Supermax Operating System

Introductory Guide

Dansk Data Elektronik A/S
1 May 1984

1. Introduction

This manual describes the basic features of the Supermax Operating System. The manual is intended for the first time users of the Supermax computer.

2. What is an Operating System?

An operating system is the program that controls the execution of all other programs on a computer.

The user does not see the operating system directly. The user merely communicates with one or more running programs such as a word processing system, a BASIC interpreter, a bookkeeping package, etc.; but behind it all the omnipresent operating system is controlling and checking the user's programs.

Thus the user is normally not aware of the presence of the operating system, but, nonetheless, the knowlegde of a few basics is required in order to use the system properly. Also, a general knowlegde of what is going on inside the computer will help the user to make better use of the system.

One word you may hear frequently when people talk about the Supermax Operating System is the name 'Unix'. Unix is the name of a widely used operating system developed by Bell Laboratories in the USA. Unix is the basis of the Supermax Operating System, in the sense that practically all features found in the Unix operating system are found in the Supermax Operating System as well.

3. Getting Started.

In the following section reference will often be made to the System Administrator. This is the person (or persons) in your company who is responsible for the Supermax computer.

Before you can do anything on the computer, you need to be allowed access to the computer. Contact your System Administrator. He or she will assign you

1) a 'user name',
2) a secret 'password',
3) a 'user ID',
4) a 'group name',
5) a 'group ID',
6) an 'operator service program', and
7) a 'home directory'.

The user name may simply be your first or last name. The password may be any combination of letters and digits you desire. The user ID, group ID, and home directory will be discussed in detail later. The group name may be the name of your department in your company, or it may be an identification of a project you are going to work on. The operator service program is the program that you will be running on the computer once you get in contact with it; more about this later.

3.1. Logging On.

Let us assume that you have access to a running Supermax computer. Connected to the computer are a number of terminals. Find a vacant terminal. When nobody uses the terminal, the screen will normally show some introductory picture like the following:

```
**************************************************************************
*    Welcome to the Supermax Operating system version 10 May 1984        *
*                                                                        *
*                                                  Terminal number 4     *
**************************************************************************
```

This picture is called the logon picture. The actual layout of this picture may vary. It may be redesigned by the System Administrator.

Have a look at the keyboard. The keyboard consists of the keys found

on normal typewriters, plus a set of so-called function keys. The actual layout of the keyboard will depend on what kind of terminal you are using, so if the following description confuses you, talk with your System Administrator.

One key is the 'attention key'. It will often be labelled 'ESCAPE' or 'ESC'. This key is used to 'kick' the system, that is, change the flow of events inside the computer. If you press this key, the text 'Enter user name:' will appear on the screen. Type your user name and press the RETURN key. Now the text 'Enter password:' will appear, and you must type your secret password followed by the RETURN key. Note that the password will not be displayed on the screen as you type it. Because of this, somebody watching over your shoulder will have a hard time discovering you password.

If you made a mistake when typing your user name or password, the text 'Rejected' will be displayed, and you will have to try to enter your user name again.

If you have typed everything correctly, the text 'You are now logging on to processor number: 1' will appear. The number may not be '1'. This will be discussed later. For the present, simply press the RETURN key.

After a few seconds a program will start running. This is your 'operator service program' assigned to you by the System Administrator. It may be some specialized program package, such as a word processing program or a bookkeeping program, or it may be a general operator communication program, such as 'vox' or 'shell'. What you must do hereafter depends on what operator service program the computer is running for you.

An introduction to 'vox' is given in Appendix A of this manual. An introduction to 'shell' is given in Appendix B.

What you have just done is termed 'logging on'. When you stop the execution of the operator service program, the system will return to the initial logon picture. This is termed 'logging off'.

3.2. Typing to the Terminal.

You have already discovered a few features about the terminal you are working with. First, you will have noticed the block or underscore that moves in front of you text as you type, always indicating the place in which the next character will go. This indicator is called the''cursor'. Second, you will have noticed that, as you entered your user name, the keys that you pressed were shown on the screen, whereas this was not the case when you entered your password. Normally the terminal will display what you type; we say that the computer 'echoes' your keystrokes.

Terminals differ from normal typewriters in one very important respect: On a typewriter the letter l (small L) is often used for the digit 1 (one) and the letter O (capital o) is often used for the digit 0 (zero). This is not the case on a terminal. The terminal is equipped with keys bearing the digits zero and one, and these keys must be used when you want to type the digits, as most programs will not accept the letters l and O. Some terminals and printers display the digit zero with a slash through it (thus: Ø) in order to distinguish between the digit and the letter. (This, of course, does not help Norwegians and Danes, for whom Ø is a letter in its own right.)

As you type a line to the terminal, you may make mistakes. In most cases the line you are writing will not be sent to the computer program before you press the RETURN key. This means that as long as you do not hit the RETURN key you are allowed to change the contents of the line. A number of keys on the terminal aid you in editing the line, these keys will be described in the following paragraphs. You need not worry about committing everything below to memory, merely read the following, and return to it as you become aquainted with the computer.

The left arrow and right arrow keys move the cursor left or right. You may use these keys to position the cursor on an erroneous letter and then hit the correct key. The new letter will replace the old one.

The start-of-line and end-of-line keys (on some terminals labelled with a double left and double right arrow, respectively) will move the cursor to the beginning of the line and the end of the line, respectively. The end of the line is the position following the last visible character on the line. On many terminals you have to hold down the SHIFT key in order to activate the end-of-line function.

The line you are writing has a maximum length. If you try to type more characters than the length of the line, you will hear a beep, and the cursor will not move beyond the end of the line.

The insert character key makes space for a new character at the cursor position. The rightmost character will be lost if the line is full. Suppose, for example, that your line looks like this (the underscore represents the cursor):

Friends, Romans, countrymn, lend me your ears.

Pressing the insert character key will change the line to:

Friends, Romans, countrym_n, lend me your ears.

Now you may enter the missing letter.

The delete character key deletes the character at the cursor position. Suppose, for example, that your line looks like this (the underscore represents the cursor):

Friends, Romans, countrymean, lend me your ears.

Pressing the delete character key will change the line to:

Friends, Romans, countrymen, lend me your ears.

The erase to end-of-line key deletes the characters at and to the right of the cursor. Suppose, for example, that your line looks like this (the underscore represents the cursor):

Friends, Romans, countrymean, lend me your ears.

Pressing the erase to end-of-line key will change the line to:

Friends, Romans, countryme_

The erase line key erases the whole line and moves the cursor to the start of the line.

The tab key moves the cursor to the next tab stop. Tab stops are located at every eighth position on the line.

The underscore key (_) underlines the character at the cursor position. Some terminals are not able to display underlined characters. On such terminal an underlined character will be shown black on white or with a different light intensity. If you type a new character on top of an underlined character, the underline will disappear.

The rub key (labelled rubout, del, or delete on some terminals) removes the underline from an underlined character.

Most other keys on the keyboard will terminate the line editing as the RETURN key does it.

Special attention should be paid to the shift key, the caps lock key, and the ctrl key:

The shift key works like the corresponding key on normal typewriters. When the shift key is held down, pressing the A key yields an upper case A instead of a lower case a.

The caps lock key works almost like the shift lock key on typewriters. However, the caps lock key only locks the letters in upper case; you still have to press the shift key to get the upper case variants of the other characters.

The ctrl (that is, 'control') key is another kind of shift key. Like the shift key it changes the meaning of any character pressed while the key is held down. However, keys pressed while the ctrl key is down will normally not cause anything to be displayed on the terminal screen. Normally only three keys will be used with the ctrl key, namely the D, S, Q, R, and T keys:

If the computer is writing a lot of text to the terminal, you may temporarily stop the terminal by holding down the ctrl key and pressing the S key. When you want the terminal to continue, hold down the ctrl key and press the Q key.

Occationally, you may hit the ctrl and S keys by mistake, thus suspending output. If nothing is written to your terminal, try pressing ctrl and Q, it may be that the output has been accidentally suspended.

A few programs may request you to enter a so-called 'end-of-file' in response to a question. An end-of-file is typed by holding down the ctrl key and pressing the D key.

Some programs enable you to press the so-called 'interrupt' or 'quit' keys on the terminal. These are much like attention. The interrupt key is normally activated by holding down the ctrl key and pressing the R key. The quit key is an analogous combination of the ctrl key and the T key.

Finally, note that the cursor need not be at the end of the line when you press the RETURN key.

3.3. The Edit Operation.

In many cases when you are requested to type something to the terminal, you will be typing onto a blank line on the screen. However, occationally the line will not be blank when you are requested to type. Instead the computer will have placed some characters on the line for you. These characters will often be what the computer assumes you are most likely to type. In this case, what you have to do is change the contents of the line, if required, and press the RETURN key. The operation performed by the computer in this case is termed an Edit Operation.

Example: The computer displays the following line on the screen (the underscore represents the cursor):

Do you want to continue? YES

In this case, moving the cursor left and right may reveal to you that you are allowed to type different letters on top of the YES. If you are content with the YES answer, merely press the RETURN key. If you want to answer NO to the question, type NO (followed by the space bar to remove the S from YES) and press the RETURN key.

4. Iounits.

Note: Not all users of the Supermax computer need know about iounits. It all depends on the kind of programs you will be running.

All input and output (commonly known as i/o) from the computer takes place between the computer and some device, such as a terminal, a printer, or magnetic disk storage.

For all these devices the Supermax Operating System uses the generic term 'iounits' (pronounced 'eye-oh-units'). This chapter will briefly describe the characteristics of each kind of iounit.

Each iounit connected to the computer has a name. Iounit names will be discussed in greater detail later; examples of iounit names are

```
/dev/print3
/usr/cxp/my_file
Bilbo
master/Frodo
../Here_and_There
```

4.1. Terminals.

You are already acquainted with terminals. Terminals are used for direct communication with an operator. Terminals perform line-by-line or character-by-character i/o.

Normally, the iounit names of terminals are:

```
/dev/tty00
/dev/tty01
/dev/tty02
:
:
etc.
```

The iounit names contain a number, the so-called 'terminal number'. Occationally, teminals may have other names as well. For example, you may prefer to refer to your own terminal as '/dev/Johns_term' rather than '/dev/tty12'. This can be arranged by your System Administrator.

The System Administrator may inform you about the naming of the terminals at your installation.

The iounit name '/dev/tty' will always refer to the terminal at which you are working.

4.2. Printers.

A printer is normally, but not always, an output-only device. It is like a typewriter without a keyboard, and it is used to provide computer output on paper.

Normally, the iounit names of printers are:

```
/dev/print0
/dev/print1
/dev/print2
:
:
etc.
```

The iounit names contain a number, the so-called 'printer number'. Occationally, printers may have other names as well. For example, you may prefer to refer to your own printer as '/dev/Johns_print' rather than '/dev/print5'. This can be arranged by your System Administrator.

The System Administrator may inform you about the naming of the printers at your installation.

4.3. Disks.

A disk is a magnetic device on which data can be stored and retrieved. Disks may be fast or slow devices, they may be removable or fixed. A special kind of disk is the so-called 'floppy disk' or 'diskette', which looks like a grammophone record.

Normally, the iounit names of disks are:

```
/dev/disk0
/dev/disk1
/dev/disk2
:
:
etc.
```

The iounit names contain a number, the so-called 'disk number'. Occationally, disk may have other names as well. For example, it may be convenient to refer to the floppy disk as '/dev/floppy' rather than '/dev/disk5'. This can be arranged by your System Administrator.

The System Administrator may inform you about the naming of the disks at your installation.

Note: The user is rarely allowed to perform i/o directly to or from a disk. Normally, the storage space on the disk is divided into several 'lumps' of data, called 'files'.

4.4. Files.

Files are groups of data located on disks. A file may contain text or some other kind of data. Files are not necessarily permanently present on the disk: They may be created or deleted as required, and the size of a file may grow or diminish.

A disk may be partitioned into files in many different manners, called 'file systems'. The Supermax Operating System is able to handle two such file systems, called 'the Unix file system' and 'Mikfile'. Mikfile was originally created for the SPC/1 micro computer from DDE. Disks with files created on an SPC/1 may be used directly on a Supermax using the Mikfile file system. The Unix file system is the basic file system on the Supermax.

The Mikfile file system is described in the Supermax System Operation Guide.

The Unix file system is described in the following chapter.

4.5. Boxes.

Sometimes programs want to communicate with one another. For this purpose a running program may create a so-called 'box'. A box may be thought of as a fast file to which one program may write while another program is reading from it. Boxes are described in greater detail in the Supermax System Operation Guide.

Typically, the iounit names of boxes consist of the character strings '/dev/box/', '/dev/sysbox/', or /dev/combox/' followed by up to eight characters.

Examples of valid box names are:

```
/dev/box/aragorn
/dev/sysbox/strider
/dev/combox/1a2b3c4d
```

4.6. The null device.

One iounit in the system is called the null device. When data is written to the null device, the data is simply discarded. When a program tries to read from the null device, the device immediately signals end-of-file, that is, no more data available.

The iounit name of the null device is

```
/dev/null
```

What, then, do you use the null device for? Well, many programs produce output which you from time to time feel is irrelevant. Suppose, for example, that you have a program that counts the number of lines in a file and at the same time writes the lines to an iounit. Assume that the program asks you to enter the name of the iounit to which it is to write the lines, and assume that for the present you are interested only in the number of lines in the file and don't want the listing of the lines. If you specify '/dev/null' as the iounit to which the program writes the lines, the program will run faster and only the count will be produced; provided, of course, that the count is not output to the same iounit as the line list.

5. The Unix File System.

On practically all computers - regardless of the file system used - it is customary to assign each file a name. These names are stored in one or more so-called directories or catalogs. In the Unix file system the files are grouped in a structure that looks like a tree.

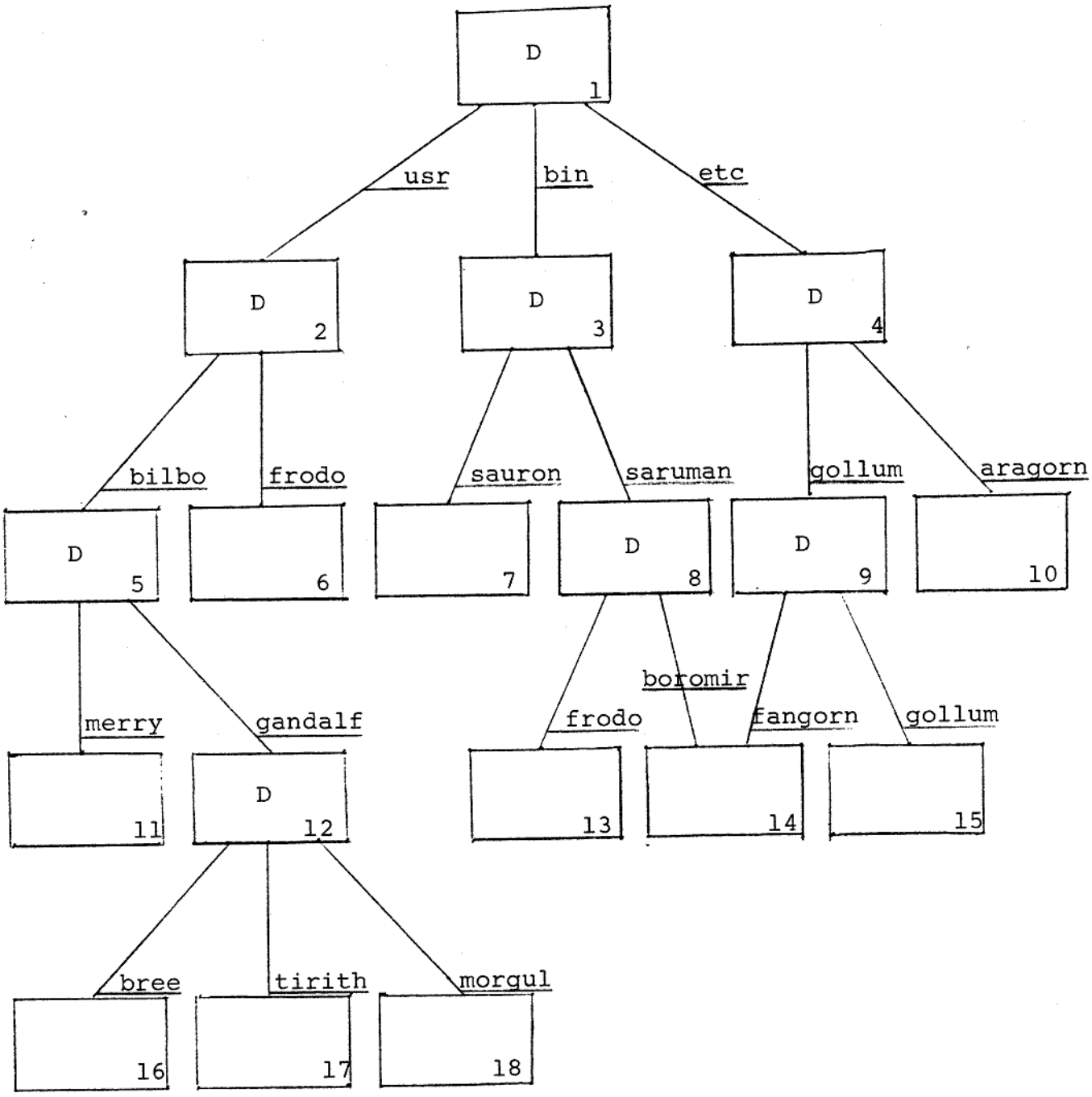
It is customary to draw the file system 'tree' with the root at the top and the branches at the bottom. (But that's how trees grow on the southern hemisphere, isn't it?)

Please refer to the example on the following page. This example is to be interpreted in the following manner:

Each box represents a file. The number given within the file has nothing to do with the file system, it is put there so that it is possible to refer to the file in this text.

Several of the files contain the letter D. These files are directories. In the Unix file system, directories are just a special kind of files.

The files are linked together by lines. Each line has a name. The fact that file number 11 is linked to file (directory) number 5 by a line labelled 'merry' should be interpreted in the following manner: A reference to file number 11 is stored in directory file number 5 under the name 'merry'.


D
1
usr
bin
etc
D
2
D
3
D
4
bilbo
frodo
sauron
saruman
gollum
aragorn
D
5
6
7
D
8
D
9
10
boromir
merry
gandalf
frodo
fangorn
gollum
11
D
12
13
14
15
bree
tirith
morgul
16
17
18

Let us try to determine the full name of file number 11. A reference to file number 11 is stored in directory file number 5 under the name 'merry'. A reference to file number 5 is stored in directory number 2 under the name 'bilbo'. A reference to file number 2 is stored in directory number 1 under the name 'usr'. Now, in order to specify the full name of file number 11 we put these file names together starting at directory number 1, seperating each component file name by a slash:

/usr/bilbo/merry

This is called the 'pathname' of file number 11, because this name specified the 'path' along the branches of the tree that lead to file number 11.

As another example, the pathname of file number 6 is /usr/frodo and the pathname of file number 13 is /bin/saruman/frodo. Note that although the word 'frodo' is a component of both these pathnames, there is no ambiguity, because the first 'frodo' is located in directory file number 2, whereas the second 'frodo' is located in directory file number 8.

File number 14 is interesting in that is has two pathnames. Both /bin/saruman/boromir and /etc/gollum/fangorn designate file number 14. This is not uncommon in the Unix file system, and it may be practical in certain circumstances. We say that file number 14 is 'linked' to both directory file 8 and 9.

Although directories are just a special kind of files, they differ from normal files in that the only data that can be stored in directories are references to other files.

Directories have pathnames themselves. For example, the pathname of directory file number 5 is /usr/bilbo. The pathname of directory file number 1 is simply /; this directory is called the 'root directory' or simply the 'root'.

Instead of saying "the directory /usr/bilbo contains a reference, labelled 'merry' to another file", we normally say "the directory /usr/bilbo contains a file called 'merry'".

Each component of a pathname may be at most 14 characters long. Note that the pathname components 'Frodo', 'frodo', and 'FRODO' are considered different. Normally, the characters used in pathname components

are the letters a-z (practically always in lower case), digits, the underscore character (_), and the period character (.). Other characters may be used, but this is rarely done.

5.1. Current Directory.

The Unix file system (a so-called hierarchical file system) can be a very useful way of storing data. As noted in chapter 3 a user is assigned a 'home directory' by the system administrator. The home directory is a directory which is the property of that user. In this directory the user may create all his or her files. If a user has his own home directory, he will not need to worry about whether file 'alpha' is his own or belongs to someone else. It is further possible to group related files together by putting them in the same directory, and to seperate unrelated files by putting them in different directories. This will greatly ease file administration.

Of course it can be very tedious to specify a full pathname every time a file is referenced. Pathnames can often be very long. There is, however, an easier way of specifying a pathname.

Every running program (for example, your operator communications program) is assigned a 'current directory'.

When a pathname starting with a slash is specified, the path is assumed to start at the root directory. For example, in the pathname /etc/aragorn, the name 'etc' is located in the root directory. If, however, the first character is not a slash, the path is assumed to start at the current directory.

If, for example, the current directory of the program you are running is directory file number 5 (also known as /usr/bilbo), you may refer to file 11 simply as:

```
merry
```

And you may refer to file number 17 simply as:

```
gandalf/tirith
```

In this case the names 'merry' and 'gandalf' are located in your current directory. Of course, you may also use the full pathnames /usr/bilbo/merry and /usr/bilbo/gandalf/tirith, if you prefer that.

If you wish to refer to file number 10, you will still have to specify the full pathname.

When you log on to the Supermax computer your operator communications program will be assigned a current directory which is identical to your home directory. Some operator communications programs allow the user to change the current directory at will. This is, for example, the case with vox and shell.

5.2. Special Files.

As noted in the previous chapter, the iounit names of a terminal and a printer are, for example, /dev/tty04 and /dev/print3. Note that these iounit names have exactly the same format as normal files located in a directory called /dev.

There does, in fact, exist a directory called /dev containing files tty04 and print3 and all the other names of i/o devices. These files, however, do not contain data as normal files do. Instead they contain a reference to the particular i/o device. Such files are termed 'special files'. (For historical reasons Unix distinguishes between 'block special files' and 'character special files', but this distinction has no meaning any more, so we have dropped it.) Whenever a program tries to perform an operation on a special file, the operating system redirects this operation to the relevant i/o device.

6. Access Rights.

This chapter is about security. In many cases you may want to protect the contents of your files against other users. The reason for this may either be that you fear the some other user (or perhaps yourself) will accidentally or maliciously destroy the contents of your files. Or your files may contain confidential information which you do not want others to access.

To protect your files, and indeed all iounits, against unathorized access, each iounit is assigned an 'owner ID', a 'group ID', and a set of 'protection bits'. The owner ID and group ID are the user ID and group ID of the person who owns the iounit; for files and boxes this will normally be the person who created the file or box. The protection bits control the access rights for various users of the computer. This will be described in greater detail below.

6.1. User IDs and Group IDs.

Each user of the computer is assigned a user ID and a group ID by the System Administrator. These are numbers in the range from 0 to 65535.

Each user of the computer has a unique user ID, but several users may have the same group ID. Users with the same group ID will typically be working on the same project or belong to the same department in a company.

6.2. Protection Bits.

The protection bits of an iounit specify who has access to that iounit. A 'bit' may be thought of as a switch that may be either off or on. Each iounit has 12 such 'switches'. The first three are discussed in the Supermax System Operation Guide.

The last nine protection bits are grouped as three 'triplets' of three bits each. The first set of triplets specifies the access rights of the owner of the iounit, the second set specifies the access rights of users with the same group ID as the owner, the third set specifies the access rights of all others.

The three bits within each triplet have the following significance. If the first bit (the first 'switch') is on, the persons in question are allowed to read the contents of the iounit ('read-access' is granted). If the second bit is on, the persons in question are allowed to write to the iounit, that is, change its contents ('write-access' is granted). The third bit is relevant only for files containing programs and for directories. If this bit is on and the file contains a program, the persons in question are allowed to execute the program contained in the file ('execute-access' is granted). If the file is a directory, the bit allows the persons in question to refer to files located in the directory ('search-access' is granted).

If a person wants to access file /alpha/beta/gamma for reading, he or she must have search-access to the directories /, /alpha, and /alpha/beta, and read-access to the file /alpha/beta/gamma.

Here is how access rights are checked. If a user (or rather a program) wants to access an iounit the following steps are performed:

1) If the user's user ID and the owner ID of the iounit are identical, the first triplet is inspected to see if the desired access mode can be granted.
2) If the user's user ID and the owner ID of the iounit differ, but the user's group ID and the group ID of the iounit are identical, the second triplet is inspected to see if the desired access mode can be granted.
3) If both the user's user ID and group ID differ from the owner ID and group ID of the iounit, the third triplet is inspected to see if the desired access mode can be granted.

A typical set of access rights for a file may be: Read and write access to the owner of the file, read access to users with the same group ID as the owner of the file, and no access rights for others.

How do you set and change the access rights of an iounit? When a file or box is created, the program creating the iounit assigns a set of protection bits to it. The owner of the iounit will normally be the user who executed the program. You may, however, change the access rights of the iounits that you own, provided that your operator service program allows you to do so. Further, you may change the owner ID and/or group ID of the iounits that you own.

In order to create a file in a directory, write-access to that direc-

tory is necessary, because a reference to the new file is to be written in the directory.

The deletion of a file does not require write access to the file itself, but to the directory that contains a reference to that file. This is because the deletion of a file is done by removing the reference to the file in the directory in which it resides.

6.3. The Super-user.

One user of your computer, typically the System Administrator, is called the 'super-user'. The super-user has user ID 0.

The super-user has special rights within the computer. He may change the access rights of any iounit, including iounits he does not own, and he may change the owner and group ID of any iounit. Several other operations, such as aborting any program running on the computer or setting the internal clock of the computer, may be performed only by the super-user. Further, the super-user may access any iounit, regardless of the access rights associated with the iounit.

7. The Computer.

This chapter gives a very brief description of the internal structure of the Supermax computer. Although the normal user need not be aware of these things, a little understanding of the anatomy of the computer often helps in getting the most out of the machine.

Internally the Supermax computer really is up to 16 computers integrated into one. More specifically, the Supermax computer contains up to 16 independent micro processors, each having their own memory. Some of these micro processors are dedicated to special tasks, such as writing to and reading from a terminal, while others are general purpose 'computers' that perform the actual computing for you. The latter are called MCUs, 'Main Computing Units'. The other constituent 'computers' are called, for example, SIOC (Serial I/O Controller, which controls terminals and printers), DIOC (Disk I/O Controller, which takes care of disks), and CIOC (Communications I/O Controller, which handles communications).

As described in chapter 3, when you log on to the Supermax the statement 'You are now logging on to processor number: 1' is presented, and you are allowed to change the number indicated. This number is actually the number of the MCU on which you are going to run your programs. The processor number suggested at log on time is the one which the computer thinks is currently least used.

7.1. System Crash.

Nobody is infallible. Error-free programs are as rare as oases in a desert, or perhaps even more so. Therefore it is inevitable that now and then an error in the Supermax Operating System occurs. The system has been designed in such a way that it often is able to dectect internal errors and stop, rather than continue erroneous execution. This is known as a system crash.

One terminal on the computer in the so-called 'system crash terminal'. If you are working at this terminal when a system crash occurs, you will see the following text on the terminal:

```
**********************
**** SYSTEM CRASH ****
**********************

INSERT FLOPPY DISK ON WHICH TO DUMP MEMORY
NOTE: PREVIOUS DISK CONTENTS WILL BE DESTROYED!
TYPE 'DUMP' AND PRESS RETURN:
```

When you see this text, stop working immediately! Do not type 'DUMP' and press the return key, as the instructions suggest. Contact the System Administrator, who will know what to do in this case.