IDE Hard Disc System

User Guide

Risc developments

This User Guide was produced using *Ovation* -
the professional desktop pub|ishing system from
RISC Deve|opments

IDE Hard Disc System

User Guide

Risc developments

Reference is made in the text to the *RISC OS User Guide* and *Welcome Guide* supplied with each Archimedes and A3000 computer, and the *RISC OS Programmer's Reference Manual.* All these are published by Acorn Computers Limited, Cambridge.

Any correspondence should be addressed to the publisher.
RISC Developments Limited
117 Hatfield Road
St. Albans
Hertfordshire
ALl 4JS

Published by RISC Developments Limited

Edition 1
Published 1991: Issue 1

Contents

Preface

Scope

This manual covers version 1.12 and later of the IDE software ROM when used in conjunction with RISC OS version 2.00 or 2.01. The IDE version number can be found by entering the RISC OS command

```
*Podules
```

having first pressed key f12 to get a 'star' prompt if in the Desktop. Somewhere in the resulting list the following line, or similar, will appear.

```
RISC Developments IDE Interface (x.xx)
```

where the value in brackets is the version number. An 'a' after the version number indicates that the software is for the A3000 internal IDE interface.

If the version of your software is older than 1.12 then you can obtain a free upgrade from RISC Developments, by sending to the address given at the end of this section.

Using this manual

This manual is designed to contain all the information you are ever likely to need to use a RISC Developments IDE hard disc drive, but also to allow users of all levels to get the system up and running as quickly and painlessly as possible.

Chapter 1 introduces the IDE concept and the RISC Developments implementation. This section should be read by all users.

Chapter 2 covers installation of the IDE interface card and both internal and external drives. It also shows how to configure the software to match the installed hardware, and how to format IDE drives. This information should only be needed when initially installing the system, or adding additional drives.

Chapter 3 is probably the one that will be of most relevance to the average user. It explains how to use IDE discs, both from within the RISC OS Desktop, and from command line applications.

Chapter 4 addresses the problems that may be encountered when transferring existing applications to IDE drives. Such problems are luckily *few,* and can normally be solved with ease.

Chapter 5 is aimed at the programmer, detailing the calls necessary to access IDE discs and the IDE software at a low level.

Appendix A explains the meaning of the various error messages that can be produced by the IDE software. This only covers errors that are unique to the IDE system, and the information will only be needed when an inexplicable error is reported.

Appendix B contains useful information to help identify and cure any faults that may occur.

Appendix C contains specialist technical information. It explains the various problems that may be encountered when connecting 'unsolicited drives to the IDE interface, and then deals with the subject of *drive translation modes* - probably the most problematic area with IDE drives.

Further information

RISC Developments adopt a policy of continual development and improvement of all its products. Therefore, some of the information in this manual may be superseded in time. Any major amendments will be found in a printed release note included with this manual, while any minor ehanges or additions are detailed in a file called *ReadMe* on the utilities disc. This file is in plain text format, and can be loaded either into *Edit,* or into a DTP package such as RISC Development's *Ovation.*

Programmers who wish to access the IDE hardware directly, for example for implementing forms of dise protection, will need information that is beyond the scope of this user guide. RISC Developments can supply an application note containing technical details of the interface card for a handling charge of £2. This can be obtained from the address given on the following page, marking the envelope 'IDE Technical Information'. Additionally, most drive manufacturers will be able to supply data sheets relevant to their drive products.

Conventions used in this manual

Throughout this manual, various conventions are used to clarify the text.

Courier typeface is used to represent input typed at the keyboard. For example:

```
    *Devices
```

and Courier-Bold is used for output generated by the computer, for example:

```
   Drive Type : Conner Peripherals 100MB - CP3104
```

Within command definitions, angle brackets indicate parameters that must be supplied, while square brackets are used to enclose optional items. For example,

```
*IDEFSTimer <drive> [<delay>]
```

shows that the command *IDEFSTimer takes one parameter representing a drive number, and an optional second parameter representing a delay.

Problems and suggestions

All IDE systems are fully tested before dispatch. However, devices do fail occasionally, and we obviously have no control over any separately sourced drives connected to the interface card; therefore problems beyond our control can arise. In such cases, you should always check that everything has been installed properly, and the software configured correctly, before suspecting a hardware failure. Appendix B contains some useful hints that can help-out when unexplained problems occur.

If, after checking all possibilities, you believe there to be a problem with the IDE system then you should consult in the first instance the dealer from whom the package was purchased. If they are unable to help you, then you should contact RISC Developments directly at the following address:

RISC Developments Ltd.
Dept. IDE
117 Hatfield Road
St. Albans
Hertfordshire
AL1 4JS

Tel. 0727 40303
Fax. 0727 860263

When reporting problems, you must include the following:

The exact nature of the problem
Full details of the computer system and IDE interface in use
The IDE software version number (as explained at the start of this section)
Details of the IDE drives in use
Any other details you consider relevant

RISC Developments greatly value the opinions of their customers. Suggestions for improvement of any part of this package should also be sent to the above address.

1. What is IDE?

IDE hard drives

Any hard disc system requires two distinct blocks of electronic circuitry between the computer and the physical discs within the drive. The first of these, which is almost always mounted on the drive unit itself, is responsible for reading and writing of the disc data and control functions such as head stepping and power-on diagnostics.

The second circuit block is the *hard disc controller (HDC).* This is located between the computer's interface bus and the electronics on the drive. The HDC's function is to convert commands from the computer into the necessary signals needed by the drive. Traditionally, the HDC circuitry has been located within the computer, either as an integral part, or as an upgrade card. This is the method employed by the ST506 type HDC employed by ADFS in the Archimedes.

However, with the IDE system, which stands for *Integrated Drive Electronics,* the HDC is combined with the drive's electronics to form a single unit contained on the drive itself. With this arrangement only a relatively simple interface circuit is needed to 'glue' the IDE drive to the particular computer's bus - hence reducing the price of the interface card. Moreover, because IDE is now the main hard disc interface used by IBM PC compatibles, the cost of IDE drives is very favourable. These two factors result in a highly cost-effective hard drive solution.

The RISC Developments IDE interface

The RISC Developments IDE interface card performs two separate functions. Firstly, as just described, it provides the glue circuitry needed to connect the drive to the Archimedes podule bus. Secondly, the interface card is fitted with an EPROM that contains the software needed to allow IDE hard drives to be used within the RISC OS environment. This software consists of two RISC OS relocatable modules - the IDE filing system, IDEFS, which performs the low-level accesses to the drive, and the desktop filer, IDEFiler, which provides icons on the icon bar representing the IDE drives. Additionally, the ROM contains code and data to identify the interface card and load the software into RAM.

The internal A3000 IDE interface is special in that it is connected to the computer by an eight-bit data bus only, while the IDE drive uses a sixteen-bit bus. Additional circuitry is used to expand data being written to the drive, and to funnel data read back.

2. Installation

This chapter explains step-by-step how to install the various models of the RISC Developments IDE interface card and IDE drives. It then goes on to show how to configure the IDE software and format the hard drive (or drives).

A word of warning

Before attempting the installation proeess you should locate and read the relevant parts of the following sections. If you do not understand the installation process, or are in any doubt about performing it, then contact the dealer from whom you purchased the package. They will be able to install the upgrade for you, but you should expect to pay an appropriate charge for this.

Many electronic components are particularly sensitive to static electricity and some simple precautions are needed to avoid the possibility of damaging the IDE interface card, the computer, or the hard disc drive. You should avoid wearing nylon elothes or walking on synthetic carpets during the installation process. Any build-up of static charge in your body can be removed by touching an earthed point, such as an exposed water pipe, or any bare metal on an electrical appliance that is connected to the mains supply.

You will need the following tools to fit the IDE interface eard and drive:

No. 6 Pozidriv screwdriver
M2.5 Pozidriv screwdriver (A3000 only)
Medium sized flat-blade screwdriver (A3000 only)

Installing the Archimedes IDE interface (not A310)

1. Turn off the computer and disconnect it from the mains and all attached peripherals. Remove the monitor.
2. Referring to figure 1, locate and remove the five screws securing the case lid.
3. Remove the lid by sliding it back a few inches and lifting it off at a shallow angle. Sometimes considerable force is needed to remove the lid.
4. If you are not fitting an internal IDE drive proceed to step 14.
5. Locate plug PL8 which connects two pairs of red and black wires and two white wires to the bottom-lefthand corner of the circuit board (when viewed from the front). Unplug the connector avoiding pulling on the actual wires.
6. Take the cable adaptor assembly supplied with the IDE upgrade and attach the end with six pins on it to the now free connector removed from PL8. The adaptor should be orientated so that the cable colours line up.
7. Plug the six-way connector on the adaptor into PL8, ensuring that the side with the two lugs faces the locating tab (i.e. the white cables are nearest the back of the computer).
8. The LED assembly supplied with the upgrade is only needed for use with an A310, but should be kept in case it is needed in the future.

9. The IDE hard drive should have the mounting brackets and ribbon cable already fitted. If the cable becomes detached it is important to re-connect it with the red stripe on one side aligned with pin one of the drive connector, which can be located by referring to the information sheet packed with the drive.

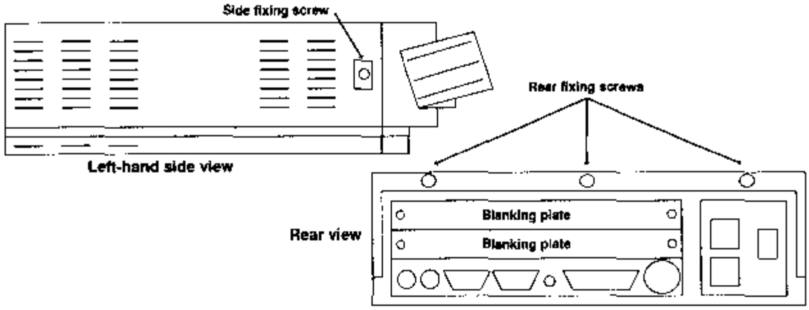


Figure 1. Archimedes top cover removal

10. Position the hard drive in the computer next to the floppy disc drive, resting it on the metal cross bar. You will need to insert the drive at an angle in order to clear the backplane.
11. Identify the unused hard disc drive power cable coming from the computer's power supply and attach this to the power connector on the IDE drive. You may need to cut a cable tie to free the cable. Although the power connector is shaped to make it non-reversible, it is not impossible to connect it back-to-front, and extra care should be taken. (Note: some low-profile hard drives will have a different type of power plug, and in this case the spare connector on the floppy disc drive power cable should be used instead.)
12. With some combinations of computer and hard drive you may find that the power cable is too short. In this case, RISC Developments can supply an extender for a nominal charge.
13. Position the drive so that the holes in the brackets align with those in the metal cross member and secure the drive using the two self-tapping screws provided. You may find that a piece of 'Blu-Tak' comes in handy to stop the screws slipping while trying to tighten them.
14. Identify the backplane, whieh is the vertical circuit board mounted across the middle of the computer. This will have on the rear side either two or four connectors to attach interface cards to. You may already have one or more interface cards fitted to this backplane.
15. If you are fitting a RISC Developments internal hard drive, it is best to fit the IDE interface card into the top, left-hand slot as viewed from the front of the computer as this avoids the drive cable fouling any other cards.
16. Having selected the slot to be used, it is necessary to remove the corresponding blanking plate shown in figure 1. If the interface card is being fitted next to another card then only the half-width blanking plate need be removed, leaving the 'T shaped joining piece.

17. Remove the IDE interface card from its protective packaging and holding it by its edges only, slide it in, component side up, from the rear of the machine and engage the 64-way plug with the corresponding socket on the backplane. Do not use excessive force and make sure that no pins on the connector are bent. Before pushing the card fully home, eonnect the free end of the cable from the internal IDE drive (if fitted) to the 40-way connector towards the back of the IDE interface card, with the red stripe facing left (when viewed from the front of the computer). Take care not to offset the connector from the pins.
18. You may find that in order for the backplate of the interface card to lie flush against the computers back panel you need to loosen the two screws holding the backplane and adjust it slightly. On Archimedes A310 and old-style 440s you should refer to the instructions supplied with the two small spacers that form part of the upgrade package, as you may need to fit these.
19. If the IDE interface card is being installed next to another card then you can now secure it using the screws that previously held the half-width blanking plate. Otherwise you will need to use the supplied half-width plate, 'T' joining piece and screws to extend the baekplate to its full width before securing it.
20. If an internal IDE drive has been fitted, connect the remaining three-way connector from the cable adaptor, or LED assembly, to the plug on the IDE interface card, with the lugs faeing the tab on the connector.
21. Refit the top cover by flexing the sides slightly, lowering it into position and sliding it forwards.
22. Refit the five screws holding the top cover.
23. Reconnect the mains supply and any peripherals.
24. The installation is now complete and you should proceed to the section *Testing the IDE interface card,* or *Connecting external drives* as appropriate, and then the section on *Software installation.*

Installing the Archimedes IDE interface in an A310

Installing the IDE interfaee card into a 310 is very similar to installing it into any other model of Archimedes. However, if an internal drive is being fitted then you should replace steps 5 to 8 above with those given below.

5. Locate plug PL9 which connects two white, one red and one black wire to the lower-lefthand corner of the computer's main circuit board. Disconnect this by pulling the connector (not the wires) gently.
6. Remove the two part plastic moulding from the front of the computer by undoing the three screws on the underside at the front, and the one screw on each side, noting the screw lengths. Separate the grey front facia from the rest of the moulding by undoing the two internal screws.
7. Fix the LED supplied with the IDE package into the spare slot below the power light, using either super glue or 'hot melt'. Reassemble the two parts of the moulding and refit the complete moulding to the case ensuring that the cables pass through and appear between the power supply and main circuit board. Take extra care to locate the disc drive eject button properly.
8. Refit PL9, ensuring that the side with the two lugs faces the locating tab.

The cable adaptor assembly supplied with the IDE interface card is not needed with an A310, but should be kept inease you upgrade your system at a later date.

If your computer only has a two-slot backplane, then the cable from the IDE interface card to the drive must be routed diagonally. If you find the cable is not long enough for this then please consult RISC Developments. If no backplane is fitted, then you will need to purchase one separately.

Installing the A3000 internal IDE interface

1. Turn off the computer and disconnect it from the mains and any peripherals.
2. Remove the monitor and monitor plinth if fitted.
3. Turn the computer upside down, and place it on a firm surface.
4. Locate and undo the centre fixing screw shown in figure 2. It is not necessary to remove this totally, but it must be loosened enough to free the top cover.
5. With the rear of the A3000 facing you, locate the two lock plates shown in figure 3 and loosen the retaining serews sufficiently to allows the plates to be slid out.
6. Turn the A3000 the correct way up and by simultaneously pressing the two clips that were held by the lock plates separate the top cover from the rest of the case just enough to prevent the clips re-engaging.
7. Turn the computer upside down again and release the three front clips (figure 2) by inserting a flat-bladed screwdriver behind each clip and twisting it. If necessary, push the top cover up slightly to stop the clips from re-engaging.

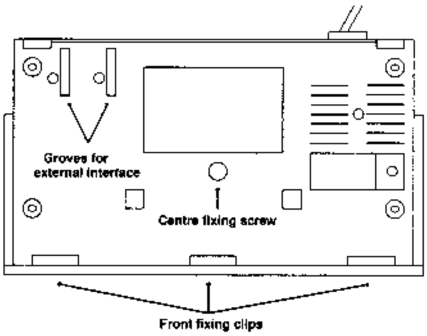


Figure 2. The underside of the A3000

8. Very carefully turn the A3000 the correct way up again, making sure the case doesn't separate. The top cover can now be lifted clear.
9. Locate the blanking plate shown in figure 3 and remove it by undoing the screw on either side. The plate should be saved in case the IDE interface is removed at a later date.
10. Identify the two seventeen-way sockets located on the A3000's main circuit board (see figure 4). Remove the IDE interface card from its protective bag, and holding by its edges only, position it so that the pins on the underside of the card align with the two sockets.

11. If necessary, bend the red and black power leads connected to the eomputer's main cireuit board so that they will not catch on the IDE interface card when ii is pushed down. Also check that the yellow and green earth cable connecting the power supply to the corner of the circuit board will not be trapped by the IDE interface card.

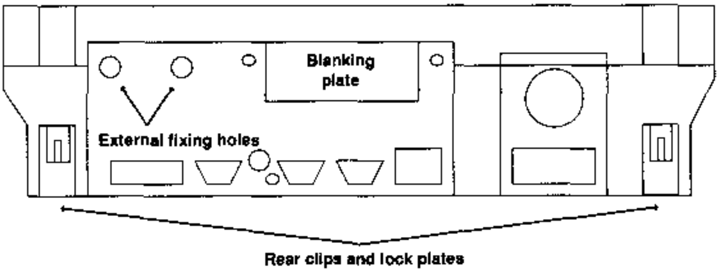


Figure 3. The A3000's rear panel

12. Very carefully push the IDE interface card down until both connectors are fully mated. Do not use excessive force, and ensure that all the pins line up, and none of the cables in the area are pinched under the card. If the card will not push home check that it is not being fouled by any of the cables before proceeding. The connectors are not properly inserted until the plastic bodies on the seventeen-way plugs rest on the socket housings.
13. With the rear of the computer facing you, use the two screws that held the blanking plate in to secure the IDE interface card to the rear panel. Note that there is considerable flexibility in the connectors between the card and the main circuit board, and you may need to move the board gently to align the screw holes.

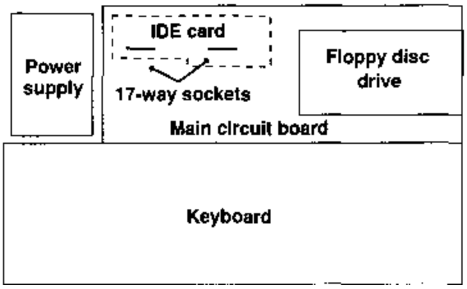


Figure 4. Positioning the A3000 IDE interface card

14. Position the top cover in place and push down at the back to engage the rear clips, ensuring that the back panel rests in the slot on the top lid.
15. Engage the front clips by gripping the front of the top cover over each clip and pulling forward and downwards until the clips latch.
16. Refit the lock plates to the rear clips, making sure the washers lie on the outside.
17. Turn the A3000 over and tighten the centre fixing screw.

18. The installation is now complete and you should proceed to the section *Testing the IDE interface card,* or *Connecting external drives* as appropriate and then *Software installation.*

Installing the A3000 external IDE interface

1. Turn off the computer and disconnect it from the mains and all attached peripherals.
2. Remove the monitor and monitor plinth if fitted.
3. Turn the computer upside down with the rear facing you.
4. Take the interface card unit and slide the flange into the two groves on the underside of the computer. These can be seen in figure 2.
5. Push the interface card home until it butts up against the rear panel. While doing this ensure that the 64-way plug on the interface engages with the corresponding socket on the computer.
6. While supporting the card, turn the computer the correct way up.
7. Fit the two large retaining screws through the tab on the interface card case.
8. Refit the monitor plinth and monitor and re-attach any peripherals
9. Connect the IDE drive as described in the following section.

Connecting external drives

RISC Developments external drives are provided with two cables, one to connect the drive to the mains supply, and one to connect it to the IDE interface card. The 40-way socket at one end of the cable should be connected to the plug on the rear of the IDE interface card, and the 50-way plug at the other end to either of the two sockets on the rear of the drive unit.

If two external drives are being connected then the second drive should be daisy chained to the spare socket on the first using the short spur cable supplied with the drive.

Testing the IDE interface card

It is not possible to test the IDE interface card fully until the software has been configured as described next. However, as a very quick test, turn on the computer and issue the command *Podules (pressing f12 first if necessary). You should see the IDE interface card in the resulting list. If not, or if the computer fails to function normally, turn off immediately and re-check the installation.

Software installation

Although the software needed to control IDE disc drives is included on the interface card, it is necessary to initially configure the system to recognise this software, and to set up certain preferences.

The configuration can be split into two distinct operations. The first, which is only necessary when the interface is first installed, or if a program has altered the configuration, is necessary to ensure that the two relocatable modules are loaded correctly. To do this, enter the commands:

```
*RMReinit Podule
*Modules
```

remembering to press key f12 if you are within the Desktop to get the star prompt.

Somewhere in the resulting list, you should see the module titles:

```
IDEFiler
IDEFS
```

If any of these are missing, you should enter the command:

```
*RMReInit <module>
```

where *module* is the title of the missing module, as given above. Having completed this for all the missing modules, press Ctrl-Reset to reload and initialise them.

Having ensured that the modules are present, it is necessary to issue a number of configuration commands to set various options before the system can be used. Again, press key f12 to get the star prompt.

The first of these is:

```
*Configure IDEFSDiscs <0~2>
```

for example:

```
*Configure IDEFSDiscs 1
```

This command specifies the number of IDE hard drives attached to the IDE interface card. Use a value of 1 or 2 depending on the number of drives.

The next command is:

```
*Configure IDEFSDrive <n>
```

for example,

```
*Configure IDEFSDrive 4
```

This selects which drive will be used as the default in the absence of another one being mounted. This should normally be drive 4.

To speed up disc accesses, IDEFS allows for part of the directory structure to be cached. The amount of cache space can be set using the command:

```
*Configure IDEFSDirCache <nK>
```

for example,

```
*Configure IDEFSDirCache 16K
```

which will set the cache size to the specified value. Too low a value will result in unnecessary disc accesses, while too high a value will waste memory. You should initially set a value of zero `(*Configure IDEFSDirCache OK)` which will set a default size appropriate to the available memory.

The final configuration command provided by IDEFS is:

```
*Configure IDEFSTimer <delay>
```

for example,

```
*Configure IDEFSTimer 60
```

This sets the default timeout for drives that support a standby mode of operation. This default timeout is applied to all drives when they are first mounted, but will only have any effect on certain drives, for example the PrairieTek range. You should only use this command if you have fitted an internal A3000 drive, and want the drive to shutdown when idle, rather than remain spinning. See the description of *IDEFSTimer (which allows the eonfigured default to be overridden) in the next chapter for more details.

Having configured IDEFS, you should issue the RISC OS commands:

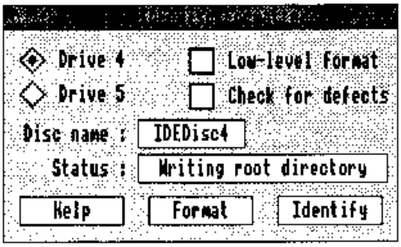
```
    *Configure FileSystem IDEFS
    *Configure Dir
```

to make IDEFS the default filing system, and force it to mount the disc on start-up. See the *RISC OS User Guide* for more details of these commands.

You must now press Ctrl-Break to load the new configuration. The installation process is then complete. If, however, you are installing an IDE drive that was not produced by RISC Developments then it may need formatting as described in the following section. (RISC Developments' drives are supplied already formatted.)

Formatting IDE drives

Before any hard disc drive can be used for the storage of data it is necessary to format the disc. This both lays down the basic structure of sectors into which data will be stored, and creates some vital boot information needed to make the disc useable. To simplify the process of formatting IDE discs, the utilities disc supplied with the interface card contains a desktop-based formatter. The formatter is called *IDEForm,* and when run will bring up the dialogue box shown in figure 5.



Disc name : IDEDisc4
Status : Writing root directory

Figure 5. The *IDEForm* dialogue box

The first step in the formatting process is to select the drive to be formatted by clicking on one of the buttons labelled *Drive 4* or *Drive 5.* The name to be given to the specified disc will default to 'IDEDiscn', where 'n' is the drive number. You can however select an alternate name by typing it into the writable field entitled *Disc name.*

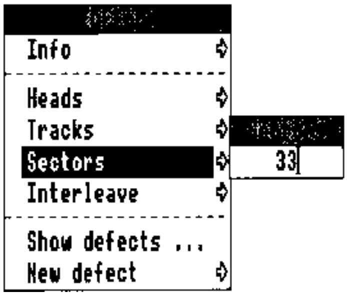
Selecting the *Low-level format* button will cause the drive to be formatted at a very low-level by recreating each sector individually. This is only necessary if the drive isn't already formatted (most drives are factory pre-formatted), or has previously been formatted with a large number of defects and you wish to try and re-format some of the bad sectors.

Selecting *Check for defects* will cause the disc to be verified for any unusable sectors before formatting. These sectors will be mapped out during formatting, and not used for data storage. This process occurs automatically when a low-level format is performed, and so the button is faded in this case.

Clicking on the *Format* button will start the actual formatting process. If you have opted to perform a low-level format then a warning will be issued that this may take some time, and you are given the chance to abort the operation. Another warning will be issued if the formatter detects that the disc is already formatted. A final warning and a chance to quit is given immediately before the formatting operation is started. The actual formatting is single-tasking to prevent possible problems with other software trying to access the disc being formatted. During this period the RISC OS hourglass will be displayed and the *Status* field of the dialogue box will indicate the process.

Clicking on the *Identify* button will open a window displaying various parameters about the drive being formatted. This information is similar to that listed by the *IDEFSDevices command described in the next chapter. Interactive help is available via the *Help* application, and clicking on the *Help* button will display a window containing additional help information.

For specialist formatting, clicking Menu over the *IDEForm* dialogue box will bring up the menu shown in figure 6.



Moving right over the *Info* entry will just produce a standard RISC OS information box about the program. The next four menu entries allow the drive geometry (heads, tracks, sectors and interleave) to be adjusted (see appendix C.) These are initially set to the defaults for the drive being formatted, but can be changed by entering new values into the writable icons.

Choosing *Show defects* will open a window showing the current defect list. Any defect can be deleted by double clicking on its entry in the list and responding 'yes' to the confirming prompt. *New defect* provides a writable entry to allow a new defect to be added to the list. This can be in the form of a disc address (in hexadecimal), or a combination of head, track and sector numbers, with each component separated by a comma.

3. Using IDE hard discs

One of the beauties of RISC OS is the orthogonality it provides for filing systems. This means that both the programmer ean write applications that will work with any filing system, and the user can use an application independent of the filing system being used. The first part of this chapter covers details of using IDE drives from within the Desktop. This is followed by information on how to specify IDE drives in file pathnames, and how to seleet the IDE filing system. The various star commands that are specific to the IDE filing system are then described, and the chapter finishes with a description of how to set up the PC Emulator.

Using IDE drives from the Desktop

From within the RISC OS Desktop, IDE drives appear identical to ADFS-based hard disc drives in almost every respect. The same icon is used for both ADFS and IDE hard drives, though to distinguish between them if both types are fitted, the drive number for IDE discs is prefixed with the word 'IDE'. Figure 7 shows the typical appearance of the icon bar. Unlike current versions of ADFS, as soon as an IDE drive is recognised, the text under its ieon changes to show the disc name, as seen in figure 7. More on this is given later.

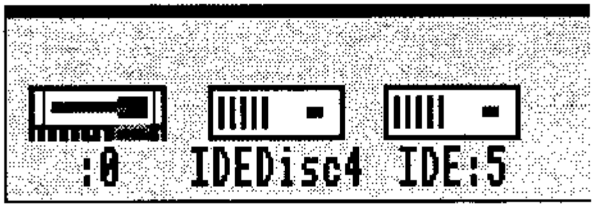


Figure 7. The appearance of IDE drives

If no IDE drive icons appear then you should check that the software has been configured properly, as explained in the previous chapter.

The root directory of an IDE disc is accessed by clicking on the appropriate drive icon, just as for an ADFS disc. All operations on the disc contents thereafter are exactly as for other disc types. If for any reason the drive cannot be accessed, an error will be given, either immediately or after a short delay. Trouble-shooting is covered in appendix B.

Pressing the menu button over an IDE drive icon will produce a menu similar to the one for an ADFS hard disc drive, as shown in figure 8.

As with all other types of hard disc drives, the *Format* and *Backup* options are greyed out, as both of these operations require special programs for hard disc drives. The remaining four options perform the same functions, and operate in the same way, as their ADFS equivalents.

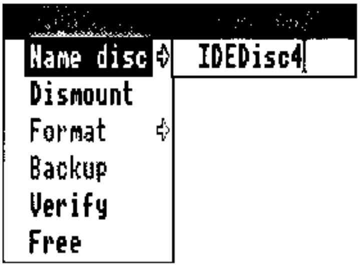


Figure 8. The IDE disc menu

Most IDE drives are auto-parking, and therefore it is not essential to dismount them before powering them down. However, it is good practice to dismount any hard drive before turning the system off, so that RISC OS can perform any tidying up needed.

For more details of manipulating files within the Desktop, refer to the *RISC OS User Guide,* supplied with your computer.

A note about the icon bar names

As already explained, the IDEFiler will attempt to display the drive name under its icon as soon as the drive can be identified. It does this by trying to access each drive at regular intervals until all the names are known. After this, the name displayed will only be changed if the actual name is changed by choosing the *Name disc* option from the icon's menu. The displayed name will not be changed if the disc name is altered using the *NameDisc command. In this case it will be necessary to quit the Desktop and re-enter it in order to reflect the change.

Further, whenever the name under an icon is changed, the icon has to be deleted and re-created, as its width may have changed. The effect of this is to push the IDE drive icons towards the centre of the icon bar whenever the text under one of them is changed.

Using IDE discs from outside the Desktop

IDE discs can be used in non-desktop programs and from the command line in the same way as discs belonging to any other filing system.

The command:

```
    *IDEFS
```

will select IDEFS as the current filing system. Alternatively, IDEFS can be configured as the current filing system as explained in the previous chapter. The current filing system will be used everywhere that a file has to accessed, and another filing system is not explicitly given in the pathname.

IDEFS can be specified as a temporary filing system for the duration of a command by prefixing the command with the filing system name, just as for other filing systems. For example,

```
    *IDEFS:FREE 4
or *-IDEFS-FREE 4
```

will both direct the command 'FREE 4' to IDEFS, regardless of the current filing system. The first form of the command given above (fsname:) should always be used in preference to the obsolescent second form (-fsname-).

Similarly, IDEFS can be specified as part of a full path name. For example:

```
    LOAD "IDEFS::HardDisc4.$.BasicProgs.Game"
or LOAD "-IDEFS-:HardDisc4.$.BasicProgs.Game"
```

The first form being the preferred one. Again, more details can be found in the *RISC OS User Guide.*

Star commands specific to IDEFS

There are three star commands provided by the IDE filing system in addition to the configuration commands described in the previous chapter. The first of these, *IDEFS, has already been described.

The command *IDEFSTimer is used to control the *sleep made* provided by some IDE drives. These sleep modes are provided to shut the drive down and stop it rotating after it hasn't been accessed for a period of time. The drive is then restarted when it is next accessed. This feature is typically used when IDE drives are fitted to portable computers where battery power is at a premium. Normally, it is better to keep a drive running all the time as this prevents delays when accessing data and reduces wear on the drive. However, there are some circumstances when the sleep mode may be useful - for example in a simulation application where the computer runs constantly for many days, but only occasionally writes data to disc. In this ease, stopping the drive will reduce power consumption and hence heat dissipation.

The syntax of the command is:

```
    *IDEFSTimer <drive> [<delay>]
```

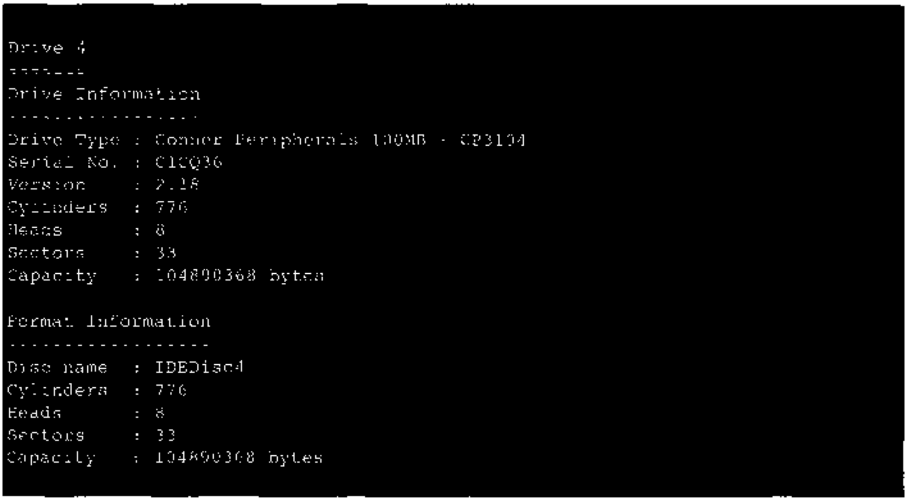
where *drive* is a RISC OS drive number, and *delay* is a time delay in seconds. If the delay is zero, or missed off altogether, the sleep mode is disabled and the drive will run continuously. Otherwise, the timeout will be set to the specified delay, rounded up to the nearest five seconds. The maximum delay is 150 seconds (two and a half minutes). Setting a delay of 5 seconds (the minimum) will cause the drive to behave like a floppy, being shutdown after every operation.

Not all drives support sleep modes, and if *IDEFSTimer is used on drives that do not, it will be ignored. Most 2.5" drives provide a sleep mode, as do some 1" high 3.5" units.

The final star command, which takes no parameters, is:

```
    *IDEFSDevices
```

The purpose of this command is to display details about the attached drives and the way in which they are formatted. Figure 9 shows the typical output for a single drive system.



```
Drive 4
=======
Drive Information
-----------------
Drive Type : Conner Peripherals 100MB - CP3104
Serial No. : C1CQ36
Version    : 2.18
Cylinders  : 776
Heads      : 8
Sectors    : 33
Capacity   : 104890368 bytes

Format Information
------------------
Disc name  : IDEDisc4
Cylinders  : 776
Heads      : 8
Sectors    : 33
Capacity   : 104890368 bytes
```

Figure 9. Typical output from *IDEFSDevices

The drive type, serial number and revision are useful when reporting any problems encountered with the drive, while the combination of the remaining hardware information and the format data can be used to check that the drive has been formatted in its native mode and not using a translation mode. This is described in appendix C.

The PC Emulator and IDE discs

To make full use of the PC Emulator with IDE discs it is necessary to create one or more partitions on the disc to act as DOS hard disc drives. It is only possible to do this properly with version 1.33 of the PC Emulator or later. Owners of earlier versions should contact their dealer for information on upgrading.

The size of a drive partition can be set between about 1 and 32 Mbytes, and must be a multiple of the value &8800. Some versions of the PC Emulator (those running DR DOS) require the partition to be set to all zeros initially, and the PC Emulator must be told in its boot sequence where to locate the partition. To simplify this entire process an application called *PCPMake* is provided on the accompanying utilities discs. When run, you are presented with the dialogue box shown in figure 10.

By selecting and deselecting the option buttons labelled C: and *D:* you can select which DOS drive you wish to create a partition for. Both partitions can be created together if required simply be selecting both C: and *D:*. Once selected, you can set the desired size of the partition by clicking on the up and down arrows. The size displayed is only approximate because of the need to keep to a multiple of &8800.

The three option boxes on the lower left of the dialogue box control the behaviour when the partition is created. The first of these, *Zero partition,* controls whether the partitions will be filled with zeros or not when created. This option is on by default, but can be turned off to speed up the creation, provided the partition will be used with MS DOS and not DR DOS.

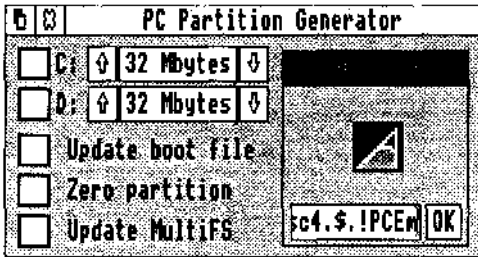


Figure 10. The *PCPMake* dialogue box

If *Update boot file* is selected, which it is by default, *PCPMake* will attempt to update the PC Emulator's boot files to point to the new partitions when they have been created. For this to be possible, the application must know the whereabouts of the PC Emulator. If a directory display containing the Emulator had been opened before running *PCPMake* then the necessary information is read automatically. If this is not the case, this option will be faded out and unselectable. You can however, drag the PC Emulator application onto the *PCPMake* window, and it will then pick up the relevant information. If the *Update boot file* option is not selected then no attempt is made to alter the boot files.

The *Update MultiFS* option performs a similar function to the previous option, but is used to modify the *MultiFS* DOS filing system provided with PC Emulator version 1.60 onwards, so that it can locate the partitions. Again, this is only available if *MultiFS* has been seen, or has been dragged into the window.

Once all the options have been set, you can create the partitions by dragging the icon at the right of the dialogue box to the appropriate directory display. Alternatively, you can type in a pathname and click on OK or press Return. Note, however, that unlike a normal save operation, the pathname should be the name of the directory into which the partitions will be saved, and not the partition names themselves - these are always set to 'Drive_C' and 'Drive_D'.

If *PCPMake* 'knows' where the PC Emulator is, then it will offer a default path that will place the partitions within the PC Emulator's application directory. If the *Update boot file* option is also selected, the updating is done so that the partitions are accessed relative to the PC Emulator's directory, and not by an absolute pathname. This allows the Emulator and its partitions to be moved or copied.

While creating the partition, a warning will be given if an existing partition is about to be overwritten, and confirmation is sought before proceeding.

Having created the DOS partition, or partitions, it is neeessary to perform a number of steps from within the DOS environment before the drive can be used. The way this is done depends on whether MS DOS or DR DOS is being used. In either case, though, you should start by performing a hard reset (Ctrl-Break), running the PC Emulator, and booting it from the DOS disc supplied as part of the PC Emulator package.

If you are using DR DOS, a setup program will be run automatically on boot-up. You should follow the instructions given on screen to first run the *FDISK* utility to setup the disc for DOS, and then to install DOS onto the hard drive. More details can be found in the user guide supplied with DR DOS.

Installation with MS DOS is rather more tricky as no automatic installer is provided. Having booted the PC Emulator you should run the *FDISK* utility. Accept all the defaults by pressing Return at each prompt, and the system will initialise the partition and re-boot DOS. If a second partition has been ereated, you should repeat the above, but first choosing option '5' from *FDISK* to change to the next fixed disc. The DOS system files must then be copied to the hard disc by using the command:

```
        Format C: /S
```

Reply 'yes' to the warning about all data on the fixed disc being lost - it only means the DOS partition file, not the RISC OS files as well!

Having done this, you must then copy the DOS utilities manually to the hard drive, and set up 'autoexec.bat' and 'config.sys' files if required. More details are given with the PC Emulator package.

4. Converting existing applications

This chapter attempts to solve any problems that may arise when trying to integrate existing software into an IDE environment. The nature of the RISC OS Desktop environment ensures that the problems with legally written multi-tasking applications are likely to be few and far between. Such applications are termed *RISC OS compliant,* and the software's documentation should identify this. There will however, always be some programs that will need changes to a greater or lesser extent to run under IDEFS, and a few specialised programs that will be completely incompatible.

The RISC Developments IDE software does comply fully with Acorn's specification for IDE systems. This should minimise any problems.

It is not possible in the space of two pages to predict every programming technique that is likely to be used, and give advise accordingly. Instead, the information here is very general, covering just anticipated problem areas. The problems described are not just related to IDE systems - any software that has been written to run on a particular filing system (most probably ADFS) will need modifications for other filing systems. The remainder of this section categorises the problems and gives hints as to solving them.

Hard-wired filing system names

Some non RISC OS compliant applications will hard-wire filing system names into a program. This will either be in the form of commands to select a filing system, such as *ADFS, or as part of a pathname, for example "adfs::4.$.Temp.Setup". In this case, conversion should simply consist of changing all occurrences of the filing system name, in this case from "ADFS" to "IDEFS".

RISC OS compliant applications should be written to access any resource files relative to <Obey$Dir> which is set up by RISC OS before running the application, thereby removing the need for hard-wired names.

Filing system numbers

A more subtle occurrence of the above problem is when a program hard-wires a filing system number. All RISC OS filing systems are identified by both name and number, and there is a remote chance that a set number may be used. ADFS has the filing system number 8, while that for IDEFS is 49. The workaround for this simply involves changing the number within the program.

Killing modules

Some applications, particularly when running on a 1Mbyte machine, will attempt to kill various relocatable modules to free more memory. This is considered a bad practice in any circumstance, but if a program must do this, it should always positively identify the modules to be killed. This means that it should have a list of modules that can be killed, and kill these, rather than a list of modules that must be kept, and kill everything else. Unfortunately, some programs do use the latter method, and these will inadvertently kill the IDE driver software modules. In this case, you must modify the program to preserve the following modules:

IDEFiler
IDEFS

Low level calls

A number of programs will make use of low level calls similar to those described in the next chapter. Unfortunately, the current RISC OS filing system protocols do not provide any legal means of low level calls being filing system independent, even if several filing systems provide effectively identical calls.

All these low level calls will be via SWI calls, and table 1 gives a comparison between the ADFS low level calls, and those provided by IDEFS. You should also note, that although both IDEFS and ADFS provide a *DiscOp* call, the range of reason codes and other features supported by both is not the same.

ADFS_DiscOp	(&40240) =	IDEFS_DiscOP	(&41FC0)
ADFS_HDC	(&40241) =	No equivalent	
ADFS_Drives	(&40242) =	IDEFS_Drives	(&41FC2)
ADFS_FreeSpace	(&40243) =	IDEFS_FreeSpace	(&41FC3)
ADFS_Retries	(&40244) =	No equivalent	
ADFS_DescribeDisc	(&40245)	IDEFS_DescribeDisc	(&41FC5)

Table 1. Filing system SWI equivalences

Where no equivalent is available, it will be necessary to modify the application to remove the need for the call, though in some cases this may not be possible.

While table 1 only compares IDEFS and ADFS, other filing system based around *F ileC ore* (see chapter 5) should provide similar calls.

Using Ovation with IDE drives

If you wish to install *Ovation* - the professional desktop publishing package from RISC Developments onto a an IDE drive, you must obtain a special version of *Ovation* as the disc protection employed with the original version works on ADFS and SCSI discs only. If ordering *Ovation,* then please state that it is for an IDE drive. If you already have *Ovation,* then you should contact the Software Manager at the address given in the Preface to arrange an upgrade.

Help

If you do find any problems with commercial applications running under IDEFS, you should in the first instance check the program's documentation, and then if necessary contact the supplier of the application. They may well already know of the problem and have a new version that cures it.

5. Low level control of IDEFS

This chapter explains a set of low level calls to access IDE discs, and the IDE filing system, directly. All these calls take the form of RISC OS SWI calls. Knowledge of the information in this chapter will only be required by specialist programmers.

IDEFS, the IDE filing system, is designed to work in conjunction with the RISC OS *FileCore* module as described in volume 3 of the *RISC OS Programmer's Reference Manual* (PRM). The first four calls described here correspond direetly to calls provided by *FileCore*. Further documentation of these calls can be found in the PRM. The remaining four calls are unique to IDEFS. Before explaining the individual calls, some background information is given.

Disc addresses

When accessing the contents of a disc directly, the concepts of directories and files no longer apply. Instead, locations on the disc are identified by *disc addresses,* which are analogous to memory addresses in that they identify each available location in a sequential order. Disc addresses range from zero, up to a maximum determined by the size of the disc. For example, a 20Mbyte disc will contain 20 * 1024 * 1024 bytes = 20971520. The valid range of disc addresses is therefore 0 to 20971519, or in hex, 0 to &13FFFFF.

Certain operations that require a disc address impose a restriction on the possible values, just as some operations require a memory address to be word-aligned. A general restriction that applies to all disc addresses is that they must specify the first byte of a sector, as disc-based data can only be accessed in whole sectors. On an IDE disc, the sector length (also called the block length) is 512 bytes, making 0, &200, &400 etc. legal disc addresses. Operations that work on whole tracks, such as ' seeking' to a particular track, require that the disc address specify the start of a complete track. This necessitates knowledge of how many sectors there are per track, and how many heads per cylinder. This information is contained in the dise record described below.

When specifying disc addresses as parameters to calls, the drive number in the range 0 to 7 is combined with the disc address in bits 29-31 to give a thirty-two bit value.

Disc records

When working at a low level, you need to know details such as the number of bytes per sector, the number of sectors per track, the number of heads (equivalent to the tracks per cylinder), the total number of cylinders etc. All this information is bundled together into a 64-byte disc record contained on each disc. When accessing disc data directly, it is also possible to override the default disc record with an alternative one. This is particularly useful when the disc does not have a correct disc record, for example when it is first formatted.

The actual format of disc records can be found in volume 3 of the *RISC OS Programmer's Reference Manual.*

The SWI calls

The remainder of this chapter details the available SWI calls. Information on how to make these calls from various languages can be found in volume 1 of the *RISC* OS *Programmer's Reference Manual.* The standard convention is used, whereby any registers not mentioned in the entry or exit conditions are preserved across the call. The SWI number is given in brackets after the name.

IDEFS_DiscOp

(SWI &41FC0)

On entry:

R1 bits 0-3 = reason code
bits 4-7 = option bits
bits 8-31 = bits 2-25 of pointer to alternative disc record (or zero)
R2 = disc address
R3 = pointer to buffer
R4 = length in bytes

On exit

R2 = disc address of next byte to transfer
R3 = pointer to next buffer location
R4 = number of bytes not transferred

This call performs a low-level access directly to an IDE disc drive. Bits 0-3 of R1 determine the operation, as shown in table 2. The option bits modify the basic behaviour. If bit 4 is set, then an alternative defect list 64 bytes after the start of the disc record pointed to by R1 will be used. This is necessary if the disc is not formatted. If bit 5 is elear, data will be transferred to or from the buffer pointed to by R3. If bit 5 is set, R3 instead points to a list of pairs of words. The first word of each pair is a buffer address, and the second is the length of that buffer. The pairs are used in sequence until the total amount of data specified by R4 has been transferred. R3 is then updated to point to the first unused pair, and that pair is updated to indicate the remaining buffer space. If bit 6 is set, Escape conditions are ignored during the operation; otherwise they will cause an error. Bit 7 is unused.

Value	*Meaning*	*Uses*	*Updates*
0	Verify	R2, R4	R2, R4
1	Read Sectors	R2, R3, R4	R2, R3, R4
2	Write Sectors	R2, R3, R4	R2, R3, R4
3	Read drive information	R2, R3	R3
4	Format track	R2, R3	R3
5	Seek	R2	
6	Restore	R2	

Table 2. IDEFS_DiscOp reason codes

All of the possible operations except for *Read drive information* and *Format track* are detailed in the PRM.

Read drive information will cause 512 bytes of data to be read into the address specified by R3 using the IDE Read Parameters command. The value in R2 specifies which drive to read from. You should consult the drive's data sheet for the interpretation of the data read.

Format track is used to low-level format the track identified by the value in R2. Register R3 should point to a list of byte pairs, one for each sector on the track. The first byte in the pair identifies the type of sector, and the second byte the sector number. Again, refer to the drive's data sheet for more details.

IDEFS_Drives (
SWI &41FC2)

On entry: -

On exit
RO = Default drive
R1 = 0
R2 = Number of IDE drives

This call returns the number of drives set up using *Configure IDEFSDiscs, and the default drive set using *Configure IDEFSDrive.

IDEFS_FreeSpace (
SWI &41FC3)

On entry:
RO = Pointer to disc specifier (null or carriage return terminated)

On exit
R0 = Total free space on disc
R1 = Size of largest object that can be created

This call returns, for a given disc, the total size of the disc, and the remaining space available for use. The disc specifier should be given as an ASCII string and can be a drive number or disc name, for example ":4" or ":IDEDisc4".

IDEFS_DescribeDisc (
SWI &41FC5)

On entry:
R0 = Pointer to disc specifier (null or carriage return terminated)
R1 = Pointer to 64 byte block

On exit -

This call reads the disc record (see above) for the specified drive into the block of memory pointer to by R1. You should refer to volume 3 of the *RISC OS Programmer's Reference Manual* for details of the format of the disc record.

IDEFS_TestReady

(SWI &41FFC)

On entry:

R0 = RISC OS Drive number

On exit

R0 = value indicating drive status
0 => Drive not present
1 => Drive not ready
2 => Drive present and ready

This call allows the instantaneous state of a drive to be read. Its main use is to see if a drive will respond to a command request within a reasonable period of time. The IDEFiler module uses this call to decide whether or not to attempt reading the disc name to put on the icon bar.

IDEFS_Drivelnfo

(SWI &41FFD)

On entry:

R0 = RISC OS Drive number
R1 = Pointer to buffer (0 for screen)
R2 = Length of buffer (if R1<>0)
R3 = Bit 0 =1 => list drive information Bit
1 = 1 => list format information Bits
2-31 reserved (set to zero)

On exit

RO Corrupted
R1 = Pointer to byte after terminating null
R2 = Remaining space in buffer
R3 Corrupted

This call is used to return, in textual form, information on the drive geometry and type, its RISC OS format, or both. The output can be into a buffer, or directly to the screen. See the documentation of *IDEFSDevices, which uses this call, for details of the information returned. Note, however, that the exact information and its layout should not be relied upon, as it may change in future versions of the software.

IDEFS_SysInfo

(SWI &41FFE)

On entry:

RO = 0 (SysInfo_ReadBusWidth)

On exit

RO = Width of interface to computer in bits
(16 for Arc, 8 for A3000 internal)

This call is principally designed to allow a formatter to decide on the correct interleave value when performing a low level format. Future version of IDEFS may support further reason codes.

IDEFS_Timer

(SWI &41FFF)

On entry:

RO = RISC OS Drive number
R1 = Timeout period (in seconds) -
1 for no change

On exit

RO = Current drive state
0 => drive is in shutdown mode -
1 => drive is spinning

This call sets the drive timeout period for drives supporting this feature. The instantaneous drive state (spinning or standby) is preserved by this call. An application can also use this call with R1=-1 to find the current state of the drive.

A. Error messages

The following is a list of the error messages that can be directly generated by the IDE filing system, IDEFS, and the desktop filer, IDEFiler. It is possible that other error messages will occur while using IDE discs, for example from *FileCore* or RISC OS itself.

For each error, its text, error number and meaning is given, along with possible causes where appropriate.

IDEFiler errors

`Use *Desktop to start the IDE Filer application (&8032A0)`
The IDEFiler can only be started automatically when the Desktop is started. Do not try to use the *Desktop_IDEFiler command.

`Window manager too old / not present (&8032A1)`
The installed Window Manager is not suitable.

`Icon not known (&8032A2)`
An unknown icon number was encountered. This is most probably caused by workspace corruption and can be cured by re-entering the Desktop.

IDEFS errors

`Unrecognised IDEFS SWI (&13100)`
A non-existent IDEFS SWI call was issued.

`Bad IDE drive number (&13101)`
An illegal drive number was given in one of the star commands that expects a drive number.

`Bad number of IDE discs (&13102)`
An invalid number of IDE discs was specified with the command *Configure IDEFSDiscs.

`Specified timeout is too long (&13103)`
A timeout greater than 150 seconds was given with the commands *Configure IDEFSTimer or *IDEFSTimer.

`IDE drive not ready (&13104)`
The selected IDE drive failed to become ready to accept commands within a period of ten seconds. One possible cause of this is that Ctrl-Break was pressed during a previous operation, and the drive is stuck in the middle of executing a command. This can be cured by pressing Ctrl-Reset. See the next chapter for other causes of this error.

`IDE drive not present (&13105)`

An attempt has been made to access a non-existent drive.

`Cannot mount IDE drive (&13106)`

A disc error occurred while attempting to mount a drive. Future versions of the software may report the actual disc error, as is done for operations other than mounting a drive (see below).

`Cannot read drive information (&13107)`

An error occurred while trying to read the 512-byte information block from the hard drive.

`Format error (&13108)`

An error occurred while performing a low-level format of a track.

`IDE command` `rejected (&13109)`

The drive rejected a command sent to it by IDEFS. This most probably indicates an incompatible drive model.

`Buffer overflow (&1310A)`

The buffer specified for SWI IDEFS_Drivelnfo overflowed. The data is truncated to the buffer length, and no terminating null is placed in the buffer.

Disc errors

A disc error is an error that the drive itself generates while trying to access a particular block of data on the disc. Disc errors are reported in the following form:

`Disc error xx at :d/aaaaaaaa`

where *xx* is a number indicating the cause of the error, *d* is the drive number, and *aaaaaaaa* is the disc address.

The possible values for *xx* (in hexadecimal) are:

01	The data address mark was not read eorrectly
02	Track zero not found
04	Command aborted
08	Sector not found
10	Uncorrectable data error
20	Attempt to read sector formatted as unusable

Errors &01 and &10 usually indicate a defect on the disc. This can be mapped out to prevent that sector being used, by using the command:

```
*Defect <drive> <disc address>
```

where disc address is as given in the error message. For example:

```
*Defect :4 1C24000
```

Error &02 indicates that while initialising, the drive failed to locate the outermost track. This is almost certainly due to a hardware failure of the drive itself, and should be referred to your dealer.

Errors &08 and &20 are usually a result of the drive not being formatted, or being formatted incorrectly. You should attempt to reformat the drive as explained in chapter 2 and try the operation again.

Error number &04 is likely to indicate that the drive detected a fault during an operation. This may be due to hardware failure of the drive.

Loader errors

The following two errors are generated by the code that loads the IDEFS and IDEFiler modules into the computer at reset time. They should never occur, but if they do, it may be due to a corrupted ROM, or another piece of software illegally accessing the IDE interface card.

```
Address too big for this podule (&803200)
This podule does not support writable devices (&803201)
```

B. Trouble-shooting

This section deals with the annoying situation when your system won't behave itself. In many cases, what at first appears to be a fault with some component in the system turns out to be a misunderstanding, or something trivial which can be cured immediately.

This chapter is split into two sections - firstly problems that appear immediately after installation, and secondly problems that oecur in a previously working system.

Problems that become apparent immediately after installation tend to be rather troublesome, as there are so many factors involved. The first step is to re-check the entire installation procedure, paying particular attention to cables that are reversed, commands missed out, and so on. If the problems are still present then the next step is to ascertain whether the eomputer is recognising the IDE interface card. The section entitled Scope in the Preface showed how to find out the version number of the IDE software. This should be tried, and if the interface card is not listed by the *Podules command then it is not being recognised by the computer. However, before condemning the card, remove it and check for bent pins.

If the interface card is recognised, but the computer cannot access the drive, then check first if you can hear the drive spinning. If not, then check the power supply lead to the drive. Note, however, that 2.5" drives do not start spinning until they are first accessed. If the error messages "IDE drive not ready" or "IDE drive not present" come up, then it may be that various links on the drives are set incorrectly. If the drive is a genuine RISC Developments unit then contact them, else refer to appendix C for more information. In a dual drive system, try disconnecting each drive in turn to see if this resolves the problem.

If you get a dise error when trying to access the drive, or the error "Bad free space map" or "Bad defect list" then it could mean that the drive is not formatted correctly. Formatting is covered in chapter 2. These errors can also occur in a dual drive system when both drives are set to be masters or slaves, rather than one of each (see appendix C).

Problems that arise in a previously working system are normally easier to resolve. Firstly, does the problem only arise with a certain application, or does it appear all the time? If only certain programs are affected then it is almost certainly an incompatibility problem with those programs. You should in this case read chapter 4 for further information.

If the IDE system fails to function at all, for example no icons appear on the icon bar, then it is possible that a piece of software has 'unplugged' one of the IDE modules. The worst culprits for this are games programs. If this appears to be the ease, you should follow the instructions in chapter 2 for re-initialising the software.

If the problem is that the PC Emulator doesn't recognise the hard drive, then have you created and initialised the DOS partition eorrectly? Chapter 3 eovers this in detail.

If the problem appears to be hardware related, then before suspecting the drive or interface card, check all the conneetions between the drive, or drives, and the interface card. One particular point to consider is that if both an external and internal drive are connected, the external drive must be powered up for either drive to work properly. It is also worth noting, that if the IDE software is loaded correctly (for example the icons appear), but the drive cannot be accessed, then the fault is more likely to lie with the drive than with the interface card.

The last resort

If after trying everything else you are convineed that there is a problem, then you should contact RISC Developments as described in the Preface before returning any equipment. This can save both unnecessary time and expense.

C. Connecting third-party drives

Every 'standard has its areas of total confusion, and IDE's main one concerns connecting two drives together. This section attempts to summarise some of the pitfalls to be avoided. Obviously, any drives supplied by RISC Developments will be configured to work correctly together.

Link settings

The IDE standard allows for two drives to be connected in parallel on the same cable. One of these is termed the *Master* and the other the *Slave*. In RISC OS these correspond to drives :4 and :5. When two drives are connected together various links on them must be set to configure one as the master and one as the slave. Unfortunately, this can end up being far from trivial.

Some IDE drives (such as the PrairieTek range) have a single link which is made for a master drive, and unmade for a slave drive. Other drives (for example Kalok models) have two links - one made for a master drive, the other for a slave. However, some drives have three modes of operation - Single drive, Dual drive master, and Dual drive slave. With these units you normally need to set one link to specify that the drive is a master, and another to indicate that a slave drive is present. The Seagate ST1102A is such a drive. All 'first' drives supplied by RISC Developments are configured to allow a second drive to be fitted with no further link changes to the first drive.

Having got that, the fun starts. The IDE bus contains a signal line to allow the slave drive to indicate its presence to the master. Unfortunately the same signal line can also be used to carry an active signal, allowing a remote drive-select light to be fitted. Most modern drives don't require the slave present signal to be connected when they are configured as a master, and most drives allow the pin's function to be switched between the active or slave present signals.

On all current RISC Developments IDE interface cards, the active signal is routed from the internal drive to the front panel LED via the interface, and the signal from the external drive is ignored. Future versions of the interface card may include a link to allow the two active/slave present signals to be connected together. This is necessary if the internal drive is of the type that does require a slave present signal.

Other problems

IDE was originally devised for internal PC hard drives, and this causes two problems. Firstly, the maximum cable length is very restricted. A safe maximum for unscreened ribbon cables is about eighteen inches. If you exceed this length then you are likely to run into weird problems such as random and intermittent data corruption when reading from the disc. If greater cable lengths are needed, or the system is used in an (electrically) noisy environment then you will need to use screened cable.

A more subtle problem concerns drives failing to start up at power up. All drives will function perfectly when run from the same power supply as the computer, as is the case with internal drives. However, when the drive has its own power supply it may be necessary to power the drive and computer up in the correct order (usually the drive first). If this isn't done, the drive can get locked into a state where only a power-down can return it to correct operation.

Unpowered drives

On a similar vein, problems occur if a drive attached to the IDE bus is not powered up properly. This will often result in the other drive failing to function correctly as well. In some cases, the unpowered drive will attempt to power itself from the IDE bus signals. This can lead to high levels of power dissipation in the interface card and should be avoided by ensuring that all attached drives are powered-up whenever the host computer is turned on.

Understanding drive translation

If connecting IDE drives together eauses total confusion, then understanding drive translation modes comes a close second. In this section we explain what drive translation is, and how to avoid it!

Every hard disc drive consists of a number of recording surfaces (or heads), each having a certain number of tracks, with eaeh track divided into a number of sectors. The combination, sectors per track, tracks per surface and total number of surfaces is termed the drive's *geometry* or *shape*. Normally, a hard dise drive must be formatted to correspond exactly to this shape.

Unfortunately, IBM PCs and compatibles, the main market for IDE drives, only support a limited number of drive shapes, none of which may correspond to the physical geometry of the drive. To get around this, IDE drives can be programmed to fake any other shape, provided that the emulated shape doesn't exceed the capacity of the drive. Some drives allow totally arbitrary translation, while others support only a limited range of shapes, and others enforce restrictions, such as stating that the number of sectors per track in the translation must be no more than that of the physical geometry.

The IDEFS software will always attempt to use IDE drives in their native physical geometry. It does this by finding out the physical geometry and using this to mount the drive, and then changing the geometry to match that specified in the disc record (see chapter 5). Normally, the formatter sets this to the physical geometry, though this can be overridden if really desired.

Some drives (for example NEC models) allow links to be set to force the drive to power-up into one of a set of translation modes. Such drives should always be set to power-up into their native mode.