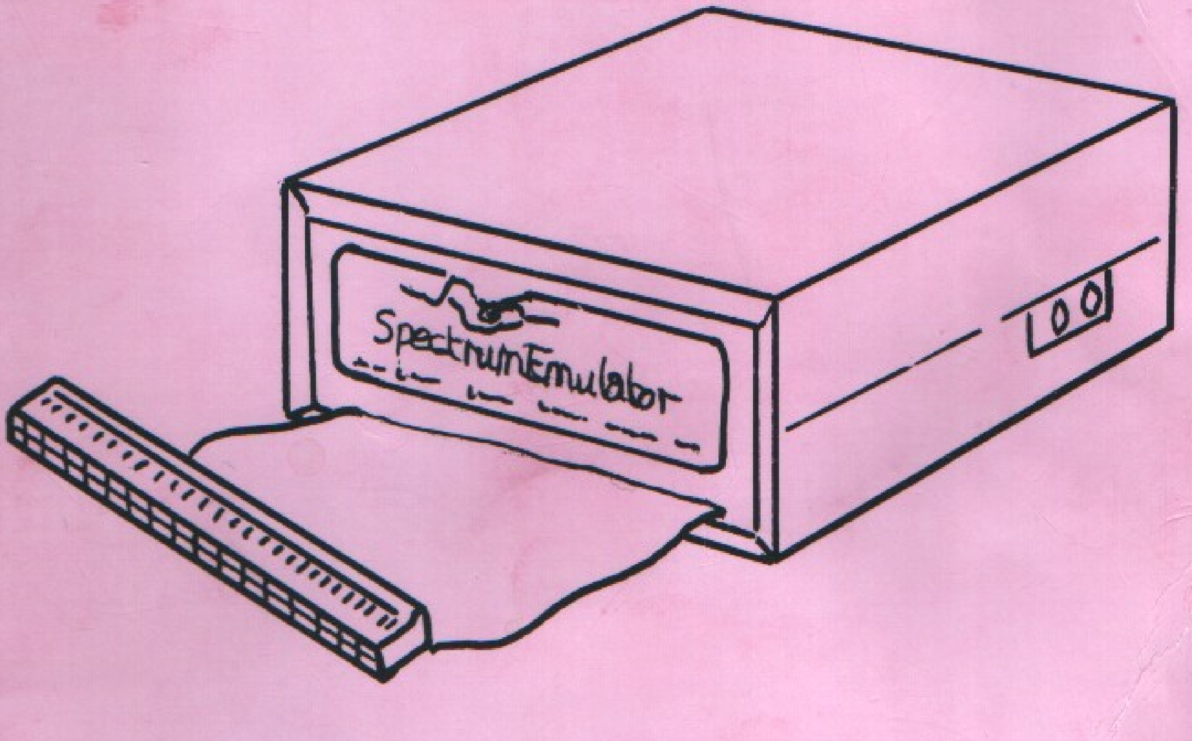
THE EINSTEIN SPECULATOR



MIKE

SMALLMAN

ERRATUM

Since this book went to print I have been made aware of changes made to some of the games featured. There follows revised loaders for those that to ny knowledge, have been altered together with corrections for errors in code that crept in.

FEUD there is a mistake in the code on page 31 the correct code should read.

```
018D6011729FCD56906 <E>
C3B5A2. <E>
```

RESCUE the main code for RESCUE is a headerless file and the CD56O6 should read CD59O6.

SOS there is a version of S.O.S. that loads without a screen, for this version modify SP1.COM from 1969.

SPITFIRE 40 the correct loader for this program should read..

```
Modify SP1.COM from 1969..
      01004011121BCD5606 <E>
      010060110096CD5906 <E>
      C30060. <E>
```

THRUST there is a version of THRUST that uses headerless files.

```
Modify SP1.COM from 1969..
      0178FF111E00CD56O6 <E>
      01004011E8BDCD59O6 <E>
      014A31116000CD5906 <E>
      C300FA. <E>
```

CAVELON in one version requires an extra piece of code before the jump to start, the loader for this version is..

```
015078117869CD5606 <E>
3E02326B5C21004011 <E>
014001001B3600EDB0 <E>
C394EA. <E>
```

PAGE 7 Line 1410 should read..

```
1410 LET T=(48+U-INT(U/16)*16): IF T>57 THEN LET T=T+7
```

* THE EINSTEIN SPECULATOR *

INTRODUCTION

This book is for those Einstein users who intend to or have purchased the Einstein Speculator and would like either a deeper insight into its workings or simply wish to transfer Spectrum programs whose loaders are not included in the original package.

For those with no knowledge of machine code the second half of this book contains the loading code for several games which may be used by following the instructions, however to get the full use of this book even those who have not used machine code will find it beneficial to read through the explanations given.

For the first half I shall have to assume a knowledge of Hexadecimal, and a minimal knowledge of machine code, you should also make yourself acquainted with the use of the Einsteins MOS functions. I advise you to read the Einstein DOS/MOS manual and try using the H, D and T functions in MOS.

In order to retrieve the information from the games programs necessary for transfer to the Einstein you will require the use of a Spectrum. If you have a good grasp of machine code and a working knowledge of the Spectrum you may get by without the use of a Spectrum by using the CLOAD.COM program that is available for the Speculator from B&H of Halifax. (0422 52905).

I must say now that I will be dealing with normal and headerless programs only, programs that use Speed, Slow or Jerky loaders are beyond the scope of this book. I shall also be showing you how to use the joystick option on programs you transfer.

IMPORTANT: You should first make a backup copy of the Speculator disc and use this copy for all work with this book.

Note that <E> is used throughout this book to represent the ENTER key.

THE SPECULATOR HARDWARE

When you purchase your Speculator you will receive a cream coloured box 5" * 4" * 2" at the front of which is a ribbon cable terminated in a 60way plug which is inserted into the Einsteins PIPE connector at the rear of the machine. CAUTION : this plug must NOT be inserted or removed while there is power to the Einstein. On the right-hand side of the Speculator there are two 1.5mm sockets. The socket nearest the rear of the Specuator is for loading programs and the socket to the front is for saving programs to tape. The save facility is used in games which allow you to save your position for resuming play at a later date.

Due to a certain amount of code being carried on integrated circuts, (chips) in the Speculator, programs will not run if the Speculator is not connected.

THE SPECULATOR SOFTWARE

The Speculator software is held on the disc supplied as SP1.COM and SPID.COM. If you purchased your Speculator from B&H of Halifax you may also have on the disc CLOAD.COM and CSAVE.COM. These two additional programs are available from B&H of Halifax by sending your original Speculator disc and a nominal handling charge.

SP1.COM:- This is the main program containing the Speculator operating system and loaders for twenty Spectrum programs.

SPID.COM:- This program puts a pseudo directory on your games disc.

CLOAD:- This program allows you to load any Spectrum program, with a normal loader, to Einstein disc. Used for transferring ASCII word processing files or for examining the Spectrum code. Programs transferred using CLOAD.COM will NOT run on the Einstein with or without the Speculator connected.

CSAVE:- This program allows you to save any file from Einstein disc to tape in Spectrum format.

USING YOUR SPECULATOR

Power off your Einstein and insert the 60 way plug of the Speculator into the PIPE connector at the rear of the computer. Once you are sure this connection is properly made you can switch on your Einstein, do not have a disc in your drive at this time. Before actually using the Speculator FORMAT a blank disc by inserting your master disc, supplied with the Einstein, into drive 0. Press CTRL/BREAK as normal to boot the DOS. Type BACKUP <E> and select 'F' from the menu. Press the ENTER key, this will read the DOS from your disc into memory. Take out your disc and insert a blank disc into drive 0 and press '0'. Once the 'A' side of the disc has been Formatted and verified you can FORMAT the 'B' side by answering 'Y' to the prompt turning the disc over and pressing '0'. When that process is complete press 'N' and 'X' this should return you to DOS. What we have done is FORMAT a disc ready for saving the games we are to transfer from tape to disc.

Using the Speculator we can save only three games to each side of the disc, this is because the Speculator saves programs to disc in 50k byte blocks irrespective of the actual length of the game. Another point to note is that the Speculator does not write a directory entry, this is overcome by the SPID.COM program that is used to write a pseudo directory onto our FORMATTED disc. This is our next task. Remove the disc you have just FORMATTED and insert your Speculator disc and type SPID <E> follow the prompts by removing the Speculator disc, reinserting the FORMATTED disc and pressing ENTER, turn the disc over and write the directory to the second side. Exit the SPID.COM program as prompted. Now type DIR <E> and you will see F1.SPC, F2.SPC and F3.SPC the same will be true for the other side of the disc. If you were to save Speculator games to your disc without running SPID.COM, when you did a DIR <E> you would get a disconcerting 0k Size, 180k Free, 190k Total. If you have forgotten to run SPID.COM before saving your Speculator games you can still run SPID.COM without affecting the games.

Which drive ?... The Speculator always uses drive 0, however since we can only get three games per side on our 3" disc the cost of holding a large number of games is high. There are two ways around this problem, both of which require external 5 1/4" drives. The first is for those who have a 80 track double sided drive, you can simply save each file to the 80 track disc by first loading the file from DOS with LOAD Fn.SPC <E>, where n is the file number, selecting the drive and saving the file with SAVE 200 FN.SPC <E>, where FN is a distinguishing name such as JET PAC etc. This enables you to save up to 14 games on each 5 1/4" disc. When you wish to use the games with your Speculator, select the 80 track drive type LOAD FN.SPC <E>, select drive 0 in which you have inserted a prepared disc and type SAVE 200 Fn.SPC <E>. You can then load the games as per the instructions.

The second method is for those with a 5 1/4" 40 track single sided drive. By modifying a byte of SP1.COM you can select the external drive for use by the Speculator. If you want to do this modification insert your backup copy of the Speculator disc in drive 0 and from DOS type the following....

```
LOAD SP1.COM <E>
MOS <E>
M2EFD <E>
nn. <E>         where nn is the drive number of your 5 1/4"
                drive, (01 02 or 03).
```

Press CTRL/BREAK to return to DOS and select type SAVE 48 SP1.COM <E>. Now when SP1.COM is run all loads/saves of files will be done from or to your 5 1/4" drive. Though you still can save only three programs per side 5 1/4" discs are very cheap.

The rest of this book will assume you are using the drive 0 version.

BACK TO TRANSFERS:
You are now ready for your first transfer, take out the disc you are going to save games to and insert the Speculator disc. Type SP1 <E> and after the program has loaded you will be greeted with the menu below. If you do not have the Speculator connected or the plug is incorrectly fitted you will get the message.

'Speculator not connected'.

Remember to power off your Einstein before connecting or adjusting the plug.

MENU

A	ARCADIA
B	ASTRONUT
C	ATIC ATAC
D	DECATHALON
E	FLIGHT SIMULATION
F	GRID RUNNER
G	HUMPTY DUMPTY
H	HUNCHBACK
I	JETPAC
J	JUMP CHALLENGE
K	LASERWARP
L	POTTY PIGEON
M	PROJECT FUTURE
N	SPECTIPEDE
O	STARRION
P	STOP THE EXPRESS
Q	THE HOBBIT
R	TORNADO LOW LEVEL
S	TRAXX
T	TWIN KINGDOM VALLEY

You do of course require the original Spectrum game tapes in order to transfer them to the Einstein. Unfortunately some of the games, SPECTIPEDE and STARRION in particular were released with several different loaders and consequently the game you acquire may not load.

LOADING FROM THE MENU

Let us say you wish to load JETPAC, connect your cassette player to the Speculator, insert your JETPAC tape and rewind to the start. Select 'I' from the menu and then press any key. Press play on the cassette player and JETPAC will load. Once the load is completed the game will run automatically, you can now save the game to disc. Remove the Speculator disc, insert your previously prepared disc and hold down Function key 5 together with '1'. You will then get the prompt 'Save file 1 ?' press 'Y' and JETPAC will be saved to disc. You can now reload JETPAC at any time provided the Speculator is connected, SP1.COM is running and the disc with JETPAC on is in drive 0. To do this simply hold down Function key 6 and '1', you will get the prompt 'Load file 1 ?', press 'Y' and JETPAC will load, you will now get the prompt 'INTERRUPT LEVEL 1 or 2 ?'. There are two interrupt levels used by the Spectrum, 1 and 2. Most games run under level 1, which is true for JETPAC, a few programs will run under level 2, make a note of these as the wrong selection here may require you to reset your machine, rerun SP1.COM and reload your game from disc. If a program fails at one level reload and try the other level. After selecting the interrupt level the game will run.

Sound is switched on by pressing Function key 1 and switched off with Function key 2. If you have a joystick you can select it with Function key 3 and switch back to keyboard with Function key 4. Function key 0 will pause a game, pressing any key will resume play. Function keys 5 and six are the Save and Load keys respectively. Pressing Function key 7 together with the ESCAPE key returns you to the main menu.

Make a note on your disc which file contains which game since all you have to go on in the directory is F1.SPC, F2,SPC or F3.SPC.

SPECTRUM LOADING TECHNIQUES

If we are to transfer programs other than those in the menu of SP1.COM we have to modify the loader code in SP1.COM, we must therefore have some idea of what code to alter and why...

If you listen to a Spectrum game tape from the start you will hear a steady tone followed by a short warble, then comes another steady tone followed by a longer burst of warbling. The first warble is the header for the program that follows in the longer warble. The header on a Spectrum is seventeen bytes long, these seventeen bytes tell the computer the nature of the block of code to be loaded. This header contains information as follows:.

A SINGLE BYTE WHICH IS:- 0 FOR A BASIC PROGRAM.
1 FOR A NUMERIC ARRAY.
2 FOR A CHARACTER ARRAY.
3 FOR A BLOCK OF MACHINE CODE.

TEN BYTES HOLDING THE FILENAME.

TWO BYTES THAT HOLD THE TOTAL LENGTH OF A BLOCK CODE. IN THE CASE OF A BASIC PROGRAM ONLY THE PROGRAM AREA AND THE VARIABLE AREA ARE KEPT ON TAPE.

TWO BYTES THAT FOR A BLOCK OF MACHINE CODE HOLDS THE START ADDRESS. FOR A BASIC PROGRAM THESE TWO BYTES HOLD THE STARTING 'LINE' NUMBER.

TWO BYTES THAT FOR A BASIC PROGRAM HOLD THE LENGTH OF THE PROGRAM AREA.

Although the header will contain the start address for a block of machine code the BASIC loader for a program may tell the computer to load the program from a particular address. An example of this is where a loading screen for a game may have a header which says its start address is, say, 30000 and the BASIC loader may look something like.. 20 LOAD ""SCREEN$ or 20 LOAD ""CODE 16384 both of which will load the following piece of code starting at address 16384 (4000hex), (the start address of the Spectrum screen), irrespective of what the header says. A Spectrum screen is 6912 (1B00hex) bytes long and this is something to look out for when looking at headers. This loading to a different address from that held in the header is equally true for any block of machine code and care must be taken not to take the header information as the whole truth.

MODIFYING SP1.COM

To alter SP1.COM to load a program not on the menu requires the information held in the headers for a game together with information that may be held in the BASIC loader. If a program relies on BASIC for anything other than actually loading and calling machine code it will not run on the Speculator this is because the Speculator does not support Spectrum BASIC. This will not be a major problem since in general only the poorer programs rely on BASIC.

The following program is a header reader for Spectrum programs. This is a SPECTRUM program and will NOT run on the Einstein. Type it into a Spectrum and save it to tape with SAVE "HEADREAD"LINE 50. Later I will show you how to modify CLOAD.COM to turn it into a header reader.

```
  50 CLS
 100 CLEAR 32499
 150 FOR Z=32500 TO 32509: READ Y: POKE Z,Y: NEXT Z
 200 DATA 175,55,221,33,08,127,205,86,5,201
 250 LET Y=32520: DEF FN Z(A)=PEEK (Y+A)+256*PEEK (Y+A+1)
 300 RANDOMISE USR 32500
 350 LET X=PEEK Y
 400 IF X>3 THEN GO TO 300
 450 PRINT "FILENAME: ";
 500 FOR Z=Y+1 TO Y+10: PRINT CHR$ PEEK Z;: NEXT Z
 550 PRINT : PRINT "TYPE     : ";
 600 GO SUB 1000+100*X
 650 PRINT : PRINT
 700 POKE Y,255
 750 GO TO 300
1000 PRINT "PROGRAM"
1010 LET U=FN Z(11):GOSUB 1400:PRINT "TOTAL LENGTH: ";C$;" BYTES"
1020 LET U=FN Z(15):GOSUB 1400:PRINT "PROG. LENGTH: ";C$;" BYTES"
1030 IF FN Z(13)>9999 THEN PRINT "LOAD ONLY": RETURN
1040 PRINT "RUNS FROM LINE ";FN Z(13)
1050 RETURN
1100 PRINT "NUMBER ARRAY"
1110 LET A$="": GO TO 1220
1200 PRINT "CHR... ARRAY"
1210 LET A$="$"
1220 LET U=FN Z(11):GOSUB 1400:PRINT "ARRAY LENGTH: ";C$;" BYTES"
1230 LET W=PEEK (Y+14)
1240 PRINT "ORIG. ARRAY NAME: ";CHR$ (64+32*(W/32-INT (W/32)));A$
1250 RETURN
1300 IF FN Z(11)=6912 AND FN Z(13)=16384 THEN PRINT "SCREEN STR$":
     RETURN
1310 PRINT "BYTES"
1320 LET U=FN Z(13):GOSUB 1400:PRINT "START ADDRESS: ";C$
1330 LET U=FN Z(11):GOSUB 1400:PRINT "LENGTH: ";C$;" BYTES"
1340 RETURN
1400 LET C$=""
1410 LET T=(48+2-INT (U/16)*16:IF T>57 THEN LET T=T+7
1415 LET C$=CHR$ (T)+C$
1420 LET U=INT (U/16)
1430 IF U>0 THEN GOTO 1410
1435 LET E$="0000":LET S=LEN C$:IF S<4 THEN LET S=4-S:
     LET C$=E$(1 TO S)+C$
1440 RETURN
```

When this is run, play your game tape and you will get a display like this...

```
FILENAME: JETPAC                        FILENAME: JPSP
TYPE     : BASIC PROGRAM                TYPE     : SCREEN STR$
TOTAL LENGTH: 0178 BYTES
PROGRAM LENGTH: 0178 BYTES
RUNS FROM LINE 1

FILENAME: 0
TYPE     : BYTES
START    : 6000
LENGTH   : 2000 BYTES
```

NOTE... All values are in Hexadecimal.

These are the first three blocks from JETPAC note that the second block is a Screen String. If you load the JETPAC BASIC loader into a Spectrum you will see the command LOAD ""SCREEN$ so this block of code will load to the start of the screen at 4000 Hex and is 1B00 Hex bytes long. The example above shows only the first three parts of JETPAC, it does in fact load in six parts - the BASIC loader, a loading screen and four blocks of machine code.

The header reader gives us two pieces of the puzzle in our attempt to transfer a game to the Speculator. Loading the code in will not help us if we don't know where the machine code runs from, only occasionally is it run from the start address of the code. This run address is normally found in the BASIC loader in one of several formats..

```
30 PRINT USR 30000
30 LET L = USR 30000
30 RANDOMISE USR 30000
```

Each of these three statements will have the same effect, they will call the machine code at 30000 (7530 Hex). This is the final part of the puzzle, at least for the time being.

Now we will look at the loader code for JETPAC in SP1.COM. Insert your Speculator disc into drive 0 and press CTRL/BREAK to go into DOS. Next type MOS <E> where <E> represents the ENTER key. We now use the TAB command to look at JETPAC loader code, so type T1966 1992 <E> this will give you...

```
1966 CD 13 06 CD 45 06 01 00

196E 60 11 00 20 CD 56 06 01

1976 80 5B 11 0F 00 CD 56 06

197E 01 B0 5C 11 01 00 CD 56

1986 06 01 78 5C 11 02 00 CD

198E 56 06 C3 00 60
```

I have included the spaces for ease of reading only.

The assembly code for this is...

```
ADDRESS   CODE     MNEMONIC      COMMENT
 1966     CD1306   CALL 0613     ;LOAD A 'BASIC' BLOCK.
 1969     CD4506   CALL 0645     ;LOAD A SCREEN.
 196C     010060   LD BC,6000    ;LOAD 'BC' REG. WITH START ADDRESS.
 196F     110020   LD DE,2000    ;LOAD 'DE' REG. WITH LENGTH.
 1972     CD5606   CALL 0656     ;LOAD A 'CODE' BLOCK.
 1975     01805B   LD BC,5B80    ;LOAD 'BC' REG. WITH START ADDRESS.
 1978     110F00   LD DE,000F    ;LOAD 'DE' REG. WITH LENGTH.
 197B     CD5606   CALL 0656     ;LOAD A 'CODE' BLOCK.
 197E     01B05C   LD BC,5CB0    ;LOAD 'BC' REG. WITH START ADDRESS.
 1981     110100   LD DE,0001    ;LOAD 'DE' REG. WITH LENGTH.
 1984     CD5606   CALL 0656     ;LOAD A 'CODE' BLOCK.
 1987     01785C   LD BC,5C78    ;LOAD 'BC' REG. WITH START ADDRESS.
 198A     110200   LD DE,0002    ;LOAD 'DE' REG. WITH LENGTH.
 198D     CD5606   CALL 0656     ;LOAD A 'CODE' BLOCK.
 1990     C30060   JP 6000       ;JUMP TO ADDRESS 6000.
```

As I have already mentioned JETPAC loads in six parts, if you look at the assembly listing you will see that we have six loads, the first at 1966 is a simple call to 0613, the routine at 0613 will load a Spectrum BASIC program into the Einsteins memory, although the BASIC is loaded it will not be acted upon since as we know the Speculator does not support Spectrum BASIC. Next at 1969 we have a call to 0645, this is the address of the screen load routine. If we have a program for transfer that does not have a screen then we can either write 0's into the three bytes starting at 1969 or simply overwrite this portion of code with our new loader. The next four loads are for the four blocks of machine code that make up JETPAC. Each load, a call to 0656, is preceded by loading the 'BC' register pair with the start address of that block of code and loading the 'DE' register pair with its length. All these numbers are in Hexadecimal and are held in reverse order in memory, thus the start address of the first block of code, which is 6000 Hex is held in memory as 0060 and its length, which is 2000 Hex is held in memory as 0020. At 1990 is the final jump to the start of the game, in this case it is, 6000, the same as the start address of the first block of machine code loaded.

To load in a different program it should now be clear that all we need to do is find out the start address and length of each block of code, together with the jump or USR address for that particular game and modify the JETPAC loader code as applicable.

First let us look again at JETPAC. If you run the header reader on a Spectrum and run JETPAC through it you will get the following..

```
FILENAME:         JETPAC
TYPE    :         PROGRAM          FILENAME: JPSP
TOTAL LENGTH:     0178 BYTES       TYPE    : SCREEN STR$
PROGRAM LENGTH:   0178 BYTES
RUNS FROM LINE    1

FILENAME:         0
TYPE    :         BYTES            FILENAME:        1
START ADDRESS:    6000             TYPE    :        BYTES
                                   START ADDRESS:   5B80


LENGTH  :         2000 BYTES       LENGTH  :        000F BYTES

FILENAME:         2
TYPE    :         BYTES            FILENAME:        3
START ADDRESS:    5CB0             TYPE    :        BYTES
LENGTH  :         0001 BYTES       START ADDRESS:   5C78
                                   LENGTH  :        0002 BYTES
```

Looking at the display from the header reader you will see how they correspond with the figures in the SP1.COM listing, and that the way the numbers occur in reverse order in the Einsteins memory.

AN EXAMPLE TRANSFER - FEUD

We can now try altering SP1 to load a program not appearing in the SP1 menu. For this example we shall show you how to load one of the budget, £1:99 games from BULLDOG called 'FEUD'. 'FEUD' is an arcade style adventure. The first thing we must do is run the program through the header reader on a Spectrum, this gives us the following information:

```
FILENAME:         feud             FILENAME:         code
TYPE    :         PROGRAM          TYPE:             BYTES
TOTAL LENGTH:     003E BYTES       START ADDRESS:    608D
PROGRAM LENGTH:   003E BYTES       LENGTH:           9F72
RUNS FROM LINE    1
```

As you can see this program loads in just two parts with no loading screen so we will overwrite the SP1.COM code from address 1969. Before we do the modifications we still require the jump address so load the BASIC program into a Spectrum and BREAK it, listing the program we get..

```
1 CLEAR 24716
2 REM code 24717,40818
3 LOAD ""CODE
4 RANDOMISE USR 41653
```

As you can see the programmer has been very kind and actually told us the start address and length of the code, this is very rare. Line four is the one we're interested in as it gives us the call address for the code. This address is in decimal and all our modifications to SP1.COM will be in hexadecimal, so on your Einstein enter MOS and type..

```
H 41653 <E> the answer will be displayed as A2B5.
```

We now have enough information to load 'FEUD', so assuming we are in MOS with SP1.COM loaded, type.. M 1969 <E>

```
                    018D6011727FCD5606C3B5A2. <E>
```

The full stop completes the modify command.

Our modified code now looks like this.

```
ADDRESS    CODE      MNEMONIC           COMMENT
 1966      CD1306    CALL0613     ;LOAD A 'BASIC' BLOCK.
 1969      018D60    LD BC,608D   ;LOAD 'BC' REG. WITH START ADDRESS.
 196C      11727F    LD DE,7F72   ;LOAD 'DE' REG. WITH LENGTH.
 196F      CD5606    CALL 0656    ;LOAD A 'CODE' BLOCK.
 1972      C3B5A2    JP A2B5      ;JUMP TO ADDRESS A2B5.
```

At 1966 we still load a BASIC block but we have overwritten the screen load call at 1969 since 'FEUD' does not have a loading screen. Now at 1969 we load the start address of our block of machine code into the 'BC' register pair. At 196C we load the 'DE' register pair with the length of the machine code block, we then, at 196F call the routine at 0656 that loads a block of machine code and finally at 1972 we jump to the start of the game, which as we have seen is at A2B5.

We are now ready to load in the new game to the Einstein via the Speculator so type.. G0100 <E> and the Speculator menu will be displayed. Select 'I' (JETPAC) from the menu and with your cassette player connected to the Speculator and with the 'FEUD' tape rewound to the start, press play and 'FEUD' should load. Once loaded the program will run and you can then save it to a formatted disc by pressing Function key 5 and selecting either 1, 2 or 3. Once the game has been saved we can overwrite the JETPAC loader with the loader code for another game.

That was all fairly straight forward, no protection is used on 'FEUD' so there was no difficulty in getting the information required to modify SP1.COM for its transfer to the Einstein. As you may have guessed things can't always be that easy... Read on.

HEADERLESS FILES

We have seen that a Spectrum program has a header which is used to pass information to the computer about the program to follow. Then where do we get the title for this section ? What some programmers do to try preventing people from copying their programs is to write the header for a block of code somewhere before that block of code. It may be in the BASIC loader or in another piece of code that is loaded before the 'headerless' code.

Where the 'headerless' codes header is held in the BASIC loader there are two common methods used. The first is straight forward, a short piece of machine code, looking a little like our loader routine in SP1.COM, is poked into memory using Data statements, this when called will load the 'headerless' file.
The actual machine code for loading a piece of 'headerless' machine code into a Spectrum is as follows..

```
CODE      MNEMONIC     COMMENTS
37        SCF         ;Sets the carry flag (indicates LOAD).
3EFF      LD A,FF     ;Says that the block to load is machine code.
DD210040  LD IX,4000  ;Loads the 'IX' reg. with the start address.
11001B    LD DE,1B00  ;Loads the 'DE' reg. with the length.
CD5605    CALL 0556   ;Loads a block of code.
```

This piece of code will actually load in a screen, since the start address is the same as the start of the Spectrums screen and the length is the same as that of the screen data area.
Note the similarity of the above code to the loader for JETPAC in SP1.COM. When we are transferring our programs we do not have to worry about setting the carry flag or loading the 'A' register with 'FF' these are taken care of within SP1.COM. We can see that the start address of the block of code is loaded into the 'IX' register pair on the Spectrum. When you do your conversions you simply change this to loading the 'BC' register pair. The call to the load routine is also slightly different, here it is 0556, in SP1.COM it is 0656. However if you were able to look at 0556 in SP1.COM you would find that it simply has a jump to 0656. By the way you can't directly look at 0556 in SP1.COM and see C35606 (JP,0656) since SP1.COM completely modifies itself upon running. In fact some of the code is actually written into space in the Einsteins VDP (Video Display Processor) memory.

Okay back to our 'headerless' file, this first method is simple enough and we can get the information we require to transfer the program from the data statements, though we do need to convert the numbers to Hex and then look up what they mean in a Hex/assembly language converter table. The second method is a little more complicated though we are still looking for a piece of code like that shown for the first method. What the programmer does is to save a piece of machine code as a BASIC program. To get at the code in this type of program you need either a disassembler for your Spectrum or you can load the BASIC program into the Einstein using CLOAD.COM and disassemble it with 'ZEN' or one of the other assembler/disassemblers available for the Einstein. There are in fact a couple of disassemblers available in the U.K. Einstein User Groups Public Domain library. Remember you are still looking for a piece of code that loads an address into the 'IX' register pair, loads the 'DE' register pair with the amount of bytes and then calls 0556.

The programmer may of course load a piece of code that when run, loads the 'headerless' block of code. For this method you will again require a disassembler either on the Spectrum or on the Einstein.

Whether in BASIC or in a seperate machine code program you will find, after the load section the address the program jumps to once all is loaded. This we need for our JP, statement in SP1.COM. For the BASIC it may occur as a simple PRINT USR statement, or it may be at the end of the code in the Data statements poked into memory. For the machine code saved as a BASIC program and for a piece of machine code that loads a 'headerless' block of code the jump address will invariably follow the call to 0556.

To make things a little clearer, I shall show you an example of the BASIC Data statement and the machine code loader methods. The first example will be one from the SP1.COM menu and for the second example I shall use a game not appearing in the SP1.COM menu.

EXAMPLES OF HEADERLESS FILES

First let us look at what a typical BASIC 'headerless' loader would look like. We will use the BASIC loader for 'TORNADO LOW LEVEL' so that we can see how it compares to the loader in SP1.COM. I shall only show you the relevant parts of the program.

```
  10 BORDER 1: PAPER 1: INK 7: CLEAR 65535
  20 DATA 55,62,255,221,33,0,64,17,156,191,205,86,5,201
  50 FOR n = 40000 TO 40013
  60 READ a: POKE n,a: NEXT n
3000 RANDOMIZE USR 40000
```

What we have to do now is to convert the numbers in the Data statement to Hexadecimal. This we can do by using the H MOS function, so if you are in DOS type MOS <E>. Now you can type Hn <E> where n is a decimal number and the Hexadecimal value for that number will be displayed. You can by the way also use Dn <E> to find the decimal value of a Hexadecimal number. If you've used the H fuction for all the numbers in the Data statement in line 20 you should have..

```
0037 003E 00FF 00DD 0021 0000 0040 0011 009C 00BF 00CD 0056 0005 00C9
```

We can ignore the first two zeros of each number.

These numbers should look familiar, if we convert them to assembler mnemonics we get..

```
CODE        MNEMONIC     COMMENTS
37          SCF         ;Set the carry flag.
3EFF        LD A,FF     ;Says that the block to load is machine code.
DD210040    LD IX,4000  ;Loads the 'IX' reg. with the start address.
119CBF      LD DE,BF9C  ;Loads the 'DE' reg. with the length.
CD5605      CALL 0556   ;Loads a block of code.
C9          RET         ;Returns in this case to BASIC.
```

As you will see this is the same as our previous example. the only differences are, the length is BF9C bytes and we have a Return instruction at the end. This piece of machine code is called by the RANDOMIZE USR 40000 in line 3000 of the BASIC loader. The return will take us back to the next statement in Basic, but you may say there isn't one. Here is another device the programmer uses, the code is loaded from the start of the screen table, overwriting the Spectrum variable area and writing a new piece of BASIC into memory.

At this stage we come across one of the other little problems, my copy of 'TORNADO LOW LEVEL' has a different jump address to that in SP1.COM however the principle is the same.

We'll now look at the loader code in SP1.COM for TLL.
From DOS with your Speculator disc inserted in drive 0. Type LOAD SP1.COM <E> then type MOS <E> T1C78 1C91 <E> this will give you..

```
1C78 CD 13 06 01 E0 3F 11 9C
1C80 BF CD 59 06 01 01 00 CD
1C88 80 1D 3E 99 32 9B FF C3
1C90 A2 8D
```

The assembly code for this is....

```
ADDRESS  CODE      MNEMONIC          COMMENTS
 1C78    CD1306    CALL 0613     ;Load a 'BASIC' block.
 1C7B    01E03F    LD BC,3FE0    ;Load the 'BC' reg. with start address.
 1C7E    119CBF    LD DE,BF9C    ;Load the 'DE' reg. with length.
 1C81    CD5906    CALL 0659     ;Load a 'headerless' block of code.
 1C84    010100    LD BC,0001    ;Load the 'BC' reg. with 1.
 1C87    CD801D    CALL 1D80     ;
 1C8A    3E99      LD A,99       ;Load the 'A' reg. with 99 Hex.
 1C8C    329BFF    LD (FF9B),A   ;Load the 99 into location FF9B.
 1C8F    C3A28D    JP 8DA2       ;Jump to address 8DA2.
```

Now you can see the problems that lie in there being various versions of a particular program. Do not despair, the principles still hold true and if you have a copy of TLL or any of the other programs in the original menu that do not load, look at the start addresses, lengths and jump addresses and modify SP1.COM to suit.

An interesting point about the loader above is that it gives the start address of the code as 3FE0. This address is actually part of the Spectrum ROM and cannot be altered, this does not matter as the screen data will be correct from 4000. It is only another of the programmers little tricks to try to prevent copying. The call to 0659 at 1C91 loads in a 'headerless' file block, it forms part of the routine for loading a normal block of machine code which as we have seen is at 0656 and simply skips over the part of SP1.COM that loads in a header.

We now move on to a piece of machine code loaded by a BASIC loader that when called loads a 'headerless' block of code. An example of the BASIC loader may look something like:-

```
10 LOAD ""SCRN$
20 CLEAR 64999
30 LOAD ""CODE 65000
40 LET A=USR 65000
```

Obviously the code loaded from address 65000 onwards is not the actual game since there is only 535 bytes left before the end of the Spectrums memory which like that on the Einstein is at 65535 (FFFFhex) The code will in fact load in one or more 'headerless' blocks of machine code that go up to make the game. The code loaded from 65000 may load in a single block as in our earlier example or it may be more complicated and load in several blocks of code. There may also be other things done within this piece of code like , writing to the Spectrums variable area, used in some cases to reposition the pointer to the graphics table, or there may even be a byte written to a specific location that will be checked once the whole game has loaded. In the latter case if the particular byte is not present the game will crash.

The following is an example of a piece of machine code used to load 'headerless' code. For this example I shall use 'THE SNOWMAN' a game from Quicksilva based on the book by Raymond Briggs. I've chosen this program because not only are there two 'headerless' blocks loaded but also because it does a few other things within the machine code loader that we have to use for correct transfer to the Einstein.

First here is the BASIC loader..

```
0 POKE 23693,0: BORDER 0: CLEAR 24999: FOR i=2 TO 6: BEEP .09,i: NEXT
  i: PRINT INK 7; AT 8,5;"THE SNOWMAN IS LOADING"; AT 12,10;"PLEASE
  WAIT";AT 0,0;: LOAD ""CODE
```

We can safely ignore everything in this part of the program except the final statement, LOAD""CODE. If we ran the next block of code through our header reader we would get a start address of 4000 and a length of 1D4C. This means the next block is a screen, however its length is longer than that of the screen area and it therefore overwrites the Spectrums variable area and auto-runs a block of machine code. The piece of code it runs starts at address 5B00. In a Spectrum the screen area finishes at 5AFF and the variable area starts at 5C00. The area from 5B00 to 5BFF is the printer buffer but this is often used to store a piece of loader code. This is true in this case, the code being as follows..

```
ADDRESS  CODE      MNEMONIC          COMMENTS
5B00     DD21A861  LD IX,61A8    ;LOAD 'IX' REG. WITH START ADDRESS.
5B04     11141E    LD DE,1E14    ;LOAD 'DE' REG. WITH LENGTH.
5B07     3EFF      LD A,FF       ;SAYS BLOCK TO LOAD IS MACHINE CODE.
5B09     37        SCF           ;SET CARRY FLAG
5B0A     CD5605    CALL 0556     ;LOAD A BLOCK OF CODE.
5B0D     DD21B0B3  LD IX,B3B0    ;LOAD 'IX' REG. WITH START ADDRESS.
5B11     115046    LD DE,4650    ;LOAD 'DE' REG. WITH LENGTH.
5B14     3EFF      LD A,FF       ;SAYS BLOCK TO LOAD IS MACHINE CODE.
5B16     37        SCF           ;SET CARRY FLAG.
5B17     CD5605    CALL 0556     ;LOAD A BLOCK OF CODE.
5B1A     ED5F      LD A,R        ;LOAD 'A' REG. WITH CONTENTS OF 'R' REG
5B1C     A7        AND A         ;RESETS CARRY FLAG.
5B1D     28FB      JR Z,5B1A     ;JUMP BACK TO 5B1A IF ZERO FLAG SET.
5B1F     32795C    LD (5C79),A   ;LOAD 5C79 WITH CONTENTS OF 'A' REG.
5B22     C3627E    JP 7E62       ;JUMP TO ADDRESS 7E62.
```

To transfer 'THE SNOWMAN' we must modify SP1.COM using the above information. We can again use the JETPAC loader and modify it to our requirements.

What we would do is load SP1.COM into the Einstein and go into MOS as in previous examples. Then using the Modify command type..

```
M 1969 <E>

010040114C1DCD5606 <E>

01A86111141ECD5906 <E>

01B0B3115046CD5906 <E>

ED5FA728FB32795CC3 <E>

627E. <E>
```

I have put line spacing to make it easier to read. Remember to finish the modify command with a full stop.

This code gives us an assembly listing that would read.

```
 ADDRESS  CODE      MNEMONIC        COMMENTS
  1966    CD1306    CALL 0613     ;LOAD A 'BASIC' BLOCK.
  1969    010040    LD BC,4000    ;LOAD 'BC' REG. WITH START ADDRESS.
  196C    114C1D    LD DE,1D4C    ;LOAD 'DE' REG. WITH LENGTH.
  196F    CD5606    CALL 0656     ;LOAD A BLOCK OF CODE.
  1972    01A861    LD BC,61A8    ;LOAD 'BC' REG. WITH START ADDRESS.
  1975    11141E    LD DE,1E14    ;LOAD 'DE' REG. WITH LENGTH.
  1978    CD5906    CALL 0659     ;LOAD A 'HEADERLESS' BLOCK OF CODE.
  197B    01B0B3    LD BC,B3B0    ;LOAD 'BC' REG. WITH START ADDRESS.
  197E    115046    LD DE,4650    ;LOAD 'DE' REG. WITH LENGTH.
  1981    CD5906    CALL 0659     ;LOAD A 'HEADERLESS' BLOCK OF CODE.
  1984    ED5F      LD A,R        ;LOAD 'A' REG. WITH CONTENTS OF 'R' REG
  1986    A7        AND A         ;RESETS CARRY FLAG.
  1987    28FB      JR Z,1984     ;JUMP BACK TO 1984 IF ZERO FLAG SET.
  1989    32795C    LD (5C79),A   ;LOAD 5C79 WITH CONTENTS OF 'A' REG.
  198C    C3627E    JP 7E62       ;JUMP TO ADDRESS 7E62.
```

Here we can see that in order to get a successful transfer we have had to include the code that follows the last 'headerless' load up to and including the jump to the start of the game. We have also changed the LD IX instructions to LD BC instructions. If you have 'THE SNOWMAN' you can now load it to the Einstein as previously demonstrated or if you haven't got a copy you can save the modified SP1.COM for use at a later date. To save the modified SP1.COM, press CTRL/BREAK to return to DOS and type SAVE 48 SPSNOW.COM. Now when you get a copy of 'THE SNOWMAN' you can simply type, from DOS.. SPSNOW <E>, select 'I' from the menu then load in the game.

Though this is a little more complicated there is worse to follow agh! The Spectrum programmers have grown increasingly devious in their search for a a program that is impossible to copy. Though they have failed, without a great deal of machine code knowledge, and in some cases, extra hardware, they have come very close.

THE SPECTRUM PROTECTION RACKET

I am not going to try, nor could I hope, to teach you all the ins and outs of programming on the Spectrum. After all what we are trying to do is extend the use of the Einstein. However, a knowledge of the Spectrums variable area and the way BASIC is stored will all go to help us in our aim to transfer more programs to the Einstein.

One of the earliest protection systems used was to include as the first statement in the BASIC loader:-

```
        POKE 23659,0
```

Poking 23659 with a zero will disallow the use of the lower two lines of the screen for printing messages. So any error message, such as BREAK, will crash the program. This method of protection can only be used for short programs since SCROLL and CLS will also crash the system.

Another method is to Poke locations 23613 and 23614 that go to make up ERR SP. If there is an error, say BREAK, the processor will look at the address pointed to by the contents of ERR SP, there it will find another address, which should be in the stack, to which it will jump. The programmer trying to protect his program will point the ERR SP at an address holding zeros and consequently the processor will jump to the start of memory resetting the machine.

The first of the two methods can sometimes be overcome by MERGE, if you have such a problem give it a try. You may have to enter something like CLEAR 24400:MERGE "" this will cause an error to occur when the merge is complete. If you find, when listing the program, that all you get is a blank screen or just 0> then you will first need to try changing the INK and PAPER. If you still have just the 0 line number or a 0 line number and perhaps a message, then you need to change the line number so that you can read it.

The line number is held in the first two bytes of a BASIC program line, BASIC starts at address 23755 but since the line number is held in reverse order we have to POKE 23576,1 This will now turn the zero line number into line 1. You can now edit or remove this line. Probably the rest of the program listing will be in whatever the current paper colour is and you will only be able to read the line number. To change this so that you can read the line, bring the line down to the edit area and move the cursor along until it disappears. The cursor will resist moving at first then it will vanish, at this point go into 'E' mode and press a colour key. Hey presto the line will appear in that colour.

The second method, as far as I can tell, is used in that particular form on earlier games only, but is used extensively in programs that load from the start of the screen right up to and sometimes past the top of memory. since the stack is overwritten an error occurs and the processor collects the value from ERR SP, at the address pointed to by ERR SP there is the start address of the game. Sometimes an address in the Spectrums ROM will be pointed to by ERR SP, the processor then jumps to the BASIC statement identified by a combination of pointers in the variable area.

The following is a list of relevant variable addresses in the Spectrum.

23606 Holds 256 less than the address of the character set this is
23607 often altered in a program.

5C3D This is the ERR SP that holds the address of the item on machine
5C3E stack to be used as error return.

5C42 Line to be jumped to.
5C43

5C44 Statement number in line to be jumped to.

5C45 Line number of statement currently being executed.
5C46

5C47 Statement number in line being executed.

5C53 Address of BASIC program. Often altered.
5C54

5C55 Address of next line in program.
5C56

5C5D Address of next character to be interpreted.
5C5E

5C6B Holds the number of lines (including one blank line) in the lower part of the screen. Set to zero to cause a crash.

This is only part of the Spectrums variable area but there should be enough information here to check most programs.

USING CLOAD

Using CLOAD.COM is quite straightforward, remember though that programs transferred to disc using CLOAD.COM will not run on the Einstein whether or not the Speculator is connected. CLOAD.COM is used for transferring Spectrum programs to disc so that you can then interrogate the software using the MOS Tab function or a disassembler. You can also transfer ASCII files created by a word processor such as TASWORD and use them on an Einstein word processor.

The format in either case is...

```
CLOAD d:en: f:sn
```

Where d = Einstein drive number program is to be saved to.
en = The name you wish to call the program on the Einstein.
f = Spectrum filetype... 0 BASIC program
1 Number array
2 Character array
3 Machine code
sn = The name of the Spectrum program.

To use the program the Speculator must be connected, place your disc with CLOAD.COM on in drive 0 press CTRL/BREAK and once in DOS use the format as shown above. If for instance we wished to transfer the BASIC loader in JETPAC to drive 1 under the name LUPINS we would type..

```
CLOAD 1:LUPINS 0:JETPAC <E>
```

press play on the cassette player with JETPAC rewound to the beginning and the BASIC loader for JETPAC will be saved to drive 1 as LUPINS.

You can now examine LUPINS by placing the disc with LUPINS on in drive 0 and from DOS typing.. LOAD LUPINS. <E> and displaying the file with the T function from MOS. However you will find the results in the main, unrewarding.

If we are to make any sense of this we need to know something about how Spectrum BASIC is held in memory.

As with XBAS the keywords such as CLS, LOAD etc. are tokenised, that is, a single byte is used to represent each Keyword.

In Spectrum BASIC statement lines are held in the format below..

The first two bytes hold the line number, the first byte holding the 'high byte', the second the 'low byte'.

The third and fourth byte hold the 'remaining length', in this case the 'low byte comes before the 'high byte'. The 'remaining length' is the number of bytes counted from the fifth byte to the final 'ENTER' (0D) character inclusively.

The BASIC itself now follows. ASCII is used for the alphanumeric characters and Sinclair codes are used for the tokens and a few other characters. Each statement within a line is seperated by a colon (3A) and a line is always ended with an 'ENTER' (0D).

Decimal numbers occur in two forms, first as its ASCII form ie 23555 would be 5051535353, followed by the 'NUMBER' character (0E) and the floating-point form for integers in the range -65535 to +65535, which gives us for 23555, 0000035C00. This means that for all numbers we get an extra six bytes in memory for every decimal number in a BASIC program.

With this information we can write a simple Spectrum BASIC de-tokeniser, the program that follows runs on the Einstein, it is very crude but for our purposes it will suffice. First of all you must load your Spectrum BASIC loader to disc using CLOAD.COM then using the de-tokeniser you can look at the BASIC even if it has been protected.

Type in the following XBAS program into your Einstein and save as STOK.

```
  10 CLEAR &8000
  20 INPUT"ENTER> FILE NAME.OBJ ";Q$
  30 DIMZ$(92):AG=1
  40 LOAD Q$
  50 LO=&8000:LOO=&A000:FOR F=0TO90:READZ$(F):NEXT
  60 GOSUB 5000
  70 GOSUB 6000
  80 FOR A=QTO Q+X
  90 IF PEEK(A)=255 AND PEEK(A+1)=255 AND PEEK(A+2)=255THEN PRINTLOO
     :STOP
 100 D=PEEK(A):IF D>164THEN GOTO7000
 110 IF D=14THEN A=A+5:GOTO160
 120 IFD=13THEN DOKELOO,&0A0D:LOO=LOO+2:CO=0:AG=AG+1:GOSUB9000:GOTO
     160
 130 IFD<32 OR D>122THEN POKELOO,63:LOO=LOO+1:GOTO160
 140 POKE LOO,D:LOO=LOO+1:CO=CO+1
 150 IF CO=>120THEN AG=AG+1:CO=0:GOSUB9000
 160 REM
 170 NEXT
 180 LO=LO+X
 190 AG=AG+1:GOSUB9000:GOTO60
4010 DATA "RND","INKEY$","PI","FN","POINT","SCREEN$","ATTR","AT",
     "TAB","VAL$","CODE","VAL","LEN","SIN","COS","TAN","ASN"
4020 DATA "ACS","ATN","LN","EXP","INT","SQR","SGN","ABS","PEEK","IN"
     ,"USR","STR$","CHR$","NOT","BIN","OR","AND","<=",">=","<>"
```

```
4030 DATA "LINE","THEN","TO","STEP","DEF FN","CAT","FORMAT","MOVE",
     "ERASE","OPEN£","CLOSE£","MERGE",VERIFY","BEEP","CIRCLE"
4040 DATA "INK","PAPER","FLASH","BRIGHT","INVERSE","OVER","OUT",
     "LPRINT","LLIST","STOP","READ","DATA","RESTORE","NEW"
4050 DATA "BORDER","CONTINUE","DIM","REM","FOR","GO TO","GO SUB",
     "INPUT","LOAD","LIST","LET","PAUSE","NEXT","POKE","PRINT"
4060 DATA "PLOT","RUN","SAVE","RANDOMIZE","IF","CLS","DRAW","CLEAR",
     "RETURN","COPY"
5000 REM
5010 DOKELOO,AG+48:LO=LO+2:LOO=LOO+2:POKELOO,82:POKELOO+1,69:POKELOO
     +2,77:LOO=LOO+3::RETURN
6000 X=DEEK(LO):LO=LO+2:Q=LO:RETURN
7000 D=D-165:FORG=1TOLEN(Z$(D)):B$=MID$(Z$(D),G,1):POKELOO,ASC(B$):
     LOO=LOO+1:CO=CO+1:NEXTG:
7010 IF CO=>120THEN AG=AG+1:CO=0:GOSUB9000
7020 GOTO160
9000 DOKELOO,&0A0D:LOO=LOO+2
9005 :DOKELOO,AG+48 :LOO=LOO+2
9010 :DOKELOO,AG+48 :LOO=LOO+2
9020 POKELOO,82:POKELOO+1,69:POKELOO+2,77:LOO=LOO+3:CO=CO+3
9030 RETURN
```

No error checking has been included in the de-tokeniser, also the program is only suitable for short Spectrum BASIC loader programs. This is because it reads the program into memory begining at &8000 and then writes the de-tokenised program into memory from location &A000, thus if the program is too long the de-tokenised program will overwrite the latter portion of the tokenised program.

USING THE DE-TOKENISER

We will now try the de-tokeniser out on SPITFIRE 40 from MIRRORSOFT. If you load the BASIC loader into a Spectrum and try to list it the machine will crash, you can MERGE the program but we want to test our de-tokeniser.

So connect your cassette to the Speculator rewind the SPITFIRE 40 tape to the start of the program and load the BASIC to the Einstein disc by typing..

```
        CLOAD 0:SPITFIRE 0:SPITFIRE <E>
```

press play on your cassette, you should get the response:

```
        PROGRAM: SPITFIRE
        LOADING
```

once the program has been saved to disc stop the tape. Do a directory and you will see you have a file SPITFIRE. on your disc. Type..

```
        REN SPITFIRE. TO SPITFIRE.OBJ <E>
```

Load up XBAS and run STOK. When prompted enter the filename SPITFIRE.OBJ after a few seconds a figure will be printed on the screen followed by BREAK IN 90. The figure printed to the screen is in this case 41088, this is the end of the BASIC program so when we save the file it is from &A000 to this figure as..

```
        SAVE "SPIT.OBJ",&A000,41088
```

Now you must rename the 'SPIT' file with.

```
REN "SPIT.OBJ" TO "SPIT.ASC" <E>
```

type. NEW <E> followed by. LOAD "SPIT.ASC" <E>
LIST the program and you should get a first line that is..

```
1 REMBORDER1:PAPER1:INK1:CLS:POKE23659,0:LOAD""CODE16384:RANDOMIZE
  USR23296
```

The line numbers in the program are those put in by the de-tokeniser and are for our convenience only, most of these type of BASIC loaders have a line number 0. In our line 1 we have poke 23659,0 this as we have seen causes a crash on the Spectrum if any attempt to BREAK the program is made. You will see in line 1 that the code is loaded from 16384 with: LOAD""CODE16384 followed by a call to 23296, which is 5B00hex. So let's have a look at the header readings for SPITFIRE 40..

```
        FILENAME: SPITFIRE
        TYPE    : PROGRAM                    THE 'BASIC' LOADER.
        TOTAL LENGTH: 006D BYTES
        PROGRAM LENGTH: 0054 BYTES
        RUNS FROM LINE: 0

        FILENAME: SCREEN
        TYPE    : BYTES                      THE 'LOADING' SCREEN.
        START ADDRESS: 8E20
        LENGTH: 1B12 BYTES
```

The main block of code for SPITFIRE 40 is a headerless file. If you look at the header reading for the loader screen you will see first that the start address is 8E20 which is 36384 decimal, this is taken care of in the BASIC loader by: LOAD""CODE16384. In other words this block of code loads from the start of the Spectrum screen area.

The next point is its length, a Spectrum screen is 1B00hex bytes in length (6912 decimal). However our header reading tells us that we have an extra 12hex (18dec) bytes. These are the bytes that will load in the headerless main block of code. Thus we have in our line 1... RANDOMIZE USR 23296. This calls the 12hex block of bytes which starts at 5B00hex (23296dec).

To find out what is in these bytes we can load this part of the program to disc with CLOAD.COM..

```
        CLOAD 0:SCREEN 0:SCREEN <E>

Type..  LOAD SCREEN. <E>
        MOS <E>
        T1F19 1F2A<E>
```

This will display the final 18bytes of SCREEN which form the loader code for the main code block of SPITFIRE 40. These bytes and their assembler equivalent are..

```
DD210060 LD IX,6000   ;START ADDRESS.
110096   LD DE,9600   ;LENGTH.
3EFF     LD A,FF
37       SCF
CD5605   CALL0556     ;LOAD IT.
C30060   JP 6000      ;JUMP TO START OF GAME.
```

All we have to do now is to modify SP1.COM as previously shown substituting 010060 for the DD210060 and CD5906 for the CD5605 above. As before there is no need for the 3EFF or the 37 and we simply finish the modification with C30060. The full modification for SPITFIRE 40 is in the rear section of this book.

MODIFYING CLOAD.COM TO READ HEADERS

It is possible, by a little piece of code modification, to use CLOAD.COM as a header reader. By using the modified CLOAD.COM together with the Spectrum de-tokeniser, it becomes feasible to do without a Spectrum. This is still only true for the simpler loaders but with a bit of machine code knowledge and a dis-assembler on the Einstein all but those programs using speed/jerky loaders may be transferred.

To create a header reader from CLOAD.COM insert a disc containing CLOAD.COM in drive 0 and type..

```
        LOAD CLOAD.COM <E>
        MOS <E>
        M01A0 <E>
        06031113041A67CF <E>
        A91310F6C34301. <E>
```

Press CTRL/BREAK and type..

```
SAVE 22 HEADEX.COM
```

When using HEADEX.COM the format is the same as that for CLOAD.COM i.e.

```
        HEADEX 0:en 0:sn
```

only this time 'en', 'sn', the Einstein drive number and the Spectrum filetype are all irrelavent since nothing is saved to disc. What happens is that the HEADEX.COM will continuously read data coming in through the Speculator and display header information as it comes in. Here are the readings from 'Way of the exploding fist".

```
          Program:FIST
          013F 000A 013F                THE 'BASIC' LOADER.
          Bytes:t
          1B00 8000 8000                THE 'LOADING' SCREEN.
          Bytes:k
          9800 6000 8000                THE MAIN BLOCK OF CODE.
```

the parts we are interested in are the first two numbers of each reading. The first number is the length and the second number is the start address. The third number is for BASIC programs only and is the total length of the program.

THE JOYSTICK OPTION

For most of the programs in the SP1.COM menu there is the option of selecting a joystick mode. Pressing Function key 3 selects joystick and Function key 4 selects keyboard.

If you have selected joystick, the program reads Analogue port A and the position of the joystick is mapped against the keyboard option keys for that particular game. The information that tells the program what keys the joystick position is representing is held in the last five bytes of each section of loader code in SP1.COM, the fifth byte being the byte immediately preceding the CD4506 (BASIC block loader), of the next game.

These five bytes follow FF which is used as a delimiter. Here is the block of code for JETPAC with the five bytes, used for the joystick, indicated.

```
1966 CD 13 06 CD 45 06 01 00 60 11 00 20 CD 56 06 01 M..ME...`.. MV..
1976 80 5B 11 0F 00 CD 56 06 01 B0 5C 11 01 00 CD 56 .[...MV..0\...MV
1986 06 01 78 5C 11 02 00 CD 56 06 C3 00 60 11 01 16 ..x\...MV.C.`...
1996 00 0D 20 4A 45 54 50 41 43 20 16 03 0C 20 55 6C .. JETPAC ... Ul
19A6 74 69 6D 61 74 65 20 11 00 FF 02 08 10 01 03 timate .......
```

These bytes in themselves tell us very little and will not help us give a joystick option to the new games we wish to transfer. We need to know exactly which keys the bytes are meant to represent and also whether they are for Up, Down, Left, Right or Fire. Here is a complete list of the keyboard representation, note also the five bytes are in the order... UP - LEFT - RIGHT - FIRE - DOWN

SPECTRUM KEY	VALUE IN SP1	SPECTRUM KEY	VALUE IN SP1
A	01	U	1D
B	27	V	20
C	18	W	0A
D	11	X	10
E	12	Y	25
F	19	Z	08
G	21	SPACE	07
H	26	CAP/SHIFT	00
I	15	SYM/SHIFT	0F
J	1E	ENTER	06
K	16	0	04
L	0E	1	03
M	17	2	0B
N	1F	3	13
O	0D	4	1B
P	05	5	23
Q	02	6	24
R	1A	7	1C
S	09	8	14
T	22	9	0C

Now we have this information all we have to do is look at what keys are used in the program we intend to transfer and substitute the value for that key into its appropriate position in the five bytes at the end of the loader in SP1.COM. If a key is not used then that position in the five bytes is filled with FF. You may have found that a little confusing so let's go back to 'FEUD' and transfer it again only this time we shall also include a joystick option.

As before type, from DOS...

```
LOAD SP1.COM <E>
MOS <E>
M1969 <E>
018D6011727FCD5606C3B5A2. <E>
```

This has now modified the loader section of JETPAC to allow us to transfer 'FEUD' now to obtain correctly functioning joystick option once the transfer is complete, we must modify the five bytes from 19B0 in the order Up, Left, Right, Fire, Down. So first of we find out what keys are used in 'FEUD', and convert them using the table shown. These turn out to be...

UP	Q	= 02
LEFT	O	= 0D
RIGHT	P	= 05
FIRE	SPACE	= 07
DOWN	A	= 01

So type..

```
M19B0 <E>
020D050701. <E>
G0100 <E>
```

and load the 'FEUD' tape to the Speculator as normal. You can now use a joystick if you press Function key 3.

THE END

By now you may have gathered that there may be quite a lot of work involved with the transfer of some programs. Some games that appear to be straight forward have very complex load sequences, modifying the code in various ways, others use special OUT instructions that upset the Speculator or perhaps using some ROM calls that have not been simulated by the Speculator and its code. Unless you are heavily into hacking these programs are really more trouble than they're worth.

Many of the newer games on the Spectrum market have special loaders such as SPEEDLOCK which have a jerky sound similar to that heard on Daleys Decathalon, the variations on this are many and each requires hacking and modification to SP1.COM beyond the scope of this book.

SPECULATOR - THE GAMES

This section contains the necessary modification details for transferring a number of Spectrum programs to the Einstein. I have included the joystick option code where possible and each piece of code has a short resume of what the game is like. Some programs require you, when making selections prior to the start of the game, to press the space key between each selection.

For each game conversion type from DOS...

```
                    LOAD SP1.COM <E>
                    MOS <E>
                    M 196n <E> where n is 9 or C.
```

followed by the code, completing with a full stop and ENTER. Where a joystick code is given simply type, after the last ENTER of the loader code....

```
                    M 19B0 <E>
```

and type in the five bytes, a full stop and ENTER.

If you have any difficulties refer to the example transfer on page 11.

LORDS OF MIDNIGHT from BEYOND SOFTWARE

The classic graphic adventure with multi-role involvement.

```
          Modify SP1.COM from 1969..
               01004011581BCD5606 <E>
               01B05C114FA3CD5906 <E>
               31355CC37560. <E>
```

Type G0100 select 'I' from the menu and load LORDS OF MIDNIGHT.
Not joystick.

NONTERRAQUEOUS from MASTERTRONIC

Over 1000 screens to explore in this arcade adventure, the follow up to this game was SOUL OF A ROBOT which is available for the Einstein.

```
          Modify SP1.COM from 1969..
               01004011001BCD5906 <E>
               019C6311C486CD5906 <E>
               C39C63. <E>
```

For joystick modify SP1.COM from 19B0..

```
               1C23140424. <E>
```

Type G0100 select 'I' from the menu and load NONTERRAQUEOUS.
For joystick, select 'PROTEK' option from game menu and press function key 3.

AD ASTRA from GARGOYLE GAMES

This is a space shoot 'em up game.

```
          Modify SP1.COM from 196C..
               01B45F11F82FCD5606 <E>
               01309011BE66CD5606 <E>
               C3E880. <E>
```

For joystick modify SP1.COM from 19B0..

```
               05081F070E. <E>
```

Type G0100 select 'I' from the menu and load AD ASTRA.

RASTERSCAN from MASTERTRONIC

An animated adventure with no text, using your spherical bouncing robot you must solve problems and repair the RASTERSCAN.

```
          Modify SP1.COM from 196C..
               01407611C089CD5606 <E>
               3EA93210803E913211 <E>
               80C30080. <E>
```

For joystick modify SP1.COM from 19B0..

```
               020D050F01. <E>
```

Type G0100 select 'I' from the menu and load RASTERSCAN.

FEUD from BULLDOG SOFTWARE

An arcade adventure, you take the role of Learic moving through the kingdom collecting ingredients to make spells to use against the opposing wizard, Leanoric.

```
          Modify SP1.COM from 1969..
               018D11729FCD5606 <E>
               C3B5A2. <E>
```

For joystick modify SP1.COM from 19B0..

```
               020D050701. <E>
```

Type G0100 select 'I' from the menu and load FEUD.

RESCUE from MASTERTRONIC

The space pirates are after the ultimate weapon can you save it and the scientists in this space arcade adventure.

```
          Modify SP1.COM from 1969..
               0100401178BFCD5606 <E>
               C380AF. <E>
```

Type G0100 select 'I' from the menu and load RESCUE.
For joystick press function key 3 then select 'define keys' from the games menue then for each direction move the joystick accordingly returning it to the centre between selections.

S.O.S. from MASTERTRONIC

Save Sydney the Droid in this 3D arcade adventure.

```
          Modify SP1.COM from 196C..
               01FA6711B888CD5606 <E>
               C30080. <E>
```

For joyatick modify SP1.COM from 19B0..

```
               02271F0401. <E>
```

Type G0100 select 'I' from the menu and load S.O.S.

TERMINUS from MASTERTRONIC

Five different droids with five different functions are used in this arcade space adventure.

```
          Modify SP1.COM from 196C..
               01BC5F117E29CD5606 <E>
               01548D11AC72CD5606 <E>
               C30089. <E>
```

For joystick modify SP1.COM from 19B0..

```
               020D051701. <E>
```

Type G0100 select 'I' from the menu and load TERMINUS.

SPITFIRE 40 from MIRRORSOFT

Fly a Spitfire in training or combat.

```
          Modify SP1.COM from 196C..
               010060110096CD5606 <E>
               C30060. <E>
```

For joystick modify SP1.COM from 19B0..

```
               1C23140424. <E>
```

Type G0100 select 'I' from the menu and load SPITFIRE40.
For joystick, select 'CURSOR' option from game menu and press function key 3.

THRUST from FIREBIRD

Thrust is a Penetrator type game.

```
          Modify SP1.COM from 196C..
               0178FF111E00CD5606 <E>
               01004011E8BDCD5606 <E>
               014A31116000CD5606 <E>
               C300FA. <E>
```

For joystick modify SP1.COM from 19B0..

```
               1501090517. <E>
```

Type G0100 select 'I' from the menu and load THRUST.

CAVELON from OCEAN

As a crusader you must collect the parts to the castle door to escape the saracens.

```
          Modify SP1.COM from 196C..
               015078117869CD5606 <E>
               3E02326B5C <E>
               C394EA. <E>
```

For joystick modify SP1.COM from 19B0..

```
               1C23140424. <E>
```

Type G0100 select 'I' from the menu and load CAVELON.
For joystick, select 'PROTEK' option from game menu and press function key 3.

GREMLINS from ADVENTURE INTERNATIONAL

A graphic adventure based on the hit film.

```
        Modify SP1.COM from 196C..
            010060110B9FCD5606 <E>
            C30060. <E>
```

Not joystick.

LUNAR JETMAN from ULTIMATE

A space arcade adventure in which you need fast shooting and quick thinking.

```
        Modify SP1.COM from 196C..
            01FF7F11017CCD5606 <E>
            3EE932B05C3E343278 <E>
            5C3E8332795C21FF7F <E>
            017C00AFED672310FB <E>
            0D20F8C30080. <E>
```

```
For joystick modify SP1.COM from 19B0..
            1C23140424. <E>
```

Type G0100 select 'I' from the menu and load LUNAR JETMAN.
For joystick, select 'CURSOR' option from game menu and press function key 3.

WHEELIE from MICROSPHERE

Ride your motorcycle through the caverns, if you make it to the end race the ghost rider back again.

```
        Modify SP1.COM from 1969..
            CD4506C3115D. <E>
```

```
For joystick modify SP1.COM from 19B0..
            0220270401. <E>
```

Type G0100 select 'I' from the menu and load WHEELIE.

THE SNOWMAN from QUICKSILVA

In this game based on the Raymond Briggs story you have to collect the snow to build a snowman.

```
        Modify SP1.COM from 1969..
            010040114C1DCD5606 <E>
            01A86111141ECD5906 <E>
            01B0B3115046CD5906 <E>
            ED5FA728FB32795C <E>
            C3627E. <E>
```

```
For joystick modify SP1.COM from 19B0..
            1C23140424. <E>
```

Type G0100 select 'I' from the menu and load THE SNOWMAN.
For joystick, select 'CURSOR' option from game menu and press function key 3.

GRAND PRIX SIMULATOR from CODE MASTERS

A motor racing simulator with overhead view.

```
        Modify SP1.COM from 196C..
            0100601100A0CD5606 <E>
            CD4506C330CA. <E>
```

Type G0100 select 'I' from the menu and load GRAND PRIX SIMULATOR.
For joystick press function key 3 then select 'define keys' from the games menue then for each direction move the joystick accordingly returning it to the centre between selections.

DESERT RATS from CCS

A war game simulation using historical scenarios.

```
        Modify SP1.COM from 196C..
            01446111 5D4DCD5606 <E>
            0159C911A736CD5606 <E>
            C37C76. <E>
```

Type G0100 select 'I' from the menu and load DESERT RATS.
Not joystick.

TRANZAM from ULTIMATE

Race across America in the 21st century collecting cups and avoiding the killer cars.

```
        Modify SP1.COM from 196C..
            01005F110021. <E>

        Modify SP1.COM from 1991..
i             805B. <E>
```

```
For joystick modify SP1.COM from 19B0..
            02271FFF01. <E>
```

Type G0100 select 'I' from the menu and load TRANZAM.

COOKIE from ULTIMATE

Collect the ingradients to bake a cake.

```
        Modify SP1.COM from 196C..
            01005F110021. <E>

        Modify SP1.COM from 1991..
            805B. <E>
```

```
For joystick modify SP1.COM from 19B0..
                1C23140424. <E>
```

Type G0100 select 'I' from the menu and load COOKIE.
For joystick, select 'CURSOR' option from game menu and press function key 3.

---

180 DARTS from MASTERTRONIC

Probably the best darts simulation on the market.

```
        Modify SP1.COM from 196C..
            01006011C49FCD5606 <E>
            C3728D. <E>
```

```
For joystick modify SP1.COM from 19B0..
            1C23140424. <E>
```

Type G0100 select 'I' from the menu and load 180 DARTS.
For joystick, select 'CURSOR' option from game menu and press function key 3.

ROBIN OF SHERWOOD from ODIN

Robin must collect his weapons before he can defeat the sheriff.

```
        Modify SP1.COM from 196C
            0100601100A0CD5606 <E>
            C335C4. <E>
```

```
For joystick modify SP1.COM from 19B0..
            1C23140424. <E>
```

Type G0100 select 'I' from the menu and load ROBIN OF SHERWOOD.
For joystick, select 'CURSOR' option from game menu and press function key 3.

RED ARROWS from DATABASE

Join the RED ARROWS team in formation flying.

```
        Modify SP1.COM from 196C..
            018265111E00CD5606 <E>
            016D6011929FCD5906 <E>
            C300E8. <E>
```

Type G0100 select 'I' from the menu and load RED ARROWS.
For joystick press function key 3 then select 'define keys' from the games menu then for each direction move the joystick accordingly returning it to the centre between selections.

SWEEVO'S WORLD from GARGOYLE GAMES

A madcap 3D arcade adventure.

```
        Modify SP1.COM from 196C..
            0190FB113D04CD5606 <E>
            01E06011B082CD5606 <E>
            C3E060. <E>
```

```
For joystick modify SP1.COM from 19B0..
            05020E1701. <E>
```

Type G0100 select 'I' from the menu and load SWEEVOS WORLD.

DANGERMOUSE IN DOUBLE TROUBLE from CREATIVE SPARKS

Guide Dangermouse through this arcade adventure.

```
          Modify SP1.COM from 196C..
              01D45D112BA2CD5606 <E>
              C3D45D. <E>
```

For joystick modify SP1.COM from 19B0..

```
              1C23140424. <E>
```

Type G0100 select 'I' from the menu and load DANGERMOUSE.
For joystick, select 'CURSOR' option from game menu and press function key 3.

MASTER OF MAGIC from MASTERTRONIC

A graphic adventure.

```
          Modify SP1.COM from 1969..
              01DCFF112200CD5606 <E>
              01004011001BCD5606 <E>
              01E060118099CD5606 <E>
              C3A08C. <E>
```

For joystick modify SP1.COM from 19B0..

```
              0A0D051709. <E>
```

Type G0100 select 'I' from the menu and load MASTER OF MAGIC.

WAY OF THE EXPLODING FIST from MELBOURNE HOUSE

A Kung-Fu classic.

```
          Modify SP1.COM from 196C..
              010060110098CD5606 <E>
              C30088. <E>
```

Type G0100 select 'I' from the menu and load WAY OF THE EXPLODING FIST
For joystick press function key 3 then select 'define keys' from the games menu then for each direction move the joystick accordingly returning it to the centre between selections.

NOTES