TRS-80 SYSTEM 80 VIDEO GENIE PMC-80

Vol. 3, Issue 3, February 1982

5,000,000 BYTES OF HARD DISK STORAGE FOR MODELS 2 & 3

Also in this issue:

PROGRAMMING:

The Theory and Techniques of Sorting — Part 4
Better BASIC Programming — Part 7
Using the Level I Array

SOFTWARE:

- Screen Copy Utility
- Longvars — Use Long Variable Names in your Basic Programs
- Flashing Message Routine
- The Mind Reader
- Australian Income Tax Calculator

MICRO-80

P.O. BOX 213, GOODWOOD, S.A. 5034. AUSTRALIA. TELEPHONE (08) 211 7244. PRICE: AUS. $2.50, N.Z. $4.00, U.K. £1.50

***** ABOUT MICRO-80 *****

EDITOR:	IAN VAGG
ASSOCIATE EDITORS:	
SOFTWARE LEVEL I :	MICHAEL SVENSDOTTER
SOFTWARE LEVEL II:	CHARLIE BARTLETT
HARDWARE :	EDWIN PAAY

MICRO-80 is an international magazine devoted entirely to the Tandy TRS-80 microcomputer and the Dick Smith System 80/Video Genie. It is available at the following prices:

	12 MONTH SUB.	SINGLE COPY
MAGAZINE ONLY	$ 26-00	$ 2-50
CASSETTE PLUS MAGAZINE	$ 65-00	$ 4-00 (cass. only)
DISK PLUS MAGAZINE	$ 125-00	$ 10-00 (disk only)

MICRO-80 is available in the United Kingdom from:
U.K. SUBSCRIPTION DEPT. 24 Woodhill Park, Pembury, Tunbridge Wells, KENT. TN2 4NW

Prices:			
	MAGAZINE ONLY	£ 16-00	£ 1-50
	CASSETTE PLUS MAGAZINE	£ 43-60	N|A
	DISK PLUS MAGAZINE	£ 75-00	N|A

MICRO-80 is available in New Zealand from:
MICRO PROCESSOR SERVICES, 940A Columbo Street, CHRISTCHURCH 1 N.Z. Ph. 62894

Prices:			
	MAGAZINE ONLY	NZ$ 43-00	NZ$ 4-00
	CASSETTE PLUS MAGAZINE	NZ$ 89-00	NZ$ 5-00
	DISK PLUS MAGAZINE	NZ$ 175-00	NZ$ 15-00

MICRO-80 is despatched from Australia by airmail to other countries at the following rates:

(12 MONTH SUB.)	MAGAZINE	CASS + MAG	DISK + MAG
PAPUA NEW GUINEA	Aus$ 40-00	Aus$ 83-00	Aus$ 143-00
HONG KONG/SINGAPORE	Aus$ 44-00	Aus$ 88-00	Aus$ 148-00
INDIA/JAPAN	Aus$ 49-00	Aus$ 95-00	Aus$ 155-00
USA/MIDDLE EAST/CANADA	Aus$ 55-00	Aus$ 102-00	Aus$ 162-00

Special bulk purchase rates are also available to computer shops etc. Please use the form in this issue to order your copy or subscription.

The purpose of MICRO-80 is to publish software and other information to help you get the most from your TRS-80, System 80 or Video Genie and their peripherals. MICRO-80 is in no way connected with either the Tandy or Dick Smith organisations.

** WE WILL PAY YOU TO PUBLISH YOUR PROGRAMS **
Most of the information we publish is provided by our readers, to whom we pay royalties. An application form containing full details of how you can use your TRS-80 or System 80 to earn some extra income is included in every issue.

** CONTENT **
Each month we publish at least one applications program in Level I BASIC, one in Level II BASIC and one in DISK BASIC (or disk compatible Level II). We also publish Utility programs in Level II BASIC and Machine Language. At least every second issue has an article on hardware modifications or a constructional article for a useful peripheral. In addition, we run articles on programming techniques both in Assembly Language and BASIC and we print letters to the Editor and new product reviews.

** LIABILITY **
The programs and other articles in MICRO-80 are published in good faith and we do our utmost to ensure that they function as described. However, no liability can be accepted for the failure of any program or other article to function satisfactorily or for any consequential damages arising from their use for any purpose whatsoever.

***** CONTENTS *****

MICRO-80 is registered by Australia Post - Publication SQB 2207 Category B

AUSTRALIAN OFFICE AND EDITOR: MICRO-80, P.O. BOX 213, GOODWOOD, SOUTH AUSTRALIA, 5034. TEL. (08) 211 7244

U.K. SUBSCRIPTION DEPT: 24 WOODHILL PARK, PEMBURY, TUNBRIDGE WELLS, KENT TN2 4NW

Printed by: Shovel & Bull Printers, 379 South Road, MILE END SA 5031

Published in Australia by: MICRO-80, 433 Morphett Street, Adealide

***** EDITORIAL *****

How vividly I remember the thrill all those years ago (about 3½ to be precise) when I eagerly unpacked the first disk drive for my Model I TRS-80, connected it up and marvelled at the speed, efficiency and, above all, the reliability of its data storage, compared with the cassette recorder I had been using. Time has passed, the old 35 track single-side, single-density disk drive has given way to the larger capacity 40 track drives, then dual-head and even 80 track, dual-head drives and finally double-density operation, all supported by more and more powerful disk operating systems. But none of these developments gave the same thrill as that first humble, 35 track disk drive. They were evolutionary, it was revolutionary. I thought that innocence had well and truly been lost and that that sort of excitement was gone forever.

Recently, I noticed an advertisement in Australian Personal Computer, for micro-winchester drives using ACT-506 controller boards. ACT was not an unfamiliar name, I had seen similar advertisements in the American Byte magazine several months earlier but that ACT stood for American Computers and Telecommunications, whilst this ACT stood for Australian Computers and Telecommunications. Enquiries to the States at the time had indicated that a security firm in Sydney had the Australian distribution rights for the ACT controller - seemingly a strange combination. Anyway, once the advertisement appeared in APC, I called the number indicated and almost before we knew it, Eddy Paay and I were on a plane, hotfoot (literally - it was during the power restrictions in February) for Sydney. I am happy to report that all innocence had not been lost. Watching a 5M byte Winchester drive quietly and efficiently store and retrieve data in the blink of an eye was as great a thrill as that first disk drive had been 3½ years ago. The most impressive aspect of the whole demonstration, however, was the the skilful combination of hardware and software which makes the ACT hard disk sub-system live up to its name as a system. This is definitely not just another piece of advanced hardward technology which leaves the new owner with the responsibility for finding how to use it in his system. This is a complete, operating, integrated hardware/software package that will plug in and run, efficiently and effectively. Before we continue with more details of the sub-system, a few comments about ACT itself may be of interest.

ACT was indeed an American company which developed computer systems for use in large security installations. Burglar alarms, as they used to be called, become very complex when large premises are being protected and computers have been used to do the job for some years. ACT used the Zenith Z89 microcomputer for this purpose. When the new 5¼" Winchester Technology drives first became available in the U.S.A. about 12 months ago, ACT decided that these would be ideal for their systems but no suitable disk controller board was available. Ken Utley of ACT rose to the challenge and developed a controller board and software to interface to the Zenith Z89 operating under CP/M, and thus was born the ACT-506 controller. Australian Protection Industries Pty. Ltd. was the Australian distributor for the security systems developed by ACT, hence the tie-up. Recently, Ken Utley moved to Australia bringing with him the design rights for the ACT-506 and Australian Computers and Telecommunications now manufactures and exports ACT-506 controller boards all over the world, including the U.S.A. In the intervening period, Ken has developed the software and hardware to interface the ACT-506 to a number of different microcomputers including the North Start Horizon, the Superbrain, S-100 systems and, most importantly from our point of view, the TRS-80 Models II and III.

If you closely examine the the photograph on the front cover of the hard disk sub-system, you will see that it bears the Micro-80 insignia. This is because we are so impressed and excited by its performance that we have had ACT construct the unit for us. We have also negotiated a special price of $2,995.00 (tax included) for a complete sub-system for the TRS-80 Model III, together with the CP/M (org 4200H) disk operating system. For almost the first time in Australian microcomputing history, you can purchase a significant item of hardware for the same price in Australia as can Americans. If you are considering using a TRS-80 in a business application, I believe you should very seriously consider including the Micro-80 hard disk sub-system as part of your system. The minimum specification computer you require for its proper use is a 48K single disk drive TRS-80 Model III or a 64K Model II. You could therefore have a 5 Mbyte hard disk, CP/M, Model III, single floppy disk drive system for only $5895.00 (Micro-80 prices). That is little more than half what you would pay for any other hard disk computer system. An added advantage is that absolutely no modifications are needed to the Model III. The Micro-80 sub-system simply plugs onto the 50 way expansion connector on the bottom of the Model III. Tandy's warranty is unaffected. Using CP/M you will have access to a wide range of professionally written and fully supported applications programs, including accounting packages, word processing, finance and budgeting, high level languages etc. etc. To add even further support, Micro-80 has commissioned Micro Systems Software in the U.S.A. to develop a version of DOSPLUS 4.0 especially for this system. With DOSPLUS, the sub-system will have access to the full range of Model III programs. Perhaps the best news of all is that we are offering a 30 day money back trial period. So confident are we of the system that we will refund the full purchase prices, less the cost of freight in either direction, if a sub-system is returned to us in original condition, within 30 days of purchase. We are proud to be involved in bringing this new development to TRS-80 users in Australia. Now, read on to find out more about the Micro-80 hard disk sub-system.

The sub-system consists of the following components:

- 5 Mbyte 5¼" seagate ST-506 (or similar) micro-winchester hard disk drive
- ACT-506 microsequencer controller board
- HOP high-integrity data separator
- interface card for host computer
- heavy duty power supply
- free-standing, moulded plastic cabinet
- CP/M disk operating system and utilities for host computer.

The controller board itself contains in excess of 60 integrated circuits and is capable of controlling two 5 Mbyte micro-winchester hard disk drives simultaneously. The micro-winchester can read data at a burst rate of 5 million bits per second. The controller accepts data at this rate and stores it until the host computer has transferred it, via the interface card. The form of construction used, allows the sub-system to be transferred to a different computer at minimal cost, since only the host computer interface card and the CP/M need be changed.

The CP/M disk operating system supplied has a number of enhancements and is specially tailored to suit the Model III. For example, the following video capabilities are supported:

HEX	EFFECT
08	Backspace and erase
09	Tab (8, 16, 24,...)
0A	Move cursor to start of next line and erase line.
0D	Move cursor to start of next line and erase line.
0E	cursor on
0F	cursor off
15	Swap space compression/Special Characters
16	Swap special/Alternate Characters
17	double size characters
18	Back space without earsing
19	Advance cursor
1A	cursor down line
1B	Cursor up line
1C	Home cursor
1D	Erase line and start over
1E	Erase to end of line
1F	Erase to end of screen

The CP/M 2.2 supplied contains the following standard CP/M programs and files:

MOVCPM.COM	Program to relocate the CP/M system.
PIP.COM	Program to move files. (Modified to use " " characters instead of "[]".
SUBMIT.COM	Program to perform "batch" processing.
ED. COM	Text file editor.
ASM.COM	8080 assembler.
DDT.COM	System debugger. (Modified to use RST6).
LOAD.COM	Generates COM files from HEX files.
STAT. COM	Program to display/set system and file parameters.
DUMP.COM	File dumper.
DUMP.ASM	Source of above.

There are several additional programs supplied on the diskette. These are special programs written by ACT personnel to aid in system generation and error analysis. All are supplied in both source and object form.

SYSGEN.ASM SYSGEN.COM	Special system generation program for use with TRS-80 Model III floppy drives and 5MB hard disk. Program operation is identical to the standard CP/M sysgen generation program described in the CP/M manual.
FORMAT.ASM FORMAT.COM	Diskette initialization program for floppy drives. to initialize the hard disk, run DISKTEST Test 9.
BOOTHD.ASM BOOTHD.COM	This is the bootstrap program used to read in and execute the boot sector from the hard disk. The same program may be loaded into a user-supplied EPROM to be used for automatic hard disk boot on CPU power up.
BIOS.ASM	This file (supplied in source form only) is the source for the BIOS supplied with the TRS-80 Model III System.

HARDBOOT.ASM HARDBOOT.COM	This is the program loaded from the boot sector of the hard disk which will read the CP/M operating system and the BIOS into memory and execute.
DISKTEST.ASM DISKTEST.COM	Program to perform comprehensive testing of the 5MB disk sub-system.
DEFDISK.ASM DEFDISK.COM	Program to set the hard disk drive characteristics.
FIXDISK.COM	Program to locate, and make invisible, bad areas on the disk. This program works on any disk run under CP/M 2.2.
TRSCPM	Program allows copying from TRSDOS diskettes to CPM disks.
INSERTLF	Program allow automatic insertion of line feeds after carriage returns in basic programs.
TIMEDATE	Program allows setting of time and date for real clock usage.
CLOCKON	Turn-on display of time and date on CRT screen.
CLOCKOFF	Turn-off display of time and date on CRT screen.
SAVEFILE	Program allows copying of files larger than the floppy disk drive capabilities.
RESTORE	Program retrieves large files from format written by SAVEFILE.

What of the TRS-80 Model I and System 80? It is a relatively straightforward matter to implement the hard disk sub-system on these computers, if the demand is there. If you are seriously interested in having 5 Mbyte of fast, reliable storage on your Model I or System 80, write in and let us know so we can decide whether we should implement it or not.

- 0000000000 -

***** INPUT/OUTPUT *****

From: R.J. Maclean, Formartin, Qld.

I have recently typed in the "Sound Effects Revisited" program which was published in the September edition. During the typing in I came across a bug which may also be a potential problem in other programs. When I ran the program my computer (System 80) threw up a '?SN ERROR IN 580'. After examining the relevant line I came to the conclusion that the lack of a space was causing the problem. Accordingly, I changed the line from "580 IFM>LETHENM=1" to "580 IFM>LE THENM=1". The changed line ran properly. Perhaps the removal of all blanks outside of REMark and PRINT lines is not always the best course.

As I am renewing my subscription with this letter I will take the opportunity to state that I have found Micro-80 to be very helpful and educational.

(The particular bug you discovered was probably due to the variable LE and the THEN statement forming the reserved word LET. Although LET is almost never used, it is available in BASIC. However, not all '80 interpreters run into this problem so there are obviously variations between machines. Perhaps it would be wise for authors to avoid variable names which form the first two letters of a reserved word. - Ed.)

- 0000000000 -

***** READERS' REQUESTS *****

This column is a regular feature of MICRO-80. In it, we list all those articles, programs, etc. requested by our readers. We invite contributions from readers to satisfy these requests and will, of course, pay a publication fee for all articles, programs etc. printed. As a guide, we will pay a minimum publication fee of $10 for any article or review published. In the case of software reviews, we will aim to pay in accordance with the value of the program, up to a maximum of $25. So, if you write a good review which we publish and the usual selling price of the program in Australia is $19.95, then we would pay you $20. In that way, the successful reviewer will get the program he reviews, free. (Make sure you include the selling price in your review). Unfortunately, we cannot afford that policy on hardware (!) so we will pay in accordance with the merits of the review - generally of the order of $25. Submission of a review for publication automatically means that you are prepared to accept the figure we decide to pay you and no correspondence will be entered into. Payment will be made within 30 days of publication.

** ARTICLES **

- File handling on the '80
- Description of the functions performed by the Expansion Interface
- Reviews of '80 compatible printers
- Reviews of commercially available software (including that produced by us!)
- Reviews of commercially available hardware
* A master index to the appropriate sections in the Tandy Manuals in Level I, Level II, DOS etc.
- Comparative reviews of disk drives
- How to convert a Level I program to Level II
* A simple guide to using Level I Arrays
- An explanation of how to make full use of USR, PEEK and POKE statements
- Discussion of the various electric fields produced by the keyboard, tape recorder, monitor disk drives etc., how to measure them, how important they are and how to combat them

** SOFTWARE **

* A m.l. program to enable the break key to work like RESET when using an expansion interface
- Stock market program
* Horse racing system
* Morse code decoder
- Sub-routine Forum
- A new STAR-TREK game
* Modification to SCRIPSIT which will enable it to output to the SYSTEM 80 printer port
- Programs of pharmaceutical interest such as Pharmacokinetics, Patient Medication Records, Drug information services etc.
- Multiple file tape backup (as in Australian Personal Computing Aug/Sept '81 for PET)
- A "PRINT SCREEN" command for the Model I
- A football game using graphics and sound
- Graphics car race as seen from the driver's position
- Tank game for a single player

** HARDWARE **

- Interfacing the '80 to external hardware
- Review on the performance of line filters
- Real Time clock
* Radio Teletype/Morse interfacing
- RFI (Radio Frequency Interference) suppression
* Interface for a Teletype printer
- Fast tape storage
- A cheap add-on keypad
- High speed card readers, interfacing and documentation for the Model II.

NOTE: An * denotes that we already have some suitable material on hand for this topic.

- 0000000000 -

***** MICROBUGS *****

Vol. 3 Issue 2 Page 11 - The Theory and Techniques of Sorting - Part 3. There were some errors in the PASCAL Code used to illustrate "A RECURSIVE APPROACH". The corrected code for the procedure INSERTSORT is as follows:

```
(*  MAIN PROGRAM BLOCK DECLARATIONS  *)
TYPE  LIST=ARRAY(1..N) OF REAL;
VAR   TABLE:LIST; L,U,POSITION:INTEGER;  VALUE:REAL;

PROCEDURE INSERTSORT (VAR TABLE:LIST; FIRST,LAST:INTEGER);
VAR I:INTEGER;  TEMP:REAL;
  PROCEDURE BINARYSEARCH (VAR TABLE:LIST; LOW,UPP:INTEGER;
                          VALUE:REAL; VAR POS:INTEGER);
  VAR MID:INTEGER;
    BEGIN
      IF LOW > UPP THEN POS:=LOW
      ELSE
         BEGIN
           MID:=(LOW+UPP) DIV 2;
           IF VALUE < TABLE(MID) THEN
               BINARYSEARCH (TABLE,LOW,MID-1,VALUE,POS)
           ELSE
               IF VALUE > TABLE(MID) THEN
                   BINARYSEARCH (TABLE,MID+1,UPP,VALUE,POS)
               ELSE
                   POS:=MID
         END
    END;
```

```
PROCEDURE BLOCKMOVE (VAR TABLE:LIST; L,U:INTEGER);
VAR I:INTEGER;
  BEGIN
    FOR I:=U DOWNTO L DO TABLE(I+1):=TABLE(I)
  END;

BEGIN   (*  OF MAIN  *)
  FOR I:=2 TO N DO
    BEGIN
      IF TABLE(I) < TABLE(I-1) THEN
        BEGIN
          TEMP:=TABLE(I);
          BINARYSEARCH (TABLE,1,I-1,TEMP,POSITION);
          BLOCKMOVE (TABLE,POSITION,I-1);
          TABLE(POSITION):=TEMP
        END
    END
END;
```

The corrections were made in the following:

In Procedure BINARYSEARCH,

IF LOW > UPP THEN POS:=LOW instead of -1, because although the procedure was demonstrated as a search routine for a specific value, it needed a modification to allow it to be used in the insertion sort for determining the correct position (in sequence) of a value that may not be currently in the table.

In the Main Procedure block,

The second parameter used in the BLOCKMOVE call (U+1) should read POSITION, because this is the parameter as returned from the CALL to the BINARYSEARCH procedure. Same correction to the next line where TEMP is moved into that POSITION in the TABLE.

- 0000000000 -

***** BETTER BASIC PROGRAMMING - PART 7 by Rod Stevenson *****

** YET MORE ASSEMBLY **

WHY SO MUCH ASSEMBLY?

The emphasis on assembly in these last three articles is not merely that I find it gives far greater satisfaction than BASIC, but that so much can be done in assembly, whereas BASIC is limited to that which is provided by the writers of the BASIC interpreter in ROM. Not to put down BASIC at all - I still use it for most of the "real" programming I do.

From the previous two articles and with what follows here, I hope you will share my enthusiasm for assembly, not only for the speed and because it's the only way to modify the operating system, but for the self-satisfaction in having got a job done efficiently.

LIST DISABLE.

```
                00100 ;
                00110 ;TO DISABLE SHIFT @, CSAVE, BREAK, LIST, SAVE, LLIST
                00120 ;DISABLED COMMANDS MAY BE ALTERED IN TABLE FROM LINE 870
                00130 ;
                00140 ;       BY E. R. PAAY
                00150 ;       FOR R. STEVENSON
                00160 ;       24-8-1980
                00170 ;
                00180 ;NOTE : SHIFT A = SHIFT @, SHIFT B = BREAK
                00190 ;       AFTER THIS PROGRAM IS LOADED.
                00200 ;
4016            00210         ORG     4016H
4016 F142       00220         DEFW    START           ;SET NEW KB. VECTR.
42F1            00230         ORG     42F1H
                00240 ;
                00250 ;       DEFINE LABELS.
                00260 ;
40A7            00270 BPOINT  EQU     40A7H
06CC            00280 BASIC   EQU     06CCH
0080            00290 DELIM   EQU     80H
```

```
00FF            00300 ENDMRK  EQU     0FFH
03E3            00310 KBSCAN  EQU     3E3H
                00320 ;
                00330 ;       START OF PROGRAM.
                00340 ;
42F1 CDE303     00350 START   CALL    KBSCAN          ;SCAN KEYBOARD
42F4 FE65       00360         CP      65H             ;SHIFT E ?
42F6 2841       00370         JR      Z,ENTER
42F8 FE62       00380         CP      62H             ;SHIFT B ?
42FA 2811       00390         JR      Z,BREAK
42FC FE61       00400         CP      61H             ;SHIFT A ?
42FE 2810       00410         JR      Z,SHFTAT
4300 FE0D       00420         CP      0DH             ;ENTER ?
4302 280F       00430         JR      Z,CHKBUF
4304 FE01       00440         CP      01H             ;BREAK ?
4306 2803       00450         JR      Z,IGNORE
4308 FE60       00460         CP      60H             ;SHIFT @ ?
430A C0         00470         RET     NZ
430B AF         00480 IGNORE  XOR     A
430C C9         00490         RET                     ;IGNORE KEY ENTRY.
430D 3E01       00500 BREAK   LD      A,1             ;BREAK !
430F C9         00510         RET
4310 3E60       00520 SHFTAT  LD      A,60H           ;SHIFT @ !
4312 C9         00530         RET
4313 D9         00540 CHKBUF  EXX                     ;SAVE REGISTERS
4314 ED5BA440   00550         LD      DE,(40A4H)
4318 AF         00560         XOR     A
4319 67         00570         LD      H,A
431A 2E53       00580         LD      L,TABLE-START+8
431C 19         00590         ADD     HL,DE
431D EB         00600         EX      DE,HL
431E 2AA740     00610 LOOP1   LD      HL,(BPOINT)     ;GET BUFFER ADDRESS
4321 2B         00620         DEC     HL
4322 D7         00630         RST     10H             ;FIND FIRST CHARACTER
4323 1A         00640 LOOP2   LD      A,(DE)          ;GET CHARACTER
4324 13         00650         INC     DE              ;BUMP POINTER
4325 B7         00660         OR      A
4326 FACC06     00670         JP      M,BASIC         ;IF MINUS REINIT. BASIC
4329 BE         00680         CP      (HL)            ;COMPARE CHARACTERS
432A 23         00690         INC     HL              ;BUMP POINTER
432B 28F6       00700         JR      Z,LOOP2         ;GET NEXT CHARACTER
432D 1A         00710 LOOP3   LD      A,(DE)
432E 13         00720         INC     DE
432F FE80       00730         CP      80H             ;FIND NEXT ENTRY
4331 20FA       00740         JR      NZ,LOOP3        ;     IN TABLE
4333 1A         00750         LD      A,(DE)
4334 FEFF       00760         CP      ENDMRK          ;END OF TABLE MARKER ?
4336 20E6       00770         JR      NZ,LOOP1
4338 D9         00780         EXX                     ;RESTORE REGISTERS
4339 3E0D       00790 ENTER   LD      A,0DH           ;PUT ENTER CODE BACK IN A
433B C9         00800         RET                     ;DONE
                00810 ;
                00820 ;THE TABLE THAT FOLLOWS MAY HAVE AS MANY ENTRIES AS
                00830 ;REQUIRED AS LONG AS ENTRIES ARE SEPARATED WITH A ZERO
                00840 ;BYTE.  THE MAIN PROGRAM WILL MAKE BASIC IGNORE ANY
                00850 ;COMMAND CONTAINED IN THIS TABLE.
                00860 ;
433C 4C         00870 TABLE   DEFM    'LIST'
4340 80         00880         DEFB    DELIM
4341 4C         00890         DEFM    'LLIST'
4346 80         00900         DEFB    DELIM
4347 43         00910         DEFM    'CSAVE'
434C 80         00920         DEFB    DELIM
434D 53         00930         DEFM    'SAVE'
4351 80         00940         DEFB    DELIM
4352 FF         00950         DEFB    ENDMRK          ;END OF TABLE MARKER.
06CC            00960         END     BASIC
```

This is the source code for the BASIC routine presented last time to disable BREAK, LIST, and others. As you see from the comments, any other commands may be added to the table as required.

The whole basis of the program is that it intercepts the keyboard scanning routine normally provided by the operating system. This is accomplished by an address being given to 4Ø16H, so that the operating system will go through this new routine each time it scans the keyboard instead of doing its own normal routine.

Having jumped into this new routine, a ROM routine is used to get the value of any key pressed, then to check it as noted in the comments. If it's none of the new operators provided, it just returns to the usual system. But if it is, it goes through the program and checks the keyboard buffer contents against the table provided.

Two items contained in this program you'd be unlikely to know without Eddy Paay's ROM Reference Manual - that 4Ø16H is the address to be used to set a new keyboard vector, and a CALL to 3E3H will provide the results of a keyboard scan in the A register.

BREAK-handling.

As mentioned in the first article, there are some simple POKE methods for disabling the BREAK-key - as well as the routine above. However, they all require knowing that the BREAK-key will force a jump to 4ØØCH (16396 decimal). If you have a disk system, TRSDOS will have put an address there to make the BREAK-key the method of jumping to DEBUG. From which it follows that you can also put something there to provide your own handling of the BREAK-key. Other systems have a statement of ON BREAK GO TO, but we can simulate this quite easily.

Normally, level II will have a 1Ø1 (RET) at 16396. And I use decimal here because you will probably be doing these modifications from BASIC, when you require a decimal address. Which means it will just return with whatever is in the A register (the ROM Reference Manual tells you A holds the result of the keyboard scan). So this leaves the way open for us to operate on the A register before it gets back! You will remember that I suggested in the first episode that you could simply POKE 16396,23 to disable the BREAK-key in level II. This is, in effect, rotating the accumulator left (23 decimal = 17H = RLA) which means it no longer holds 1 (the value of the BREAK-key), and it will RETurn on encountering the 2Ø1 (= RET) that level II has at 164ØØ. But you will also remember I suggested that this is a bit simple - and it won't work for a disk system. If you zero the accumulator (by POKEing a value to do so, 151 = 97H = SUB A, or 175 = AFH = XOR A, into 16396, then a 2Ø1 (= C9 = RET) into 16397, can you see that this is the better alternative?

Yes, I know NEWDOS has the BREAK-on/off facility, and you may say "why bother to POKE?" The reason is that (I hope) you've learned something.

```
10 ON ERROR GOTO 32700   'TELLS WHERE TO GO WHEN AN ERROR IS DETECTED
20 POKE 16396,195: POKE16397,49: POKE 16398,40    'TELLS BREAK-KEY WHERE TO GO
26 R=45/0   'DEMONSTRATES A DDIVIDE BY ZERO ERROR
27 RERE   'DEMONSTRATES A SYNTAX ERROR
30 PRINT" PRESS BREAK-KEY TO STOP THIS     ";:GOTO30   'THIS IS ONLY A DEMONSTRAT
ION LINE
32700 IF ERR/2+1=12 THEN PRINT:PRINT:PRINT "BREAK-KEY WAS PRESSED: PRESS BREAK-K
EY AGAIN TO STOP PROGRAM NOW                          OR PRESS ENTER TO CONTINUE";:
INPUTA:RESUME   'HANDLES BREAK-KEY
32710 PRINT:PRINT"ERROR"ERR/2+1;"HAS OCCURED IN LINE"ERL;": PRESS ENTER TO CONTI
NUE ";:INPUTA:RESUME NEXT   'HANDLES ANY ERROR
```

This is another method of handling BREAK. And this, in effect, provides a function of "ON BREAK GO TO". It's not as neat as could be done by assembly, but does illustrate another approach.

The whole idea is that the BREAK-key will simulate an ID error, which will not normally occur in the RUN mode. So when one does, it can only be because BREAK has been pressed.

The BREAK-handling address (16396-9) has POKEd into it the address of the ID error entry point - yet again found from the ROM Reference Manual. So if we do nothing more, BASIC will stop and provide a message "ID ERROR IN LINE X". But to prevent this, we make use of the ON ERROR GO TO function. Incidentally, I don't favour general use of this function, as I believe your program should be fully debugged before you let it go! (I cannot agree with Rod in this. There is nothing more annoying in an applications program if, through gross operator error, the program crashes, for the user to see an error message generated by the BASIC interpreter such as OV error or OM error. Far better for the program to resume control and instruct the operator what to do next. - Ed.)

To explain the program.

Line 1Ø tells the ON ERROR routine where to GO TO.
Line 2Ø puts the entry point of ID ERROR routine into the BREAK-handling routine in the operating system.
Lines 26, 27, 3271Ø are just added in this example to show that the routine will handle otherwise-occurring errors too - though as I wouldn't expect your program to have any, these lines would of course be unnecessary!
Lines 3Ø and 327ØØ are the real "bones" of the program, and allow for continued pressing of BREAK as a demonstration.

Of course, this is just an example routine. Your real program would have only lines 1Ø, 2Ø, 327ØØ as well as the whole of its real working code. And these line numbers are such for this very reason - lines 1Ø and 2Ø are necessary right at the start of the program to tell it where to go to. Line 327ØØ would be replaced by a subroutine at the end of your program to do what you actually want to do when the BREAK-key is pressed. An obvious thing is to close disk files before exiting from the program. Or perhaps you'd want to allow only one pressing of BREAK in your program by disabling it in line 327ØØ after the first press, then return to the program. Or perhaps you will have given an instruction not to press the BREAK-key, and take some action if the operator does press the BREAK-key after your instruction not to. As usual with computing, the possibilities are limited only by the imagination.

In fact, all the routines presented in this series are intended only to provoke thought - and to use in your own programs. As I keep saying, the way to learn and understand is to use these ideas for yourself.

SOUND.

The whole basis of the simple sound routine found in many programs is that to record onto tape, the '80s PORT 255 is used to send positive or negative voltages to the cassette.

System 80s can use the second cassette by enabling bit 2 of PORT 254 which acts as a switch to change from the on-board cassette, so sound will still come down the AUX line to the second cassette. All that is necessary is to put an instruction at the start of the program to OUT 254,4 and this will be effective throughout the whole program until you go back to BASIC and get the READY message again. This also works in machine language programs, so if you modify the many sound programs by adding machine language code for OUT 254,4 at the entry point (move the entry point if necessary), the sound will stay with you. No, it's not hard. I added the necessary 4 bytes to the "BIG FIVE" game "Attack" and can tell you that's all that is necessary. However, a better way is a hardware modification to get the sound directly from the board, which will give a far better tone than even the TRS-80's AUX line does without any modifications.

```
7F00            00100           ORG     32512
7F00 CD7F0A     00110 ENTRY     CALL    0A7FH   ;GET VALUE FROM USR(0) IN BASIC
7F03 0E8C       00120           LD      C,140   ;LENGTH OF NOTE (= TIMES CYCLE REPEA
TED
7F05 45         00130 REPEAT    LD      B,L     ;VALUE FROM USR(0), CAN BE 0-255
7F06 3E01       00140           LD      A,1H    ;VALUE FOR POSITIVE VOLTAGE
7F08 D3FF       00150           OUT     (0FFH),A;SEND PULSE OUT PORT
7F0A 10FE       00160 HIVOLT    DJNZ    HIVOLT  ;DELAY FOR NUMBER CYCLES IN USR(0),
= PITCH
7F0C 45         00170           LD      B,L     ;VALUE FROM USR(0)
7F0D 3E02       00180           LD      A,2H    ;VALUE FOR NEGATIVE VOLTAGE
7F0F D3FF       00190           OUT     (0FFH),A;SEND PULSE OUT PORT 255
7F11 10FE       00200 LOVOLT    DJNZ    LOVOLT  ;DELAY
7F13 0D         00210           DEC     C       ;AFTER ONE WHOLE CYCLE COMPLETED
7F14 20EF       00220           JR      NZ,REPEAT;REPEAT 140 TIMES (VALUE IN C)
7F16 C9         00230           RET             ;BACK TO BASIC
7F00            00240           END     ENTRY
```

This is the simple sound generator program I presented at the end of the LIST disable routine in BASIC last article. The comments make the source code fairly self-explanatory. Because of the speed of machine code it has been necessary to incorporate delays into the program. Even so, the speed required is too high for a program such as this to be written successfully in BASIC.

```
1 CLEAR150:FORI=1TO72:READD:S$=S$+CHR$(D):NEXT:SL=PEEK(VARPTR(S$)+1):SM=PEEK(VAR
PTR(S$)+2):POKE16526,SL:POKE16527,SM:DATA205,127,10,203,124,40,4,34,28,65,201,34
,30,65:DATA219,255,31,31,31,47,230,248,95,58,57,65,254,4,32,2
2 DATA171,95,58,32,65,87,237,75,28,65,43,124,181,40,6:DATA221,227,221,227,24,12,
42,30,65,122,7,7,87,230,3,179:DATA211,255,3,120,177,32,228,123,211,255,201
3 INPUT"WHAT TYPE OF SOUND (1 TO 10)";T
4 IFT<1ORT>10THENPRINT"MUST BE BETWEEN 1 & 10":GOTO3ELSEONTGOTO6,7,8,9,10,11,12,
13,14,15
5 GOTO4
6 T=1:GOTO16
7 T=5:GOTO16
8 T=6:GOTO16
9 T=17:GOTO16
10 T=18:GOTO16
11 T=22:GOTO16
12 T=25:GOTO16
13 T=86:GOTO16
14 T=90:GOTO16
15 T=102:GOTO16
```

```
16 POKE16672,T
17 INPUT"DURATION (1 TO 32767)";D:IFD<1ORD>32767THEN17ELSED=D-32768:X=USR(D)
18 INPUT"PITCH (1 TO 100)";P:X=USR(P):R$="":INPUT"DO YOU WANT TO REDEFINE TYPE";
R$:IFLEFT$(R$,1)="Y"THEN3ELSEINPUT"DO YOU WANT TO REDEFINE DURATION";R$:IFLEFT$(
R$,1)="Y"THEN17ELSE18
```

This is a more complex sound-generation program, but it still relies on the principle explained above (it must if it's to use the cassette port).

I won't attempt an explanation of this code - it's not exactly elementary. It has two arguments passed to the machine language routine which returns in the middle of itself, as well as one POKEd into fixed RAM. But a re-reading of Ron Sully's article - Sound Effects Revisited, in Issue 22 may make it clearer. What it does show though, is that our '80 is indeed capable of some wondrous things - but you already knew this from Ron's program. Going even further, and without the hardware provided by "Orchestra 80" there is a program called "Opera" which does as Ron suggests - uses the BASIC interpreter to produce such works as "The William Tell Overture". So with thought, experimentation, and perseverance, much is possible!

A READER RESPONDED!

Being shocked, horrified, and disgusted at the gross inefficiency of my screen white-out routine presented two issues ago, a reader (Eddy Paay) has provided this code to do the same.

```
3FFF            00100 VIDRAM  EQU     3FFFH           ;END OF SCREEN RAM
00BF            00110 WHITE   EQU     00BFH           ;CODE FOR SOLID GRAPHIX BLOC
K
                00120 ;
7F00            00130         ORG     32512
7F00 21FF3F     00140 START   LD      HL,VIDRAM       ;POINT TO END OF SCREEN
7F03 36BF       00150 LOOP    LD      (HL),WHITE      ;WHITE-OUT CURRENT SCREEN PO
SITION
7F05 2B         00160         DEC     HL              ;POINT TO NEXT (BACKWARDS) S
CREEN POSITION
7F06 7C         00170         LD      A,H             ;LOOK AT MSB SCREEN ADDRESS
7F07 E604       00180         AND     4               ;IS SCREEN POINTER <3CH?
7F09 20F8       00190         JR      NZ,LOOP         ;IF NOT, CONTINUE
7F0B C9         00200         RET                     ;FINISHED, SO RETURN TO BASI
C
7F00            00210         END     START
```

I think the comments make all clear - the only "fancy" bit is the ANDing of the MSB of the screen-pointer to determine if it's past the start of the screen (having commenced at the end). As the screen starts at 3CØØH, it is convenient to be able to test that as soon as the MSB does not have CH as its low-order nibble (nibble = 4 bits, so high and low order nibble for each hex digit in a byte) the screen address is no longer pointed to.

So having got a response from a reader, I anxiously await more. MICRO-80 has already installed a bigger letterbox to hold them all, so don't let it go to waste.

- 0000000000 -

***** THE THEORY AND TECHNIQUES OF SORTING - PART 4 by B. Simson *****

The last article demonstrated some variations to the general method of sorting "by insertion". By now, you would be under the impression that sorting by insertion is a very popular sorting technique, if only for the number of different variations possible to the main theme. Some fairly efficient variations have emerged, all with their own advantages. However, all of these variations so far have one thing in common. They are all subject to the restrictions inherent in a linear, sequential data structure, the "array". You can only stretch an elastic so far, then it snaps...

A NEW DATA STRUCTURE.

What we need is something of a different nature that allows for increased efficiency, without overbearing complexity. Before introducing this structure of different nature, examine one of the major disadvantages in all of the insertion sorts so far - that of item (element, or key) movement. Straight and multiple insertion required movement of individual items as they were ranked. Binary insertion required movement of items as a logical block when the insertion position was found. This is a necessary evil when dealing with insertions/deletions in a sequential data structual such as the straight "array". To avoid this type of movement, and still retain necessary ordering of items, a non-sequential data structure could be used - that of "linked" organization, commonly found in linked linear lists.

LINKED LIST ORGANIZATION AND PROCESSING.

In order to understand its use in the insertion sort (the "list insertion" sort), a brief explanation of linked linear lists is offered. Elements of any kind that are logically adjacent (as items in a sorting array) need not be physically adjacent (as the physically adjacent cells in an array). These elements, known as "nodes", are logically linked by pointers. The first logical node in a linked list is defined by a special variable, known as the "head" of the linked list, which is itself a special node. The last element is denoted as such by the use of a special null value in its pointer field. Figure 1 illustrates the logical representation of such a data structure.

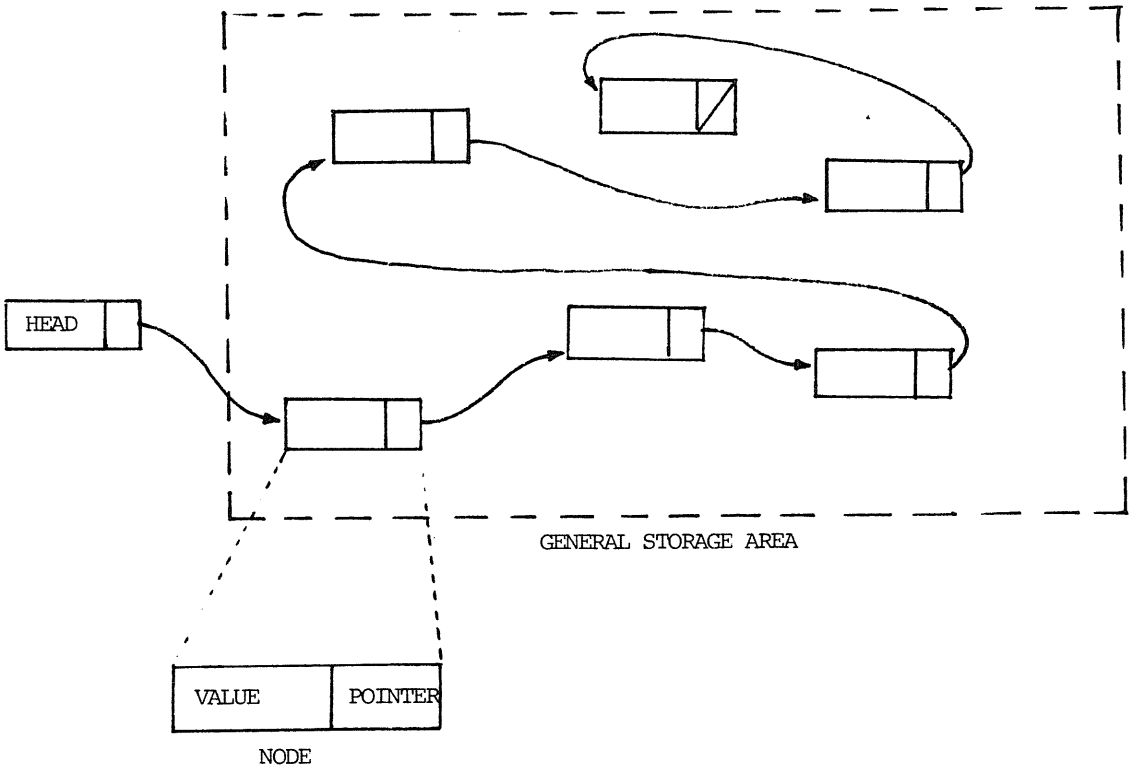
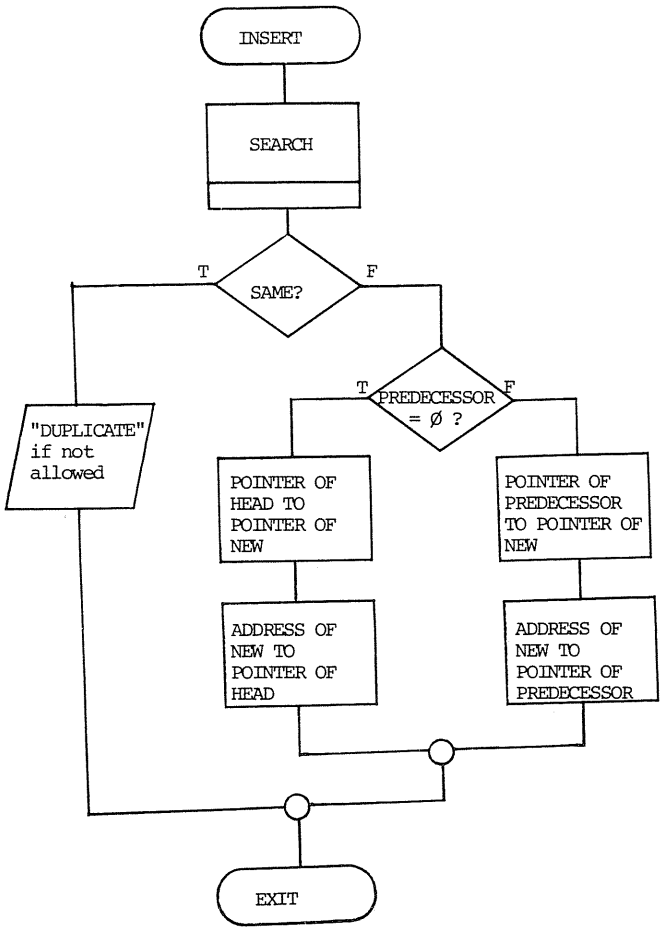


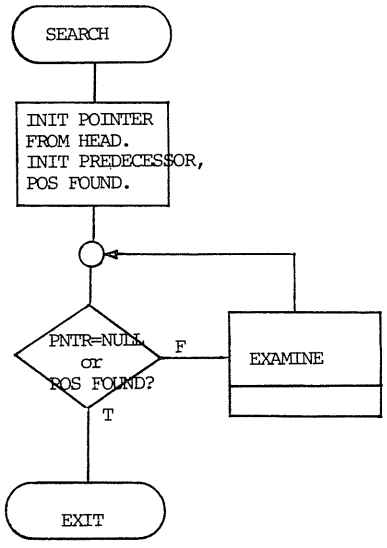
FIGURE 1 - General structure of a singly linked list.

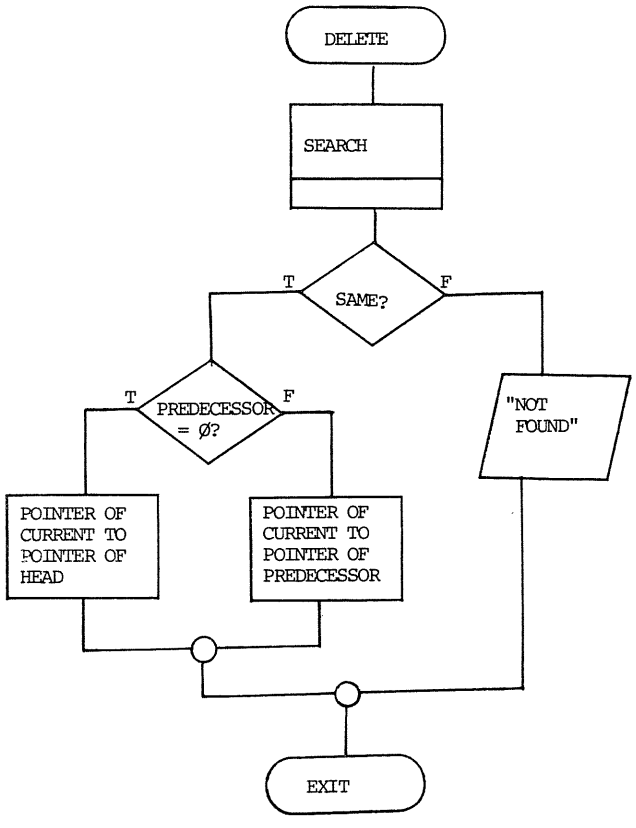
However, unlike standard Pascal, and some other programming languages, BASIC does not support such dynamic linked structures, so arrays are used to simulate them. You may feel that to do so would be no better than the use of arrays in the straight insertion and binary insertion sort. However, this is not so, as is demonstrated below. To change the relationship of one item to another, at least one item, and usually many items, needed physical movement in a sequential data structure. To change the relationship of one item to another in a linked data structure, a maximum of only two pointers needs to be moved and no actual item (data) movements need occur. For example, in a sequential structure, insertion of an item at the third position of the structure requires the movement down of all items from the third position to the last. In a linked structure, insertion at the third position requires only the following:

- Pointer of second item moved to pointer of new;
- Address of new item moved to pointer of second item.

When inserting (or deleting) nodes from a linked structure, the address of the predecessor node to the candidate node for the operation is required. This is obtained by chaining through the structure, following the pointers. Now, this is much more efficient than the movement of masses of items in a sequential structure, especially if the number of items in a sort array is large. In fact, in a linked structure, after the predecessor node is located, the number of items in the sort array is irrelevant when changing the relationship between two items. The general operations of insertion and deletion are formally defined in the flowcharts in figure 2. An important assumption here is that the head of the list is located at row zero in the table. (Some programming languages do not allow element zero addressing). If the head is located in some other pointer variable, then the special cases of insertion of a node in an empty list (head contains null) or insertion in the first position in the list must be considered, as does deletion.







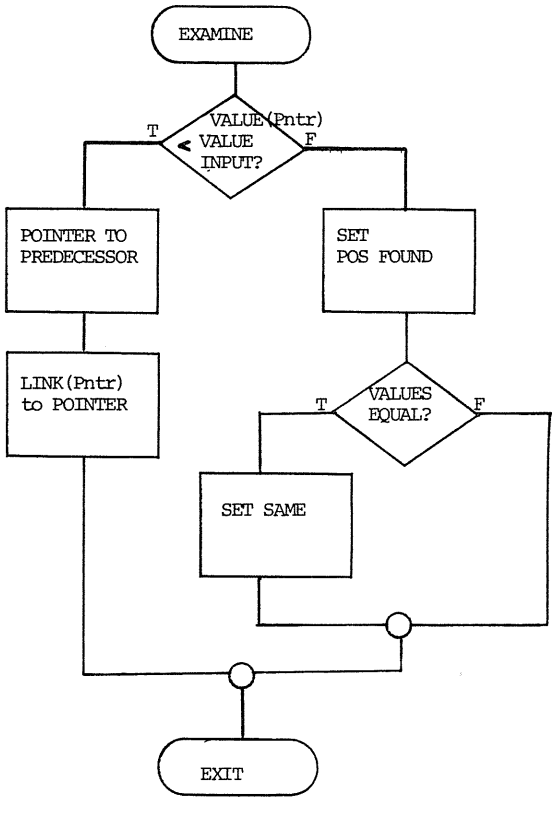


FIGURE 2 - General processing algorithms - singly linked linear lists.

Note: Pntr is an abbreviation for POINTER

APPLYING THE NEW DATA STRUCTURE.

In order to demonstrate the increased efficiency that this data structure can give to the insertion sort, some of the list processing algorithms in the flowchart need to be coded:

```
1195  GOTO2100       'Needed to route initial processing to driver.
1200  '              Search
1250  PNTR=A(0,2):PRED=0:FOUND=FALSE:SAME=FALSE
1300  IF PNTR=0 OR FOUND THEN RETURN
1350  IF A(PNTR,1)<A(AD,1) PRED=PNTR:PNTR=A(PNTR,2):ELSE FOUND=TRUE
1450  GOTO1300
1500  '              INSERT
1550  GOSUB1250
1600  A(AD,2)=A(PRED,2)
1650  A(PRED,2)=AD
1700  RETURN
```

The flag "SAME" in the flowchart is not required here, because insertions of duplicates in the list are allowed. Also, the list insertion sort does not do any deletion operations. Now that the essential list processing algorithms have been coded, the rest of the list insertion sort and driver follow:

```
1850  A(0,2)=0       'INITIALIZE HEAD.
1900  FOR AD=1TON
1950  GOSUB1550      'LINK ITEM TO STRUCTURE
2000  NEXT AD
2050  RETURN
2100  '              MAIN DRIVER
2200  'A()=LINKED LIST STRUCTURE, AS SIMULATED BY THE ARRAY
2250  'PRED=ADDRESS OF PREDECESSOR NODE
2300  'FOUND=FLAG SET TRUE WHEN INSERTION POSITION FOUND
2350  'PNTR=TEMPORARY POINTER VARIABLE
2355  'AD=ADDRESS OF CURRENT ITEM UNDER CONSIDERATION
2360  'A(AD,1)=VALUE OF NEW ITEM TO BE INSERTED
2370  '
2380  DEFINTA-Z
2400  CLS:INPUT"NUMBER OF ITEMS FOR SORTING";N:IFN<1THEN2400
2450  DIMA(N,2):TRUE=-1:FALSE=0:RANDOM
2500  PRINT"LIST BEFORE SORT:"
2550  FORI=1TON
2600  A(I,1)=RND(1000):A(I,2)=0:PRINTA(I,1);
2650  NEXT
2700  PRINT:INPUT"HIT ENTER TO START SORT";I
2750  GOSUB1850
2800  PRINT"LIST AFTER SORT:"
2850  PNTR=A(0,2)
2900  IF PNTR=0 THEN 3050
2950  PRINTA(PNTR,1);:PNTR=A(PNTR,2)
3000  GOTO2900
2050  END
```

SOME POINTS OF EXPLANATION.

The initial unsorted data is placed in a 2-dimensional table rather than a one-dimensional array, because this is the structure needed to simulate the linked list.

The head of the linked list is located in row zero, with the point value in column 2.

The value of the null pointer is zero, since an address of zero is not valid for an item being sorted.

The method of displaying the results of the sort is not simply a sequential list of column 1 of the table, since no actual shifting of items occurred (they are still in the same physical sequence as before the sort). Rather, the results are extracted logically by following the logical relationship of the items as established by the link information of the pointers.

In short, this method of insertion sorting obviates the necessity of moving a potentially large number of items down to vacate a position for insertion of a new item. Instead, insertion of a new item is achieved by modifying the link information inherent in the data structure.

YET ANOTHER VARIATION.

There is another variation of the insertion sort that has become popular, known as the Shell sort or diminishing-increment sort, developed by Donald Shell in 1959, and documented in the

A.C.M. Collected Algorithms, Algorithm 201. This algorithm is similar to the straight insertion sort, except that instead of comparing adjacent items, comparisons are made on items that are separated by an increment that diminishes with each pass. One version of this algorithm appeared in Tandy Newsletter Volume No. 9, May/June 1980, and is reproduced below, with modified driver and variable names.

A variation to this method of rearranging items in line 4180 (when an out of sequence situation has been detected) is known as Hibbard's modification. (A.C.M.) This states that X pair swaps (as above) can be substituted by one save, (X-1) moves and one insertion (of the saved item). This coincides exactly with the method of rearranging an item ranked in the straight insertion sort. It can be seen from this code that the diminishing increment (DI) in line 4120 starts at half of the list size, and diminishes by a factor of .5 for each successive pass. This has the effect of increasing the pass size in line 4140 from half of the list size to 1 less than the list size.

MATHEMATICAL ANALYSIS.

The diminishing increment sort is among many other sorting algorithms whose efficiency approaches N*(Log of N to Base 2). This sorting algorithm has been chosen to demonstrate that fact because the number of passes and comparisons in each pass can be easily seen. For instance, with a list size of 16, the number of passes = 4, because the DI (diminishing increment) starts at 8, and is modified to 4, 2 and 1. This means that the minimum number of comparisons in each of the passes are 8, 12, 14 and 15, as the pass size increases in the same proportion that the DI is decreasing. (These represent the minimum number of comparisons because when an out of sequence situation is detected, further comparisons and moves are made before continuing with the pass). In fact, the total number of passes involved (P) is related to the list size (N) through the relationship $2^P = N$. A list size of 32 requires 5 passes ($2^5=32$), i.e. pass sizes of 16, 16+8, 16+8+4, 16+8+4+2 and 16+8+4+2+1.

DIMINISHING-INCREMENT SORT

Let n = total number of items being sorted
p = number of passes
c = number of comparisons

Since $2^p = n$, $p = (\frac{\log n}{\log 2})$ i.e. INT (LOG N/LOG 2)

C in Pass 1 = $\frac{n}{2}$

C in Pass 2 = $(\frac{n}{2} + \frac{n}{4})$

C in Pass 3 = $(\frac{n}{2} + \frac{n}{4} + \frac{n}{8})$

C in Pass P = $(\frac{n}{2} + \frac{n}{4} + \frac{n}{8} + \ldots)$

$$= \sum_{i=1}^{p} \frac{n}{2^i}$$

This represents the summation of a geometric progression of general formula

$\frac{A(1-R^N)}{1-R}$ where A = first term
R = common ratio
N = n^{th} term

In this case, $A = \frac{n}{2}$

$R = \frac{1}{2}$

N = Pn (Pass n)

Substituting, C in Pass Pn = $\frac{\frac{n}{2}(1-(\frac{1}{2})^{Pn})}{\frac{1}{2}} = n(1-1/2^{pn})$

So, no. comparisons in any pass is given by formula

$n(1-1/2^{pn})$ where Pn = Pass number.

$C_{total} = n(1 - \frac{1}{2^1}) + n(1 - \frac{1}{2^2}) + \ldots + n(1 - \frac{1}{2^p})$ where P = total number of passes

$= n((1- \frac{1}{2^1}) + (1 - \frac{1}{2^2}) + \ldots.+(1 - \frac{1}{2^p}))$

$= n(1 + 1 + \ldots+1 - (\sum_{i=1}^{p} \frac{1}{2^i}))$

The inner parenthesis also represents a summation of a geometric progression, where A = $\frac{1}{2}$, R = $\frac{1}{2}$, n = P.

Also, since the units have been extracted from each pass parenthesis, their sum must equal P.

Substituting $C_{total} = n(P - \frac{\frac{1}{2}(1-(\frac{1}{2})^P)}{\frac{1}{2}})$

$= n(P - (1 - \frac{1}{2^P}))$

$= n(P - 1 + \frac{1}{2^P})$

Now as the number of passes (p) becomes large, -1 becomes insignificant, and 2^P becomes huge, causing $\frac{1}{2^P}$ to become insignificant.

Therefore C_{total} approx. = n.P

Since $p = (\frac{\log n}{\log 2})$

Ctotal approx. = $n \log_2 n$

That is, no. comparisons approaches the order of $n \log_2 n$.

SELECTION SORT.

This algorithm was discussed in the second article of this series. During the first pass, in which the smallest item was found, (n-1) items are compared. Therefore we can say that for the pth pass, (n-p) comparisons are required. Also, (n-1) passes are required.

The total number of comparisons (C) is therefore (n-1)+(n-2)+(n-3)+...+2+1

$= \sum_{p=1}^{n-1}(n-p)$ where p represents the pass number.

This represents the summation of an arithmetic progression, of general formula $\frac{n}{2}(2A+d(N-1))$

where A = first term, d = common difference, N = n^{th} term

In this example, A = (n-1)
d = -1
N = no. of passes, = (n-1)

Substituting,

$C_{total} = \frac{n-1}{2}(2(n-1)+ -1(n-1-1))$

$= \frac{n-1}{2}(2n-2 -n+2)$

$= \frac{1}{2}n(n-1)$

As n becomes large, -1 becomes insignificant. Therefore no. comparisons approx. $\frac{1}{2}n^2$, which is of the order of n^2.
This is much less efficient than one to the order of $n \log_2 n$.

TO SUMMARIZE.

Two further variations to straight insertion sorting were demonstrated; one using a data structure based on linked organization, using a linked linear list, the other involving comparison of items that are separated by a diminishing increment. It can be shown mathematically that the efficiency of this and like algorithms is greater than some of the more elementary algorithms discussed earlier, like the selection sort.

Well, this finalizes the discussion on sorting "by insertion", although this is by no means all there is to insertion sorting. There are many other hybrid algorithms that employ a mixture of techniques, endeavouring to obtain the best features of insertion, exchange, selection and merging techniques. Next month, a very efficient form of sorting by selection (more so than the Shell sort) will be discussed, due to the use of yet another kind of data structure.

- 000000000 -

***** '80 USERS' GROUPS *****

The following is a list of '80 Users' Groups. If you have a group that is not included here, please let us know about it so that we can publish details. Owners of System '80s are welcome at all the groups.

** AUSTRALIA **

AUSTRALIAN CAPITAL TERRITORY

CANBERRA GROUP

Contact: Bill Cushing, 10 Urambi Village, Kambah, ACT. 2902.
MEETINGS: 3rd Thursday of each month at 7.30 p.m. in: Urambi Village Community Centre, Crozier Circuit Kambah.

NEW SOUTH WALES

COMPUTERTOWN CAMDEN (CTAUS!)

Contact: Keith Stewart
P.O. Box 47, Camden, NSW. 2570.

SYDNEY MAPPER CP/M GROUP

Contact: Dan Lawrence, Mapper CP/M Users' Group, c/o G.P.O. Box 2551, SYDNEY, NSW. 2001.

SYDNEY EASTERN SUBURBS

Contact: Dan Lawrence, TRS-80 Eastern Suburbs Group, c/o G.P.O. Box 2551, SYDNEY, NSW. 2001.

WOLLONGONG GROUP

Contact: Paul Janson, P.O. Box 397, DAPTO, NSW. 2530.

BLUE MOUNTAINS OF N.S.W.

Contact: Greg Baubman, Tel: Home (047)51 3221
MEETINGS: 1st Friday of the month at 7.30 p.m. in Springwood Civic Centre.

NORTHERN TERRITORY

DARWIN GROUP

Contact: Tony Domigan, P.O. Box 39086, WINNELLIE, NT. 5789.

QUEENSLAND

BRISBANE GROUP

Contact: Lance Lawes, Tel: Home (07)396 2998 Bus. (07)268 1191 Ext. 15
MEETINGS: 1st Sunday each month at 2 p.m. at 21 Rodney Street, Lindum.

SOUTH AUSTRALIA

ADELAIDE GROUP

Contact: Rod Stevenson, 36 Sturt Street, Adelaide, SA. 5000. Tel: 515241 between 9-4.

VICTORIA

EASTERN SUBURBS GROUP

Contact: Mr. John Fletcher Tel: Home (03)737 9544 Bus. (03)89 0677 (9-4)
MEETINGS: 4th Wednesday of the month at 7 p.m. at Kingswood College, 355 Station Street, Box Hill.

MICOM 80 USERS' GROUP

Contact: Len Sanders Tel: (03)560 8132
MEETINGS: 1st Wednesday of each month at Alvie Hall, High Street, Mt. Waverley.

NORTHERN AND WESTERN SUBURBS

Contact: Mr. David Coupe Tel: (03)370 9590
MEETINGS: C.P.M. Data Systems, 284 Union Road, Moonee Ponds - Alternate Thursdays at 7 p.m.

PENINSULAR GROUP

Contact: M.G. Thompson - (03)772 2674
MEETINGS: 2nd Tuesday of the month (except January) at State College, Frankston.

GEELONG COMPUTER CLUB

Contact: The Geelong Computer Club, P.O. Box 6, Geelong, Vic. 3220.
MEETINGS: 2nd Thursday of the month at Tybar Engineering Hampton Street, Newton.

WESTERN AUSTRALIA

PERTH '80 USERS' GROUP

Contact: C. Powell (09)457 6849
MEETINGS: 1st Tuesday of each month at 7.30 p.m. in the Community Recreation Hall, Macdonald Street, Yokine.

** UNITED KINGDOM **

NATIONAL NATIONAL USERS' GROUP

Contact: Brian Pain, 40 High St. Stoney Stratford, Milton Keynes.

CO. DURHAM NORTH-EAST TRS-80 GROUP

Contact: Barry Dunn, 8 Ethick Tce. North Craighead, Stanley, Co. Durham, DH9 6BE. Tel: 0207-30184

COMPUTERTOWN UNITED KINGDOM (CTUK!)

Contact: Dave Tebbutt, c/o 14 Rathbone Place, London W1P 1DE

TANDY OWNERS PROGRAM & INFORMATION CO-OP.

Contact: Derek Higbee, 12 Shelley Close, Ashley Heath, Ringwood. Tel: Ringwood 6720.

NATIONAL NEWCASTLE PERSONAL COMPUTING SOCIETY

Contact: John Stephen Bone - 0632 770036

BOLTON NORTH-WEST TRS-80 GROUP

Contact: The Secretary, North-West TRS-80 Users Group, 40 Cowlees, West Houghton, Bolton BL5 3EG.

COMPUTERTOWN NORTH-EAST (CTNE!)

Contact: c/o 2 Claremont Place, Gateshead, Co. Tyne & Wear NE8 1TL. Tel: 0632-770036/643417/679119/559167

EDUCATIONAL USERS' GROUP

Contact: Dave Futcher - Head Teacher, Beaconsfield First & Middle School, Beaconsfield Rd, Southall, Middlesex. England.

WEST HERTS 80 USERS GROUP

Contact: Terry Bradbury, 20 Spruce Way, St. Albans Herts. Tel: PARK STREET 73663.

** NEW ZEALAND **

AUCKLAND

Contact: Ron Feasy Bus. 799 366 Home. 469 455
MEETINGS: 1st Tuesday of each month, 7.30 p.m. at: NZ Solenoid Co. Ltd., 28 Kalmia Street, Ellerslie, Auckland.

DON'T BE HELD BACK BY AN ANTIQUATED DISK OPERATING SYSTEM

MOVE UP TO

NEWDOS 80 **$149 incl. p&p**

NEWDOS 80 is a completely new DOS for the TRS-80 SYSTEM 80. It is well-documented, bug free and increases the power of your system many times over. It is upward compatible with TRSDOS AND NEWDOS (ie TRSDOS and NEWDOS+ programs will run on NEWDOS 80 but the reverse is not necessarily so).

These are just a few of the many new features offered by NEWDOS 80.

* New BASIC commands that support variable record lengths up to 4095 bytes long.
* Mix or match disk drives. Supports any track count from 18 to 96. Use 35, 40, 77 or 80 track 5¼ inch mini disk drives, 8 inch disk drives OR ANY COMBINATION.
* An optional security boot-up for BASIC or machine code application programs. User never sees "DOS-READY" or "READY" and is unable to "BREAK", clear screen or issue any direct BASIC statements, including "LIST".
* New editing commands that allow program lines to be deleted from one location and moved to another or to allow the duplication of a program line with the deletion of the original.
* Enhanced and improved RENUMBER that allows relocation of subroutines.
* Create powerful chain command files which will control the operation of your system.
* Device handling for routing to display and printer simultaneously.
* MINIDOS — striking the D, F and G keys simultaneously calls up a MINIDOS which allows you to perform many of the DOS commands without disturbing the resident program.
* Includes Superzap 3.0 which enables you to display/print/modify any byte in memory or on disk.
* Also includes the following utilities:
  - Disk Editor/Assembler
  - Disassembler (Z80 machine code)
  - LM offset — allows transfers of any system tape to Disk file — automatically relocated.
  - LEVEL I — Lets you convert your computer back to Level 1.
  - LVIDKSL — Saves and loads Level 1 programs to disk.
  - DIRCHECK — Tests disk directories for errors and lists them.
  - ASPOOL — An automatic spooler which routes a disk file to the printer whilst the computer continues to operate on other programs.
  - LCDVR — a lower case drives which display lower case on the screen if you have fitted a simple lower case modification.

DISK DRIVE USERS ELIMINATE CRC ERRORS AND TRACK LOCKED OUT MESSAGES FIT A PERCOM DATA SEPARATOR

$37.00 plus $1.20 p&p.

When Tandy designed the TRS-80 expansion interface, they did not include a data separator in the disk-controller circuitry, despite the I.C. manufacturer's recommendations to do so. The result is that many disk drive owners suffer a lot of Disk I/O errors. The answer is a data separator. This unit fits inside your expansion interface. It is supplied with full instructions and is a must for the serious disk user.

MPI DISK DRIVES HIGHER PERFORMANCE — LOWER PRICE

MPI is the second largest manufacturer of disk drives in the world. MPI drives use the same form of head control as 8" drives and consequently, they have the fastest track-to-track access time available — 5msec! All MPI drives are capable of single or double-density operation. Double-density operation requires the installation of a PERCOM doubler board in the expansion interface.

As well as single head drives, MPI also makes dual-head drives. A dual-head drive is almost as versatile as two single-head drives but is much cheaper.

Our MPI drives are supplied bare or in a metal cabinet — set up to operate with your TRS-80 or SYSTEM 80. All drives are sold with a 90 day warranty and service is available through MICRO-80 PRODUCTS.

MPI B51 40 Track Single Head Drive. . . . . .only $349
MPI B52 40 Track Double Head Drive. . . . .only $449

Prices are for bare drives and include p&p. Add $10.00 per drive for a cabinet and $60.00 for a power supply to suit two drives. 40 track drives are entirely compatible with 35 track drives. A 40 track DOS such as NEWDOS 80 is necessary to utilise the extra 5 tracks.

OVER 800 KILOBYTES ON ONE DISKETTE! WITH MPI 80 TRACK DRIVES

MPI 80 track drives are now available. The B91 80 track single-head drive stores 204 Kilobytes of formatted data on one side of a 5¼ inch diskette in single-density mode. In double-density mode it stores 408 Kilobytes and loads/saves data twice as quickly.

The B92 80 track dual-head drive stores 204 Kilobytes of formatted data on EACH side of a 5¼ inch diskette in single-density mode. That's 408 Kilobytes per diskette. In double-density mode, the B92 stores a mammoth 408 Kilobytes per side or 816 Kilobytes of formatted data per diskette. With two B92's and a PERCOM double, you could have over 1.6 Megabytes of on line storage for your TRS-80 for less than $1500!!

MPI B91 80 Track Single Head Drive. . . . . .only $499
MPI B92 80 Track Dual Head Drive . . . . . .only $619

Prices are for bare drives and include p&p. Add $10.00 per drive for a cabinet and $60.00 for a power supply to suit two drives. Note: 80 track drives will not read diskettes written on a 35 or 40 track drive. If drives with different track counts are to be operated on the same system, NEWDOS 80 must be used.

CARE FOR YOUR DISK DRIVES? THEN USE 3M's DISK DRIVE HEAD CLEANING DISKETTES

$30.20 incl. p&p.

Disk drives are expensive and so are diskettes. As with any magnetic recording device, a disk drive works better and lasts longer if the head is cleaned regularly. In the past, the problem has been, how do you clean the head without pulling the mechanism apart and running the risk of damaging delicate parts. 3M's have come to our rescue with SCOTCH BRAND, non-abrasive, head cleaning diskettes which thoroughly clean the head in seconds. The cleaning action is less abrasive than an ordinary diskette and no residue is left behind. Each kit contains:

- 2 head cleaning diskettes
- 1 bottle of cleaning fluid
- 1 bottle dispenser cap

PLEASE USE ORDER FORM ON PAGE 36

USE TANDY PERIPHERALS ON YOUR SYSTEM-80 VIA SYSPAND-80 - $97.50 incl. p&p

The SYSTEM-80 hardware is not compatible with the TRS-80 in two important areas. The printer port is addressed differently and the expansion bus is entirely different. This means that SYSTEM-80 owners are denied the wealth of economical, high performance peripherals which have been developed for the TRS-80. Until now, that is. MICRO-80 has developed the SYSPAND-80 adaptor to overcome this problem. A completely self-contained unit in a small cabinet which matches the colour scheme of your computer, it connects to the 50-way expansion part on the rear of your SYSTEM 80 and generates the FULL Tandy 40 way bus as well as providing a Centronics parallel printer port. SYSPAND-80 enables you to run an Exatron Stringy Floppy from your SYSTEM 80, or an LNW Research expansion interface or any other desirable peripherals designed to interface to the TRS-80 expansion port. Make your SYSTEM 80 hardware compatible with the TRS-80 via SYSPAND-80.

PROGRAMS BY MICROSOFT

EDITOR ASSEMBLER PLUS (L2/16K) $37.50 + $1.20 p&p

A much improved editor-assembler and debug/monitor for L2/16K TRS-80 or SYSTEM 80. Assembles directly into memory, supports macros and conditional assembly, includes new commands-substitute, move, copy and extend.

LEVEL III BASIC $59.95 plus $1.20 p&p

Loads on top of Level II BASIC and gives advanced graphics, automatic renumbering, single stroke instructions (shift-key entries) keyboard debounce, suitable for L2/16K and up (Not Disk BASIC)

ADVENTURE ON DISK $35.95 plus $1.20 p&p

This is the original ADVENTURE game adapted for the TRS-80. The game fills an entire diskette. Endless variety and challenge as you seek to rise to the level of Grand Master. Until you gain skill, there are whole areas of the cave that you cannot enter. (Requires 32K One Disk)

BASIC COMPILER $208 plus $2.00 p&p

New improved version, the Basic Compiler converts Disk BASIC programs to machine code, automatically. A compiled program runs, on average, 3-10 times faster than the original BASIC program and is much more difficult to pirate.

UPGRADE TO 16K FOR ONLY $30.00!!

MICRO-80's 16K MEMORY EXPANSION KIT HAS BEEN REDUCED IN PRICE EVEN MORE

Larger volume means we buy better and we pass the savings on to you. These are our proven, prime, branded 200 ns (yes, 200 nanosecond) chips. You will pay much more elsewhere for slow, 350 ns. chips. Ours are guaranteed for 12 months. A pair of DIP shunts is also required to upgrade the CPU memory in the TRS-80 – these cost an additional $4.00. All kits come complete with full, step-by-step instructions which include labelled photographs. No soldering is required. You do not have to be an experienced electronic technician to instal them.

DISK DRIVE CABLES SUITABLE FOR ANY DISK DRIVES

DC-2 2 Drive Connector Cable . . . . . $39 incl. p&p
DC-4 4 Drive Connector Cable . . . . . $49 incl. p&p

DOUBLE THE SPEED AND CAPACITY OF YOUR DISK DRIVES PERCOM DOUBLER ONLY $220 plus $2.00 p&p

Installing a Doubler is like buying another set of disk drives, only much cheaper!! The doubler works with most modern disk drives including:- MPI, Micropolis, Pertec, TEAC (as supplied by Tandy). The doubler installs in the TRS-80 expansion interface, the System-80 expansion interface and the LNW Research expansion interface in a few minutes without any soldering, cutting of tracks, etc. It comes complete with its own TRSDOS compatible double density operating system.

DOUBLE-ZAP II – DOUBLE DENSITY PATCH FOR NEWDOS 80 ONLY $53.00 plus $1.00 p&p

DISCONTINUED

If you are using NEWDOS 80, then you also need DOUBLE-ZAP on diskette. This program upgrades your NEWDOS 80 to double density with ADR (automatic density recognition). It retains all the familiar features, including the ability to mix and match track counts on the same cable. In addition, it gives NEWDOS 80 the ability to mix densities on the same cable, automatically. If you place a single density diskette in drive Ø, say and a double density diskette in drive 1, Double-ZapII will recognise this and read/write to drive Ø in single density whilst at the same time it reads/writes to drive 1 in double density!

FLOPPY DOCTOR AND MEMORY DIAGNOSTIC (by MICRO CLINIC) $29.95 plus 50c. p&p

Two machine language programs on a diskette together with manual which thoroughly test your disk drives and memory. There are 19 possible error messages in the disk drive test and their likely causes are explained in the manual. Each pass of the memory tests checks every address in RAM 520 times, including the space normally occupied by the diagnostic program itself. When an error occurs the address, expected data, and actual data are printed out together with a detailed error analysis showing the failing bit or bits, the corresponding IC's and their location. This is the most thorough test routine available for TRS-80 disk users.

BOOKS

LEVEL II ROM REFERENCE MANUAL $24.95 + $1.20 p&p

Over 70 pages packed full of useful information and sample programs. Applies to both TRS-80 and SYSTEM 80.

TRS-80 DISK AND OTHER MYSTERIES $24.95 + $1.20 p&p

The hottest selling TRS-80 book in the U.S.A. Disk file structures revealed, DOS's compared and explained, how to recover lost files, how to rebuild crashed directories – this is a must for the serious Disk user and is a perfect companion to any of the NEWDOS's.

LEARNING LEVEL II $16.95 + $1.20 p&p

Written by Daniel Lien, the author of the TRS-80 Level I Handbook, this book teaches you, step-by-step, how to get the most from your Level II machine. Invaluable supplement to either the TRS-80 Level II Manual or the System-80 Manuals.

MORE AUSTRALIAN SOFTWARE

All programs designed to run on both the TRS-80 or the SYSTEM 80 without modification. Most programs include sound

TRIAD VOL 1 – L2/16K
Cassette $10.95 Disk $15.95
+ 60c p&p

Three separate games which test your powers of memory and concentration. The programs combine graphic displays and sound:

SIMON-SEZ: Just like the electronic music puzzles on sale for more than $20. Numbers are flashed on the screen and sounded in a sequence determined by the computer. Your task is to reproduce the sequence, correctly.

LINE?: Rather like a super, complicated version of noughts and crosses. You may play against another player or against the computer itself. But beware, the computer cheats!

SUPER CONCENTRATION: Just like the card game but with more options. You must find the hidden pairs. You may play against other people, play against the computer, play on your own, or even let the '80 play on its own.

TRIAD VOL 2 – L2/16K
Cassette $10.95 Disk $15.95
+ 60c p&p

Remember those "NUMERO" puzzles in which you had a matrix of numbers (or letters) with one blank space and you had to shuffle the numbers around one at a time until you had made a particular pattern? Well, **SHUFFLEBOARD**, the first program in this triad, is just this, except that the computer counts the number of moves you take to match the pattern it has generated – so it is not possible to cheat.

MIMIC is just like SHUFFLEBOARD except that you only see the computer's pattern for a brief span at the beginning of the game, then you must remember it!

In **MATCHEM**, you have to manoeuvre 20 pegs from the centre of the screen to their respective holes in the top or bottom rows. Your score is determined by the time taken to select a peg, the route taken from the centre of the screen to the hole and your ability to direct the peg into the hole without hitting any other peg or the boundary.

VISURAMA L2/16K
Cassette $10.95 Disk $15.95
+ 60c p&p

Two programs which give fascinating, ever-changing patterns on the screen.

LIFE is the fastest implementation of the Game of Life you will see on your '80. Machine language routines create up to 1200 new generations per minute for small patterns or up to 100 per minute for the full 128 x 48 screen matrix. Features full horizontal and vertical wraparound.

EPICYCLES will fascinate you for hours. The ever-changing ever-moving patterns give a 3D effect and were inspired by the ancient Greek theories of Ptolemy and his model of the Solar system.

EDUCATION AND FUN – L1/4K, L2/16K
Cassette $10.95 Disk $15.95
+ 60c p&p

Written by a primary school teacher to make learning enjoyable for his pupils, there are five programs in both Level I and Level II to suit all systems:

BUG-A-LUG: a mathematics game, in which you must get the sum correct before you can move.

AUSTRALIAN GEOGRAPHY: learn about Australian States and towns, etc.

SUBTRACTION GAME: build a tower with correct answers.

HOW GOOD IS YOUR MATHS? Select the function (+, –, ÷ or X) and degree of difficulty.

HANGMAN: That well known word game now on your computer.

Recommended for children from 6 to 9 years.

COSMIC FIGHTER & SPACE JUNK – L2/16K
Cassette $10.95 Disk $15.95
+ 60c p&p

Both programs have sound to complement their excellent graphics. In **COSMIC FIGHTER**, you must defend the earth against seven different types of alien aircraft. It is unlikely that you will be successful but you will have a lot of fun trying!

You mission in **SPACE JUNK** is to clean up all the debris left floating around in space by those other space games. It is not as simple as it sounds and space junk can be quite dangerous unless you are very careful.

SPACE DRIVE L2/4K & 16K
Cassette $8.95 Disk $13.95
+ 60c p&p

Try to manoeuvre your space ship through the meteor storms then land it carefully at the space port without running out of fuel or crashing. Complete with realistic graphics.

STARFIRE AND NOVA INVASION L2/16K
Cassette $10.95 Disk $15.95
+ 60c p&p

Both programs include sound to improve their realism.

STARFIRE seats you in the cockpit of an X-wing fighter as you engage in battle with the deadly Darth Vader's Tie-fighters. Beware of the evil one himself and may the Force be with you.

In **NOVA INVASION**, you must protect your home planet of Hiberna from the invading NOVADIANS. You have two fixed guns at each side of the screen and a moveable one at the bottom. Apart from shooting down as many invaders as possible, you must protect your precious hoard of Vitaminium or perish!

AIR ATTACK AND NAG RACE – L2/16K
Cassette $10.95 Disk $15.95
+ 60c p&p

An unlikely combination of programs but they share the same author who has a keen sense of humour.

AIR ATTACK includes sound and realistic graphics. The aircraft even have rotating propellors! But they also drop bombs on you, so it's kill or be killed!

NAG RACE lets you pander to your gambling instinct without actually losing real money. Up to five punters can join in the fun. Each race results in a photo-finish whilst there is a visible race commentary at the bottom of the screen throughout the race. Happy punting!

FOUR LETTER MASTERMIND L2/16K
Cassette $8.95 Disk $13.95
+ 60c p&p

There are 550 four-letter words from which the computer can make its choice. You have 12 chances to enter the correct word. After each try, the computer informs you of the number of correct letters and those in the correct position. You can peek at the list of possible words but it will cost you points. Makes learning to spell fun.

MUSIC IV – L2/16K
Cassette $8.95 Disk $13.95
+ 60c p&p

Music IV is a music compiler for your '80. It allows you to compose or reproduce music with your computer that will surprise you with its range and quality. You have control over duration (full beat to 1/16 beat) with modifications to extend the duration by half or one third for triplets. Both sharps and flats are catered for as are rests. Notes on whole sections may be repeated. The program comes with sample data for a well-known tune to illustrate how it is done.

*****SAVE 00$'s ***SAVE 00$'s ***SAVE 00$'s ***MICRO-80 EXPANSION INTERFACE *****

MICRO-80's expansion interface utilises the proven LNW Research Expansion board. It is supplied fully built up and tested in an attractive cabinet with a self contained power supply, ready to plug in and go. The expansion interface carries MICRO-80's full, no hassle, 90-day warranty.

Features include:- • Sockets for up to 32K of memory expansion • Disk controller for up to 4 disk drives • Parallel printer port • Serial RS232C/20mA I/O port • Second cassette (optional)

The expansion interface connects directly to your TRS-80 L2/16K keyboard or, via SYSPAND-80 to your SYSTEM 80VIDEO GENIE

Prices: HD-010-A Expansion Interfaces with Ø K : $499.00 HD-010-B Expansion Interfaces with 32K : $549.00 HD-011 Data separator fitted (recommended) : add $29.00 HD-012 Dual cassette Interfaces fitted : add $19.00

The MICRO-80 Expansion Interface is also available in kit form.

Prices: HD-013 Kit consisting of LNW Research PC board and manual, ALL components including cabinet & power supply : $375.00 HD-011 Data separator for above $25.00 HD-013 Dual cassette Interface kit : $15.00

TURN THIS into this

for $49.00 plus $2.00 p & p

A choice of upper and lower case display is easier to read, gives greater versatility.

The Micro-80 lower case modification gives you this facility, plus the symbols for the 4 playing-card suits for $49.00 + $2.00 p. & p.

The Micro-80 modification features true below-the-line descenders and a block cursor.

Each kit comes with comprehensive fitting instructions and two universal lower-case drive routines on cassette to enable you to display lower case in BASIC programs.

The driver routines are self-relocating, self-protecting and will co-reside with other machine language programs such as Keyboard-debounce, serial interface driver programs etc.

Both programs give your TRS-80™ Model I or System 80™ an optional typewriter capability, i.e. shift for upper case.

The second programme also includes Keyboard-debounce and a flashing cursor.

You fit it. Or we can.

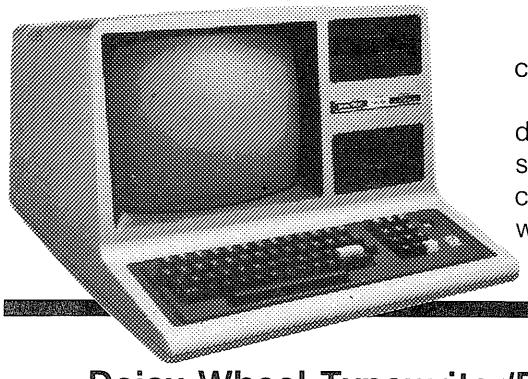
Fitting the modification requires soldering inside the computer. This should only be carried out by an experienced hobbyist or technician.

If you are at all dubious, a fitting service is available in all capital cities for only $20.00.

A list of installers is included with each kit.

Save $120 now.

ADD A DISK DRIVE TO YOUR TRS-80™ MODEL III FOR ONLY $875.00 OR ADD TWO FOR ONLY $1199.



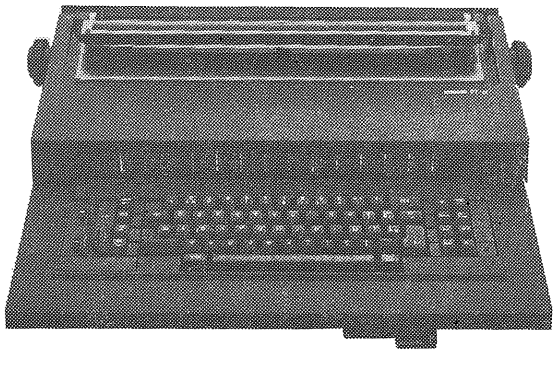
The Micro-80 disk drive upgrade for the TRS-80™ Model III contains the following high quality components:

1 or 2 MPI 40-track single head disk drives, 1 VR Data double-density disk controller board and 1 dual drive power supply plus all the necessary mounting hardware, cables and comprehensive fitting instructions, which can be carried out with a minimum of fuss by any average computer owner.

Fitting service is available for $25.00 in most capital cities.

Daisy Wheel Typewriter/Printer

ONLY $2049 INC. S.T.



MICRO-80 has converted the new OLIVETTI ET-121 DAISY WHEEL typewriter to work with the TRS-80 and SYSTEM 80 or any other microcomputer with a Centronics parallel port (RS 232 serial interface available shortly). The ET-121 typewriter is renowned for its high quality, fast speed (17 c.p.s.), quietness and reliability. MICRO-80 is renowned for its knowledge of the TRS-80/SYSTEM 80 and its sensible pricing policy. Together, we have produced a dual-purpose machine:- an attractive, modern, correcting typewriter which doubles as a correspondence quality Daisy-wheel printer when used with your micro-computer.

How good is it? - This part of our advertisement was typeset using an ET-121 driven by a TRS-80. Write and ask for full details.

BUY YOUR MODEL 3 FROM MICRO-80 AND SAVE $000's

MICRO-80 fits reliable MPI disk drives to the TRS-80 Model 3 to give system capacities and capabilities far in excess of those available elsewhere. All our conversions utilise low dissipation, switching-mode power supplies to avoid screen jitter and overheating. The disk controller boards used incorporate special compensation circuitry for 80 track disk drives and may also be used to run 8 inch disk drives with an appropriate cable and DOS.

MODEL 340
2 40 TRACK SINGLE-HEAD DISK DRIVES GIVING
350K FORMATTED STORAGE, 48K RAM **$3130**

MODEL 340+
2 40 TRACK DUAL-HEAD DRIVES GIVING
700K FORMATTED STORAGE, 48K RAM **$3350**

MODEL 380+
2 80 TRACK DUAL-HEAD DRIVES GIVING
1.4 MEGABYTE FORMATTED STORAGE, 48K RAM **$3800**

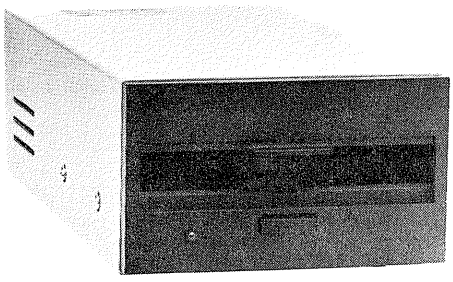
★ *NEW* ★ ★ *NEW* ★ ★ *NEW* ★

***MODEL 500 — 5+* MEGABYTE MODEL 3**
1 40 TRACK DUAL-HEAD DRIVE GIVING
350K OF FLOPPY DISK STORAGE FOR TRANSFERRING
PROGRAMS AND BACKUP, 48K RAM, EXTERNAL
5 MEGABYTE WINCHESTER SUB-SYSTEM,
CP/M (ORG 4200N) DISK OPERATING SYSTEM **$5895**

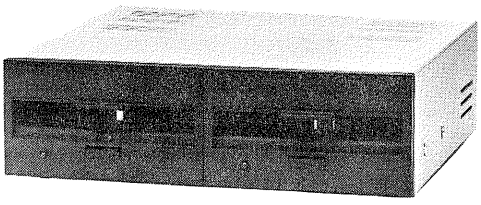
The MODEL 500 offers the high speed, mass storage capacity and reliability of a Winchester drive for thousands of dollars less than you would pay for any comparable system. Model 500 is a serious business computer able to tackle the most demanding tasks.

All prices are in Australian dollars, include Sales Tax and are subject to change without notice. Prices are FOB Adelaide. Add $20 road freight anywhere in Australia. All computers and systems carry MICRO-80's 90-day Warranty covering parts and labour.

SAVE A PACKET ON MICRO-80's DISK DRIVE PACKAGES FOR TRS-80 MODEL 1 AND SYSTEM 80 MICROCOMPUTERS



SINGLE DRIVE PACKAGE *from ...* $499



DUAL DRIVE PACKAGE *from ...* $874

Bigger volume means lower cost price, which we are passing on to you. Avoid the annoying bundle of cables, wires and separate boxes. MICRO-80 is now offering our well-proven MPI disk drives in attractive, self-contained single or dual-drive cabinets complete with internal power supply. Our drive Ø and dual-drive packages also include the appropriate version of DOSPLUS and dual-drive cable.

The best news of all is the specially reduced package prices ...
SAVE $23 — $107 over our already low prices!

Choose the appropriate system from the table below:

DRIVE TYPE	No. of Tracks	No. of Heads	Capacity	Dosplus Version	Price	* Saving
DRIVE Ø						
1 x MPI B51	40	1	100K	3.3	$499	$77.95
1 x MPI B52	40	2	200K	3.4	$639	$97.95
1 x MPI B92	80	2	400K	3.4	$799	$107.95
DRIVE 1						
1 x MPI B51	40	1	100K	—	$415	$23.00
1 x MPI B52	40	2	200K	—	$525	$23.00
1 x MPI B92	80	2	400K	—	$695	$23.00

*Represents the saving compared with buying all the items included in the package separately

•Drive Ø package includes one bare disk drive, self-contained single-drive cabinet/power supply as illustrated, two drive cable and the version of DOSPLUS indicated.

•Drive 1 package includes one bare disk drive and self-contained single-drive cabinet/power supply as illustrated.

If it's a dual-drive system you need, then take advantage of our dual-drive package and
SAVE a further $40 on the price of two single-drive packages ...

DRIVE TYPE	No. of Tracks	No. of Heads	Capacity	Dosplus Version	Price
2 x MPI B51	40 ea	1 ea	2 x 100K	3.3	$874
2 x MPI B52	40 ea	2 ea	2 x 200K	3.4	$1125
2 x MPI B92	80 ea	2 ea	2 x 400K	3.4	$1454

Dual-drive package includes two bare disk drives, self-contained dual-drive cabinet/power supply as illustrated, two drive cables and the version of Dosplus indicated.

NOTE: All 40 track drives are completely compatible with 35 track operating systems such as TRSDOS. DOSPLUS allows you to realise an additional 14% capacity compared with TRSDOS. Under DOSPLUS 3.4, 80 track drives can read 35/40 track diskettes.

All disk drive components are still available separately:

BARE DRIVES — MPI drives offer the fastest track-to-track access time (5 milliseconds) available. All drives are capable of operating in double density for 80% greater storage capacity.

	Price		Freight
MPI B51 40 track, single-head, 100K	$399	New, Reduced Price	$5.00
MPI B52 40 track, dual-head, 200K	$449		$5.00
MPI B92 80 track, dual-head, 400K	$619		$5.00
Simple, wrap-around cabinet	$12		$2.00
Separate, dual-drive power supply	$85		$8.00

	Price	Freight
Self-contained, single drive cabinet/power supply	$99	$5.00
Self-contained, dual-drive cabinet/power supply	$135	$5.00
Two drive cable	$39	$2.00
Fan drive cable	$49	$2.00
DOSPLUS 3.3	$99.95	$2.00
DOSPLUS 3.4	$149.95	$2.00

Prices are FOB Adelaide. Add $5.00 freight for single drive package, $10.00 for dual-drive package. Prices are in Australian dollars. Freight is road freight anywhere in Australia.

All items carry a 90-day parts and labour warranty. Repairs to be carried out in our Adelaide workshops.

SOFTWARE BY AUSTRALIAN AUTHORS

All our software is suitable for either the SYSTEM 80 or the TRS-80

NEW SOFTWARE FROM MICRO-80 PRODUCTS

BUSINESS PROGRAMS

MICROMANAGEMENT STOCK RECORDING SYSTEM (L2/16K)

Cassette version. . . . . $29.95 + $1.00 p&p
Stringy Floppy version. . . . . $33.95 + $1.00 p&p

This system has been in use for 9 months in a number of small retail businesses in Adelaide. It is therefore thoroughly debugged and has been tailor made to suit the requirements of a small business. MICROMANAGEMENT SRC enables you to monitor the current stock level and reorder levels of 500 different stock items per tape or wafer. It includes the following features:—

- Add new items to inventory
- Delete discontinued items from inventory
- List complete file
- Search for any stock number
- Save data to cassette or wafer
- Load data from cassette or wafer
- Adjusts stock levels from sales results and receipt of goods
- List all items requiring reordering

We can thoroughly recommend this program for the small business with a L2/16K computer.

SCOTCH BRAND COMPUTING CASSETTES

Super-quality personal computing cassettes.
C-10 pack of 10 ... ... $26.00 incl. p&p
C-30 pack of 10 ... ... $28.00 incl. p&p

UTILITIES

S-KEY by Edwin Paay $15.95 plus 50c. p&p

S-KEY is a complete keyboard driver routine for the TRS-80 and becomes part of the Level II basic interpreter. With S-KEY loaded the user will have many new features not available with the standard machine.

S-KEY features:

* S-KEY provides an auto-repeat for all the keys on the keyboard. If any key is held down longer than about half a second, the key will repeat until it is released.
* Graphic symbols can be typed direct from the keyboard, this includes all 64 graphic symbols available from the TRS-80/SYSTEM 80.
* S-KEY allows text, BASIC commands and/or graphics to be defined to shifted keys. This makes programming much easier as whole commands and statements can be recalled by typing shift and a letter key.
* Because S-KEY allows graphics to be typed directly from the keyboard, animation and fast graphics are easily implemented by typing the appropriate graphics symbols directly into PRINT statements.
* S-KEY allows the user to LIST a program with PRINT statements containing graphics, properly. S-KEY does this by intercepting the LIST routine when necessary.
* S-KEY allows the user to list an updated list of the shift key entries to the video display or line printer.
* S-KEY can be disabled and enabled when required. This allows other routines which take control of the keyboard to run with S-KEY as well.

Each cassette has TRS-80, DISK and SYSTEM 80 versions and comes with comprehensive documentation.

BMON by Edwin Paay $19.95 plus 50c. p&p
THE ULTIMATE HIGH MEMORY BASIC MONITOR L2/16-48K

Our own personnel refuse to write BASIC without first loading this amazing machine language utility program into high memory! BMON Renumbers; Displays BASIC programs on the screen while they are still loading; tells you the memory locations of the program just loaded; lets you stop a load part-way through; merges two programs, with automatic renumbering of the second so as to prevent any clashes of line numbers; recovers your program even though you did type NEW: makes one program invisible while you work on a second (saves hours of cassette time!); lists all the variables used in the program; makes SYSTEM tapes; lets you Edit memory directly . . . the list goes on and on. Cassette comes with 16K, 32K and 48K versions, ready to load. Can anyone afford NOT to have BMON?

EDUCATIONAL

RPN CALCULATOR (L2/16K & 32K)
$14.95 $ 50c. p&p

Give your computer the power of a $650 reverse polish notation calculator with 45 functions and selectable accuracy of 8 or 16 digits. The main stack and registers are continuously displayed whilst the menu is always instantly accessible without disturbing any calculations or register values. The cassette comes with both the 16K and 32K versions, the latter giving you the additional power of a programmable calculator. Comes with a very comprehensive 15 page manual, which includes instructions to load and modify the 32K programmable version to run in 16K. Whether for business or pleasure, this package will prove invaluable, and turn you '80 into a very powerful instrument.

GAMES

MICROPOLY (L2/16K) $8.95 + 60c p&p

Now you can play Monopoly on your micro. The old favourite board game has moved into the electronic era. This computer version displays the board on the screen, obeys all the rules and, best of all, the banker does not make mistakes with your change!

CONCENTRATION (L2/16K) $8.95 + 60c p&p

Another application of supergraphics. There are 28 "cards" displayed on the screen, face down. Players take it in turn to turn them over with the object of finding matching pairs. There are 40 different patterns which are chosen at random, so the game is full of endless variety. This is of particular value in helping young children to learn the art of concentrating and, at the same time, to introduce them to the computer.

METEOR AND TORPEDO ALLEY (L2/16K)
$10.95 + 60c p&p

Those who frequent games arcades will recognize these two electronic games. In METEOR you must destroy the enemy space ships before they see you. In its most difficult mode, the odds are a thumping 238 to 1 against you being successful. In torpedo alley you must sink the enemy ships without hitting your own supply ship. Both games include sound effects and are remarkably accurate reproductions of the arcade games.

AUSTRALIAN SOFTWARE (Cont.)

GAMES

SHEEPDOG (L2/16K) **$8.95 + 60c p&p**

Ever wondered how a sheepdog manages to drive all those awkward sheep into a pen? Well, here is your chance to find out just how difficult it is and have a lot of fun at the same time. You control the sheepdog, the computer controls the sheep! As if that isn't enough, look out for the dingoes lurking in the bush!

U BOAT **$8.95 + 60c p&p**

Real time simulation at its best! Comes with working sonar-screen and periscope, a full rack of torpedoes, plenty of targets, working fuel and battery meters, helpful Mothership for high-seas reprovisioning and even has emergency radio for that terrible moment when the depth charges put your crew at risk. Requires Level II/16K.

SPACE INVADERS WITH SOUND **$8.95 + 60c p&p**

Much improved version of this arcade favourite with redesigned laser and cannon blasts, high-speed cannon, 50 roving drone targets, 10 motherships and heaps of fun for all. Level II with 4K and 16K versions on this cassette.

GOLF (L2/16K) **$8.95 + 60c p&p**

Pit your skills of mini-golf against the computer. Choose the level of difficulty, the number of holes and whether you want to play straight mini golf or crazy golf. Complete with hazards, water traps, bunkers and trees. Great fun for kids of all ages.

DOMINOES(L2/16K) **$8.95 + 60c p&p**

Pit your skill at dominoes against the computer, which provides a tireless opponent. Another application of supergraphics from the stable of Charlie Bartlett. Dominoes are shown approximately life size in full detail (except for colour!). The monitor screen is a window which you can move from one end of the string of dominoes to the other. Best of all, you don't lose any pieces between games!

KID'S STUFF (formerly MMM-1) **$8.95 + 60c p&p**

Three games on one cassette from that master of TRS-80 graphics, Charlie Bartlett. Includes INDY 500, an exciting road race that gets faster and faster the longer you play, SUBHUNT in which your warship blows up unfortunate little submarines all over the place, and KNIEVEL (as in motorcycle, ramp and buses).

OTHER PROGRAMS

INFINITE BASIC BY RACET (32K/1 DISK)
$49.95 + 50c. p&p

Full matrix functions – 30 BASIC commands; 50 more STRING functions as BASIC commands.

GSF/L2/48K **$24.95 + 50c. p&p**

18 machine language routines including RACET sorts.

BUSINESS ADDRESS AND INFORMATION SYSTEM (48K/DISK) **$24.95 + 50c. p&p**

Allows you to store addresses and information about businesses, edit them and print them out.

HISPED (L216, 32 or 48K) $29.95

This machine language program allows you to SAVE and LOAD programs and data to tape at speeds up to 2000 band (4 times normal) using a standard cassette recorder. A switch must be installed to remove the XRX III loading board, if fitted.

LOWER CASE FOR YOUR TRS-80/SYSTEM 80

Kit only $49.00 plus $2.00 p&p

Give your TRS-80 or SYSTEM 80 a lower case display with proper descenders and a block cursor (similar to the TRS-80 Model III). Also includes symbols for the four suits of cards. Includes full fitting instructions, all necessary components and a special machine language driver program to enable lower case in BASIC. The modification is similar to the Tandy model and does not work with Electric Pencil without further modifications.

These kits require disassembly of your computer and some soldering. They should only be installed by someone who has experience in soldering integrated circuits, using a low power, properly earthed soldering iron. If you do not have the necessary experience/equipment, we will install the modification for you for $20 plus freight in both directions. Make sure you arrange the installation with us first, before despatching your computer, so that we can assure you of a rapid turn-around. We are also arranging to have installers in each State. See elsewhere in this issue for their names and addresses.

PRICES

Cat No.
HD-020 Lower case mod kit for TRS-80
$49.00 plus $2.00 p&p
HD-021 Lower case mod kit for SYSTEM-80
$49.00 plus $2.00 p&p

EPSON MX-80 PRINTER

ONLY *$949 Inc. Cable for TRS-80 and p&p
(*Printer only – $940 incl. p&p)

The EPSON MX-80 printer is compact, quiet, has features unheard of only 2-3 years ago in a printer at any price and, above all, is ultra-reliable. All available print modes may be selected under software control. Features include:

- high quality 9x9 dot-matrix character formation
- 3 character densities
  - . 80 characters per line at 10 chars/inch
  - .132 characters per line at 16.5 chars/inch
  - . 40 characters per line at 5 chars/inch
- 2 line spacings
  - . 6 lines per inch 8 lines per inch
- 80 characters per second print speed
- bi-directional printing
- logical seeking of shortest path for printing
- lower case with descenders
- TRS-80 graphics characters built in
- standard Centronics printer port

The bi-directional printing coupled with the logical seeking of the shortest print path (which means that the print head will commence printing the next line from the end which requires the least travel, thereby minimising unutilised time) gives this printer a much higher throughput rate than many other printers quoting print speeds of 120 c.p.s. or even higher.

GREEN SCREEN SIMULATOR

$9.50 incl. p&p

The GREEN SCREEN SIMULATOR is made from a deep green perspex, cut to fit your monitor. It improves contrast and is much more restful to the eyes than the normal grey and white image.

All editorial staff of MICRO-80 are now using GREEN SCREEN SIMULATORS on their own monitors.

Please make sure to specify whether you have an old (squarish) or new (rounded) style monitor when ordering. Not available for Dick Smith monitors.

***** HOW TO USE THE TRS-80 LEVEL I ARRAY EFFECTIVELY by Otto Reberg *****

BEGINNERS:

The level I array is very unusual. It has only one dimension and is automatically DIM'med to (MEM.ory/4-1) (Divide by 4 because each element occupies 4 bytes and subtract 1 byte which separates program and array). The following example shows an easy way to use the array:

```
10 LET A(1)=0 : LET A(2)=1
20 FOR I=3 TO 20
30 LET A(I)=A(I-1)+A(I-2)
40 PRINT A(I);
50 NEXT I
```

This program generates and prints the number sequence of Fibonnachi, in which the two first numbers are 0 and 1, and each of the following numbers are the sum of the two preceders. Let us analyse the program: In line 10, the two first numbers are put into A(). In line 30, A(I) is set to the sum of A(I-1) and A(I-2), and the result is printed in line 40. This is repeated several times (line 20 and 50).

ADVANCED PROGRAMMERS.

You often have to use several arrays within one program. The solution to this is to divide A() into many pieces. Suppose you want to create three arrays, one with 14 elements, the second with 10 elements and the third with 20. You want to print the F'th element of each array. This is written:

```
PRINT A(F), A(14+F), A(14+10+F)
```

(Just remember to keep the index within the proper limits).

EXPERTS.

If you are in need of a 2-dimensional array, also this can be done. You have an array (10,5) (i.e. having 10 rows and 5 columns) and want to know the value of (I,J). You then type:

```
PRINT A(5*I+J)
```

By using the simple principles shown above, you are able to create rather complicated structures: You have an array (5,7,9), the elements from 0 to 20 in A() are already used, and you would like to put PI into (I,J,K). I suggest that you should express it this way:

```
LET A(20+9*7*I+9*J+K) = 3.1415926
```

You should always be careful with these arrays, as they occupy large parts of your RAM. This one ate 5*7*9*4=1260 bytes.

- 000000000 -

***** TRANSFERRING TOUCHTYPE TO DISK by G.F. Hilder *****

TOUCHTYPE is one of the more popular educational programs we have published on cassette. It does take a long time to load in all the lessons, however and for some time there has been a Readers' Request to publish instructions for transferring this program and its data files to disk. Mr. Hilder has accomplished this task and describes this procedure as follows.

INSTRUCTIONS

1. Load the TOUCHTYPE tape into the recorder and adjust the volume control to a suitable level. I advise you not to attempt the rest of the instructions until you are sure the tape will be read reliably.

2. Place the recorder in the playback mode and CLOAD the first program.

3. List the program to make sure the load was successful then type SAVE"TTYPE1/BAS".

4. Leaving the recorder in the play mode type NEW then type in and run the following program:-

```
10 CLEAR 1000
20 OPEN"O",1,"TTYPE3/TXT"
30 INPUT#-1,D$
40 PRINT D$
50 D$=CHR$(34)+D$+CHR$(34)
60 PRINT#1,D$
70 CLOSE:END
```

The following message will be displayed and then saved on disk.

```
################
#   THANK YOU   #
################
```

5. CLOAD the second program.
6. Type SAVE"TTYPE2/BAS"
7. Load and run the following program.

```
1Ø  CLEAR 1ØØØ
2Ø  OPEN"O",1,"TTYPE4/TXT"
3Ø  INPUT#-1,A,B,D$
4Ø  PRINT A,B,D$
5Ø  D$=CHR$(34)+D$+CHR$(34)
6Ø  PRINT#1,A,B,D$
7Ø  GOTO3Ø
```

You should see each block of data displayed on the screen and then saved to disk. The total time involved will be about 20 mins. but it is worth watching the proceedings to make sure that there are no data reading errors.

N.B. With suitable changes to the input and print statements the above procedure would also allow the production of back-up tape copies.

8. The following changes must also be made to allow the programs to run under Disk BASIC:

   (a) Type LOAD"TTYPE1/BAS"
   (b) Type 25 OPEN"I",1,"TTYPE3/TXT"
   (c) Type 35 CLOSE
   (d) Modify lines 3Ø and 8Ø to look like the following:-

```
3Ø  FORK=ØTO6ØØØ:NEXT:CLS:PRINT"THIS  COURSE  RELIES  HEAVILY  ON  THE  USE  OF  DATA
    STATEMENTS  AND  THE  LOADING  OF  SUBROUTINES  FROM  DISK.  PLEASE  LEAVE  YOUR  DISK
    INSERTED. ":INPUT#1,D$:PRINTD$:
8Ø  PRINT"READY";CHR$(34);"YOU  SHOULD  TYPE  ";CHR$(34);"RUN";CHR$(34);"  AND  PRESS
      ENTER  YOU WILL THEN BE ASKED TO SELECT YOUR STARTING POINT.  ";:FORT=ØTO1ØØØ:
    NEXT:CLEARØ:LOAD"TTYPE2/BAS:1"
```

   (e) Type SAVE"TTYPE1/BAS"
   (f) Type LOAD"TTYPE2/BAS"
   (g) Type 75 OPEN"I",1, "TTYPE4/TXT"
   (h) Modify line 8Ø to look like:-

```
8Ø ONERRORGOTO8Ø: INPUT#1,LL,C,TE$:IFLN<LLTHENCLS:PRINT"REWIND IT PLEASE":GOSUB27Ø:CLS
   :GOTO8Ø:ELSEIFLL<>LNPRINT@256,"READING";LL,:GOTO8Ø
```

   (i) Type SAVE"TTYPE2/BAS"

I also noticed an error in line 13Ø of my copy of TOUCHTYPE which resulted in the "Ø" being omitted from the keyboard display. The correct line 13Ø is shown below.

```
130  CLS:PRINT@384,STRING$(64,131);"    1!    2";CHR$(34);"    3#    4$    5%    6&    7'    8(    9)
Ø    :*    -=","  [    Q    W    E    R    T    Y    U    I    O    P    @    ";CHR$(93),"  ";CHR$(92);"
A   S   D   F   G   H   J   K   "
```

- 0000000000 -

***** DESCRIPTION OF THE FUNCTIONS PERFORMED BY THE EXPANSION INTERFACE - by Andrew Jack *****

The Expansion Interface for the TRS-80 performs five main user accessible functions:

(1) A relay to provide two cassette ports.
(2) A centronics parallel printer port.
(3) A floppy disk controller and support circuitry.
(4) Provision for an extra 32K of RAM.
(5) 25ms interrupts (real time clock).

The relay which selects the cassette port is accessible through memory location 37E4H (143Ø8), using POKE143Ø8,Ø cassette one is selected (normal), and POKE143Ø8,1 selects cassette two. This function has more possibilities than are immediately obvious, for example, by shorting pins one and three on the Expansion Interface cassette input, and connecting a low powered circuit to pins one and three of cassette output port two, it is possible to close the connected circuit by a simple POKE143Ø8,1 and open it with POKE143Ø8,Ø. Similar results can be obtained by using the relay inside the keyboard, but it is only suitable for very low power use and has a tendency to weld shut, and also to revert to open when in BASIC command mode.

The printer port is a straightforward centronics parallel port, driven by the LII ROM printer driver and simply transmits data when a READY status is shown in location 37E8H (14312). This location can be PEEKed to show whether a printer is connected, and if it is out of paper. If the value is less than 127, the printer is ready to roll. If it is 127 then the printer is out of paper. There is more information provided by this location, but it is not the same for every printer, so the best way is to experiment and find out the status checks for yourself.

The floppy disk control circuitry mainly consists of the Western Digital FD17713 floppy disk controller, which is detected by the Microsoft ROM's test routine and instructed to commence loading the Disk Operating System (DOS) if (BREAK) is not pressed to generate a forced jump to LII. The Expansion Interface then accepts the data and control information is passed to the floppy disk through the DOS.

The extra RAM is straightforward, simply being an extension of the keyboard RAM, with the exception of the more complicated addressing techniques made necessary by the limited number of pins on the card edge.

The 25ms interrupts are not used by LII and the interrupt routine automatically returns without doing anything. This useful feature of the Z-8Ø is, however, exploited to a small extent in TRSDOS, and to a larger extent by NEWDOS8Ø and other DOS's. In TRSDOS it is used for the real time clock, and in NEWDOS8Ø its full potential is used to check for the triple key depressions, and for the print spooler.

I own a 16K LII with lowercase and an E/I with 32K of MICRO-80 RAM, along with two Tandy drives and Microline 80 printer, and I have found that the Expansion Interface performs all these functions for what is quite a modest price, when compared with the expansion connectors for computers like the Apple and, with the exception of Tandy's $220 for 16K RAM price tag (MICRO-80's RAM is faster and only $30 for 16K), is good value.

- 0000000000 -

★ ★ SOFTWARE SECTION ★ ★

***** SCREEN COPY UTILITY L2/m.l. - by N.J. Coleman *****

This program provides two extra BASIC instructions which enable a copy of the screen display to be stored, and recalled, at will.

The program provides these two commands:

- LSET to store an image of the screen.
- RSET to recall an image.

Both commands are reasonably self-explanatory.

Since the program is in machine language, it is very fast - instantaneous to the user. This makes it ideal for graphics and board game programs where a design must be redrawn constantly.

The program is very simple for a BASIC user. First set Mem Size to 31690 and then key in the program. Once the program has been RUN it may be deleted and the two new commands used in any BASIC Program. (Change Line 23Ø to 23Ø DELETE - 23Ø)

For a machine language programmer this program is simplicity itself. All it does is:

(i) set up the LSET and RSET vectors and return to BASIC.
(ii) upon the respective commands being used, either the screen memory is moved to a reserved portion of high RAM, or the reserved RAM is moved to the screen locations; all done using the LDIR command.

Since a LD is in fact a copy instruction, the original remains undisturbed and ready for continued use.

- 0000000000 -

***** FLASHING MESSAGE ROUTINE L2/16K - by A. Park *****

If you have ever wanted to draw attention to a particular piece of information within a program when the screen is packed full of data, the answer is to make the important line(s) flash. This can be achieved by using BASIC statements. However, they waste a lot of memory, especially if you want to use the keyboard and flash the message at the same time.

This program flashes any one line message at a specific rate without being under BASIC program control. It might just be the finishing touch to make your programs more impressive, especially when you find that after typing BREAK, LIST and NEW, the message is still flashing. This makes it the ideal program to flash rude messages at your mother-in-law (as she won't know how to turn it off!)

Two commands have been included to allow the user to stop or start the program after it has been initialized. Typing KILL then ENTER/NEWLINE, whilst in BASIC, will stop the program and typing LOAD will re-initialize it. Disk BASIC users will obviously have to be content with continuous operation.

For those of you with good memories (no pun intended), a reasonable amount of the program should look familiar. The reason for this is that it is designed around Eddy Paay's Cursor Driver program, listed in MICRO-80's November issue, for the sake of simplicity.

The program is best entered via an editor/assembler for a number of reasons:

1. The copyright messages will be printed whilst loading, thus indicating that the program is loading correctly.
2. It is much easier to use a DEFM statement rather than to calculate the hex equivalent of each individual character.
3. The program will run itself, without typing "/".

If the program is entered by an editor/assembler, ignore the first two columns of the LISTing and type in the source code, starting at line 11Ø. The label VIDEO may be changed to position the message anywhere on the screen (the top line is most suitable due to linefeed action).

One word of caution - the first character of your message in line 113Ø must not be a space as the message will not flash if this is done.

The program sets memory size by itself, as explained by Eddy in the MICRO-80 November issue and so it is not necessary to answer MEMORY SIZE? or READY?

Disk BASIC users should use location 4Ø49H for TOPMEM and delete lines 38Ø to 49Ø inclusive and lines 119Ø to 129Ø.

The program may also be entered via a monitor. BMON would not be suitable here without the addition of a block move routine, described in the September 1980 issue of MICRO-80. ZMON L from the MICRO-80 Software Library would be suitable. In this case, the object code should be entered directly starting at location 7EE5H by entering F3 then 21, 0C etc. Locations 7F71H to 7FBOH must be filled with the HEX equivalent of the ASCII characters to be used in your message, e.g. in the example given, the first character is "=" whose ASCII code in HEX is 3D (61 in decimal). The first character must not be a space (or 20H) and all of the locations should be filled so that one complete line is printed. Locations 7FB2H to 7FF1H must all be filled with 20H's so that the previous message is wiped out. If all goes well, you should finish at location 7FFEH after typing in 05H. Punch out a tape with the following parameters:

START = 7EE5H END = 7FFEH ENTRY = 7EE5H NAME = MESAGE

The message is embedded in the machine code from address 7F71H to 7FBOH. To change the message, it would normally be necessary to alter the source code and reassemble. If you include the following lines of BASIC code in your own BASIC program, then you may change the message at will. You may even change the message which flashes at different stages of your program and shift it around the screen using the POKES shown in Line 32Ø.

```
100 X$="THIS IS THE MESSAGE YOU WANT TO FLASH.  MUST BE 64 CHARACTERS LONG, PAD
WITH SPACES IF NECESSARY"
110 FOR X=0 TO 63
120 POKE 32625+X,ASC(MID$(X$,X+1,1))
130 NEXT X
140 REM THE FOLLOWING CODE WILL ALLOW YOU TO SELECT THE LINE ON THE SCREEN IN WH
ICH YOUR MESSAGE WILL BE DISPLAYED.
150 INPUT "LINE NO. REQUIRED (1-16)";LN: IF LN<1 OR >16 THEN 150
160 ADDRESS=LN*64+15296:LSB=ADDRESS AND 255:MSB=INT(ADDRESS/256)
170 POKE 32557,LSB:POKE 32558,MSB:
    POKE 32569,LSB:POKE 32570,MSB:
    POKE 32596,LSB:POKE 32597,MSB
```

The BASIC program which is LISTed after the machine language source code is used to change the flash rate of the message if desired. Rates of less than 100 tend to make the display a little hard to read due to keyboard scanning. Both programs do NOT have to be in the computer at the same time as the flash rate is set to a default value of 592(Dec) by the machine language program.

- 0000000000 -

***** LONGVARS L2/48K - by R.T. Worley *****

One of the limitations of the MICROSOFT BASIC used in '80 microcomputers is the restriction placed on variable names. It can be very helpful when writing a program to use meaningful words as variables, such as LENGTH or WIDTH or perhaps MIDDLE. These words make the program logic much easier to follow and the BASIC code is, to an extent, self-documenting. This approach is possible on the TRS-80 but with two important restrictions. The variable name must not contain any reserved words such as LEN and MID in the examples above, and the interpreter recognises only the first two letters of the variable name. This means that the first two letters in each long variable name used must differ from all others. In practice, these two limitations are so restricting that only the most intrepid programmers use long variable names in all but very small programs.

LONGVARS overcomes these limitations. You may use whatever names you wish, such as BECOS (includes a reserved word), and AARDVARK and AARDVOLF (to represent different variables), and COUNT OF WORDS (includes blanks). The only restriction is that variable names starting with a reserved word must be followed by a type flag (!,#,$, or %).

When you have written your program, and it is in memory (and a spare copy saved!) load and run LONGVARS. It will process your program systematically altering all variable names so that different variables have different one- or two-letter names. Variable types are preserved. However all statements defining variable types must occur in the program before any variable is used (and it is preferable to type CLEAR (ENTER) before running LONGVARS to reset the variable type table to the default value of single precision).

LONGVARS may also be used to change the names of specified variables only, or to produce a cross-reference of all variables (or specified variables only) in a program. It has an option of adding type flags to all variables, and another option of not using single letter variable names (always using double letter names may reduce variable access time by up to 5%).

The program produces its listing on either printer or video. Video output may be halted by pressing any key, and restarted by pressing a different key.

The source code runs to over 20 pages so is, unfortunately, too long to reproduce in the magazine. Instead, a hex dump has been printed. Enter the hex listing using a suitable monitor such as BMON (Issues 3, 4 and 5) or ZMON (from the MICRO-80 Software Library - Vol 1). The code is 2836 bytes long, starting at F400, finishing at FF14 with an entry point at F9A4. Should you require a hard copy listing of the source code, a photocopy is available from MICRO-80 for $5.00 including postage. A dump of the source code is included on both the monthly cassette and disk. Line numbers mentioned in the remainder of these instructions refer to the source code.

The value of 100 loaded into register BC in line 1120 is to allow room for program expansion. This may be altered to a larger value if required, though expansion will only occur when processing a normal BASIC program extending variable names to two characters and/or adding type flags.

The following is a brief description of how LONGVARS works. After first finding out what you wish to do, it scans the BASIC program in memory looking for variables. Whenever it finds one it scans the table of variables it has already found (or you have entered) for a match. If there is a match it adds the line number to its table and continues. If there is no match it adds the name and the line number to its tables (unless specified variables only are being handled). During this phase a graphics character at the top of the screen will wiggle to indicate each new line scanned.

In the next phase the tables will be scanned and the variables listed (for a cross reference), new names will be allocated and listed if required, and the lines in which the variable occurs will be listed. An unusual feature is that names are allocated in decreasing frequency of occurrence, in order that space may be saved when allowing single letter names.

In the final phase (if required) the program is scanned again and the old variable names replaced by the new names. Once again, the graphics byte wiggles to confirm that the program is working. When loaded the program displays the following menu:-

```
B...RETURN TO BASIC
C...COMPLETE RENAME OF VARIABLES
S...SPECIFIED VARIABLE RENAMING
X...CROSS REFERENCE ONLY
?
```

Typing "B" simply returns you to BASIC leaving the amended program intact in memory.
Type "C" and LONGVAR will display:-

```
ENTER VARIABLE NAMES THAT YOU WISH TO DECLARE, IE. ALL NAMES THAT INCLUDE RESERVED WORDS. THE
TYPE BYTE MUST BE INCLUDED.
PRESS ENTER AFTER EACH ONE
ENTER "*" TO TERMINATE
```

You should now type in any variable names that you have used that included reserved words, i.e.
FORK$ (this contains the reserved word FOR)
STIFF! (this contains the reserved word IF). Note also the inclusion of the type byte, even if the variable appears in your program as STIFF it must be identified for LONGVAR as STIFF!

The program will then ask you:-

PRESS "Y" IF YOU SPECIFIED ANY VARIABLES STARTING WITH RESERVED WORDS, ELSE ANY OTHER KEY

Using the examples of STIFF! and FORK$, the answer would be "Y" as FORK$ starts with a reserved word. The next prompt from the program asks:-

PRESS "T" TO SET ALL VARIABLES WITH TYPE DECLARATION FLAG ELSE PRESS ANY OTHER KEY

THis will simply add a type flag to all variables. A note of caution here: if your lines already contain the maximum number of characters your program could be scrambled. Always SAVE your program first.

The next prompt from the program is:-

PRESS "S" TO ALLOW SINGLE LETTER NAMES, ELSE ANY OTHER KEY

If you type "S" then LONGVAR will replace all your variable names with single letter names, i.e. A,B,C,D,E,F,G etc. if you type any other key then LONGVAR will use double letter names, i.e. AA,BB,CC,DD,EE,FF,GG etc.

The last question asked by this section of the program is:-

PRESS "V" TO SET VIDEO OUTPUT, ANY OTHER KEY FOR PRINTER

Depending on your response, the display will either go to the VDU or the printer. After that question has been answered, the program will start renaming your variables listing the original variable name, the name it has been changed to and the line numbers where the changes were made. E.g.

```
VARIABLE STIFF! RENAMED TO A! IN LINE     70     80     90     105
VARIABLE FORK$  RENAMED TO B$ IN LINE     35     60    130
etc. etc.
```

While the renaming is taking place a graphic dot at the top left of the screen will wiggle anytime that the program appears inactive so as to show that LONGVAR is still alive.

The "S" function on the program menu allows you to rename specified variables whilst leaving the other variables in your program alone. When selected, the program will display:-

ENTER VARIABLE NAMES THAT YOU WISH TO ALTER, USING THE FORM VT=N(T), WHERE V(...V) IS THE OLD NAME, T IS THE TYPE BYTE AND N(...N) IS THE NEW NAME, PRESS ENTER AFTER EACH ONE ENTER "*" TO TERMINATE

Using our previous example variables, they would be typed in exactly as follows:-

```
STIFF!=A!
FORK$=B$
```

or if the old names had no type byte then you would type:-

```
STIFF=A!
FORK=B$
```

or

```
STIFF=A
FORK=B
```

Remember though, if you use the last example, LONGVAR does not check or change DEF statements. So if at the beginning of your original program you had:- DEFSTR F$ and you used the last example of:- FORK=B without type flags, then you should also change the DEF statement to DEFSTR B$

When renaming starts the program will list the variables being changed, whilst wiggling the graphics dot previously mentioned.

```
VARIABLE STIFF! RENAMED TO A! IN LINE     10
VARIABLE FORK$   RENAMED TO B$ IN LINE    95
etc. etc.
```

The last function "X", in LONGVAR simply lists out all the variables in the program and the lines that they are in. The program provides the option for this listing to go to the VDU or the printer, e.g.

```
VARIABLE A! IN LINE     70     80     90     100
VARIABLE A$ IN LINE     65     90     78     105
VARIABLE B$ IN LINE    200    205    300     375
```

LOADING LONGVAR

With NEWDOS:-

Type BASIC, 62464
Load in the BASIC program to be worked on.
Type CMD"LONGVAR/CMD"
The LONGVAR menu will then be displayed and you can use the functions you require.

With TRSDOS

Type LOAD LONGVAR/CMD
Type BASIC
Answer MEMORY SIZE with - 62464
Load in the BASIC program to be worked on.
Type SYSTEM
In response to the next prompt type:-

/63908

The LONGVAR menu will then be displayed and you can use the functions you require.

- 0000000000 -

***** THE MIND READER - by P.J. Smith *****

Is this Artificial Intelligence? or just computer trickery!!! Well you might be able to deceive your computer-illiterate friends or sceptical relatives. The mind reader is simple, fast and entertaining. Simply think of a number, the computer will show you several lists of numbers and will then ask you if you see the number you thought of. THEN it will stare at you very deeply and will tell you the number you picked and it's right EVERY time. The Mind Reader does not reveal his tricks, but you should be able to unravel the mystery by studying the program listing.

- 0000000000 -

***** AUSTRALIAN INCOME TAX CALCULATOR L2/4K by D. Weaver *****

In Australia, every citizen is required to lodge a Taxation Return during the month of July each year, in which he or she discloses all earnings for the preceding financial year (July 1 to June 30). (Yes, we know there are exceptions but we don't need to go into that here). The Government, in its munificence, allows one or two minor items of expenditure as "deductions" (i.e. non-taxable). After identifying these, the taxpayer is then required to calculate his net taxable income and thence the tax which is payable from a table supplied. The majority of the population works for someone else and pays tax each week or month on a pay-as-you-earn (PAYE) scheme. These people complete an "S" (for Salaries and Wages) Income Tax Return Form. Having calculated how much tax he should have paid and knowing from the "Group Certificate" supplied by his employer, how much he has paid, the taxpayer then completes the form, sends it off and waits expectantly if the Taxation Department owes him money or, with great anxiety if the opposite is true.

Once upon a time, the Government always made sure that everyone overpaid under the PAYE scheme, thus everyone was entitled to a refund (often of several hundred dollars). Such a situation perpetrated by anyone else would, of course, be fair game for all the consumer protection authorities, etc. but passed with little comment except from a few "cranks". Indeed, from August on, one was treated to the curious sight of taxpayers rejoicing as they received back their own money which had been forcefully removed from them over the previous 12 months! This became a popular source of "mad-money" and many projects were financed by the refund cheques. Holidays were taken, new carpets purchased, even down payments made on new cars all from this windfall income. This made the annual preparation of the Income Tax Return a tedious but rewarding exercise. Then, suddenly, it all changed. The Government adjusted the PAYE rates so that they very nearly collected the exact amount of money due during the year and refunds became negligible in size. By now, our overseas readers will be making plans to move to Australia where we obviously have a uniquely honourable Government which is prepared to forego the interest-free use of hundreds of millions of dollars of its citizens' money, just to ensure that they are treated fairly.

Well, unpack your bags; the reason they did it was that it became fashionable for Governments to control (!) the economy via the money supply and, if you release hundreds of millions of dollars into the economy just when you are trying to hold things back, you no longer have control. To Governments, of course, control is far more important even than money, hence the money went.

All that is now left for Australia's taxpaying citizens then, is the annual tedium of completing their Income Tax Returns with little hope of reward. This program will reduce that tedium a little by assisting you to calculate just how much refund you should get or how much extra you will need to render unto Caesar. Overseas readers may find it interesting to compare their own rates of direct taxation with Australia's whilst those readers in the Middle East and other tax havens can reinforce the wisdom of their choice of domicile.

The following variables are used in the program.

LL = Lower Tax Limit
TD = Tax Dollar Payable on LL
TC = Tax Cents Payable on LL
GC = Gross Income from Group Certificate
OI = Other Income (apart from group certificate)
DD = Deductions
TI = Taxable Income
TP = Tax Paid
IT = Income Tax Payable

The Tax Scales themselves are contained in DATA Lines 41Ø-44Ø. These should be altered whenever the Tax Scales change. As an example, shown below are the Tax Scales in two financial years 1979/1980 and 1980/81 and the changes which would have been required in the DATA Lines.

Tax Scales 1979/1980

Taxable Income From $	To $	Tax on Taxable Income
1	3893	NIL
3893	16607	NIL plus 33.07 cents for each $1 above $3893
16608	33215	$4204.8505 plus 47.07 cents each $1 above $16608
33216 and over		$12022.326 plus 61.07 cents each $1 above 33216

Tax Scales 1980/1981

1	4041	NIL
4041	17238	NIL plus 32 cents for each $1 above $4041
17239	34477	$4223.36 plus 46 cents for each $1 above 17239
34478 and over		$12153.30 plus 60 cents for each $1 above 34478

DATA Lines

1979/1980

```
41Ø DATA 1, Ø, Ø
42Ø DATA 3893, Ø, Ø.3307
43Ø DATA 166Ø8, 42Ø4.85, Ø.47Ø7
44Ø DATA 33216, 12Ø22.236, Ø.61Ø7
```

DATA Lines

1980/1981

```
1, Ø, Ø
4Ø41, Ø, Ø.32
17239, 4223.36, Ø.46
34478, 12153.3Ø, Ø.6Ø
```

ANALYSIS OF THE PROGRAM

The following shows the functions carried out by different parts of the program:

Lines 8Ø-14Ø	Ask for information to be input
21Ø-23Ø	Read in the TAX SCALES from the DATA in lines 41Ø-44Ø
26Ø-28Ø	Search the TAX SCALES for the Lower Tax Limit (LL) that is greater than the Net Income (TI)
29Ø	Ensures that if Net Income is greater than last DATA line (44Ø) that this line is used
31Ø	Calculates the Tax Payable (IT) by going back one DATA line
32Ø	Prints Tax Payable
35Ø-37Ø	Print out Who Owes Who and how much
41Ø-44Ø	DATA lines which store the Tax Scales.

This program is correct for the 1980/81 Tax Year.

- 0000000000 -

```
                00010 ;                    MESAGE
                00020 ;
                00030 ; THIS ROUTINE FLASHES A MESSAGE INDEPENDENTLY OF PROGRAM
                00040 ;CONTROL. MEMORY SIZE IS SET AUTOMATICALLY AND PROGRAM
                00050 ;RUNS BY ITSELF IF ENTERED VIA AN EDITOR-ASSEMBLER.
                00060 ;  PROGRAM IS DISABLED BY 'KILL' COMMAND IN BASIC AND
                00070 ;ACTIVATED BY THE 'LOAD' COMMAND.
                00080 ;  THE RATE OF FLASHING MAY BE ADJUSTED BY FOLLOWING
                00090 ;BASIC PROGRAM OTHERWISE IT IS SET TO DEFAULT VALUE.
                00100 ;
3DC0            00110           ORG     3DC0H
3DC0 2A         00120           DEFM    '** **    FLASHING  MESSAGE  ROUTINE :   (C)
  JULY - 1981   ** **'
3E00 20         00130           DEFM    '       O   BY : ANTHONY PARK , MIDDLE COVE
N.S.W  2068.'
                00140 ;PROGRAM FOLLOWS.
                00150 ;
7EE5            00160           ORG     7EE5H
                00170 ;
                00180 ;DEFINE LABELS :
                00190 ;
06CC            00200 BASIC     EQU     06CCH
03E3            00210 KBSCAN    EQU     03E3H
4016            00220 KBVEC     EQU     4016H
40B1            00230 TOPMEM    EQU     40B1H
0250            00240 DELAY     EQU     0250H
40A0            00250 SPOINT    EQU     40A0H
1B6E            00260 SETPTR    EQU     1B6EH
0020            00270 SPACE     EQU     0020H
3C00            00280 VIDEO     EQU     3C00H
                00290 ;
                00300 ;INITIALIZE
                00310 ;
7EE5 F3         00320 INIT      DI
7EE6 210C7F     00330           LD      HL,START
7EE9 221640     00340           LD      (KBVEC),HL
7EEC 2B         00350           DEC     HL
7EED 2B         00360           DEC     HL
7EEE 22B140     00370           LD      (TOPMEM),HL     ;SET MEMORY SIZE
7EF1 113200     00380           LD      DE,32H
7EF4 B7         00390           OR      A
7EF5 ED52       00400           SBC     HL,DE
7EF7 22A040     00410           LD      (SPOINT),HL
                00420 ;
                00430 ;SET 'LOAD' AND 'KILL' COMMAND VECTORS.
                00440 ;
7EFA 21E57E     00450           LD      HL,INIT
7EFD 228941     00460           LD      (4189H),HL
7F00 21F67F     00470           LD      HL,KILL
7F03 229241     00480           LD      (4192H),HL
7F06 CD6E1B     00490           CALL    SETPTR          ;SET ALL OTHER POINTERS.
7F09 C3CC06     00500           JP      BASIC
                00510 ;
                00520 ;MAIN ROUTINE
                00530 ;
7F0C 3AF57F     00540 START     LD      A,(TEST)        ;TEST FOR PROGRAM RUNNING
7F0F B7         00550           OR      A
7F10 2004       00560           JR      NZ,CONT
7F12 D9         00570           EXX
7F13 CD2C7F     00580           CALL    PRINT           ;PRINT MESSAGE.
7F16 CDE303     00590 CONT      CALL    KBSCAN
7F19 D9         00600           EXX
7F1A B7         00610           OR      A
7F1B 2046       00620           JR      NZ,RESCNT
7F1D 2AF37F     00630           LD      HL,(COUNT)
7F20 7C         00640           LD      A,H
7F21 B5         00650           OR      L
7F22 3E00       00660           LD      A,0
7F24 2806       00670           JR      Z,PRINT
7F26 2B         00680           DEC     HL              ;DECREMENT COUNTER
7F27 22F37F     00690           LD      (COUNT),HL
7F2A D9         00700           EXX
7F2B C9         00710           RET
```

```
                00720 ;
                00730 ;THIS FLASHES MESSAGE
                00740 ;
7F2C 21003C     00750 PRINT   LD      HL,VIDEO        ;TOP LEFT OF SCREEN.
7F2F 7E         00760         LD      A,(HL)
7F30 FE20       00770         CP      SPACE           ;IS MESSAGE PRINTED ?
7F32 201B       00780         JR      NZ,MESOFF       ;YES ! TURN IT OFF !!
7F34 21717F     00790         LD      HL,MESAGE       ;NO ! THEN PRINT IT !!
7F37 DD21003C   00800         LD      IX,VIDEO
7F3B 32F57F     00810         LD      (TEST),A
7F3E 7E         00820 LOOP    LD      A,(HL)
7F3F FE00       00830         CP      0       ;CHECK FOR END OF MESSAGE.
7F41 2808       00840         JR      Z,ENDMES
7F43 DD7700     00850         LD      (IX),A
7F46 23         00860         INC     HL
7F47 DD23       00870         INC     IX
7F49 18F3       00880         JR      LOOP            ;KEEP ON PRINTING
7F4B AF         00890 ENDMES  XOR     A
7F4C F5         00900         PUSH    AF
7F4D 1819       00910         JR      FIXCNT          ;FIX COUNT AGAIN
7F4F 21B27F     00920 MESOFF  LD      HL,CLEAR        ;WIPE OUT MESSAGE
7F52 DD21003C   00930         LD      IX,VIDEO
7F56 7E         00940 LOOP1   LD      A,(HL)
7F57 FE00       00950         CP      0               ;SAME AS BEFORE.
7F59 28F0       00960         JR      Z,ENDMES
7F5B DD7700     00970         LD      (IX),A
7F5E 23         00980         INC     HL
7F5F DD23       00990         INC     IX
7F61 18F3       01000         JR      LOOP1
7F63 F5         01010 RESCNT  PUSH    AF
7F64 AF         01020         XOR     A
7F65 32F57F     01030         LD      (TEST),A
7F68 215002     01040 FIXCNT  LD      HL,DELAY        ;GET DELAY.
7F6B 22F37F     01050         LD      (COUNT),HL
7F6E F1         01060         POP     AF
7F6F D9         01070         EXX
7F70 C9         01080         RET
                01090 ;
                01100 ;CHOOSE YOUR OWN MESSAGE HERE.
                01110 ;N.B.  FIRST CHARACTER MUST   N O T   BE A SPACE !!!
                01120 ;
7F71 3D         01130 MESAGE  DEFM    '==>          SYSTEM 80   16 K / LII  COMPUTE
R SYSTEM.           <=='
7FB1 00         01140         DEFB    0       ;CONFIRM END OF MESSAGE.
7FB2 20         01150 CLEAR   DEFM    '
                      '       ;CLEAR WITH 64 SPACES.
7FF2 00         01160         DEFB    0       ;AS BEFORE.
7FF3 0000       01170 COUNT   DEFW    0
7FF5 00         01180 TEST    DEFB    0
                01190 ;
                01200 ;THIS SECTION RESETS KEYBOARD VECTOR.
                01210 ;
7FF6 21E303     01220 KILL    LD      HL,KBSCAN
7FF9 221640     01230         LD      (KBVEC),HL
7FFC C3CC06     01240         JP      BASIC
                01250 ;
                01260 ;THIS SECTION RUNS THE PROGRAM BY ITSELF
                01270 ;
4015            01280         ORG     4015H
4015 01E57E     01290         LD      BC,7EE5H
7EE5            01300         END     INIT            ;INIT IS A BACKUP ???
```

```
10 '** **  FLASH RATE DRIVER FOR MESSAGE ROUTINE  ** **
20 CLS
30 PRINT@448,"                  PRESENT FLASH RATE = ";PEEK(3261
7)+(PEEK(32618)*256)
40 PRINT:PRINT
50 INPUT"                  ENTER NEW FLASH RATE ";R
60 IFR>32767THENR=32767
70 IFR<0THENR=100
80 POKE32617,RAND255:POKE32618,(R/256)AND255
90 GOTO10
100 END
```

```
10 '     (C)       N. J. COLEMAN
20 '               6/13 HOWITT ST
30 '               SOUTH YARRA VIC  3141
40 '
50 '
60 '     ***       SCREEN COPY UTILITY    ***
70 '
80 '
90 '
100 '
110 CLS
120 POKE 16553,255    'FIX DATA BUG
130 FOR X = 31690 TO 31740
140 READ A
150 POKE X,A
160 NEXT X
170 ONERROR GOTO 175:DEFUSR=31690: GOTO 180
175 POKE 16526,202:POKE 16527,123
180 XX=USR(N)
190 DATA 33,225,123,62,195,50,151,65,34,152,65,33,239,123,50
200 DATA 154,65,34,155,65,195,204,6,229,33,0,60,17,253,123,1,0
210 DATA 4,237,176,225,201,229,33,253,123,17,0,60,1,0,4
220 DATA 237,176,225,201
230 END
```

Hex dump of LONGVAR

START = F400 END = FF14 ENTRY = F9A4

```
F400:  2A F9 40 11 64 00 19 22 02 F3 23 36 FF 22 08 F3
F410:  22 0A F3 21 76 F5 22 A3 F4 CD 89 F9 21 00 00 22
F420:  0C F3 E1 C1 31 00 F3 C5 E5 AF 67 6F 39 2B 01 64
F430:  00 54 5D 1B ED B8 AF 32 40 F3 21 7E B9 22 3C F7
F440:  22 5B F7 C9 2A A4 40 2B 3E 81 32 3F 3C 2B 11 4D
F450:  F4 D5 D7 DA 6C 0E F5 3A 46 F3 B7 CC E4 F4 C1 78
F460:  28 40 FE 2E CA 6C 0E FE 26 CA 94 41 FE A3 CA F4
F470:  FE FE 93 CA 07 1F FE 22 28 2B FE 88 CA 05 1F FE
F480:  D6 28 2E B7 28 41 23 FE 9A CA 06 1E FE 99 CA 03
F490:  1E FE 98 CA 00 1E FE 9B CA 09 1E 2B CD 3D 1E 23
F4A0:  D8 2B C3 76 F5 01 22 00 D7 B8 C8 23 B9 C8 2B 18
F4B0:  F7 23 7E FE CF C0 01 D4 CF 23 7E B7 C8 B8 20 F9
F4C0:  23 7E B9 20 F8 23 C9 23 22 04 F3 7E 23 B6 28 68
F4D0:  23 7E 32 06 F3 23 7E 32 07 F3 23 3A 3F 3C EE 03
F4E0:  32 3F 3C C9 ED 5B 02 F3 E5 21 00 00 22 0C F3 13
F4F0:  1A FE FF 28 42 ED 53 00 F3 06 00 E1 E5 1A B7 22
F500:  0E F3 28 34 04 2B D7 1A BE 13 23 28 F0 FE 26 30
F510:  21 4F 2B 2B D7 28 0F C5 CD D8 F5 C1 7E 20 13 C5
F520:  CD F6 F5 C1 28 0C 22 0E F3 E1 E5 CD 3A F5 05 B9
F530:  28 06 CD 62 F5 18 B9 B7 E1 C9 7E D6 41 21 01 41
F540:  85 6F 7E D5 11 59 F5 D5 16 25 D6 02 C8 15 3D C8
F550:  16 21 3D C8 16 23 D6 04 C9 7A D1 C9 1A B7 13 20
F560:  FB C9 CD 5C F5 1A 4F 13 1A 47 13 13 13 13 2A 0C
F570:  F3 09 22 0C F3 C9 CD E4 F4 CC CD F5 22 00 F3 20
F580:  13 CC 90 F5 C4 A8 F5 CD 0D F6 C9 3A 44 F3 3C C9
F590:  2A 0E F3 C9 CD 8B F5 20 E8 2B D7 CD D8 F5 20 FA
F5A0:  CD E4 F5 C9 7E 12 13 23 CD 36 F6 2B D7 CD D8 F5
F5B0:  20 F2 CD E4 F5 F5 CD BE F6 EB F1 77 23 AF 77 23
F5C0:  36 01 23 06 04 77 23 10 FC 3D 77 EB C9 13 EB 4E
F5D0:  23 46 03 70 2B 71 EB C9 38 07 CD 3D 1E 30 02 18
F5E0:  01 F6 AF C9 7E CD F6 F5 20 03 7E 23 C9 E5 2A 00
F5F0:  F3 CD 3A F5 E1 C9 D5 01 59 F5 C5 16 02 FE 25 C8
F600:  14 FE 24 C8 14 FE 21 C8 16 08 FE 23 C9 E5 D5 CD
F610:  A7 F6 D1 E1 D0 E5 D5 2A 08 F3 EB 2A 0C F3 29 19
F620:  23 EB 2A 0A F3 44 4D 03 03 CD B4 F6 2A 06 F3 EB
F630:  73 23 72 D1 E1 C9 E5 D5 CD 44 F6 D1 44 4D 03 CD
F640:  B4 F6 E1 C9 2A 0A F3 CD A0 F6 30 0D 23 22 0A F3
F650:  E5 2A 08 F3 CD 61 F6 E1 C9 2A 08 F3 CD A0 F6 30
F660:  05 23 22 08 F3 C9 21 6F F6 CD 75 2B C3 CC 06 50
F670:  52 4F 47 52 41 4D 20 54 4F 4F 20 4C 4F 4E 47 20
F680:  54 4F 4F 20 48 41 4E 44 4C 45 20 2D 20 50 52 4F
F690:  47 52 41 4D 20 55 4E 43 48 41 4E 47 45 44 0D 00
F6A0:  11 9C F2 E5 DF E1 C9 2A 0A F3 CD A0 F6 D0 23 23
F6B0:  22 0A F3 C9 DF C8 7E 02 DF C8 2B 0B 18 F8 E5 D5
F6C0:  06 06 CD 44 F6 10 FB D1 44 4D 03 03 03 03 03 C3
F6D0:  3E F6 21 00 00 22 0C F3 E5 2A 02 F3 23 7E FE FF
F6E0:  28 25 22 00 F3 EB CD 62 F5 EB 22 12 F3 C5 D1 E1
F6F0:  E5 DF 30 0E E1 D5 2A 0C F3 22 42 F3 2A 00 F3 22
F700:  10 F3 2A 12 F3 18 D6 E1 C9 21 14 F3 54 5D 13 01
F710:  28 00 36 00 ED B0 C9 CD D2 F6 7C B5 22 3E F3 C9
F720:  ED 5B 10 F3 CD 5C F5 13 12 1B 12 CD 8B F5 CA 50
F730:  FE 1B 1B D5 1A CD F6 F5 4F 21 01 41 7E B9 CC 7A
F740:  F7 CA C5 F7 2C 7D FE 1B 20 F2 2E 01 7E B9 CC 87
F750:  F7 28 79 2C 7D FE 1B 20 F3 2E 01 7E B9 C4 BB F7
F760:  20 71 2C 7D FE 1B 20 F3 2E 01 7E B9 C4 B1 F7 20
F770:  6A 2C 7D FE 1B 20 F3 C3 07 F8 11 14 F3 83 5F 1A
F780:  95 30 2C 7D 12 AF C9 11 1F F3 83 5F 1A 95 30 1F
F790:  13 13 13 1A 3C FE 30 38 FB FE 5B 28 0A FE 3A 20
F7A0:  02 3E 41 12 47 AF C9 3E 2F 12 1B 1B 1B 7D 12 B4
F7B0:  C9 11 2B F3 CD 8A F7 20 EC 18 F4 11 17 F3 CD 7D
F7C0:  F7 20 E2 18 EA 7D C6 40 4F AF 18 15 7D C6 40 4F
F7D0:  B7 18 0E 7D C6 40 4F AF 37 18 06 7D C6 40 4F B7
F7E0:  37 11 50 F3 79 12 E1 D5 28 03 78 13 12 38 06 3A
F7F0:  40 F3 B7 28 03 7E 13 12 AF 13 12 EB E1 01 03 00
F800:  13 13 13 13 ED B0 C9 21 10 F8 CD 75 2B C3 CC 06
F810:  52 55 4E 20 4F 55 54 20 4F 46 20 55 53 41 42 4C
F820:  45 20 56 41 52 49 41 42 4C 45 53 00 CD 09 F7 CD
F830:  17 F7 CA 36 FD CD 20 F7 CD 41 F8 CD 86 F8 C3 2F
F840:  F8 21 65 F8 CD 75 2B 2A 10 F3 CD 7E 2B 2A A7 40
F850:  CD 75 2B 3A 45 F3 B7 C0 21 70 F8 CD 75 2B 21 50
```

```
F860: F3 CD 75 2B C9 0D 56 41 52 49 41 42 4C 45 20 00
F870: 20 52 45 4E 41 4D 45 44 20 54 4F 20 00 20 49 4E
F880: 20 4C 49 4E 45 00 21 7D F8 CD 75 2B 2A 3E F3 E5
F890: 2A 42 F3 29 EB 2A 08 F3 19 18 1C E5 56 2B 5E 2B
F8A0: E5 CD C5 F8 EB CD AF 0F 2A 0A F3 23 EB C1 E1 C5
F8B0: CD BD F8 E1 D1 1B D5 7A B3 20 E0 D1 C9 23 03 7E
F8C0: 02 DF C8 18 F8 3A 9C 40 B7 20 15 CD 5F FE CD 48
F8D0: 03 E6 07 2F 3C C6 08 47 3E 20 CD 2A 03 10 FB C9
F8E0: 3A 9B 40 FE 78 FA D1 F8 3E 0D C3 2A 03 21 F6 F8
F8F0: 22 A3 F4 C3 44 F4 CD E4 F4 C2 99 F5 13 13 13 D5
F900: 01 FF 04 1A 0C 13 B7 28 02 10 F8 E5 06 00 C5 09
F910: ED 5B 0E F3 AF ED 52 CD 25 F9 C1 E1 D1 41 1A 77
F920: 13 23 10 FA C9 44 4D 78 B7 FA 72 F9 2A F9 40 09
F930: ED 5B 02 F3 DF 38 23 21 40 F9 CD 75 2B C3 CC 06
F940: 0D 45 52 52 4F 52 2E 20 50 52 4F 47 52 41 4D 20
F950: 44 45 53 54 52 4F 59 45 44 00 ED 5B F9 40 22 F9
F960: 40 ED 4B 0E F3 1A 77 7B B9 20 03 7A B8 C8 2B 1B
F970: 18 F3 2A 0E F3 54 5D 09 44 4D C3 E5 2B 05 00 0E
F980: 07 F3 10 DD E6 E5 E9 E4 00 3A 04 38 B7 C8 E5 D5
F990: C5 21 7D F9 C6 2C 57 1E 00 06 0C 1A 86 12 10 FB
F9A0: C1 D1 E1 C9 CD C9 01 21 0A FE CD 75 2B 21 AD F9
F9B0: E5 21 CB FA CD 75 2B CD 00 F4 CD 3F FE FE 43 28
F9C0: 11 FE 53 28 28 FE 58 28 31 FE 42 C0 CD FE 20 C3
F9D0: 7A 2C CD 86 FC CD 6F FE CD FB FC AF 32 44 F3 CD
F9E0: 44 F4 AF 32 45 F3 CD 2C F8 CD ED F8 C9 CD BA FC
F9F0: CD 6F FE CD 24 FD 3E FF 18 E2 CD 0F FA CD 6F FE
FA00: CD 24 FD CD 44 F4 3E FF 32 45 F3 CD 2C F8 C9 21
FA10: 2A FA CD 75 2B CD 3F FE D6 5A 32 44 F3 3C C2 86
FA20: FC 21 83 FA CD 75 2B C3 8C FC 0D 0D 50 52 45 53
FA30: 53 20 22 59 22 20 49 46 20 59 4F 55 20 57 49 53
FA40: 48 20 54 4F 20 43 52 4F 53 53 52 45 46 20 53 50
FA50: 45 43 49 46 49 45 44 20 56 41 52 49 41 42 4C 45
FA60: 53 2C 20 45 4C 53 45 20 20 20 20 20 20 20 50 52
FA70: 45 53 53 20 41 4E 59 20 4F 54 48 45 52 20 4B 45
FA80: 59 0D 00 0D 45 4E 54 45 52 20 56 41 52 49 41 42
FA90: 4C 45 53 20 59 4F 55 20 57 49 53 48 20 54 4F 20
FAA0: 52 45 46 45 52 45 4E 43 45 2E 20 54 48 45 20 54
FAB0: 59 50 45 20 42 59 54 45 20 4D 55 53 54 20 42 45
FAC0: 20 20 20 20 47 49 56 45 4E 0D 00 0D 42 2E 2E 2E
FAD0: 52 45 54 55 52 4E 20 54 4F 20 42 41 53 49 43 0D
FAE0: 43 2E 2E 2E 43 4F 4D 50 4C 45 54 45 20 52 45 4E
FAF0: 41 4D 45 20 4F 46 20 56 41 52 49 41 42 4C 45 53
FB00: 0D 53 2E 2E 2E 53 50 45 43 49 46 49 45 44 20 56
FB10: 41 52 49 41 42 4C 45 20 52 45 4E 41 4D 49 4E 47
FB20: 0D 58 2E 2E 2E 43 52 4F 53 53 20 52 45 46 45 52
FB30: 45 4E 43 45 20 4F 4E 4C 59 0D 00 0D 45 4E 54 45
FB40: 52 20 56 41 52 49 41 42 4C 45 20 4E 41 4D 45 53
FB50: 20 59 4F 55 20 57 49 53 48 20 54 4F 20 44 45 43
FB60: 4C 41 52 45 2C 20 49 2E 45 2E 20 41 4C 4C 20 4E
FB70: 41 4D 45 53 20 54 48 41 54 20 20 20 49 4E 43 4C
FB80: 55 44 45 20 52 45 53 45 52 56 45 44 20 57 4F 52
FB90: 44 53 2E 20 54 48 45 20 54 59 50 45 20 42 59 54
FBA0: 45 20 4D 55 53 54 20 42 45 20 49 4E 43 4C 55 44
FBB0: 45 44 2E 0D 00 50 52 45 53 53 20 45 4E 54 45 52
FBC0: 20 41 46 54 45 52 20 45 41 43 48 20 4F 4E 45 0D
FBD0: 45 4E 54 45 52 20 22 2A 22 20 54 4F 20 54 45 52
FBE0: 4D 49 4E 41 54 45 0D 00 0D 0D 45 4E 54 45 52 20
FBF0: 56 41 52 49 41 42 4C 45 20 4E 41 4D 45 53 20 54
FC00: 48 41 54 20 59 4F 55 20 57 49 53 48 20 54 4F 20
FC10: 41 4C 54 45 52 2C 20 55 53 49 4E 47 20 54 48 45
FC20: 20 46 4F 52 4D 20 20 20 20 20 20 20 56 54 3D 4E 28
FC30: 54 29 2C 20 57 48 45 52 45 20 56 28 2E 2E 2E 56
FC40: 29 20 49 53 20 54 48 45 20 4F 4C 44 20 4E 41 4D
FC50: 45 2C 20 54 20 49 53 20 54 48 45 20 54 59 50 45
FC60: 20 42 59 54 45 20 41 4E 44 20 4E 28 2E 2E 2E 4E
FC70: 29 20 49 53 20 54 48 45 20 4E 45 57 20 4E 41 4D
FC80: 45 2E 00 E1 18 34 21 3B FB CD 75 2B 21 B5 FB CD
FC90: 75 2B CD 59 FE D7 FE 2A C8 CD C0 1B 23 CD A2 FC
FCA0: 18 F0 AF 32 C1 F5 CD E4 F4 22 00 F3 CC 90 F5 C8
FCB0: C4 E7 FE 3E 01 32 C1 F5 B7 C9 21 E8 FB CD 75 2B
FCC0: 21 B5 FB CD 75 2B CD 59 FE D7 FE 2A C8 CD C0 1B
FCD0: E5 D7 28 AF FE D5 20 F9 36 00 E3 23 CD A2 FC E1
FCE0: C8 1B 1B 1B 23 CD ED FC C4 F8 FC 18 D9 06 03 7E
FCF0: 12 B7 C8 13 23 10 F8 AF 12 3D C9 21 B5 FD CD 75
FD00: 2B CD 3F FE FE 54 20 01 F6 AF 32 40 F3 21 78 FD
FD10: CD 75 2B CD 3F FE FE 53 28 0A 21 18 0E 22 3C F7
FD20: 25 22 5B F7 21 3C FD CD 75 2B CD 3F FE FE 56 C8
FD30: 3E 01 32 9C 40 C9 CD FE 20 C3 69 21 0D 0D 50 52
FD40: 45 53 53 20 22 56 22 20 54 4F 20 53 45 54 20 56
FD50: 49 44 45 4F 20 4F 55 54 50 55 54 2C 20 41 4E 59
FD60: 20 4F 54 48 45 52 20 4B 45 59 20 46 4F 52 20 50
FD70: 52 49 4E 54 45 52 0D 00 0D 50 52 45 53 53 20 22
FD80: 53 22 20 54 4F 20 41 4C 4C 4F 57 20 53 49 4E 47
FD90: 4C 45 20 4C 45 54 54 45 52 20 4E 41 4D 45 53 2C
FDA0: 20 45 4C 53 45 20 41 4E 59 20 4F 54 48 45 52 20
FDB0: 4B 45 59 0D 00 0D 50 52 45 53 53 20 22 54 22 20
FDC0: 54 4F 20 53 45 54 20 41 4C 4C 20 56 41 52 49 41
FDD0: 42 4C 45 53 20 57 49 54 48 20 54 59 50 45 20 44
FDE0: 45 43 4C 41 52 41 54 49 4F 4E 20 46 4C 41 47 0D
FDF0: 45 4C 53 45 20 50 52 45 53 53 20 41 4E 59 20 4F
FE00: 54 48 45 52 20 4B 45 59 0D 00 0D 0D 0D 0D 56 41
FE10: 52 49 41 42 4C 45 20 52 45 4E 41 4D 49 4E 47 20
FE20: 50 52 50 4F 47 52 41 4D 0D 0D 52 2E 54 2E 20 57
FE30: 4F 52 4C 45 59 2C 20 31 39 38 31 0D 0D 0D 00 3E
FE40: 3F CD 2A 03 3E 20 CD 2A 03 CD 49 00 CD 2A 03 C9
FE50: 13 13 21 50 F3 EB C3 ED FC 2A A7 40 C3 B3 1B CD
FE60: 5B 03 B7 C8 47 CD 5B 03 B7 28 FA B8 28 F7 C9 21
FE70: 7E FE CD 75 2B CD 3F FE D6 59 32 46 F3 C9 0D 50
FE80: 52 45 53 53 20 22 59 22 20 49 46 20 59 4F 55 20
FE90: 53 50 45 43 49 46 49 45 44 20 41 4E 59 20 56 41
FEA0: 52 49 41 42 4C 45 53 20 53 54 41 52 54 49 4E 47
FEB0: 20 57 49 54 48 20 20 20 20 20 20 20 20 20 20 52
FEC0: 45 53 45 52 56 45 44 20 57 4F 52 44 53 2C 20 45
FED0: 4C 53 45 20 41 4E 59 20 4F 54 48 45 52 20 4B 45
FEE0: 59 0D 00 7E 12 13 23 CD 36 F6 7E B7 20 F5 2B 1B
FEF0: 7E C3 B5 F5 D7 CD 6C 0E 2B D7 FE 2C C0 D7 CD 6C
FF00: 0E 2B D7 FE 41 C0 23 7E FE 53 C0 D7 CD 3D 1E D8
FF10: CD A2 F4 18 E3
```

```
10 '"THE MIND READER" (C) COPYRIGHT 1980 BY PETER J. SMITH
                        C/- P.O. BOX 145. PENRITH N.S.W. 2750
20 CLS:PRINTTAB(20)"T H E  M I N D  R E A D E R":PRINTTAB(20)STR
ING$(28,129)
30 PRINT:PRINTTAB(31)"B Y":PRINT:PRINTTAB(23)"P E T E R  S M I T
 H"
40 PRINT:PRINT"CHOOSE A LIST 1, 2, 3, 4, 5, 6"
50 PRINT:PRINT"THINK OF A NUMBER FROM 1 TO 63
                              - DO NOT TELL ME WHAT IT IS -":PRINT
"MEMORIZE THE NUMBER AND ANSWER THE QUESTIONS BY PRESSING
<Y> OR <N> !!"
60 PRINT:PRINT"NOW WHICH LIST WOULD YOU PREFER ?"
70 A$=INKEY$:IFA$=""THEN70   ELSEA=VAL(A$)
80 IF A<1ORA>6 GOTO70
90 CLS:Y=1:N=2
100 ON A GOTO 110  ,210  ,310  ,410  ,510  ,610
110 GOSUB 950  :GOSUB1560
120 PRINTTAB(23)"LIST NO. 1"
130 GOSUB 1550
140 B$=INKEY$:IFB$=""THEN140
150 IF B$="Y" THEN 180
160 U=0
170 GOTO 190
180 U=1
190 CLS
200 IF A=2 THEN 720
210 GOSUB 1050 :GOSUB1560
220 PRINTTAB(23)"LIST NO. 2"
230 GOSUB 1550
240 C$=INKEY$:IFC$=""THEN240
250 IF C$="Y" THEN 280
260 V=0
270 GOTO 290
280 V=2
290 CLS
300 IF A= 3 THEN 720
310 GOSUB 1150 :GOSUB1560
320 PRINTTAB(23)"LIST NO. 3"
330 GOSUB1550
340 D$=INKEY$:IFD$=""THEN340
350 IF D$="Y" THEN 380
360 W=0
370 GOTO 390
380 W=4
390 CLS
400 IF A=4 THEN 720
410 GOSUB 1250 :GOSUB1560
420 PRINTTAB(23)"LIST NO. 4"
430 GOSUB 1550
440 E$=INKEY$:IFE$=""THEN440
450 IF E$="Y" THEN 480
460 X=0
470 GOTO 490
480 X=8
490 CLS
500 IF A= 5 THEN 720
510 GOSUB 1350 :GOSUB 1560
520 PRINTTAB(23)"LIST NO. 5"
530 GOSUB 1550
540 F$=INKEY$:IFF$=""THEN540
550 IF F$= "Y" THEN 580
560 T=0
570 GOTO 590
580 T=16
590 CLS
600 IF A = 6 THEN 720
610 GOSUB 1450 :GOSUB 1560
620 PRINTTAB(23)"LIST NO. 6"
630 GOSUB 1550
640 H$=INKEY$:IFH$=""THEN640
650 IF H$="Y" THEN 680
660 Z=0
670 GOTO 690
680 Z=32
690 CLS
700 IF A = 1 THEN 720
710 IF A>1 THEN 110
720 PRINT@460,"LOOK AT ME CLOSELY WHILE I READ YOUR MIND!!!"
730 FOR Q= 1 TO 1000:NEXT:CLS
740 M=U+V+W+X+T+Z
750 PRINT@463, TAB(13)"I HAVE IT!!! THE NUMBER YOU SELECTED WAS
-:
760 PRINT:FORT=1TO1000:NEXT
770  PRINT  "                                  "M
780 PRINT
790 PRINT TAB(20)"TELL ME, WAS I CORRECT ?"
800 J$=INKEY$:IFJ$=""THEN800
810 IF J$="N" THEN 880
820 PRINT@0,TAB(19)"I TOLD YOU I CAN READ MINDS !!!"
830 PRINT
840 PRINT"WOULD YOU LIKE TO HAVE ANOTHER TRY ?";
850 K$=INKEY$:IFK$=""THEN850
860 IF K$="Y" THEN 20
870 END
880 CLS
890 PRINT@463,"WHAT!! IMPOSSIBLE. I DO  N-O-T MAKE MISTAKES.
900 PRINT
910 PRINT"YOU MADE AN ERROR. LET'S TRY AGAIN!!"
920 FOR X=1TO2000:NEXT:CLS
930 GOTO 20
940 END
950 PRINT@448,""
960 PRINT"1        3","5        7","9       11","13     15"
970 PRINT
980 PRINT"17      19","21      23","25      27","29     31"
990 PRINT
1000 PRINT"33      35","37      39","41      43","45     47"
1010 PRINT
1020 PRINT"49      51","53      55","57      59","61     63"
1030 RETURN
```

```
1040 END
1050 PRINT@448,""
1060 PRINT"2         3","6         7","10        11","14        15"
1070 PRINT
1080 PRINT"18       19","22        23","26        27","30        31"
1090 PRINT
1100 PRINT"34       35","38        39","42        43","46        47"
1110 PRINT
1120 PRINT"50       51","54        55","58        59","62        63"
1130 RETURN
1140 END
1150 PRINT@448,""
1160 PRINT"4         5","6         7","12        13","14        15"
1170 PRINT
1180 PRINT"20       21","22        23","28        29","30        31"
1190 PRINT
1200 PRINT"36       37","38        39","44        45","46        47"
1210 PRINT
1220 PRINT"52       53","54        55","60        61","62        63"
1230 RETURN
1240 END
1250 PRINT@448,""
1260 PRINT"8         9","10        11","12        13","14        15"
1270 PRINT
1280 PRINT"24       25","26        27","28        29","30        31"
1290 PRINT
1300 PRINT"40       41","42        43","44        45","46        47"
1310 PRINT
1320 PRINT"56       57","58        59","60        61","62        63"
1330 RETURN
1340 END
1350 PRINT@448,""
1360 PRINT"16       17","18        19","20        21","22        23"
1370 PRINT
1380 PRINT"24       25","26        27","28        29","30        31"
1390 PRINT
1400 PRINT"48       49","50        51","52        53","54        55"
1410 PRINT
1420 PRINT"56       57","58        59","60        61","62        63"
1430 RETURN
1440 END
1450 PRINT@448,""
1460 PRINT"32       33","34        35","36        37","38        39"
1470 PRINT
1480 PRINT"40       41","42        43","44        45","46        47"
1490 PRINT
1500 PRINT"48       49","50        51","52        53","54        55"
1510 PRINT
1520 PRINT"56       57","58        59","60        61","62        63"
1530 RETURN
1540 END
1550 PRINT@395,"IS THE NUMBER YOU SELECTED LISTED HERE ?";:RETUR
N
1560 PRINT@0,TAB(15)"T H E  M I N D  R E A D E R    ":PRINTTAB(1
5)STRING$(28,129):RETURN
```

```
10 REM * COPYRIGHT BY DAVID WEAVER
20 REM * 11 PINE GROVE SALE VICTORIA 3850  051 445019
30 REM * 03 APRIL 1981
40 REM * PROGRAM TO COMPUTE NET INCOME AND TAX PAYABLE
50 DIM LL(4),TD(4),TC(4)
60 A$="$$########.##
70 CLS
80 PRINT @15,"PERSONAL INCOME AND TAX PAYABLE";
90 REM * COMPUTES NET INCOME
100 PRINT:PRINT:INPUT"WHAT IS YOUR GROSS INCOME FROM GROUP CERTI
FICATE  ";GC
110 PRINT:INPUT"ANY OTHER INCOME ?
  ";OI
120 PRINT:PRINT "YOUR TOTAL INCOME IS
    ";USING A$ ;GC+OI
130 PRINT:INPUT"WHAT ARE YOUR TOTAL DEDUCTIONS
  ";DD
140 TI =(GC+OI)-DD
150 PRINT:PRINT"YOUR NET TAXABLE INCOME IS
   ";USING A$;TI
160 PRINT:INPUT"WHAT IS THE TAX PAID FROM YOUR GROUP CERTIFICATE
  ";TP
170 REM * PROGRAM TO FIND TAX PAYABLE ON NET INCOME
180 REM * LL STORES LOWER LIMIT OF TAX SCALE * TD STORES TAX
      DOLLAR ON LOWER LIMIT
190 REM * TC STORES CENTS IN THE DOLLAR ON NET INCOME OVER THE
    LOWER LIMIT
200 REM * INPUTS TAX SCALES
210 FOR J=1 TO 4
220 READ LL(J),TD(J),TC(J)
230 NEXT J
240 RESTORE
250 REM * SEARCH FOR FIRST VALUE IN LIST LL WHICH EXCEEDS TI
260 FOR J=1 TO 4
270 IF LL(J) >= TI THENGOTO310
280 NEXT J
290 LET J = 5
300 REM * PRINT OUT INCOME TAX PAYABLE ON NET INCOME TI
310  IT = TD(J-1)+TC(J-1)*(TI-LL(J-1))
320 PRINT:PRINT"INCOME TAX PAYABLE
   ";USING A$;IT
330 IF IT > TP THENGOTO370
340 REM * PRINTS OUT WHO OWES WHO
350 PRINT:PRINT "THE TAX DEPARTMENT < * OWES YOU * >
    ";USING A$;INT(((TP-IT)+.005)*100)/100
360 GOTO380
370 PRINT:PRINT"< * YOU OWE * > THE TAX DEPARTMENT
   ";USING A$;INT(((IT-TP)+.005)*100)/100
380 INPUT"DO YOU WANT TO CONTINUE   ";Z$
390 IF LEFT$(Z$,1)="Y" GOTO70 ELSECLS:END
400 REM * IF TAX SCALES CHANGE ALTER VALUES IN DATA LINES
410 DATA 1,0,0
420 DATA 4041,0,0.32
430 DATA 17239,4223.36,00.46
440 DATA 34478,12153.3,0.60
```

***** NEXT MONTH'S ISSUE *****

Next month's issue will contain at least the following programs plus the usual features and articles.

** LEMNISCATES LII/4K **

This program is for all those people who hate typing in long program listings. Lemniscates has one (1) line, yes that's all. The author wanted to see just how much could be done in one line of program. If you know what Lemniscates are, you will have an idea just how much this one line program does. If you don't know.....look it up before the next issue!!!

** WHEEL LOADER PRODUCTION LI/4K **

This program is designed to calculate wheel loader production in tons per hour. It can be used for all sizes of loaders in all types of conditions. If you are in the earthmoving or mining business, this should be very useful.

** RESTORE LII/16K **

On some personal computers, the EXIDY SORCERER to name one, it is possible to include in the program a statement such as RESTORE 300. This then sets the data pointer to look at line 300 when the program encounters a READ statement instead of having to read in all the data in the program - a great time saver. Next month you too will have this function on your computer.

** ALIEN INVASION LII/16K **

In this game you have to fight off the invading aliens. If you let five of them land the game moves on into a second section which is even harder, with lots of aliens all fighting back.

WANTED - ARTICLES AND PROGRAMS FOR THE TRS-80 COLOUR COMPUTER AND THE HITACHI PEACH.

Starting with Vol. 3 Issue No. 5, MICRO-80 will be supporting the TRS-80 Colour Computer and the Hitachi Peach. Both these computers have a 6809 micro-processor and use similar Microsoft BASIC interpreters. We require articles and programs which will be of interest to the users of these computers, for publication. One good source of program ideas is past issues of MICRO-80. Why not write programs similar to some of those we have published for the Models 1 and 3 with the added features of sound and colour. Naturally, we will pay a publication fee to each author. So, don't delay, use the coupon below to send in your contribution.

APPLICATION FOR PUBLICATION OF A PROGRAM IN MICRO-80

Date .....

Tick where appropriate

To **MICRO-80** SOFTWARE DEPT. P O BOX 145 MORPHETTVALE SA 5162

Please consider the enclosed program for . . .

(i) Publication in MICRO-80 .....

(ii) Publication on disk or cassette only .....

(iii) Both .....

Name .....

Address .....

.....

Postcode .....

*** CHECK LIST ***

Please ensure that the cassette or disk is clearly marked with your name and address, program name(s), Memory size, Level I, II, System 1 or 2, Edtasm, System, etc. The use of REM statements with your name and address is suggested, in case the program becomes separated from the accompanying literature.

Ensure that you supply adequate instructions, notes on what the program does and how it does it, etc.

For system tapes, the start, end, and entry points, etc.

The changes or improvements that you think may improve it.

Please package securely – padabags are suggested – and enclose stamps or postage if you want your cassette or disk returned.

***** CASSETTE/DISK EDITION INDEX *****

The cassette edition of MICRO-80 contains all the software listed each month, on cassette. All cassette subscribers need do is CLOAD and RUN the programs. Level II programs are recorded on side 1 of the cassette. Level I programs are recorded on side 2. Level I programs are not compatible with the System 80. All programs are recorded twice in succession. Note, System 80 computers have had different tape-counters fitted at different times. The approximate start positions shown are correct for the very early System 80 without the volume control or level meter. They are probably incorrect for later machines. The rates for a cassette subscription are printed on the inside front cover of each issue of the magazine.

The disk edition contains all those programs which can be executed from disk, including Level I programs. Level I disk programs are saved in the NEWDOS format. Users require the Level I/CMD utility supplied with NEWDOS + or NEWDOS 80 version 1.0 to run them.

	TYPE	I.D.	DISK FILESPEC	APPROX. START POSITION CTR-41	CTR-80	SYSTEM-80
SIDE 1						
LONGVAR	SYSTEM	LONGVR	LONGVAR/CMD	15	10	11
"	"	"		46	31	33
LONGVAR	EDTASM	LONGVR	LONGVAR/EDT	74	50	52
FLASHING MESSAGE	SYSTEM	MESAGE	MESSAGE/CMD	240	162	170
"	"	"		246	166	174
FLASHING MESSAGE	EDTASM	MESAGE	MESSAGE/EDT	252	170	178
"	"	"		275	186	195
FLASHING MESSAGE	L2/16K	M	MESSAGE/BAS	300	202	211
"	"	"		305	206	216
SIDE 2						
INCOME TAX	L2/4K	T	TAX/BAS	15	10	11
"	"	"		36	24	25
MIND READER	L2/4K	M	MINDREAD/BAS	55	37	39
"	"	"		92	62	65
VDU SAVE	L2/16K	V	VDUSAVE/BAS	126	85	89
"	"	"		133	90	94
LONGVAR	EDTASM	LONGVAR	LONGVAR/EDT	140	95	100

A SPECIAL OFFER!

TO SUBSCRIBERS OF MICRO 80

FOR THE FIRST TIME IN AUSTRALIA!
GENUINE TANDY TRS–80 MICRO-COMPUTERS, HARDWARE AND SOFTWARE AT

GREAT DISCOUNT PRICES!!!

HOW TO ORDER –

1. SELECT ITEMS FROM 1980 TANDY CATALOGUE
2. DEDUCT 10% FROM ADVERTISED PRICES
3. POST US YOUR ORDER STATING DESCRIPTION CAT. No AND A CHEQUE OR MONEY ORDER.

WE WILL –

1. ATTEND TO YOUR ORDER WITHIN 7 DAYS *
2. SUPPLY GOODS SELECTED FREIGHT FREE !
3. SEND ADVERTISING REGULARLY TO KEEP YOU INFORMED OF CURRENT SPECIALS !

* *subject to availability*

to **CONQUEST ELECTRONICS** Pty. Ltd.
212 Katoomba St. KATOOMBA 2780

Please supply –

QTY.	CAT NO	DESC.	ADV. PRICE

SUB TOTAL _____
LESS 10% _____
FIND CHEQUE FOR **TOTAL** _____

SEND **FREIGHT FREE** TO

NAME .....

ADDRESS .....

.....P/ Code .....

TANDY ELECTRONICS DEALER

CONQUEST ELECTRONICS PTY. LTD.

212 KATOOMBA ST KATOOMBA N.S.W. 2780 PHONE (047) 82 2491

MICRO-80

LEVEL 2 ROM
ASSEMBLY LANGUAGE TOOLKIT
by Edwin Paay
FOR TRS-80 MODEL 1, MODEL 3
AND SYSTEM 80/VIDEO GENIE

This is a new package consisting of two invaluable components:

- **A ROM REFERENCE** Manual which catalogues, describes and cross-references the useful and usable ROM routines which you can incorporate into your own machine language or BASIC programs.

- **DBUG**, a machine language disassembling debugging program to speed up the development of your own machine language programs. DBUG is distributed on a cassette and may used from disk or cassette.

Part 1 of the ROM REFERENCE manual gives detailed explanations of the processes used for arithmetical calculations, logical operations, data movements etc. It also describes the various formats used for BASIC, System and Editor/Assembly tapes. There is a special section devoted to those additional routines in the TRS-80 Model 3 ROM. This is the first time this information has been made available, anywhere. Differences between the System 80/Video Genie are also described. Part 1 is organised into subject specific tables so that you can quickly locate all the routines to carry out a given function and then choose the one which meets your requirements.

Part 2 gives detailed information about each of the routines in the order in which they appear in the ROM. It describes their functions, explains how to use them in your own machine language programs and notes the effect of each on the various Z80 registers.

Part 2 also details the contents of system RAM and shows you how to intercept BASIC routines. With this knowledge, you can add your own commands to BASIC, for instance, or position BASIC programs in high memory — the only restriction is your own imagination!

The Appendices contain sample programmes which show you how you can use the ROM routines to speed up your machine language programs and reduce the amount of code you need to write.

DBUG: Eddy Paay was not satisfied with any of the commercially available debugging programs, so he developed his own. DBUG: allows you to single-step through your program; has a disassembler which disassembles the next instruction before executing it or allows you to bypass execution and pass on through the program, disassembling as you go; displays/edits memory in Hex or ASCII; allows Register editing; has the ability to read and write System tapes and all this on the bottom 3 lines of your screen, thus freeing the rest of the screen for program displays. Four versions of DBUG are included in the package to cope with different memory sizes.

The best news of all is the price. The complete Level 2 ROM ASSEMBLY LANGUAGE TOOLKIT is only:

— Aus. $29.95 + $2.00 p&p
— UK £18.00 + £1.00 p&p

SPECIAL OFFER TO OWNERS OF THE LEVEL II ROM REFERENCE MANUAL ...
UPGRADE TO THIS ASSEMBLY LANGUAGE TOOKIT FOR ONLY $19.95!
Send back your original Level II ROM Reference Manual plus a cheque, money order or Bankcard authorisation for $19.95 plus $2.00 p&p and we will send you the new ASSEMBLY LANGUAGE TOOLKIT

MICRO-80