14	Ø7,1A	Ø7,Ø6	Ø7,ØC	Ø7,12	Ø7,18
13	Ø7,Ø4	Ø7,ØA	Ø7,1Ø	Ø7,16	Ø8,Ø1	Ø8,Ø7	Ø8,ØD	Ø8,13
14	Ø8,19	Ø8,Ø5	Ø8,ØB	Ø8,11	Ø8,17	Ø8,Ø3	Ø8,Ø9	Ø8,ØF
15	Ø8,15	Ø8,Ø2	Ø8,Ø8	Ø8,ØE	Ø8,14	Ø8,1A	Ø8,Ø6	Ø8,ØC
16	Ø8,12	Ø8,18	Ø8,Ø4	Ø8,ØA	Ø8,1Ø	Ø8,16	Ø9,Ø1	Ø9,Ø7
17	Ø9,ØD	Ø9,13	Ø9,19	Ø9,Ø5	Ø9,ØB	Ø9,11	Ø9,17	Ø9,Ø3
18	Ø9,Ø9	Ø9,ØF	Ø9,15	Ø9,Ø2	Ø9,Ø8	Ø9,ØE	Ø9,14	Ø9,1A
19	Ø9,Ø6	Ø9,ØC	Ø9,12	Ø9,18	Ø9,Ø4	Ø9,ØA	Ø9,1Ø	Ø9,16
1A	ØA,Ø1	ØA,Ø7	ØA,ØD	ØA,13	ØA,19	ØA,Ø5	ØA,ØB	ØA,11
1B	ØA,17	ØA,Ø3	ØA,Ø9	ØA,ØF	ØA,15	ØA,Ø2	ØA,Ø8	ØA,ØE
1C	ØA,14	ØA,1A	ØA,Ø6	ØA,ØC	ØA,12	ØA,18	ØA,Ø4	ØA,ØA
1D	ØA,1Ø	ØA,16	ØB,Ø1	ØB,Ø7	ØB,ØD	ØB,13	ØB,19	ØB,Ø5
1E	ØB,ØB	ØB,11	ØB,17	ØB,Ø3	ØB,Ø9	ØB,ØF	ØB,15	ØB,Ø2
1F	ØB,Ø8	ØB,ØE	ØB,14	ØB,1A	ØB,Ø6	ØB,ØC	ØB,12	ØB,18
2Ø	ØB,Ø4	ØB,ØA	ØB,1Ø	ØB,16	ØC,Ø1	ØC,Ø7	ØC,ØD	ØC,13
21	ØC,19	ØC,Ø5	ØC,ØB	ØC,11	ØC,17	ØC,Ø3	ØC,Ø9	ØC,ØF
22	ØC,15	ØC,Ø2	ØC,Ø8	ØC,ØE	ØC,14	ØC,1A	ØC,Ø6	ØC,ØC
23	ØC,12	ØC,18	ØC,Ø4	ØC,ØA	ØC,1Ø	ØC,16	ØD,Ø1	ØD,Ø7
24	ØD,ØD	ØD,13	ØD,19	ØD,Ø5	ØD,ØB	ØD,11	ØD,17	ØD,Ø3
25	ØD,Ø9	ØD,ØF	ØD,15	ØD,Ø2	ØD,Ø8	ØD,ØE	ØD,14	ØD,1A
26	ØD,Ø6	ØD,ØC	ØD,12	ØD,18	ØD,Ø4	ØD,ØA	ØD,1Ø	ØD,16
27	ØE,Ø1	ØE,Ø7	ØE,ØD	ØE,13	ØE,19	ØE,Ø5	ØE,ØB	ØE,11

Mapping of Clusters to Tracks/Sectors - Large Disks (cont.)

Cluster	Sectors Within Each Cluster							
	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc
28	ØE,17	ØE,Ø3	ØE,Ø9	ØE,ØF	ØE,15	ØE,Ø2	ØE,Ø8	ØE,ØE
29	ØE,14	ØE,1A	ØE,Ø6	ØE,ØC	ØE,12	ØE,18	ØE,Ø4	ØE,ØA
2A	ØE,1Ø	ØE,16	ØF,Ø1	ØF,Ø7	ØF,ØD	ØF,13	ØF,19	ØF,Ø5
2B	ØF,ØB	ØF,11	ØF,17	ØF,Ø3	ØF,Ø9	ØF,ØF	ØF,15	ØF,Ø2
2C	ØF,Ø8	ØF,ØE	ØF,14	ØF,1A	ØF,Ø6	ØF,ØC	ØF,12	ØF,18
2D	ØF,Ø4	ØF,ØA	ØF,1Ø	ØF,16	1Ø,Ø1	1Ø,Ø7	1Ø,ØD	1Ø,13
2E	1Ø,19	1Ø,Ø5	1Ø,ØB	1Ø,11	1Ø,17	1Ø,Ø3	1Ø,Ø9	1Ø,ØF
2F	1Ø,15	1Ø,Ø2	1Ø,Ø8	1Ø,ØE	1Ø,14	1Ø,1A	1Ø,Ø6	1Ø,ØC
3Ø	1Ø,12	1Ø,18	1Ø,Ø4	1Ø,ØA	1Ø,1Ø	1Ø,16	11,Ø1	11,Ø7
31	11,ØD	11,13	11,19	11,Ø5	11,ØB	11,11	11,17	11,Ø3
32	11,Ø9	11,ØF	11,15	11,Ø2	11,Ø8	11,ØE	11,14	11,1A
33	11,Ø6	11,ØC	11,12	11,18	11,Ø4	11,ØA	11,1Ø	11,16
34	12,Ø1	12,Ø7	12,ØD	12,13	12,19	12,Ø5	12,ØB	12,11
35	12,17	12,Ø3	12,Ø9	12,ØF	12,15	12,Ø2	12,Ø8	12,ØE
36	12,14	12,1A	12,Ø6	12,ØC	12,12	12,18	12,Ø4	12,ØA
37	12,1Ø	12,16	13,Ø1	13,Ø7	13,ØD	13,13	13,19	13,Ø5
38	13,ØB	13,11	13,17	13,Ø3	13,Ø9	13,ØF	13,15	13,Ø2
39	13,Ø8	13,ØE	13,14	13,1A	13,Ø6	13,ØC	13,12	13,18
3A	13,Ø4	13,ØA	13,1Ø	13,16	14,Ø1	14,Ø7	14,ØD	14,13
3B	14,19	14,Ø5	14,ØB	14,11	14,17	14,Ø3	14,Ø9	14,ØF
3C	14,15	14,Ø2	14,Ø8	14,ØE	14,14	14,1A	14,Ø6	14,ØC
3D	14,12	14,18	14,Ø4	14,ØA	14,1Ø	14,16	15,Ø1	15,Ø7
3E	15,ØD	15,13	15,19	15,Ø5	15,ØB	15,11	15,17	15,Ø3
3F	15,Ø9	15,ØF	15,15	15,Ø2	15,Ø8	15,ØE	15,14	15,1A
4Ø	15,Ø6	15,ØC	15,12	15,18	15,Ø4	15,ØA	15,1Ø	15,16
41	16,Ø1	16,Ø7	16,ØD	16,13	16,19	16,Ø5	16,ØB	16,11
42	16,17	16,Ø3	16,Ø9	16,ØF	16,15	16,Ø2	16,Ø8	16,ØE
43	16,14	16,1A	16,Ø6	16,ØC	16,12	16,18	16,Ø4	16,ØA
44	16,1Ø	16,16	17,Ø1	17,Ø7	17,ØD	17,13	17,19	17,Ø5
45	17,ØB	17,11	17,17	17,Ø3	17,Ø9	17,ØF	17,15	17,Ø2
46	17,Ø8	17,ØE	17,14	17,1A	17,Ø6	17,ØC	17,12	17,18
47	17,Ø4	17,ØA	17,1Ø	17,16	18,Ø1	18,Ø7	18,ØD	18,13
48	18,19	18,Ø5	18,ØB	18,11	18,17	18,Ø3	18,Ø9	18,ØF
49	18,15	18,Ø2	18,Ø8	18,ØE	18,14	18,1A	18,Ø6	18,ØC
4A	18,12	18,18	18,Ø4	18,ØA	18,1Ø	18,16	19,Ø1	19,Ø7
4B	19,ØD	19,13	19,19	19,Ø5	19,ØB	19,11	19,17	19,Ø3
4C	19,Ø9	19,ØF	19,15	19,Ø2	19,Ø8	19,ØE	19,14	19,1A
4D	19,Ø6	19,ØC	19,12	19,18	19,Ø4	19,ØA	19,1Ø	19,16
4E	1A,Ø1	1A,Ø7	1A,ØD	1A,13	1A,19	1A,Ø5	1A,ØB	1A,11
4F	1A,17	1A,Ø3	1A,Ø9	1A,ØF	1A,15	1A,Ø2	1A,Ø8	1A,ØE

Mapping of Clusters to Tracks/Sectors - Large Disks (cont.)

Cluster	Sectors Within Each Cluster							
	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc
50	1A,14	1A,1A	1A,06	1A,0C	1A,12	1A,18	1A,04	1A,0A
51	1A,10	1A,16	1B,01	1B,07	1B,0D	1B,13	1B,19	1B,05
52	1B,0B	1B,11	1B,17	1B,03	1B,09	1B,0F	1B,15	1B,02
53	1B,08	1B,0E	1B,14	1B,1A	1B,06	1B,0C	1B,12	1B,18
54	1B,04	1B,0A	1B,10	1B,16	1C,01	1C,07	1C,0D	1C,13
55	1C,19	1C,05	1C,0B	1C,11	1C,17	1C,03	1C,09	1C,0F
56	1C,15	1C,02	1C,08	1C,0E	1C,14	1C,1A	1C,06	1C,0C
57	1C,12	1C,18	1C,04	1C,0A	1C,10	1C,16	1D,01	1D,07
58	1D,0D	1D,13	1D,19	1D,05	1D,0B	1D,11	1D,17	1D,03
59	1D,09	1D,0F	1D,15	1D,02	1D,08	1D,0E	1D,14	1D,1A
5A	1D,06	1D,0C	1D,12	1D,18	1D,04	1D,0A	1D,10	1D,16
5B	1E,01	1E,07	1E,0D	1E,13	1E,19	1E,05	1E,0B	1E,11
5C	1E,17	1E,03	1E,09	1E,0F	1E,15	1E,02	1E,08	1E,0E
5D	1E,14	1E,1A	1E,06	1E,0C	1E,12	1E,18	1E,04	1E,0A
5E	1E,10	1E,16	1F,01	1F,07	1F,0D	1F,13	1F,19	1F,05
5F	1F,0B	1F,11	1F,17	1F,03	1F,09	1F,0F	1F,15	1F,02
60	1F,08	1F,0E	1F,14	1F,1A	1F,06	1F,0C	1F,12	1F,18
61	1F,04	1F,0A	1F,10	1F,16	20,01	20,07	20,0D	20,13
62	20,19	20,05	20,0B	20,11	20,17	20,03	20,09	20,0F
63	20,15	20,02	20,08	20,0E	20,14	20,1A	20,06	20,0C
64	20,12	20,18	20,04	20,0A	20,10	20,16	21,01	21,07
65	21,0D	21,13	21,19	21,05	21,0B	21,11	21,17	21,03
66	21,09	21,0F	21,15	21,02	21,08	21,0E	21,14	21,1A
67	21,06	21,0C	21,12	21,18	21,04	21,0A	21,10	21,16
68	22,01	22,07	22,0D	22,13	22,19	22,05	22,0B	22,11
69	22,17	22,03	22,09	22,0F	22,15	22,02	22,08	22,0E
6A	22,14	22,1A	22,06	22,0C	22,12	22,18	22,04	22,0A
6B	22,10	22,16	23,01	23,07	23,0D	23,13	23,19	23,05
6C	23,0B	23,11	23,17	23,03	23,09	23,0F	23,15	23,02
6D	23,08	23,0E	23,14	23,1A	23,06	23,0C	23,12	23,18
6E	23,04	23,0A	23,10	23,16	24,01	24,07	24,0D	24,13
6F	24,19	24,05	24,0B	24,11	24,17	24,03	24,09	24,0F
70	24,15	24,02	24,08	24,0E	24,14	24,1A	24,06	24,0C
71	24,12	24,18	24,04	24,0A	24,10	24,16	25,01	25,07
72	25,0D	25,13	25,19	25,05	25,0B	25,11	25,17	25,03
73	25,09	25,0F	25,15	25,02	25,08	25,0E	25,14	25,1A
74	25,06	25,0C	25,12	25,18	25,04	25,0A	25,10	25,16
75	26,01	26,07	26,0D	26,13	26,19	26,05	26,0B	26,11
76	26,17	26,03	26,09	26,0F	26,15	26,02	26,08	26,0E
77	26,14	26,1A	26,06	26,0C	26,12	26,18	26,04	26,0A

Mapping of Clusters to Tracks/Sectors - Large Disks (cont.)

Clusters	Sectors Within Each Cluster Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc
78	26,10	26,16	27,01	27,07	27,0D	27,13	27,19	27,05
79	27,0B	27,11	27,17	27,03	27,09	27,0F	27,15	27,02
7A	27,08	27,0E	27,14	27,1A	27,06	27,0C	27,12	27,18
7B	27,04	27,0A	27,10	27,16	28,01	28,07	28,0D	28,13
7C	28,19	28,05	28,0B	28,11	28,17	28,03	28,09	28,0F
7D	28,15	28,02	28,08	28,0E	28,14	28,1A	28,06	28,0C
7E	28,12	28,18	28,04	28,0A	28,10	28,16	29,01	29,07
7F	29,0D	29,13	29,19	29,05	29,0B	29,11	29,17	29,03
80	29,09	29,0F	29,15	29,02	29,08	29,0E	29,14	29,1A
81	29,06	29,0C	29,12	29,18	29,04	29,0A	29,10	29,16
82	2A,01	2A,07	2A,0D	2A,13	2A,19	2A,05	2A,0B	2A,11
83	2A,17	2A,03	2A,09	2A,0F	2A,15	2A,02	2A,08	2A,0E
84	2A,14	2A,1A	2A,06	2A,0C	2A,12	2A,18	2A,04	2A,0A
85	2A,10	2A,16	2B,01	2B,07	2B,0D	2B,13	2B,19	2B,05
86	2B,0B	2B,11	2B,17	2B,03	2B,09	2B,0F	2B,15	2B,02
87	2B,08	2B,0E	2B,14	2B,1A	2B,06	2B,0C	2B,12	2B,18
88	2B,04	2B,0A	2B,10	2B,16	2C,01	2C,07	2C,0D	2C,13
89	2C,19	2C,05	2C,0B	2C,11	2C,17	2C,03	2C,09	2C,0F
8A	2C,15	2C,02	2C,08	2C,0E	2C,14	2C,1A	2C,06	2C,0C
8B	2C,12	2C,18	2C,04	2C,0A	2C,10	2C,16	2D,01	2D,07
8C	2D,0D	2D,13	2D,19	2D,05	2D,0B	2D,11	2D,17	2D,03
8D	2D,09	2D,0F	2D,15	2D,02	2D,08	2D,0E	2D,14	2D,1A
8E	2D,06	2D,0C	2D,12	2D,18	2D,04	2D,0A	2D,10	2D,16
8F	2E,01	2E,07	2E,0D	2E,13	2E,19	2E,05	2E,0B	2E,11
90	2E,17	2E,03	2E,09	2E,0F	2E,15	2E,02	2E,08	2E,0E
91	2E,14	2E,1A	2E,06	2E,0C	2E,12	2E,18	2E,04	2E,0A
92	2E,10	2E,16	2F,01	2F,07	2F,0D	2F,13	2F,19	2F,05
93	2F,0B	2F,11	2F,17	2F,03	2F,09	2F,0F	2F,15	2F,02
94	2F,08	2F,0E	2F,14	2F,1A	2F,06	2F,0C	2F,12	2F,18
95	2F,04	2F,0A	2F,10	2F,16	30,01	30,07	30,0D	30,13
96	30,19	30,05	30,0B	30,11	30,17	30,03	30,09	30,0F
97	30,15	30,02	30,08	30,0E	30,14	30,1A	30,06	30,0C
98	30,12	30,18	30,04	30,0A	30,10	30,16	31,01	31,07
99	31,0D	31,13	31,19	31,05	31,0B	31,11	31,17	31,03
9A	31,09	31,0F	31,15	31,02	31,08	31,0E	31,14	31,1A
9B	31,06	31,0C	31,12	31,18	31,04	31,0A	31,10	31,16
9C	32,01	32,07	32,0D	32,13	32,19	32,05	32,0B	32,11
9D	32,17	32,03	32,09	32,0F	32,15	32,02	32,08	32,0E
9E	32,14	32,1A	32,06	32,0C	32,12	32,18	32,04	32,0A
9F	32,10	32,16	33,01	33,07	33,0D	33,13	33,19	33,05

Mapping of Clusters to Tracks/Sectors - Large Disks (cont.)

Clusters	Sectors Within Each Cluster							
	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc
A0	33,0B	33,11	33,17	33,03	33,09	33,0F	33,15	33,02
A1	33,08	33,0E	33,14	33,1A	33,06	33,0C	33,12	33,18
A2	33,04	33,0A	33,10	33,16	34,01	34,07	34,0D	34,13
A3	34,19	34,05	34,0B	34,11	34,17	34,03	34,09	34,0F
A4	34,15	34,02	34,08	34,0E	34,14	34,1A	34,06	34,0C
A5	34,12	34,18	34,04	34,0A	34,10	34,16	35,01	35,07
A6	35,0D	35,13	35,19	35,05	35,0B	35,11	35,17	35,03
A7	35,09	35,0F	35,15	35,02	35,08	35,0E	35,14	35,1A
A8	35,06	35,0C	35,12	35,18	35,04	35,0A	35,10	35,16
A9	36,01	36,07	36,0D	36,13	36,19	36,05	36,0B	36,11
AA	36,17	36,03	36,09	36,0F	36,15	36,02	36,08	36,0E
AB	36,14	36,1A	36,06	36,0C	36,12	36,18	36,04	36,0A
AC	36,10	36,16	37,01	37,07	37,0D	37,13	37,19	37,05
AD	37,0B	37,11	37,17	37,03	37,09	37,0F	37,15	37,02
AE	37,08	37,0E	37,14	37,1A	37,06	37,0C	37,12	37,18
AF	37,04	37,0A	37,10	37,16	38,01	38,07	38,0D	38,13
B0	38,19	38,05	38,0B	38,11	38,17	38,03	38,09	38,0F
B1	38,15	38,02	38,08	38,0E	38,14	38,1A	38,06	38,0C
B2	38,12	38,18	38,04	38,0A	38,10	38,16	39,01	39,07
B3	39,0D	39,13	39,19	39,05	39,0B	39,11	39,17	39,03
B4	39,09	39,0F	39,15	39,02	39,08	39,0E	39,14	39,1A
B5	39,06	39,0C	39,12	39,18	39,04	39,0A	39,10	39,16
B6	3A,01	3A,07	3A,0D	3A,13	3A,19	3A,05	3A,0B	3A,11
B7	3A,17	3A,03	3A,09	3A,0F	3A,15	3A,02	3A,08	3A,0E
B8	3A,14	3A,1A	3A,06	3A,0C	3A,12	3A,18	3A,04	3A,0A
B9	3A,10	3A,16	3B,01	3B,07	3B,0D	3B,13	3B,19	3B,05
BA	3B,0B	3B,11	3B,17	3B,03	3B,09	3B,0F	3B,15	3B,02
BB	3B,08	3B,0E	3B,14	3B,1A	3B,06	3B,0C	3B,12	3B,18
BC	3B,04	3B,0A	3B,10	3B,16	3C,01	3C,07	3C,0D	3C,13
BD	3C,19	3C,05	3C,0B	3C,11	3C,17	3C,03	3C,09	3C,0F
BE	3C,15	3C,02	3C,08	3C,0E	3C,14	3C,1A	3C,06	3C,0C
BF	3C,12	3C,18	3C,04	3C,0A	3C,10	3C,16	3D,01	3D,07
C0	3D,0D	3D,13	3D,19	3D,05	3D,0B	3D,11	3D,17	3D,03
C1	3D,09	3D,0F	3D,15	3D,02	3D,08	3D,0E	3D,14	3D,1A
C2	3D,06	3D,0C	3D,12	3D,18	3D,04	3D,0A	3D,10	3D,16
C3	3E,01	3E,07	3E,0D	3E,13	3E,19	3E,05	3E,0B	3E,11
C4	3E,17	3E,03	3E,09	3E,0F	3E,15	3E,02	3E,08	3E,0E
C5	3E,14	3E,1A	3E,06	3E,0C	3E,12	3E,18	3E,04	3E,0A
C6	3E,10	3E,16	3F,01	3F,07	3F,0D	3F,13	3F,19	3F,05
C7	3F,0B	3F,11	3F,17	3F,03	3F,09	3F,0F	3F,15	3F,02

Mapping of Clusters to Tracks/Sectors - Large Disks (cont.)

Clusters	Sectors Within Each Cluster							
	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc
C8	3F,Ø8	3F,ØE	3F,14	3F,1A	3F,Ø6	3F,ØC	3F,12	3F,18
C9	3F,Ø4	3F,ØA	3F,1Ø	3F,16	4Ø,Ø1	4Ø,Ø7	4Ø,ØD	4Ø,13
CA	4Ø,19	4Ø,Ø5	4Ø,ØB	4Ø,11	4Ø,17	4Ø,Ø3	4Ø,Ø9	4Ø,ØF
CB	4Ø,15	4Ø,Ø2	4Ø,Ø8	4Ø,ØE	4Ø,14	4Ø,1A	4Ø,Ø6	4Ø,ØC
CC	4Ø,12	4Ø,18	4Ø,Ø4	4Ø,ØA	4Ø,1Ø	4Ø,16	41,Ø1	41,Ø7
CD	41,ØD	41,13	41,19	41,Ø5	41,ØB	41,11	41,17	41,Ø3
CE	41,Ø9	41,ØF	41,15	41,Ø2	41,Ø8	41,ØE	41,14	41,1A
CF	41,Ø6	41,ØC	41,12	41,18	41,Ø4	41,ØA	41,1Ø	41,16
DØ	42,Ø1	42,Ø7	42,ØD	42,13	42,19	42,Ø5	42,ØB	42,11
D1	42,17	42,Ø3	42,Ø9	42,ØF	42,15	42,Ø2	42,Ø8	42,ØE
D2	42,14	42,1A	42,Ø6	42,ØC	42,12	42,18	42,Ø4	42,ØA
D3	42,1Ø	42,16	43,Ø1	43,Ø7	43,ØD	43,13	43,19	43,Ø5
D4	43,ØB	43,11	43,17	43,Ø3	43,Ø9	43,ØF	43,15	43,Ø2
D5	43,Ø8	43,ØE	43,14	43,1A	43,Ø6	43,ØC	43,12	43,18
D6	43,Ø4	43,ØA	43,1Ø	43,16	44,Ø1	44,Ø7	44,ØD	44,13
D7	44,19	44,Ø5	44,ØB	44,11	44,17	44,Ø3	44,Ø9	44,ØF
D8	44,15	44,Ø2	44,Ø8	44,ØE	44,14	44,1A	44,Ø6	44,ØC
D9	44,12	44,18	44,Ø4	44,ØA	44,1Ø	44,16	45,Ø1	45,Ø7
DA	45,ØD	45,13	45,19	45,Ø5	45,ØB	45,11	45,17	45,Ø3
DB	45,Ø9	45,ØF	45,15	45,Ø2	45,Ø8	45,ØE	45,14	45,1A
DC	45,Ø6	45,ØC	45,12	45,18	45,Ø4	45,ØA	45,1Ø	45,16
DD	46,Ø1	46,Ø7	46,ØD	46,13	46,19	46,Ø5	46,ØB	46,11
DE	46,17	46,Ø3	46,Ø9	46,ØF	46,15	46,Ø2	46,Ø8	46,ØE
DF	46,14	46,1A	46,Ø6	46,ØC	46,12	46,18	46,Ø4	46,ØA
EØ	46,1Ø	46,16	47,Ø1	47,Ø7	47,ØD	47,13	47,19	47,Ø5
E1	47,ØB	47,11	47,17	47,Ø3	47,Ø9	47,ØF	47,15	47,Ø2
E2	47,Ø8	47,ØE	47,14	47,1A	47,Ø6	47,ØC	47,12	47,18
E3	47,Ø4	47,ØA	47,1Ø	47,16	48,Ø1	48,Ø7	48,ØD	48,13
E4	48,19	48,Ø5	48,ØB	48,11	48,17	48,Ø3	48,Ø9	48,ØF
E5	48,15	48,Ø2	48,Ø8	48,ØE	48,14	48,1A	48,Ø6	48,ØC
E6	48,12	48,18	48,Ø4	48,ØA	48,1Ø	48,16	49,Ø1	49,Ø7
E7	49,ØD	49,13	49,19	49,Ø5	49,ØB	49,11	49,17	49,Ø3
E8	49,Ø9	49,ØF	49,15	49,Ø2	49,Ø8	49,ØE	49,14	49,1A
E9	49,Ø6	49,ØC	49,12	49,18	49,Ø4	49,ØA	49,1Ø	49,16
EA	4A,Ø1	4A,Ø7	4A,ØD	4A,13	4A,19	4A,Ø5	4A,ØB	4A,11
EB	4A,17	4A,Ø3	4A,Ø9	4A,ØF	4A,15	4A,Ø2	4A,Ø8	4A,ØE
EC	4A,14	4A,1A	4A,Ø6	4A,ØC	4A,12	4A,18	4A,Ø4	4A,ØA
ED	4A,1Ø	4A,16	4B,Ø1	4B,Ø7	4B,ØD	4B,13	4B,19	4B,Ø5
EE	4B,ØB	4B,11	4B,17	4B,Ø3	4B,Ø9	4B,ØF	4B,15	4B,Ø2
EF	4B,Ø8	4B,ØE	4B,14	4B,1A	4B,Ø6	4B,ØC	4B,12	4B,18
FØ	4B,Ø4	4B,ØA	4B,1Ø	4B,16	4C,Ø1	4C,Ø7	4C,ØD	4C,13
F1	4C,19	4C,Ø5	4C,ØB	4C,11	4C,17	4C,Ø3	4C,Ø9	4C,ØF
F2	4C,15	4C,Ø2	4C,Ø8	4C,ØE	4C,14	4C,1A	4C,Ø6	4C,ØC

MAPPING OF CLUSTERS TO TRACKS/SECTORS - SMALL DISKS

Clusters	Sectors Within Each Cluster Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc
00	03,01	03,06	03,0B	03,10	03,03	03,08	03,0D	03,12
01	03,05	03,0A	03,0F	03,02	03,07	03,0C	03,11	03,04
02	03,09	03,0E	04,01	04,06	04,0B	04,10	04,03	04,08
03	04,0D	04,12	04,05	04,0A	04,0F	04,02	04,07	04,0C
04	04,11	04,04	04,09	04,0E	05,01	05,06	05,0B	05,10
05	05,03	05,08	05,0D	05,12	05,05	05,0A	05,0F	05,02
06	05,07	05,0C	05,11	05,04	05,09	05,0E	06,01	06,06
07	06,0B	06,10	06,03	06,08	06,0D	06,12	06,05	06,0A
08	06,0F	06,02	06,07	06,0C	06,11	06,04	06,09	06,0E
09	07,01	07,06	07,0B	07,10	07,03	07,08	07,0D	07,12
0A	07,05	07,0A	07,0F	07,02	07,07	07,0C	07,11	07,04
0B	07,09	07,0E	08,01	08,06	08,0B	08,10	08,03	08,08
0C	08,0D	08,12	08,05	08,0A	08,0F	08,02	08,07	08,0C
0D	08,11	08,04	08,09	08,0E	09,01	09,06	09,0B	09,10
0E	09,03	09,08	09,0D	09,12	09,05	09,0A	09,0F	09,02
0F	09,07	09,0C	09,11	09,04	09,09	09,0E	0A,01	0A,06
10	0A,0B	0A,10	0A,03	0A,08	0A,0D	0A,12	0A,05	0A,0A
11	0A,0F	0A,02	0A,07	0A,0C	0A,11	0A,04	0A,09	0A,0E
12	0B,01	0B,06	0B,0B	0B,10	0B,03	0B,08	0B,0D	0B,12
13	0B,05	0B,0A	0B,0F	0B,02	0B,07	0B,0C	0B,11	0B,04
14	0B,09	0B,0E	0C,01	0C,06	0C,0B	0C,10	0C,03	0C,08
15	0C,0D	0C,12	0C,05	0C,0A	0C,0F	0C,02	0C,07	0C,0C
16	0C,11	0C,04	0C,09	0C,0E	0D,01	0D,06	0D,0B	0D,10
17	0D,03	0D,08	0D,0D	0D,12	0D,05	0D,0A	0D,0F	0D,02
18	0D,07	0D,0C	0D,11	0D,04	0D,09	0D,0E	0E,01	0E,06
19	0E,0B	0E,10	0E,03	0E,08	0E,0D	0E,12	0E,05	0E,0A
1A	0E,0F	0E,02	0E,07	0E,0C	0E,11	0E,04	0E,09	0E,0E
1B	0F,01	0F,06	0F,0B	0F,10	0F,03	0F,08	0F,0D	0F,12
1C	0F,05	0F,0A	0F,0F	0F,02	0F,07	0F,0C	0F,11	0F,04
1D	0F,09	0F,0E	10,01	10,06	10,0B	10,10	10,03	10,08
1E	10,0D	10,12	10,05	10,0A	10,0F	10,02	10,07	10,0C
1F	10,11	10,04	10,09	10,0E	11,01	11,06	11,0B	11,10
20	11,03	11,08	11,0D	11,12	11,05	11,0A	11,0F	11,02
21	11,07	11,0C	11,11	11,04	11,09	11,0E	12,01	12,06
22	12,0B	12,10	12,03	12,08	12,0D	12,12	12,05	12,0A
23	12,0F	12,02	12,07	12,0C	12,11	12,04	12,09	12,0E
24	13,01	13,06	13,0B	13,10	13,03	13,08	13,0D	13,12
25	13,05	13,0A	13,0F	13,02	13,07	13,0C	13,11	13,04
26	13,09	13,0E	14,01	14,06	14,0B	14,10	14,03	14,08
27	14,0D	14,12	14,05	14,0A	14,0F	14,02	14,07	14,0C

Mapping of Clusters to Tracks/Sectors - Small Disks (cont.)

Cluster	Sectors Within Each Cluster							
	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc	Tk,Sc
28	14,11	14,04	14,09	14,0E	15,01	15,06	15,0B	15,10
29	15,03	15,08	15,0D	15,12	15,05	15,0A	15,0F	15,02
2A	15,07	15,0C	15,11	15,04	15,09	15,0E	16,01	16,06
2B	16,0B	16,10	16,03	16,08	16,0D	16,12	16,05	16,0A
2C	16,0F	16,02	16,07	16,0C	16,11	16,04	16,09	16,0E
2D	17,01	17,06	17,0B	17,10	17,03	17,08	17,0D	17,12
2E	17,05	17,0A	17,0F	17,02	17,07	17,0C	17,11	17,04
2F	17,09	17,0E	18,01	18,06	18,0B	18,10	18,03	18,08
30	18,0D	18,12	18,05	18,0A	18,0F	18,02	18,07	18,0C
31	18,11	18,04	18,09	18,0E	19,01	19,06	19,0B	19,10
32	19,03	19,08	19,0D	19,12	19,05	19,0A	19,0F	19,02
33	19,07	19,0C	19,11	19,04	19,09	19,0E	1A,01	1A,06
34	1A,0B	1A,10	1A,03	1A,08	1A,0D	1A,12	1A,05	1A,0A
35	1A,0F	1A,02	1A,07	1A,0C	1A,11	1A,04	1A,09	1A,0E
36	1B,01	1B,06	1B,0B	1B,10	1B,03	1B,08	1B,0D	1B,12
37	1B,05	1B,0A	1B,0F	1B,02	1B,07	1B,0C	1B,11	1B,04
38	1B,09	1B,0E	1C,01	1C,06	1C,0B	1C,10	1C,03	1C,08
39	1C,0D	1C,12	1C,05	1C,0A	1C,0F	1C,02	1C,07	1C,0C
3A	1C,11	1C,04	1C,09	1C,0E	1D,01	1D,06	1D,0B	1D,10
3B	1D,03	1D,08	1D,0D	1D,12	1D,05	1D,0A	1D,0F	1D,02
3C	1D,07	1D,0C	1D,11	1D,04	1D,09	1D,0E	1E,01	1E,06
3D	1E,0B	1E,10	1E,03	1E,08	1E,0D	1E,12	1E,05	1E,0A
3E	1E,0F	1E,02	1E,07	1E,0C	1E,11	1E,04	1E,09	1E,0E
3F	1F,01	1F,06	1F,0B	1F,10	1F,03	1F,08	1F,0D	1F,12
40	1F,05	1F,0A	1F,0F	1F,02	1F,07	1F,0C	1F,11	1F,04
41	1F,09	1F,0E	20,01	20,06	20,0B	20,10	20,03	20,08
42	20,0D	20,12	20,05	20,0A	20,0F	20,02	20,07	20,0C
43	20,11	20,04	20,09	20,0E	21,01	21,06	21,0B	21,10
44	21,03	21,08	21,0D	21,12	21,05	21,0A	21,0F	21,02
45	21,07	21,0C	21,11	21,04	21,09	21,0E	22,01	22,06
46	22,0B	22,10	22,03	22,08	22,0D	22,12	22,05	22,0A
47	22,0F	22,02	22,07	22,0C	22,11	22,04	22,09	22,0E
48	23,01	23,06	23,0B	23,10	23,03	23,08	23,0D	23,12
49	23,05	23,0A	23,0F	23,02	23,07	23,0C	23,11	23,04
4A	23,09	23,0E	24,01	24,06	24,0B	24,10	24,03	24,08
4B	24,0D	24,12	24,05	24,0A	24,0F	24,02	24,07	24,0C
4C	24,11	24,04	24,09	24,0E	25,01	25,06	25,0B	25,10
4D	25,03	25,08	25,0D	25,12	25,05	25,0A	25,0F	25,02
4E	25,07	25,0C	25,11	25,04	25,09	25,0E	26,01	26,06
4F	26,0B	26,10	26,03	26,08	26,0D	26,12	26,05	26,0A
50	26,0F	26,02	26,07	26,0C	26,11	26,04	26,09	26,0E
51	27,01	27,06	27,0B	27,10	27,03	27,08	27,0D	27,12
52	27,05	27,0A	27,0F	27,02	27,07	27,0C	27,11	27,04

Error Messages

In the event of a system malfunction, CDOS displays a complete error message to aid in the diagnosis and correction of the problem. The following section describes these messages and their interpretation.

6.1 Disk Access Error Messages

When the operating system cannot successfully access a disk an error message is displayed.

Format:

WRD Error, Drive X, Track YY, Sector ZZ, Status=HH

where:

WRD	stands for one of the following words:
Seek	Error occurred in seeking a track on the disk.
Read	Error occurred during a read from the disk.
Write	Error occurred during a write to the disk.
Init	Error occurred during disk initialization.
X	is A, B, C, or D which represents the disk drive with the error.
YY	is the track number (in hexadecimal) where the error occurred.

ZZ — is the sector number (in hexadecimal) where the error occurred.

HH — is the 8 bit status byte displayed in hexadecimal which describes the error and the conditions at the time the error occurred.

The following table may be consulted for a detailed analysis of the Status portion of an error message.

Status Bits set (=1)	Seek	Read
7	not ready	not ready
6	write protect*	record type*
5	head engaged*	record type*
4	seek error	record not found
3	crc error	crc error
2	track Ø*	lost data
1	index*	data request*
Ø	busy*	busy*

Status Bits set (=1)	Write	Init
7	not ready	not ready
6	write protect	write protect
5	write fault	write fault
4	record not found	not used*
3	crc error	not valid
2	lost data	lost data
1	data request*	data request*
Ø	busy*	busy*

The asterisk (*) in the above table indicates that the condition is not the cause of the error message, but that it was present when the error occurred. For example, if the status byte was 3ØH during a Seek error, this means that bits 4 and 5 are set (=1). There is a Seek error and the head is engaged. The head is supposed to be engaged during a seek and therefore this condition is not an error and is marked with an asterisk. CRC

stands for Cyclic Redundancy Check. This indicates that data was not transferred without error.

There are four possible responses to the error message:

R	This will cause the system to retry the disk access which caused the error. Note: The error message does not appear until after the disk access instruction has been repeated ten times.
I	This will cause the system to Ignore the error message and continue. The function which caused the error message is not completed and no error code is returned to the calling program.
C	This will cause the system to Continue. The function which caused the error message is not completed and an error code is returned to the calling program.
CTRL-C	This will abort the program and return control to the CDOS monitor.

Examples:

The following examples use some of the more common status codes:

Seek Error, Drive A, Track 17, Sector 1A, Status=36

During a Seek operation, status code 36 or B6 indicates that the system expected to find a mini disk drive when there was actually a maxi drive (or vice versa) at the location (specified by A above). CDOSGEN may be run to correct this problem. Be sure that the disk drives are correctly specified as small and large during the system generation.

Read Error, Drive B, Track 1C, Sector 1Ø, Status=1Ø

During a Read operation, status code 1Ø or Ø8 indicate that the data is not readable. This may be caused by bringing the disk close to a magnetic

source or by scratching or otherwise mishandling the disk.

Write Error, Drive C, Track 3C, Sector 14, Status=40

During a Write operation, status code 40 indicates that an attempt is being made to write on a write protected disk. The write protect sticker must be in place on a large disk and removed from a small disk in order to be able to write on the disk

6.2 System Error Messages

If an instruction is executed which causes control to be transferred to a nonexistent memory location or any memory location containing 0FFH (Restart 38), the following message will be displayed:

Invalid jump to location xxxx

where:

xxxx	is the hexadecimal address to which control was transferred.

Glossary of Terms and Symbols

{ }

Braces are used to indicate a choice of items. One of the items enclosed in the braces must be used in the position indicated. An optional choice of items is indicated by braces enclosed in square brackets.

[ ]

Square brackets are used to indicate an optional quantity. The item enclosed in square brackets may be used, in the position indicated, at the user's discretion.

Ambiguous File Reference

This is a File Reference which may refer to more than one file by using a Replacement Character(s). See section 1.3.2.2.

Disk Specifier

A Disk Specifier is one of the letters A, B, C, or D followed by a colon. This letter references a disk drive and allows the user to refer to a disk located in the drive. The Disk Specifier is an optional part of a File Reference.

File or
Data File

A File defines a group of related information. This information is addressed by means of a File Reference and usually resides on a floppy diskette.

File Area (disk)

User Files are stored on this part of the disk. The contents of this part of the disk are listed by the DIRectory command. See Section 1.2.1.

File Name

This is a one to eight character label which is used to refer to a File. Several Files may have the same File Name. These Files may be uniquely identified by the use of a Disk Specifier and/or a File Name Extension. A File Name is a necessary part of a File Reference. See section 1.3.2.1.

File Name Extension

This is a one to three character label which is frequently used to indicate how a file is to be used. A File Name Extension is an optional part of a File Reference.

File-ref or
File Reference

A File Reference identifies and locates a File.

Format: [x:]filename[.ext]

where:

x	is an optional Disk Drive Specifier.
filename	is a File Name.

ext is an optional File Name Extension.

A File Reference is a Single File Reference unless it is specifically stated that it may incorporate Replacement Characters to form an Ambiguous File Reference. See sections 1.3.2.1 and 1.3.2.2.

Replacement Character

A Replacement Character is an asterisk (*) or a question mark (?). These characters may be used where specifically indicated in order to create an Ambiguous File Reference. See section 1.3.2.2.

Single File Reference

This is a label specifying a unique File. This File Reference may not include Replacement Characters. See section 1.3.2.1.

System Area (disk)

CDOS is stored on this part of the disk. This section is normally accessed only by CDOS. It does not appear in the User Area DIRectory. See Section 1.2.1.

User Area (memory)

The User Area is memory which is available to user programs. This is the part of memory from 100H up to the bottom of CDOS. Refer to section 1.1.

I N D E X