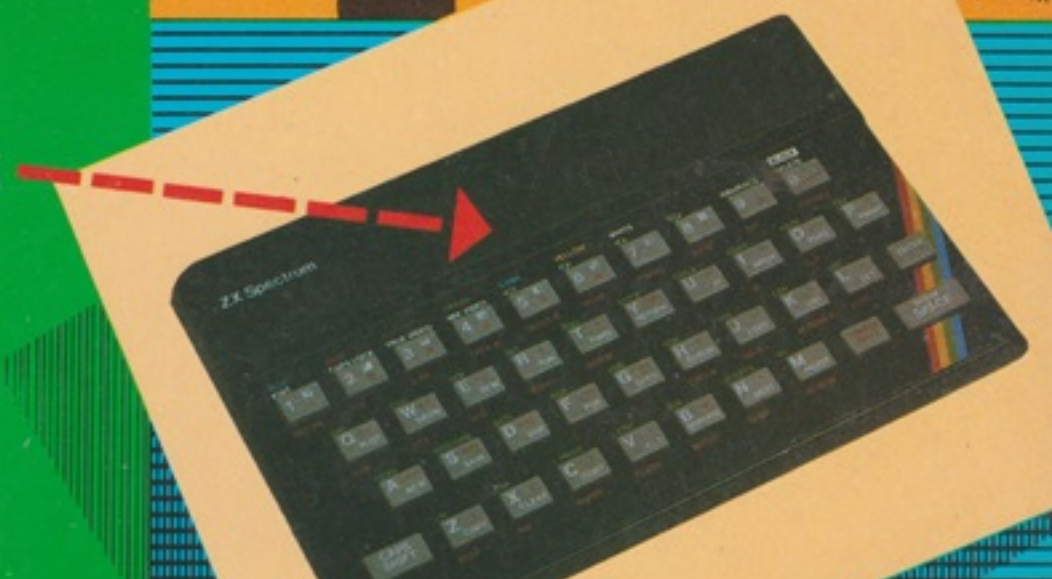
SPECTRUM

Melbourne House

MICRONET BOOK

Alan Giles

SPECTRUM



Chapter 3

Chapter 4

Chapter 5

Conclusion

Appendices

Introduction

This book is about the Prism VTX5000, which gives a Spectrum owner access to the enormous amount of information available on the Prestel computers, and in particular those parts of the Prestel database concerned with microcomputing, such as Micronet or Viewfax.

Readers who have not yet purchased a VTX5000 may wish to leave Chapter 1 for later reading and move directly to Chapters 2, 3 and 4, where they will find a lot of interesting background information about Prestel and its telesoftware and message facilities. In the same way, some readers may wish to leave until later some of the more complicated discussion of machine code routines. All the machine code routines are presented in a way which makes them easy to enter from BASIC by relatively inexperienced programmers, though some care is needed in typing the actual numeric machine code lists.

On occasions, the book refers to hexadecimal numbers. Such numbers are followed by the letter h, thus 15h = 21 is a true statement.

The word Prestel, used throughout this book, is a trade mark of British Telecommunications. Similar systems run by other organisations are known as Viewdata or Videotex systems. Micronet 800 is the trading style of Telemap Limited and British Telecom, the major providers of microcomputer information on Prestel. Homelink is a trade mark of Nottingham Building Society, who operate a home banking service through Prestel.

Chapter 1

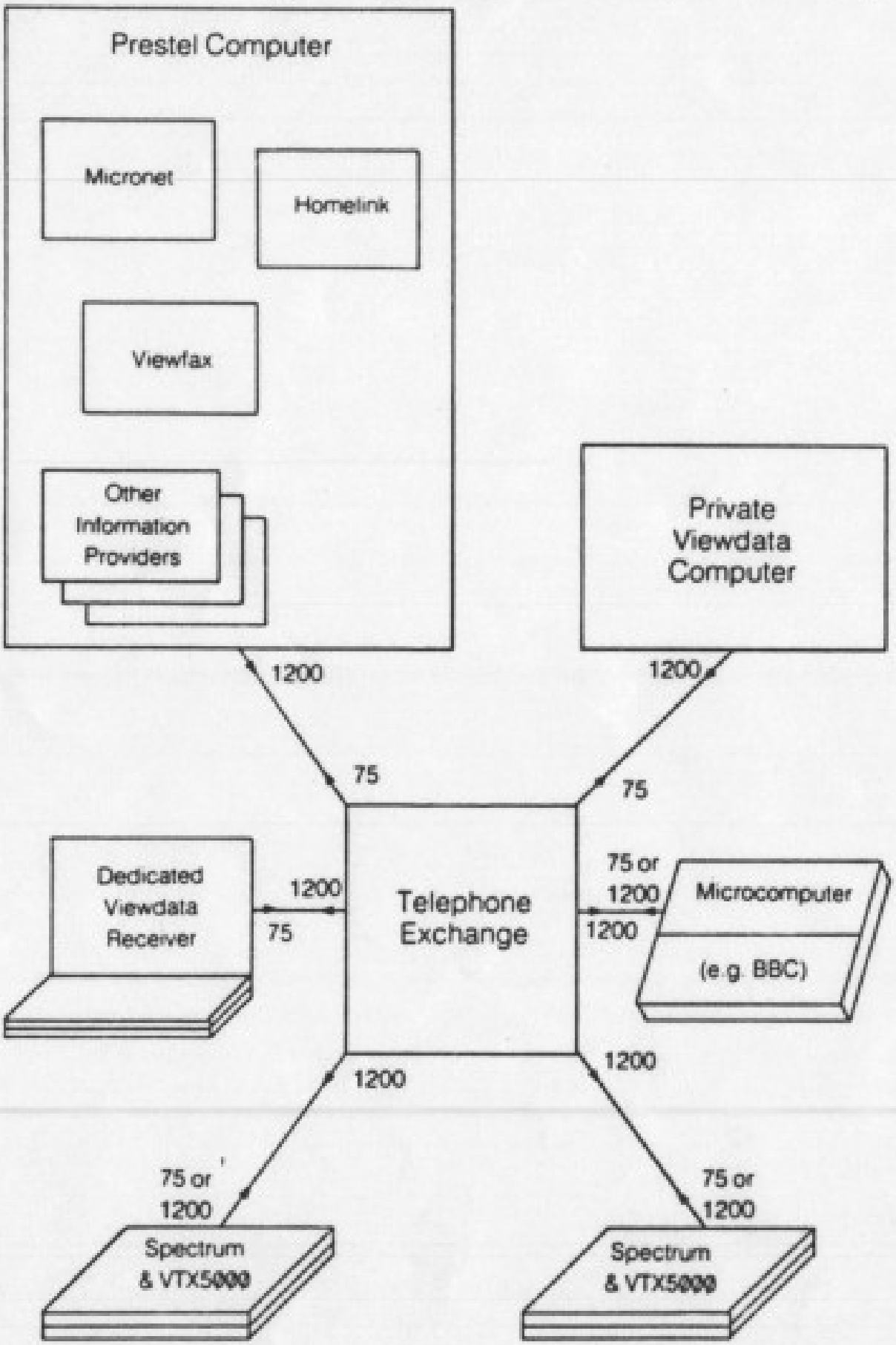
The VTX5000 hardware

The VTX5000 links a Spectrum to the outside world through a telephone line. The telephone line signals used by the VTX5000 are used by all other Prestel and Micronet equipment and by a large and growing variety of other computers. This means that the VTX5000 owner not only has instant access to Prestel, which already contains over 300,000 screens of information, but also a vast collection of information on other private computers.

Diagram 1.1 shows some of the possible connections that can be made through the telephone system.

Telephone lines do not provide a direct connection between a caller and the person or computer he calls. There are many electronic circuits in the path and, while these circuits improve normal telephone conversation by acting as amplifiers, or squeeze many telephone calls on to a single pair of wires, or whatever, they effectively prevent normal computer data signals from being passed directly down the line. What can be passed down the line are electronic representations of sound in the form of tones and whistles. A modem (modulator demodulator) is a device which turns computer data signals into tones which can be transmitted down a telephone line, and also receives tones and converts them into data signals a computer can understand. Two tones are used for transmission from the Prestel computer: 1300 Hz (about five times the frequency of middle C) is used for binary '1', and 2100 Hz is used for binary '0'. The modem electronics can demodulate these tones successfully as long as switching between the two tones happens no more than every 1/1200th of a second. Thus, transmissions from Prestel arrive at 1200 bits per second. These tones have used the high frequency end of the telephone bandwidth but another pair, 390 Hz for binary '1' and 450 Hz for binary '0', can be used for reverse transmissions to Prestel, to request information. These latter tones are switched at 75 bits per second.

Diagram 1.1 — Possible Data Connection Paths (and speeds — in bits per second)



The VTX5000 has a special 'Tx' mode which allows it to act like the Prestel computer and use the 1300 Hz/2100 Hz tone pair for transmission to other microcomputers or normal Prestel receivers, but it cannot receive or demodulate the 390 Hz/450 Hz tone pair.

The VTX5000 is often called a modem but it is more than this. If you have the courage to open up the plastic case and look inside, you will find two circuit boards connected by a few wires, as shown in diagram 1.2. The larger circuit board, and the more complicated of the two, is indeed a modem board (as used in Micronet adaptors for other microcomputers) but the smaller board is more interesting to us as it contains an 8251 USART (Universal Synchronous Asychronous Receiver Transmitter) and an 8K ROM chip, plus a few smaller supporting chips.

The 8251 converts 8 bit parallel data presented on the Spectrum bus connector into a bit by bit serial data stream suitable for use by the modem board. It also takes a serial data stream from the modem board and sorts it into 8 bit bytes which can be read by the Spectrum. This is a very similar function to the RS232 part of Interface 1. Indeed, there are inputs and outputs on the 8251 corresponding to the DTR and CTS signals of the normal RS232 socket. Such signals are ignored, because the Prestel computer will send data as fast as it can within the 1200 bit per second speed limitation of the line, and will not wait for the Spectrum to signal that it is ready for the next byte. The 8251 relieves the Spectrum of much of the work that it has to do on the Interface 1 RS232 link, by buffering bytes within itself, allowing the Spectrum 1/120th of a second to act on the previous byte. Consequently, the Spectrum can easily cope with the 1200 bit per second data rate, and the use of a VTX5000 to link a Spectrum to another microcomputer may well allow faster data transfer than the nominal 9600 bits per second of an RS232 link, which is implemented by using Interface 1.

One of the spare RS232 outputs from the 8251, called RTS (Request To Send), is used to enable or disable the 8K ROM in place of the normal Spectrum ROM, or Interface 1 ROM. All these ROMs share the same memory space in the Spectrum because the Z80 CPU chip can only access 64K of memory and the Spectrum may contain 48K RAM, all of which can be used by any of the ROM programs.

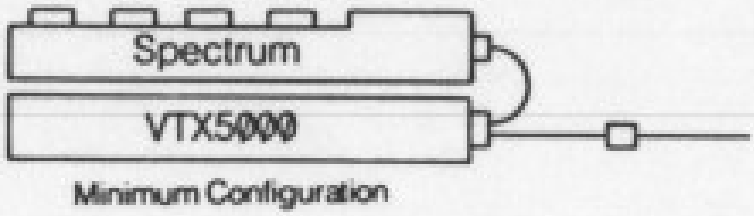
The ROM paging mechanism

On power up, the 8251 switches on in a mode where the VTX5000 8K ROM is enabled. The 8251 is sensitive to mains fluctuations during power up, and it is possible for the Z80 processor to become confused

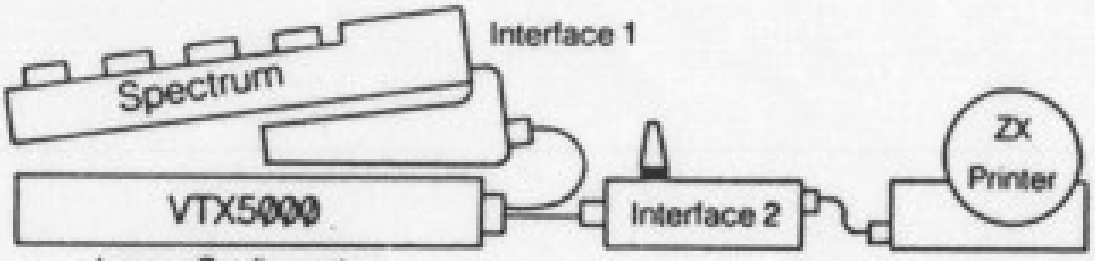
Diagram 1.2

The Equipment

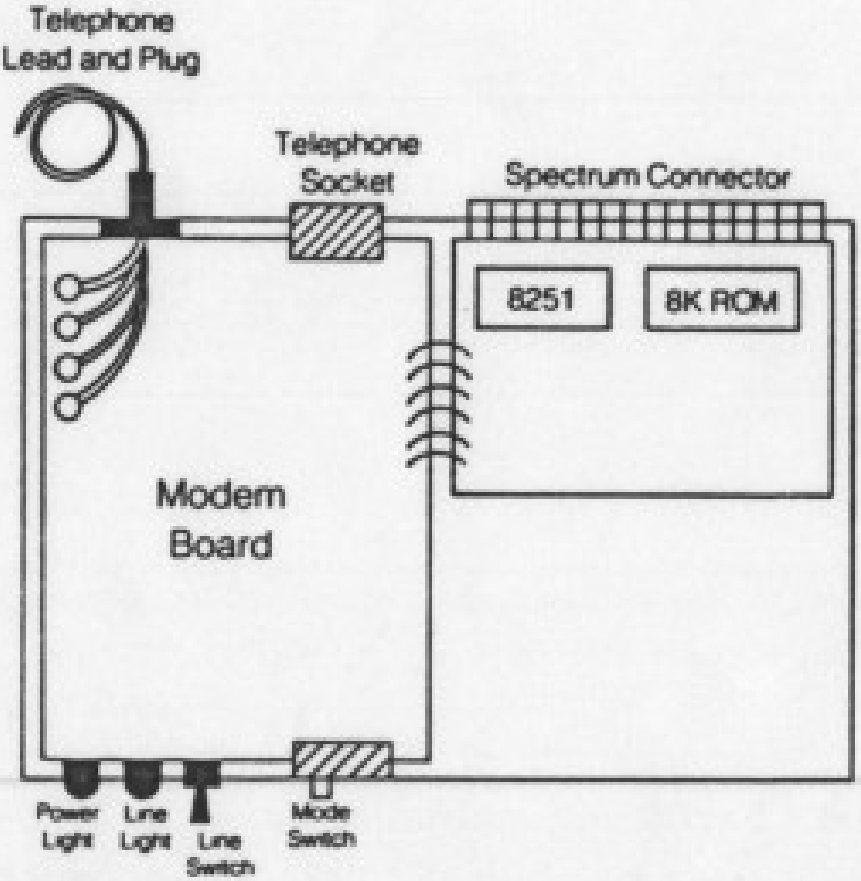
ART DELETE THE WORDS (side view) IN TOP HEADING



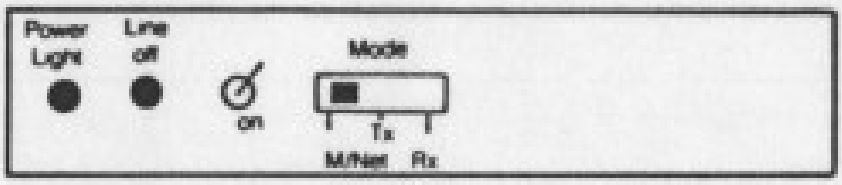
Minimum Configuration



Larger Configuration



Lights & Switches



53 = 1 transmit enable
2 dtr
3 receiver enable
4 normal operation
5 send break
6 rts

and show odd graphic patterns on the screen while ignoring keyboard input. The chance of this happening is considerably reduced, particularly if you have an Interface 1, if the power is switched on at the mains socket, rather than by inserting the power supply plug into the Spectrum. Dirty or badly fitted connectors between the Spectrum/Interface 1 and the VTX5000 can cause similar problems.

When the program in the VTX5000 is executed correctly, it determines whether the Spectrum is a 16K or 48K machine, and copies some data, machine code "paging" routines and a BASIC control program into the RAM. A welcome page is shown on the Spectrum screen until a key other than SPACE is pressed, when the VTX5000 ROM "pages out" and control is passed to the BASIC program, which displays a menu of functions. At this point it is possible to BREAK into the program, and alter the BASIC, adding new features to simplify and improve use of the VTX5000.

The VTX5000 instruction manual suggests that the ROM can be re-started from BASIC by the execution of RANDOMIZE USR 65507 or 32739, for 48K or 16K machines respectively. If you look at the routine at that address, and know a little about Z80 machine code, you will see that the code used to "page in" the ROM is:

```
LD      A,15h
OUT     (FFh), A
EX      (SP), HL
EX      (SP), HL
```

The last two commands are just a time delay while the 8251 hardware does the work. To "page out" the ROM, the 15h is simply replaced by 35h.

If you try executing the equivalent BASIC command:

```
OUT     255,21
```

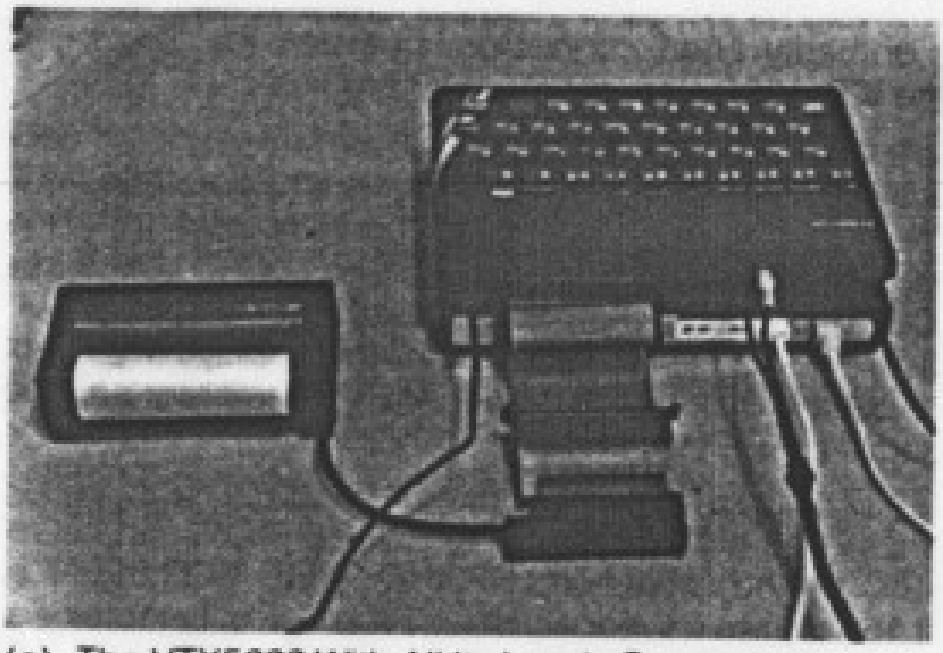
with a VTX5000 connected to your Spectrum, you are essentially exchanging the ROM in use while it is being executed. You will see from the screen that the Spectrum crashes and then, if you do not have an Interface 1, it generally recovers with the VTX5000 ROM controlling the computer. With Interface 1 connected, the Spectrum does not recover, and leaves a random graphic pattern on the screen.

Further details of the operation of the 8251 and its two I/O ports, the control port FFh and the data port 7Fh, are contained in Chapter 5. This includes a full discussion of the transmission of data and programs between a Spectrum/VTX5000 and another microcomputer.

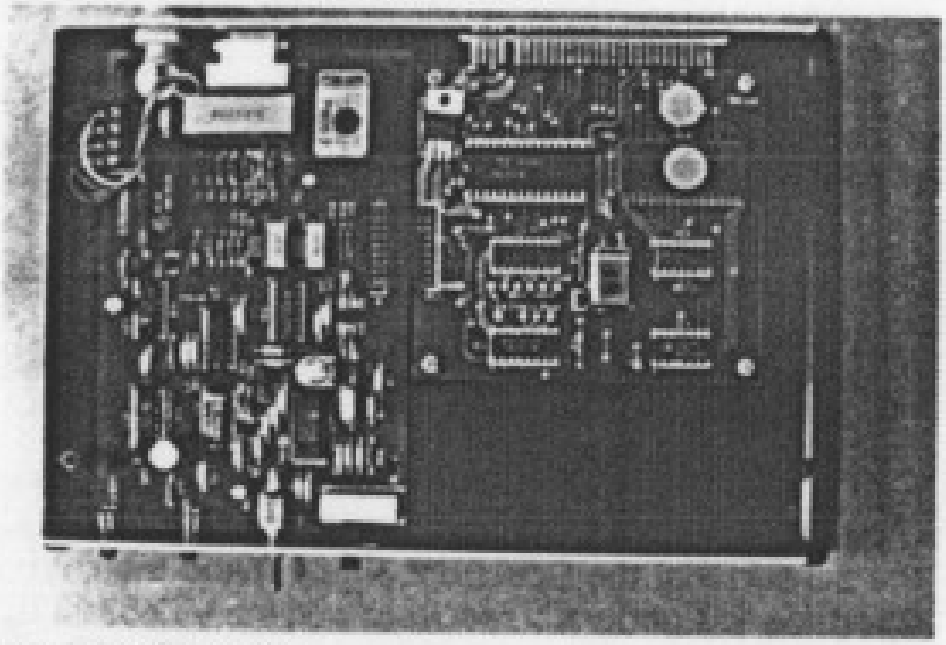
How the ROM paging affects Interface 1 and Interface 2 operation

Interface 2 is a very simple device, and if there is a ROM cartridge in the ROM socket, that ROM takes control of the computer. There is no

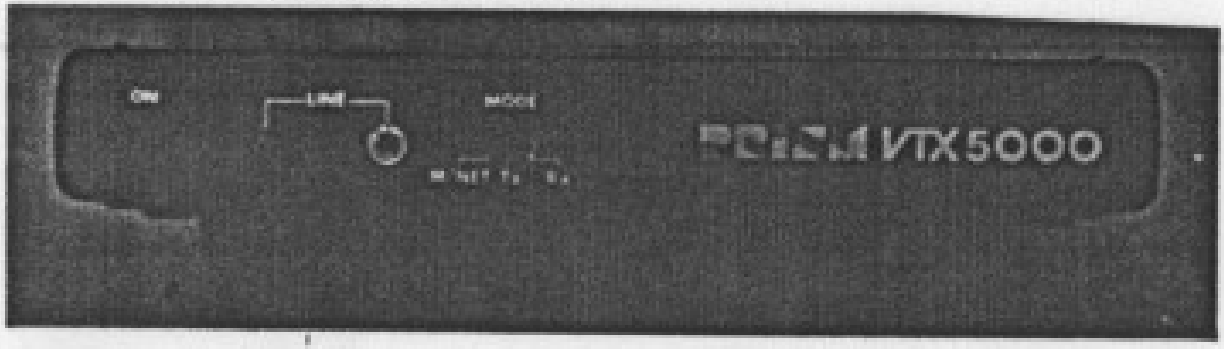
Photographs 1.1
The Equipment



(a) The VTX5000 With All Its Leads Connected



(b) The Contents



(c) The Front Panel

paging hardware in the Interface 2, so the program there will not allow the VTX5000, Interface 1 or original Spectrum ROMs to operate. Interface 2 can successfully be used as a joystick interface by telesoftware programs downloaded from Prestel using the VTX5000, so long as no ROM is put in the Interface 2 socket. It is always possible that someone could develop an Interface 2 ROM to replace the VTX5000 ROM but still use the 8251 and the modem board in the VTX5000. This would give the programmer 16K of ROM space in which to write his program, but this 16K would also have to contain all the machine code for any SAVEing and LOADing, as routines in the Spectrum or Interface 1 ROMs could not be called.

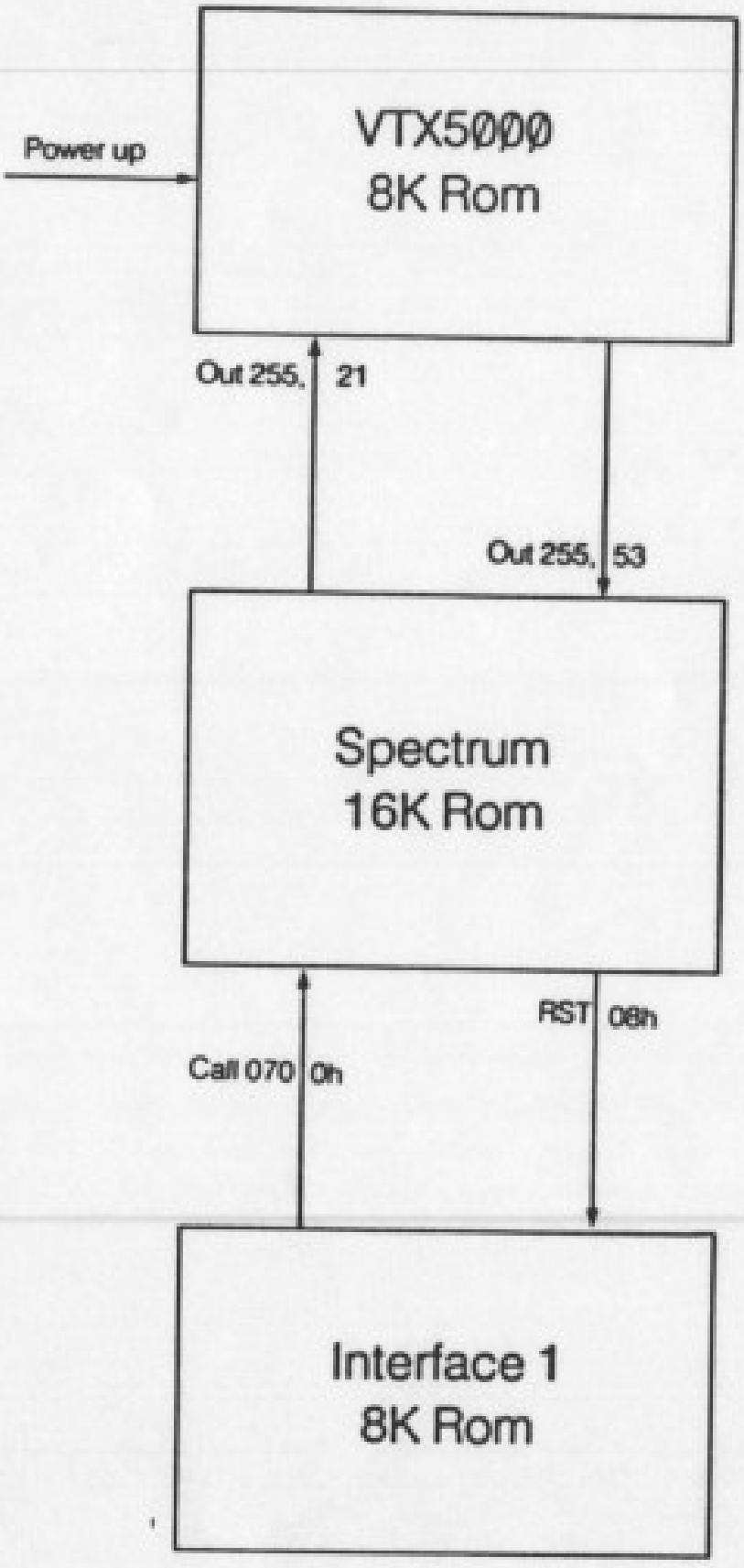
Interface 1 does not cause any problems with the VTX5000 until a BASIC error occurs, such as BREAKing out of the BASIC control program. Then the Interface 1 inserts its extra system variables, moving the BASIC program in memory, which may prevent the VTX5000 "Print Frame" routine from working, as it calls a machine code routine at a supposedly fixed address within the BASIC. (A solution to this problem is described in Chapter 2.) The extra memory usage by Interface 1 may also cause memory shortage or addressing problems when downloading telesoftware. (A way round this problem is discussed in Chapter 3)

The Interface 1 ROM is only "paged in" by the execution of a machine code command (to be precise an M1 Z80 CPU cycle) at address 0008h or 1708h. Neither of these locations in the VTX5000 ROM contains a command to be executed in an M1 cycle, so the Interface 1 ROM should only ever be "paged in" by the main Spectrum ROM. The normal routes for transferring control between the VTX5000 ROM, Spectrum ROM and Interface 1 ROM, in a system which contains all three, are shown in diagram 1.3.

The Interface 1 and microdrives are very useful for the rapid LOADing of the improvements to the BASIC control program described in this book. It is quite possible to LOAD from cassette every time you need the altered control program but using a microdrive is much faster. You should normally be wary of using option 7 on the VTX5000 main menu, as it clears out the whole of RAM, including the VTX5000 system variables and paging routines needed by the BASIC control program but, if you SAVE the variables, our version of the control program can LOAD them back, and adjust system variables UDG and P-RAMT. This is particularly useful with Interface 1, as there is no other simple way of restoring lost VTX5000 variables. If we do this, and SAVE our improved control program as the file "run" on microdrive 1, the procedure for LOADing from microdrive can be reduced to applying power, then keying "any key", 7, RUN and ENTER.

Diagram 1.3

ROM In Use



If you have a microdrive, the first improvement to the control program that you might like to make is to BREAK out of the control program while it is displaying one of its menus, then type:

```
SAVE*"m"; 1; "vtxvars"CODEix,128
90 LET d=PEEK 23766: LOAD *"m";d; "vtxvars"CODEix: LET
   udg=ix-960-600-255-21*8: POKE 23675, udg-256*INT
   (udg/256): POKE 23676, INT(udg/256): LET pramt=
   udg+167: POKE 23732,pramt-256*INT(pramt/256): POKE
   23733, INT(pramt/256)
```

The PEEK 23766 determines the microdrive which was used to LOAD the control program. You may continue to use the variable 'd' as the default microdrive. While we are making changes to suit microdrives, you can add:

```
* "m"; d;
```

in the appropriate position in the SAVE commands in lines 3000 and 7300, and in the LOAD commands in lines 4210 and 7400 of the BASIC control program produced from the VTX5000 ROM. The procedure for making these alterations is to simply LIST and EDIT each line in turn. When you have completed your alterations, type:

```
SAVE * "m"; 1; "run" LINE 10
```

and the new version of the control program is ready to be LOADed back whenever you need it. Many more alterations to the control programs are given in Chapter 2 and at appropriate points in later chapters.

Having both Interface 1 and the VTX5000 connected to the Spectrum can occasionally be slightly too much for the power supply. This shows itself as the Spectrum working correctly with either Interface 1 or the VTX5000 connected, but with both connected a random graphic pattern appears on the screen. If this ever happens to you, I find that leaving the Spectrum and Interface 1 switched on for ten minutes or so, before switching off, connecting the VTX5000 and switching on again, will stabilise the power requirements and allow everything to work correctly.

How to gain access to the ROM to disassemble it

There is just enough room in a 16K Spectrum to copy the contents of the 8K VTX5000 ROM into RAM, but you really need a 48K machine to embark on a proper disassembly. This is the machine code routine that I use to copy the ROM into the top 8K of a 16K machine:

```
DI
LD      A,15h
```

```
OUT     (FFh),A
LD      DE,6000h
LD      HL,0000h
LD      BC,2000h
LDIR
LD      A,35h
OUT     (FFh),A
EX      (SP),HL
EX      (SP),HL
EI
RET
```

To code this up in BASIC we first clear out the BASIC control program, either by choosing option 7 on the VTX5000 main menu, or by using the command NEW, then type in:

```
10      CLEAR 24575
20      LET c = PEEK 23637 + 256 * PEEK 23638 +5
30      REM 123456789012345678901234
40      FOR i = c TO c + 23: READ a: POKE i, a: NEXT i
50      DATA 243,62,21,211,255,17, 0, 96, 33, 0, 0, 1, 0, 32, 237,
        176, 62, 53, 211, 255, 227, 227, 251, 201
60      RANDOMIZE USR c
```

It is best to SAVE this before RUNning it, in case you have made any typing errors, as these would probably cause the machine to crash.

Once the program has been RUN, you can:

```
SAVE "vtxcode" CODE 24576,8192
```

and subsequently LOAD it back at the correct address for your favourite disassembler. (You will probably need to SAVE it in small parts if you are using a disassembler on a 16K machine.) When disassembling, remember that the code is being held at a different address to its proper ROM address, so that relative and absolute jumps will have to be interpreted differently.

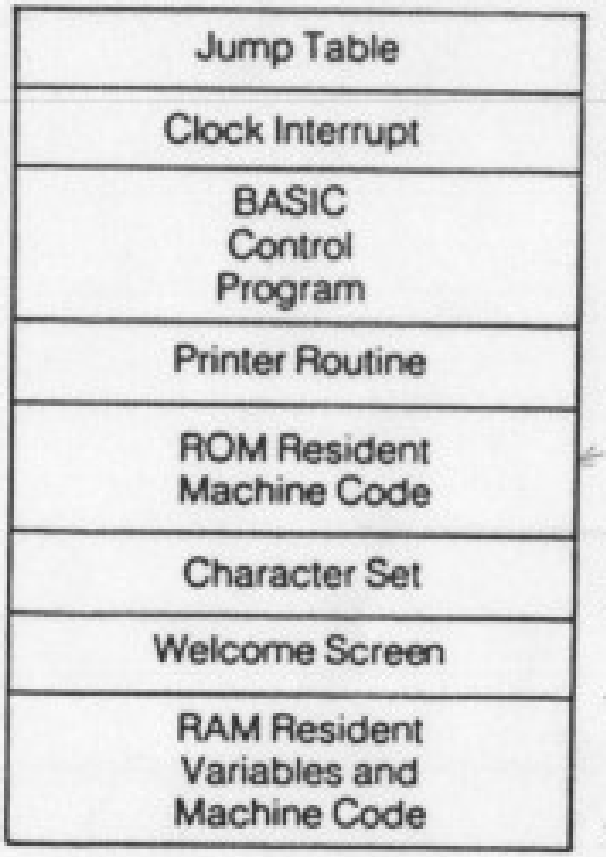
The ROM memory map

Disassembling the ROM contents allows us to construct the memory map shown in diagram 1.4. The ROM contains over 4K of BASIC, a little over 2½K of machine code routines, almost ½K of the special Prestel character set, and ⅝K devoted to the power on welcome screen with the Micronet or Homelink logo or whatever.

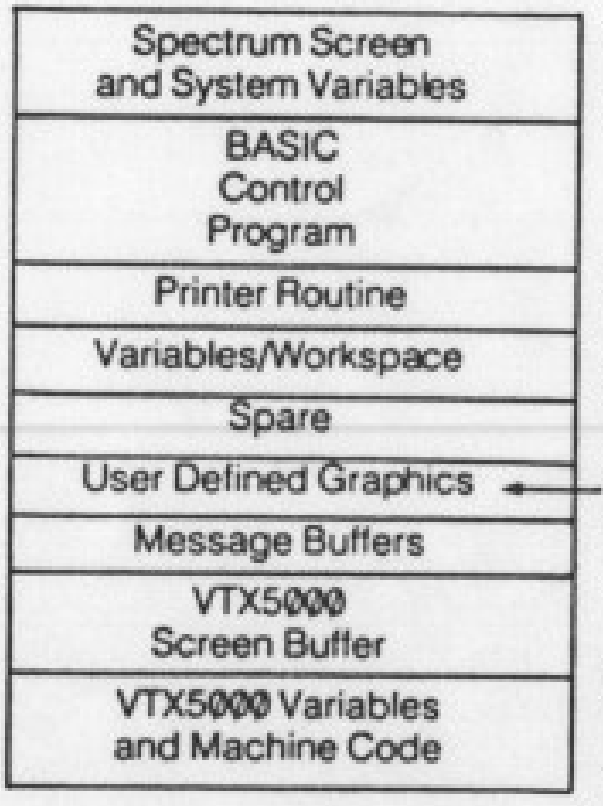
Diagram 1.4

VTX5000 Memory Map

a) ROM



b) RAM



The design of the ROM usage is slightly odd in a number of ways. The placement of a "jump table" at the beginning of memory is reminiscent of programming techniques on the 6800 or 6502 microcomputers, where there are special commands for jumping to the first 256 bytes of memory, in order to save space. The ROM resident machine code was probably developed in two separate parts (there is a small patch of FFh's in the middle which supports this theory), and the "jump table" is used by each of these two parts to call routines in the other part, when the precise address is not known. There are no direct jumps between the two parts which do not use the jump table.

It is also odd that the "printer routine" was placed in the BASIC section rather than in the RAM resident machine code section, which would have properly fixed its address and prevented the problems with the "Print Frame" routine.

The VTX5000 moves system variable P-RAMT, nominally the end of physical memory, in an odd way and places its buffers, variables and "paging" routines beyond that, rather than the more usual placement between a modified RAMTOP and UDG. The programmer obviously became confused carrying out this rather devious manoeuvre, and the first CLEAR in the BASIC control program sets RAMTOP to a point in the middle of the user defined graphics. As the machine stack moves up and down it destroys the shape of a number of the graphic characters. Also, if you LOAD a set of user defined graphics at the new address UDG without moving RAMTOP, you will crash the machine by overwriting vital information on the machine stack. Alternatively, if you LOAD CODE some user defined graphics, expecting UDG to be in its normal position, you will overwrite part of the VTX5000 screen buffer and all its variables and machine code routines without changing the shapes of the user defined graphics. Very confusing! Option 7 on the VTX5000 main menu does reset all this if you want to RUN your own programs which worked before you acquired a VTX5000, and any downloadable telesoftware program is aware of the problem and relocates RAMTOP and UDG appropriately for its own needs. This and other minor bugs in the VTX5000 ROM only become apparent in fairly obscure situations and, in general, we can congratulate the programmers involved.

The forty column display

The VTX5000 manages to mimic quite closely the forty column full colour display standard used by Prestel. This display standard is similar to the BBC microcomputer's MODE 7, or the broadcast Teletext services such as Ceefax and Oracle. Much of the work of the VTX5000 ROM machine code is concerned with displaying forty column pages on the Spectrum screen, and we can harness this power within our own programs without too much effort.

First, we will examine the special Prestel character set, which is held in the VTX5000 ROM starting at address 1B20h. If you have used the routines in the earlier section to copy the VTX5000 ROM into RAM, we can examine the slightly unusual orientation of the character set. There are five bytes for each of the 96 displayable characters, so we can write a short BASIC program to display these characters on the first three lines of the Spectrum screen in 32 column mode. (If you have moved your copy of the ROM from address 24576 you will have to change line 10 appropriately.) Thus:

```
10    CLS: LET start = 24576 + 6944
20    FOR i = 0 TO 95
30    FOR j = 0 TO 4
40    POKE 16384 + i + 256 * j, PEEK (start + i * 5 + j)
50    NEXT j
60    NEXT i
```

The following routine converts normal BASIC PRINTing on stream 2, and any other stream using channel "s", to work on a forty column screen. The routine can be incorporated either in the BASIC control program or in any other BASIC program to make use of the display routines in the VTX5000. It is thus best to SAVE the routine on its own, so that it can be MERGEd into programs as required. It uses the VTX5000 screen buffer and variables, so reset the VTX5000 by an OUT 255,21 or other appropriate action, BREAK out of the ROM control program, and use NEW to clear the BASIC control program out of memory, then type:

```
9900  LET s40 = PEEK 23637 + 256 * PEEK 23638 + 5
9910  REM 12345678901234567890123456789012345678901234
9920  RESTORE 9930: FOR w = s40 TO s40 + 33: READ v:
      POKE w,v: NEXT w
9930  DATA 217, 197, 213, 229, 245, 243, 62, 21, 211, 255, 227,
      227, 241, 245, 205, 179, 19, 251, 205, 14, 22, 243, 62, 53,
      211, 255, 241, 225, 209, 193, 217, 167, 251, 201
9940  LET cs = PEEK 23631 + 256 * PEEK 23632 + 5: POKE
      cs, s40 - 256 * INT (s40/256): POKE cs + 1, INT (s40/256)
9950  PRINT CHR$ 12; : RETURN
```

Like most of the machine code routines in this book, the code is stored in a movable REM statement, so as to avoid using any fixed addresses which you may wish to use for your own purposes. However, this means that whenever you add any lines of BASIC, add any Interface 1 variables, or OPEN or CLOSE any Interface 1 channels, you must execute the routine again, using GO SUB 9900, if you wish to do any forty column PRINTing. Normal 32 column PRINTing can be restored at any time by use of CLS, CLEAR, or by an automatic LISTing of part of the BASIC program.

The machine code essentially "pages in" the VTX5000 ROM and calls the ROM routine at address 160Eh with a Prestel character code in the A register, then "pages back" the Spectrum ROM and returns. PRINTing CHR$ 12 clears the screen and sets the current PRINT position to the top left corner.

The ROM routine at address 160Eh does, of course, stick strictly to Prestel control codes listed in Appendix I, so Spectrum INK, PAPER, AT, TAB and PRINT commas will not work in the usual Spectrum manner; nor will any scrolling take place, or keywords PRINT out in full. We could add more machine codes to our routine to mimic all these Spectrum features, but they can all be achieved by careful choice of the characters you PRINT, and writing this code would divert us from the task of investigating Prestel and its screen format. Do, however, note that lines of normal PRINTing should be terminated by CHR$ 10 to ensure movement on the next line.

You can now try all the Prestel codes using this short program:

```
10      GO SUB 9900
20      INPUT v: PRINT CHR$ v;: GO TO 20
```

AT can be simulated by POKEing new row and column values into locations ix + 12 and ix + 13 respectively, where ix is 32640 or 65408 depending on whether the machine has 16K or 48K RAM.

It is also worth noting that the whole VTX5000 character set can be changed by POKEing the address of the byte containing the left hand column of pixels of the first character, SPACE, into locations ix + 70 and ix + 71 in the usual "least significant byte first" format. If this is done, 128 characters can be added to the set, making 224 user defined characters. As the ROM routine at 160Eh starts with the machine code command AND 7Fh, you need to CALL 1610h instead to allow these extra 128 characters, i.e. the numbers 205, 14, 22 in the middle of DATA statement 9930 need to become 205, 16, 22.

Make sure you SAVE a copy of the forty column display routine, as it is useful when preparing Prestel messages, which we move on to in Chapter 4. Meanwhile, we must learn much more about Prestel itself.

Chapter 2

Calling PRESTEL

Before we delve into the intricacies of calling up Prestel pages using a Spectrum and VTX5000, I think it would help to look at the history and development of Prestel. Much of the way Prestel behaves hinges on design decisions made in the early 1970s, when technology in general, and the microcomputer in particular, was nowhere near as advanced as it is today.

The Early Days

Sam Fedida is credited with the invention of Prestel or Post Office Viewdata as it became known at the time. He was working on a "viewphone" project, trying to squeeze the signals of slow scan television pictures into the limited bandwidth of a normal telephone line. He and his team realised that the problems they were encountering were too difficult for "viewphone" to be developed, but by using current technology they could send text and simple diagrams down a telephone line for display on the recipient's television set.

Thus viewdata was born. It was conceived as a cheap, mass information service — so cheap and so massive that everyone would have a receiver in their home and would use it regularly in place of an encyclopaedia, railway timetable, or other reference book.

To illustrate the practicality of the system, the Post Office asked for tenders for a trial batch of fifty receivers. To keep the costs down, these receivers were to be black and white only, with a 32 column by 16 row display in capital letters only, having a 12 button keypad (like push button telephones with 0 to 9, plus * and #), and using an asymmetric transmission system with data from the viewdata computer coming at 1200 bits per second, but with the return channel limited to 75 bits per second.

At about the same time, the BBC and IBA were working on a system called teletext, which broadcast pages of information coded into the area of the television signal just above the normal picture. This system had a 40 column, 24 row colour display, and when the Post Office invitation to tender reached one of the companies involved in the manufacture of early teletext receivers, that company replied that, because of this existing product, it would be cheaper for the company to supply teletext display standard full colour receivers, rather than the black and white specification.

Apart from a few minor alterations, this is how the basic viewdata receiver design was created.

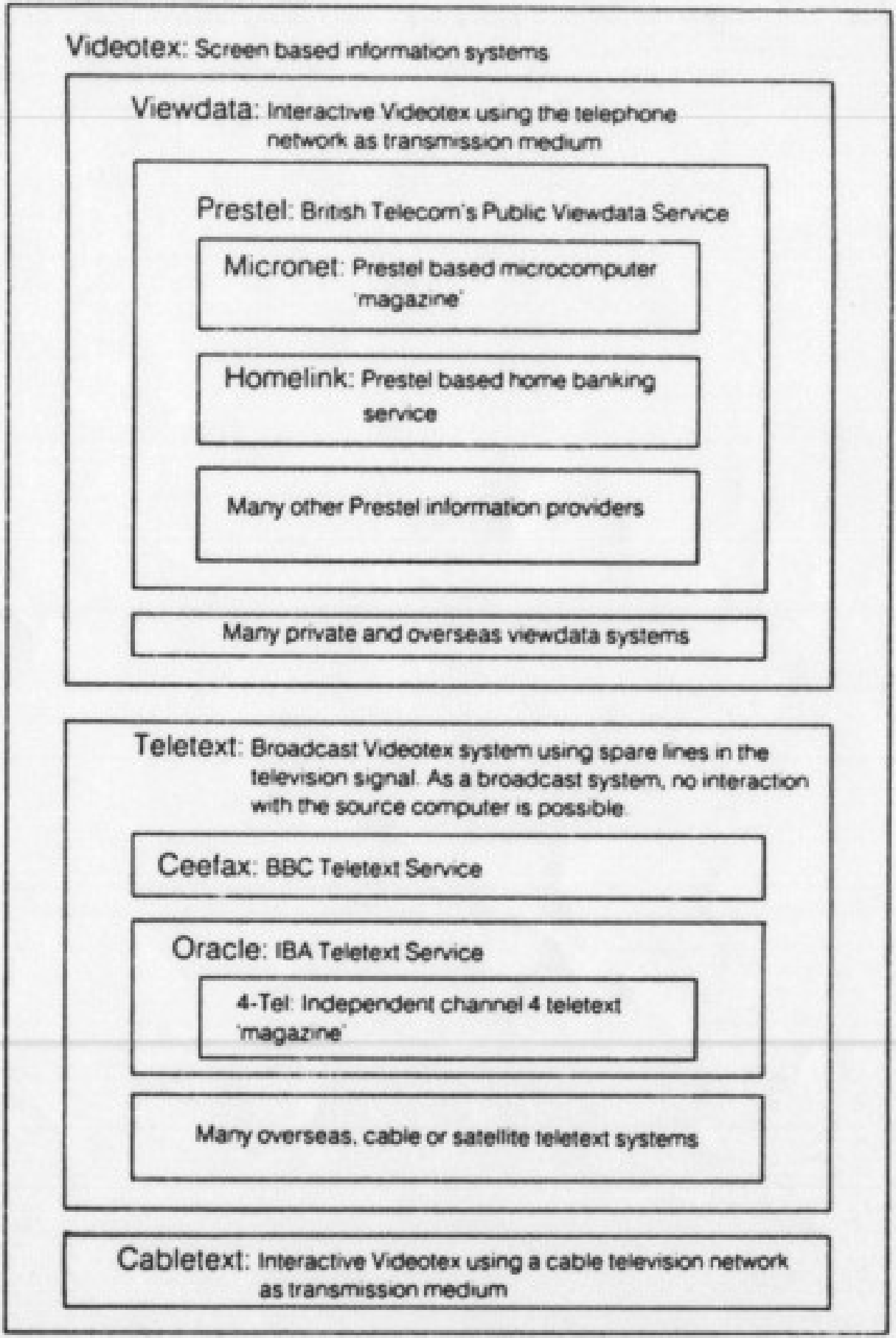
One of the problems, which still affects us today on the Spectrum, was that the teletext character set had no # symbol, as it had been moved to make way for a £ sign; however the cheap mass produced telephone keypad to be used with viewdata sets already had a # engraved on one of the keys. So, by arrangement with the BBC and IBA, this symbol was re-introduced into the character set but in an odd position in the code table.

The electronic circuitry needed in one of these early receivers took up three circuit boards, each larger than the Spectrum, and the modem was a large heavy metal box full of relays, transformers and coils.

Receiver Developments

Slowly, over the ensuing years, many refinements have been added to viewdata receivers. The electronics have been reduced considerably in size by "large scale integration" techniques and the use of a microprocessor chip to do much of the labour of shifting data around inside the receiver. An expanded display standard has been developed, adding Background Colour and Double Height to the original colour on black background limitations. Due to the design limitations on teletext, each colour or display mode change takes up a space on the screen. To overcome this, a mode called Hold Graphics was also felt necessary. This mode essentially allows a gap between two areas of graphics to be closed up by the repetition of the previous graphic shape on the colour control character position which would normally appear in the background colour.

Diagram 2.1
The Terminology



Display Standards

The teletext display's need for a space while changing colour does help the Spectrum to fit forty characters on a line with only thirty two separately colourable character positions. It is just possible to ensure that the INK colour for normal text never has to change in the middle of a character. However, the Spectrum cannot guarantee to start its PAPER colour in the exact position instructed by a New Background control code. In a similar manner, Flash and Hold Graphics cause positional problems. Indeed the Spectrum Flash is completely different in nature to the teletext Flash. The Spectrum exchanges INK and PAPER while the teletext definition is that the flashed character should disappear, leaving the Background Colour visible.

Double Height is achieved on the Spectrum in software, and there are a couple of minor bugs in the VTX5000 ROM associated with the interpretation of characters apparently intended for display on the hidden line immediately below the Double Height line. If a Double Height control code is received on the hidden line, the VTX5000 incorrectly blanks out the next line down, making that another hidden line. (Animation effects are often achieved on normal viewdata receivers by the use of a "Venetian blind", where there are Double Height codes on every line, and the uppermost such code is alternately written over and retransmitted, causing the receiver to change between displaying the odd and even numbered lines. This is not possible on the VTX5000.) If Line Noise is received on a hidden line on a message page, the characteristic blob, Prestel character code 7Fh, can be displayed, blocking the view of the lower half of a doubled character.

Modem developments

Modem electronics, in the same way as display electronics, have reduced in size to the point where they easily fit inside viewdata receivers or adapters as small as the VTX5000. Generally these modems also have the ability to generate dialling pulses on the telephone line, to call up Prestel without user intervention, automatically dialling a number stored in battery-backed non-volatile RAM.

Concurrent with these improvements to the receiver, similar improvements have been taking place in the Prestel computer network.

The Prestel computers

Sam Fedida's team quickly chose the GEC 4000 minicomputer series for use as their central computer. The computer's internal architecture and, in particular, its I/O processor and microcoded multi-process execu-

tive kernel Nucleus, make it particularly well suited to the simultaneous handling of a large number of telephone calls. The current computer network uses a collection of GEC 4082 computers, each with a typical 384K bytes of RAM and eight 70 megabyte disc drives to hold the information, which are capable of handling over 200 simultaneous telephone calls and responding to requests on any of these lines within a maximum of two seconds. There are plans for a second generation network using a mixture of GEC 4065, 32-bit GEC 4190 and even more powerful GEC 6300 series computers.

The first viewdata computer was based at the Post Office Research Centre at Martlesham Heath, near Ipswich. Once a decision was made to open the newly retitled Prestel commercially, the data was moved to a computer in London, known as the Gresham Street computer. As one computer could not cope with the traffic expected, the network was soon expanded to a "central update" computer Duke and a user computer Byron in London. Byron was quickly followed by Atlas, Vigilant and Juniper in other parts of London, all named after old telephone exchanges. Outside London, the computers tended to come in pairs — Derwent and Enterprise in Croydon, Dickens and Keats in Birmingham, Arkwright and Wordsworth in Manchester, Burns and Scott in Edinburgh, Chippendale and Priestley in Leeds, Dylan and Megan in Cardiff, and Constable and Gainsborough in Chelmsford. There was also a separate Prestel International service on a computer called Hogarth.

The point of having all these computers was to give as many people as possible access to Prestel at the cost of a local telephone call, and to cope with the expected massive demand for Prestel by users. The computers were paired together, so that if one of them was not working for some reason, users could call the other one and still obtain the information they needed. At first this was achieved by calling a separate telephone number, but later alternate lines on the same telephone number were taken to the other computer. This interleaving means that you are much more likely to get through to the Prestel computer on your first call, but you cannot choose which of your two local computers you obtain. This is usually not important but can lead to problems if, for example, you change your password on one computer but not on the other.

Other towns, like Liverpool or Newcastle, Belfast or Bournemouth, were served by multiplexers in their local telephone exchange, which concentrated a number of Prestel calls on to a single permanently connected line to a demultiplexer in one of the computer centres.

At this stage, Prestel was essentially an information source but it had a very simple message service using response frames, which are partially pre-formatted pages which a user can send to the appropriate Information Provider. 'Send' is not quite the right word for this early version, as the messages were simply stored on the computer being used. I worked for an Information Provider at this time and I can remember the chore, which we did once a week on Tuesday afternoons, of ringing round all twenty computers to collect the week's messages.

Various technological advances came in quick succession at this point in history. Multiplexers became better and cheaper, the first 200 telephone line GEC 4000 computer was put together, and so the decision was made to shift the bias of the network away from a large number of computer centres to a more rationalised service. The first stage of this move came, confusingly, with the opening of two new computers in London, called Dryden and Kipling. Gradually most of the other computer centres were closed down and replaced by multiplexers, leaving just Duke, Dryden, Kipling, Derwent, Enterprise, Dickens and Keats. During this changeover, the Prestel computer software was improved so that messages to Information Providers were ingathered to Duke and user to user messages were allowed; but these user to user messages, or mailbox frames, stayed on the computer they were sent on like the old response frames, so it was agreed to use just one of the computers, Enterprise, for mailbox.

Only now, as this book is being written, is mailbox being improved to become a national service, with messages being ingathered to a new computer, called Pandora, which then sends them to the computer next used by the intended recipient.

Another technological addition to Prestel has been the connection through Gateway pages to External Computers. Prestel has always been designed as a minimum cost information source, and has therefore given users the absolute minimum share in the GEC 4000 computer's processing power, so as to maximise the number of users each computer can serve. The connection of External Computers allows Service Providers to give you access to the power of their computers. Thus, Homelink can give you electronic banking services on their computer, Skytrack can allow travel agents to book airline tickets on a network of airline computers throughout the world, and so on.

The database

While the design of Prestel receivers and Prestel computers has advanced considerably in the ten years or so since they were first thought of, the databases — the collections of pages of information — have

advanced no less dramatically, fuelled in part by the advances in receiver display technology and in the Prestel computer hardware and software.

For instance, in the early days all pages of information had what is still called "standard" or "strict" routes (a route is a connection between one page and another). If you were looking at page 156 and keyed 3, you would receive in reply page 1563. Very early on, this restriction was removed and "free" routes became available, where the editor could independently link the ten routes on any frame to any page.

That last sentence used the words frame and page in a very technical manner. Strictly, a frame is a screen full of information and a page is a number of frames (up to 26) all with the same page number. The first frame of a page is the 'a' frame, which is the only frame to which numbered routes can lead from other pages or frames. Keying # while looking at an information frame will call up the frame with the next suffix letter. # is thus the last remnant of the old strict route scheme, as you can always tell from the frame number and letter at the top of any frame you look at, exactly which frame you are requesting if you key #.

Once the ability to use free routes came in, route 0, and to a lesser extent route 9, came to be widely used as routes "back" to earlier index pages. So, in general, 0 is something of a "help" button when looking at a Prestel page.

Normally, when looking at Prestel frames you simply obey the instructions written on them, but you can get away from the current frame by using * commands, such as:

* #	go back to the previous frame (can be used a maximum of three times in a row)
* number #	go directly to the 'a' frame of the numbered page
* 0 #	go directly to your page zero, typically the microcomputing or Homelink index page, depending on the coding given on your Prestel application form
* 1 #	go to the Prestel general index (page one)
* 0 0	repeat the current frame to check that it has been displayed correctly
* 0 9	update the current frame by recalling it from Prestel disc store (you are charged again if the page costs money)
**	wipe out what you have just typed, correcting an error

The top line of every Prestel frame tells you three things you may want to know — the name of the Information Provider who owns the page, the frame number and the price of that frame. The bottom line is reserved for error messages from the Prestel computer.

You can leave Prestel at any time by calling up page 90, using * 90 #. This is what the "Log Off" option on the VTX5000 menu does. Page 90 tells you if there are any new messages waiting for you, which may have arrived during the course of your call, and how to look at them. Note that you cannot see any second message page until you have either ERASEd or STOREd your first message, and it is possible for users not to have any storage allocation, in which case an attempt to STORE the message will be refused. This problem should be removed by the arrival of Pandora and the National mailbox service.

You can also leave Prestel keeping the last frame on the screen.

On the VTX5000 this is achieved by switching the line switch to its upper position and then following menu options to view the current frame. This can be a useful way of saving telephone charges, if all the information you need is held on one frame.

The Information Providers

The travel industry was the first major group to discover the commercial viability of Prestel technology. Sealink had a problem, in that they sell so many car ferry spaces that it is impossible for them to cope with telephone calls from travel agents every time someone goes into a travel agency to book a crossing. So they instigated a manual "sell and record" system, whereby the travel agent could sell tickets without reference to Sealink, using a chart to tell him whether plenty of space was still available on each crossing or not. Weekly updates to the chart were sent by post to all the agents involved. Of course, many agents did not have time to keep their chart up to date, and this resulted both in overbooking and in over cautious telephone calls to Sealink. Then Prestel came on the scene and Sealink put their charts on to Prestel pages. They could now keep these pages much more up to date than the old charts, and it was cost effective to subsidise the installation of Prestel receivers in their top 2,000 travel agents, thus saving postage charges and telephone time for their staff.

With this market of travel agents and their Prestel receivers out there to be tapped, many other travel companies — ferry operators, tour operators etc. — found it economic to embark on similar ventures, putting travel news and availability details on to Prestel.

In the same way, with all this information available, many travel agents found it economic to pay the full price for Prestel receivers. Today we have reached the point where it is unusual to go into a travel agency which does not have at least one Prestel receiver.

Another interesting travel database is Sky Guide, provided on the pages of American Express. This shows the aircraft departure and arrival boards at all major UK airports, so you can check up on the latest details of delays on any flight you might be taking or meeting.

Micronet is another success story, and the reason for the creation of devices like the VTX5000. Micronet subscribers have access to frequently updated microcomputer news pages, background information on a wide variety of microcomputer topics and, most interesting of all, computer programs known as telesoftware which can be downloaded into a computer's memory direct from the screen, as we shall discover in Chapter 3. The falling prices of hardware and software mean that a 16K Spectrum and a VTX5000 together form almost the cheapest Prestel adaptor currently available, and yet this combination has all the features of the Spectrum computer, and all the benefits of the telesoftware on Micronet.

Homelink is another example of an economically viable use of Prestel technology. If you invest a certain amount of money in the Nottingham Building Society you gain instant electronic access to the details of your account and a related Bank of Scotland account. You can transfer money freely between these accounts at any time of day or night, and thus effectively obtain a Building Society account with a cheque book, or alternatively, a current account with the interest rate you would expect from a Building Society. A VISA card is also included in the package, giving you cash 24 hours a day from Barclaybank dispensers, and you can use the card number to teleshop, ordering goods through Prestel. You can pay your VISA bill and other regular bills directly from your Building Society account by giving the appropriate electronic authority.

These are just some of the projects currently making good use of Prestel. There are many others in the pipeline, or hidden away in closed user groups, or in private viewdata systems intended for particular groups of specialist users.

Using the VTX5000

Now that we know what Prestel is, let us see how to call it up on a Spectrum. The VTX5000 does not have a couple of pieces of hardware that most Prestel adaptors or dedicated receivers have, namely an auto-dialler and some battery-backed non-volatile RAM. The non-volatile memory usually holds the telephone numbers for use by the auto-dialler, and your customer identity, the number which identifies you to Prestel.

On power-up, the VTX5000 knows that its memory is empty and holds no customer identity so, even if you claim to want to do an "automatic log

on" following the screen menus, it will ask for your customer identity. The software is very pedantic in requiring a ten digit number in reply. Once this number is entered, the VTX5000 will ask you to dial the computer.

Your customer identity and the local Prestel computer telephone number are shown in a letter to you from Prestel, along with your initial personal password and Prestel account number. The customer identity and personal password are secret numbers. Known only to you and the computer, they allow you to correctly identify yourself for charging purposes when you call Prestel. The Prestel account number (also sometimes known by its previous name of systel number) is a public number you can tell people so that they can send mailbox messages to you, put you into a closed user group, or know which bill you are complaining about.

I assume that having followed the menus which the VTX5000 puts on the Spectrum screen (telling it that you wish to "log on"), and having typed in your ten digit customer identity and pressed ENTER, the screen now says "Phone Computer". At this point it is worth checking that all the plugs and switches are in their correct positions. The line switch on the VTX5000 should be in its upper position, and the mode switch should be in the left position, called M/NET. Your telephone should be plugged into the back of the VTX5000, and the VTX5000 telephone lead should be plugged into a telephone socket. If any of these positions are wrong, you will not get through. Also, if you type anything on the Spectrum keyboard it will send the customer identity prematurely, and the Prestel computer will ask you to type it again when you do get through.

Now, just pick up the telephone handset and dial the number given by Prestel. If the Prestel computer is working, it will answer with a high pitched whistle, which is recognisably different from the normal telephone system's "unobtainable" tone. Switch the line switch into its lower position, and replace the telephone handset. A few seconds will elapse while the two computers identify their respective whistles and request and transmit your customer identity, which appears on screen as echo dashes, so that anyone watching over your shoulder cannot tell what it is. Then the Prestel computer asks for your personal password, which you type in by hand on the Spectrum keyboard. There is no need to press ENTER at any stage; in fact you will probably cause problems if you do. If you make a mistake in keying the password, press SYMBOL SHIFT and then SYMBOL SHIFT again, and retype all four digits of the password. To help you move around Prestel quickly without too many typical Spectrum multi-key depressions, the VTX5000 interprets SYMBOL SHIFT on its own as *, and ENTER is interpreted as #.

CAPS SHIFT used with ENTER takes you to the VTX5000 main menu, so you can abandon a call to Prestel by selecting "log off", or PRINT or SAVE the current frame you were looking at.

CAPS SHIFT used with 1 (i.e. EDIT) is equivalent to the reveal key that some Prestel pages may ask you to press.

There are also a couple of special key combinations which are useful on message pages. These will be discussed in Chapter 4.

Adding a permanent automatic customer identity

To speed up your use of Prestel, you can add your customer identity to the BASIC control program in a more permanent fashion.

BREAK into the program while it is displaying one of its menus and type the line:

```
1000 IF Is=I THEN GO TO 1700
```

Delete lines 1100, 1110, 1400, 1440, 1450 and 1900, which are no longer needed, and include your customer identity in line 340 as follows:

```
340 LET i$ = "customer identity"
```

You can include your personal password in this text string, to remove the need to key it, and even a * number # command to take you off to a particular page.

After a telesoftware downloading failure, the BASIC control program in the VTX5000 ROM is copied back into RAM and you will have to reLOAD your own version. If you don't mind using the ROM control program, but would like your customer identity to be remembered, add the line:

```
341 FOR i=o TO 9 : POKE id+i, CODE i$ (i+1): NEXT i
```

There is, unfortunately, no room for the personal password in the VTX5000 variable space reserved at address 'id'.

Before you lose it, SAVE the new version of the control program, starting at LINE 10. To use it, you have to BREAK and LOAD every time you switch on the VTX5000, or after downloading telesoftware.

The control program relies on the presence of the VTX5000 variables and machine code in high memory. If these are lost, for example by the use of option 7 on the VTX5000 menu, you will have to OUT 255,21 (or, if you have an Interface 1, it is best to LOAD the variables from microdrive, as described in Chapter 1).

Unfortunately, altering all these lines of the BASIC control program has caused another problem, as it has shifted some machine code — used by the 'Print Frame' routine — from address 28041. Do not try it now, or you will crash the program.

Solving the bug in the Print Frame routine we have just caused

The normal Spectrum COPY command only prints the top 22 lines of the screen, which is a problem when you come to printing out the 24 lines of a Prestel screen. The VTX5000 has a little machine code routine:

```
DI
LD      B,C0h
JP      0EAFh
```

which overcomes this by setting the line count to C0h (192 lines of pixels) before jumping into the COPY routine in the Spectrum ROM. These six bytes of machine code are located at the end of the BASIC program area of RAM, being copied there during VTX5000 initialisation.

Additional lines of program, or Interface 1 system variables, shift this code so that if you try to print a Prestel screen, the program will try to execute part of the BASIC program as if it were machine code, and crash.

You can actually add a few bytes of program without apparently upsetting the print routine, because just before the routine is a patch of unused memory, filled with FFh. This is interpreted as machine code command RST 38h which mimics a Spectrum real time clock interrupt, and succeeds in advancing the system variable FRAMES by a count of one each time it is executed.

The following program stores its own version of the print machine code in a movable REM statement, whose address is recalculated every time it is executed:

```
5000    LET print = PEEK 23637 + 256 * PEEK 23638 + 5
5001    REM 123456789001
5002    RESTORE 5003: FOR w = print TO print + 5: READ v:
        POKE w, v: NEXT w
5003    DATA 243, 6, 192, 195, 175, 14
5004    POKE mf, 16: LET x = USR str: RANDOMIZE USR print:
        GO TO mm
```

System variable NXTLIN is used to find the address of the line containing the REM, which has eleven bytes in it rather than six, because the last program byte is 0Eh ("number") which prevents the listing of the

next five bytes. If the extra five bytes were not there, the BASIC listing would go awry at that point. Unfortunately, it has not been possible to prevent listing problems in a number of the other REM statements in this book.

When the print routine above has been RUN once, lines 5002 and 5003 can be deleted, leaving the relocatable machine code, the routine to work out where it is, and the routine to call it.

Telephone number reminder

To remind yourself of your local Prestel telephone numbers, you may wish to EDIT the PRINT statement in line 1460. For example, if the telephone number you use is "246 8081" the line might become:

```
1460 LET T$ =i$: GO SUB tx: CLS: PRINT AT 5, 8; FLASH 1;
     "Phone Computer"; AT 7, 11; "246 8081": LET ls = l: LET
     con = l + l: GO TO 2020
```

Where you need to remember a list of telephone numbers for different computers, you could add all the numbers to a list on the screen. While this does not give the VTX5000 a built in autodialler, it does help make entry to Prestel simpler.

Program version reminder

So that you can quickly tell which control program you have in memory at any time, you can change the main menu in your version by altering line 900, for example:

```
900 DATA "VTX5000 Main Menu", 8, 10, "Log ON or OFF", "Go
    Directly On-Line", "Save Frame", "View Frame", "Print
    Frame", "Downloader", "Mailbox Message", "Enter BASIC"
```

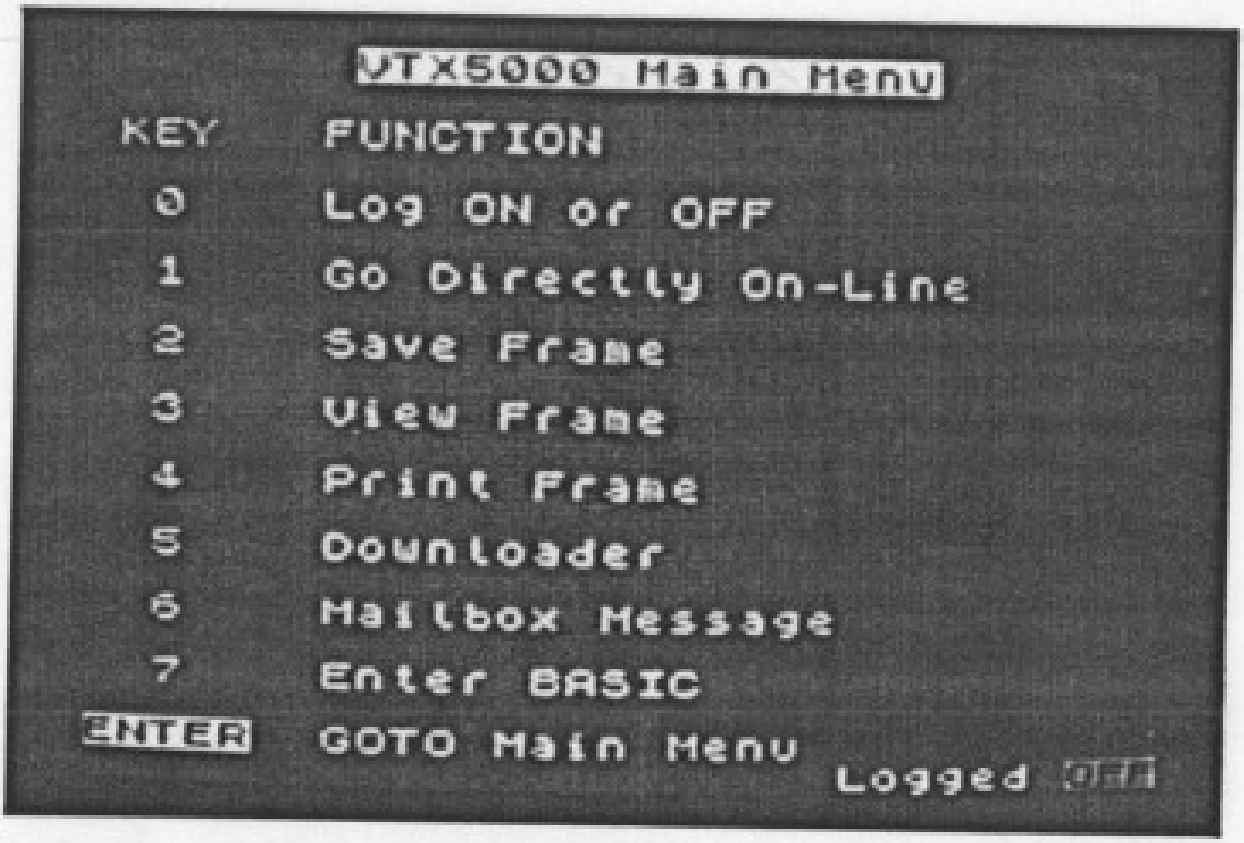
Or you could even change this and other menus to use a language other than English. This could be very useful, as Prestel technology manifests itself as a system called Bildschirmtext in Germany, Viditel in the Netherlands, and a variety of names in other countries.

Saving Frames in RAM or on microdrive

The VTX5000 manages to almost fill the memory of a 16K Spectrum, particularly if you ever need room for Interface 1 system variables and a microdrive channel buffer. However, on a 48K Spectrum there is plenty of spare memory, and we can easily write some routines to store copies of Prestel frames in RAM, to look through later without running up your telephone bill, to print, or whatever. The numbers given therefore refer to a 48K machine; on a 16K machine there is just enough room for one RAM copy of the screen or one microdrive buffer, and you may like to rewrite these routines to cope with whichever of these suits you best.

Photograph/Printout 2.1

Your Own Version of the VTX5000 "Main Menu"




```
          VTX5000 Main Menu
KEY     FUNCTION
 0      Log ON or OFF
 1      Go Directly On-Line
 2      Save Frame
 3      View Frame
 4      Print Frame
 5      Downloader
 6      Mailbox Message
 7      Enter BASIC
ENTER   GOTO Main Menu
                    Logged OFF
```

While we are moving RAMTOP, we will correct the VTX5000 error in its position, so that user defined graphics are not overwritten. This does not restore the normal graphic patterns which have been destroyed before you BREAK, but it does prevent the machine crashing when you LOAD in a set of graphics. If you are going to use UDG, note that on a 48K machine the VTX5000 has shifted the graphics to address 63425, so you will want to use something like:

```
LOAD "udg" CODE 63425
```

to LOAD your favourite set of graphics.

To allow twenty six frames to be stored in RAM we alter line 80 to:

```
80 CLEAR 38464: LET ix = 65408: LET str = 65484: LET o = 0:
   LET I = 1: DIM a$ (22): DIM b$ (600): POKE 23609, 50 LETb$
   (I TO I + I) = CHR$ o + CHR$ o: LET r$ = "a"
```

r$ is used to store the letter code of the next memory space available for use.

The changes needed to view these frames and microdrive frames are as follows:

```
4100 LET E$ = "Current Frame"
4110 POKE mf, 16: LET X = USR str: PRINT #1; AT 1, 1; PAPER
     1; INK 7; E$;: PAUSE o
4120 IF CODE INKEY$ = 7 THEN POKE ix + 9, 2 * (PEEK (ix +
     9)<> 2): GO TO 4110
4130 LET E$ = "" : GO TO mm
```

This makes the exit from viewing frames much easier than before, while retaining the reveal function on the EDIT key (CHR$ 7). To continue:

```
4200 POKE 65534, 1: GO SUB 4600: IF q$ = CHR$ 13 THEN GO
     TO 4000
4210 LET E$ = "Frame " + q$: GO TO 4110
```

RAM address 65534 is used to store a flag which indicates to a machine code routine whether a "view" or "save" is taking place. Also, as usual, ENTER can be used as an exit, hence the check on CHR$ 13.

```
4300 GO SUB 4800: PRINT AT 5, 11; "Start Tape": LOAD q$
     CODE 64448: LET E$ = "Frame " + q$: GO TO 4110
4400 GO SUB 4700: IF m$ + q$<>"m" THEN LOAD * m$; d; q$
     CODE 64448: LET E$ = "Frame "" + m$ + ";"+STR$ d +
     ";"+q$: GO TO 4110
4410 GO TO 4000
```

Line 4400 checks that you are not trying to load a microdrive file with a blank name. Now, we have the subroutines that the above lines use:

```
4600 GO SUB cl : PRINT AT 3, 1; "Memory Frame Letter?":
     PAUSE o: LET q$ = INKEY$ : IF q$ = CHR$ 13 THEN
     RETURN
4610 IF q$< "a" OR q$> "z" THEN GO TO 4600
4620 POKE 65535, CODE q$ - 96
4630 LET copy = PEEK 23637 + 256 * PEEK 23638 + 5
4631 REM 12345678901234567890123456789Ø
4632 RESTORE 4633: FOR w = copy TO copy + 29: READ v:
     POKE w, v: NEXT w
4633 DATA 33,193,247, 17, 192, 3, 237, 75, 254, 255, 121, 183,
     237, 82, 16, 252, 235, 33, 192, 251, 1, 192, 3, 183, 40, 1, 235,
     237, 176, 201
4640 RANDOMIZE USR copy: RETURN
4700 GO SUB cl : PRINT AT 3, 1; "Storage Device? (m, n OR b)":
     PAUSE o: LET m$ = INKEY$ : LET d = 1: LET q$ = CHR$ 8
     + CHR$ 143: IF m$ = "b" THEN RETURN
4710 INPUT "Drive" AND m$ <> "n"; "Station" AND m$ = "n";
     " Number?"; LINE d$: IF d$ <> "" THEN LET d = VAL d$
4720 IF m$ = "n" THEN RETURN
4730 LET m$ = "m"
4800 GO SUB cl : INPUT "Filename?"; LINE q$: RETURN
4900 DATA "View Frames Menu", 4, 1, "Current Frame", "RAM
     Frame", "Cassette Frame", "Interface 1 Frame"
```

In a similar manner we need to change the "View Frames" routine:

```
3000 LET mn = 3: GO SUB dm: GO TO mc
3100 IF r$> "z" THEN LET E$ = "Memory Full": GO TO mm
3110 LET q$ = r$: LET r$ = CHR$ (Code r$ + 1): POKE 65534, 0:
     GO SUB 4620: GO TO 3210
3200 POKE 65534, o: GO SUB 4600: IF q$ > = r$ THEN LET r$ =
     CHR$ (Code q$ + 1)
3210 LET E$ = "Saved in "+ q$: GO TO mm
3300 GO SUB 4800: IF q$ <> " " THEN SAVE q$ CODE 64448,
     960: GO TO 3210
3310 GO TO mm
3400 GO SUB 4700: IF m$ + q$ <> "m" THEN SAVE * m$; d; q$
     CODE 64448,960: LET E$ = "Saved in *"+ m$ + ";" +
     STR$ d + ";" + q$: GO TO mm
3410 GO TO mm
3900 DATA "Save Frames Menu",4, 1, "Next RAM Frame - " +
     (r$ AND r$ < = "z") + ("Full" AND r$ > "z"), "RAM Frame",
     "Cassette Frame", "Interface 1 Frame"
```

NOTE the devious use of the conditional AND within a DATA statement to make the VTX5000 menu more informative about the state of the

machine. Note also that the general "RAM frame" routine allows you to store the curent frame in any of the twenty six memory spaces, overwriting frames you no longer need, as appropriate.

The necessary re-arrangement of lines 3100 and 4400 has affected SAVEing and LOAD messages, which used the old 3100 as a subroutine for INPUTing a filename into q$, exactly as line 4800 does now. Thus, we need to EDIT lines 7300 and 7400 to read roughly as follows:

```
7300 GO SUB 4800: SAVE q$ DATA b$ (): GO TO 7000
7400 GO SUB 4800: PRINT AT 5, 11; "Start Tape": LOAD q$
     DATA b$()
```

We could, of course, change these lines to use microdrive, by calling subroutine 4700 and changing the SAVE and LOAD statements appropriately.

Double height

As indicated earlier, any frame which includes a double height command on a line hidden by another double height, causes display problems due to a design fault in the VTX5000. While we cannot change the ROM to overcome the problem, and indeed it would be quite difficult to change the ROM to cope with the full requirements of dynamic Prestel, we can include a machine code routine in the "view frames" section of the program which allows static frames to be displayed correctly.

This machine code, which could have been included in the ROM, is basically as follows, once the VTX5000 ROM is paged in:

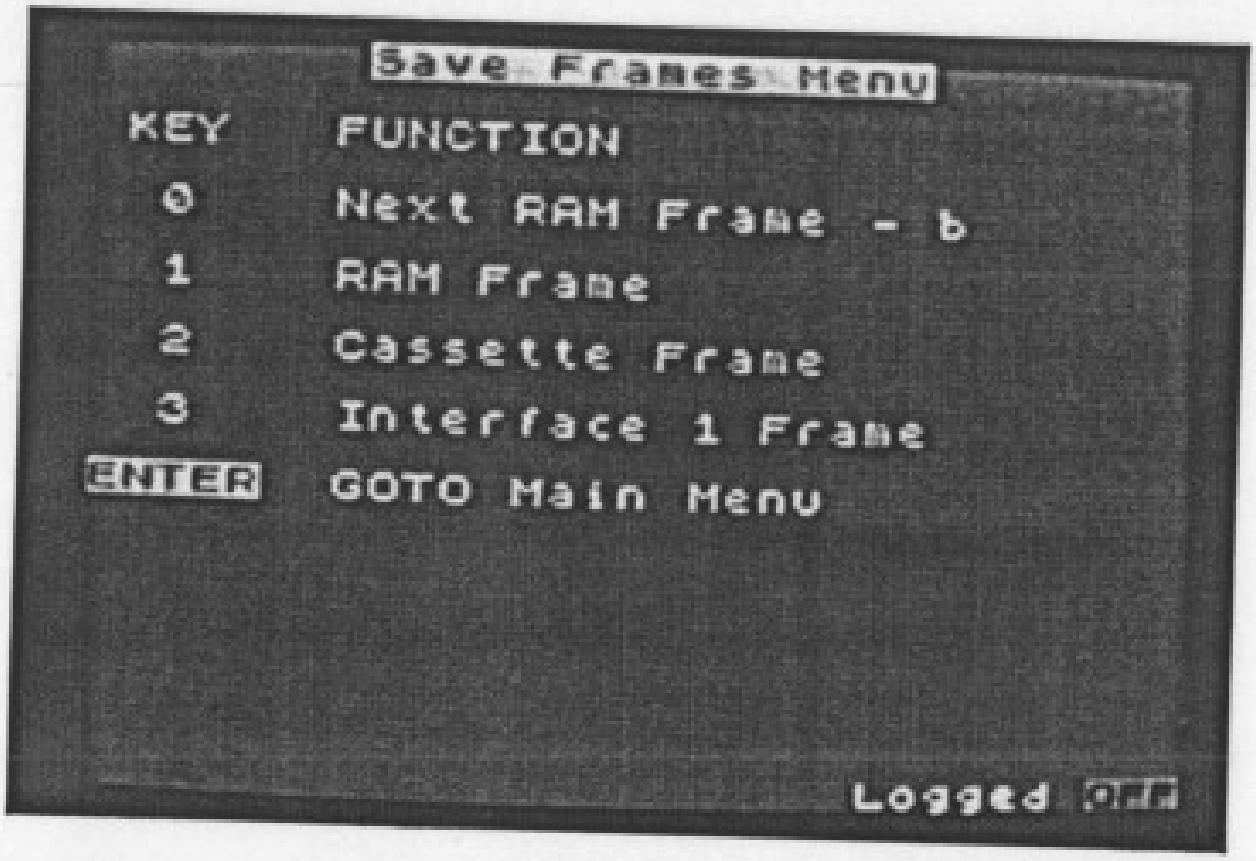
```
        LD      L,00h     : set line number to zero
LOOP:   CALL    17C5h     : display the line
        CALL    16F8h     : check for double height
        JR      Z,NH      : skip instruction if no double
                            height
        INC     L         : skip line if double height
NH:     INC     L         : move on to next line
        LD      A,17h     : check for end of screen
        CP      L
        JR      NC, LOOP  : jump back if more lines yet
        RET
```

Fitting this into a routine which can be called from BASIC gives a subroutine:

```
4140 LET disp = PEEK 23637 + 256 * PEEK 23638 +5
```

Photograph/Printout 2.2

The New "Save Frames" Menu




```
          Save Frames Menu
KEY     FUNCTION
 0      Next RAM Frame - b
 1      RAM Frame
 2      Cassette Frame
 3      Interface 1 Frame
ENTER   GOTO Main Menu


                          Logged OFF
```

```
4141 REM      12345678901234567890123456789012345678901
     12345
4142 RESTORE 4143: FOR w = disp TO disp + 44: READ v:
     POKE w, v: NEXT w
4143 217, 229, 213, 197, 243, 62, 21, 211, 255, 227, 227, 205,
     179, 19, 251, 46, 0, 205, 197, 23, 205, 248, 22, 40, 1, 44, 44,
     62, 23, 189, 48, 241, 243, 62, 53, 211, 255, 227, 227, 193,
     209, 225, 217, 251, 201
4144 PAPER o.CLS: RANDOMIZE USR disp: PAPER 1:
     RETURN
```

which may then be called in two appropriate places, namely:

```
4110 GO SUB 4140: PRINT #1; AT 1,1: PAPER 1; INK 7; E$;:
     PAUSE o
```

and

```
5004 GO SUB 4140: RANDOMIZE USR print: GO TO mm
```

Again, it is worth remembering that once this subroutine has been RUN, lines 4142 and 4143 can be deleted, thus saving memory and considerably speeding up subsequent execution of the subroutine.

Chapter 3

Downloading Telesoftware from MICRONET

The concept of telesoftware has quite a long history, not quite as long as Prestel itself but not far off. In the very early days of Prestel, quite a few Information Providers experimented with the idea of putting programs on to Prestel pages. Prestel themselves had a few pages up in the Babbage language used on their computers, CAP (Computer Analysts and Programmers) made a serious commercial attempt to distribute business software in their Micro Cobol intermediate code, GEC had a number of programs up in BASIC, and various other Information Providers experimented with Fortran and other languages.

The format which is most commonly used today was devised by CET (the Council for Educational Technology). They used Research Machines 380Z computers for the most part but the format they devised is merely a structure into which any data can be fitted, whether it is a program in BASIC or another language or whatever. The original concept was that programs could be written in some form of portable BASIC which could be loaded by a variety of machines, but in practice all telesoftware on Prestel is machine specific.

Viewfax became the first Information Provider on Prestel with a database aimed solely at the microcomputer user, particularly those with BBC micros.

Then Micronet came on the scene, aiming to make their database such a cheap and attractive source of telesoftware that a large number of microcomputer users would join Micronet, and the resulting subscription income would allow Micronet to recover their large initial investment.

How it works

A program is coded on to a number of Prestel frames, originally in free text, with a few surrounding markers consisting of character code 7Ch (which appears as '▮▮') and another character. One of these markers introduces a checksum, which allows the microcomputer to determine whether line noise has corrupted receipt of the frame.

The procedure for LOADing a telesoftware program is roughly as follows.

You move through Prestel frames in the normal manner described in Chapter 2 until you come to a header frame, which includes a single line of telesoftware at the bottom. You tell your microcomputer to LOAD the program. It then checks the checksum on the header, re-requesting the frame using *00 (which costs nothing) as necessary, and when that has been received correctly it uses # to step on to the next frame. That frame is similarly checked, then copied into program RAM and # used to call the next frame and so on; except that 0 is used to call the frame after a 'z' frame, and the microcomputer will abandon downloading if it has to re-request a frame too many times. On many microcomputers, the Spectrum/VTX5000 combination included, you have to indicate the frame letter of the header frame (usually 'c') before you start downloading, so that the computer knows when to use 0 to select the next frame.

Once the program has been downloaded, the microcomputer can either leave Prestel and RUN the program independently, or can stay on-line and RUN the program, which can then access other Prestel pages. It is up to the program to offer you the chance to SAVE itself.

Programs are charged for, using normal Prestel frame prices. The cost is added to your Prestel bill and credited to the Information Provider's account, less a Prestel handling charge. Prestel frames are currently limited to a price of 99 pence, except for 'a' frames which are limited to 50 pence. If line noise or a Prestel computer failure, or whatever, causes you to fail to download a program, you have still paid for the frames that you have seen. So do be careful when downloading expensive programs — do not start if the Prestel computer ever warns you of impending closure, or if the line noise is so bad that most frames appear to be corrupted on receipt. The best telephone line quality is generally obtained on local calls during cheap rate periods, when nearby lines in the telephone exchange are quietest. If your normal telephone calls suffer from excessive crackling, or are always very quiet, you cannot expect the VTX5000 to 'hear' all the Prestel whistles correctly, and you should call in a telephone engineer.

Some of the other telesoftware formats, such as that developed by GEC, allowed the microcomputer to recover from very bad line noise by redialling Prestel in the hope of getting a quieter telephone line. This was achieved by using only 'a' frames for programs, and recording the page number of the current and next pages within the checksummed part of the frame. The microcomputer then used * number # commands to step through the frames of the program. This removed the 'z' frame inconsistency, and allowed for CATalogues of named programs. The downloading technique caused the microcomputer to go to a pre-determined CATalogue page and search for the name of the program to be downloaded, which then gave it the number of the first page in the program. There is a suggestion that a future version of the CET format may be expanded to include some of these ideas.

The format and its limitations

The Spectrum/VTX5000 implementation of the CET format imposes a couple of limitations not found on some other microcomputers, or not in the equivalent normal Spectrum cassette LOAD instruction.

Firstly, keywords cannot be written out in English on the telesoftware frames — they have to be put there as the appropriate 8-bit Spectrum token. However, since the Prestel character code is a 7-bit code, special CET format markers are involved to indicate an offset to be added to a 7-bit character. Thus, rather like the multiple shift keys required to type a Spectrum keyword, there are a number of cryptic characters needed on a telesoftware screen to represent each keyword.

Secondly, the telesoftware is loaded into the Spectrum in the memory range indicated by system variables PROG and VARS, thus no variables or DATA or CODE files can be loaded. This can also be a problem if you have Interface 1 connected, as the extra system variables and possible channel buffers which this invokes can shift PROG up memory. This will either cause embedded machine code programs to become uselessly misplaced, or cause the Spectrum to run out of memory, as VARS gets pushed up to or beyond RAMTOP (despite the telesoftware downloader moving RAMTOP to just 1000 below "ix", 1127 below the end of memory — which is needed because the 960 character frame buffer is used while checking the checksum and decoding the CET format). Thus, the header frame of a number of such telesoftware programs includes an instruction to you to clear out Interface 1 variables before downloading.

NOTE the words 'St. Andrew' seven lines up from the bottom, in Printout 3.1 on the following page. They are actually part of a 'REM' statement, and are the only legible part of the program after keywords have been coded as 8-bit tokens.

Printout 3.1
A Typical Telesoftware Frame

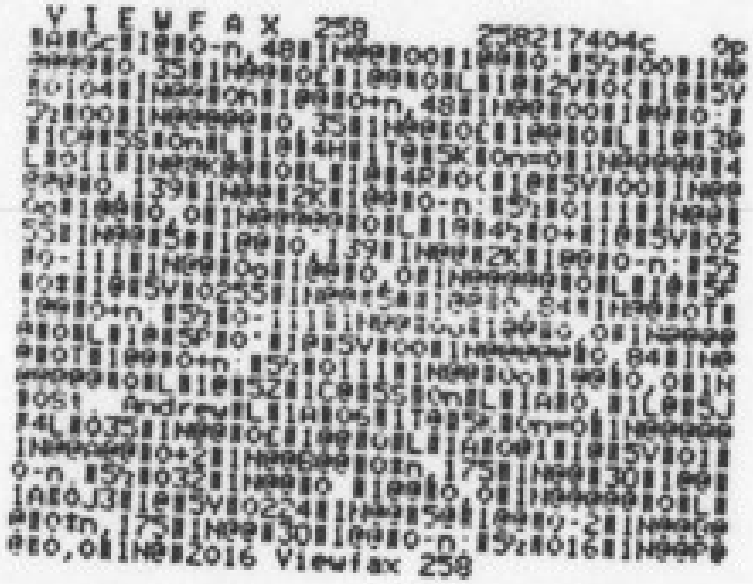
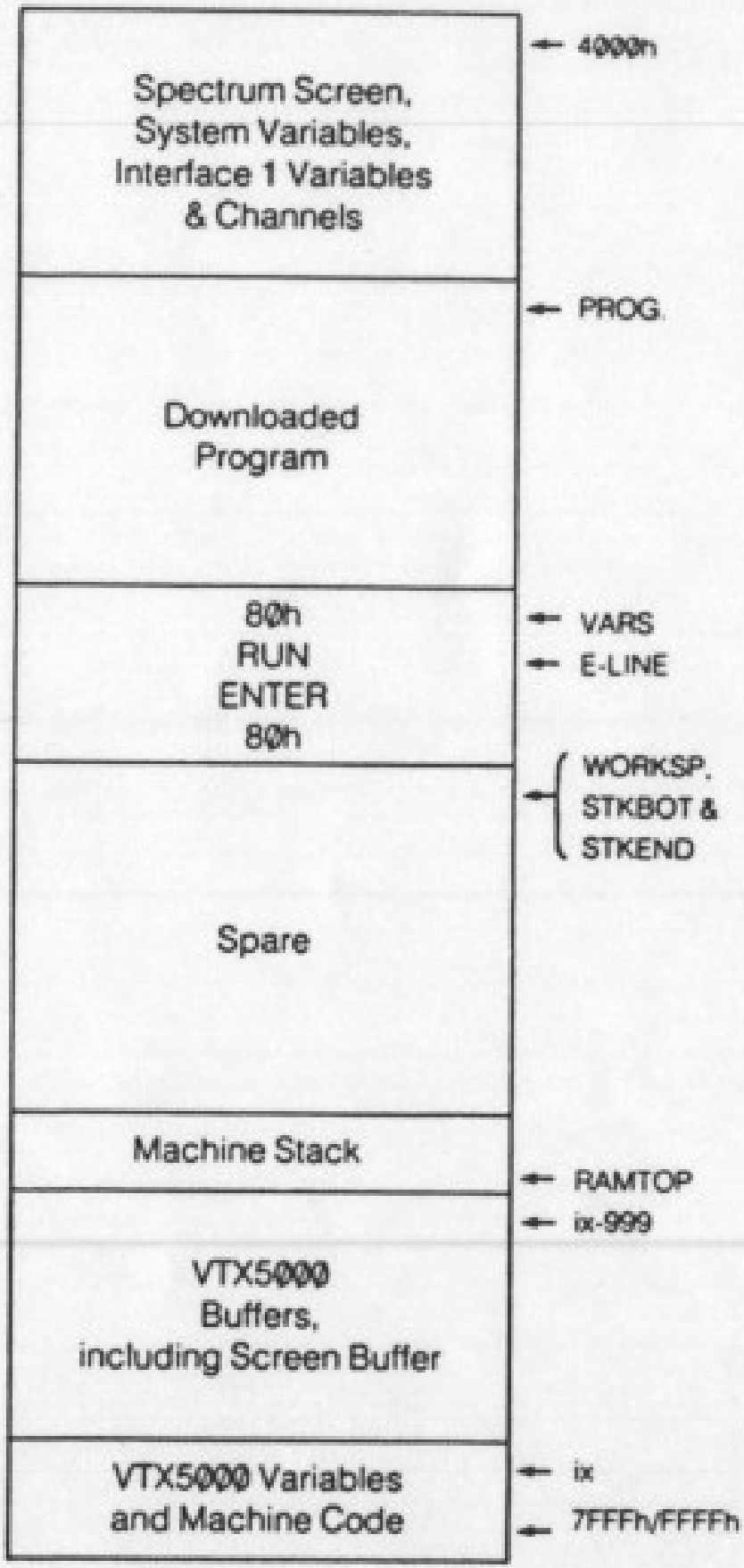


Diagram 3.1
The 'II' Marker Codes

II A	Start of telesoftware 'block'
II E	The code for ' II ' itself (7Ch)
II F	End of telesoftware 'file'
II G	Frame letter follows (not checked by VTX5000)
II I	End of 'non-standard' marker section
II L	End of line, equivalent to 'ENTER' (0Dh)
II Z	End of 'block', followed by 3 digit checksum
II 0	Store received characters unchanged in RAM
II 1	Subtract 40h from the following characters
II 2	Add 40h to the following
II 3	Add 60h to the following
II 4	Add 80h to the following
II 5	Add A0h to the following
¾	Equivalent to 'SPACE' (20h)
II ¾	The code for '¾' (7Dh)

Diagram 3.2

Memory Map After Downloading Telesoftware



At this point, it is well worth discussing the different effects of RANDOMIZE USR o, NEW and LOAD.

RANDOMIZE USR o is used by option 7 on the VTX5000 menu to remove all trace of the VTX5000 from memory. It invokes a complete Spectrum reset, clears out all memory, sets system variable P-RAMT to the highest physical memory location, and restores the user defined graphics characters to the shapes of the letters A to U. No RST 08h instruction is performed, so no Interface 1 variables are created.

NEW clears out all Spectrum memory up to and including RAMTOP. This usually leaves the user defined graphics characters alone and, of course, leaves all the VTX5000 variables and buffers, which are above P-RAMT; but it does clear out any Interface 1 variables.

LOAD, or the act of downloading a telesoftware program, only affects the memory between system variables PROG and RAMTOP. It clears out BASIC program variables, but does not remove Interface 1 variables. Indeed, LOAD * will add Interface 1 variables if they are not already there.

Thus, to remove Interface 1 variables but keep VTX5000 variables, you need to do a NEW rather than any of the other options. It is always possible that future telesoftware programs may depend on the presence of Interface 1 variables, and we can always add them without causing an error by using CLOSE # 0 or a similar ineffective command (if you do not have an Interface 1 this does not, of course, add any variables).

So we can alter the BASIC control program in the VTX5000 as follows:

```
6400 IF PEEK 23631 + 256 * PEEK 23632 = 23734
     THEN CLOSE # o: GO TO 6000
6410 GO SUB c!: PRINT AT 3,1; "Press Any Key Then:" ' '
     "RANDOMIZE USR "; ix+99: PAUSE o: NEW
6900 DATA "Downloader Menu", 4, 1, "Load, Exit & RUN", "Load,
     Stay & RUN", "Change Start Frame — ' " +s$+ "'",
     ("Remove "AND PEEK 23631 + 256*PEEK 23632 <>
     23734) +"Interface 1 Vars"
```

This gives an extra option on the Downloader Menu to either add or remove Interface 1 variables as appropriate. When removing the variables, you have to key in the appropriate RANDOMIZE USR command. Be very careful when typing this, as even the slightest syntax error will cause the Interface 1 variables to be recreated. If you do get the flashing syntax error marker, you must use NEW before going back to try RANDOMIZE USR.

The RANDOMIZE USR does, of course, load the ROM version of the BASIC control program into memory but as you are going to immediately use it to download a telesoftware program in its place, this does not really matter.

Who provides the programs on Prestel?

While most of the telesoftware programs on Prestel now appear on the pages of a small number of Information Providers, it is possible for anyone to go to these IPs and offer a program for sale through the system in return for a royalty on the price charged. Many software houses have done this, and their software is available on Prestel more cheaply than in the shops because they do not have any tape or packaging costs. It is even possible to sample some such programs very cheaply, because they prevent you SAVEing them, before paying out a more usual price for a version you can SAVE.

Some telesoftware does appear on the pages of unexpected Information Providers. Meditel, for instance, have some programs to help doctors and, if you wanted to give your software away free, or the main IPs decided your program was not commercially viable, you could always pay an Information Provider to put the program up, or even become an Information Provider yourself, buying blank pages direct from Prestel.

How to write programs suitable for the format

Any program that you write to RUN immediately it is downloaded must take into account the memory restrictions imposed by the presence of VTX5000 variables and buffers, and the possibility of the presence of Interface 1 variables. In addition, as we have already discovered, system variables P-RAMT, RAMTOP and UDG are oddly located. Any program variables or machine code that the program needs must be coded into DATA or REM statements, and this may make the program take up much more memory space than you would normally find.

All these restrictions can be overcome by the use of two basic tricks.

Firstly, the program can SAVE to cassette a sequence of files which when reLOADed reconstitute the desired program rather than the downloaded program. In extreme cases, this could involve the downloaded program containing a machine code cassette output routine which could send anything it wanted to the MIC output. More likely, the downloaded program would SAVE parts of itself to form a BASIC loader, a SCREEN$ file and a machine code file, on tape.

Secondly, where the program needs more memory than that available between PROG and RAMTOP, it could be split into parts and downloaded using "Load, Stay & RUN". Then the first part would SAVE some of itself, then "page in" the VTX5000 ROM and initiate downloading of a second part, and so on. At the end, the program Exits Prestel and reminds you how to LOAD the tape program before self destructing with a RANDOMIZE USR 0.

If you go to an Information Provider with a program to be published as telesoftware, they will usually have suitable "uploading" software available which will build the various cassette or microdrive files you use into a suitable reconstituting program. However, if your program is particularly large, they may have problems and may need to split some of the CODE or program files into a number of sections. Uploading techniques are continually advancing to cope with a wider and wider range of software.

Chapter 4

Sending messages

How often do you telephone someone only to find that they are out, or only to be answered by a machine which records a mushy and almost incomprehensible message from you? Prestel can solve your problems. It records visual messages which await the recipient's next call to the computer, and you have the confidence that what you see on your screen will be what the recipient sees on his screen — apart from occasional line noise.

Types of message page

There are three basic types of message page on Prestel: the response frame which carries a message to the Information Provider who created it, the mailbox frame which can carry a message to any Prestel user, and the data collection frame which you find on external computers when you pass through a gateway.

All these frame types are very similar, the only difference being that on a mailbox frame you have to type the recipient's Prestel account number, which may be rejected by the computer if it does not belong to a user, or if the line to the Pandora computer is not working. On data collection frames every field, or section of the page that you complete, can have a different matching instruction on the next to bottom line of the screen.

The contents of message pages are partly fixed, partly filled by the computer from the details it holds about sender and recipient, and usually partly free to be completed by the sender. The fields which have to be completed by the sender usually need to be terminated by '#'. After completing the last field, you are asked to confirm whether you want to "send" the message or not. If you do not "send" the message, the recipient will never know that you have been to that page.

The sending of a message is extremely fast — it takes less than a second for it to travel down the line to Pandora and, after it has arrived there, the recipient will be advised of its existence whenever he calls Prestel or if he tries to leave Prestel via page 90 (*90#).

Methods of sending messages

On the VTX5000, messages can be sent "on-line" or prepared "off-line", using the facilities listed on the "Mailbox Message Menu".

The standard VTX5000 off-line message preparation routine has one or two oddities. The routine takes in lines of up to 39 characters, but displays them in the standard 32 character per line Spectrum character set. If an input line does not end with a Spectrum '#', the VTX5000 adds 0Dh and enough 0Ah's to move the cursor to the next line. (On Prestel message pages 0Dh takes the cursor back to the start of the current field and 0Ah moves the cursor down a line as long as it stays within the field.) This approach is acceptable so long as there is room within the field; if not, overflow lines will overwrite the last line sent.

More confusion can arise where the space available does not start at the beginning of a Prestel line, causing input lines to spread across the end of one line on to the start of the next, possibly splitting a word across the line boundary in an unsightly manner. The privileged space at the start of fields also causes the first line of the message to be offset one space to the right compared with subsequent lines. (When you type a message on-line you can see the odd behaviour of the cursor at the start of fields, associated with the "privileged space".)

You can mix single line fields terminated by '#', and one large multi-line field within a prepared message, but if you try to include two or more multi-line fields, too many 0Ah's will be included after the first line of the second multi-line field and at the end of subsequent lines not terminated by '#', as the BASIC control program has not been written to cover this case.

Once you have prepared your message, you can optionally SAVE it, reLOAD another one, display the current message on screen, and eventually "log on" to Prestel and send the message.

It is easy to include symbols in prepared messages by the normal use of the SYMBOL SHIFT, or even extended mode, though the final appearance of the symbol on Prestel will be affected by the difference in character sets given in Appendix I. When on-line, the VTX5000 interprets SYMBOL SHIFT as *, causing further problems.

Problems with £ and other symbols

When on-line you can type various symbols as part of your message. You need to hold down CAPS SHIFT and press SYMBOL SHIFT to get the Spectrum into an almost normal mode, where you hold down SYMBOL SHIFT and press the key marked with the symbol you want. The code sent to Prestel is the code of the single character in red on the key.

Three keys send the Prestel # code 5Fh, ENTER is translated by the VTX5000 to 5Fh, "underscore" is the Spectrum character with code 5Fh, and this passes through to Prestel without being converted to anything else, and the Spectrum #, code 23h, is converted to 5Fh too.

This occurs because different 'national use options' have been invoked in the Spectrum and Prestel interpretations of the 'standard' ASCII (American Standard Code for Information Interchange).

As mentioned in Chapter 2, the odd position of '#' in the Prestel character set was due to its late addition by request from British Telecom (then part of the Post Office) to the Teletext character set. The Teletext character set was devised by a BBC/IBA/BREMA joint committee, which decided that they needed such things as the fractions ¼, ½ and ¾, and that they needed both the '£' and '$' signs.

Subsequently, Sinclair also decided that all three signs '£', '$' and '#' were needed in the Spectrum character set, but it was the '£' sign that ended up in an odd place in the code table.

The Prestel '£' is code 23h, and we have already noted that the equivalent Spectrum character '#' is converted to 5Fh. This makes it impossible to type '£' in a message, unless you have one of the later VTX5000s issued by Homelink, where the VTX5000 ROM has been altered to convert the Spectrum '£' code 60h to code 23h, the Prestel '£'.

There is a way to check which version of the ROM you have, and this technique is also generally useful for testing what is transmitted by any message-sending software you write. Switch the VTX5000 mode switch to the middle 'Tx' position, but leave the line switch in its upper position (so that you do not engage your telephone line even if it is still connected). Follow option 1 on the VTX5000 main menu to "go directly on-line", and you will see the current frame, but anything you type now appears on-screen in its converted form. SYMBOL SHIFT displays "*" as it does when on-line, so type CAPS SHIFT/SYMBOL SHIFT and SYMBOL SHIFT/£ to see which ROM version you have. Usually you will see the Prestel hyphen, code 60h, but on the modified version you will see a Prestel '£'.

If your VTX5000 is not one of these later models, you can still send '£' if you prepare a message off-line, but not by typing '£'.

The theory goes as follows. The 8251 chip makes bit 7 of every code sent to Prestel a parity check bit. Bit 7 is set to 0 or 1 to make the number of ones in the byte an even number. The Prestel computer can thus spot and reject input corrupted by noise on the telephone line (unless two or more bits are corrupted). Now, if you add 128 to the code for a Prestel '£', the code will still actually reach Prestel as a '£', but the VTX5000 routine which sends the code to Prestel will not regard it as a Spectrum '#' and will not convert it to another character. The character we need to type is:

23h + 80h = A3h

which is Spectrum user defined graphic 'T'.

To help you to remember that this is a '£' sign when sent down the line to Prestel, you can add code to set up the user defined graphic character, thus:

```
7001 FOR j=0 TO 7: POKE USR "t"+j, PEEK (16128+j): NEXT j
```

Luckily, graphic 'T' is outside that part of the user defined graphics affected by the RAMTOP positioning error, and this POKEing does not crash the machine.

Other Spectrum graphic codes can also be useful when preparing devious messages off-line. Appendix I gives a full comparison of all Spectrum and Prestel codes.

If you use '£' a lot, you could prepare a message consisting of just a graphics 'T'. Then, to send '£', all you need to type is CAPS SHIFT/ ENTER followed by '6' and then '0'; but this has the disadvantage of also sending codes 0Dh and 0Ah, leaving you to move the cursor back to the correct position. So it is better, particularly if you also want to be able to prepare messages, to alter line 7900 to:

```
7900 DATA "Mailbox Message Menu", 6,1,"Send Message",
     "Prepare Message","SaveMessage", "Fetch Message",
     "Display Message","Send '£'"
```

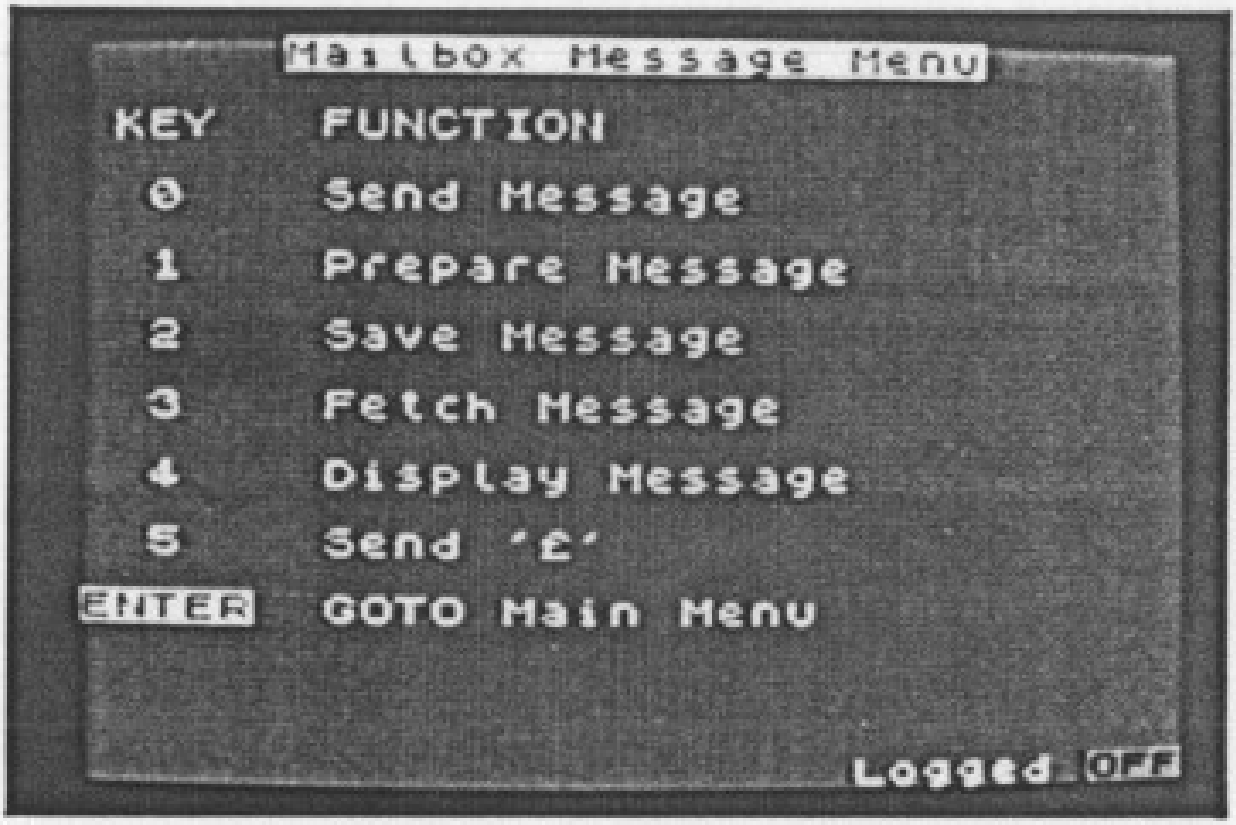
and add:

```
7600 POKE t, l: POKE t+1,0: POKE t+l+l, 163: GOTO 2000
```

Then CAPS SHIFT/ENTER followed by '6' and '5' will send a '£' sign while on-line.

Photograph/Printout 4.1

The New "Mailbox Message" Menu




```
        Mailbox Message Menu
  KEY    FUNCTION
   0     Send Message
   1     Prepare Message
   2     Save Message
   3     Fetch Message
   4     Display Message
   5     Send '£'
 ENTER   GOTO Main Menu


                        Logged OFF
```

The on-line keyboard input routine in the VTX5000 ROM converts ENTER to '#', SYMBOL SHIFT to '*' and DELETE to 'cursor left', while also keeping track of CAPS LOCK and the "reveal" function, which is switched by the EDIT key. The routine also takes appropriate note of CAPS SHIFT/SYMBOL SHIFT and CAPS SHIFT/ENTER. It otherwise interprets the keys according to the system variables MODE and FLAGS and, on power up, the Spectrum is in 'K' mode until INPUT or BREAK and LOAD, which can cause confusion. You can always send any codes you want by BREAKing out of the control program and manually typing:

```
LET t$ = CHR$ the code you want to send
GO SUB tx
GO TO 2000
```

In this routine the VTX5000 only changes the code for '£', and we have solved this problem above by using code 163. Code 12 is useful if you are using 'Tx' mode to test the screen display, as it clears the screen.

You can put a longer string into t$, but the maximum length string that subroutine 'tx' can cope with is 255 characters.

How to prepare messages to suit specific mailbox pages

The BASIC control program's message preparation routine at line 7200 puts your message into the string b$ (3 TO) and makes CODE b$ (1) +256 * CODE b$ (2) equal to the number of characters to be sent. The message sending routine at line 7100 copies b$ into the VTX5000 message buffer at address 't'. Both b$ and the message buffer are 600 characters long, so you can prepare messages of up to 598 characters. With this information it is possible to write your own version of the message preparation routine to suit the particular layout of the message page you want to use.

It may be that you are a company representative working from home and your employers are either Prestel Information Providers or have their own private viewdata system, and they would like you to submit reports, orders etc. on specific message pages. In extreme cases you could write a program to completely replace the BASIC control program; this new program could question you about your day or week, calculate statistics, expenses claims etc., then ask you to call Prestel and send a series of appropriate messages. If you are designing such a program, it is advisable to allow yourself to manually "key 1 to send", so that you can check that the messages have not been corrupted by line noise — after keying CAPS SHIFT/ENTER to return to the program, it can then ask you if that message was successful, and either move on to the next message or repeat the failed message as appropriate.

As a normal Prestel user sending messages to other Prestel users, you may find yourself using the general mailbox frame on page 77. This frame has two fields, a nine character field to take the recipient's account number, and a field which is 13 lines long, apart from the first two characters on the first line and the last character on the last line. The following routine uses the 40 column display routine from Chapter 1 to allow you to see the message you are preparing in a form as close as possible to that which will be sent if you use page 77:

```
7200 BORDER 1: GO SUB 9900: POKE ix+13,15: PRINT "Enter
     Text": POKE ix+12,3: PRINT "To:..... ": POKE ix+12,6:
     PRINT ">": POKE ix+12,19: PRINT CHR$ 8; CHR$ 95:
     POKE ix+12,3: POKE ix+13,3: LET i=3
7205 PRINT CHR$ 127; CHR$ 8;: PAUSE 0: LET q$=INKEY$: IF
     q$ >= "0" AND q$ <= "9" THEN GO TO 7220
7210 IF q$ = CHR$ 8 OR q$= CHR$ 12 THEN IF i>3 THEN LET
     i=i-1: PRINT ".."; CHR$ 8; CHR$ 8;
7215 GOTO 7205
7220 PRINT q$;: LET b$(i)=q$: LET i=i+1: IF i<12 THEN GO
     TO 7205
7225 LET b$(i)= CHR$ 95: LET i=i+1: POKE ix+12,6: POKE
     ix+13,2
7230 PRINT CHR$ 127; CHR$ 8;: PAUSE 0: LET q$=INKEY$: IF
     q$<" " THEN GO TO 7260
7235 IF q$="#" OR i>529 THEN LET q$=CHR$ 95
7240 IF q$="£" THEN LET q$=CHR$ 163
7245 IF q$="*" THEN GO TO 7200
7250 LET b$(i)=q$: LET i=i+1: IF q$=CHR$ 95 THEN LET
     i=i-3: LET b$(1)=CHR$ (i-256*INT(i/256)): LET b$
     (2)=CHR$ INT (i/256): BORDER 0: GO TO 7010
7255 PRINT q$;: GO TO 7230
7260 IF q$=CHR$ 8 OR q$= CHR$ 12 THEN IF i>13 THEN LET
     i=i-1: PRINT " "; CHR$ 8; CHR$ 8;: GO TO 7230
7265 IF q$=CHR$ 13 THEN LET q$=CHR$ 95: GO TO 7250
7270 GO TO 7230
```

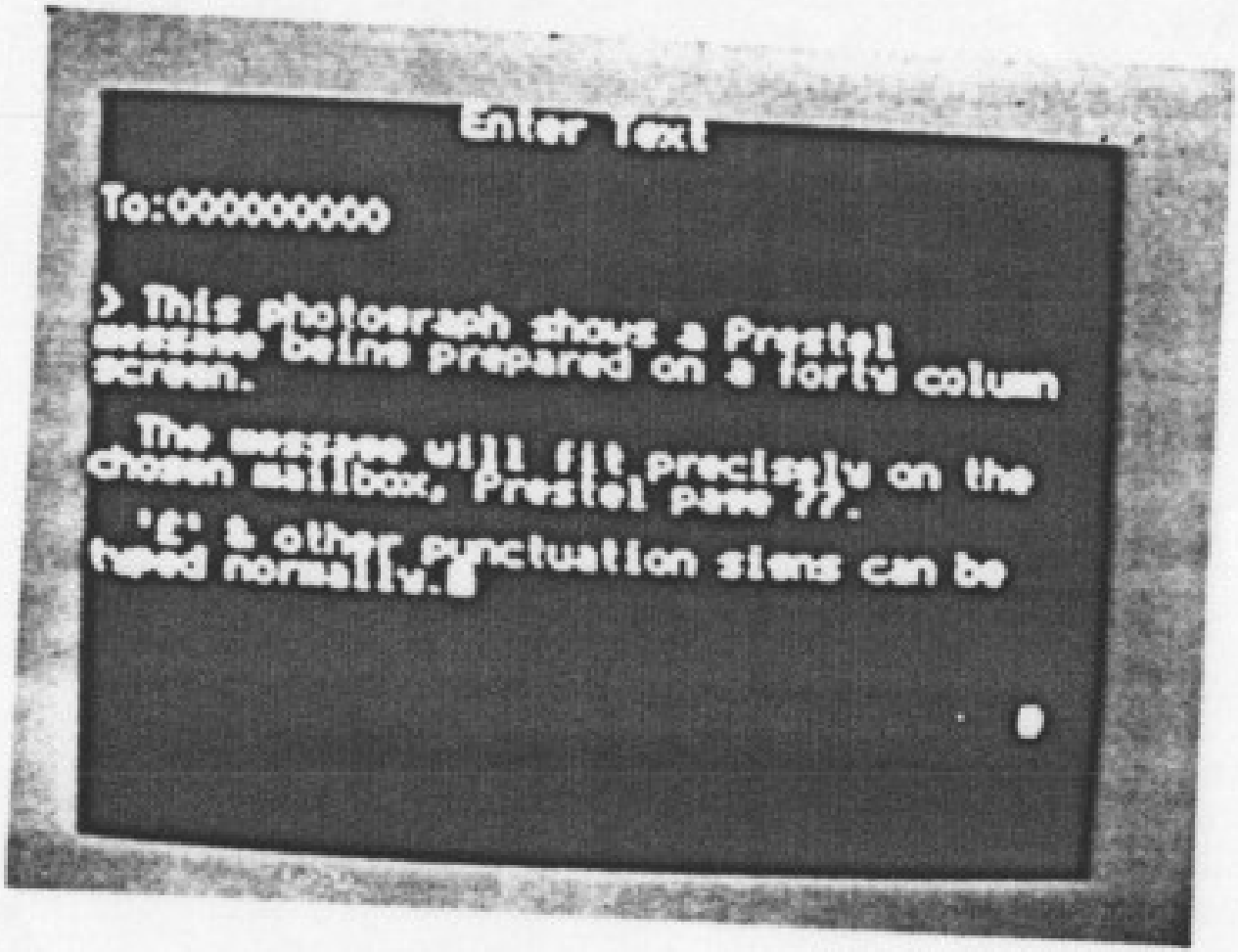
NOTE the use of POKE to mimic AT on the forty column screen, and the use of CHR$ 127 as a cursor, because the VTX5000 ROM will only maintain its flashing cursor so long as its machine code is in control. Either "backspace" or DELETE are allowed to erase any mis-typed characters.

Similar routines could be written for other formats of message page.

We could re-write the message display routine at line 7500 to also use a forty column screen, but I find it quite interesting to be able to see all messages in the common format used by the existing ROM routine.

Photograph/Printout 4.2

Forty Column Message Preparation



Enter Text

To:00000000

> This printout shows a Prestel message being prepared on a forty column screen.

The message will fit precisely on the chosen mailbox, Prestel page 77.

'£' & other punctuation signs can be typed normally.

Preparing multipage messages off-line

The space on a single message page is very limited and you will soon find yourself writing messages longer than the space available. There is no special multi-page software on Prestel, so it is conventional to end the first page of a message with a few dots ("..."), send that page and then use *00 or *09 to continue on the same message page, starting the text with a few more dots.

Using *00 allows you to # past the mailbox account number or any other part of the original message you want to leave, and type over those bits you want to change — keying # in the middle of a field wipes out the rest of the field.

If you have a microdrive and have modified lines 7300 and 7400 in the BASIC control program, as suggested earlier, you can very quickly pick up and send the separate parts of a multi-page message. Otherwise, you can resort to copying messages in RAM as we did with frames; but, there is a difference, as the messages are stored in the BASIC string array b$, so we can easily use BASIC to do the copying without writing a special machine code routine.

For example:

```
  85 DIM c$ (5,600): FOR i=1 TO 5: LET c$ (i,1 TO 2)=
     CHR$0+CHR$0: NEXT i
7300 GO SUB 7350: IF q$="c" THEN GO SUB 4800: SAVE q$
     DATA b$(): GO TO 7000
7310 IF q$>"0" AND q$<"6" THEN LET c$ (VAL q$)=b$: GO
     TO 7000
7320 IF q$=CHR$ 13 THEN GO TO 7000
7330 GO TO 7300
7350 GO SUB cl: PRINT AT 3,1; "Select RAM store number (1 TO
     5) " ' ' "OR 'c' for cassette": PAUSE o: LET q$=INKEY$:
     RETURN
7400 GO SUB 7350: IF q$="c" THEN GO SUB 4800: LOAD q$
     DATA b$(): GO TO 7500
7410 IF q$>"0" AND q$<"6" THEN LET b$=c$ (VAL q$): GO
     TO 7500
7420 IF q$ = CHR$ 13 THEN GO TO 7000
7430 GO TO 7400
```

Again, this uses too much memory to fit in a 16K Spectrum, and with 5 message stores and 26 frame stores, even a 48K machine is getting quite full. We can obviously trade off frame stores against message stores, or with more delicate use of machine code they could dynamically share the same memory space, but I find that this split copes with most of my needs.

Sending the same message to a long mailing list

If you are a microcomputer club secretary, or you are in business, you may often find the need to send the same mailbox message to a number of people. You can use the normal message preparation routines to construct and send the first message, but is there an easy way to send this same message to a list of other people?

The answer is 'yes'. After sending the first message, rather than keying "# to continue" as instructed on screen, you need to type *00 which redisplays the same page with the same message and moves the cursor to the first field. This field is usually for the recipient's account number which you need to key. Then you need to key '#' and as many other '#'s as necessary to reach the bottom of the frame. Luckily there is a two code sequence which the Prestel computer will accept in place of this unknown number of '#'s, namely ESC (code 1Bh) followed by J (code 4Ah).

Suppose d$ contains the list of recipient's mailbox account numbers, and b$ contains the message to be sent, except that the mailbox number at its start is just a random number you put into the message preparation routine to let you move on to the message section. Then, all you need to do is call up Prestel, go to the required message page and call up the following "send" routine:

```
7100 CLS: PRINT AT 5,1; "Please Wait": GO SUB 7700: FOR j=1
     TO 2+i: POKE t+j-1, CODE b$(j): NEXT j: FOR j=1 TO 9:
     POKE t+j+1, CODE d$(j): NEXT j
7110 POKE mf,o: LET X=USR str: IF X <> 1 THEN GO TO 2050
7120 LET d$=d$ (10 TO): IF LEN d$<9 THEN LET E$ =
     "Finished": GO TO mm
7130 LET t$=""*00"+d$ (TO 9) + CHR$ 27 +"J": GO SUB tx: GO
     TO 7110
```

NOTE that the routine progressively destroys d$, in order that after any "Line Break" or other problem, it can recover and continue sending the message. You have to visually verify the accuracy of each message, in case any of the mailbox numbers were incorrect. Key 1 to send the message, and then CAPS SHIFT/ENTER to call up the next number. If the message does get hopelessly scrambled, you should lift the line switch to cause a "Line Break", rather than keying CAPS SHIFT/ ENTER, which will only succeed in calling up the next number.

The remaining question is how the numbers got into d$ in the first place. If d$ is a simple list of your friends, it can be SAVEd and LOADed direct from cassette or microdrive using an intermediate string array t$.

For example:

```
DIM f$(LEN d$): LET f$=d$: SAVE "maillist" DATA f$ ()
```

and

```
LOAD "maillist" DATA f$(): LET d$=f$
```

The list can be extended whenever you find a new friend by:

```
LET d$=d$+ "new friend's number"
```

and so on, all directly from BASIC, though you may wish to occasionally check that LEN d$ is a multiple of 9. NOTE: do not be fooled into trying to SAVE d$ directly, because undimensioned strings are not SAVEd properly. Equally, d$ needs to be undimensioned to be used, as it is in line 7120, without problems.

An alternative way of constructing d$ is from directory pages on screen on Prestel. The standard mailbox directory pages on Prestel list up to nineteen names, with their matching mailbox numbers in the last 9 places of the 3rd to 21st lines on the screen. If you have a directory frame as the current frame, a routine of the type:

```
8100 LET p=ix-960+3*40-9
8110 IF PEEK p< CODE"0" OR PEEK p> CODE"9" THEN
     RETURN
8120 LET g$="": FOR i=0 TO 8:
     LET g$=g$+CHR$ PEEK (p+i): NEXT i
8130 LET d$=d$+g$: LET p=p+40: GO TO 8110
```

will extract all the mailbox numbers and add them to d$. This can be built into another CAPS SHIFT/ENTER cycle, where you move between directory frames and tell the Spectrum that the next frame is on screen by keying CAPS SHIFT/ENTER thus:

```
8200 POKE mf,0: LET X=USR str: IF X <> 1 THEN GO TO 2050
8210 GO SUB 8100: GO TO 8200
```

which can be called up from BASIC by a direct:

```
LET d$="": GO TO 8200
```

or whatever, as appropriate. The routine is terminated by causing a "line break".

A really long mailing list is best stored as a microdrive data file. Line 8130 is changed to write numbers to the file, line 7120 to read the numbers back, and appropriate OPEN# and CLOSE# commands are dotted through the program.

Chapter 5

Communicating with other VTX5000s

The VTX5000 mode switch positions 'Tx' and 'Rx' allow two micro-computers, not necessarily both Spectrums, to communicate over the telephone network using the common Prestel standard 1200 bit per second tones.

The system can easily be used to send visual messages without writing any additional software, and in this form is particularly useful for the deaf or hard of hearing. Later sections of this chapter go into detail about the sending of computer data, such as programs, between two machines, but first we will examine the simple transfer of screens of text.

The procedure to transfer screens of text

A VTX5000 owner can telephone a friend with a similar machine. They verbally agree who is to transmit and who is to receive. The sender switches his mode switch to 'Tx' and the recipient sets his to 'Rx'. The sender switches on his line switch and replaces his telephone receiver. The recipient waits to hear the sender's tone, then switches his line switch and replaces his telephone receiver. Both people then select option 1 from the VTX5000 main menu, to "go directly on-line". The sender should wait a few seconds to ensure that the recipient has had time to do this. Then, anything the sender types will appear on the recipient's screen and also on the sender's screen. As we saw in Chapter 4, this is a hardware echo, which in no way confirms that the typing is being received correctly — the sender is unaware of any line noise or other problems that the recipient might encounter.

It is best for the sender to prepare his message off line, for two reasons. Firstly, he can start his message with a code to clear the recipient's screen. (Spectrum graphics SHIFT/'3' can be used.) Secondly, the INPUT used in the message preparation routine takes his Spectrum out of its power up 'K' cursor mode.

The effect obtained if you "go directly on line" via option 1 on the VTX5000 main menu straight after power up, is that keying 'a' transmits the character 'f', 'b' gives 'g' and so on, each letter giving a character five letters later in the alphabet. This is because the 'K' mode interpretation of the key 'a' is the keyword NEW, whose 8 bit token code is exactly 128 more than 'f' (see Appendix I). If you follow the normal log on routine and key in your identity number, or prepare a message off line, or BREAK and LOAD an altered version of the control program, your use of the Spectrum INPUT routine changes the cursor mode to 'L' and keying 'a' will then correctly give you 'a'. To solve this bug the VTX5000 ROM should have set bit 3 of system variable FLAGS during its initialisation procedures.

If you ever set the BASIC control program RUNning by typing RUN : and ENTER, the fault will reappear, as the ':' sets the Spectrum into 'K' mode ready for the next keyword. To prevent this obscure possibility from upsetting our version of the BASIC control program, we can add

```
20 POKE 23611,204
```

along with our other alterations. 204 is the normal result of PEEKing into address 23611 if everything is working correctly.

Going back to the problem of typing text between two VTX5000s, either user can use CAPS SHIFT/ENTER at any point to go off and print what is on the screen, or SAVE it in RAM or whatever. However, while the user is doing this, any text sent to him down the telephone line will be ignored and lost.

To return to voice contact when the session is over, the sender lifts his telephone receiver and raises his line switch. The recipient should raise his receiver shortly before this time, and when "Line Break" happens, should raise his line switch. Bad timing in this manoeuvre can lose the telephone connection. It may be more convenient to disconnect completely at the end of the last screen, by the sender simply raising his line switch to its upper position.

Transferring full screens automatically

The prepared message routines are very useful for reducing telephone line time by sending text faster than normal typing speed, but they are limited to 598 characters, and normally do not include colour or graphics. We can write a program to send the current frame of up to 960 characters, including graphics and colour. This is useful if you want to send your friend copies of things you have seen on Prestel or, in my case, frames you are designing for Prestel. Using the RAM storage routines given earlier, you can have up to 26 frames ready for transmission in any one telephone call.

```
8300 LET q$=CHR$ 12: GO SUB 8400: PAUSE 2: FOR i=ix-960
     TO ix-80 STEP 40
8310 LET j=40
8320 IF PEEK (i+j-1)=32 OR PEEK (i+j-1)=0 THEN LET
     j=j-1: IF j>0 THEN GO TO 8320
8330 FOR k=1 TO j: IF PEEK (i+k-1)<32 THEN LET q$=CHR$
     27: GO SUB 8400
8340 LET q$=CHR$ PEEK (i+k-1): IF CODE q$ <32 THEN
     LET q$=CHR$ (64+CODEq$)
8350 GO SUB 8400: NEXT k: IF j=39 THEN LET q$=" ":
     GO SUB 8400
8360 IF j<39 THEN LET q$=CHR$ 13: GO SUB 8400: LET
     q$=CHR$ 10: GO SUB 8400
8370 NEXT i: FOR i=1 TO LEN t$: LET q$=t$(i): GO SUB 8400:
     NEXT i: RETURN
8400 LET s=IN 255: IF s=2*INT (s/2) THEN GO TO 8400
8410 OUT 127,CODE q$: RETURN
```

This routine can be started direct from BASIC by a command of the form:

```
LET l=1 : LET o=0: LET t$="Hang on for more": GO SUB 8300
```

The routine has a number of interesting features and problems. First, because it uses the current frame buffer and because it is written in BASIC rather than machine code, the sender cannot see what he is sending. On the benefit side, line 8320 tries to minimise the number of characters transmitted, and so speed up the process, by spotting lines which end with a lot of spaces or nulls, and cutting them short, terminating with CHR$ 13 and CHR$ 10 as in line 8360. Lines 8330 and 8340 identify the colour control codes and expand them to CHR$ 27 plus the appropriate capital letter. Line 8370 puts the string t$ on the bottom line of the transmitted frame, so that a simple instruction can be passed to the recipient. The subroutine at line 8400 checks the 8251 status register using IN 255 (see diagram 5.1) and when the 8251 is ready, transmits the next character, q$, and can be used as a general purpose byte sending routine.

Unfortunately, being written in BASIC, the routine is quite slow, taking over a minute to transmit a frame, so you can see that when it comes to transferring data or programs, which need a special program in the recipient's machine to receive data at the equivalent of a frame every 8 seconds, we will have to write the program in machine code.

Making sure the data gets through without error

English is a very redundant language. If the recipient of a VTX5000 message sees the word "Spebtrum" on the screen, he will conclude

without any special prompting that the intended word was "Spectrum"; but, it we are transmitting computerised data, how is the recipient able to tell whether a B4h he receives is correct or whether he should really have received C4h?

In a normal two way system, like that used when downloading telesoftware, there is a checksum on each block of data and the microcomputer can ask for a repeat transmission until the checksum matches the data. In a one way system, like LOADing a program from cassette tape, or LOADing a program broadcast over the Interface 1 network, a checksum can still be included but, if the checksum does not match correctly, the user simply gets an error message on the screen.

We could easily use the one way method for checking data sent over the telephone line between two VTX5000s, but line noise is much more of a problem on telephone lines than on cassette or the Interface 1 network.

So, suppose we break up the program being transmitted into blocks of 256 bytes, and attach to each of these blocks an identification number and a checksum; the receiving machine makes a note of those blocks which fail to arrive, or arrive with incorrect checksums, and the sender is informed, so that these blocks can be retransmitted until they are received correctly. The method of informing the sender of the badly received blocks has to be fairly reliable. We could use the same system for transmission in the reverse direction, but it is generally easier for the user of the receiving machine to read out a list of block numbers for the sender to type into his machine.

We are now almost ready to produce a machine code program to transfer data between two VTX5000s but, before we write the program, we need to examine the workings of the 8251 in greater detail.

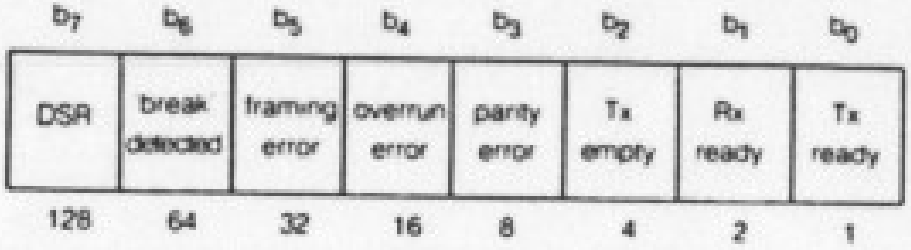
The 8251 registers

The 8251 status register can be read at any time in BASIC by an IN 255 instruction, and appears as in diagram 5.1.

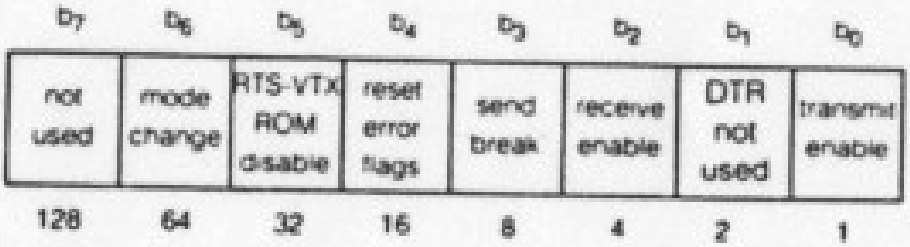
When transmitting, we are interested in the 'Tx ready' bit, b_0, which indicates that the 8251 is ready to accept another byte of data on port 127. The 8251 can accept a byte while it is still transmitting the previous byte. 'Tx empty' is only set when all bytes presented to the 8251 have been transmitted, and officially should be examined before switching into receive mode. With the VTX5000 hardware, a switch to receive mode is so slow that 'Tx empty' can be safely ignored.

Diagram 5.1

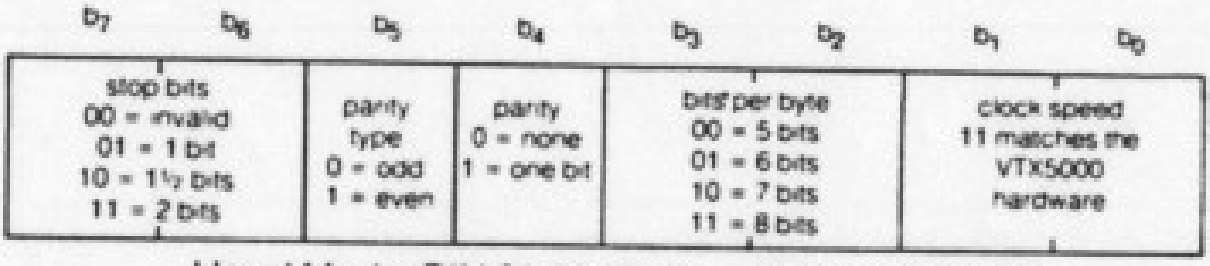
The 8251 Status Register (IN 255)



The 8251 Commands (OUT 255)

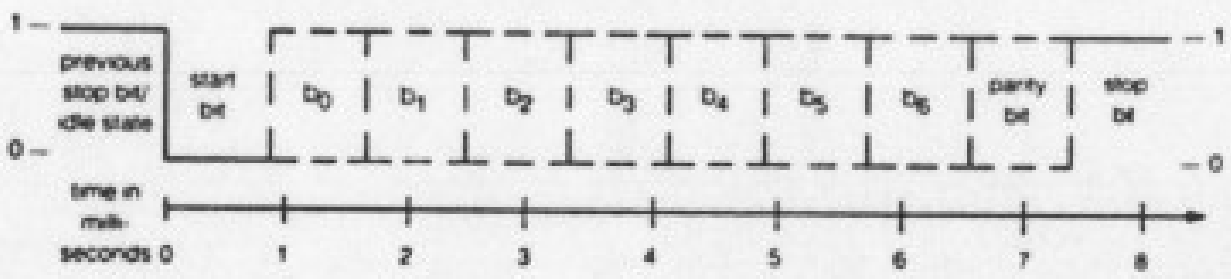


The 8251 Mode Instructions (OUT 255 after "mode change" command)



Usual Mode: BIN 01 11 10 11 — 7 bit even parity
Alternative: BIN 01 10 11 11 — 8 bit no parity

The Transmission of a Byte



When receiving, 'Rx ready' indicates that a byte is available for input on port 127. There can be various things wrong with received bytes. A 'parity error' means that the received parity bit is incorrectly matched to the other seven bits, indicating that the byte has been corrupted by line noise. A 'framing error' occurs when the stop bit does not arrive in the expected place, a worse case of line noise. An 'overrun error' happens when the program has failed to remove a byte from port 127 before the next byte arrived, meaning that one or more bytes have been lost. 'Break' shows that the line signal remains at level '0' for longer than two bytes — you could experiment using this to signal the end of transmission. If any of these error bits are set, they will not be reset until an 8251 command including b_4 is used. There are only two conventionally used OUT 255 commands, 15h and 35h, which enable both receive and transmit, reset any error flags, and enable either the VTX5000 ROM or the Spectrum ROM respectively.

DSR is a signal from the modem board indicating that tones matching a 1200 bit/second signal are present on the telephone line. DSR is thus tested to determine the presence of a 'Line Break' condition. In Tx mode, DSR is set at level 1 all the time, which explains why the use of Tx mode to test the keyboard and message sending software in Chapter 4 worked.

During initialisation, the VTX5000 sets the 8251 into the normal Prestel 7 bit plus even parity bit mode. When we want to send 8 bit data, as we do in the next section, we change this to an 8 bit no parity mode by:

```
LA      A, 40h
OUT     (FFh), A
LD      A, 6Fh
OUT     (FFh), A
LD      A, 35h
OUT     (FFh), A
```

A program to transfer data

The most general and most useful item of data we could transfer between Spectrums is a BASIC program with its variables. However, any program which transfers a BASIC program must be in machine code and must not be embodied in a BASIC program; it should be located in an odd position in memory. I have chosen to use the memory space reserved by the VTX5000 for its screen buffer at address ix−960, i.e. 64448 on a 48K Spectrum; 16K Spectrum users will have to alter some of the addresses. This allows the program to transfer most normal BASIC programs, including variations on the VTX5000 BASIC control program. Simple alterations to the program can make it transfer CODE like sections of memory. The only problems encountered come when the program to be transferred includes machine code in the same

location as the routine and this problem can usually be solved by LOADing the appropriate CODE file elsewhere in memory and informing the recipient of both its apparent and proper addresses.

The program is presented as assembly language mnemonics, with matching decimal values of machine code in the right hand column. You can choose either to use your own favourite assembler, or to use a simple BASIC routine to POKE the decimal values into place.

The following program forms a suitable decimal loader:

```
10 LET i=64448
20 INPUT a: POKE i,a: LET i=i+1: GO TO 20
```

When the machine code for the routine has been typed in, it can be SAVEd by a command of the type:

```
SAVE "transmit" CODE 64448, i-64448
```

which ensures that the file can be LOADed back by the BASIC control program as if it were a frame, and also ensures that it does not overwrite the machine stack if it is LOADed back after a RANDOMIZE USR 0

Firstly, we look at the sender's transmission routine:

```
        ORG     64448
TX:     LD      A, 40h          62,64,
        OUT     (FFh), A        211,255,
        LD      A,6Fh           62,111,
        OUT     (FFh),A         211,255,
        LD      A,35h           62,53,
        OUT     (FFh),A         211,255,
        LD      HL,(23641,E-LINE)   42,89,92,
        LD      DE,(23635,PROG)     237,91,83,92,
        AND     A               167,
        SBC     HL,DE           237,82,
        LD      B,H             68,
        INC     B               4,
        LD      HL,BUF          33,227,252,
        LD      C,B             72,
        LD      A, 00h          62,0
CLR:    LD      (HL),A          119,
        INC     HL              35,
        DJNZ    CLR             16,252,
        SUB     C               145,
        LD      B,A             71,
        DEC     B               5,
        LD      A,01h           62,1,
FIL:    LD      (HL),A          119,
        INC     HL              35,
        DJNZ    FIL             16,252,
```

```
        LD      (HL),00h        54,0
TRANS:  LD      HL,BUF          33,227,252,
        EX      DE,HL           235,
        LD      B,00h           6,0,
NBLK:   LD      A,(DE)          26,
        AND     A               167,
        CALL    Z,TXBLK         204,84,252,
        INC     DE              19,
        INC     B               4,
        JR      NZ,NBLK         32,247,
        CALL    0D6Bh,CLS       205,107,13,
        LD      A,FEh           62,254,
        CALL    1601h, CHAN-OPEN 205,1,22,
        LD      HL,FINMS        33,171,242,
        LD      B,56            6,56,
NCH:    LD      A,(HL)          126,
        RST     10h,PRINT-A-1   215,
        INC     HL              35,
        DJNZ    NCH             16,251,
KI:     CALL    10A8h,KEY-INPUT 205,168,16,
        JR      NC,KI           48,251,
        PUSH    AF              245,
        RST     10h,PRINT-A-1   215,
        POP     AF              241,
        CALL    2D1Bh,NUMERIC   205,27,45,
        RET     C               216,
FDIG:   SUB     30h             214,48,
        LD      H,00h           38,0,
        LD      L,A             111,
NDIG:   PUSH    HL              229,
NKI:    CALL    10A8h,KEY-INPUT 205,168,16,
        JR      NC,NKI          48,251,
        PUSH    AF              245,
        RST     10h,PRINT-A-1   215,
        POP     AF              241,
        CALL    2D1Bh,NUMERIC   205,27,45,
        JR      C,NUMOV         56,13,
        SUB     30h             214,48,
        POP     HL              225,
        PUSH    HL              229,
        ADD     HL,HL           41,
        ADD     HL,HL           41,
        POP     DE              209,
        ADD     HL,DE           25,
        ADD     HL,HL           41,
        ADD     A,L             133,
```

```
        LD      L,A             111,
        JR      NDIG            24,229,
NUMOV:  POP     HL              225,
        LD      DE,BUF          17,227,252,
        ADD     HL,DE           25,
        LD      (HL),00H        54,0,
VKI:    CALL    10A8h,KEY-INPUT 205,168,16,
        JR      NC,VKI          48,251,
        PUSH    AF              245,
        RST     10h,PRINT-A-1   215,
        POP     AF              241,
        CALL    2D1Bh,NUMERIC   205,27,45,
        JR      NC,FDIG         48,204,
        LD      DE,(23635,PROG) 237,91,83,92,
        JR      TRANS           24,153,
```

The following routine transmits a block of 256 bytes plus header and checksum:

```
TXBLK:  DEC     A               61,
        LD      (DE),A          18,
        PUSH    BC              197,
        PUSH    HL              229,
        LD      A,FDh           62,253,
        CALL    TXCH            205,124,252,
        LD      A,FDh           62,253,
        CALL    TXCH            205,124,252,
        LD      C,00h           14,0,
        ADD     HL,BC           9,
        LD      A,B             120,
        CALL    TXCH            205,124,252,
        INC     B               4,
        JR      Z,TXFF          40,28,
        LD      B,00h           6,0,
TXN:    LD      A,(HL)          126,
        CALL    TXCH            205,124,252,
        INC     HL              35,
        DJNZ    TXN             16,249,
        LD      A,C             121,
        CALL    TXCH            205,124,252,
        POP     HL              225,
        POP     BC              193,
        RET                     201,
```

The following routine transmits one byte, and keeps track of the checksum:

```
TXCH:   EX      AF,AF'          8,
```

```
WAIT:   IN      A,(FFh)         219,255,
        RRA                     31,
        JR      NC,WAIT         48,251,
        EX      AF,AF'          8,
        OUT     (7Fh),A         211,127,
        XOR     C               169,
        LD      C,A             79,
        RET                     201,
```

And the final section of code deals with the special "block 255":

```
TXFF:   POP     DE              209,
        LD      HL,(23627,VARS) 42,75,92,
        AND     A               167,
        SBC     HL,DE           237,82,
        LD      A,L             125,
        CALL    TXCH            205,124,252,
        LD      A,H             124,
        CALL    TXCH            205,124,252,
        LD      HL,(23641,E-LINE) 42,89,92,
        AND     A               167,
        SBC     HL,DE           237,82,
        LD      A,L             125,
        CALL    TXCH            205,124,252,
        LD      A,H             124,
        CALL    TXCH            205,124,252,
        LD      A,C             121,
        CALL    TXCH            205,124,252,
        POP     BC              193,
        RET                     201,
FINMS:  TEXT 'Transmission Finished'
                                84,114,97,110,
                                115,109,105,115
                                115,105,111,110,
                                32,70,105,110,
                                105,115,104,101,
                                100,

        DEFB    0Dh             13,
        DEFB    0Dh             13,
        TEXT    'Key block numbers of any errors'
                                75,101,121,32
                                98,108,111,99,
                                107,32,110,117,
                                109,96,101,114,
                                115,32,111,102,
                                32,97,110,121,
                                32,101,
```

```
                                        114,114,111,114,
                                            115,
        DEFB    0Dh                     13,
        DEFB    0Dh                     13,
BUF:    DEFS    0100h
```

The program uses a table 'BUF' with one byte for each block of data that could be possibly transmitted. You can PEEK into this table to check the status of any one block, a value of 0 indicates that the block is yet to be transmitted, 1 indicates that the block is outside the program area and will never be transmitted, 255 indicates that the block has been transmitted.

As hinted earlier, each of the transmitted blocks consists of 256 bytes of data, plus a header which consists of two bytes set to FDh and a byte containing the block number. The whole block is followed by a check-sum byte which is calculated as the exclusive or of the block number byte and the 256 data bytes. The FDh bytes allow the receiving program to resynchronise with the start of a block, should any error occur, as it is not normal for two FDh to appear as successive bytes of either data or program.

Block 255 is a special block containing the values of VARS minus PROG, and E-LINE minus PROG, which allow the receiving program to identify the separate parts of the BASIC program. As when downloading telesoftware, if any parts of the program depend on being at specific addresses, the recipient needs to be told whether to include or exclude Interface 1 variables and/or VTX5000 variables etc.

To operate the program, the sender LOADs the transmission program into address 64448 and LOADs the BASIC program he wants to transmit. He telephones the recipient, who LOADs and starts his receiving program, with his VTX5000 mode switch on 'Rx'. The sender sets his mode switch to 'Tx', switches on his line switch, replaces his telephone receiver and executes:

```
RANDOMIZE USR 64448
```

The program takes some time to transmit the data, about 20% longer than the time taken to LOAD the BASIC program from cassette. At the end it asks on the screen for the numbers of those blocks which were not properly received. To find out this information, the sender raises his telephone receiver and line switch, and speaks to the recipient who will list the blocks with errors. Until block 255 is correctly received, the program is unsure how many blocks it will have to receive, and thus may only indicate the absence of block 255 after the first session. The data line is re-established, and the sender keys the block numbers on the screen, separating each number with a non-numeric character, such as SPACE or ENTER. After the last number is entered, a second non-numeric character initiates retransmission of the offending blocks, and

the procedure can be repeated until the recipient is happy that all blocks have been successfully received.

To alter the transmission routine to send sections of CODE rather than the BASIC program, you just need to alter the seven bytes starting at address 64460 to put the start and end addresses of the CODE in the DE and HL registers respectively. The TXFF routine should also be altered to use the correct finishing address, although the effects of forgetting to do this can be overcome by verbal explanation to the recipient. Similar alterations are not particularly necessary at the receiving end, as long as the recipient does a SAVE "name" CODE PROG, 'length' before doing anything like an automatic listing, which would try to LIST the CODE as a BASIC program, with possibly disastrous results.

The equivalent program in the VTX5000 screen buffer of the recipient's machine is:

```
        ORG     64448
RX:     LD      A,40h           62,64,
        OUT     (FFh),A         211,255,
        LD      A,6Fh           62,111,
        OUT     (FFh),A         211,255,
        LD      A,35h           62,53,
        OUT     (FFh),A         211,255,
        LD      DE,(23635,PROG) 237,91,83,92,
        LD      HL,BUF          33,1,253,
        LD      (HL),FFh        54,255,
        LD      B,FEh           6,254,
SET:    INC     HL              35,
        LD      (HL),01h        54,1,
        DJNZ    SET             16,251,
        INC     HL              35,
        LD      (HL),FFh        54,255,
WDSR:   CALL    1F54h,BREAK-KEY 205,84,31,
        RET     NC              208,
        IN      A,(FFh)         219,255,
        RLA                     23,
        JR      NC,WDSR         48,247,
WFD:    LD      A,35h           62,53,
        OUT     (FFh),A         211,255,
BST:    CALL    RXCH            205,168,252,
        CP      FDh             254,253,
        JR      NZ,WFD          32,245,
        CALL    RXCH            205,168,252,
        CP      FDh             254,253,
        JR      NZ,WFD          32,238,
        LD      C,00h           14,0,
        CALL    RXCH            205,168,252,
```

```
        PUSH    DE              213,
        LD      HL,BUF          33,1,253,
        LD      E,A             95,
        LD      D,00h           22,0,
        ADD     HL,DE           25,
        POP     DE              209,
        LD      (BKA),HL        34,210,252,
        BIT     0,(HL)          203,70,
        JR      Z,WFD           40,217,
        CP      FFh             254,255,
        JR      Z,RXFF          40,25,
        LD      H,A             103,
        LD      L,00H           46,0,
        LD      B,L             69,
        ADD     HL,DE           25,
GNC:    CALL    RXCH            205,168,252,
        LD      (HL),A          119,
        INC     HL              35,
        DJNZ    GNC             16,249,
        CALL    RXCH            205,168,252,
        LD      A,C             121,
        AND     A               167,
        JR      NZ,WFD          32,194,
        LD      HL,(BKA)        42,210,252,
        LD      (HL),A          119,
        JR      BST             24,192,
```

You can see that the program uses a similar initialisation procedure, and similar buffer (BUF) to the transmit program. The codes used in BUF are FFh if the matching block is expected but has not arrived, 01h if the block is not expected and has not arrived, and 00h if the block has arrived. The program waits for DSR to be set, indicating an active telephone line, then for each block, or after any error, waits for two characters FDh before accepting a block into memory, unless it has already been correctly received. Block 255 needs special treatment, to produce values for VARS and E-LINE, as follows:

```
RXFF:   CALL    RXCH            205,168,252,
        LD      L,A             111,
        CALL    RXCH            205,168,252,
        LD      H,A             103,
        ADD     HL,DE           25,
        LD      (23627,VARS), HL    34,75,92,
        CALL    RXCH            205,168,252,
        LD      L,A             111,
        CALL    RXCH            205,168,252,
        LD      H,A             103,
```

```
        ADD     HL,DE           25,
        LD      (23641,E-LINE),HL   34,89,92,
        CALL    RXCH            205,168,252,
        LD      A,C             121,
        AND     A               167,
        JR      NZ,WFD          32,157,
        LD      HL,(BKA)        42,210,252
        LD      (HL),A          119,
        LD      HL,(23641,E-LINE)   42,89,92,
        SBC     HL,DE           237,82,
        LD      B,H             68,
        INC     B               4,
        LD      HL,BUF          33,1,253,
TSTN:   BIT     0,(HL)          203,70,
        JR      Z,REC           40,2,
        LD      (HL),FFh        54,255,
REC:    INC     HL              35,
        DJNZ    TSTN            16,247,
        LD      HL,TXF          33,212,252,
MES:    PUSH    DE              213,
        PUSH    HL              229,
        CALL    0D6Bh,CLS       205,107,13,
        LD      A,FEh           62,254,
        CALL    1601h,CHAN-OPEN 205,1,22,
        POP     HL              225,
NMC:    LD      A,(HL)          126,
        AND     A               167,
        JR      Z,BNOS          40,4,
        RST     10h,PRINT-A-1   215,
        INC     HL              35,
        JR      NMC             24,248,
```

After receiving block 255 correctly, the program can identify the numbers of any missing blocks. An attempt is also made to produce such a list on a 'Line Break' or on keying BREAK, which is checked for by the main character receiving loop, RXCH. The block numbers are selected and printed by the following routine:

```
BNOS:   LD      DE,BUF          17,1,253,
        LD      HL,0000h        33,0,0,
        LD      (BKA),HL        34,210,252,
CBN:    LD      A,(DE)          26,
        RLA                     23,
        JR      NC,NBN          48,6,
        LD      (BKA),A         50,210,252,
        CALL    PNO             205,163,252,
```

```
NBN:    INC     DE              19,
        INC     L               44,
        JR      NZ,CBN          32,242,
        LD      A,(BKA)         58,210,252,
        AND     A               167,
        POP     DE              209,
        JP      NZ,WDSR         194,223,251,
        LD      HL,AOK          33,236,252,
NEC:    LD      A,(HL)          126,
        AND     A               167,
        JP      Z,12A9h,MAIN-1  202,169,18,
        RST     10h,PRINT-A-1   215,
        INC     HL              35,
        JR      NEC             24,247,
```

The block number printing is achieved by jumping into the Spectrum ROM line number printing routine:

```
PNO:    PUSH    DE              213,
        PUSH    HL              229,
        JP      1A2Eh,OUT-NUM-HL 195,46,26,
```

Now, we have the single character input routine. The routine checks the BREAK key, and for 'Line Break' and line noise errors, before reading a character from the 8251 and keeping the running checksum in the C register up to date:

```
RXCH:   CALL    1F54h,BREAK-KEY 205,84,31,
        JR      NC,BRK          48,22,
        IN      A,(FFh)         219,255,
        BIT     7,A             203,127,
        JR      Z,LBRK          40,22,
        LD      B,A             71,
        AND     78h             230,120,
        JR      NZ,LERR         32,22,
        BIT     1,B             203,72,
        JR      Z,RXCH          40,236,
        IN      A,(7Fh)         219,127,
        LD      B,A             71,
        XOR     C               169,
        LD      C,A             79,
        LD      A,B             120,
        RET                     201,
BRK:    LD      HL,BRKM         33,249,252,
NORET:  POP     BC              193,
        JR      MES             24,156,
LBRK:   LD      HL,LBRKM        33,244,252,
        JR      NORET           24,248,
```

```
LERR:   POP     BC              193,
        JP      WFD             195,232,251,
```

Location BKA is used as a temporary store on two occasions; firstly to store the address of the BUF element for the block currently being received, and secondly for a flag indicating whether any blocks are missing:

```
BKA:    DEFS    2               0,0,
TXF:    TEXT    'Transmission Finished'
                                84,114,97,110,
                                115,109,105,115,
                                115,105,111,110,
                                32,70,105,110,
                                105,115,104,101,
                                100
        DEFB    0Dh             13,
        DEFB    0Dh             13,
        DEFB    00h             0,
AOK:    TEXT    'ALL OK'        65,108,108,32,
                                    79,75,
        DEFB    0Dh             13,
        DEFB    00h             0,
LBRKM:  TEXT    'Line'          76,105,110,101,
        DEFB    20h             32,
BRKM:   TEXT    'Break'         66,114,101,97,
                                    107,
        DEFB    0Dh             13,
        DEFB    0Dh             13,
        DEFB    00h             0,
BUF:    DEFS    0100h
```

The receiving program is started by:

```
RANDOMIZE USR 64448
```

and accumulates data received in mode 'Rx' until the end of transmission is indicated by receipt of block 255, 'Line Break' or BREAK. It displays a list of missing blocks for the recipient to read to the sender, then immediately tries to receive more, until all blocks are received or BREAK is used while there is no signal on the telephone line.

To alter the program to correctly locate sections of CODE, rather than a BASIC program, the four bytes starting at address 64460 need changing to load the DE register pair with the start address of the CODE.

The block format used in these programs is slightly non standard, and is unlikely to match any commercially available data transmission software, either for the Spectrum or other microcomputers; but the format is particularly simple and the programs are short enough to key in on the Spectrum keyboard without too much difficulty.

Conclusion

In the course of this book I have covered a wealth of information about Prestel and the VTX5000. Even so, I have only managed to scrape the surface of these interesting and complex subjects. There are many facets of Prestel which I have not covered, through lack of space, and there are many interesting and useful routines hidden in the depths of the VTX5000 ROM which I leave you to discover for yourself.

Appendix 1

The Prestel and Spectrum character sets compared

Dec	Hex	Spectrum	Prestel		Teletext (Used in Screen Buffer)	Codes 128 Greater		
			Normal	Graphics		Spectrum	Dec	Hex
0	00	—	NUL (no action)		—		128	80
1	01	—	—		Alphanumerics RED		129	81
2	02	—	—		Alphanumerics GREEN		130	82
3	03	—	—		Alphanumerics YELLOW		131	83
4	04	—	—		Alphanumerics BLUE		132	84
5	05	—	ENQ (request for identity no.)		Alphanumerics MAGENTA		133	85
6	06	PRINT comma *	—		Alphanumerics CYAN		134	86
7	07	EDIT *	—		Alphanumerics WHITE		135	87
8	08	cursor left	cursor left (backspace)		flash		136	88
9	09	cursor right	cursor right (HT)		steady		137	89
10	0A	cursor down	cursor down (line feed)		end box		138	8A
11	0B	cursor up	cursor up (VT)		start box		139	8B
12	0C	DELETE *	clear screen (FF)		normal height		140	8C
13	0D	ENTER *	carriage return (CR)		double height		141	8D
14	0E	number *	—		—		142	8E

* These codes are specially interpreted by the VTX5000 Input Routine

Dec	Hex	Spectrum	Prestel		Teletext (Used in Screen Buffer)	Codes 128 Greater		
			Normal	Graphics		Spectrum	Dec	Hex
15	0F	—	—		—	■	143	8F
16	10	INK control	—		—	graphic A	144	90
17	11	PAPER control	cursor on (visible)		graphics RED	graphic B	145	91
18	12	FLASH control	—		graphics GREEN	graphic C	146	92
19	13	BRIGHT control	—		graphics YELLOW	graphic D	147	93
20	14	INVERSE control	cursor off (invisible)		graphics BLUE	graphic E	148	94
21	15	OVER control	—		graphics MAGENTA	graphic F	149	95
22	16	AT control	—		graphics CYAN	graphic G	150	96
23	17	TAB control	—		graphics WHITE	graphic H	151	97
24	18	—	CAN (erase to end of line)		conceal display	graphic I	152	98
25	19	—	—		contiguous graphics	graphic J	153	99
26	1A	—	—		separated graphics	graphic K	154	9A
27	1B	—	ESCape		—	graphic L	155	9B
28	1C	—	—		black background	graphic M	156	9C
29	1D	—	—		new background	graphic N	157	9D
30	1E	—	cursor home (top left)		hold graphics	graphic O	158	9E
31	1F	—	—		release graphics	graphic P	159	9F
32	20	SPACE	SPACE		—	graphic Q	160	A0
33	21	!	!		—	graphic R	161	A1
34	22	"	"		—	graphic S	162	A2
35	23	# *	£		—	graphic T	163	A3
36	24	$	$		—	graphic U	164	A4
37	25	%	%		—	RND	165	A5
38	26	&	&		—	INKEY$	166	A6
39	27				—	PI	167	A7
40	28	(	(		—	FN	168	A8

* 23h is translated to 5Fh by the VTX5000 Output Routine

Dec	Hex	Spectrum	Prestel			Codes 128 Greater		
			Normal	Graphics	After ESCape	Spectrum	Dec	Hex
41	29	)	)		—	POINT	169	A9
42	2A	*	*		—	SCREEN$	170	AA
43	2B	+	+		—	ATTR	171	AB
44	2C	,	,		—	AT	172	AC
45	2D	—	—		—	TAB	173	AD
46	2E	.	.		—	VAL$	174	AE
47	2F	/	/		—	CODE	175	AF
48	30	0	0		—	VAL	176	B0
49	31	1	1		remote identity programming-1	LEN	177	B1
50	32	2	2		rem id prog step 2	SIN	178	B2
51	33	3	3		rem id prog step 3	COS	179	B3
52	34	4	4		rem prog mode	TAN	180	B4
53	35	5	5		tape pause on playback	ASN	181	B5
54	36	6	6		tape start	ACS	182	B6
55	37	7	7		tape stop	ATN	183	B7
56	38	8	8		—	LN	184	B8
57	39	9	9		—	EXP	185	B9
58	3A	:	:		—	INT	186	BA
59	3B	;	;		—	SQR	187	BB
60	3C	<	<		—	SGN	188	BC
61	3D	=	=		—	ABS	189	BD
62	3E	>	>		—	PEEK	190	BE
63	3F	?	?		—	IN	191	BF
64	40	@	@		—	USR	192	C0
65	41	A	A		Alphanumerics RED	STR$	193	C1
66	42	B	B		Alphanumerics GREEN	CHR$	194	C2
67	43	C	C		Alphanumerics YELLOW	NOT	195	C3
68	44	D	D		Alphanumerics BLUE	BIN	196	C4
69	45	E	E		Alphanumerics MAGENTA	OR	197	C5

Dec	Hex	Spectrum	Prestel			Codes 128 Greater		
			Normal	Graphics	After ESCape	Spectrum	Dec	Hex
70	46	F	F		Alphanumerics CYAN	AND	198	C6
71	47	G	G		Alphanumerics WHITE	<=	199	C7
72	48	H	H		flash	>=	200	C8
73	49	I	I		steady	<>	201	C9
74	4A	J	J		end frame	LINE	202	CA
75	4B	K	K		start frame	THEN	203	CB
76	4C	L	L		normal height	TO	204	CC
77	4D	M	M		double height	STEP	205	CD
78	4E	N	N		—	DEF FN	206	CE
79	4F	O	O		—	CAT	207	CF
80	50	P	P		—	FORMAT	208	D0
81	51	Q	Q		graphics RED	MOVE	209	D1
82	52	R	R		graphics GREEN	ERASE	210	D2
83	53	S	S		graphics YELLOW	OPEN #	211	D3
84	54	T	T		graphics BLUE	CLOSE #	212	D4
85	55	U	U		graphics MAGENTA	MERGE	213	D5
86	56	V	V		graphics CYAN	VERIFY	214	D6
87	57	W	W		graphics WHITE	BEEP	215	D7
88	58	X	X		conceal display	CIRCLE	216	D8
89	59	Y	Y		contiguous graphics	INK	217	D9
90	5A	Z	Z		separated graphics	PAPER	218	DA
91	5B	[	←		—	FLASH	219	DB
92	5C	\	½		black background	BRIGHT	220	DC
93	5D	]	→		new background	INVERSE	221	DD
94	5E	↑	↑		hold graphics	OVER	222	DE

Dec	Hex	Spectrum	Prestel			Codes 128 Greater		
			Normal	Graphics	After ESCape	Spectrum	Dec	Hex
95	5F	—	#		release graphics	OUT	223	DF
96	60	£ *	—		—	LPRINT	224	E0
97	61	a	a		—	LLIST	225	E1
98	62	b	b		—	STOP	226	E2
99	63	c	c		—	READ	227	E3
100	64	d	d		—	DATA	228	E4
101	65	e	e		—	RESTORE	229	E5
102	66	f	f		—	NEW	230	E6
103	67	g	g		—	BORDER	231	E7
104	68	h	h		—	CONTINUE	232	E8
105	69	i	i		—	DIM	233	E9
106	6A	j	j		—	REM	234	EA
107	6B	k	k		—	FOR	235	EB
108	6C	l	l		—	GO TO	236	EC
109	6D	m	m		—	GO SUB	237	ED
110	6E	n	n		—	INPUT	238	EE
111	6F	o	o		—	LOAD	239	EF
112	70	p	p		—	LIST	240	F0
113	71	q	q		—	LET	241	F1
114	72	r	r		—	PAUSE	242	F2
115	73	s	s		—	NEXT	243	F3
116	74	t	t		—	POKE	244	F4
117	75	u	u		—	PRINT	245	F5
118	76	v	v		—	PLOT	246	F6
119	77	w	w		—	RUN	247	F7
120	78	x	x		—	SAVE	248	F8
121	79	y	y		—	RANDOMIZE	249	F9
122	7A	z	z		—	IF	250	FA
123	7B	{	¼		—	CLS	251	FB
124	7C	|	½		—	DRAW	252	FC
125	7D	}	¾		—	CLEAR	253	FD
126	7E	~	÷		—	RETURN	254	FE
127	7F	©	■		—	COPY	255	FF

* 60h is translated to 23h by some of the later versions of the VTX5000 Output Routine.

Appendix II

Prestel Commands

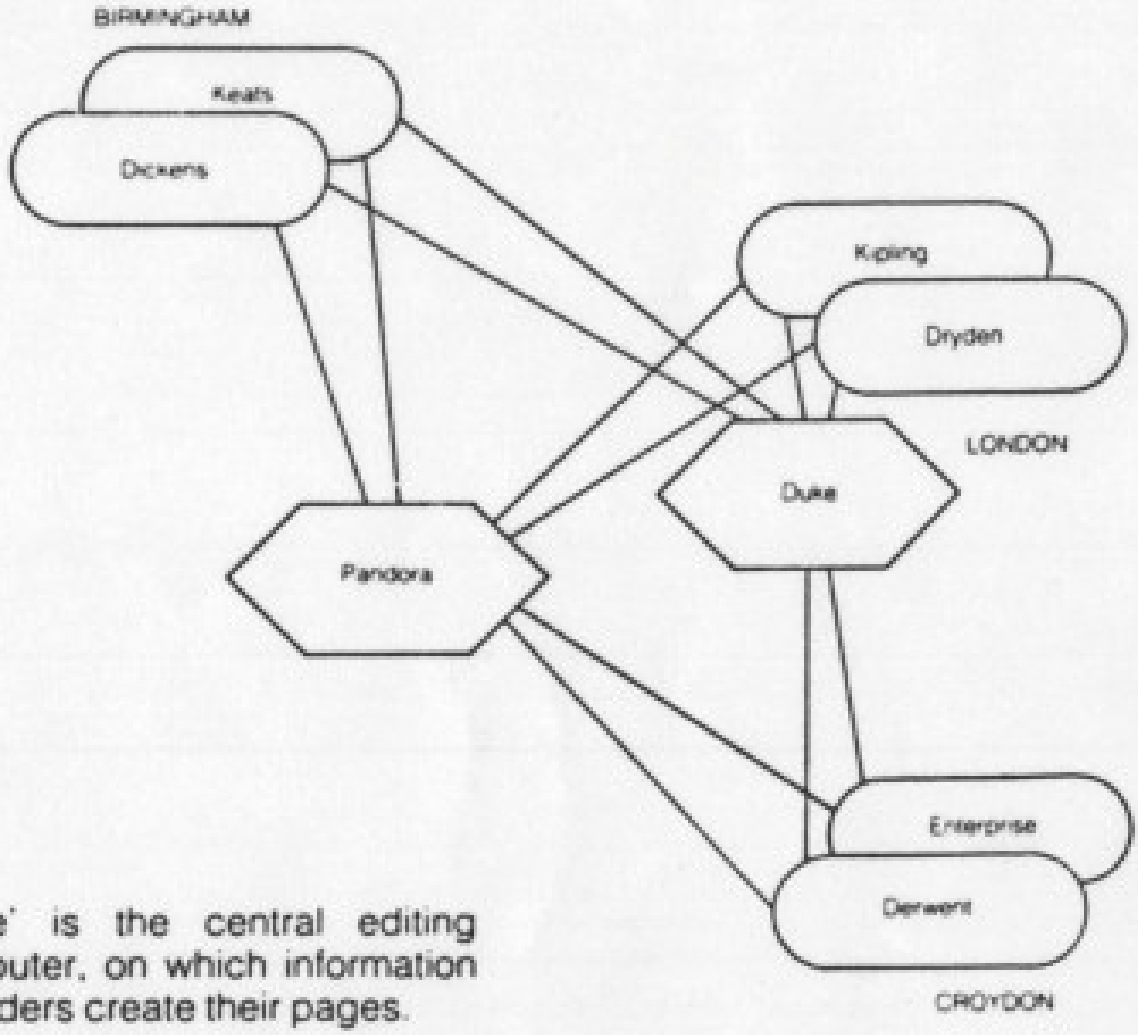
Command	Effect on Prestel Information Page	Effect on Prestel Message Page	Effect on External Computer
Any Digit 0 to 9	Follow the route to a new a frame	Accept digit into message field	As Prestel
#	Request the next frame of the same page	Move onto the next message field. Keying # at the start of a field leaves the existing contents, otherwise the rest of the field is erased	As Prestel, except that a # route can be linked to any page, and # is not needed if you overflow a field
* * (or CAN - 18h)	Cancel any other * command you are in the middle of	Used in the middle of a field erases that field Used at the start of a field moves the cursor back to the previous field	As Prestel
* #	Request previously viewed frame (only 3 numbers are held in the Prestel computer's memory)	As on information pages, the current message is abandoned	Depends on the external computer. On some *# is not allowed. On others, more than 3 numbers are remembered
* 00	Redisplay current frame (no extra charge)	Redisplay frame, leaving field contents, moving cursor to the start of the first field	As Prestel
* 02	Illegal	Illegal	Display a special 'goodbye' frame and return control to Prestel
* 09	Update current frame with any recent change (frame charge is added to your bill). On external computer 'Goodbye' frame, it displays the Gateway frame	Redisplay frame, with blank fields, and the cursor at the first field	As Prestel

Command	Effect on Prestel Information Page	Effect on Prestel Message Page	Effect on External Computer
* 0#	Display your page zero (which depends on your user record)	As on information pages	As Prestel, returning control to Prestel
* 1#	Display the Prestel main index	As on information pages	Depends on the external computer
* number #	Request the a frame of the numbered page	As on information pages	Depends on the external computer
Printing characters e.g. A B C £ $ etc	Ignored	Accepted into message field	As Prestel
Cursor movement	Ignored	08h-0Bh, accepted so long as the cursor stays within current field 0Dh, move to start of current/previous field 1Eh, move to start of first field	08h-0Bh, roughly as Prestel, but some movement between fields is allowed 0Dh, 1Eh, illegal at present. This may shortly be changed to match Prestel
ESCape J	Ignored	Go to end of last field, ready to send message	Illegal
ESCape K	Ignored	Display your page zero	Illegal
Other ESCape Sequences	Ignored	Ignored, except on special, colour accepting fields	Illegal

Appendix III

The Prestel Computer Network

a. National Network



'Duke' is the central editing computer, on which information providers create their pages.

'Pandora' is the "National Mailbox" central message computer, on which messages are held until retrieved (open late 1984).

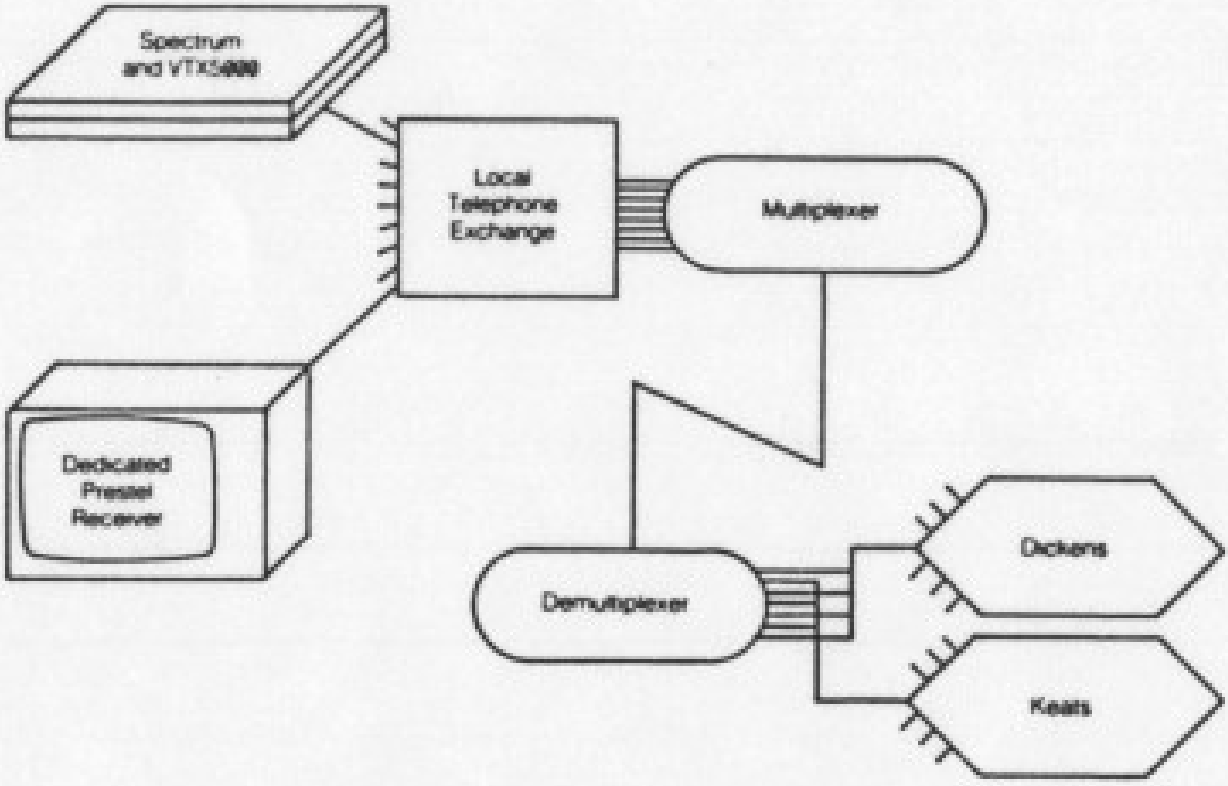
'Dickens', 'Keats', 'Dryden', 'Kipling', 'Derwent' and 'Enterprise' are locally accessed information retrieval computers.

'Enterprise' was initially the only user to user Mailbox computer.

b. Local Network Example

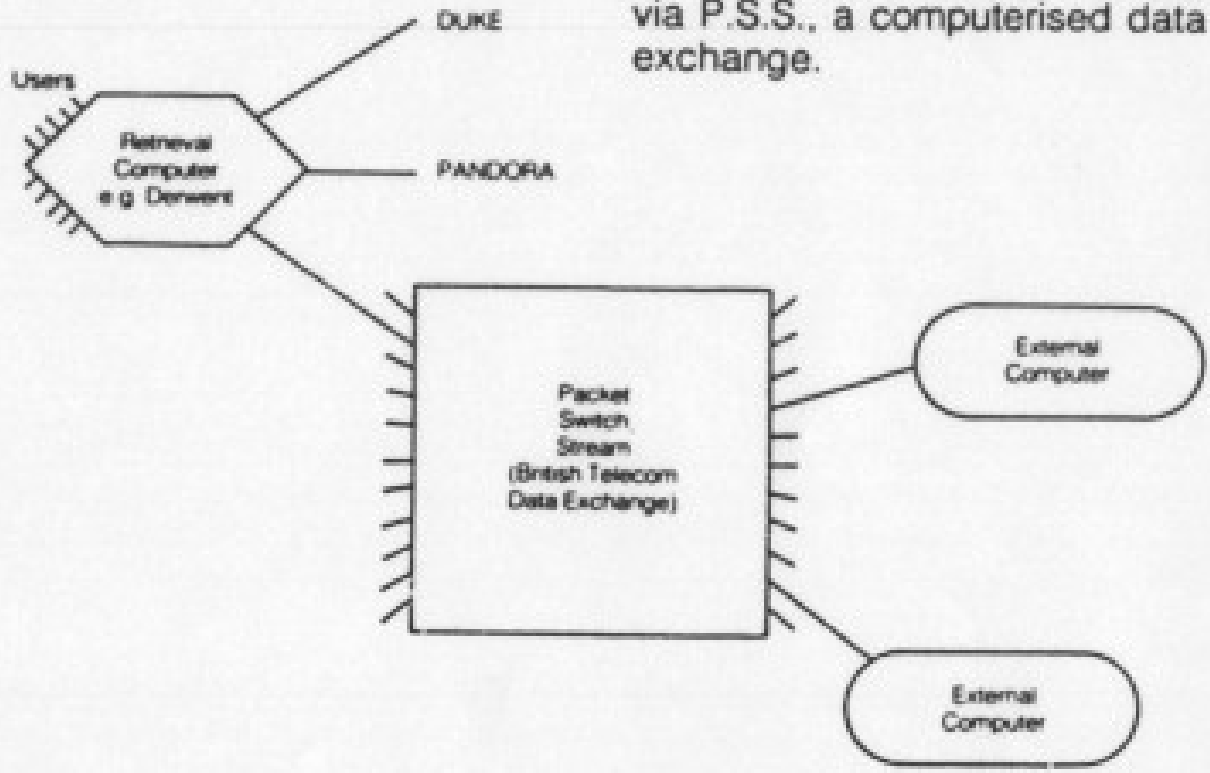
94% of the U.K. population is within a local call of a multiplexer, giving access to a pair of information retrieval computers.

Alternate telephone lines are 'interleaved', so that if one computer is not working, you still get through to the other.

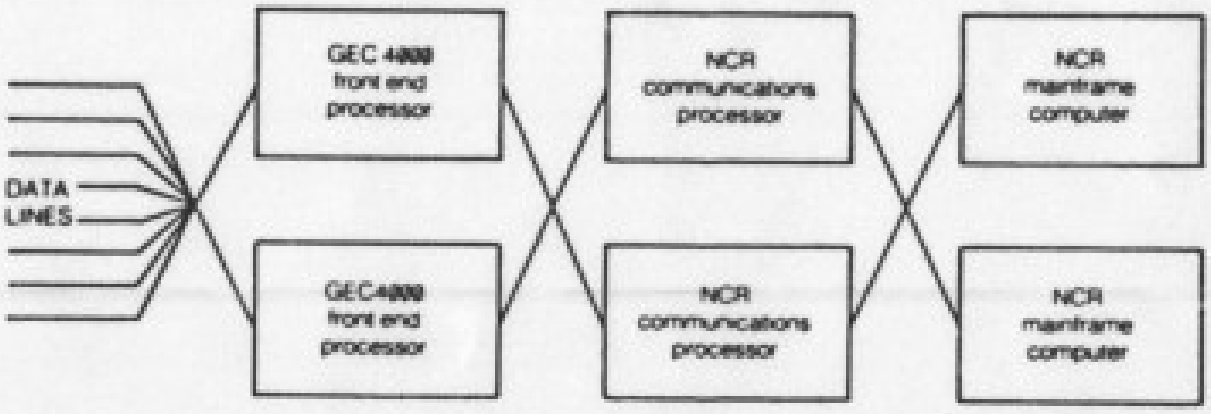


c. Connection to External Computers

All Prestel retrieval computers are connected to all the appropriate external computers, usually via P.S.S., a computerised data exchange.



d. The Homelink External Computer configuration, a complex example



As long as one front end processor, one communications processor and one mainframe computer are working, Homelink can maintain their home banking service. Other 'external computers' frequently involve only one machine, and are thus more likely to be unavailable.

Index

Published in the United Kingdom by:
Melbourne House (Publishers) Ltd.,
Church Yard,
Tring, Hertfordshire HP23 5LU.
ISBN 0 86161 167 5

Published in Australia by:
Melbourne House (Australia) Pty. Ltd.,
70 Park Street,
South Melbourne, Victoria, 3205.

Printed in Hong Kong by Colorcraft Ltd.
1st Edition

D C B A 9 8 7 6 5 4 3 2 1 0