Rotronics Wafadrive

2004 OCRed by Wilko Schröter

THE ROTRONICS WAFADRIVE

User manual

SPECTRUM VERSION

16/48K Sinclair Spectrum

model No: 2302

Written and prepared by
Graham Booth B. Eng

WAFADRIVE
is a trade mark of
ROTRONICS LTD.
Santosh House
Marlborough Trading Estate
West Wycombe Road
High Wycombe
BUCKS. HP11 2LB.
ENGLAND

ROTRONICS LTD
A division of Rovo Financial Corporation Ltd
Santosh House, Marlborough
Trading Estate, West Wycombe Road
High Wycombe
Bucks. HP11 2LB

CONTENTS

INTRODUCTION

The Wafadrive is a fast access storage system using special magnetic tape drives. It enables programs and data to be loaded into your Spectrum much more quickly and conveniently than is possible using cassettes. Furthermore, file handling is easily performed since the Wafadrive locates files automatically without the operator having to position the tape.

The Wafadrive unit also incorporates two interfaces which enable the Spectrum to drive most printers that are Centronics or RS232 compatible. This covers the vast majority of printers that are generally available. A full specification Word Processor is supplied with the Wafadrive so you can start using your new system straight away. The Word Processor allows documents and correspondence to be prepared and filed quickly and efficiently. This greatly increases the usefulness of your computer – particularly if you have a printer and add-on keyboard unit.

Readers of this manual should be familiar with the use of BASIC on the Spectrum. If you have purchased a Wafadrive at the same time as your Spectrum, first read the booklets supplied with the computer.

UNPACKING AND CONNECTING

Upon opening the Wafadrive carton you should find the following items:

1. The Wafadrive unit itself with integral connecting lead.
2. A blank (unformatted) wafer.
3. The Word Processor wafer.
4. This manual.

The power lead must be disconnected from the computer before connecting the Wafadrive. Failing to do this can result in damage being caused to the Wafadrive or your computer.

To connect the Wafadrive, simply push the connector into the Spectrum rear expansion port, ensuring that the locating key and slot are lined up first. Never force the connector into position.

When not required, the Wafadrive may be disconnected from the computer. However, it is recommended that this is done as infrequently as possible to minimise strain and wear of the connectors. The Wafadrive does not affect the normal operation of the Spectrum unless it has been initialised.

When disconnecting, always grip the connector assembly. Never remove the connector by pulling on the ribbon cable.

Any additional peripherals may now be connected via the slot at the rear of the unit. Ensure that the peripherals you intend using are compatible with the Wafadrive. The number of items that can be connected to the Spectrum at any one time is limited. It is wise to keep the loading on the computer to a minimum to avoid unreliable operation or possible damage.

The power may now be reconnected to the Spectrum whereupon the screen will display the normal Sinclair copyright message. The computer will behave exactly as normal since the Wafadrive system will not be effective until it is initialised (see chapter 2).

There are three indicator *LEDs* between the drive apertures. The centre on is the *power on* indicator, which will light up whenever power is applied to the computer. The remaining two, to the left and right, are the *drive active* indicators. These will illuminate whenever the drive motors are running.

Do not discard the Wafadrive packing materials. If your unit is ever required to be serviced or repaired, it should be returned in the original packaging. All parts supplied with the Wafadrive should be included, together with a clear description of the problem.

USING THE WAFADRIVE

INITIALISING THE SYSTEM

The many extra facilities that the Wafadrive adds to the Spectrum are provided by the *Wafadrive Operating System* (WOS), which is similar in many ways to the Disk Operating System (DOS) found in larger computers. The extra facilities are accessed by means of the *Extended BASIC* (EBASIC) commands made available by the Wafadrive Operating System. The Extended BASIC commands do not come into operation until the system has been initialised.

Once initialised, the Wafadrive Operating System reserves about 2K of the Spectrum's memory for its own use (system variables, read/write buffer and the two drive directories). The user has the option of retaining the computer's full RAM capacity if the Wafadrive is not required for use, without having to disconnect the unit; if the initialisation process is omitted, the computer will behave normally.

The system is initialised simply by entering:

NEW *

This is the only Extended BASIC command that the Spectrum will respond to until the system has been initialised. The Wafadrive title message will now be displayed and the system is ready for use. **NEW *** may be entered at any time. It does not affect any BASIC program currently in memory. However, once the system has been initialised, **NEW *** has no effect and will cause a syntax error. If at any time there is insufficient memory remaining to allow the Wafadrive system to operate, you will be unable to enter the **NEW *** command.

All the extra commands made available by the Wafadrive system are suffixed either with the asterisk (*) or hash (#) symbols.

USING NEW

The **NEW** command has exactly the same effect on BASIC programs as before. However, **NEW** will also reset the Wafadrive system, freeing the reserved RAM. The system must be re-initialised before it can be used again.

CLEARING BASIC PROGRAMS

The **NEW #** command clears out any BASIC program, without resetting the Wafadrive Operating System. Although the initialisation process is virtually instant, the use of **NEW #** instead of **NEW** means that the directories do not have to be re-fetched from the wafers (more of this in chapter 4).

Note that **NEW *** or **NEW #** cannot be incorporated into BASIC programs – they can only be executed directly from the keyboard.

CLEARING THE SCREEN

The Wafadrive adds an additional command which, although not directly related to the operation of the drives, is nevertheless very useful. It is:

CLS *

which performs the normal CLS function, but in addition also clears the screen attributes – resetting them to their initial values of **PAPER**=7, **BORDER**=7, **INK**=0 etc. Text will therefore revert to black on white.

THE WAFERS

Wafers are available in three sizes, having nominal capacities of 16K, 64K and 128K after formatting. The average access time (ie. the length of time you have to wait whilst the file you want is found on the wafer) increases in proportion to the capacity. The 16K wafer holds a reasonable amount of data and gives very rapid access. The larger capacity wafers give more cost effective storage but on average you will have to wait slightly longer before files are located on the wafer.

The user can choose the size of wafer to suit particular applications. The smaller wafers are recommended for program development where programs are generally small and fast access is important. Finished programs, back-up copies and data can be archived onto larger wafers, freeing the smaller ones for further use.

The tape inside the wafers is protected by an integral sliding cover which is automatically moved across to expose the tape when the wafer is inserted into the drive. Touching the tape can damage it – avoid doing so.

If It is possible to remove a wafer from a drive whilst the drive motor is running (signified by the *drive active* LED being lit). There is virtually no possibility of damage, but doing this habitually could have a detrimental effect on the life of the wafers. Removing wafers in this way should not be necessary unless the system "crashes" whilst the motor is running – only likely if you are experimenting with machine code. Usually the **BREAK** key can be used to stop the motor before a wafer is removed.

Care should be taken to ensure that wafers are withdrawn from the drives when the power is applied or removed – especially if either of the drives is active. The system has been designed to cope with this to an extent and you would be very unlucky to lose data (in the event of a power cut for instance).

However, as a general rule:

AVOID REMOVING WAFERS WHILST THE *DRIVE ACTIVE* LEDS ARE ILLUMINATED. ALWAVS REMOVE WAFERS BEFORE SWITCHING THE POWER TO THE COMPUTER ON OR OFF.

You will soon learn that your Wafadrive system is very reliable. The wafers have been designed to sustain 5000 passes and with careful use are capable of achieving double this figure. Despite this, it should be remembered that the wafers do have a finite life and that accidents can happen. It is strongly recommended that all important programs and data are backed-up onto other wafers – or possibly even cassettes.

Do not leave wafers in the drives when not in use. Store them in a dry place, where they will not be subject to extremes of temperature. Keep wafers away from the TV set, Spectrum power supply, cassette recorder and any other equipment likely to be the source of strong magnetic fields.

PREPARING AND NAMING WAFERS

Like disks (but unlike cassettes), blank wafers cannot be used straight away as supplied from the factory. They must first be *formatted* using the **FORMAT *** command. Commercially duplicated wafers are formatted during the duplication process and can be used straight away.

The most important thing to remember about formatting is that any data and programs stored on the wafer will be erased. Therefore, be absolutely sure that nothing of value is stored on wafers before you attempt to format them.

Once formatted, wafers will not normally require re-formatting – unless you want to "wipe" them clean for any reason.

The **FORMAT** command allows each wafer to be individually named. This name appears on the screen when the directory of the wafer is displayed, enabling wafers to be easily identified. It is very important that you give each of your wafers a unique name. If you do not, you run the risk of confusing not only yourself but the operating system also – with potentially disastrous results.

To format a wafer first insert it into either of the drives – say drive A (the left hand one). Ensure that it is pushed completely home. It is also advisable to ensure that there is no wafer in the other drive. Now type in:

FORMAT *"a:xyz.001"

where **"xyz.001"** represents your initials and a wafer number (a different number for each of your wafers). You don't have to name your wafers in this manner but it is advisable to be systematic, and this is a good method.

The **"a:"** section of the name string is the drive specifier. This can only be either **"a:"** or **"b:"** (upper or lower case), according to which drive is being used. Often the drive specifier can be omitted (see chapter 4). The maximum total length of the wafer name is ten characters – not including the drive specifier.

During formatting the tape length is measured to determine the capacity of the wafer. The individual sectors on the tape are then created and verified, and finally the *directory* is written. Whilst this process is taking place, the border will flash and status information is displayed on the screen. The wafer capacity is measured in sectors with each sector corresponding to 1K bytes of storage. One sector is reserved for the directory and so the total program storage available is 1K less than the capacity of the wafer.

If you enter the **FORMAT** command in error, you will usually have enough time to **BREAK** before the wafer starts to be erased.

Since up to four passes round the tape are required during formatting, it is a relatively lengthy process. This is not really important since it is not often performed more than once or twice during the life of a wafer.

The formatting process should now be completed and the wafer is ready for use. The system will display the wafer capacity and whether any errors occurred during formatting. No errors should occur. If they do, it could be a sign that the wafer is damaged or worn, or that the drive read/write head is in need of cleaning (try again on the other drive). Wafers which consistently give trouble during formatting should be discarded – or at the very least treated with suspicion.

PROTECTING AGAINST ACCIDENTAL ERASURE

Like cassettes and disks, wafers may be protected against accidental erasure or overwriting of stored data. This is done by removing the *write protect* tab which is on the left hand side of the wafer as it is inserted in the drive.

The original read/write status of the wafer may be restored by covering the recess with a piece of tape.

Wafers cannot be formatted if the write-protect tab has been removed.

Some pre-duplicated wafers will not have the write-protect tab removed. The programs on these wafers will usually have been recorded so that they cannot normally be erased or overwritten, although they will *not* be immune from formatting.

MAINTENANCE

The Wafadrive unit requires no more maintenance than does a conventional tape recorder. The main point to observe is that the read/write heads are kept clean so as to prevent loss of data.

The heads should be cleaned at intervals of 500 passes or so. It is performed simply by wiping with a clean, cotton-tipped stick, (of the type used to clean babies' important little places!), or by means of a special head cleaning wafer. Use no cleaning fluid.

THE DIRECTORY

A directory of files is maintained on each formatted wafer. Every time a program is added to or deleted from wafer, the directory is updated automatically. When loading programs, the system first examines the directory to determine whether or not the file is stored on the wafer, and if so, its location.

DISPLAYING THE DIRECTORY

To look at the directory, or *catalogue*, of the wafer you have just formatted, make sure it is inserted in drive A and enter:

CAT *"a:"

The drive will start up, the border will flash and after a short while the screen will display:

1. The drive (A in this case), and the wafer name.
2. A list of names of the files stored on the wafer.
3. The type of each file (program, data or machine code).
4. The size of each file (in Kilobytes).
5. The number of files on the wafer.
6. The size of the wafer (in Kilobytes).
7. The spare capacity left on the wafer (in Kilobytes).

Of course, there are no files on the wafer as yet because it has just been formatted.

Once a wafer has been accessed, the directory will be stored in the computer's memory. To demonstrate this, perform **CAT** again.

This time the border flashes very briefly and the directory re-appears almost immediately.

The directory of a wafer in drive B is examined in exactly the same way, just change the drive specifier to **"b:"** (upper or lower case).

The maximum number of files that can be held in the directory is 32, although 16K wafers are physically limited to 16 separate files. If a long directory causes the screen to overflow, the message **scroll?** will appear, the same as with a program listing.

Incidentally, if you press the **N** key when asked **scroll?**, you will get the message **BREAK - CONT repeats** as usual. If you then enter **CONTINUE**, expecting to see the rest of the directory, you will be disappointed. The computer will "lock-up" – although **BREAK** will restore control. This is not the fault of the Wafadrive system, the same happens with ordinary program listings. See chapter 2 in the BASIC Programming Manual.

The Wafadrive system maintains a *default drive*. This allows the drive specifier to be dropped from Wafadrive commands when referring to a specific drive. When the system is first initialised, drive A is made the default. We can therefore simply enter:

CAT *

to display the directory of drive A. This applies to all Wafadrive commands that require a drive specifier. We could have used it when we formatted the wafer in the last chapter, although it is a good discipline to be specific when formatting to reduce the possibility of costly mistakes.

The default drive feature simplifies the use of the Wafadrive and saves typing. You will find this feature most helpful if you use A as the general working drive, with B as the secondary drive.

SETTING THE DEFAULT DRIVE

Should you ever want to, you can alter the default drive yourself using a variation of the **CAT** command. For example, to change it to B, enter:

CAT #"b:"

If there is no wafer in the specified drive you will get an error message. Apart from setting the default. **CAT #** also loads in the directory of the wafer without displaying it on the screen, thus ensuring that the version of the directory in RAM is valid for the wafer in the drive. This could be useful if we wanted to read the directory from BASIC for example (see chapter 9).

Entering:

CAT #

just causes the directory of the current drive to be loaded, without displaying it.

Another way to set the default – without loading in the directory – is to **POKE** one of the extended system variables:

POKE 23767,0 sets the default to drive A
POKE 23767,1 sets the default to drive B

Similarly, if you **PEEK** this location, you can determine which drive is the current default. Be careful to adhere to the above values: careless **POKE**s can be the equivalent of putting a spanner in the works!

STORING AND RETRIEVING PROGRAMS

SAVING

The procedure for saving programs on Wafer is very similar to that when using cassettes. Type a short program into your Spectrum – if you want an example, try this:

```
10 REM colours
20 FOR n=1 TO 704
30 PRINT INK 8*RND;CHR$ 143;
40 NEXT n
```

To save this, assuming you have a wafer in drive A, enter:

```
SAVE *"a:colours"
```

The program is first saved, then the directory of the wafer is updated indicated by the border flashing.

Since A is usually the default drive, we could have abbreviated the previous statement to:

```
SAVE *"colours"
```

From now on, commands will include the drive specifier to illustrate the syntax, but remember that it can be omitted when referring to the default drive.

As with names of programs stored on cassette, Wafadrive file names can be up to ten characters long (this does not include the drive specifier character and colon when used). Unlike cassette program names however, Wafadrive file names are always converted to upper case so as to avoid ambiguity. File names can take the form of valid string variables, as well as literal strings.

Now that the program is stored, examine the directory again. You will see that the name now appears (in upper case), with the *file extension* .PRG to Indicate that it is a BASIC program. Also shown is the file size (rounded up to the nearest Kilobyte).

When saving a program using **SAVE ***, a **File exists** error message will be displayed if a file of the same name already exists on the wafer.

Note that there is no direct equivalent of the cassette **SAVE...DATA** command. This function is performed in a different manner on the Wafadrive (see chapter 7).

VERIFYING

The program can be verified by entering:

VERIFY *"a:colours"

Note that the file name may be specified in upper or lower case. The file extension is only something which appears in the directory. It is not part of the program name, and should not be included in Wafadrive commands.
The command:

VERIFY *"b:"

checks the program against the first program appearing in the directory of the specified drive (in this case drive B), and:

VERIFY *

checks it against the first program appearing in the directory of the default drive.

The entries in the wafer directory are in chronological order so the first entry in the directory is always the earliest file to be stored on the wafer.

LOADING

Clear the program from the computer using **NEW #**. Now enter:

LOAD *"a:colours"

The program can be listed to check that it has been fetched correctly. The same comments apply to loading as to verifying, for example:

LOAD *

causes the first program that appears in the directory of the wafer in the default drive to be loaded in.

ERASING

Assuming that the program *colours* is still in the computer; save it again, but this time give it the name *test*.

You should now have two versions of the program on the wafer. To erase the unwanted one you have just saved, enter:

ERASE *"a:test"

There is a *wildcard* facility that operates in conjunction with the **ERASE** command. This is best illustrated by an example. The statement:

ERASE *"a:te*"

causes all files begining with **TE** to be erased from the wafer in drive A. The asterisk takes the place of the missing letters. The statement:

ERASE *"a:*"

causes all files on the wafer to be erased. This feature can be very useful (it is a lot faster than re-formatting) but should be used with caution, for obvious reasons. To avoid confusion, do not use file names ending in asterisks. Note that the asterisk cannot be put at the begining of the file name:

ERASE *"*st"

Does not erase all the files ending in **ST**.

Commercially produced software is sometimes protected in a special manner. Protected files (indicated by a # suffix after the file extension in the directory) cannot be removed using the **ERASE** command. These files can only be removed by re-formatting the wafer.

MERGING

The procedure for merging BASIC programs is straightforward. Enter:

NEW #

then type in:

```
100 REM colours II
110 CLS
120 FOR n=1 TO 704
130 PRINT PAPER 8*RND; INK 8*RND;CHR$134;
140 NEXT n
```

followed by:

MERGE *"a:colours"

whereupon the two programs will be combined as normal.

You can merge an auto-run BASIC program. The resultant program will not auto-run when merging is completed.

SAVING USING AN EXISTING FILENAME

Normally, if you try to save a-program using a file name that already appears in the wafer directory, a **File exists** error message results and the program will not be saved. This is to prevent programs being accidentally overwritten.

Sometimes however, we may *want* any files of the same name to be overwritten. For instance, when developing a program on a small wafer, a cycle of two programs can be used to avoid having to repeatedly erase redundant programs, whilst retaining the last edited version as a back-up. In these circumstances we use a variation on the **SAVE** command – **SAVE #**.

At the moment, if you have been following the examples given, you will have a program in memory as a result of merging the two *colours* programs. We can cause this version to be saved instead of the original program thus:

SAVE #"a:colours"

The program will be saved, and the original version will be lost.

As a general rule, use **SAVE *** which of course protects against inadvertent overwriting of programs. Use **SAVE #** with care, and only when you are sure that there is no risk to anything of value stored on the wafer.

Protected files cannot be overwritten. Attempting to use **SAVE #** to overwrite a protected file will result in an error report.

AUTO-RUN PROGRAMS

BASIC programs may be made to auto-run on loading by using **LINE**, as with programs stored on tape. If you still have the *colours* program in the computer, enter:

SAVE *"a:autocolour" LINE 10

Examining the directory will show that the program *autocolour* appears with an asterisk after the file extension, indicating its auto-run status.

If you have followed the examples so far, you will have two programs on the wafer (in fact two versions of the same program). Load in *colours* then re-save it, with the same name, using **SAVE #**. This has the effect of making the *autocolour* program the first one to appear in the directory.

Now enter:

LOAD *

whereupon *autocolour* will load and auto-run.

If problems are experienced with loading a file, four attempts will be made before the system halts with a **Faulty wafer** error report. As usual, the **BREAK** key can be used to abort the loading process if required. During loading of protected programs however, the **BREAK** key is disabled. If a protected program fails to load, you should wait until the error report appears. If you are really impatient, you can pull out the wafer and start again, but this is not recommended.

SAVING AND LOADING MACHINE CODE

The procedure for saving and loading machine code on the Wafadrive is similar to that used with cassettes (see chapter 20 of the BASIC Programming Manual). For example, to save the screen after the program *colours* has finished, use:

SAVE *"a:display",16384,6912

Note that **CODE** is omitted. The Wafadrive operating system recognises that it is machine code by the presence of the *start* and *length* parameters, treating it accordingly. Machine code files appear in the directory with a .BYT suffix.

In this particular case, **SCREEN$** can also be used:

SAVE *"a:display"SCREEN$

A useful feature of the Wafadrive is the ability to save machine code such that it will auto-run when loaded back. This is done simply by specifying a third *run* parameter – the address from which execution will start when loading has been completed. The general form of the Wafadrive **SAVE** command is therefore:

SAVE *"d:filename",start,length,run

SAVE # can be used in the normal manner if it is required that any existing file of the same name should be overwritten.

The syntax used when loading machine code is simply:

LOAD *"d:filename",start

Note again that **CODE** is omitted. If it is required that the code be relocated upon loading, the *start* parameter should specify the address at which it should be located. If the code is to be loaded back to the same address it was originally saved from, *start* can be omitted.

To load back the previous display, simply use:

LOAD *"a:display"

VERIFY operates in a similar fashion to **LOAD**, and machine code files can be deleted using the **ERASE** command, as with any other file. Attempting use **MERGE** with machine code files will result in a **Wrong file type** error report.

COPYING PROGRAMS

It is important to make back-up copies of your files to guard against accidental erasure, overwriting or loss. The obvious way to do this would be to load each program individually, then re-save it onto the back-up wafer. The fact that the Wafadrive is a twin drive unit would obviously help here. Even so, if more than one or two files are involved, it can be a tedious process.

The Wafadrive has a facility which makes the copying of files much easier. It takes the form:

MOVE *"a:*_filename1_*" TO "b:*_filename2_*"

In this case, A is the *source* drive and B the *destination* drive. As usual the default drive specifier may be omitted. If a second file name is not given, the copy will take the same name as the original. Files can be copied onto the same wafer, but only if the copies are given names which do not already appear in the wafer's directory.

As with **ERASE**, **MOVE** also includes the very useful wildcard facility. For example:

MOVE *"a:te*" TO "b: "

copies all files begining with **TE** from drive A to drive B. Also:

MOVE *"a:*" TO "b:"

copies all files from A to B.

MOVE does not affect any program in memory as the files are copied a sector at a time (via a read/write buffer). If there is a large amount of information to be copied, the process may take a few minutes. Copying the entire contents of a 128K wafer is best left until you are ready for a coffee break!

MOVE works with program, machine code and data files but will not allow copies of protected files to be made.

MODIFYING EXISTING PROGRAMS

The vast majority of BASIC programs should function normally with the Wafadrive connected. Simply convert the **LOAD**, **SAVE** etc. commands into the appropriate forms. However, there may be problems with programs that make extensive use of **PEEK** and **POKE**. Without detailed knowledge of the program's operation, these may be tricky to convert. The fact that the Wafadrive reserves just over 2K of RAM could also prevent larger programs from working, particularly with the 16K version of the Spectrum.

Commercially produced software in cassette form is invariably written in machine code and protected against copying in any way. It is (intentionally) very difficult to break into such programs, so there is little chance of you successfully transferring them onto wafer. If you are really determined, writing to the Software house concerned may produce some information which could be of use.

STREAMS

STREAM HANDLING ON THE UNEXPANDED SPECTRUM

The Wafadrive operating system makes available commands which extend the *stream handling* capabilities of the Spectrum. Streams can be used to some extent on the basic machine however, without the Wafadrive being connected.

Streams are the routes along which data flows in the computer. Data is anything which can be put in a **DATA** statement (numbers, letters, symbols) plus variables, as distinct from programs, which *operate* upon data.

At each end of a stream is a channel, which is one of the computer system's input/output devices. In the basic Spectrum, the channels are:

channel **K** (or **k**) – the keyboard (input/output)
channel **S** (or **s**) – the screen (output)
channel **P** (or **p**) – the Sinclair printer (output)

The tape recorder interface is also an "I/O" device but does not have any channels associated with it since data is handled in a different manner on tape.

There are sixteen available streams on the Spectrum, numbered 0 to 15. Four of these are automatically linked to channels when the computer is powered up – for use by the Spectrum operating system. These are:

stream #0	input and output from keyboard
stream #1	input and output from keyboard
stream #2	output to screen
stream #3	output to printer

Streams 0 and 1 require a little further explanation. Although there is *physically* only one keyboard device (which can only input data), *logically* on the Spectrum there is another keyboard device – which is capable of outputting data from the computer. This is the bottom part of the screen, which as you will know, is distinct from the main part of the screen. That is why channel **K** is specified as an input/output channel above.

Little is said about stream handling in the BASIC Programming Manual supplied with the computer, but you may know already that some manipulation of streams is possible on the basic Spectrum.

Information is sent to the screen and received from the keyboard by means of the **PRINT** and **INPUT** statements respectively. In a similar fashion, **PRINT** and **INPUT** are also used to send and receive data to/from streams. The commands are simply suffixed with the # symbol and the *stream identifier* thus:

```
PRINT #2;"The ROTRONICS Wafadrive"
INPUT #0;a$
```

INPUT is of little use with the unexpanded Spectrum since there is only one input channel – the keyboard. We can have some fun with **PRINT** however, since all three channels are capable of output.

The **PRINT** and **LPRINT** commands can in fact be thought of as abbreviations of **PRINT #2** and **PRINT #3** (remember that stream 2 is assigned to the screen and stream 3 to the printer). If you have a Sinclair printer, plug it in and run this program:

```
10 LET a$="The ROTRONICS Wafadrive"
20 PRINT #1;a$
30 PRINT #2;a$,#3;a$
40 PAUSE 0
```

The message will appear at the top of the screen, at the bottom of the screen and also on the printer. The **PAUSE** statement prevents the bottom line being overwritten by the **OK** report. Line 30 illustrates the fact that than one stream can be included in a **PRINT** statement.

The program will also work in exactly the same way if **LPRINT** is substituted for **PRINT**. The same principles apply to other I/O statements such as **LIST**, **LLIST**, **INPUT** and **INKEY$**. The special printer commands **LPRINT** and **LLIST** could have been omitted from Spectrum BASIC, without causing any great inconvenience. In fact, most other versions of BASIC do not provide them.

Streams can be assigned to channels using the **OPEN #** command. Streams 4-15 cannot actually be used without first being opened, whereas 0-3 are opened automatically by the computer when switched on. **OPEN #** may be used to re-direct streams:

```
10 LET a$="The ROTRONICS Wafadrive"
20 OPEN #2,"p"
30 PRINT a$
```

Line 20 determines that subsequent output via stream 2 (which is normally assigned to channel **S** – the screen) should go to channel **P** – the printer. A single line similar to this at the begining of a program will cause all screen data to be sent to the printer. The opposite effect is also possible of course.

A stream can be closed, to "disconnect" it from the channel it has been assigned to. Closing any of the streams 0-3 has the effect of re-assigning them all to their original channels and is accomplished thus:

```
10 LET a$="The ROTRONICS Wafadrive"
20 OPEN #2,"p"
30 PRINT a$
40 CLOSE #2
50 PRINT a$
```

Additional streams can be opened for use if required. This can be demonstrated simply by altering the stream number in the program above from 2 to any number between 4 and 15 inclusive. Attempting to send (or receive) information to (or from) an unopened stream will result in an error report.

A bug in the Spectrum ROM can cause some problems when trying to close streams in the range 4-15. If an attempt is made to close one of these streams when it is not open, strange things can happen. If you are lucky, an irrelevant random error report is given. If you are not, the computer crashes and your program is lost. The advice here is to be careful!

STREAM HANDLING WITH THE WAFADRIVE

The Wafadrive unit adds two new channels to the Spectrum. These are:

channel **R** (or **r**) – the RS232 serial port (input/output)
channel **C** (or **c**) – the Centronics parallel port (output)

The use of the serial and parallel ports is covered in chapter 8.

In addition, the user can also set up data file channels – files stored on wafer which can be read to or written from. These are used to store *data* (as distinct from programs and machine code), and are discussed in the next chapter.

Three new commands are also made available:

OPEN #* acts in the same way as **OPEN #**, but is used when opening streams to the new Wafadrive channels described above. Note that **OPEN #** will not recognise these new channels. Also, if **K**, **S**, **P**, or their lower case counterparts, are used in an **OPEN #*** statement, the Wafadrive system will open data file channels with these names, instead of opening streams to the keyboard, screen or printer.

CLOSE #* is used to dose any stream and should be used instead of **CLOSE #** whenever the Wafadrive is in use. The problem with the original Spectrum **CLOSE #** command does not occur using **CLOSE #***.

CLEAR * has the effect of closing all currently opened streams and data files. Streams 0-3 are reset to their initial allocations.

STORING AND RETRIEVING DATA

CREATING A WAFADRIVE DATA FILE

We have seen how programs can be saved and loaded to and from the Wafadrive, and the way in which streams are used in the Spectrum to handle data. Now we shall look at the way in which special files are created on wafer to enable data to be both stored and retrieved.

A stream can be opened and assigned to a Wafadrive data file thus:

OPEN #*4,"a:testfile"

The rules governing data file names are the same as for program names, ie. ten characters maximum, forced to upper case. The only restriction is that files cannot take the names *R* or *C*. These names are reserved for the RS232 and Centronics channels.

After opening to a Wafadrive file, the directory of the wafer in the specified (or default) drive is checked to determine whether or not the file exists. If no file of the same name is found, a new one is opened and designated an output (write) file. If the file does exist, it is opened as an input (read) file. Attempting to open a non-data file (ie. one not carrying the .DAT file extension in the directory) will result in a **Wrong file type** error.

The action of opening a new file does not mean that it will immediately be created on the wafer. Initially, an area of memory (about 1K bytes) is reserved called a *buffer*. The contents of the buffer are only transferred to the wafer when it becomes full, or when the file is closed.

When an existing file is opened, a buffer is again reserved in memory. The first sector is read from the file and stored in the buffer ready for reading.

If, when a data file is opened, there is insufficient spare memory for the buffer to be created, an **Out of memory** error report will be obtained.

WRITING DATA TO A WAFADRIVE FILE

Data is written to the opened data file buffer using the **PRINT** command as with any other stream. For example:

PRINT #4;"HELLO"'a$'32

writes the three pieces of data to stream 4. The individual elements should be separated by carriage return (**ENTER**) markers. This can be achieved either by using the apostrophe as a separator, or by using separate **PRINT** statements. If semi-colons or commas are used instead, the data will be written, but the **INPUT** statement will not be able to read it back again properly.

Remember that the data will not be stored on the wafer or the directory updated until the file has been properly closed. Nor will the system allow a wafer to be accessed if any files opened to it have been left unclosed (attempting to do so results in a **Directory locked** report). Unclosed files are invariably unreadable.

A data file is closed simply by closing its associated stream. For example:

CLOSE #*4

closes stream 4and the corresponding data file. The data will then be stored on the wafer and the directory updated. The global command **CLEAR *** may also be used.

Program listings may also be written to data files using **LIST #** (or **LLIST #**).

READING DATA FROM A WAFADRIVE FILE

To read items of data from an opened file, use **INPUT #**. For example:

```
INPUT #4;a$;b$;n
```

fetches the first three items of data from the file previously opened to stream 4 and assigns them to variables. The variable types (ie. string or numeric) must correspond to the data stored.

As you will know, the **INPUT** statement is also able to output to the bottom part of the screen. Commas and apostrophes have an effect on output print format and therefore give rise to **Writing to a read file** error report. This is why semi-colons should be used to separate the variable names – they have a null effect on print format.

The **INPUT** statement expects to see **ENTER** characters after every item of data – just as it expects the **ENTER** key to be pressed after data has been typed in at the keyboard. This is why apostrophes are used as data separators when writing to a data file – they have the same effect as **ENTER**.

An alternative way of reading data from a file is to use **INKEY$**. This returns the next character in the file. For example:

```
 10 REM filetest
 20 OPEN #*4,"a:testfile"
 30 LETa$=""
 40 FOR n=32 TO 143
 50 LET a$=a$+CHR$ n
 60 NEXT n
 70 PRINT #4;a$
 80 CLOSE #*4
100 OPEN #*4,"a:testfile"
110 FOR n=1 TO 112
120 PRINT INKEY$#4;
130 NEXT n
140 CLOSE #*4
```

For simplicity, the above program "cheats" a little. Usually one would write the number of entries into the file, or terminate the data with a control character to enable the correct number of entries to be read back. If an attempt is made to read more entries than the file contains, the program will stop with an **End of file** error report. The user would then have to close the file before the wafer could be accessed again.

As an example of how data files are used, let us consider a group of names and exam results to be stored on wafer for reading back at a later date. This program illustrates the way in which this can be accomplished:

```
 10 REM writ-result
 20 DATA 5,"RESULTS 31.12.84","FRED",76,"BILL",19,"JOHN",82,"ALISON",84
    "COLIN",64
 30 OPEN #*4,"a:results/01" .
 40 READ c,h$:PRINT #4;c'h$
100 FOR t=1 TO c
110 READ n$,m
120 PRINT #4;n$'m
130 NEXT t
200 CLOSE #*4
210 STOP
```

A program to read the data back in could take the form:

```
500 REM read-result
510 OPEN #*4,"a:results/01"
520 INPUT #4;c;h$
530 DIM n$(c,15): DIM m(c)
540 FOR t=1 TO c
550 INPUT #4;n$(t);m(t)
560 NEXT t
570 CLOSE #*4
600 REM display
610 PRINT h$: PRINT
620 FOR t=1 TO c
630 PRINT n$(t),m(t)
640 NEXT t
```

COPYING FILES

Just as with programs, it is important to take back-up copies of files to guard against accidental erasure, overwriting or loss. The obvious way to do this would be to read in each file individually, then re-write it onto the back-up wafer. This is not a very convenient solution, so you will be glad to know that the **MOVE** command works with data files in exactly the same way as programs. When copying, no distinction is made between file types.

See chapter 5 if you need to refresh your memory about **MOVE**.

CENTRONICS AND RS232 INTERFACES

The Wafadrive is capable of driving most devices that are compatible with the "Centronics" parallel and RS232 serial interface standards. Although primarily intended to allow commercially available software to communicate with printers, modems and other computers, users can also utilise the facility for their own purposes – to reproduce text and produce program listings for example.

THE CENTRONICS INTERFACE

The Centronics interface is a unidirectional one – it can only transmit data from the *host* (the Wafadrive) to the peripheral (normally a printer). It is referred to as a parallel interface because data is sent a byte at a time along eight wires. Since most microprocessors work a byte at a time, it is relatively simple to implement at both the printer and computer end. The Wafadrive provides an "eleven wire" implementation of the Centronics standard, supplying the signals required by the vast majority of popular printers.

The parallel port may also be connected to a joystick via a simple adaptor. Most of the software available on wafer should be compatible with joysticks connected in this manner. For the electronics enthusiast, the parallel interface has the incidental benefit of acting as an eight-bit output port, enabling communication with other non-standard peripherals. See appendix 4 for further information.

THE RS232 INTERFACE

The RS232 interface is bidirectional, enabling data to be both transmitted and received. It is referred to as a serial interface because data is transmitted a bit at a time along a single wire. The computer and peripheral must perform the conversion between serial and parallel at either end, deal with the two-way data flow and provide relatively large signals. It is therefore somewhat more difficult (and hence usually more expensive) to implement than the Centronics interface.

Unfortunately, the RS232 specification is a loose one. There are many variations and options to confuse the novice and expert alike. If you are considering buying a printer, one offering a Centronics interface will usually be the most straightforward to connect and get operational.

The RS232 interface on the Wafadrive takes the following format:

Lines:	RXD (Received Data – input) TXD (Transmitted Data – output) RTS (Ready to Send – input) CTS (Clear to Send – output)
Parity check:	None
Number of bits:	Eight
Stop bits:	One
Baud rate:	110, 150, 300, 600, 1200, 2400, 4800, 9600 or 19200 (software selectable)
Default rate:	1200 Baud
Baud rate error:	3% max. (at 19200 Baud)
Output swing:	10V p-p nominal

The peripheral should be adjusted to conform with this. As far as the Baud rate is concerned, choose the highest one that your peripheral will support. Avoid using the higher Baud rates with long cables however.

CONNECTING PERIPHERALS

At the rear of the Wafadrive unit are three sets of connections. The largest of these is the familiar Spectrum expansion bus. The smallest is the RS232 serial port, and the centre one is the Centronics port.

To connect peripherals to the Wafadrive, you will require a lead terminated at each end with the appropriate connector. If you are handy with a soldering iron, it is possible to make your own by referring to the peripheral manual and the connection details below. Generally however, it is simpler, safer and cheaper to buy a ready made lead. A suitable one is available through your ROTRONICS dealer.

The edge connectors are depicted below, viewed from the rear of the unit:

CENTRONICS

```
A1  A2  A3  A4  A5  A6  A7  A8  A9 A10 A11 A12 A13
 =   =   =   =   =   =   =   =   =   =   =   =   =
=============================================
 =   =   =   =   =   =   =   =   =   =   =   =   =
B1  B2  B3  B4  B5  B6  B7  B8  B9 B10 B11 B12 B13
```

RS232

```
A1 A2 A3 A4 A5
 =  =  =  =  =
==============
 =  =  =  =  =
B1 B2 B3 B4 B5
```

These are usually connected as follows:

CENTRONICS INTERFACE

signal	edge connector	peripheral connector
Ground	A1, A2, A13, B1-13	16, 17, 19-30
Data strobe	A4	1
Busy	A3	11
Data 0	A12	2
Data 1	A11	3
Data 2	A10	4
Data 3	A9	5
Data 4	A8	6
Data 5	A7	7
Data 6	A6	8
Data 7	A5	9

RS232 INTERFACE

signal	edge connector	peripheral connector
Ground	A1, B1-5	1, 7
RXD	A2	2
TXD	A5	3
RTS	A4	4
CTS	A3	5

Further information will be found in your peripheral handbook.

If making your own lead, it is wise to leave the B side contacts unconnected so as to prevent possible damage by wrong insertion of the connector.

Ensure that the power is disconnected from the Spectrum before plugging anything into either of the ports. Failing to do this is unlikely to cause damage, but could result in the computer "crashing"

SENDING INFORMATION VIA THE INTERFACES

Assuming that you have a printer connected, to the Wafadrive by means of a suitable lead, it should now be possible to print out programs and data.

Spectrum programs are *tokenised* in order to maximise the use of memory. This means that all the commands **LIST**, **PRINT**, **NEW** etc. are represented by a single code number – as described in appendix A of the Spectrum BASIC Programming Manual. When using a printer to produce program listings, the tokens must be expanded into their full form. The commands **LIST** and **PRINT** perform this function and can be used directly.

Associated with the RS232 and Centronics ports are two new channels, **R** and **C** respectively. Before using the RS232 interface however, we must first set the Baud rate using the **FORMAT** command. Here are two programs, which produce listings on printers via the appropriate ports:

```
10 REM CENTRONICS
30 OPEN #*4,"c"
40 PRINT #4;"The ROTRONICS Wafadrive'"
50 LIST #4
60 CLOSE #*4
10 REM RS232
20 FORMAT *"r";2400
30 OPEN #*4,"r"
40 PRINT #4;"The ROTRONICS Wafadrive'"
50 LIST #4
60 CLOSE #*4
```

These programs use streams in the normal manner so should be quite straightforward. Line 20 above sets the RS232 port Baud rate to 2400. Substitute your own figure here as required. As always, be careful with the syntax of the **FORMAT** command. When first switched on, the Baud rate defaults to 1200. If your peripheral is set to operate at 1200 Baud, there is no need to use the **FORMAT** statement.

Assigning stream 3 to the output ports causes **LPRINT** and **LLIST** to work with your printer, instead of the normal Sinclair printer.

Peripherals will not interpret the graphics characters correctly when using **PRINT**, **LIST** etc. If the item to be printed contains graphics characters, these will either be ignored or will cause strange effects. One way round this is to refer to graphics characters using **CHR$**. To print out graphics characters correctly on a printer that is capable of reproducing graphics, you will have to write or buy special software suitable for your particular printer.

To send control characters to peripherals, use the **CHR$** function. For example, if your printer requires the sequence of hexadecimal codes 1B 5B 31 6D to switch it into extended typeface mode, use:

PRINT #4;CHR$ 27 + CHR$ 91 + CHR$ 49 + CHR$ 109

Hexadecimal conversions are given in appendix A of the Spectrum BASIC Programming Manual. The appropriate codes for your printer will be found in the manual supplied with it.

The directory may be printed out using a simple routine like this:

```
10 OPEN #*4,"c"
20 PRINT #4;
30 CAT *"a: "
40 CLOSE #*4
```

RECEIVING INFORMATION VIA THE RS232 INTERFACE

Information can also be received via the RS232 interface from peripherals such as terminals, modems, or other computers. BASIC is too slow and restricted for efficient communication, but it is possible to obtain some useful results.

INPUT and **INKEY$** will both work, but the latter is the most commonly used because it does not require carriage return (**ENTER**) characters to be received before the data is accepted. If you have a terminal or other RS232 device which is capable of transmitting, this program will display anything sent:

```
10 FORMAT *"r";2400 (or to suit)
20 OPEN #*4,"r"
30 PRINT INKEY$ #4;
40 GO TO 30
```

FURTHER INFORMATION

ROTRONICS, as a service to Wafadrive users, will make available an information sheet giving details of RS232 connections found to work with various popular printers. This will be free upon request, and can be obtained by sending an s. a. e. with a first class stamp to the address given at the beginning of this manual.

ROTRONICS, at their discretion, will supply a free blank wafer cartridge to anyone supplying original information (connections, listings etc.) which may be of assistance to other Wafadrive owners in relation to the operation of the RS232 interface.

HOW THE WAFADRIVE WORKS

This chapter is aimed at those people who wish to explore the workings and facilities of the Wafadrive further. It is assumed that the reader is familiar with the basic operation of the Spectrum computer system.

SYSTEM OPERATION

The drives run at two speeds depending upon whether they are searching for a file or reading/writing from/to it. The system works out how long to wind the tape at fast forward before slowing down ready to access a file or the directory. It can do this because it maintains a *pointer* in the extended systems variables which indicates the last sector accessed. The wafer directory contains all other information required to perform the calculation. Incidentally, if you are clever with machine code it is possible to implement a *predictive access* technique whereby the tape is positioned such that the next required block can be accessed with virtually zero delay. If done under interrupt control, this can be performed transparently to the operation of any program being executed.

In the wafers, the tape loop is completed by a conductive splice – a section of metallic leader tape which is detected by the Wafadrive hardware and acts as an *index* marker. The directory is stored in the first good sector after the index. When a wafer is first accessed, the tape is fast wound until the index is reached, then slowed ready to read the directory.

The directory sector has the facility to store a *bootstrap* routine. If implemented, this routine will be executed as soon as the wafer directory is accessed, overriding control of the system. This is done principally to allow software to be made secure and for obvious reasons, details of this aspect of Wafadrive operation are not released to the general public.

Files can be tagged as *protected* in the directory. Protected files cannot be copied or merged, nor can they be erased without re-formatting the wafer. The **BREAK** key cannot be used to abort loading of a protected program. Again this is an aid to software security and details are not made generally available.

OPERATION OF THE EXTENDED BASIC COMMANDS

The Extended BASIC (EBASIC) commands are provided by the Wafadrive Operating System (WOS), which resides in 8K of ROM. This ROM is located at address 0000 of the Spectrum's Z80 memory map, sharing this space with the lower 8K of the existing Spectrum 16K ROM.

Obviously, it is not possible for the two ROMs to co-reside in the same address space without disrupting each other's control over the CPU. Whenever the Wafadrive ROM comes into operation therefore, the Spectrum ROM must be "paged out", and vice versa. This is performed by hardware in the Wafadrive which monitors the address bus. Whenever the ROM routine at location 0008(H) is called, the Spectrum ROM is disabled, the Wafadrive ROM is paged in and control passes to the Wafadrive Operating System. The above address is the location of the Spectrum's error handling routine which is called whenever a BASIC syntax error occurs – either during program entry or execution. So, whenever a syntax error occur when the Wafadrive is connected, the Wafadrive ROM is paged in to deal with it. A check is made to determine whether or not the "error" is caused by an EBASIC command being encountered and the system is made to behave accordingly. "Genuine" syntax errors are dealt with in the normal manner. EBASIC commands are either accepted into the program or executed (depending upon whether the error occurs during program entry or execution). The Spectrum ROM is then paged back in.

The hardware which controls the ROM paging is controlled by Z80 IN instruction. A little thought reveals that when control is passed pack from the Wafadrive to the Spectrum ROM the appropriate instructions to perform this cannot be incorporated into the Wafadrive ROM. Instead, a short routine in RAM is set up in the extended system variables area to perform this function.

The syntax checking routine is accessed via a vector in RAM (see appendix 3). This can be altered so that special custom commands can be written in machine code and added to Spectrum BASIC.

It may be of interest to know how the main commands which access the drives operate:

FORMAT
There are four stages to the process of formatting a wafer. Firstly, the tape is fast wound to reach the index. Secondly, the tape is fully wound at normal speed and the length (number of sectors) is determined. Thirdly, the individual sector headers and "dummy" data are written to the tape. Finally, the sector information is verified and the directory created. Any sectors which fail the verification process are indicated in the directory as being unusable.

CAT
If there is already a directory in RAM for the specified drive, the system first makes a check to determine whether it is valid for the wafer currently in the drive. This is done by reading the next sector on the tape. Each sector header contains the wafer name given by the user during formatting, the wafer name is checked against that appearing in the RAM directory, if the two match then the system knows that the RAM directory is valid and can be displayed on screen. If the two do not match then the directory must first be loaded from the wafer. This method saves a great deal of time; if the wafer has previously been accessed, the directory will appear on screen in about a second.

SAVE
When saving, the directory is first validated. Information in the directory is then used to determine whether or not a file of the same name already exists, whether there is enough room on the wafer to store the program, and free sectors available for storage. The program is written to the tape sector-by-sector; then the directory is updated.

LOAD
Again, the directory is first validated. After checking that the program exists, and the sectors it occupies, the program is loaded in sector-by-sector. The directory does not need to be modified afterwards, so loading is usually quicker than saving.

ERASE
The directory is first validated. The entry relating to the specified program(s) is then deleted from the wafer directory.

VERIFY
This is very much like **LOAD**, except that the specified program on the wafer is checked against that stored in memory.

MERGE
Again, very similar to LOAD except that the program on the wafer is combined with that in memory.

READING THE DIRECTORY FROM BASIC

Two directories are maintained in RAM by the Wafadrive Operating System, one for each drive. These hold all important information about the wafers the drives, and the files stored on them.

The directories each occupy 582 bytes and are located between the read/write buffer and the channel information area, starting at address 23734. Reference should be made to appendix 4 for more complete information.

A subroutine to read the RAM directories from BASIC could take the form

```
9600  REM Directory read routine
9610  DIM w(4): DIM s(32): DIM d$(32,14)
9620  GO SUB 9900
9630  LET w$=a$: LET a=a+13
9640  FOR n=1 TO 4: LET w(n)=PEEK a: LET a=a+1: NEXT n
9650  LET a=a+21
9660  FOR e=1TO 32
9670  GO SUB 9800
9700  IF CODE a$=0 THEN RETURN
9710  LET d$(e)=a$
9720  NEXT e
9730  RETURN
9800  REM Build entry
9810  GO SUB 9900
9820  LET c=PEEK a-4*INT ((PEEK a)/4)
9830  IF c=0 THEN LET a$=a$+".PRG"
9840  IF c=1 THEN LET a$=a$+" .BYT"
9850  IF c=2 THEN LET a$=a$+".DAT"
9860  LET a=a+1: LET z=PEEK a: LET s(e)=z
9870  FOR n=1 TO z+1: LET a=a+1: NEXT n
9880  RETURN
9900  REM Build name
9910  LET a$=""
9920  FOR n=1 TO 10
9930  LET a$=a$+CHR$(PEEK a): LET a=a+1
9940  NEXT n
9950  RETURN
```

Enter this with variable a=24862 for directory A and a=25444 for directory B. To ensure that the directories are valid for the wafers in the drives, use **CAT #** to "boot up" the wafers first. However, note that this command also alters the default drive. On exit, the string variable *a$* contains the name of the wafer (WAFER_NAME). The array *w* contains information about the wafer as follows:

***w(1)* = The overall capacity of the wafer (WAFERSIZE)**
***w(2)* = Number of good sectors on the wafer (GOOD_SECT)**
***w(3)* = Number of sectors left available (FREE_SECT)**
***w(4)* = Number of files currently stored (FILES)**

The array 5 contains the sizes of the individual files and the array *d$*, their names.

If the following lines are added, the action of the above subroutine can be demonstrated:

```
 10 REM Display directory
 20 LET a=24862
 30 INPUT "DRIVE A OR B?", LINE a$
 40 IF a$="B" OR a$="b" THEN LET a=25444
 50 GO SUB 9600
100 PRINT "WAFER NAME:",w$
110 PRINT "'SIZE:";w(2);"K SPARE: ";w(3);"'K
120 PRINT "FILES:";w(4): PRINT
130 FOR e=1 TO w(4)
140 PRINT d$(e);": ";s(e);"K"
150 NEXT e
160 STOP
```

The subroutine could also form the basis of a program to search for a particular file name on a wafer.

ERROR REPORTS

With the Wafadrive connected to your Spectrum, several new reports are added as listed below. Some of the reports will not normally be encountered when operating in BASIC, although special programs may use them. The error code for each report is given in brackets, although this is not displayed.

See also the Spectrum BASIC Programming Manual, appendix B.

Directory full (22)
The maximum number of files that can be held on the wafer (32) has been reached.

Directory locked (24)
An attempt has been made to access the wafer whilst a data file remains opened to it. Close all files before saving, erasing etc.

Drive write-protected (8)
An attempt has been made to save, erase or copy to a wafer which has had the write protect tab removed.

Faulty Wafer (15)
Obtained after four attempts at loading a file have failed, or if verification fails on a significant number of sectors during formatting.

File exists (18)
An attempt has been made to save a program using a name which already appears in the wafer directory. This error is suppressed if **SAVE #** is used instead of **SAVE ***.

File opened for write (16)
Not normally encountered.

File name missing (4)
Not normally encountered.

File not found (11)
The specified file does not appear in the wafer directory.

File read only (23)
An attempt has been made to copy a protected file.

File write-protected (21)
You have tried to erase or overwrite a protected file.

Invalid Baud rate (17)
The baud rate specified in the RS232 **FORMAT** statement is not one of the nine available.

Invalid drive (3)
A drive has not been specified as either **"A:"**, **"a:"**, **"B:"** or **"b:"**. Also occurs if trying to copy multiple files onto the same wafer.

Invalid hook code (19)
Not normally encountered.

Invalid name (2)
The file name specified is invalid – more than ten characters for example. Also occurs if attempting a copy a single file onto the same wafer without renaming.

Invalid stream number (1)
The stream number specified is not within the allowed range 0 to 15.

Merging error (12)
The **MERGE** command has not been executed successfully.

Nonsense in BASIC (0)
The contents of a string do not form a valid (Extended) BASIC expression.

Out of range (20)
The argument of a function is outside the permitted range.

Program finished (255)
Occurs if a line number is specified in a **RUN**, **GO TO** or **GO SUB** statement which exceeds the highest line number in the program. Also obtained if **RUN** is entered when no program is in memory.

Reading from a write file (7)
INPUT or **INKEY$** has been used in conjunction with a file opened for writing only.

Stream already opened (5)
The stream specified in an **OPEN** command is already opened. Streams must be closed before they can be re-assigned to other channels.

Verification failed (13)
The program in memory does not match that specified in the **VERIFY** command.

Wafer full (9)
There is insufficient space remaining on the wafer to allow the specified file to be stored.

Wafer not inserted (10)
There is no wafer present in the specified drive.

Writing to a read file (6)
PRINT, **LIST**, or **INPUT** with invalid separator has been used in conjunction with a file opened for reading only.

Wrong file type (14)
Occurs when an attempt is made to load a data file, merge a data or machine code file, open to a program file etc.

COMMAND SUMMARY

The commands relevant to Wafadrive operation are summarised below, in alphabetical order.

BREAK
Direct command, may be used to abort execution of BASIC at any time, except when loading a protected program. During some Wafadrive operations it may be necessary to hold the **BREAK** key down for a second or two until the operation is halted.

CAT *"*d:"
Display the directory of the water in the specified drive.

CAT #"*d:"
Loads in the directory of the specified drive, without displaying it on screen. Also sets the default drive to that specified.

CLEAR *
Closes all previously opened streams, setting streams 0-3 to their initial allocations. Also closes any opened data file channels.

CLOSE #*stream*
Closes stream previously opened using **OPEN #**. If any of the streams 0-3 are specified, these are all set to their default values. The use of **CLOSE #*** is preferred when the Wafadrive is in operation.

CLOSE #stream***
Closes stream previously opened using **OPEN #** or **OPEN #***. Closing streams assigned to data file output channels causes the file to be stored on wafer and the directory to be updated.

CLS *
Clears the screen but also resets the attributes to **PAPER**=7 (white), **BORDER**=7, **INK**=0 (black).

ERASE *"*d:filename"
Removes the specified file entry from the wafer, treeing the reclaimed sectors for subsequent use. "Wildcard" facility operates with **ERASE**.

FORMAT *"R";*baudrate*
Sets the RS232 transmit/receive data rate. The default value is 1200 baud.

FORMAT *"*d:wafername"
Used to prepare blank wafers ready for use. Tape is completely erased of files, and wafer is named.

INKEY$ #*stream;variables*
Reads next single character from previously opened input channel, via assigned stream.

INPUT #*stream;variables*
Reads data from previously opened input channel, via assigned stream.

LIST ***#stream,line***
Lists program, starting at optional line number to specified stream.

LOAD ****"d:filename"***
Loads specified program, irrespective of whether it is BASIC or machine code. **LOAD *** alone loads first program in directory of wafer in default drive.

MERGE ****"d:filename"***
Merges specified BASIC program with that currently stored in memory.

MOVE ****"d:filename1"*** **TO** ***"d:filename2"***
Makes a copy of file specified in *filename1*, optionally giving it the name specified in *filename2*. "Wildcard" facility also operates with **MOVE**, allowing entire wafer contents to be copied if required.

NEW
Clears BASIC program and also resets Wafadrive Operating System, freeing reserved RAM.

NEW *
Initialises the Wafadrive Operating System, reserving around 2K bytes for the directories and extended system variables.

NEW #
Clears the BASIC program area without resetting the Wafadrive Operating System.

OPEN ***#stream,"channel"***
Opens the specified stream and assigns it to the specified channel.

OPEN ***#*stream,"port"***
Opens the specified stream and assigns it to the specified port channel.

OPEN ***#*stream,"d:filename"***
Open the specified stream and assign it to a data file channel. If the file is a new one, the channel is opened as an output channel. ready for writing. If the file already exists, the channel is opened as an input channel, ready for reading.

PRINT ***#stream;"string",data,variables***
Sends string or numeric data to previously opened output stream. If sending to a data file channel, this must be configured as an output channel.

SAVE ****"d:filename"*** **LINE** ***run***
Saves BASIC program in memory to drive specified. Error report if file already exists. Optional **LINE** suffix gives auto-run facility upon loading.

SAVE ****"d:filename",start,length,run***
Saves machine code bytes from memory. Error report if file already exists. Option to auto-run code upon loading when required.

SAVE #
If used instead of **SAVE *** above, does not produce error report if file of same name already exists in wafer directory. Any such file is automatically over-written.

VERIFY *_"d:filename"_
Verifies BASIC or machine code program on wafer against that stored in memory.

The following abbreviations are used above:

baudrate	RS232 transmit/receive data rate. Can take the value 110, 150, 300, 600, 1200, 2400, 4800, 9600 or 19200.
"channel"	Channel specifier string. Valid ones are **"P"**, **"K"**, **"S"**, and their lower case counterparts.
data	Collection of numbers and/or strings.
"d:"	Drive specifier string. Takes the form **"A:"**, **"B:"**, or their lower case counterparts. If none stated then default assumed.
"filename"	Name of program or data file. Up to 10 characters, system forces uppercase.
length	Number of machine code bytes to be saved or loaded.
line	line number.
"port"	String specifying RS232 or Centronics port channels. Valid ones are **"R"**, **"C"**, and their lower case counterparts.
run	Line number or address from which program execution is to auto-run when loaded.
start	Address in memory from which saving or loading of machine code block is to commence.
stream	Stream number lying between 0 and 15.
"string"	Any chosen string of characters.
variables	List of variable names representing numeric or string data.
"wafername"	Unique name given to wafers during formatting. May be up to 10 characters, upper or lower case.

EXTENDED SYSTEMS VARIABLES

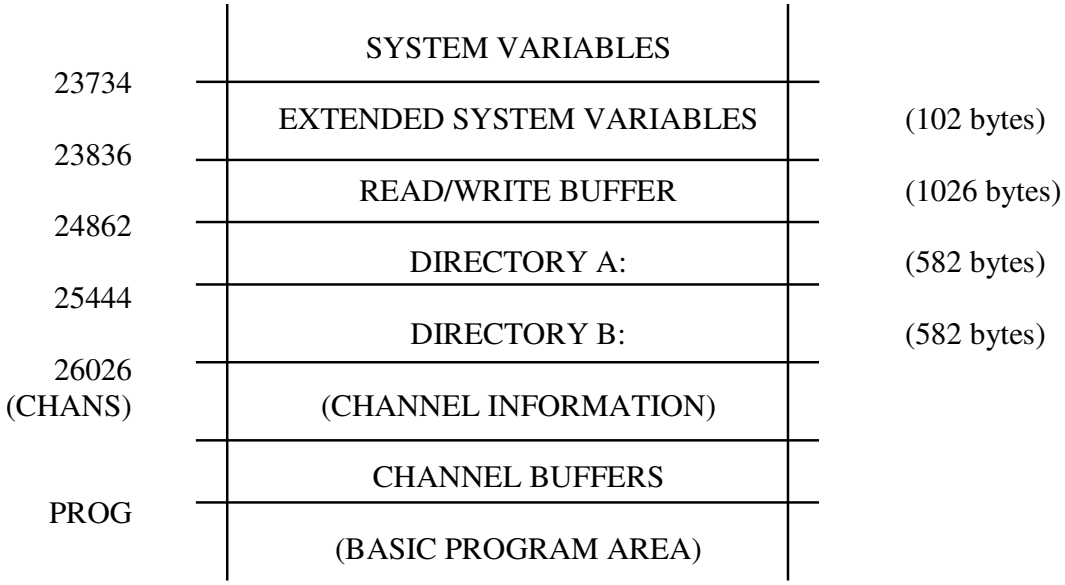
Several new systems variables are added by the Wafadrive. These occupy 102 bytes above the original Spectrum system variables, starting at address 23734. Some of these can be usefully *peeked* and *poked*, although it is very easy to "crash" the system if this is done inadvisedly.

Bytes	Address	Name	Comments
1	23734	INITFG	Initialisation flag.
2	23735	SYNVCT	Vector for syntax checking routines.
2	23737	ERRVCT	Vector for error handling.
2	23739	STRVCT	Vector for string printing.
11	23741	ROMPAG	Machine code paging routine.
2	23752	BAUDRT	Used for calculating Baud rate.
1	23754	CHFLGA	"Channel opened" flag for drive A.
1	23755	CHFLGB	"Channel opened" flag for drive B.
1	23756	TMPFLG	Temporary channel flags.
1	23757	LOADNG	Program load flag.
1	23758	RTYCNT	Retry count for sector read.
1	23759	TMPDT0	Temporary data storage byte.
1	23760	TMPDT1	Temporary data storage byte.
2	23761	XCHADD	Initial text pointer.
2	23763	CHADHL	Sector allocation address.
1	23765	ERRFLG	Error flag – used during formatting.
1	23766	REPORT	Current control port status.
1	23767	LOGDRV	Current logged drive (0=A, 1= B).
1	23768	DRIVNR	Drive number.
1	23769	SECTRA	Sector under head – drive A.
1	23770	SECTRB	Sector under head – drive B.
1	23771	SLFAST	Fast forward data for I/O control.
1	23772	SLREAD	Read control port.
1	23773	SLWRIT	Write control lines for current drive.
1	23774	SLINDX	Index mask.
2	23775	DIRPNT	Points to current RAM directory.
1	23777	RDYDYA	Ready flag – directory A.
1	23778	RDYDYB	Ready flag – directory B.
2	23779	TEMPW0	Temporary word storage.
2	23781	TEMPW1	Temporary word storage (for **SAVE/LOAD**).
2	23783	TEMPW2	Temporary word storage (**RUN** address etc.).
12	23785	SCRAT0	Used when scanning names.
16	23797	SCRAT1	Used when reading sector header.
12	23813	SCRATM	Destination file name in **MOVE** statement.
11	23825	BUFFER	Supplementary read/write butter area.

MEMORY MAP AND PORT ADDRESSES

THE MEMORY MAP

When the Wafadrive is initialised, the Spectrum memory map is altered as follows:



Note that the BASIC program will be moved as channels are opened and closed. Any machine code programs stored in **REM** statements should take account of this.

THE DIRECTORIES

The directories each occupy 582 bytes and are located between the read/write and channel buffers, starting at address 24862.

Each directory appears in memory as follows:

ITEM	BYTES	COMMENTS
WAFERNAME	10	Name given to wafer when formatted
ID_BYTES	2	Not normally used (set to spaces)
DIR_ATTR	11	Not normally used (set to spaces)
WAFERSIZE	1	Capacity of wafer (sectors)
GOOD_SECT	1	Number of useable sectors
FREE_SECT	1	Number of sectors yet unused
FILES	1	Number of files on wafer
DIR_SECT	1	Directory sector number
SAM	20	Sector allocation (bit map)
FILENAME_1	10	Name of first file
FILETYPE_1	1	Indicates PRG, DAT or BYT
FILESIZE_1	1	Size of file (sectors)
SECTORS_1	Varies*	Sector numbers occupied by file
		.
		(files 2 to 31)
		.
FILENAME_32	10	Name of last file
FILETYPE_32	1	Indicates PRG, DAT or BYT
FILESIZE_32	1	Size of file (sectors)
SECTORS_32	Varies*	Sector numbers occupied by file

* One byte for each sector occupied by the file. Total number of bytes therefore given by contents of FILESIZE.

A BASIC program which reads the directory is given in chapter 9.

THE READ/WRITE BUFFERS

There are actually four buffers created by the Wafadrive system in RAM. The main read/write buffer is approximately 1K bytes in size and is permanently located between the system variables and directory areas. It is used by the system to hold sections of programs as they are read from, or written to, the wafers.

In addition, there are three channel read/write buffers. The largest of these channel buffer, which again is about 1K bytes in size. This is created in memory when a data file channel is opened for input or output using the **OPEN #*** command. There are also buffers created when streams are opened to the RS232 and Centronics ports. These however, consume just 11 bytes each.

CONTROL PORTS

There are several ports which the Wafadrive uses to control internal functions and access the communications interfaces. Probably the most useful of these are the motor control and Centronics ports. The RS232 port is not likely to be of very much use in this context and is not described here.

It should be noted that it is possible to damage the Wafadrive circuitry, and possibly also the computer, by wrongly connecting the external lines. Only connect your own devices to the Wafadrive if you are absolutely sure of what you are doing. The Wafadrive guarantee does not cover repairs to caused by misuse.

The Centronics port is essentially an eight-bit output port plus two handshake lines – one input (BUSY), one output (STROBE). If you are knowledgeable about electronics, it can be used for general control, D to A, A to D etc. If you are going to experiment with it regularly, it is recommended that you connect via an optically-coupled buffer to minimise the possibility of accidental damage.

The eight-bit port can be written to in BASIC using the expression:

LET x=IN (14+256*byte)

Where *byte* is the value to be output. This should lie between 0 and 255. Variable *x* is a *dummy* variable and returns no useful value. It may seem strange to use the **IN** command to output something; this is a quirk of the Wafadrive hardware and is done to prevent clashes with other I/O devices.

The single bit input port (BUSY) can be read by performing:

LET x=IN 2

However, the port status is represented by the inverse of bit 5 of the returned value. To determine the true status, we have to *mask off* the other 7 bits and invert the result. This is very simple in machine code (and most dialects of BASIC) – but not so straightforward in Spectrum BASIC. This program will do the job though:

```
10 REM Read "BUSY" line
20 LET x=IN 2
30 LET x=INT(x/32)
40 LET x=NOT(x-2*INT(x/2))
50 PRINT x
```

Normally this program will yield the result 1 since the BUSY line is internally pulled high via a 1K resistor. Shorting the BUSY line to ground will give the result 0.

The above technique can be used to "read" a joystick connected to the Centronics port via an adaptor, which is available separately. Details of how to scan a joystick are included with the adaptor.

The additional Centronics interface output line (STROBE) may also come in useful. To "write" a value (1 or 0) to this line in BASIC, use:

LET x=IN (10+8192*NOT s)

Where *s* is the value to be "written" to the port, and can be either 0 or 1. Variable *x* is again a dummy variable and returns no useful result. When first switched on, the STROBE line is at logic 1.

It is possible to control the Wafadrive motors directly. The ability to "fast wind" the wafers under program control can greatly reduce the access time under certain circumstances. This would normally be done by a machine code program which intercepted the interrupt handling routine of the Spectrum so that the "fast wind" period could be timed, transparently to the operation of the main program. After winding the tape, the sector pointers in the system variables area must be updated as appropriate. The implementation of such a system is fairly involved, and beyond the scope of this introductory manual.

SPECIFICATION

PHYSICAL

Size:	230 mm wide x 110 mm deep x 80 mm high
Weight:	900 g
Cabinet:	Injection moulded, textured ABS
Colour:	Black with colour graphics
Indicators:	Red LEDs – two drive active, one power

DRIVES

Type:	Dual BSR "stringy floppy"
Capacity:	Min. 128K formatted (using 50' wafers)
Tape format:	Single track, 1K sectors
Data format:	FM encoded The
Tape speed:	10"/sec (read/write) 15"/sec (search)
Data integrity:	1 in 10^8 bit error rate
Reliability:	5000 hours MTBF
Transfer rate:	18K Baud (2K Bytes/sec approx.)
Access time:	Worst case – 6.5 secs (16K wafers), 45 secs (128K wafers)

WAFERS

Type:	"Entrepo stringy floppy" cartridges
Capacity:	16K, 64K and 128K (formatted)
Tape:	Infinite loop, 1/16" width. Specially developed backing and lubrication
Life:	5000 passes min.
Size:	67 x 45 x 6 mm
Write protection:	By removable tab
Tape protection:	Automatic sliding cover prevents damage

ELECTRICAL

Connections:	To computer via ribbon cable and standard Spectrum expansion connector. Connections passed on at rear of Wafadrive, enabling connection of Sinclair printer etc.
Power source:	9V nominal, derived from computer

RS232 INTERFACE

Implementation:	Five wire – RXD, TXD, RTS, CTS and GND
Baud rates:	Software selectable – 110, 150, 300, 600, 1200, 2400, 4800, 9600 & 19200

CENTRONICS INTERFACE

Implementation:	Eleven wire – DATA 0-7, BUSY, STROBE and GND

OPERATING SYSTEM

Size:	8K ROM – paged automatically
Reserved RAM:	2292 bytes (with no files opened)
Files supported:	Program, machine code or data files
Number of files:	32 maximum (16 using 16Kwafers)

ROTRONICS reserve the right to alter the above specification at any time, without prior notice.

INDEX

In this Index, entries referring to the normal Spectrum BASIC commands (eg. **NEW**), also relate to their Extended BASIC counterparts (eg. **NEW ***, **NEW #**).