

Editor: G.A.BOBKER of ZX-GUARANTEED

RAMBO-v-COMMANDO

WARNING: Do NOT buy both of these as they are identical. ONLY real difference is Ocean Software paid Stallone a fortune just to get permission to use his name! RAMBO by Ocean, has better graphics. COMMANDO is by Elite and by today's standards graphics are very poor but is a slightly better game.

Sinclair has officially denied that they are developing a larger computer based on the 68000. This is good news as it should mean it will be available later this year!

Still difficult to assess if the Sinclair 128 is liable to catch on. Number of people saying they are eagerly awaiting it is only matched by people saying they don't see the point as Spectrum has so much going for it. Might be prudent to wait for the "rush" of expected "add-ons" to "upgrade" to a 128K...without the silly numerical pad which isn't really wanted anyway.

M-DOC REVEIW by G.TODD of Crawley, W.Sussex

Software/Utility reveiws are only printed if generally several members have made similar comments about the item. Only good reports have been received about it. Over to G.T. M-DOC is a "Microdrive Doctor" which also includes a program for Basic Files. Basically it allows user to find, load and repair bad sectors and thereby Load corrupted Files. It is fairly User Friendly. Includes a small Dissassembler/Monitor type sector editor which allows characters in Decimal, Hex or ASCII to be inserted for repairing/rewriting. It has a "Filter" option to allow (say) Tasword Two text files to be rewritten and corrected Documentation is well written but not extensive. It explains how Microdrive saves programs and therefore how to manipulate them. Sold by: 7-Star Publishing, 34 Squirrel Rise, Marlow, Bucks. SL7 3PN £6.95 for M-DOC & B-DOC, or just £4.95 for M-DOC. G.TODD added that it has saved several hours work when a Tasword File which had been corrupted was "rescued" by this program, and considers it well worth the outlay. (MY COMMENT: I like the option which allows you to mark a sector "unreliable, do not use" as often I find I cannot Save/Verify a short program even when plenty of K left due to a sector having gone faulty).

QUILL TRANSFER by R.BAILEY of Thornlie, West Australia

This transfer routine applies to version AD6 only. The short Code after the main block is a red herring and the POKES jump past a Call to Loading routine.

1. Load the main long block of code by;
CLEAR 54999: LOAD "" CODE 55000
2. Save to Microdrive by; SAVE*"m";1;"QUILLc" CODE 55000,10536
3. Enter this Basic Loader;
10 CLEAR 24999: LOAD*"m";1;"QUILLc" CODE 55000
20 POKE 58519,195: POKE 58520,181: POKE 58521,228:
POKE 58558,0: POKE 58559,0: POKE 58560,0:
POKE 58561,0: POKE 58562,0: POKE 58563,0
30 GO TO USR 58500

Save to Microdrive by; SAVE*"m";1;"QUILL" LINE 10

FIGHTER PILOT TRANSFER by D.WHITTALL, Bahrain

This is for the version with a one part machine-code block with start 16384, length 49140. Dave found that the actual program is only 33184 Bytes long plus 6912 for screen. Rest is padding to make it appear longer. Initial screen part can be ignored as a copy of the screen is held in the program.

1. Enter CLEAR 24383: LOAD "FPC" CODE 24384 then play in from beginning of tape. (The short Basic will be ignored).
2. Save part required by entering;
CLEAR 31999: SAVE*"m";1;"FPC" CODE 32000,33184
4. Enter this Basic Loader;
10 CLEAR 65535: LOAD*"m";1;"FPC" CODE 32000
20 DATA 33,0,125,17,192,93,1,160,129,237,176,195,192,93
30 FOR X=23300 TO 23313: READ A: POKE X,A: NEXT X
40 RANDOMIZE USR 23300
5. Save to Microdrive by; SAVE*"m";1;"FP" LINE 10

Routine loads code higher, chopping off end part we don't want. The Save actually Saves whatever is in 24000 onwards of original. This is Loaded back into 32000, then Block Moved down to 24000 & last three Bytes of Data statement does a JUMP 24000 to start it. The initial screen will look a bit weird, but game runs O.K.

RAMBO TRANSFER by John Croy, Scotland

These is a Speedlock game and has to be converted first by using 007 DE-PULSER. This then gives two Headerless-Files.

1. Make up two False Headers for Machine-Code. First to be for 31327 Bytes, and second for 11628 Bytes.
2. Load the 31327 False Header by LOAD "" CODE 26384 then play in the first long Headerless-File.
3. Save to Microdrive by;
SAVE*"m";1;"RAMBO\$" CODE 26384,6912 <—Omit this if Screen\$
SAVE*"m";1;"RAMBO1" CODE 25500,22211 not required.
4. Load second False Header by LOAD "" CODE 30000 then play in the second long Headerless-File.
5. Save to Microdrive by; SAVE*"m";1;"RAMBO2" CODE 30000,11628
6. Type in this Basic Loader:
10 CLEAR 25499: LOAD*"m";1;"RAMBO\$" CODE 16384
20 LOAD*"m";1;"RAMBO1" CODE 25500
30 LOAD*"m";1;"RAMBO1" CODE 53396
40 RANDOMIZE USR 26368

Save to Microdrive by; SAVE*"m";1;"RAMBO" LINE 10

CHUKIE-EGG II Transfer by S.J.NUTTING of Histon,Cambridge

This game is a Headerless File starting at 16384 and 49152 long. Obviously too long to transfer normally. S.J.Nutting found the part required is from 24770 to 65007. His method loads the File into the ROM at location 7998. This means that the part starting originally at 24770 is now loaded into 16384. Then a Block Move moves this upto place we require of 24770 onwards. Next a CLEAR one Byte below this, CLEAR 24769, then a NEW. All this is done by the Machine-Code routine in line 10. After the Machine-Code NEW the program STILL in 24770 and 40238 Bytes long, is Saved. If you have a Dissassembler, do study method used as it could be used on other 48K programs possibly.

1. Enter the following:
10 CLEAR 65535
20 DATA 221,33,62,31,17,0,192,62,255,55,205,86,5,33,45,221,17,
239,235,253,1,46,157,237,184,33,193,96,34,178,92,195,183,17
30 RANDOMIZE USR 65300
2. RUN the above program then play in the long Headerless-File of your CHUKIE tape. When all loaded in, program will NEW but part required is still in Machine. Therefore Save by;
SAVE*"m";1;"CHUKIEc" CODE 24770,40238
3. Type in this Basic Loader;
10 CLEAR 24769: LOAD*"m";1;"CHUKIEc" CODE 24770: RUN USR 24770
4. Save to Microdrive by; SAVE*"m";1;"CHUKIE" LINE 10

The method used changes two System Variables at 23672 and 23673. (the frame clock variables). This will cause 2 attributes at top of the Dog House to be changed. This does NOT affect the game.

Just in case you want a chance to see the 120 Screens of the game, S.J.N., gives POKE 35453,0 as the Infinite Lives POKE.

POINT DEMO, ETC

This interesting routine by M. RICHARDS of CORNWALL creates an emphasised Print format and demonstrates use of the POINT command. Each character is displayed on bottom of screen then thickened by line 60. Line 80 transfers the character to address A.

```
10 CLEAR 64511: LET A=64512
20 FOR I=32 TO 127
30 PRINT AT 21,0; CHR$ I
40 FOR Y=7 TO 0 STEP -1
50 FOR X=6 TO 0 STEP -1
60 IF POINT (X,Y) THEN PLOT (X+1,Y)
70 NEXT X
80 POKE A, PEEK (22432-256*Y)
90 LET A=A+1
100 NEXT Y: NEXT I
110 POKE 23607,251
120 BORDER 1: PAPER 1: INK 6: CLS: LIST
```

The new character set is put high in memory but below UDG area so can still use the UDGs. System Variables held in 23606/7 has to point to start of character set minus 256. Normally 23606=0 and 23607=60.

To use the new characters NEW Spectrum then enter POKE 23607,251. To see comparison of characters could enter this short program:

```
10 POKE 23607,60: LIST
20 POKE 23607,251: LIST
```

(POKE 23607,60 returns to the normal character set.)

MACHINE CODE TO DATA

Can be very useful when you want to include Machine-Code routine but want to load in one block. Main difficulty is that the DATA line must contain sufficient 000's for the Code. Program includes a routine to POKE in the commas to make it easier to do.

```
1 DATA 000,000,000,000,000,000
2 REM set for 6 Bytes
1000 CLS
1010 INPUT "Start & Length?",S,L
1020 LET D=23818
1030 FOR J=S TO S+L-1
1040 LET X=PEEK J
1050 LET C=INT (X/100)
1060 LET W=X-C*100
1070 LET B=INT (W/10)
1080 LET A=INT (X-(B*10)-(C*100))
1090 POKE D,C+48
1100 POKE D+1,B+48
1110 POKE D+2,A+48
1120 LET D=D+4: NEXT J
2000 STOP
8000 REM ZEROS OUT DATA LINE
9000 LET A=0: FOR J=23818 TO 65535
9010 IF PEEK J=13 THEN STOP
9020 POKE J,48: LET A=A+1
9030 IF A=4 THEN POKE J,44:
9040 LET A=0: NEXT J
```

Line 1 must hold 000's for the Code to go into it. Do alter line 2 to remind yourself at a later date, the number of Bytes it's set for. Initially line 1 can be set by typing in blocks of 000's. To make this easier to do type in line 1 thus; 1 REM 00000000000000 etc. Then enter GO TO 9000 & the Commas will be inserted. If you haven't added sufficient 0's, simply Edit down line 1 add in more 0's then again GO TO 9000. When ready, Edit line 1 by changing the REM to be DATA. Location of first Byte in DATA statement is important, hence the CLS to ensure in m/d mode so start of the

DATA info will be at 23818. When program is RAN line 1010 will request Start and Length of the Code you intend transferring to the DATA statement. Enter the start location you have previously put the Machine-Code into, press Enter key, then enter Length of the M/Code. Program will then move the Code into the DATA line. If you try to move more Code than room in the DATA line, program will lock-up.

After the Code put into the DATA line, the rest of the program can be deleted leaving just the DATA statement for you to MERGE into your program (with suitable FOR loop to POKE it in memory).

READING KEYBOARD BY MACHINE-CODE

The 40 Keys are divided into 8 half rows and the selected 5 keys are read into the A register (bits D0 to D5). To read keyboard do a LD A,254 IN A,(xxx) with xxx being half row code required. The keyboard layout and codes are as shown below.

CODE.		16	8	4	2	1	<--- Value.
HEX	DEC	D4	D3	D2	D1	D0	<--- Bit number.
FE	254	V	C	X	Z	C/Shift	
7F	127	B	N	M	S/Shift	Space	
FD	253	G	F	D	S	A	
BF	191	H	J	K	L	Enter	
FB	251	T	R	E	W	Q	
F7	247	5	4	3	2	1	
EF	239	6	7	8	9	0	

EXAMPLE
 KEYS LD A,254
 IN A,(191)
 AND 8
 RET Z
 JR KEYS

In this example, we are reading half row H,J,K,L & Enter keys. The AND 8 strips of D3 and sets Z flag if D3 was zero, which it would be if key was pressed. If J key pressed, we return. If not pressed, program cycles till it is. (All bits in A reg would be high if no key pressed). Note that the AND 8 is checking D3 ONLY and would work on ALL versions of Spectrums. This method is only useful for checking a single key. To check all the half row we would mask off the bottom 5 bits by an AND 31 then a CP 1 for bit 1, or CP 8 for bit D3. A successful compare would set Z flag. Alternative methods to the above will appear in next issue.

If you are thinking of changing to a Disc system it is advisable to find out how many Bytes the Disc interface uses, and where. The OPUS claims to use none (but I suspect it uses a temporary Buffer of 512 Bytes when an Opus command issued). The BETA uses 112 Bytes. The KEMPSTON uses 703 Bytes. The Swedish only CS-DOS uses 1385 Bytes. The BETA, KEMPSTON & CS-DOS like the Microdrive lifts the Basic starts up to accomodate these extra Bytes.SP-DOS system uses a massive 8K at top of memory, OR "mini" 4096 Bytes at around 57000!! Try getting around that in order to transfer! (Owners of INTERFACE 007 should note that if they get OPUS, BETA KEMPSTON Disc that a FREE convertor tape for INT 007 to Disc is available FREE on request). These systems have a thro' connector BUT this does NOT take thro' the NMI pin which is ALWAYS used by any transfer Interface.The BETA is cheapest at present at around £100, but several ads quote special offer for OPUS at £120. This means shortly it'll be £120 anywhere.SP-DOS claims to be fastest TASWORD problems...Switching on a Printer after Loading Tasword can sometimes result in a partial error. Usually text file appears to have dissappeared & only gives "replace word" option. Very annoying after a long text file entered. Can usually rescue file by pressing Break key, then Saving by;
 SAVE*"m";1;"x" CODE 32000,L with L being about 4K for each page you had in the text. If you can't remember, make L 16384. Reset Spectrum and reload Tasword, then load File "x".

SPECTRUM+ MANUAL. This does NOT give a list of System Variables or the Character Codes. A copy of these are available on request to members....just send a stamped addressed envelope please.

JOHN HUNTER of Coventry disputes my comments about Micronet.I am sticking to my original statements. He also claimed "False-Head" method printed in MDX15 was a direct copy of his letter printed on Micronet. This is NOT possible as write-up was entirely mine and I had NOT seen this on the Micronet.(I sold my Modem).Whilst it is possible that J.HUNTER was originator of this method, it was not known, and acknowledgement couldn't be given.

ALL OF MICRODRIVE-EXCHANGE IS COPYRIGHT AND MUST NOT BE COPIED OR REPRODUCED WITHOUT PRIOR CONSENT OF THE EDITOR (NEVER GIVEN)